

Internet of Things Hardware Platform Security Advisor: A Framework for Facilitating Secure Design and Development of IoT Systems

Musa Gwani Samaila

Tese para obtenção do Grau de Doutor em

Engenharia Informática

(3^o ciclo de estudos)

Orientador: Prof. Doutor Pedro Ricardo Morais Inácio

Júri:

Prof. Doutor Mário Marques Freire

Prof. Doutor Miguel Nuno Dias Alves Pupo Correia

Prof. Doutor Paulo Alexandre Ferreira Simões

Prof. Doutor João Paulo da Silva Machado Garcia Vilela

Prof. Doutor Pedro Ricardo Morais Inácio

Prof. Doutor Virginia Nunes Leal Franqueira

Prof. Doutor Tiago Miguel Carrola Simões

abril de 2021

Dedication

I dedicate this thesis to the most High Yahuah, the Aluah of Yahshral, my source of inspiration, wisdom, knowledge, and understanding, and to His son Yahusha Ha'Mashiach. I also dedicate this work to my lovely family for their patience, endless support, and encouragement.

Acknowledgements

First and foremost, I give all the praise to Yahuah who in His foreknowledge and predestination has planned and provided all that was needed to start and complete this Ph.D. program. He took care of everything that would have stopped me along the way and strengthened me even through my most difficult times, and continues to make the impossible possible. May His name be forever praised: HalleluYah!

There are several people without whose help, support, and encouragement, this research work might not have been possible, and to whom I am greatly indebted. First of all, my deepest gratitude goes to my supervisor, Professor Dr. Pedro R. M. Inácio, for his support, direction, and guidance through all these years. Even though I came from a different background he has been patient with me and guided me on this journey of exploration that I knew would be immensely challenging. I am especially grateful for the deft ways in which he kindly challenged and supported me throughout the whole of this research work, knowing when to push and when to let up. I remain indebted to him for his unwavering support, corrections, encouragement, and patience through this process. I also thank him for the opportunity given me to contribute my quota to the **SECUR IoT ESIGN** Project. I would also like to express my appreciation to Professor Mário M. Freire for his concern, advice, and encouragement. I also appreciate every professor in the University of Beira Interior that has, in one way or the other, imparted knowledge, advised, or encouraged me along the way.

I would also like to thank both my former and current colleges at the Network and Multimedia Computing Group (NMCG) Laboratory and the **SECUR IoT ESIGN** Project team members who in one way or the other have contributed towards the successful completion of this study. I thank, in particular, Carolina Lopes (former undergraduate student) and Édi Aires (former masters student) who worked with me and partially developed the SRE and SBPG components, respectively. My appreciation also goes to Miguel Neto, Diogo A. B. Fernandes, Francisco Vigário, Manoel Campos da Silva Filho, Vinícius Rios, Vanice Cunha, Acácio F. P. P. Correia, Francisco Chimuco, João B. F. Sequeiros, and Tiago Simões. I would like to express my special thanks again to João B. F. Sequeiros for his friendly help and unfailing support. I will never forget how he had gone the extra mile to help me in a myriad of ways, and often takes my problems (such as language, my children school, and other bureaucratic issues) as his problems and helps me in overcoming them. Thanks a lot! Unfortunately, I cannot thank everyone by name and mention the important roles each of you has played in this journey, but I just want you all to know that you count so much and that I gratefully acknowledge your contributions, ideas, suggestions, support, and encouragement while working in the Laboratory and/or during several years of joint research activities.

I am especially grateful to all the computer and electronic engineers, as well as the developers that volunteered to participate in the several tests conducted on the Internet of Things Hardware Platform Security Advisor framework tool. Had it not been for your participation and helpful suggestions, I would never have completed this thesis. Thank you so much! I would have liked to name and individually thank each of you, but to preserve your privacy I will not do that. I sincerely appreciate your time and, in some cases, your effort in sharing your experience and expertise with me.

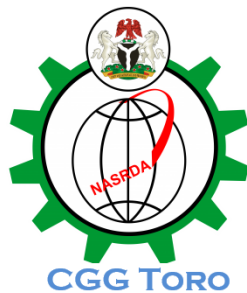
I am most grateful to the Director, Centre for Geodesy and Geodynamics (CGG), National Space Research and Development Agency (NASRDA), Toro, Nigeria, Dr. Tahir Abubakar Yakubu for his understanding, support, encouragement, and patience during this study. I would also like to especially thank the Head of Department (HOD), Information System Management, CGG, Dr. Mahmud U. Muhammad (who happens to be my HOD) for recommending the University of Beira Interior Covilhã to me; because without him, I would not have come to this university in the first place. Thank you very much for your recommendation, support, and encouragement. I also appreciate my former HOD, Dr. Joseph Dodo, for the role he had played for me to be here. I also thank my other colleges at CGG for their goodwill and encouragement. Thank you so much!

I gratefully acknowledge the funding entities or sources that made this Ph.D. work possible. It is with immense gratitude I acknowledge the support of the Centre for Geodesy and Geodynamics, National Space Research and Development Agency, Toro, Bauchi State, Nigeria, during this Ph.D. work. The work was also funded by the FCT research grant with reference BIM/n°32/2018-B00582. This research work was performed under the **SECUR IoT ESIGN** Project (reference POCI-01-0145-FEDER030657) with funds from FCT/COMPETE/FEDER.

Lastly, but most certainly not the least, my utmost gratitude goes to my family. Firstly, to my dear wife, Vashira, for her patience and unwavering belief that I can make it. I wonder how difficult it would have been if I had gone through this academic journey without her support, help, and encouragement. Most of the time she takes care of the children alone while I work in the laboratory, and sometimes return late in the night. She has actively supported me in myriad ways throughout these years and has always been a source of encouragement and inspiration to me. Without her, this thesis would not have been possible. Thank you so much for your faithful love and endless help. To my dear children, Lizzy (i.e., Elizabeth), David, 'My Bebê' (i.e., Esther), and Sara, for their endurance, understanding, and love, who even at such tender ages have had to endure so much stress and discomfort just for me. You have been so supportive even when being 'without Dad' was hard. I know you may not understand now, but I would like you to know that this work is for, and because of you and all the generations to come. I would also like to appreciate my dear brother, Professor Nuhu K. Samaila and my dear sisters: Astira, Hajo, Rahila,

Maryamu, and Saratu and their families, the family of my late brother, Malam Yusufu Samaila, as well as my dear father-in-law, Baba Titus Ayuba Zuya, my dear sister-in-law and dear brothers-in-law, Rambas, and Mafidin and their families, respectively, for their prayers, support, and encouragement. I particularly thank my dear sister, Saratu Samaila, again for her persistent help and untiring sacrifice during these years. To all of you, I say "Muito Obrigado!".

The work described in this thesis was carried out at the Instituto de Telecomunicações, Network and Multimedia Computing Group (NMCG) Laboratory, University of Beira Interior, Covilhã, Portugal. This research work was supported in part by the Centre for Geodesy and Geodynamics (CGG), National Space Research and Development Agency (NASRDA), Toro, Bauchi State, Nigeria. The research work was also funded in part by the **SECUR I o T E S I G N** Project through FCT/COMPETE/FEDER under Grant POCI-01-0145-FEDER-030657, in part by the FCT/MCTES through national funds and, when applicable, co-funded EU funds under the project UIDB/50008/2020, and in part by the Fundação para a Ciência e Tecnologia (FCT) research grant with reference BIM/nº32/2018-B00582.



Resumo

O termo Internet das coisas (IoT) é utilizado para descrever um ecossistema, em expansão, de objetos físicos ou elementos interconetados entre si e à Internet. Os dispositivos IoT consistem numa gama vasta e heterogénea de objetos animados ou inanimados e, neste contexto, podem pertencer à IoT um indivíduo com um implante que monitoriza a frequência cardíaca ou até mesmo um animal de estimação que tenha um biochip. Estes dispositivos variam entre eletrodomésticos, tais como máquinas de café ou lâmpadas inteligentes, a ferramentas sofisticadas de uso na automatização industrial. A IoT está a revolucionar e a provocar mudanças em várias indústrias e muitas adotam esta tecnologia para incrementar as suas vantagens competitivas. Este paradigma melhora a eficiência operacional e otimiza o desempenho de sistemas através da gestão de dados em tempo real, resultando num balanço otimizado entre o uso energético e a taxa de transferência. Outra área de aplicação é a IoT Industrial (IIoT) ou internet industrial ou Indústria 4.0, ou seja, uma aplicação de IoT no âmbito industrial, onde os sistemas ciberfísicos estão interconectados a diversas tecnologias de forma a obter um controlo de rede sem fios, bem como fabricações avançadas e automatização fabril. As aplicações da IoT estão a crescer e a tornarem-se predominantes em muitos domínios de aplicação inteligentes como sistemas de saúde, cidades, redes, agricultura e sistemas de fornecimento. Da mesma forma, a IoT está a transformar estilos de vida e de trabalho e assim, a procura por produtos inteligentes está constantemente a aumentar. As grandes indústrias e *startups* competem entre si de forma a dominar o mercado com os seus novos serviços e produtos IoT, desbloqueando o valor de negócio da IoT.

Apesar da sua crescente popularidade, benefícios e capacidades comprovadas, a IoT está ainda a dar os seus primeiros passos e é confrontada com muitos desafios. Entre eles, problemas de conectividade, compatibilidade/interoperabilidade entre dispositivos e sistemas, falta de padronização, gestão das enormes quantidades de dados e ainda falta de ferramentas para investigações forenses. No entanto, preocupações quanto ao estado de segurança e privacidade ainda estão entre os fatores adversos à adesão universal desta tecnologia. Estudos recentes revelaram que existem questões de segurança e privacidade associadas ao design e implementação de vários dispositivos IoT e aplicações inteligentes (smart apps.), isto pode ser devido ao facto, em parte, de que alguns fabricantes e empresas de desenvolvimento de dispositivos (especialmente *startups*) IoT e smart apps., recolham o valor de negócio dos grandes mercados IoT, negligenciando assim a importância da segurança, resultando em dispositivos IoT e smart apps. com carências e violações de segurança da IoT nos últimos anos.

Esta tese aborda os desafios de segurança e privacidade que foram supra mencionados. Visto que a Internet e os sistemas informáticos tradicionais são por vezes considerados inseguros, os sistemas IoT tornam-se ainda mais inseguros, devido a restrições inerentes a

tais dispositivos. Estas restrições são impostas devido ao custo, uma vez que se espera que muitos dispositivos de ponta sejam de baixo custo e descartáveis, com recursos energéticos limitados, bem como limitações na capacidade de armazenamento e computacionais, e redes com perdas devido a um desempenho de hardware de qualidade inferior, quando comparados com computadores convencionais. Uma das raízes do problema é o facto de que muitos fabricantes, *startups* e empresas de desenvolvimento destes dispositivos e smart apps não adiram ao conceito de *segurança por construção*, ou seja, logo na conceção, não preveem a proteção da privacidade e segurança. Assim, alguns dos produtos e dispositivos produzidos apresentam vulnerabilidades na segurança.

Nos últimos anos, hackers maliciosos têm explorado diferentes vulnerabilidades de segurança nas infraestruturas da IoT, causando violações de dados e outros incidentes de privacidade envolvendo dispositivos IoT e smart apps. Estes têm atraído uma atenção significativa por parte das comunidades académica e industrial, que culminaram num grande número de propostas apresentadas por investigadores científicos. Ainda que as abordagens de pesquisa e os resultados variem entre os diferentes estudos, há um consenso e pré-requisito fundamental para enfrentar os desafios de privacidade e segurança da IoT, que buscam construir proteção de segurança e privacidade em dispositivos IoT e smart apps. desde o fabrico. Para esta finalidade, esta tese investiga como produzir segurança e privacidade destes sistemas desde a produção, e como principal objetivo, concentra-se em fornecer soluções que possam promover a conceção e o desenvolvimento de dispositivos IoT e smart apps., nomeadamente um conjunto de ferramentas chamado Consultor de Segurança da Plataforma de Hardware da IoT (IoT-HarPSecA). Espera-se que o conjunto de ferramentas forneça apoio a designers e programadores em startups durante a conceção e implementação destes sistemas ou que facilite a integração de mecanismos de segurança nos sistemas pré-existentes.

De modo a alcançar o objetivo proposto, recorre-se à seguinte abordagem. A primeira fase consiste num levantamento exaustivo de diferentes aspetos da segurança e privacidade na IoT, incluindo requisitos de segurança na arquitetura da IoT e ameaças à sua segurança, os seus domínios de aplicação e os ativos cibernéticos associados, a complexidade das vulnerabilidades da IoT e ainda possíveis contramedidas relacionadas com a segurança e privacidade. Evolui-se para uma breve visão geral das plataformas de desenvolvimento de hardware da IoT. As fases seguintes consistem na identificação dos desafios e questões associadas à IoT, que foram restringidos às questões de segurança e privacidade. As demais etapas abordam o processo de pensamento de conceção (*design thinking*), design e implementação e, finalmente, a avaliação do desempenho.

O IoT-HarPSecA é composto por três componentes principais: a Obtenção de Requisitos de Segurança (SRE), Orientações de Melhores Práticas de Segurança (SBPG) e a recomendação de Componentes de Algoritmos Criptográficos Leves (LWCAR) na implementação

de software e hardware. O autor implementou uma ferramenta em linha de comandos usando linguagem C++ que serve como interface entre os utilizadores e a IoT-HarPSecA. Esta tese apresenta ainda uma descrição detalhada, desenho e implementação das componentes SRE, SBPG, e LWCAR. Apresenta ainda cenários práticos do mundo real que demonstram como o IoT-HarPSecA pode ser utilizado para elicitar requisitos de segurança, gerar boas práticas de segurança (em termos de recomendações de implementação) e recomendar algoritmos criptográficos leves apropriados com base no contributo dos utilizadores. De igual forma, apresenta-se a avaliação do desempenho destes três componentes, demonstrando que o IoT-HarPSecA pode servir como um roteiro para o desenvolvimento seguro da IoT.

Palavras-chave

Algoritmos criptográficos, algoritmos criptográficos leves, conselheiro de segurança para plataformas de hardware, criptografia, criptografia leve, domínio de aplicação, elicitação de requisitos de segurança, framework de segurança, implementações de hardware de algoritmos criptográficos leves, implementações de software de algoritmos criptográficos leves, internet das coisas, IoT, IoT-HarPSecA, melhores diretrizes de práticas seguras, melhores práticas seguras, modelo de ameaça, modelo de sistema, privacidade, recomendação de algoritmos criptográficos leves, requisitos de segurança, segurança, segurança por construção.

Resumo Alargado

Introdução

Foco e âmbito da Tese

O termo Internet das Coisas (IoT), refere-se a uma rede que compreende uma variedade de objetos físicos ou coisas com endereços de *Internet Protocol* (IP), ou outras tecnologias de comunicação, que lhes permitem ligarem-se umas às outras, bem como serem monitorizadas e/ou controladas através da Internet. Os sensores são em muitos casos cruciais para o funcionamento dos sistemas IoT [1]. Portanto, as coisas também podem ter incorporados sensores que lhes permitem detetar quer parâmetros físicos, quer químicos do seu ambiente circundante. As entradas dos sensores incorporados podem prover de uma variedade de fontes, tais como temperatura, pressão, luz, movimento, potencial hidrogénico (pH) (isto é, concentração de iões de hidrogénio) e gases perigosos. Estes sensores produzem informação valiosa que pode ser partilhada com outros elementos ligados. Adicionalmente, os dados dos sensores podem ser enviados para sistemas de gestão de forma a serem analisados para detetar tendências que possam fornecer a informação necessária de modo a que se tomem decisões informadas. Assim, um dispositivo ou uma coisa IoT pode ser qualquer coisa incorporada com um sensor ou um atuador que se ligue a uma rede, partilhe dados com outros dispositivos de uma rede e transmita dados através da Internet.

Essencialmente, a IoT está a ampliar a conectividade à Internet para além dos dispositivos tradicionais, tais como computadores, *tablets* e *smartphones*, para uma gama diversificada de produtos de uso diário, como frigoríficos e câmaras de segurança, mudando assim o mundo tal como o conhecemos. Assim, a previsão feita por Kevin Ashton, há cerca de 20 anos atrás, de que "a IoT tem o potencial de mudar o mundo" [2, 3], e que outrora parecia ser ficção científica, está a tornar-se cada vez mais uma realidade. Atualmente, a tecnologia IoT já está a mudar o modo de viver e trabalhar das pessoas, através da melhoria da produtividade e eficiência nas indústrias e organizações, assim como a proporcionar um nível mais elevado de conforto aos seus utilizadores individuais. Algumas áreas onde estas mudanças já se tornam visíveis incluem os dispositivos ubíquos centrados e usados por humanos (vulgo *wearables*), cuidados de saúde, indústria, transportes, áreas urbanas, automóveis conectados, eletrodomésticos, e uma miríade de outros exemplos [4].

O número de dispositivos ligados à Internet está a crescer a um ritmo acelerado [5], de tal forma que a IoT está a tornar-se progressivamente no centro da conectividade que permite a permuta de dados que vão desde coisas a pessoas, e processos. Consequentemente, muitas pessoas e empresas estão a adotar o paradigma, devido aos seus vastos potenciais benefícios e impacto nas suas vidas e negócios, respetivamente [6]. Por exemplo, muitos

dos líderes da indústria tais como Google, Apple, Microsoft, e Samsung lançaram as suas próprias plataformas IoT que já funcionam em pleno [7]. Inúmeras são as aplicações da IoT; incluem-se a monitorização ambiental inteligente, agricultura inteligente, e rede inteligente. Novos domínios de aplicação onde a IoT está a criar fontes de rendimento, novos modelos de negócio, e uma panóplia de oportunidades de negócio abrangem o retalho, logística e cadeia de fornecimento, monitorização do fluxo de produtos, e gestão de inventário. Paralelamente, a IoT está estreitamente interligada com os sistemas ciberfísicos (CPS), e assim torna-se um elemento-chave na Indústria 4.0, igualmente conhecida como a quarta revolução industrial [8].

Uma outra e nova área de aplicação da IoT que está atualmente a revolucionar o espaço industrial é a Internet Industrial das Coisas (IIoT), que permite a conectividade da Internet às indústrias transformadoras. Para além de uma maior eficiência e produtividade, os benefícios da IIoT comportam uma redução considerável nos custos e nos prejuízos para as empresas. Um caso promissor de utilização do IIoT é a manutenção preditiva [9], que permite às empresas identificar potenciais falhas e evitar dispendiosos períodos de inatividade. Ao integrar na IIoT mecanismos de Inteligência Artificial (AI) como a aprendizagem automática, problemas mais complexos nas indústrias podem vir a ser ultrapassados [10], nomeadamente a redução das emissões de carbono e outros problemas de poluição, tais como derrames de petróleo da indústria petrolífera.

Motivação

Diversas previsões de desenvolvimento, estatísticas e estimativas têm sido feitas por várias fontes de mercado e por diferentes analistas da indústria, indicando um futuro promissor para a IoT [11, 12]. Embora as estatísticas e os números possam variar entre diferentes estudos, é consensual que esta tecnologia tem um potencial significativo no setor das Tecnologias de Informação (IT) e que as suas perspetivas futuras são bastante promissoras. Independentemente disto e das tendências, a IoT tem de superar muitos desafios para demonstrar a sua maior valia. Neste sentido, apesar dos seus potenciais benefícios, a IoT enfrenta muitos desafios que limitam a sua generalização.

Análises ao panorama atual da IoT revelam uma série de desafios e questões que esta tecnologia enfrenta [13]. Mais ainda, enquanto se espera que novos *standards* de ligações sem fios (i.e., nova rede móvel de quinta geração (5G) e a nova *Wireless Fidelity* (Wi-Fi) 6 (i.e., IEEE 802.11ax)) respondam a algumas das questões tal como a melhoria da conectividade e promoção da implementação, também elas apresentam as suas limitações. O lançamento da rede 5G, por exemplo, acarreta desafios adicionais, incluindo questões de atribuição de bandas de frequência, custos de implementação, e questões a nível físico [14, 15]. Embora os desafios da IoT sejam sobretudo de natureza tecnológica e empresarial, a aceitação social é outro requisito importante para a disseminação comercial generalizada desta tecnologia [16]. Seguem-se alguns dos principais desafios tecnológicos

que a IoT tem de suplantar para se tornar uma realidade e alcançar a aceitação universal:

- **Heterogeneidade:** No âmbito da IoT, esperam-se trocas contínuas de informações e dados, sem qualquer intervenção humana. No entanto, a IoT é atualmente caracterizada por uma grande heterogeneidade no que respeita às redes e dispositivos que participam na troca de dados, resultando em diferentes aptidões do ponto de vista da comunicação e da computação. Diversos dispositivos conectados e Aplicações Inteligentes (*smart apps*) apresentam exigências e obstáculos contraditórios que têm de ser resolvidos para que os objetivos da IoT possam-se concretizar integralmente. A incompatibilidade entre as camadas físicas (PHY) do *Bluetooth Low Energy* (BLE) e Wi-Fi é um exemplo que ilustra como a gestão de um elevado nível de heterogeneidade representa um grande desafio no funcionamento dos ecossistemas de IOT [17];
- **Interoperabilidade:** Para garantir uma conectividade ininterrupta na IoT, os dispositivos necessitam de comunicar através de protocolos de comunicação comuns, tornando a interoperabilidade um requisito fundamental. Não obstante, no meio da IoT, os dispositivos têm diferentes representações de dados, protocolos proprietários, e Interfaces de Programação de Aplicações (APIs) heterogêneas, que representam um desafio à interoperabilidade entre dispositivos e *smart apps* [18];
- **Escalabilidade:** A adaptação às mudanças no meio envolvente, tais como a capacidade de apoiar a expansão à medida que a necessidade surge, é uma característica importante para um sistema IoT. Embora esta seja uma característica desejável, a escalabilidade pode colocar alguns obstáculos como problemas de conectividade à implementação e expansão em larga escala destes sistemas. Por exemplo, fornecer conectividade de ligação ascendente a um grande número de dispositivos conectados pode criar problemas para algumas redes móveis [19];
- **Recursos energéticos limitados:** A alimentação energética dos dispositivos de borda (do inglês *edge devices*) da IoT, nomeadamente sensores, atuadores, receptores do Sistema de Posicionamento Global (GPS), e câmaras representam um problema significativo para muitas implementações da IoT [20]. Isto torna-se particularmente importante quando se considera que tais dispositivos podem chegar aos milhares de milhões. Além disso, alguns destes dispositivos podem ser utilizados em áreas bastante remotas, onde a aquisição de fontes de energia pode ser um grande desafio. O que complica ainda mais a questão é que alguns dispositivos podem estar integrados em infraestruturas de betão ou subaquáticas, tornando a substituição de baterias extremamente difícil, ou inviável;
- **Segurança dos Dados:** Atualmente são produzidas enormes quantidades de dados que aumentam a um ritmo cada vez maior, na era *smart* (inteligente) da conectividade, dispositivos, e *smart apps* no âmbito empresarial e do consumidor. Visto que estes e a IoT estão intrinsecamente ligados, e o facto de que muitos dispositivos e *smart apps* estejam suscetíveis a ciberataques, tornam-se alvos cativantes para

indivíduos mal-intencionados. Assim, estes dispositivos e aplicações sujeitam organizações e consumidores a novas vulnerabilidades de segurança introduzidas por esta tecnologia [21]. Apesar de se esperar que as tecnologias 5G e Wi-Fi 6, que prometem elevadas taxas de transferência, baixos requisitos de potência e alta largura de banda, abram uma nova página na indústria sem fios com novos cenários para a IoT [22], também se espera que aumentem a área de ataque para ciber-criminosos. Tal significa que os desafios de segurança da IoT tornar-se-ão maiores e mais significativos [23];

- **Privacidade do utilizador:** Muitos sistemas IoT proporcionam serviços personalizados que implicam uma compreensão correta das preferências e interesses dos utilizadores, bem como atividades diárias, e padrões de comportamento. Exemplos de tais equipamentos incluem dispositivos médicos inteligentes utilizáveis e implantáveis, tais como rastreadores de *fitness* e *pacemakers*, respetivamente. Estes podem registar as atividades diárias dos utilizadores, incluindo exercício, sono e ritmo cardíaco, e desta forma recolher um enorme volume de dados que são transmitidos através da Internet para serem armazenados, processados e analisados em sistemas *cloud* (de nuvem) IoT. Infelizmente, tais dados podem direta ou indiretamente revelar uma variedade de informações confidenciais, tais como o nome, a localização, números de cartão de crédito e de segurança social, e assim expor os utilizadores a diferentes tipos de violação da privacidade [24].

O âmbito da investigação descrita nesta tese enquadra-se nos domínios da IoT e da segurança de sistemas. Como tal, incide na investigação e tentativa de abordar os dois últimos desafios-chave acima mencionados, mais especificamente, na conceção e desenvolvimento de um sistema de segurança destinado a fornecer assistência em matéria de segurança a especialistas que não estejam envolvidos ativamente na conceção e implementação de dispositivos IoT e *smart apps*. O âmbito desta tese enquadra-se nos seguintes tópicos na versão de 2012 do Sistema de Classificação Informática ACM (CCS), a referência para a área de Informática e Ciências da Computação (por ordem de importância):

- **Segurança e privacidade~Segurança de sistemas;**
- **Organização de sistemas informáticos~Sistemas integrados físico e ciber-físicos;**
- *Software e a sua engenharia~Criação e gestão de software;*
- *Hardware~Tecnologias emergentes;*
- Informática orientada para o ser humano~Informática móvel e ubíqua.

Definição do Problema

As questões de segurança e privacidade da IoT são numerosas e englobam as diferentes camadas não padronizadas de abstração da IoT [25], desde a camada inferior, conhecida

como a camada de percepção ou física, até à camada mais superior, também conhecida como a camada de aplicação. Embora os ciberataques já existam há muito tempo, a novidade é a relativa simplicidade e a dimensão dos mesmos [26, 27]. Por exemplo, os ciber-criminosos podem utilizar dispositivos IoT comprometidos que estejam ligados a redes domésticas ou empresariais para lançar ataques de larga escala a sistemas ou aplicações de missão crítica. Em ataques de *botnet* ou *thingbot*, por exemplo, os ciber-criminosos utilizam uma rede de dispositivos comprometidos para desativar redes, sistemas informáticos ou até *websites* utilizando ataques distribuídos de negação de serviço (DDoS). Outras atividades ilícitas que utilizam *botnets* são o envio de *spam* e mensagens de *phishing*, que visam a explorar dados bancários e roubar informação privada [28].

O que torna estas questões de segurança e privacidade tão importantes é o facto de um grande número de dispositivos de borda, que representam a maioria dos terminais da IoT, estarem condicionados, no que respeita recursos tais como energia, memória, e capacidades computacionais. Estas limitações inerentes a vários dispositivos IoT requerem abordagens inovadoras e holísticas, visto que muitas metodologias de segurança tradicionais não podem ser aplicadas diretamente aos sistemas IoT de forma a protegê-los [29]. Independentemente das limitações mencionadas anteriormente, outros fatores que tornam os desafios de segurança e privacidade da IoT singulares incluem:

- **Baixo custo de equipamentos/dispositivos:** A implementação de algoritmos de segurança em dispositivos com grandes limitações de recursos é um grande desafio devido aos requisitos de memória e capacidade computacional [30], o que pode aumentar significativamente o custo dos dispositivos. No entanto, a maioria dos dispositivos de borda da IoT, tais como sensores, precisam de ser baratos e descartáveis, associado ao facto de que os fabricantes precisam de oferecer preços atrativos para competirem no mercado. Como resultado, os fabricantes são confrontados com o desafio de gerir o compromisso entre custo e recursos dos dispositivos em quantidade suficiente;
- **Alguns dispositivos estão expostos e desprotegidos:** Em alguns casos, os dispositivos de borda da IoT são colocados em terrenos difíceis onde ficam expostos e abandonados durante anos [31]. Em tais cenários, estes ficam mais suscetíveis a ataques, quer física ou logicamente;
- **Proteção fora do perímetro empresarial:** Ainda que o perímetro de segurança seja habitualmente utilizado para proteger sistemas informáticos, a sua utilização exclusiva não é suficiente para proteger uma rede constituída por dispositivos IoT. Tal deve-se ao facto de que a ligação direta destes dispositivos a uma rede empresarial pode desviar um ativo cibernético para fora dos limites do seu perímetro, pois a suscetibilidade de um único dispositivo pode gerar uma falha no perímetro de segurança. Adicionalmente, abordagens herdadas, tais como *Network Access Control* (NAC), Rede Privada Virtual (VPN) e *firewalls* estão sujeitas a vulnerabilidades, o

que pode ser aproveitado por atacantes [32];

- **Dispositivos podem deixar de ser suportados pelos fabricantes:** Alguns dispositivos IoT podem ter mais tempo de existência do que as próprias empresas que os fabricaram [33]. Embora possam estar ligados e conectados, deixam de receber atualizações de segurança e por isso tornam-se mais suscetíveis a ciberataques [34];
- **Limitações nas atualizações:** Alguns dispositivos IoT não possuem mecanismos de atualização e mesmo quando estes existam e possam corrigir falhas de segurança, torna-se um processo desafiante num grande número de dispositivos [35]. Além disso, mesmo quando os investigadores de segurança descobrem vulnerabilidades nos produtos de consumo e correções são lançadas pelas empresas afetadas, os utilizadores apresentam alguma relutância no processo de atualização nos seus dispositivos inteligentes, pois há uma falta de sensibilização para a segurança. Além disso, os utilizadores podem não ser capazes de realizar manualmente tais tarefas devido à falta de interfaces de utilizador adequadas;
- **Falta de experiência suficiente por parte de alguns fabricantes no que respeita a segurança:** Como as tecnologias de *smart sensing* e de conectividade estão cada vez mais integradas em produtos de consumo e as *smart apps* estão a tornar-se cada vez mais a nova plataforma de negócios, há um crescimento exponencial no número *start-ups* de software e hardware em todo o mundo. Embora isto possa ser considerado como um incentivo à inovação no âmbito da IoT, algumas destas empresas não dão à cibersegurança a prioridade que ela merece [36]. Por conseguinte, são produzidos dispositivos e *smart apps* com vulnerabilidades de segurança. Uma das causas fundamentais do problema é que algumas destas *start-ups* são compostas por engenheiros eletrónicos ou informáticos e/ou programadores com pouca ou nenhuma experiência na área da segurança [37]. Outra questão é que algumas empresas que produziam produtos de consumo tradicionais, tais como lâmpadas e torradeiras, repentinamente tornaram-se empresas de IoT. Algumas destas empresas simplesmente adicionam sensores e *widgets* de ligação à *Internet* aos seus produtos sem compreenderem as ramificações de segurança e as consequências inesperadas dos seus atos. Estas questões constituem um grande desafio no desenvolvimento de dispositivos IoT seguros e *smart apps*.

Esta tese propõe-se a investigar e abordar as questões levantadas no último item da lista de desafios de segurança e privacidade da IoT acima mencionados. Estas são questões críticas e exacerbadas pela falta de uma linha de base globalmente aplicável para a segurança na IoT para que fabricantes possam utilizar em todos os domínios de aplicação da IoT [38], e que devem ser abordadas rapidamente. Neste sentido, a necessidade de incorporar a segurança na conceção e desenvolvimento de dispositivos e *smart apps* tornou-se bastante evidente nos últimos anos devido aos efeitos dos ciberataques relacionados com

produtos e serviços da IoT. Os domínios da saúde e da indústria, por exemplo, sofreram perdas substanciais devido a vulnerabilidades nos dispositivos e *smart apps* [39], salientando a necessidade de dar prioridade à segurança e de abordar logo desde o início.

Objetivos da Investigação

Como se segue da discussão anterior, a *security-by-design* (segurança por construção) deve ser um conceito fundamental que deve suportar a conceção e o desenvolvimento de dispositivos IoT e *smart apps*. **A definição de *security-by-design* no âmbito da IoT:** No contexto da IoT, a segurança por construção é uma abordagem que procura incorporar os aspetos de segurança nos produtos e aplicações da IoT desde o primeiro instante, o que os tornará isentos de vulnerabilidades e resistentes a ataques tanto quanto possível [40, 41]. Isto compreende a realização de engenharia de segurança, integração, e testes de tecnologia de segurança em sistemas IoT. De acordo com o conceito de *security-by-design*, esta tese apresenta uma estrutura de segurança destinada a ajudar designers, engenheiros, programadores, e hobbyistas de eletrónica com pouca ou nenhuma experiência em segurança a conceber e desenvolver dispositivos IoT e *smart apps* mais seguros/as. O trabalho de investigação descrito nesta tese tem dois objetivos principais:

- Em primeiro lugar, pretende-se explorar as questões de segurança e privacidade na IoT, apresentando algumas ideias sobre a investigação que está a ser feita nesta área, incluindo os atuais desafios e carências, bem como salientar a necessidade de integrar a segurança e a privacidade na conceção e desenvolvimento de dispositivos IoT e a *smart apps* desde a conceção;
- Em segundo lugar, pretende-se propor, conceber e implementar um protótipo de um consultor de segurança eficiente e fácil de usar que se destina a servir de assessor que possa promover a conceção e desenvolvimento de dispositivos IoT e *smart apps* seguros/as, bem como avaliar o desempenho e a utilidade da estrutura de ferramentas que concretizam o consultor de assessoria de segurança.

Afirmação da Tese

Os ciberataques e outros riscos de segurança e privacidade associados a dispositivos conectados levaram a preocupações crescentes sobre segurança e privacidade na *Internet* das coisas. Por conseguinte, existem debates contínuos sobre quem deve assumir a responsabilidade pela segurança e privacidade da IoT: fabricantes e criadores de dispositivos ou os governos? No entanto, quando a questão é analisada do ponto de vista apropriado, resume-se, sem dúvida, à forma como os fabricantes e programadores estão a abordar o desenvolvimento da IoT como um todo, e particularmente, como o conceito de *security-by-design* é muitas vezes negligenciado pelos mesmos [42]. Esta questão foi posta na Secção 1.2, como um dos fatores que tornam ímpares os desafios de segurança e privacidade da *Internet* das coisas.

Embora tenham sido descobertas várias vulnerabilidades de segurança em alguns produtos de consumo inteligentes feitos por grandes gigantes da indústria tecnológica, atualmente a maioria das falhas de segurança na IoT estão a ser descobertas em produtos produzidos por pequenas *start-ups* [43, 33]. Para além da insuficiência de conhecimentos de segurança já salientada na Secção 1.2, outra razão possível para tal pode ser o facto de muitas dessas empresas serem desconhecidas e não se preocupam em defender a sua reputação ou a marca [44]. Isto constitui um grande desafio no fornecimento de segurança e proteção de privacidade adequadas na IoT. Dado que poucas destas *start-ups* podem financiar a contratação de especialistas em segurança e sendo difícil de as impedir de produzir mais e novos produtos e aplicações, tem de haver um modo de auxiliar tais empresas a produzirem produtos seguros, se o pretendido for minimizar as fragilidades, riscos, e outras questões de segurança e privacidade associadas aos sistemas IoT. A realização desta tarefa é um dos principais objetivos do trabalho de investigação descrito nesta tese, tal como descrito na subsecção 1.2.1, pelo que se desenvolve a seguinte afirmação de tese:

Um sistema de segurança que se baseia em atividades de engenharia de segurança para facilitar a incorporação da segurança no processo de design e o desenvolvimento de sistemas IoT pode ser um trajeto a seguir, de forma a alcançar um ecossistema IoT mais seguro. Tal sistema pode ser mais vantajoso para peritos não especialistas em segurança que estejam envolvidos ativamente na conceção e desenvolvimento destes sistemas; a utilização e os benefícios de algumas características de tal sistema podem estender-se para além da IoT, abrangendo outros sistemas informáticos.

A seguinte secção apresenta as metodologias adotadas de forma a dar resposta aos problemas mencionados na Secção 1.2; O plano de investigação está constituído por uma abordagem passo a passo, de modo a obter validação para a afirmação mencionada acima.

Metodologias Adotadas e Plano de Investigação

Os passos seguintes descrevem a abordagem escolhida para dar resposta aos problemas anteriormente referidos. Estes passos também representam o plano de investigação desenhado para alcançar os objetivos escolhidos no âmbito desta tese e investigação doutoral, como referido na Secção 1.2.1. Adicionalmente, este plano de investigação também representa os passos adotados para a validação da *Afirmação de Tese* apresentada na Secção 1.3:

1. **Triagem/Levantamento:** O primeiro passo na abordagem começou com um estudo exaustivo do estado da arte em matéria de segurança e privacidade na *Internet das coisas*, que abrangeu vários domínios de aplicação. Este passo envolveu a realização de um exame exaustivo de vários mecanismos e abordagens existentes para garantir a segurança e a privacidade na IoT;

2. **Identificação do problema:** Após uma cuidadosa revisão da literatura existente sobre segurança e privacidade na IoT, o autor identificou e delineou muitos problemas como questões em aberto, alguns dos quais são apresentados na Secção 1.2. Tendo analisado os problemas, esta tese centra-se no último ponto apresentado na Secção 1.2, que, indiscutivelmente, é um dos problemas que requerem atenção imediata, nomeadamente a falta de conhecimentos suficientes sobre segurança, por parte dos fabricantes;
3. **Estudo de Diferentes Aspectos de Implementação da Segurança na IoT:** Para abordar o problema identificado, o autor estudou diferentes aspetos da segurança e da implementação da IoT. Este estudo envolveu os requisitos de segurança em diferentes domínios de aplicação, melhores práticas de segurança, plataformas de desenvolvimento de hardware, criptografia leve, e implementação de algoritmos criptográficos leves em ambientes com recursos limitados;
4. **Processo de *Design Thinking* do Sistema:** O conceito de sistema de segurança nasceu dos estudos acima mencionados. Isto foi precedido pela análise do problema identificado, o que implicou a divisão do problema em elementos menores e de compreensão mais facilitada;
5. **Conceção e Implementação do Sistema:** : Esta etapa envolveu a conceção de cada uma das três componentes do sistema de segurança, a implementação da ferramenta que permite aos utilizadores interagir com o sistema, bem como o testar e aperfeiçoar os algoritmos;
6. **Avaliação do desempenho do Sistema:** Nesta etapa, a ferramenta que é a realização do sistema de segurança foi testada e avaliada. Envolveu a realização de uma série de testes de funcionalidade e utilidade por diferentes grupos de sujeitos.

Principais Contribuições Científicas

O trabalho de doutoramento descrito nesta tese representa uma das primeiras tentativas de fornecer apoio a peritos não especializados em segurança e que estão envolvidos na conceção e desenvolvimento de dispositivos e aplicações de IoT. O trabalho forneceu uma série de contribuições científicas, algumas das quais são expostas abaixo:

- **Um levantamento exaustivo sobre segurança e privacidade na IoT:** primeira contribuição deste trabalho de investigação é um levantamento detalhado e abrangente do estado da arte no que respeita a segurança e a privacidade na IoT. Algumas partes deste levantamento são apresentadas no Capítulo 2 da presente tese. Este contém uma descrição detalhada de diferentes aspetos da segurança e privacidade da IoT e outros aspetos em geral, incluindo uma descrição de vários domínios de aplicação e identificação de ativos cibernéticos por domínio. Esta contribuição é relatada no primeiro artigo [45] na primeira lista de publicações apresentada na subsecção seguinte;

- **Identificação de Requisitos de Segurança e Ameaças de Segurança em Domínios de Arquitetura e Indústria de IoT:** A segunda contribuição é a identificação de vários requisitos de segurança e ameaças à segurança na arquitetura da IoT e em diferentes domínios da indústria IoT, bem como o fornecimento de várias contramedidas para lidar com as ameaças à segurança identificadas. A primeira parte desta contribuição está referida no sétimo artigo [46] e a outra parte no artigo de conferência [47], no terceiro item da primeira lista de publicações apresentada na subsecção 1.5.1;
- **Identificação dos Requisitos de Segurança nos Domínios de Aplicação da IoT:** O terceiro contributo é a identificação dos requisitos de segurança em vários domínios de aplicação, incluindo cuidados de saúde inteligentes, *smart grid*, cidades inteligentes, casas inteligentes, processos de fabrico inteligentes, cadeias de abastecimento inteligentes, agricultura inteligente, e meios de transporte inteligentes. Esta contribuição está mencionada tanto no primeiro como no terceiro documento [45, 47] da primeira lista de publicações delineada na subsecção 1.5.1;
- **Revisão Extensiva de Características Importantes de Várias Plataformas de Desenvolvimento de Hardware IoT:** A quarta contribuição inclui uma extensa revisão que examina várias plataformas de desenvolvimento de hardware IoT que foram lançadas no passado, que foram lançadas recentemente, e as que serão lançadas no mercado num futuro próximo. Concentra-se em alguns pontos essenciais das plataformas de desenvolvimento de hardware que incluem a velocidade de processamento, capacidade de memória, duração da bateria, e elementos de segurança. Inclui também orientações passo a passo das melhores práticas para a conceção e prototipagem de projetos IoT. Esta contribuição é relatada nos sexto e oitavo artigos [48, 49] da primeira lista de publicações apresentada na subsecção 1.5.1;
- **Estudo Abrangente das Melhores Práticas de Segurança e Privacidade na IoT:** O quinto contributo contempla um estudo extensivo dos desafios do desenvolvimento de práticas de segurança e privacidade na IoT definidas em comum acordo, bem como a exploração de várias tentativas de desenvolver melhores práticas de segurança e privacidade mais amplamente aceites para a IoT. Esta contribuição é relatada no décimo artigo da primeira lista de publicações apresentada na subsecção 1.5.1;
- **Considerações sobre a conceção e implementação de software e hardware:** A sexta contribuição diz respeito a uma discussão detalhada sobre a conceção e implementação de software e hardware e considerações para os Algoritmos Criptográficos Leves (LWCAs). Esta contribuição é relatada no segundo artigo [43] na primeira lista de publicações delineada na Subsecção 1.5.1;
- **Conceção e Implementação do Sistema IoT-HarPSecA:** A sétima contribuição é sobre a conceção e implementação das três componentes do sistema de segurança,

bem como uma avaliação detalhada do desempenho da principal componente da IoT-HarPSecA. Esta contribuição é descrita no segundo e no quarto documento [43, 50] que constam na primeira lista de publicações apresentada na subsecção 1.5.1;

- **Desenvolvimento de uma Representação de Conhecimento de um Especialista Humano:** Na oitava contribuição consta o desenvolvimento da representação do conhecimento humano especializado sob a forma de procedimentos de tomada de decisão de peritos nos domínios da IoT e da *Lightweight Cryptography* (LWC). Esta contribuição é também relatada no segundo artigo [43] da primeira lista de publicações delineada na subsecção 1.5.1;
- **Avaliação do Desempenho e da Utilidade da ferramenta do Sistema IoT-HarPSecA:** A última contribuição desta tese é a avaliação do desempenho e da utilidade dos três componentes da ferramenta do sistema IoT-HarPSecA. Esta contribuição é relatada tanto no segundo, nono e décimo artigos [43, 51] da primeira lista de publicações apresentada na subsecção seguinte.

Publicações

No âmbito desta tese de doutoramento e da investigação, os seguintes artigos foram produzidos e já publicados em revistas científicas, atas de conferências e em livros submetidos a revisão por pares, à exceção do último artigo da primeira lista, que foi submetido para publicação à revista *IEEE Internet of Things Journal* e aguarda resposta:

1. **Challenges of Securing Internet of Things Devices: A survey** [45]
Musa G. Samaila, Miguel Neto, Diogo A. B. Fernandes, Mário M. Freire and Pedro R. M. Inácio, *Security and Privacy*, vol. 1, issue 2, pp. 1-32, May 2018.
DOI: <https://doi.org/10.1002/spy2.20>
2. **IoT-HarPSecA: A Framework and Roadmap for Secure Design and Development of Devices and Applications in the IoT Space** [43]
Musa G. Samaila, João B. F. Sequeiros, Tiago Simões, Mário M. Freire and Pedro R. M. Inácio, *IEEE Access*, vol. 8, pp. 16462-16494, January 2020.
DOI: [10.1109/ACCESS.2020.2965925](https://doi.org/10.1109/ACCESS.2020.2965925)
3. **Security Threats and Possible Countermeasures in IoT Applications Covering Different Industry Domains** [47]
Musa G. Samaila, João b. F. Sequeiros, Mário M. Freire and Pedro R. M. Inácio, *In ARES 2018: International Conference on Availability, Reliability and Security*, August 27–30, 2018, Hamburg, Germany. ACM, New York, NY, USA, Article 16, 9 pages.
DOI: [10.1145/3230833.3232800](https://doi.org/10.1145/3230833.3232800)
4. **IoT-HarPSecA: A Framework for Facilitating the Design and Development of Secure IoT Devices** [50]

- Musa G. Samaila, Moser Z. V. José, João B. F. Sequeiros, Mário M. Freire and Pedro R. M. Inácio, *In Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES 2019) (ARES '19)*, August 26–29, 2019, Canterbury, United Kingdom. ACM, New York, NY, USA, Article 4, 7 pages.
DOI: 10.1145/3339252.3340514
5. **Security Challenges of the Internet of Things [52]**
Musa G. Samaila, Miguel Neto, Diogo A. B. Fernandes, Mário M. Freire and Pedro R. M. Inácio, in *Beyond the Internet of Things: Everything Interconnected*. Jordi Mongay Batalla, George Mastorakis, Constandinos X. Mavromoustakis, Evangelos Pallis (Eds.), *Springer International Publishing*, ISBN: 978-3-319-50756-9, pp. 53-82, 2017.
DOI: 10.1007/978-3-319-50758-3_3 53
 6. **IoT Hardware Development Platforms: Past, Present, and Future [48]**
Musa G. Samaila, João B. F. Sequeiros, Acácio F. P. P. Correia, Mário M. Freire and Pedro R. M. Inácio, in *Internet of Things: Challenges, Advances, and Applications*. Qusay F. Hassan, Atta ur Rehman Khan, Sajjad A. Madani (Eds.), *CRC Press*, ISBN: 13: 978-1-4987-7851-0, pp. 107-139, 2018.
 7. **A Quick Perspective on the Current State of IoT Security: A Survey [46]**
Musa G. Samaila, João B. F. Sequeiros, Acácio F. P. P. Correia, Mário M. Freire and Pedro R. M. Inácio, in *Networks of the Future: Architectures, Technologies, and Implementations*. Mahmoud Elkhodr, Qusay F. Hassan, Seyed Shahrestani (Eds.), *CRC Press*, ISBN: 13: 978-1-4987-8397-2, pp. 431-464, 2017.
 8. **A Tutorial Introduction to IoT Design and Prototyping with Examples [49]**
Manuel Meruje, Musa G. Samaila, Virginia N. L. Franqueira, Mário M. Freire and Pedro R. M. Inácio, in *Internet of things A to Z: Technologies and Applications*. Qusay F. Hassan (ed.), *Wiley-IEEE Press*, ISBN: 978-111-945674-2, pp. 153-190, 2018.
DOI: <https://doi.org/10.1002/9781119456735.ch6>
 9. **A Preliminary Evaluation of the SRE and SBPG Components of the IoT-HarPsecA Framework [51]**
Musa G. Samaila, Carolina Lopes, Édi Aires, João B. F. Sequeiros, Tiago Simões, Mário M. Freire and Pedro R. M. Inácio, *In Proceedings of the 2020 Global Internet of Things Summit (GIOTS)*, June 3-3, 2020, Dublin, Ireland. IEEE, 7 pages.
DOI: 10.1109/GIOTS49054.2020.9119590
 10. **Performance Evaluation of the SRE and SBPG Components of the IoT-HarPsecA Framework**
Musa G. Samaila, Carolina Lopes, Édi Aires, João B. F. Sequeiros, Tiago Simões, Mário M. Freire and Pedro R. M. Inácio, *IEEE Internet of Things Journal* (submitted for publication).

Outros trabalhos publicados durante o período desta tese e trabalho de investigação incluem:

1. **Attack and System Modeling Applied to IoT, Cloud, and Mobile Ecosystems: Embedding Security by Design** [53]

João b. F. Sequeiros, Francisco T. Chimuco, Musa G. Samaila, Mário M. Freire and Pedro R. M. Inácio, *ACM Computing Surveys*, vol. 53, no. 2, Article 25, pp. 1-32, March 2020.

DOI: 10.1145/3376123

Estado da Arte

O capítulo 2, baseado nas publicações 1, 3, 5, 6, 7, e 8 [45, 47, 52, 48, 46, 49], descreve alguns conceitos básicos do paradigma da IoT e depois analisa alguns conceitos importantes na segurança e privacidade desta tecnologia. O capítulo começa com a descrição de uma arquitetura IoT que consiste em três camadas, nomeadamente as camadas de perceção, de rede e a de aplicação. Apresenta ainda os principais requisitos de segurança e identifica algumas ameaças à segurança desta arquitetura. Posteriormente, descreve nove domínios de aplicação e apresenta modelos de sistema e modelos de ameaças a estes domínios. Além disso, discute a problemática de ameaças à IoT, bem como algumas questões fundamentais de segurança e privacidade. Evidenciam-se vários compromissos de desempenho *versus* compromissos de segurança e, por fim, apresentam-se contramedidas possíveis de segurança e privacidade. Finalmente, o capítulo fornece uma breve visão geral das plataformas de desenvolvimento de hardware, onde é abordada a classificação geral e características chave das plataformas de desenvolvimento de hardware da IoT.

Conceitos básicos do sistema IoT-HarPSecA

O capítulo 3 fundamenta-se principalmente nos artigos 1, 2, e 10 [45, 43], estando orientado para a discussão dos conceitos e princípios subjacentes à estrutura de segurança na IoT. O capítulo começa por discutir os requisitos de segurança, que fornecem a base sobre a qual se constrói a primeira componente do sistema de segurança. Nomeadamente, identifica alguns requisitos de segurança importantes para cada um dos nove domínios de aplicação discutidos no capítulo 2. Além disso, apresenta e discute as melhores práticas de segurança e privacidade, que constituem o fundamento da segunda componente do sistema de segurança. Isto inclui a discussão de alguns desafios de desenvolvimento de regulamentos e diretrizes acordados em comum com base nas melhores práticas industriais, bem como a indicação de algumas tentativas anteriores de desenvolver melhores práticas de segurança genericamente mais aceites para a IoT. Finalmente, o capítulo apresenta uma visão geral da Criptografia Leve (LWC) e dos LWCA, que servem de base essencial ao terceiro componente do sistema IoT-HarPSecA. Este capítulo serve de base para uma apresentação formal do sistema de segurança no capítulo seguinte.

O sistema IoT-HarPSecA

O capítulo 4 aprofunda-se na concepção e implementação do sistema IoT-HarPSecA. O capítulo é baseado nas publicações 2 e 4 [43, 50]. O IoT-HarPSecA oferece três funcionalidades principais, nomeadamente a eliciação de normas de segurança, geração de um conjunto de diretrizes de melhores práticas de segurança para o desenvolvimento seguro e, acima de tudo, uma característica que recomenda LWCA's específicas tanto para implementações de software como de hardware. Por conseguinte, o IoT-HarPSecA é composta por três componentes principais: (1) o componente Eliciação de Requisitos de Segurança (SRE), (2) o componente Orientações de Melhores Práticas de Segurança (SBPG), e (3) o componente Recomendação de Componentes de Algoritmos Criptográficos Leves (LWCAR), cada um deles servindo cada uma das características acima mencionadas, respetivamente. A estrutura é modular na configuração, e isto permite a fácil integração de novas funcionalidades, bem como a simples atualização das funcionalidades existentes. Além disso, o capítulo discute o desenho e a descrição dos diferentes módulos em cada um dos três componentes do sistema. Finalmente, descreve a implementação da ferramenta do IoT-HarPSecA, a qual os utilizadores podem utilizar para interagir com o sistema de segurança.

Testes de Funcionalidade e Utilidade, Resultados e Avaliação

O capítulo 5 apresenta os diferentes testes e resultados utilizados para avaliar o desempenho e a usabilidade do sistema IoT-HarPSecA. O capítulo baseia-se maioritariamente nas publicações 2, 4, 9, e 10 [43, 50, 51] e descreve uma série de cenários de testes semelhantes aos do mundo real utilizados na avaliação. Os testes foram realizados nas ferramentas SRE, SBPG, e LWCAR por diferentes grupos de indivíduos constituídos por programadores, engenheiros eletrónicos e informáticos. Tanto os testes das ferramentas SRE como SBPG foram realizados ao mesmo tempo por cerca de 24 participantes. Porém, devido a limitações de espaço, apenas três resultados do teste da ferramenta SRE e três resumos de resultados do teste da ferramenta SBPG são apresentados neste capítulo. Do mesmo modo, no caso dos testes da ferramenta LWCAR, apenas quatro resultados (ou seja, dois para pedidos de implementação de software e dois para pedidos de implementação de hardware) foram apresentados dos primeiros quatro testes, realizados por 17 indivíduos, devido a restrições de espaço. Na mesma ótica, seis sujeitos participaram no quinto teste realizado com a ferramenta LWCAR, mas apenas um resultado foi apresentado. A avaliação dos resultados apresentados neste capítulo demonstra que o IoT-HarPSecA pode servir como um roteiro para a concepção e desenvolvimento de sistemas IoT seguros.

Conclusões e Perspetivas Futuras

No Capítulo 6 constam as principais conclusões do trabalho de investigação descrito na presente tese. Primeiro, apresenta-se um sumário das principais contribuições, desta-

cadras na secção das *Principais Contribuições Científicas*, seguindo-se uma avaliação de resultados ou mensuração da realização dos objetivos definidos na subsecção *Objetivos de Investigação*, de modo a verificar se estes foram ou não atingidos. Por outro lado, determina-se também se o trabalho de investigação descrito nesta tese é ou não coerente com a declaração prestada na secção *Afirmação da Tese*, após a qual estão presentes as conclusões. A conclusão deste trabalho de investigação e tese e de que os resultados dos testes e a avaliação observadas, provaram que o sistema IoT-HarPSecA pode facilitar e de forma eficaz a conceção e desenvolvimento de sistemas de IoT seguros. Finalmente, o capítulo termina a tese com algumas limitações na pesquisa e investigação e providencia orientações para que mais investigações possam ser feitas sobre o assunto.

Anexos

No anexo A consta o resultado completo ou a versão alargada das diretrizes de melhores práticas de segurança produzidas pela ferramenta SBPG do sistema IoT-HarPSecA referida na Secção 5.1 e Subsecção 5.3.2. No anexo B, encontram-se as capturas de ecrã das três tabelas de base de dados MySQL das ferramentas SRE, SBPG, e LWCAR, cujos dados estão nas Tabelas 5.2-5.3, Tabelas 5.9-5.11, e Tabelas 5.22-5.24, e mencionados nas Subsecções 5.2.2, 5.3.2 e 5.4.2, respetivamente. Verificam-se também capturas de ecrã da base de dados MySQL e tabelas das ferramentas SRE e LWCAR para os seis assuntos mencionados na Subsecção 5.4.2 e na Secção 5.5. O anexo C apresenta os resultados resumidos, relativamente ao teste da ferramenta SBPG para os assuntos identificados como IDs B2278 e B7788, que foram abordados e discutidos na Subsecção 5.3.2. Por fim, o anexo D demonstra os requisitos de segurança para o sistema de IoT, cujo assunto foi identificado como ID R2143, que representa o número máximo de requisitos de segurança que a ferramenta SRE do sistema IoT-HarPSecA é capaz de gerar, como discutido na Subsecção 5.4.2. Pode-se ainda observar o relatório completo a respeito do resultado do teste da ferramenta LWCAR sobre o assunto ID S2143 abordado na Subsecção 5.4.2.

Abstract

The term Internet of Things (IoT) describes an ever-growing ecosystem of physical objects or *things* interconnected with each other and connected to the Internet. IoT devices consist of a wide range of highly heterogeneous inanimate and animate objects. Thus, a *thing* in the context of the IoT can even mean a person with blood pressure or heart rate monitor implant or a pet with a biochip transponder. IoT devices range from ordinary household appliances, such as smart light bulbs or smart coffee makers, to sophisticated tools for industrial automation. IoT is currently leading a revolutionary change in many industries and, as a result, a lot of industries and organizations are adopting the paradigm to gain a competitive edge. This allows them to boost operational efficiency and optimize system performance through real-time data management, which results in an optimized balance between energy usage and throughput. Another important application area is the Industrial Internet of Things (IIoT), which is the application of the IoT in industrial settings. This is also referred to as the Industrial Internet or Industry 4.0, where Cyber-Physical Systems (CPS) are interconnected using various technologies to achieve wireless control as well as advanced manufacturing and factory automation. IoT applications are becoming increasingly prevalent across many application domains, including smart healthcare, smart cities, smart grids, smart farming, and smart supply chain management. Similarly, IoT is currently transforming the way people live and work, and hence the demand for smart consumer products among people is also increasing steadily. Thus, many big industry giants, as well as startup companies, are competing to dominate the market with their new IoT products and services, and hence unlocking the business value of IoT.

Despite its increasing popularity, potential benefits, and proven capabilities, IoT is still in its infancy and fraught with challenges. The technology is faced with many challenges, including connectivity issues, compatibility/interoperability between devices and systems, lack of standardization, management of the huge amounts of data, and lack of tools for forensic investigations. However, the state of insecurity and privacy concerns in the IoT are arguably among the key factors restraining the universal adoption of the technology. Consequently, many recent research studies reveal that there are security and privacy issues associated with the design and implementation of several IoT devices and Smart Applications (smart apps). This can be attributed, partly, to the fact that as some IoT device makers and smart apps development companies (especially the start-ups) reap business value from the huge IoT market, they tend to neglect the importance of security. As a result, many IoT devices and smart apps are created with security vulnerabilities, which have resulted in many IoT related security breaches in recent years.

This thesis is focused on addressing the security and privacy challenges that were briefly highlighted in the previous paragraph. Given that the Internet is not a secure environ-

ment even for the traditional computer systems makes IoT systems even less secure due to the inherent constraints associated with many IoT devices. These constraints, which are mainly imposed by cost since many IoT edge devices are expected to be inexpensive and disposable, include limited energy resources, limited computational and storage capabilities, as well as lossy networks due to the much lower hardware performance compared to conventional computers. While there are many security and privacy issues in the IoT today, arguably a root cause of such issues is that many start-up IoT device manufacturers and smart apps development companies do not adhere to the concept of security by design. Consequently, some of these companies produce IoT devices and smart apps with security vulnerabilities.

In recent years, attackers have exploited different security vulnerabilities in IoT infrastructures which have caused several data breaches and other security and privacy incidents involving IoT devices and smart apps. These have attracted significant attention from the research community in both academia and industry, resulting in a surge of proposals put forward by many researchers. Although research approaches and findings may vary across different research studies, the consensus is that a fundamental prerequisite for addressing IoT security and privacy challenges is to build security and privacy protection into IoT devices and smart apps from the very beginning. To this end, this thesis investigates how to bake security and privacy into IoT systems from the onset, and as its main objective, this thesis particularly focuses on providing a solution that can foster the design and development of secure IoT devices and smart apps, namely the IoT Hardware Platform Security Advisor (IoT-HarPSecA) framework. The security framework is expected to provide support to designers and developers in IoT start-up companies during the design and implementation of IoT systems. IoT-HarPSecA framework is also expected to facilitate the implementation of security in existing IoT systems.

To accomplish the previously mentioned objective as well as to affirm the aforementioned assertion, the following step-by-step problem-solving approach is followed. The first step is an exhaustive survey of different aspects of IoT security and privacy, including security requirements in IoT architecture, security threats in IoT architecture, IoT application domains and their associated cyber assets, the complexity of IoT vulnerabilities, and some possible IoT security and privacy countermeasures; and the survey wraps up with a brief overview of IoT hardware development platforms. The next steps are the identification of many challenges and issues associated with the IoT, which narrowed down to the above-mentioned fundamental security/privacy issue; followed by a study of different aspects of security implementation in the IoT. The remaining steps are the framework design thinking process, framework design and implementation, and finally, framework performance evaluation.

IoT-HarPSecA offers three functionality features, namely security requirement elicitation,

security best practice guidelines for secure development, and above all, a feature that recommends specific Lightweight Cryptographic Algorithms (LWCAs) for both software and hardware implementations. Accordingly, IoT-HarPSecA is composed of three main components, namely Security Requirements Elicitation (SRE) component, Security Best Practice Guidelines (SBPG) component, and Lightweight Cryptographic Algorithms Recommendation (LWCAR) component, each of them servicing one of the aforementioned features. The author has implemented a command-line tool in C++ to serve as an interface between users and the security framework. This thesis presents a detailed description, design, and implementation of the SRE, SBPG, and LWCAR components of the security framework. It also presents real-world practical scenarios that show how IoT-HarPSecA can be used to elicit security requirements, generate security best practices, and recommend appropriate LWCAs based on user inputs. Furthermore, the thesis presents performance evaluation of the SRE, SBPG, and LWCAR components framework tools, which shows that IoT-HarPSecA can serve as a roadmap for secure IoT development.

Keywords

Application domain, cryptographic algorithms, cryptography, hardware implementation of lightweight cryptographic algorithms, internet of things, IoT, IoT hardware platform security advisor, IoT-HarPSecA, lightweight cryptographic algorithms recommendation, lightweight cryptographic algorithms, lightweight cryptography, privacy, security best practices, security best practice guidelines, security by design, security framework, security requirement elicitation, security requirements, security, software implementation of lightweight cryptographic algorithms, system model, threat model.

Contents

Dedication	iii
Acknowledgements	v
Resumo	ix
Resumo Alargado	xiii
Abstract	xxix
Contents	xxxviii
List of Figures	xl
List of Tables	xli
Acronyms and Abbreviations	xliii
1 Introduction	1
1.1 Thesis Focus and Scope	1
1.1.1 Motivation	2
1.2 Problem Definition	4
1.2.1 Research Objectives	6
1.3 Thesis Statement	7
1.4 Adopted Approach and Research Plan	8
1.5 Main Scientific Contributions	9
1.5.1 Publications	10
1.6 Thesis Organization	12
2 State-of-the-Art	15
2.1 IoT Architecture	15
2.1.1 Security Requirements in IoT Architecture	16
2.1.1.1 Sensing Layer Security Requirements	16
2.1.1.2 Network Layer Security Requirements	17
2.1.1.3 Application Layer Security Requirements	18
2.1.2 Security Threats in IoT Architecture	19
2.1.2.1 Security Threats in the Perception Layer	19
2.1.2.2 Security Threats in the Network Layer	20
2.1.2.3 Security Threats in the Application Layer	21
2.2 IoT Application Domains	21
2.2.1 IoT Application Domains and their Associated Cyber Assets	21

2.2.1.1	Smart Home	22
2.2.1.2	Smart Grid	22
2.2.1.3	Smart City	23
2.2.1.4	Smart Transportation	23
2.2.1.5	Smart Healthcare	23
2.2.1.6	Smart Manufacturing	25
2.2.1.7	Smart Supply Chain	25
2.2.1.8	Smart Wearable	25
2.2.1.9	Smart Farming	26
2.3	IoT System Models, Threat Models, and Threat Landscape	26
2.3.1	IoT System Models	26
2.3.1.1	Smart Home System Model	27
2.3.1.2	Smart Grid System Model	27
2.3.1.3	Smart City System Model	27
2.3.1.4	Smart Transportation System Model	27
2.3.1.5	Smart Healthcare System Model	29
2.3.1.6	Smart Manufacturing System Model	29
2.3.1.7	Smart Supply Chain System Model	29
2.3.1.8	Smart Wearable System Model	29
2.3.1.9	Smart Farming System Model	30
2.3.2	IoT Threat Models	30
2.3.2.1	Smart Home Threat Model	31
2.3.2.2	Smart Grid Threat Model	31
2.3.2.3	Smart City Threat Model	31
2.3.2.4	Smart Transportation Threat Model	32
2.3.2.5	Smart Healthcare Threat Model	32
2.3.2.6	Smart Manufacturing Threat Model	33
2.3.2.7	Smart Supply Chain Threat Model	33
2.3.2.8	Smart Wearable Threat Model	33
2.3.2.9	Smart Farming Threat Model	34
2.3.3	Overview of IoT Cyber Threat Landscape	34
2.3.3.1	IoT is Fast Changing the Cyber Threat Landscape	35
2.3.3.2	IoT is Attracting the Attention of Malicious Attackers	36
2.3.3.3	IoT Threat Actors	36
2.3.4	The Complexity of IoT Vulnerabilities	37
2.3.5	Evolving Attack Strategies Against the IoT	39
2.3.6	Attacks on the Privacy of Users	40
2.4	IoT Fundamental Security Issues, Trade-offs, and Countermeasures	41
2.4.1	IoT Fundamental Security and Privacy Issues	41
2.4.1.1	Smart Devices and Applications are not Designed with Security in Mind	41

2.4.1.2	Open Debugging Interfaces	42
2.4.1.3	Inappropriate Network Configuration	43
2.4.1.4	Lack of Encryption of Critical Information Before Storage	44
2.4.2	Security versus Performance Trade-offs in IoT	45
2.4.2.1	Security versus Energy Trade-offs	46
2.4.2.2	Security versus Processing/Memory Capacity Trade-offs .	46
2.4.2.3	Security versus Cost Trade-offs	46
2.4.2.4	Security versus Convenience Trade-offs	46
2.4.3	Possible IoT Security and Privacy Countermeasures	47
2.4.3.1	Countermeasures Against Threats Associated with Inse- secure Web Interfaces and Network Services	47
2.4.3.2	Countermeasures Against Threats on Routing Protocols .	48
2.4.3.3	Countermeasures Against Threats on Information in Transit	48
2.4.3.4	Countermeasures Against Physical/Environmental Threats	49
2.4.3.5	Countermeasure Against GPS Jamming	49
2.4.3.6	Countermeasures Against Tag Tracking and Tag Cloning .	50
2.4.3.7	Countermeasures Against Inappropriate Network Config- uration	50
2.5	Brief Overview of IoT Hardware Development Platforms	50
2.5.1	General Classification of IoT Hardware Development Platforms . .	51
2.5.1.1	Microcontroller-based IoT Hardware Development Plat- forms	51
2.5.1.2	System on Chips	51
2.5.1.3	Single Board Computers	51
2.5.2	Key Features of IoT Hardware Development Platforms	52
2.5.2.1	Processing and Memory/Storage Capacity	53
2.5.2.2	Power Consumption, Size and Cost	53
2.5.2.3	Operating Systems	54
2.5.2.4	Connectivity and Peripherals	54
2.5.2.5	Hardware Security Features	55
2.6	Conclusion	55
3	Basic Concepts of IoT-HarPSecA Framework	57
3.1	Introduction	57
3.2	Security Requirements in the IoT	58
3.2.1	Security Requirements per Domain	59
3.2.1.1	Smart Home Security Requirements	59
3.2.1.2	Smart Grid Security Requirements	60
3.2.1.3	Smart City Security Requirements	61
3.2.1.4	Smart Transportation Security Requirements	62
3.2.1.5	Smart Healthcare Security Requirements	63
3.2.1.6	Smart Manufacturing Security Requirements	64

3.2.1.7	Smart Supply Chain Security Requirements	65
3.2.1.8	Smart Wearable Security Requirements	66
3.2.1.9	Smart Farming Security Requirements	67
3.2.1.10	Security Mechanisms for Achieving some Security Requirements	67
3.3	IoT Security and Privacy Best Practices	69
3.3.1	Challenges of Developing a Commonly Agreed-upon IoT Security and Privacy Best Practices	69
3.3.2	Attempts to Develop More Widely Accepted Security and Privacy Best Practices for the IoT	70
3.3.2.1	IoT Laws and Regulations	70
3.3.2.2	IoT Standards	72
3.3.3	Security and Privacy Best Practices for IoT Manufacturers and Developers	73
3.3.3.1	Secure Boot	73
3.3.3.2	Secure OS	74
3.3.3.3	Management of Security Credentials	74
3.3.3.4	Encryption Algorithm	75
3.3.3.5	IoT App Security	76
3.3.3.6	IoT Physical Security	76
3.3.4	Security and Privacy Best Practices for Enterprises and Individual IoT Users	77
3.4	An Overview of Lightweight Cryptography and Lightweight Cryptographic Algorithms	78
3.4.1	Types of Lightweight Cryptographic Algorithms	78
3.4.1.1	Lightweight Block Ciphers	78
3.4.1.2	Lightweight Stream Ciphers	80
3.4.1.3	Lightweight Cryptographic Hash Functions	81
3.4.1.4	Lightweight Message Authentication Code (MAC)	82
3.4.1.5	Lightweight Authenticated Encryption (AE)	84
3.4.2	Lightweight Cryptography Design and Implementation Considerations	85
3.4.2.1	Software Considerations	86
3.4.2.2	Hardware Considerations	89
3.4.3	Lightweight Cryptography for IoT	92
3.5	Conclusion	93
4	The IoT-HarPSecA Framework	95
4.1	Introduction	95
4.2	Design and Description of the Security Requirements Elicitation Component	96
4.2.1	User Interface	96
4.2.2	Security Requirements Generator	98

4.2.3	Storage and Updates	98
4.2.4	Admin Interface	99
4.2.5	Output Interface	100
4.3	Design and Description of the Security Best Practice Guidelines Component	100
4.3.1	User Interface	101
4.3.2	Report Generator	102
4.4	Design and Description of the Lightweight Cryptographic Algorithms Recommendation Component	102
4.4.1	User Interface	103
4.4.2	Filtering and Query Processing	104
4.4.3	Security Manager	104
4.5	IoT-HarPsecA Tool Implementation	108
4.5.1	Implementations of the SRE and SBPG Components of the Tool	110
4.5.2	Implementation of the LWCAR Component Tool	111
4.6	Conclusion	113
5	Functionality and Usability Tests, Results and Evaluation	115
5.1	Introduction	115
5.2	Test, Results, and Performance Evaluation of the SRE Tool	116
5.2.1	SRE Tool Test Setup	116
5.2.1.1	SRE Tool Test Scenario	117
5.2.2	Results Overview of the SRE Tool Test	117
5.2.3	Evaluation and Discussion of the SRE Tool Test Results	124
5.2.3.1	Security Requirements Elicitation Accuracy of SRE Tool	124
5.2.3.2	Simplicity and Ease of Use of the SRE Tool	127
5.3	Test, Results, and Performance Evaluation of the SBPG Tool	128
5.3.1	SBPG Tool Test Setup	128
5.3.1.1	SBPG Tool Test Scenario	129
5.3.2	Results Overview of the SBPG Tool Test	129
5.3.3	Evaluation and Discussion of the SBPG Tool Test Results	130
5.3.3.1	Security Best Practices Accuracy of SBPG Tool	137
5.3.3.2	Simplicity and Ease of Use of SBPG Tool	140
5.4	Tests, Results, and Performance Evaluation of the LWCAR Tool	140
5.4.1	LWCAR Tool Test Setup	140
5.4.1.1	LWCAR Tool Test Scenarios	141
5.4.2	Results Overview of the LWCAR Tool Tests	143
5.4.3	Evaluation and Discussion of the LWCAR Tool Test Results	149
5.4.3.1	Comparison of the Decision-Making Process of LWCAR Tool Versus Decision-Making Process of Human Experts	150
5.4.3.2	Decision Accuracy of the LWCAR Tool	151
5.4.3.3	Simplicity and Ease of Use of LWCAR Tool	154
5.5	Conclusion	154

6	Conclusions and Future Work	157
6.1	Summary of Main Scientific Contributions	157
6.2	Assessing the Achievement of Research Objectives and Validating Thesis Statement	158
6.3	Final Conclusions	160
6.4	Research Limitations and Future Work	161
	Bibliography	165
A	Full-Length SBPG Result for Subject with Request ID B5555	191
B	Actual Screenshots of Request Data of Subjects as Stored in the MySQL Database	203
C	Result Summaries of SBPG Test for Subjects with Request Identifications (IDs) B2278 and B7788	207
D	Maximum Number of Security Requirements and Full Report on LWCAR Result	211

List of Figures

2.1	Basic IoT architecture (adapted from [45]).	16
2.2	A generic security architecture for IoT (adapted from [47]).	20
2.3	Typical IoT application domains (adapted from [45]).	22
2.4	Overview of typical IoT cyber assets for nine application domains (adapted from [45]).	24
2.5	Typical system models for the nine application domains (adapted from [45]).	28
2.6	Examples of IoT hardware development platforms.	52
3.1	A high-level representation of a typical Harvard based MCU memory segments (adapted from [43]).	87
4.1	IoT-HarPSecA system components (adapted from [43]).	96
4.2	A high-level architecture of the SRE component of the IoT-HarPSecA (adapted from [43]).	97
4.3	IoT-HarPSecA main menu.	98
4.4	IoT-HarPSecA admin menu.	99
4.5	A high-level architecture of the SBPG component of the IoT-HarPSecA (adapted from [43]).	101
4.6	A high-level architecture of the LWCAR component of the IoT-HarPSecA (adapted from [43]).	103
4.7	Software implementation request workflow for the LWCAR component (adapted from [43]).	105
4.8	Hardware implementation request workflow for the LWCAR component (adapted from [43]).	106
5.1	Professional competence and security experience of the 24 subjects.	120
5.2	Simplicity and ease of use of the SRE tool.	121
5.3	Request completion times of subjects for the SRE tool test.	121
5.4	Final results of SRE request for subject with request ID R1995.	122
5.5	Final results of SRE request for subject with request ID R5432.	122
5.6	Final results of SRE request for subject with request ID R1287.	123
5.7	Simplicity and ease of use of the SBPG tool.	135
5.8	Request completion times of subjects for the SBPG tool test.	135
5.9	Final results of SBPG request for subject with request ID B5555.	136
5.10	Professional competence and security experience of the 17 subjects.	143
5.11	Simplicity and ease of use of the LWCAR tool.	145
5.12	Request completion times of subjects for the LWCAR tool test.	145
5.13	Final software results for subjects with request IDs S4219 and S8888.	147
5.14	Final hardware results for subjects with request IDs H1598 and H3456.	148

A.1	Full-Length best practice guidelines for subject with request ID B5555. . .	201
B.1	Screenshots of SRE request data of subjects as stored in the database. . . .	203
B.2	Screenshots of SBPG request data of subjects as stored in the database. . .	204
B.3	Software/hardware implementation requests data as stored in the database.	205
B.4	Screenshots of SRE request data for six subjects as stored in database. . . .	205
B.5	Software/hardware implementation requests data for six subjects.	206
C.1	Final results of SBPG request for subject with request ID B2278.	208
C.2	Final results of SBPG request for subject with request ID B7788.	209
D.1	Security requirements for the IoT system of subject with request ID R2143	212
D.2	A detailed report on the result of the request of subject with request ID S2143.	214

List of Tables

3.1	Summary of security requirements for the nine application domains.	68
3.2	A summary of the cryptographic algorithms evaluated by CRYPTREC.	93
5.1	SRE tool performance evaluation form as completed by subjects.	118
5.2	SRE request data of subjects as stored in the database.	118
5.3	Continuation of SRE request data of subjects as stored in the database.	119
5.4	Description of the designations of the 18 columns in Tables 5.2 and 5.3.	120
5.5	A summary of the SRE request for subject with request ID R1995.	126
5.6	A summary of the SRE request for subject with request ID R5432.	126
5.7	A summary of the SRE request for subject with request ID R1287.	127
5.8	SBPG tool performance evaluation form as completed by subjects.	130
5.9	SBPG request data of subjects as stored in the database.	131
5.10	Continuation of SBPG request data of subjects.	132
5.11	Continuation of SBPG request data of subjects.	133
5.12	Description of the designations of the 30 unique columns in Tables 5.9, 5.10, and 5.11.	134
5.13	A summary of the SBPG request for subject with request ID No. B5555.	137
5.14	A summary of the SBPG request for subject with request ID No. B2278.	138
5.15	A summary of the SBPG request for subject with request ID No. B7788.	139
5.16	LWCAR tool performance evaluation form as completed by subjects.	142
5.17	Description of the designations of the 14 columns in Table 5.22.	142
5.18	Description of the designations of the 13 unique columns in Tables 5.23 and 5.24.	143
5.19	Summary of the final results of the LWCAR tool tests for all subjects.	144
5.20	A summary of the SRE request for subject with request ID R2143.	146
5.21	A summary of the LWCAR request for subject with request ID S2143.	146
5.22	LWCAR software implementation requests data as stored in the database.	146
5.23	LWCAR hardware implementation requests data as stored in the database.	149
5.24	Continuation of LWCAR hardware implementation requests data as stored in the database.	149
6.1	Validation of second part of the second research objective.	163

Acronyms and Abbreviations

2FA	2-Factor Authentication
2G	Second Generation
3G	Third Generation
4G	Fourth Generation
5G	Fifth Generation
ABS	Anti-lock Brake System
AD	Associated Data
AE	Authenticated Encryption
AEAD	Authenticated Encryption with Associated Data
AES	Advanced Encryption Standard
AI	Artificial Intelligence
ALMs	Adaptive LogicModules
AMI	Advanced Metering Infrastructure
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
APT	Advanced Persistent Threat
ARM	Advanced RISC Machines
ARP	Address Resolution Protocol
ASIC	Application Specific Integrated Circuit
AVR	Advanced Virtual RISC
BCMSs	Business Continuity Management Systems
BG	Blood Glucose
BLE	Bluetooth Low Energy
BP	Blood Pressure
BYOD	Bring Your Own Device
CBC	Cipher Block Chaining

CCM	Counter-with-CBC-MAC
CCS	Computing Classification System
CCTV	Closed-Circuit Television
CIA	Confidentiality, Integrity and Availability
CLBs	Configurable Logic Blocks
CLI	Command Line Interface
CMAC	Cipher-based Message Authentication Code
CPS	Cyber-Physical Systems
CPU	Central Processing Unit
CRYPTREC	Cryptography Research and Evaluation Committee
CoAP	Constrained Application Protocol
DDS	Data Distribution Service
DDoS	Distributed Denial-of-Service
DES	Data Encryption Standard
DNS	Domain Name System
DPA	Differential Power Analysis
DoS	Denial-of-Service
ECDSA	Elliptic Curve Digital Signature Algorithm
ECG	Electrocardiogram
EDA	Electronic Design Automation
EEG	Electroencephalography
ENISA	European Union Agency for Network and Information Security
ERP	Enterprise Resource Planning
EU	European Union
FELICS	Fair Evaluation of Lightweight Cryptographic Systems
FN	Feistel Network
FPGA	Field Programmable Gate Array

FRs	Functional Requirements
FTP	File Transfer Protocol
GCM	Galois/Counter Mode
GDPR	General Data Protection Regulation
GE	Gate Equivalent
GPOS	General-Purpose Operating System
GPS	Global Positioning System
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HDI	Human-Device Interaction
HDL	Hardware Description Language
HDMI	High Definition Multimedia Interface
HLOS	High-Level Operating System
HMAC	Hash-based Message Authentication Code
HMI	Human-Machine Interface
I/O	Input/Output
I2C	Inter-Integrated Circuit
I2S	Inter-IC Sound
IC	Integrated Circuit
ICT	Information and Communications Technology
ID	Identification
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IIoT	Industrial Internet of Things
IP core	Intellectual Property core
IP	Internet Protocol
IPsec	Internet Protocol Security

ISMSs	Information Security Management Systems
ISO	International Organization for Standardization
IT	Information Technology
ITS	Intelligent Transportation Systems
IoS	Internet of Space
IoT-HarPSecA	IoT Hardware Platform Security Advisor
IoT	Internet of Things
JTAG	Joint Test Action Group
kB	kilobyte
Kbps	Kilobits Per Second
LAB	Logic Array Blocks
LAB	Logic Array Blocks
LBSs	Location-based Services
LEs	Logic Elements
LFSRs	Linear Feedback Shift Registers
LTE	Long-Term Evolution
LUTs	Look-Up Tables
LWC	Lightweight Cryptography
LWCA	Lightweight Cryptographic Algorithm
LWCAR	Lightweight Cryptographic Algorithms Recommendation
M2M	Machine-to-Machine
MAC	Message Authentication Code
MAS	Muti-agent System
MCU	Microcontroller Unit
MD	Merkle-Damgård
MHz	Megahertz
MitM	Man-in-the-Middle

MQTT	Message Queue Telemetry Transport
MSP	Mixed-Signal Processor
NAC	Network Access Control
NFC	Near Field Communication
NFRs	Non-Functional Requirements
NFSRs	Non-linear Feedback Shift Registers
NIST	National Institute of Standards and Technology
NR	Non-Repudiation
NSA	National Security Agency
OCB	Offset Codebook Mode
OEM	Original Equipment Manufacturer
OS	Operating System
OSes	Operating Systems
OTA	Over The Air
OTP	One-Time Pad
OWASP	Open Web Application Security Project
PCB	Printed Circuit Board
PCI	Peripheral Component Interconnect
PETRAS	Privacy, Ethics, Trust, Reliability, Acceptability, and Security
pH	potential Hydrogen
PHY	Physical
PIC	Programmable Interface Controllers
PUF	Physical Unclonable Function
RAM	Random Access Memory
RF	Radio Frequency
RFID	Radio Frequency Identification
ROM	Read-Only Memory

RTOS	Real-Time Operating System
SAM	Secure Access Module
SBC	Single Board Computer
SBPG	Security Best Practice Guidelines
SC	Security Controller
SPI	Serial Peripheral Interface
SPN	Substitution-Permutation Network
SQL	Structured Query Language
SRE	Security Requirements Elicitation
SSH	Secure Shell
SSL	Secure Socket Layer
seccomp	secure computing mode
smart app	Smart Application
SoC	System on Chip
TAP	Test Access Port
TCG	Trusted Computing Group
TCP	Transmission Control Protocol
TEE	Trusted Execution Environment
TFTP	Trivial File Transfer Protocol
TLS	Transport Layer Security
TPM	Trusted Platform Module
UART	Universal Asynchronous Receiver/Transmitter
UAV	Unmanned Aerial Vehicle
UK	United Kingdom
UPnP	Universal Plug and Play
US	United States
USB	Universal Serial Bus

UWB	Ultra-Wide Band
VHDL	Very High Speed Integrated Circuit HDL
VPN	Virtual Private Network
WSNs	Wireless Sensor Networks
Wi-Fi	Wireless Fidelity
XMPP	eXtensible Messaging and Presence Protocol
XOR	Exclusive-OR
XSS	Cross-Site Scripting

Chapter 1

Introduction

1.1 Thesis Focus and Scope

Internet of Things (IoT) basically refers to a network comprising a variety of physical *objects* or *things* with Internet Protocol (IP) addresses or other communication technologies that allow them to connect to one another, as well as to be monitored and/or controlled via the Internet. Sensors are very crucial to the operation of IoT systems [1], therefore, *things* are also embedded with sensors that enable them to sense both physical and chemical parameters from their surroundings. The inputs of the embedded sensors can come from a variety of sources, such as temperature, pressure, light, motion, potential Hydrogen (pH) value (i.e., hydrogen ion concentration), and dangerous gases. These sensors output valuable information that can be shared with other connected *things*. Additionally, data from the sensors can be sent to management systems to be analyzed for trends that can provide the necessary information needed for making informed decisions. Hence, an IoT device or a *thing* is anything embedded with a sensor or an actuator that connects to a network, share data with other devices in a network and transmits data over the Internet.

Essentially, IoT is extending Internet connectivity beyond traditional devices such as computers, tablets, and smartphones to a diverse range of everyday consumer products like refrigerators and security cameras, thereby changing the world as we know it. Thus, the prediction of Kevin Ashton made about 20 years ago that “IoT has the potential to change the world” [2, 3], which once seemed like science fiction is becoming a reality. Today, IoT technology is already changing the way people live and work by improving productivity and efficiency in industries and organizations, as well as providing a higher level of comfort to individual users. Some areas where these changes are already becoming visible include wearable, healthcare, manufacturing, transportation, urban areas, connected cars, home appliances, and a myriad of other examples [4].

The number of Internet-connected devices is growing at an accelerated pace [5], such that IoT is increasingly becoming the hub of connectivity enabling the exchange of data from *things*, people, and processes. Consequently, many people and enterprises are adopting the paradigm due to its huge potential benefits and impact on their lives and businesses, respectively [6]. For example, many of the industry leaders such as Google, Apple, Microsoft, and Samsung have launched their own IoT platforms which are fully operational [7]. The applications of IoT are enormous; they include smart environmental monitoring, smart farming, and smart grid. New application domains where IoT is cre-

ating new revenue sources, new business models, and a myriad of business opportunities include retail, logistics and supply chain, product flow monitoring, and inventory management. In addition, IoT is tightly interwoven with Cyber-Physical Systems (CPS), and thus a key enabler of Industry 4.0, also known as the fourth industrial revolution [8].

Another new IoT application area that is currently revolutionizing the industrial space is the Industrial Internet of Things (IIoT), which basically brings Internet connectivity to manufacturing industries. Besides improved efficiency and productivity, IIoT benefits include a considerable reduction in cost and losses for companies. A promising use case for IIoT is predictive maintenance [9], which will allow companies to identify potential failures and avoid costly downtime. By integrating Artificial Intelligence (AI) capabilities such as machine learning and deep learning into the IIoT, more complex problems in industries can be solved [10], including reducing carbon emissions and other pollution problems, such as oil spills in the oil gas industry.

1.1.1 Motivation

There have been several growth forecasts, statistics, and encouraging market estimates from a variety of sources, and mostly by different industry analysts [11], which indicate a promising future for the IoT [12]. While statistics and figures may vary across different studies, the consensus is that IoT holds significant potential in the Information Technology (IT) sector and that its future prospects are both encouraging and promising. Notwithstanding the current trends and future prospects, to prove its worth, IoT has to overcome many challenges like other disruptive technologies, such as the Internet [54] and cell phone [55, 56]. Accordingly, despite its numerous potential benefits, some of which have been highlighted above, IoT is facing many challenges that are limiting its widespread adoption.

Surveying the current IoT landscape reveals numerous challenges and issues facing the technology [13]. Moreover, while new wireless standards such as the new Fifth Generation (5G) cellular network and the new Wireless Fidelity (Wi-Fi) 6 (i.e., IEEE 802.11ax) are expected to address some of the issues such as improving connectivity and fostering deployment, they also come with their own challenges. The rolling out of the 5G network, for example, comes with additional challenges including frequency bands allocation issues, cost of deployment, and physical layer issues [14, 15]. While IoT challenges are mostly technology and business-based, societal acceptance is another important requirement for widespread commercial deployment of the IoT technology [16]. The following are a few of the key technological challenges that IoT must overcome to become a reality and achieve universal acceptance:

- **Heterogeneity:** In the concept of IoT, *things* are expected to exchange information and data seamlessly without human intervention. However, IoT is currently

characterized by a large heterogeneity in terms of networks and devices taking part in data exchange, resulting in very different capabilities from the communication and computational standpoints. Diverse connected devices and Smart Applications (smart apps) pose conflicting requirements and challenges that must be resolved to fully realize the benefits of the IoT. The incompatibility between the Physical (PHY) layers of Bluetooth Low Energy (BLE) and Wi-Fi is one example that shows how managing such a high level of heterogeneity presents a major challenge in the operation of the IoT ecosystem [17];

- **Interoperability:** To achieve seamless connectivity in the IoT, devices need to communicate through common communication protocols, therefore, interoperability is a crucial requirement. Nonetheless, IoT is currently an environment where devices have different data representations, propriety protocols, and heterogeneous Application Programming Interfaces (APIs), which constitute a challenge to interoperability among devices and smart apps [18];
- **Scalability:** Adapting to changes in the environment such as the ability to support expansion as the need arises is an important attribute for an IoT system. While this is a desirable feature, scalability may pose some challenges like connectivity problems to large scale deployment and expansion of IoT systems. For example, providing uplink connectivity to very large numbers of connected devices may create problems for some cellular networks [19];
- **Limited energy resource:** Powering the edge devices of the IoT, such as sensors, actuators, Global Positioning System (GPS) receivers, and cameras pose a significant problem for many IoT deployments [20]. This particularly becomes important when one considers that such devices may number in the billions. Furthermore, some edge devices may be used in very remote areas where getting power sources can be a real challenge. What further complicates the issue is that some edge devices may be embedded into concrete infrastructures, or buried undersea, making battery replacement extremely difficult, or unfeasible;
- **Data Security:** In this era of intelligent connectivity, connected devices, and smart apps in consumer and enterprise spheres produce huge amounts of data that keeps growing at an ever-increasing rate. Because IoT and data are intrinsically linked together, and the fact that many connected devices and smart apps are susceptible to cyberattacks, they become goldmines and attractive targets for malicious actors. Thus, these devices and smart apps expose organizations and consumers to new security vulnerabilities introduced by the IoT [21]. In addition, while it is expected that the 5G and Wi-Fi 6 technologies, which promise very high transfer rates; low power requirements; and high bandwidth, will open a new page in the wireless arena with new use cases for the IoT [22], they will also increase the attack surface for cybercriminals which means that IoT security challenges will only get bigger and more significant [23];

- **User privacy:** Many IoT systems can provide customized services that may require a good understanding of user preferences and interests, daily activities, and behavior patterns. Examples of such smart devices include smart wearable and implantable medical devices, such as fitness trackers and pacemakers, respectively. These devices can track daily activities of users including exercise, sleep, and heart rate, and hence capture huge amounts of data that is transmitted over the Internet to be stored, processed, and analyzed in IoT cloud platforms. Unfortunately, such data can directly or indirectly reveal a variety of sensitive and private user information, such as name, location, credit card number, and social security number, and thus expose users to different types of privacy attacks [24].

The scope of the research study described in this thesis falls within the fields of IoT and systems security. As such, this thesis is focused on investigating and attempting to address one of the root causes (to be defined in Section 1.2) of the last two key challenges of the IoT mentioned above, namely data security and user privacy. It specifically focuses on the design and development of a security framework aimed at providing security support to non-security experts actively involved in the design and implementation of IoT devices and smart apps. The scope of this thesis falls under the following topics in the 2012 version of the ACM Computing Classification System (CCS), the de facto standard for Computer Science:

- **Security and privacy~Systems security;**
- **Computer systems organization~Embedded and cyber-physical systems;**
- *Software and its engineering~Software creation and management;*
- *Hardware~Emerging technologies;*
- Human-centered computing~Ubiquitous and mobile computing.

1.2 Problem Definition

IoT security and privacy issues are numerous and encompass the different non-standardized layers of abstraction of the IoT [25], ranging from the bottom layer, also known as the perception or physical layer to the topmost layer, otherwise known as the application layer. While cyberattacks have been around for a very long time, what is new is the relative simplicity of attacks in the IoT [26] and the scale [27]. For example, attackers can use compromised IoT devices connected to home or corporate networks to launch major attacks on mission-critical systems or applications. In *botnet* or *thingbot* attacks, for example, cybercriminals use a network of compromised IoT devices to take down networks, IT environments or important websites using Distributed Denial-of-Service (DDoS) attacks. Other malicious activities that can be carried out using *botnets* include sending

spam and phishing emails, exploiting online-banking data, and stealing private information [28].

What makes IoT security and privacy issues so critical is the fact that a large number of the edge devices, which account for the majority of the IoT end nodes, are constrained in terms of resources such as energy, memory, and computational capabilities. These inherent limitations associated with many IoT devices necessitate new and holistic approaches since many traditional security methods cannot be directly applied to secure IoT systems [29]. Apart from the previously mentioned device constraints in terms of resources, other factors that make IoT security and privacy challenges unique include:

- **Low cost of devices:** Implementing security algorithms on highly resource-constrained devices is a big challenge due to the memory and computational capability requirements [30], which may significantly add to the cost of the devices. Nonetheless, most IoT edge devices such as sensors need to be cheap and disposable, this is coupled with the fact that manufacturers need to offer attractive prices to compete in the market. As a result, manufacturers are faced with the challenge of managing the trade-off between cost and sufficient device resources;
- **Some devices are left exposed and unattended:** In some applications, IoT edge devices are deployed in difficult terrains where they are left exposed and unattended for years [31]. In such scenarios, malicious attackers can physically or logically tamper with poorly protected devices;
- **Outside of enterprise perimeter security:** While perimeter security is traditionally used to protect computer systems, using the traditional perimeter security alone is not enough to protect a network consisting of IoT devices. This is because connecting IoT devices directly to an enterprise network can take a critical cyber asset outside its perimeter boundary since a vulnerability in a single IoT device can create a security hole in the perimeter defense. In addition, legacy approaches such as Network Access Control (NAC), Virtual Private Network (VPN), and firewalls are subject to vulnerability, and hence can be exploited by attackers [32];
- **Devices may be unsupported by manufactures:** Some IoT devices may outlive the companies that manufactured them [33], therefore, such devices will no longer receive security updates. While these orphan devices may still be connected, they will be left unpatched, and hence susceptible to cyberattacks [34];
- **Difficult to update:** Some IoT devices have no update mechanisms. Even if such mechanisms exist and the updates that can patch the uncovered security flaws are available, updating a large number of devices can be a real challenge [35]. Moreover, even when security researchers discover vulnerabilities in IoT consumer products and patches are released by the affected companies, users may be reluctant to update their smart devices due to lack of security awareness. Furthermore, users may

not be able to manually carry out such tasks on some smart devices due to lack of appropriate user interfaces;

- **Lack of sufficient security experience on the part of some manufacturers:** As smart sensing and connectivity technologies are increasingly embedded in consumer products and smart apps are fast becoming the new business platform, there is an exponential growth in the number of IoT software and hardware start-up companies around the world. Although this can be considered as a boost for innovation in the IoT, some of these companies do not give cybersecurity the priority it deserves [36]. Therefore, they produce IoT devices and smart apps with security vulnerabilities. One of the root causes of the problem is that some of these start-ups are composed of electronics or computer engineers and/or developers with little or no security expertise [37]. Another issue is that some companies that were producing traditional consumer products such as lightbulbs and toasters have suddenly become IoT companies. Some of these companies simply add sensors and Internet connection widgets to their products without understanding the security ramifications and unintended consequences of doing so. These issues constitute a major challenge in the development of secure IoT devices and smart apps.

This thesis aims to investigate and address the issues raised in the last item in the list of IoT security and privacy challenges enumerated above. These are critical issues aggravated by a lack of globally applicable baseline for IoT security that manufacturers can use across all IoT application domains [38], which must be addressed quickly. Accordingly, the need to embed security into the design and development of IoT devices and smart apps has become quite apparent in recent years due to the noticeable effects of cyberattacks related to IoT products and services. The healthcare and manufacturing domains, for example, have suffered substantial losses due to vulnerabilities in IoT devices and smart apps [39], stressing the need to prioritize security and address it from the very beginning.

1.2.1 Research Objectives

As follows from the foregoing discussion, *security-by-design* must be a crucial concept that should underpin the design and development of IoT devices and smart apps.

Definition of Security-by-Design in IoT: In the context of IoT, security-by-design is an approach that seeks to embed security aspects into IoT products and applications from the outset, which will make them as free of vulnerabilities and resistant to attacks as possible [40, 41]. This encompasses performing security engineering, integrating, and testing security technology in IoT systems.

In line with the concept of *security by design*, this thesis presents a security framework intended to help designers, engineers, developers, and electronics hobbyists with little or no security expertise to design and develop secure IoT devices and smart apps. The

usage and benefits of some features of such a framework may extend beyond the IoT to include other computer systems. The research work described in this thesis has two main objectives:

- Firstly, it is intended to explore security and privacy issues in the IoT by providing some insights into the research being done in this area, including current challenges and requirements, as well as to highlight the need to bake security and privacy into the design and development of IoT devices and smart apps from the very beginning;
- The second objective is to propose, design, and implement a prototype of an efficient and easy-to-use security framework that is intended to serve as an advisor that can foster the design and development of secure IoT devices and smart apps, as well as to evaluate the performance and usability of the security advisor framework.

1.3 Thesis Statement

Cyber attacks and other security and privacy risks associated with connected devices have led to growing concerns about security and privacy in the IoT. As a consequence, there is an ongoing debate about who should take responsibility for IoT security and privacy: device manufacturers and developers or governments? However, when viewed in its proper perspective, this issue will arguably come down to how device makers and developers are approaching IoT development as a whole, and particularly, how the concept of security by design is often neglected by many device makers and developers [42]. This issue has been raised in Section 1.2, as one of the factors that make IoT security and privacy challenges unique.

While a number of security vulnerabilities have been discovered in some smart consumer products made by big technology industry giants, today most security vulnerabilities in the IoT are being discovered in IoT products produced by small start-up companies [43, 33]. In addition to lack of security expertise already highlighted in Section 1.2, another possible reason for this could be that many of such companies are unknown, and thus have no brands or reputations to protect [44]. This constitutes a major challenge in providing proper security and privacy protection in the IoT. Given that not many of these start-up companies can afford to hire security experts and the fact that it will be difficult to stop them from producing IoT products and applications, there must be a way to help such companies to produce secure IoT products if the vulnerabilities, risks, and other security and privacy issues associated with IoT systems are to be mitigated. Accomplishing this task is one of the main objectives of the research work described in this thesis as outlined in Subsection 1.2.1, and hence the following thesis statement is therefore developed:

A security framework that relies on security engineering activities to facilitate the incorporation of security into the initial design and development process of IoT systems

may offer a path forward to achieve a more secure IoT ecosystem. Such a framework may be more beneficial to non-security experts who are actively involved in the design and development of IoT systems.

The following section presents the research steps taken to address the research problem mentioned in Section 1.2; the research plan also represents a step-by-step approach to validating the aforementioned thesis statement.

1.4 Adopted Approach and Research Plan

The following steps describe the approach taken to address the aforementioned problem. The steps also represent the research plan designed to achieve the stated objectives within the scope of this Ph.D. research study, as presented in Subsection 1.2.1. In addition, this research plan also represents the steps for validating the thesis statement presented in Section 1.3:

1. **Survey:** The first step in the approach began with an exhaustive survey of the state-of-the-art in IoT security and privacy, which covered several application domains. The survey performed an extensive examination of several existing mechanisms and approaches for ensuring security and privacy in the IoT;
2. **Problem Identification:** After a careful review of the existing literature in IoT security and privacy, the author has identified and outlined many problems as open issues, some of which are presented in Section 1.2. Having studied the problems, this thesis is focused on the last problem presented in Section 1.2, which, arguably, is one of the problems that require immediate attention, namely *lack of sufficient security experience on the part of manufacturers*;
3. **Study of Different Aspects of Security Implementation in the IoT:** To address the identified problem, the author has studied different aspects of IoT security and privacy implementations. This study covered IoT security requirements in different application domains, IoT security best practices, IoT hardware development platforms, lightweight cryptography, and implementation of lightweight cryptographic algorithms in constrained environments;
4. **Framework Design Thinking Process:** The concept of the security framework was born from the aforementioned studies. This was preceded by analyzing the identified problem, which involved breaking down the problem into smaller and easier-to-understand constituents;
5. **Framework Design and Implementation:** This step involved the design of each of the three components of the security framework, the implementation of the tool that allows users to interact with the framework, as well as testing and fine-tuning of the algorithms;

6. **Framework Performance Evaluation:** In this step, the tool which is the realization of the security framework was tested and evaluated. It involved performing a number of functionality and usability tests by different groups of subjects.

1.5 Main Scientific Contributions

The Ph.D. work described in this thesis represents one of the first attempts to provide support to non-security experts involved in the design and development of IoT devices and applications. The work provided a number of scientific contributions, some of which are presented below:

- **An Extensive Survey on IoT Security and Privacy:** The first contribution of this research work is a detailed and comprehensive survey of the state-of-the-art in IoT security and privacy. Some parts of this survey are presented in Chapter 2 of this thesis. The survey provides a detailed description of different aspects of IoT security and privacy and other aspects of IoT in general, including a description of several application domains and identification of cyber assets per domain. This contribution is reported in the first article [45] in the first list of publications presented in the following subsection;
- **Identification of Security Requirements and Security Threats in IoT Architecture and Industry Domains:** The second contribution is the identification of several security requirements and security threats in IoT architecture and in different IoT industry domains, as well as the provision of various possible countermeasures to address the identified security threats. The first part of this contribution is reported in the fifth and seventh articles [52, 46] and the other part in the conference paper [47] in the third item of the first list of publications presented in Subsection 1.5.1;
- **Identification of Security Requirements in IoT Application Domains:** The third contribution is the identification of security requirements in several application domains, including smart healthcare, smart grid, smart city, smart home, smart manufacturing, smart supply chain, smart farming, and smart transportation. This contribution is reported in both the first and third papers [45, 47] in the first list of publications outlined in Subsection 1.5.1;
- **An Extensive Review of Important Attributes of Several IoT Hardware Development Platforms:** The fourth contribution includes an extensive review that examines several IoT hardware development platforms that were released in the past, those that are recently launched on the market, and those that will be released in the near future. It focuses specifically on some essential attributes of the hardware development platforms that include processing speed, memory capacity,

battery life, and security features. It also includes step-by-step best practice guidelines for designing and prototyping IoT projects. This contribution is reported in the sixth and eighth articles [48, 49] in the first list of publications presented in Subsection 1.5.1;

- **A Comprehensive Study of IoT Security and Privacy Best Practices:** The fifth contribution includes an extensive study of challenges of developing a commonly agreed IoT security and privacy best practices, as well as the exploration of various attempts to develop more widely accepted security and privacy best practices for the IoT. This contribution is reported in the tenth article in the first list of publications presented in Subsection 1.5.1;
- **Software and Hardware Design and Implementation Considerations:** The sixth contribution is a detailed discussion of software and hardware design and implementation considerations for Lightweight Cryptographic Algorithms (LWCAs). This contribution is reported in the second article [43] in the first list of publications outlined in Subsection 1.5.1;
- **Design and Implementation of IoT-HarPSecA Framework:** The seventh contribution is the design and implementation of the three components of the security framework, as well as a detailed evaluation of the performance of the main component of the IoT-HarPSecA tool. This contribution is reported in both the second and fourth papers [43, 50] in the first list of publications presented in Subsection 1.5.1;
- **Development of a Human Expert Knowledge Representation:** The eighth contribution is the development of human expert knowledge representation in the form of decision-making procedures of experts in the fields of IoT and Lightweight Cryptography (LWC). This contribution is also reported in the second article [43] in the first list of publications outlined in Subsection 1.5.1;
- **A Performance and Usability Evaluation of the IoT-HarPSecA Framework Tool:** The last contribution of this thesis is the evaluation of the performance and usability of the three components of the IoT-HarPSecA framework tool. This contribution is reported in both the second, ninth, and tenth articles [43, 51] in the first list of publications presented in the following subsection.

1.5.1 Publications

Within the scope of this Ph.D. work, the following articles have been published in journals, conference proceedings, and peer-reviewed books; except for the last article in the first list, which is currently under review at the Elsevier Journal of Computer Networks:

1. **Challenges of Securing Internet of Things Devices: A survey** [45]
Musa G. Samaila, Miguel Neto, Diogo A. B. Fernandes, Mário M. Freire and Pedro

R. M. Inácio, *Security and Privacy*, vol. 1, issue 2, pp. 1-32, May 2018.
DOI: <https://doi.org/10.1002/spy2.20>

2. **IoT-HarPSecA: A Framework and Roadmap for Secure Design and Development of Devices and Applications in the IoT Space** [43]
Musa G. Samaila, João B. F. Sequeiros, Tiago Simões, Mário M. Freire and Pedro R. M. Inácio, *IEEE Access*, vol. 8, pp. 16462-16494, January 2020.
DOI: 10.1109/ACCESS.2020.2965925
3. **Security Threats and Possible Countermeasures in IoT Applications Covering Different Industry Domains** [47]
Musa G. Samaila, João b. F. Sequeiros, Mário M. Freire and Pedro R. M. Inácio, *In ARES 2018: International Conference on Availability, Reliability and Security*, August 27–30, 2018, Hamburg, Germany. ACM, New York, NY, USA, Article 16, 9 pages.
DOI: 10.1145/3230833.3232800
4. **IoT-HarPSecA: A Framework for Facilitating the Design and Development of Secure IoT Devices** [50]
Musa G. Samaila, Moser Z. V. José, João B. F. Sequeiros, Mário M. Freire and Pedro R. M. Inácio, *In Proceedings of the 14th International Conference on Availability, Reliability and Security (ARES 2019) (ARES '19)*, August 26–29, 2019, Canterbury, United Kingdom. ACM, New York, NY, USA, Article 4, 7 pages.
DOI: 10.1145/3339252.3340514
5. **Security Challenges of the Internet of Things** [52]
Musa G. Samaila, Miguel Neto, Diogo A. B. Fernandes, Mário M. Freire and Pedro R. M. Inácio, in *Beyond the Internet of Things: Everything Interconnected*. Jordi Mongay Batalla, George Mastorakis, Constandinos X. Mavromoustakis, Evangelos Pallis (Eds.), *Springer International Publishing*, ISBN: 978-3-319-50756-9, pp. 53-82, 2017.
DOI: 10.1007/978-3-319-50758-3_3 53
6. **IoT Hardware Development Platforms: Past, Present, and Future** [48]
Musa G. Samaila, João B. F. Sequeiros, Acácio F. P. P. Correia, Mário M. Freire and Pedro R. M. Inácio, in *Internet of Things: Challenges, Advances, and Applications*. Qusay F. Hassan, Atta ur Rehman Khan, Sajjad A. Madani (Eds.), *CRC Press*, ISBN: 13: 978-1-4987-7851-0, pp. 107-139, 2017.
7. **A Quick Perspective on the Current State of IoT Security: A Survey** [46]
Musa G. Samaila, João B. F. Sequeiros, Acácio F. P. P. Correia, Mário M. Freire and Pedro R. M. Inácio, in *Networks of the Future: Architectures, Technologies, and Implementations*. Mahmoud Elkhodr, Qusay F. Hassan, Seyed Shahrestani (Eds.), *CRC Press*, ISBN: 13: 978-1-4987-8397-2, pp. 431-464, 2017.

8. **A Tutorial Introduction to IoT Design and Prototyping with Examples** [49]
Manuel Meruje, Musa G. Samaila, Virginia N. L. Franqueira, Mário M. Freire and Pedro R. M. Inácio, in *Internet of things A to Z: Technologies and Applications*. Qusay F. Hassan (ed.), *Wiley-IEEE Press*, ISBN: 978-111-945674-2, pp. 153-190, 2018.
DOI: <https://doi.org/10.1002/9781119456735.ch6>
9. **A Preliminary Evaluation of the SRE and SBPG Components of the IoT-HarPsecA Framework** [51]
Musa G. Samaila, Carolina Lopes, Édi Aires, João B. F. Sequeiros, Tiago Simões, Mário M. Freire and Pedro R. M. Inácio, *In Proceedings of the 2020 Global Internet of Things Summit (GIoTS)*, June 3-3, 2020, Dublin, Ireland. IEEE, 7 pages.
DOI: 10.1109/GIOTS49054.2020.9119590
10. **Performance Evaluation of the SRE and SBPG Components of the IoT-HarPsecA Framework**
Musa G. Samaila, Carolina Lopes, Édi Aires, João B. F. Sequeiros, Tiago Simões, Mário M. Freire and Pedro R. M. Inácio, *Elsevier Journal of Computer Networks* (under review).

Other related papers published within the period of this research work include:

1. **Attack and System Modeling Applied to IoT, Cloud, and Mobile Ecosystems: Embedding Security by Design** [53]
João b. F. Sequeiros, Francisco T. Chimuco, Musa G. Samaila, Mário M. Freire and Pedro R. M. Inácio, *ACM Computing Surveys*, vol. 53, no. 2, Article 25, pp. 1-32, March 2020.
DOI: 10.1145/3376123

1.6 Thesis Organization

The remainder of this doctoral thesis is organized into five chapters, which are outlined as follows:

- Chapter 2 provides an overview of some important concepts in IoT security and privacy. Section 2.1 presents the description of a three-layer IoT architecture, security requirements in IoT architecture, and security threats in the IoT architecture. Section 2.2 describes nine application domains. Section 2.3 presents system models and threat models for the nine application domains, as well as discusses IoT threat landscape. Section 2.4 discusses some fundamental IoT security and privacy issues, presents a number of performance versus security trade-offs, and in the end, it provided some possible security and privacy countermeasures. Section 2.5 presents a

brief overview of IoT hardware development platforms. The purpose of these sections is to provide the reader with an overview of current state of IoT security and privacy and the associated issues;

- Chapter 3 discusses the underlying concepts and principles behind the IoT Hardware Platform Security Advisor (IoT-HarPSecA) framework. Section 3.2 discusses security requirements in the IoT, which provides the basis upon which the first component of the security framework is built. Section 3.3 discusses IoT security and privacy best practices, which is the backbone of the second component of the framework. Section 3.4 presents an overview of LWC and LWCAs, which serve as essential bedrocks of the third component of IoT-HarPSecA framework;
- Chapter 4 presents the design and implementation of the three components of the IoT-HarPSecA framework. Section 4.2 presents the design and description of the Security Requirements Elicitation (SRE) component. Section 4.3 focuses on the design and description of the Security Best Practice Guidelines (SBPG) component. Section 4.4 presents the design and description of the Lightweight Cryptographic Algorithms Recommendation (LWCAR) component. Section 4.5 discusses the implementation of the tool that can be used to interact with the IoT-HarPSecA framework;
- Chapter 5 presents results of functionality and usability tests performed on the SRE, SBPG, and LWCAR components of the IoT-HarPSecA framework tool. The chapter also provides discussions on the evaluation of the results. Section 5.1 provides background information about the various tests conducted and outlines the structure of the chapter. Sections 5.2, 5.3 and 5.4 presents the tests, results, and performance evaluation of the SRE, SBPG and LWCAR tools, respectively;
- Finally, Chapter 6 summarizes the important contributions of this research work. Section 6.1 provides a summary of the main contributions of this thesis. Section 6.2 assesses the achievement of the research objectives defined in Subsection 1.2.1 to ascertain whether or not the research objectives are met; it also corroborates the thesis statement presented in Section 1.3. Section 6.3 presents the final conclusions of this thesis which summarizes the work done in this research study. Section 6.4 highlights a few of the challenges faced by the author during the course of this research work and outlines some limitations of this work; this section also outlines possible directions for future work.

Chapter 2

State-of-the-Art

This chapter describes important concepts in IoT security and privacy after presenting some basic concepts of the IoT paradigm. In Section 2.1, the chapter begins with the description of a three-layer IoT architecture, security requirements in IoT architecture, and security threats in the IoT architecture. Afterward, Section 2.2 describes nine application domains. Section 2.3 presents system models and threat models for the nine application domains, as well as discusses IoT threat landscape. Moreover, Section 2.4 discusses some fundamental IoT security and privacy issues, presents a number of performance versus security trade-offs and, in the end, it presents some possible security and privacy countermeasures. Furthermore, Section 2.5 provides a brief overview of IoT hardware development platforms. Finally, Section 2.6 concludes the chapter. This chapter is based on publications 1, 3, 5, 6, 7, and 8 [45, 47, 52, 48, 46, 49] enumerated in Subsection 1.5.1.

2.1 IoT Architecture

The large variety of heterogeneous networks and devices has made building a general architecture for the IoT a very complex task [57]. Nonetheless, IoT architecture can generally be divided into three distinct layers, namely the perception layer (also referred to as the recognition or physical layer), the network layer, and the application layer [58, 59, 60], as shown in Figure 2.1. The perception layer is responsible for gathering all kinds of data from the physical world using physical end devices, such as Radio Frequency Identification (RFID) tags and readers, cameras, GPS receivers, and all sorts of sensors [61, 62]. The network layer, which is in the middle, encompasses different protocols and communication network technologies, which serve as access networks [62]. This layer is also responsible for the assortment of data, initial processing, and transmission of data [63]. The topmost layer is the application layer, which provides support for business services and different kinds of personalized services to individual users [64].

The application layer is sometimes depicted as consisting of application support or middleware and application segments. Application layer protocols include Constrained Application Protocol (CoAP), Message Queue Telemetry Transport (MQTT), eXtensible Messaging and Presence Protocol (XMPP), Advanced Message Queuing Protocol (AMQP), and Data Distribution Service (DDS). Using the application layer interface, users can access the IoT via computers, mobile devices such as smartphones and tablets. Depending on the services, other devices like smart refrigerators, smart televisions, etc., can also be used. The network layer is critical because it serves as the link between the perception

and the application layers [63]. Long-range communication can be achieved using IP based Internet, such as Second Generation (2G), Third Generation (3G), Fourth Generation (4G), Long-Term Evolution (LTE) or 5G network, while IEEE 802.15.4 (ZigBee), Z-Wave, Thread, Ultra-Wide Band (UWB), BLE and Near Field Communication (NFC) are used for short-distance communications among the IoT devices due to limited bandwidth, lossy connections, intermittent links, and limited power constraints [65].

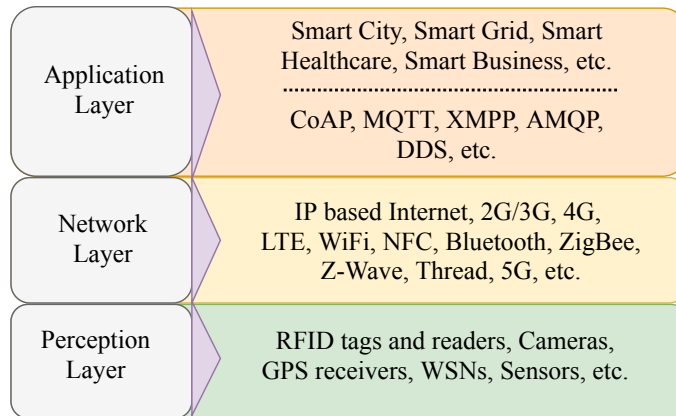


Figure 2.1: Basic IoT architecture (adapted from [45]).

2.1.1 Security Requirements in IoT Architecture

As each connected device could be a potential doorway into IoT systems, the design process of an IoT system should meet stringent security requirements for a particular application. The necessary or basic security services required in IoT systems and networks are confidentiality, data integrity, authentication, replay protection, and availability [66]. But there is no universal or single security mechanism or solution that can guarantee security in all the three layers of the IoT security architecture. This is because each layer presents unique security requirements, which are usually different from those of traditional computer system architectures mainly due to energy, computational, connectivity, and storage limitations of many IoT devices. Based on the discussion on IoT security architecture presented in Section 2.1 the following subsections briefly present the security requirements for each of the three layers. It should be noted, however, that while some applications may not need all of these security requirements, some applications may need other specific security requirements in addition to some or all of the ones presented below.

2.1.1.1 Sensing Layer Security Requirements

Sensing layer security requirements vary depending on the application and the type of devices that are deployed for data collection; they include:

- *Data Confidentiality* is necessary to prevent unauthorized access to the data generated by IoT devices such as smart sensor;

- *Authenticity* is needed to impose restrictions on logical access to IoT devices and sensitive information;
- *Data Integrity* is necessary to ensure the correctness, accuracy, and validity of data from IoT end nodes;
- *Availability* is needed to ensure that IoT resources like real-time data from end nodes or other services are readily available and can be accessed by authorized users at any time;
- *Device Resilience* is an important security requirement that enables IoT end nodes to adjust self to handle failures and avoid a single point of failure;
- *Physical Security* is essential in many applications to protect IoT devices from tampering. For example, some sensor nodes may be deployed in an environment that is open to different adversaries. Such sensors may be left unattended for a very long time, hence they can be easily tampered with;
- *Tamper Resistance and Detection* are needed to ensure that exposed IoT devices incorporate physical protection to prevent attackers from compromising cryptographic parameters and that any active attempt to compromise the integrity of an IoT device or the data associated with it is detected;
- *Threat Hunting* is an important security requirement that can ensure proactive searching of cyber threats that may be lurking undetected in an IoT device or network, which may be quietly collecting login credentials or other confidential material [67];
- *Reliability* of devices at the edge layer is essential to guarantee the consistent intended behavior of the IoT edge devices.

2.1.1.2 Network Layer Security Requirements

The following are important security requirements in the network layer:

- *Authenticity* is an essential requirement that will prevent unauthorized entities from accessing information from the network;
- *Authorization* is needed to prevent malicious users from having undue privileges that may allow them to endanger other users on the network;
- *Network-level encryption* which is implemented above the data link layer, is a requirement that employs cryptographic services for encrypting data in transit and can be achieved using Internet Protocol Security (IPsec);
- *Data Integrity* ensures that every piece of data being transmitted on an IoT network is not manipulated en-route;

- *Availability* is the requirement that determines the need for an IoT network and its services to be accessible when needed by authorized users, devices, or applications, and despite other mechanisms used to maintain confidentiality and integrity;
- *Attack detection, prevention or mitigation* schemes are needed to detect, prevent or mitigate the effects of network intrusions. Some methods include using anti-DDoS and reliable system updates;
- *Network Resilience* ensures that an IoT network continues to operate in the face of targeted attacks, natural disasters, or faults;
- *Anomaly detection* is an important security requirement that provides an approach to identifying or detecting unexpected security breach before it happens. This usually relies on the detection of security threats in IoT networks based on packet signatures that differ from the norm;
- *Reliability* is needed to ensure that IoT network components operate consistently as intended over the entire lifecycle of the given IoT system.

2.1.1.3 Application Layer Security Requirements

Application layer security requirements or properties include the following:

- *Authenticity* is essential to ensure that information transaction is from the source it claims to be from;
- *Authorization* is needed to ensure that users or devices have rights and privileges to access a resource;
- *Availability* ensures that IoT devices or smart apps are accessible and usable upon demand by authorized users or entities;
- *Secure API* are necessary to ensure that every data movement between smart devices, back-end systems, and smart apps using REST-based APIs are well authenticated and authorized, and that request timestamp is added to prevent basic replay attacks;
- *User privacy protection* refers to the control of the user over the disclosure of his/her sensitive information. It is an important security requirement that will ensure trust in the IoT. If privacy is taken seriously, users can have confidence that their sensitive private information is well protected;
- *Non Repudiation* is needed to ensure that the transfer of messages or credentials between two IoT entities cannot be denied by the participating parties;
- *Accountability*: ensures that every action can be traced back to a single user or entity;

- *Reliability* is necessary to guarantee the consistent intended behavior of an IoT system;
- *Information Forensics* is another important property that ensures that IoT devices and smart apps are supported by existing digital forensic methods and tools without endangering the privacy of users;
- Considering the amount of data that is being stored and processed in the cloud, a secure cloud environment is a crucial requirement;
- Training and creating awareness among users on the importance of Security education constitute a critical requirement. This, among other things, will enable users to know how to choose, keep and manage their passwords. Additionally, users should be taught to desist from giving security a secondary consideration.

2.1.2 Security Threats in IoT Architecture

Figure 2.2 shows a generic security architecture consisting of the three layers. The figure also shows different IoT components in each layer, mapping of IoT protocols to the Transmission Control Protocol (TCP)/IP model, some typical threats associated with each layer, and some possible security mechanisms.

2.1.2.1 Security Threats in the Perception Layer

Devices in the perception layer are often constrained in terms of resources such as energy, memory, processing capability, and often exhibit low data rate since they usually operate within an unreliable wireless environment [68]. For instance, sensing nodes are usually small in size and constrained in terms of computational and storage capacity and rely on finite energy sources, such as small batteries. Consequently, security features on some of these sensing nodes are very limited. Moreover, providing physical security for such devices in some applications would be difficult. In the environmental monitoring domain, for example, providing physical security to sensor nodes is a major challenge. While some sensing nodes may have basic security features like usernames and passwords, others may not have any at all. In addition, virtually all communications such as interconnection between devices in the sensing network are via Radio Frequency (RF) which leaves the window open to different kinds of threats. Consequently, it is difficult to set up a reliable security protection mechanism on the sensing network since most security mechanisms like public-key encryption algorithms tend to be resource-intensive. Devices in this layer are prone to different types of security threats, including tampering or unauthorized access to devices, eavesdropping, Radio Frequency (RF) jamming, tag cloning, spoofing, and Denial-of-Service (DoS) attacks.

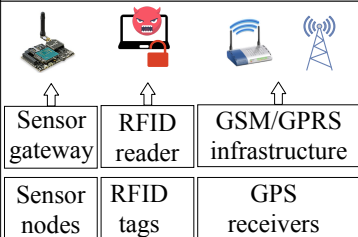
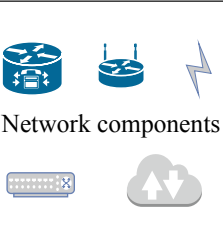
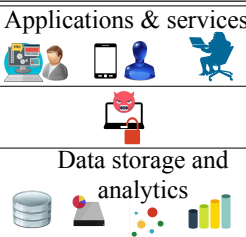
Layer	Perception	Network	Application
Description	Intelligent perception terminal	Internet, and different wireless networks	Web or End-user services
Components			
IoT protocols mapped to TCP/IP model	Network access & physical Wireless (Wi-Fi, GSM/GPRS, GPS, CDMA, 3G, 4G, LTE, 5G, BLE, NFC, etc.), Wired (Ethernet)	Transport TCP, UDP Internet IPv6, 6LoWPAN, RPL, etc.	Application HTTP, HTTPS, MQTT, AMQP, XMPP, DSS, CoAP, etc.
Threats	Tampering, unauthorized access to devices, eavesdropping, RF jamming, tag cloning, spoofing, DoS, etc.	Exhaustion of network resources, spoofing, jamming signals, sinkhole, counterfeit, MitM attacks, session hijacking, DoS & DDoS attacks, etc.	DoS & DDoS attacks, malicious code injection, sniffing, phishing, social engineering, etc.
Possible security mechanisms	Anonymity approaches, use of hash algorithms to validate data integrity, intrusion detection, lightweight encryption mechanisms, risk assessment, etc.	Routing control security mechanism, intrusion detection, lightweight encryption mechanisms, data integrity check with a shared private key, etc.	Smart firewalls, encryption mechanisms, intrusion detection, identity management techniques for controlling user access to a resource, risk assessment, etc.

Figure 2.2: A generic security architecture for IoT (adapted from [47]).

2.1.2.2 Security Threats in the Network Layer

This layer consists of a number of different communication technologies, including, but not limited to, Wi-Fi, 3G, 4G, LTE, and 5G. The core network is the Internet, which has a relatively better protection capability, and the routing protocols used in this layer are similar to those of the standard Internet. However, extremely constrained IoT devices like sensor nodes and RFID devices may be prone to different types of attacks, including counterfeit attacks, flood, and Man-in-the-Middle (MitM) attacks. Hence security in this layer is very critical. Security threats in the network layer include threats against routing protocols such as spoofed routing information; other threats in this layer include exhaustion of networking resources, jamming signals, Sinkhole, Selective forwarding, Sybil, counterfeit attacks, MitM attacks, session hijacking, DoS attacks, and flooding like DDoS attacks [69].

2.1.2.3 Security Threats in the Application Layer

The application support segment of this layer supports different cloud computing business service capabilities, such as data processing and data storage. Therefore, a high level of security is needed in this segment so as to safeguard sensitive data. Using the application segment of this layer, users can connect to the IoT and access a variety of personalized services according to their access rights or subscriptions. Services in the application layer cut across different domains such as environmental monitoring, healthcare, agriculture, logistics, connected cars, etc. Since this layer involves integrating numerous business applications in different domains, key issues of concern should include how users will safely access data, user privilege abuse and misuse, user privacy and authentication issues like bad password choice, among many others.

Moreover, in recent years, Human-Machine Interface (HMI) is widely used within the IoT industry for monitoring, remote process management, and in situations where human intervention with smart devices is necessary. However, like most IT clients with a HMI, applications within this layer can be susceptible to security vulnerabilities, including Structured Query Language (SQL) injection and Cross-Site Scripting (XSS) [69], etc. Other security threats associated with the application layer include DDoS, DoS, malicious code injection, sniffing, phishing and Social engineering attacks.

2.2 IoT Application Domains

As IoT technology invades every aspect of human activity, its applications have grown significantly in recent years. While a few application domains may still be in their early stages of development, a variety of commercially promising application use cases that span across different domains exist. This section discusses nine IoT application domains, namely smart home, smart grid, smart city, smart transportation, smart healthcare, smart manufacturing, smart supply chain, smart wearable, and smart farming, as depicted in Figure 2.3.

2.2.1 IoT Application Domains and their Associated Cyber Assets

In cybersecurity, knowing the assets that need to be protected is crucial in identifying vulnerabilities, threats, and risks. In this thesis, a cyber asset is defined as a tangible or intangible resource of value to an individual or an enterprise that is exposed on an IoT network, which can also be of interest to an attacker. Such a resource can be hardware, software, or a piece of data that can be found in any of the three layers of the IoT architecture discussed in Section 2.1. Typical IoT assets include IoT or smart devices, communication channels, communication protocols, and smart apps. Figure 2.4 depicts some of the important assets that can be found in each of the nine application domains shown in Fig-

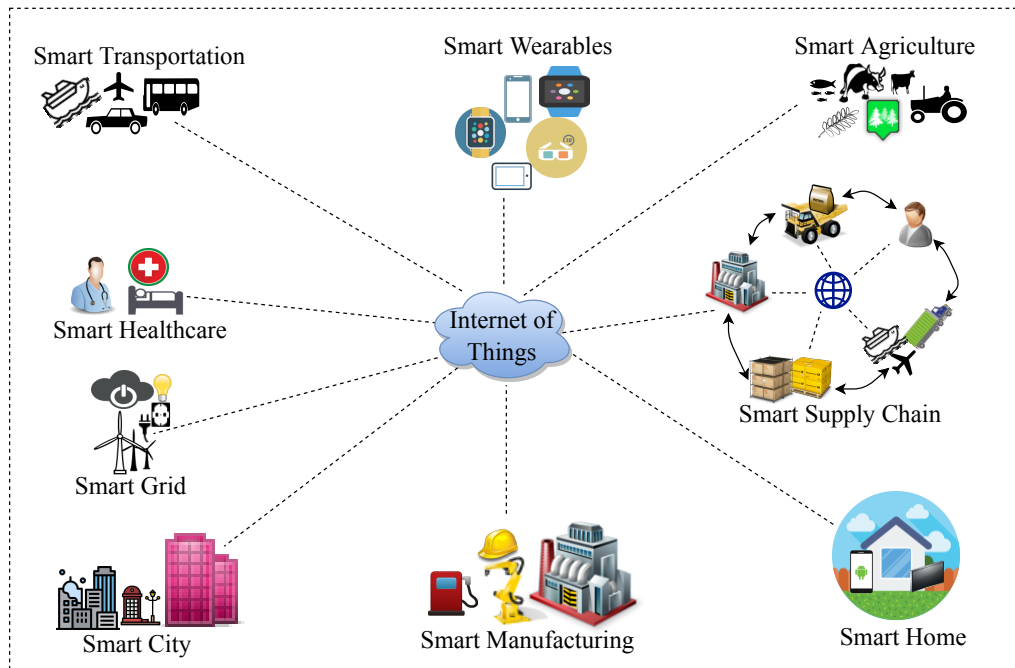


Figure 2.3: Typical IoT application domains (adapted from [45]).

ure 2.3. The following subsections present a brief discussion of each of these application domains along with the cyber assets per domain.

2.2.1.1 Smart Home

An IoT-based smart home is a house or living environment where household appliances like toasters, washing machines, and other everyday devices can be remotely monitored or managed over the Internet. These devices can be controlled using smartphones, tablets, or laptop computers from anywhere in the world via the Internet or private network [70], which allows for better home care and monitoring, access control, energy efficiency, and improved convenience and quality of everyday life. Typical cyber assets in the smart home domain include smart home appliances such as smart TVs, smart thermostats, smart refrigerators, and smart security cameras. Others include smart door locks, smart garage door openers, and smart hubs.

2.2.1.2 Smart Grid

This is an energy delivery paradigm that promises to optimally and efficiently deliver the highest quality of energy at the lowest cost possible. This paradigm, which is rapidly gaining ground in recent times, includes, among other things, a wide range of sophisticated short-range Wireless Sensor Networks (WSNs) used to facilitate the generation, transmission, and distribution of electric power [71], and smart meters used by the consumers. These smart devices constantly collect information about the grid and evaluate the real state of the grid and use this useful information to determine the parameters that can be changed in order to optimize energy delivery. In contrast to the traditional grid,

which is a centrally controlled system, a smart grid is expected to have a more consumer-driven distributed control system. This will provide more accurate monitoring and control adaptation, where consumers can analyze their consumption patterns via the two-way communication between their smart meters and the operators. Typical smart grid cyber assets are smart meters, remote power outlets, transformers, Power Line Communication (PLC), data concentrators, load balancing systems, and data centers.

2.2.1.3 Smart City

The concept of a smart city is not limited to the use of Information and Communications Technology (ICT) to reduce resource consumption and air pollution. The idea is to connect objects, utilities and citizens in a seamless manner using the IoT, so as to enhance the living conditions of people in modern urban environments. A smart city employs IoT technology to improve services in key sectors of the economy such as healthcare, water, energy, transport, and waste-water treatment for the well-being of its citizens. Smart cities are also expected to be proactive in responding to global challenges [72]. Examples of smart city cyber assets are power-generation/distribution systems, Intelligent Transportation Systems (ITS), street lights, Advanced Metering Infrastructure (AMI), water/waste-water treatment facilities, water distribution systems, WSNs, communication systems, Closed-Circuit Television (CCTV) cameras, and Location-based Services (LBSs).

2.2.1.4 Smart Transportation

Smart transportation, also known as ITS, promises to improve the efficiency of the traditional transportation systems, characterized by traffic congestion, fatalities, injuries, air pollution, etc. IoT is making vehicles and roadside infrastructures smarter. Consequently, IoT based ITS can provide a safer, cleaner, and more efficient transportation system [73] by using real-time traffic information and interconnecting vehicles and roadside infrastructures for more efficient data acquisition, processing, and decisions [43]. A legal framework for the deployment of ITS (Directive 2010/40/EU) in the road transportation sector and for interfacing with other modes of transportation was adopted by the European Parliament and Council since 7 July 2010. This is to foster the deployment of these technologies across the Member States [74]. This and other factors have sparked an increasing worldwide interest for research into these areas, including Vehicular Networks, Artificial Co-Driver systems, Intelligent car parking systems, and Self-driving cars. Typical cyber assets in smart transportation include connected cars, traffic lights, parking guidance systems, and dynamic traffic management systems.

2.2.1.5 Smart Healthcare

IoT is revolutionizing healthcare services by dramatically improving quality, lowering costs, and increasing convenience for both patients and medical practitioners. Advances in the

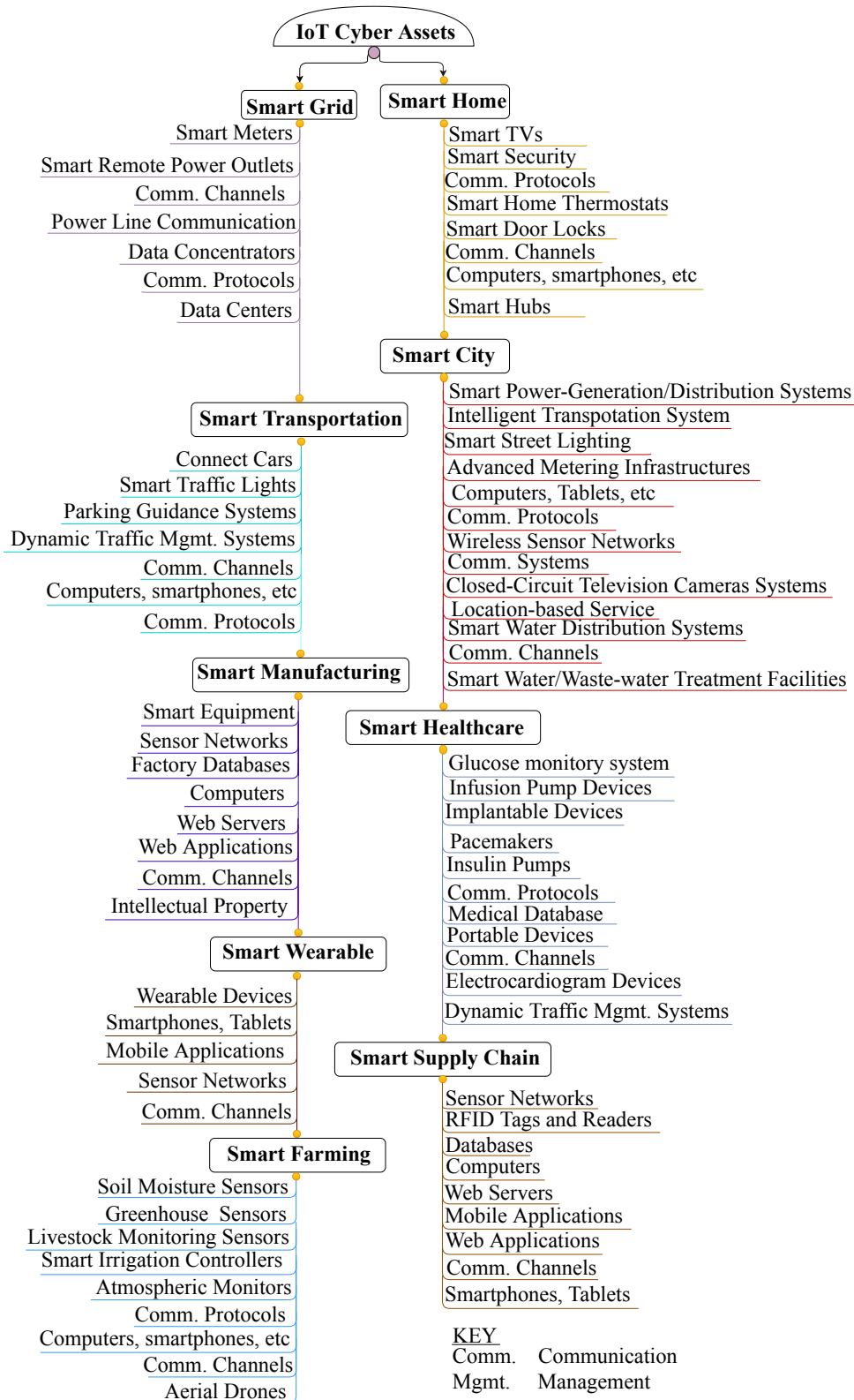


Figure 2.4: Overview of typical IoT cyber assets for nine application domains (adapted from [45]).

fields of WSNs and IT are enabling very small medical devices to be embedded with technologies that allow them to sense, record, analyze, and transmit data continuously over

the Internet. Data from such devices can be accessed anytime and anywhere by the appropriate health professionals like doctors. Thus IoT is enabling doctors and other medical practitioners to examine, diagnose, and treat patients remotely. IoT-based remote healthcare monitoring [75] and Telehealth are becoming more commonplace, especially in countries like India [75]. Typical cyber assets in this domain include glucose monitoring systems, infusion pumps, implantable devices, pacemakers, insulin pumps, electrocardiograms, medical databases, and mobile devices such as smartphones and tablets.

2.2.1.6 Smart Manufacturing

Smart manufacturing is a new paradigm shift towards the fourth industrial revolution (Industry 4.0) with the goal of optimizing the manufacturing process [76]. As the world moves toward Industry 4.0, complex manufacturing processes and global supply chains are increasingly relying on IoT, CPS, WSNs, cloud computing, AI, and big data analytics to lower cost, reduce waste, and boost innovation. This is expected to promote efficient resource utilization, and significantly reduce downtime in manufacturing plants. Examples of smart manufacturing cyber assets include smart equipment, robots, WSNs, factory databases, computers, web servers, web applications, communication channels, and intellectual property.

2.2.1.7 Smart Supply Chain

A smart supply chain refers to a proactive and customer-centric monitoring, tracking, and remote asset management system that integrates different technologies to provide information on location, status, environment, and functionality of products and services. Typical technologies used include IoT, WSNs, RFID, CPS, big data, cloud computing, advanced analytics, and semi-autonomous decisions enabled by AI to build an optimized supply chain that reduces operational costs [77], improves assets identification along the chain [78], and allows for timely responses to unexpected events. Blockchain is another novel technology that may greatly impact the smart supply chain and is recently being used in smart supply chain management. Smart supply chain uses an enormous amount of real-time data derived from different sources, allowing multiway communication between partners, thus making the system fully transparent to all stakeholders, from suppliers of raw materials to the shippers of the raw materials and the finished products, and finally to the consumers. Typical cyber assets in this domain include WSNs, RFID devices, databases, computers, web servers, mobile applications, web applications, smartphones or tablets, and communication channels.

2.2.1.8 Smart Wearable

Smart wearable devices are IoT enabled-wearable end-to-end integrated gadgets embedded with smart sensors and actuators that connect wirelessly to the smartphone or tablet of the user, often using BLE technology. They are usually worn on the wrist, clipped to

the body, or hung around the neck for the purpose of staying fit, being more organized, losing weight, staying active, or for telemedicine purposes. Their applications cut across different domains, such as entertainment, healthcare, sports, and military [79]. Typical cyber assets in smart wearable include wearable devices, smartphones or tablets, mobile applications, WSNs, and communication channels.

2.2.1.9 Smart Farming

Another area where IoT is having a significant impact is smart farming, also referred to as precision farming. This is a sustainable farming practice aimed at increasing the per unit yield of farming land by optimizing water use and preserving other natural resources in order to increase crop yields and financial returns [80]. Considering that water plays a fundamental role in farming, smart agriculture employs the use of ICT based technologies, such as WSNs and GPS services to reduce water wastage during irrigation, and hence enhance water management. Moreover, not only does IoT improve agricultural supply chain operation efficiency, but it also enhances precision livestock farming, where farm animals are monitored for prompt disease detection, early treatment, and nutrition interventions [81]. Typical cyber assets in this domain include soil moisture sensors, livestock monitoring sensors, greenhouse sensors, irrigation controllers, atmospheric monitors, and aerial drones.

2.3 IoT System Models, Threat Models, and Threat Landscape

System models, threat models, and threat landscapes can be used to analyze and describe the security status of IoT systems. These concepts can also be used to identify and deal with threats in IoT systems. The following subsections present detailed descriptions of system models and threat models for the nine application domains discussed in Subsection 2.2.1, as well as discuss IoT threat landscape.

2.3.1 IoT System Models

An IoT system model is a conceptual model that represents a given IoT system and provides the overall description of the functionality or behavior of that system. Abstraction is one of the fundamental concepts in system modeling, which involves hiding certain underlying details in order to focus on the essential features of the system. While there are many other modeling techniques [82, 83], the IoT system models presented in the following subsections consist of simple diagrams (Figure 2.5) that clearly show the interactions between system components, between actors and a system, as well as the interactions between a system and its environment. This will allow for accurate characterization of the systems with respect to their associated security risks and threats.

2.3.1.1 Smart Home System Model

Figure 2.5 (a) shows a system model of a smart home consisting of a controller, which can be any IoT home automation hub, a wireless router, and 8 Wi-Fi enabled smart home appliances that can interface with a home automation software platform such as OpenHAB and Home Assistant. The software platform allows a user to wirelessly control devices from a smartphone or any computer on the home network. The controller is connected to the home network via the router Ethernet interface. A Raspberry Pi Single Board Computer (SBC) can also be configured as the controller. The messaging protocol often employed for communication between the home automation software platform server and the smart devices is the MQTT. A typical lightweight server that implements the MQTT protocol is the Eclipse mosquitto.

2.3.1.2 Smart Grid System Model

The system model presented in Figure 2.5 (b) decomposes the smart grid system into its various sub-systems, showing physical/logical relationships between the various components and the necessary information flow among them. The figure illustrates the use case of a typical smart grid that generates electricity using solar and wind renewable technologies. The different components are generation, transmission, distribution, operations, service provider, and customer; and it is assumed that these components interact with each other using a number of communication protocols. Real-time remote management and control can be achieved using the operations subsystem via a communication network.

2.3.1.3 Smart City System Model

A typical smart city is usually made up of many components, however, in this system model, only five components, namely, citizens, smart buildings, urban infrastructure, economic activities, and smart city database management are used, as depicted in Figure 2.5 (c). Each of these components is composed of many subsystems, and it is assumed that these subsystems interact seamlessly, allowing for a smooth flow of information between the various components. For example, a few subsystems in the economic activities component include a smart banking system, smart manufacturing, and smart supply chain.

2.3.1.4 Smart Transportation System Model

This system model shows a robust transportation system that exploits seamless information coordination and exchange across the different components of the network to provide efficient and safe transportation. The system components include smart traffic lights, cameras, radar, GPS, smart vehicles, smart trains, and smart road infrastructure, as illustrated in Figure 2.5 (d). The system utilizes real-time GPS location data, traffic flow prediction mechanism, location-based services, smart traffic light scheduling man-

agement, and an enormous amount of sensor data collected from different areas in the network for making final decisions.

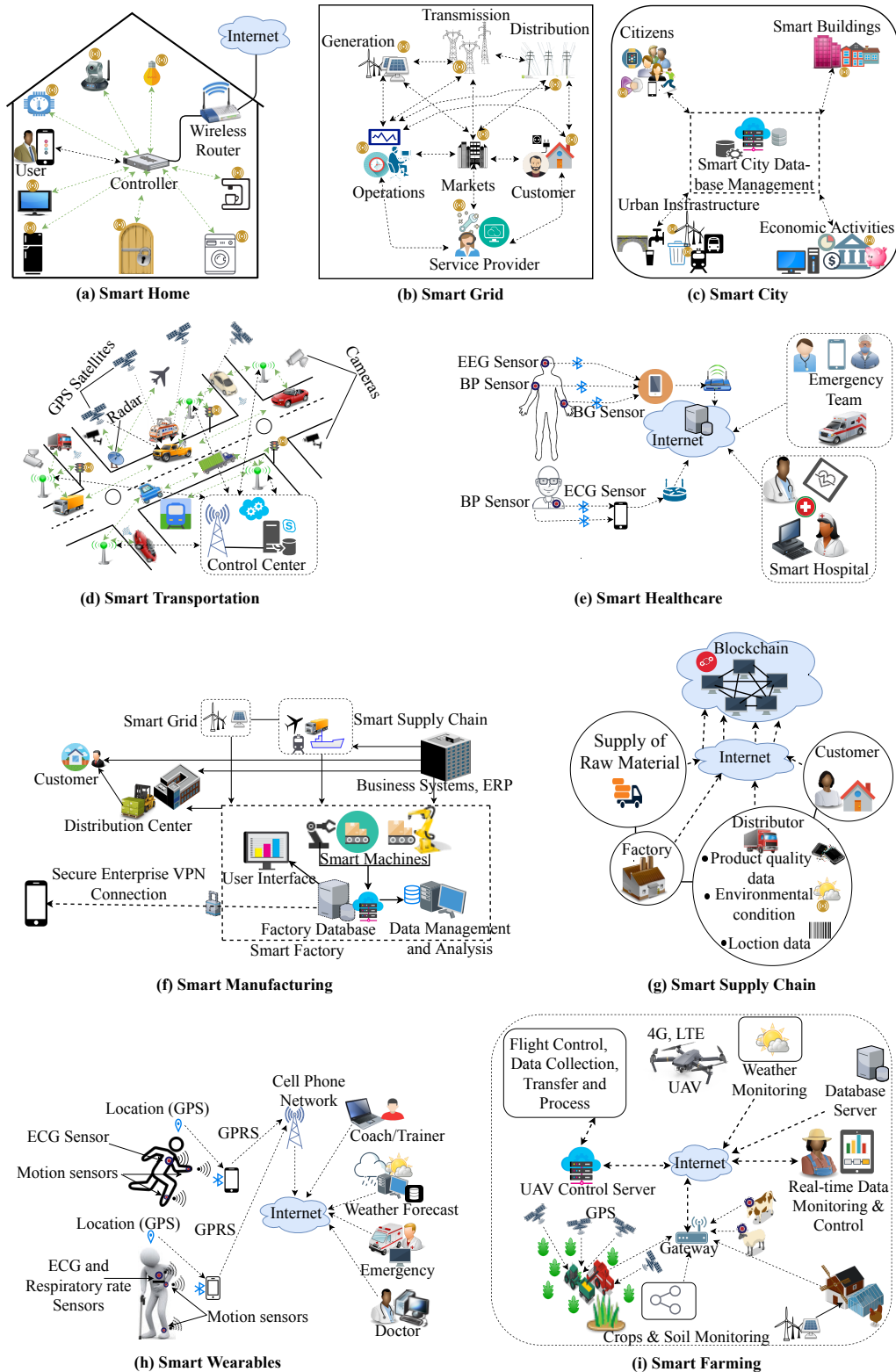


Figure 2.5: Typical system models for the nine application domains (adapted from [45]).

2.3.1.5 Smart Healthcare System Model

Figure 2.5 (e) shows a typical smart healthcare monitoring system. The system model consists of two outpatients, a smart hospital, and an emergency team. The two outpatients comprise of a typical patient wearing Electroencephalography (EEG) sensor, Blood Pressure (BP) sensor and Blood Glucose (BG) sensor, and an elderly patient that needs constant monitoring, wearing Electrocardiogram (ECG) sensor and BP sensor. The sensors on the bodies of patients continuously collect and send data to their smartphones via Bluetooth, and these smartphones, in turn, upload the data to the medical server via the Internet. In case patients are in critical condition, these sensors can immediately report the physical condition of the patient to the emergency team and to their doctors for appropriate actions to be taken.

2.3.1.6 Smart Manufacturing System Model

The system model in Figure 2.5 (f) depicts how manufacturing industries are striving to meet and adapt to the dynamic customer requirements in the current industrial transformation (Industry 4.0). The system model comprises of a smart factory with its various subsystems, a distribution center, renewable energy sources, business systems, and their associated process management tools like the Enterprise Resource Planning (ERP) software, smart supply chain, customer, and secure enterprise VPN connection that allows remote workers to securely connect to corporate networks. Sensor data and manufacturing operations across these interconnected manufacturing components can be utilized in order to achieve better business objectives. This represents a paradigm shift from centralized manufacturing towards a more decentralized production empowered by sensor data-gathering, IoT, CPS, and analytic capabilities, where information from a variety of sources and locations can be leveraged to enhance productions and services.

2.3.1.7 Smart Supply Chain System Model

Figure 2.5 (g) depicts a system model for smart supply chain based on Blockchain, comprising production and distribution systems of geographically dispersed enterprises that collaborate in a secure manner to jointly and efficiently produce and deliver end products to consumers. In the context of a supply chain, IoT security issues will typically revolve around authentication, connection, and transaction. However, using the distributed ledger in Blockchain, the enterprises, namely, suppliers of raw materials, factories, and distributors can conduct business transactions in a trusted and secure environment. The immutable record of Blockchain enables reliable creation of network histories, allowing for tracking the actions of network devices.

2.3.1.8 Smart Wearable System Model

The smart wearable system model depicted in Figure 2.5 (h) is made up of different components, including an athlete, an elderly outpatient, a coach, an emergency team, a doctor,

and a weather forecast server. The athlete is wearing ECG sensor and motion sensors, and the elderly patient is wearing an ECG sensor, respiratory rate sensor, and motion sensors. These sensors collect and send data to the smartphones via Bluetooth on a regular basis, and the smartphone uploads the data to a cloud server via a cell phone network. The coach can only access information pertaining to the progress of the athlete; similarly, the doctor can only access information pertaining to the health of his patient. In case of emergency conditions, the emergency team and the doctor will receive an urgent message from the ECG sensor on the athlete, or from the ECG sensor and/or respiratory rate sensor on the patient.

2.3.1.9 Smart Farming System Model

Figure 2.5 (i) shows a diagram representing the smart agriculture system model. Plants normally require specific environmental conditions for optimal growth, health, and overall crop yield. Thus, the system model consists of different sensor networks for measuring soil moisture, temperature, sunlight, humidity, as well as sensor networks for monitoring the location of farm animals, and for prompt disease detection. It also consists of an Unmanned Aerial Vehicle (UAV), and onboard the UAV is an SBC, a flight control system (autopilot) like Pixhawk, sensor, and camera. The sensor is used to collect real-time data from the flight control system every second and sent to a server via 4G LTE dongle attached to the SBC. The autopilot can also receive commands through the server, and hence by connecting to the server via the Internet, the farmer can control the drone by viewing the real-time data and issuing control commands.

2.3.2 IoT Threat Models

In the context of IoT security, a threat refers to anything or an entity that has the potential to cause possible danger to an IoT asset by exploiting a vulnerability or weakness. Threat modeling, in IoT context, is a process of identifying, enumerating and describing potential threats to vulnerable IoT assets and services in order to define countermeasures that can capture the characteristics of the behavior of adversaries.

The following subsections present threat models based on different possible attack scenarios for each of the nine application domains. A threat can cause different levels of negative impacts to an IoT system. In this thesis, threat impacts are classified into three levels: high, medium, and low threat levels. According to this classification, a high threat can potentially cause total destruction or corruption of an asset in any layer of the IoT architecture; a medium threat is capable of causing partial destruction of asset or denial of service, and a low threat level can cause disclosure of information or eavesdropping.

2.3.2.1 Smart Home Threat Model

Threats in a smart home usually originate from outside the network. The adversarial sources may include but are not limited to criminal groups, phishers, hackers, bot-network operators, spyware authors, and malware authors. An adversary can exploit a compromised device like smart TV to eavesdrop on smart home residents (low threat). Similarly, an attacker can interrupt a Bluetooth pairing communication between two smart home devices and masquerade himself as the real receiver or sender in order to carry out further attacks (medium threat). Furthermore, a knowledgeable adversary can leverage the fact that firmware updates for most smart home devices are usually not digitally signed and can be accessed from Trivial File Transfer Protocol (TFTP) server. As such, the attacker can redirect a smart home device to a nefarious TFTP server using Address Resolution Protocol (ARP) spoofing, or MitM attack to take control of the device or the whole network (high threat). Attackers can also exploit security vulnerabilities that exist in some smart home devices and intrude into smart home networks and launch attacks, such as DoS and MitM (medium threat). Moreover, an attacker can physically attack or tamper with a smart home device, thereby compromising the security mechanisms of the device (high threat).

2.3.2.2 Smart Grid Threat Model

Smart grid threats may arise from both within and outside the smart grid network. The adversarial sources may comprise disgruntled employees, disgruntled customers, hackers, criminal groups, ransomware operators, bot-network operators, malware authors, and spyware authors. Malicious parties can eavesdrop on communication traffic between a smart meter and a central system, or try to analyze data in transit so as to deduce information from communication patterns (low threat). Intruders can as well compromise a smart meter using IP spoofing, and then use MitM attacks and falsify the billing report (medium threat). Similarly, intruders can illegally install some monitoring software on a smart meter, or any computer within their reach that will enable them to sniff off or steal critical information about a consumer or a utility provider, such as usernames, passwords, etc., (medium threat). Attackers can also tamper with a stored data by modifying or deleting part or all of the data, which may affect the operation of some key devices, such as transformers and circuit breakers (high threat). Additionally, intruders can logically capture a smart meter and change the firmware, which will give them the opportunity to gain control over the device. Moreover, an attacker can physically attack a smart grid installation, such as substation or transformer, and change some configurations, make wrong connections, or even vandalized it (high threat).

2.3.2.3 Smart City Threat Model

Smart city threats may originate from within or outside the smart city network. Smart city adversaries can include hackers, hostile governments, terrorist groups, disgruntle

customers, criminal groups, ransomware operators, disgruntled employees, spyware authors, malware authors, bot-network operators, and spammers. Attackers can disrupt communication signals of autonomous vehicles, which can pose a potential threat to public safety (high threat). Attackers can also target smart grid networks in a smart city in order to cause disruption of power supply to consumers (high threat). Adversaries can gain remote access to a smart waste-water treatment plant or a smart water facility and cause environmental damages, or cause water shortages, which can create public health crises (high threat). Additionally, malicious actors can target a smart street lighting system and cause electrical blackouts (medium threat).

2.3.2.4 Smart Transportation Threat Model

Threats in this domain may come from both within and outside the smart transportation network. The adversarial sources may include, but not limited to criminal groups, terrorist groups, hostile governments, ransomware operators, disgruntled employees, hackers, bot-network operators, malware authors, and spammers. An adversary can decide to compromise the Anti-lock Brake System (ABS) of a smart car by attacking the controller area networks of the car using DoS, spoofing, or any other type of cyberattacks, which may result in car accidents (high threat). An attacker can also launch DoS attacks on critical infrastructure, spamming it with useless messages in order to exhaust the available bandwidth so as to prevent sensors on the infrastructure from obtaining important environmental information, such as data on traffic, proximity, signaling, and speed, which may have implications on safety (high threat). The adversary can equally create the same effects by infecting critical transportation networks with malicious software. Moreover, since GPS is critical to the smooth operations of autonomous cars, a malicious person can install a GPS jammer on an autonomous car or on any car that can move in close proximity to it in order to disrupt the performance of the in-car GPS receiver. This will result in misleading the autonomous car (medium threat). Furthermore, since autonomous cars can be considered to be Multi-agent System (MAS), where critical tasks like negotiating corners and stopping the car are at the mercy of MAS, an attacker can take control of an autonomous car by capturing its MAS (high threat).

2.3.2.5 Smart Healthcare Threat Model

Threats in smart healthcare may come from both within and outside the network. Adversaries may include, but not limited to hackers, disgruntled employees, ransomware operators, spyware authors, malware authors, bot-network operators, spammers, and criminal groups. Adversaries can compromise a smart healthcare system by exploiting a vulnerability in order to gain access to the network and carry out a variety of attacks (medium threat). The attackers can eavesdrop on the communication between a physician and a patient (low threat); they can also launch a DoS attack on the communication network (high threat), or capture/hijack a critical medical device (high threat).

2.3.2.6 Smart Manufacturing Threat Model

As all aspects of modern manufacturing are becoming smarter, moving from closed systems into Industry 4.0-based manufacturing driven by IP-based IoT and CPS, industrial plants and businesses are being exposed to numerous cyber threats. Security threats in this domain may originate from outside and within the smart manufacturing network. Adversaries may include ransomware operators, business competitors, industrial spies, disgruntled employees, innocent employees, criminal groups, hostile governments, malware authors, spyware authors, and hackers. Adversaries can compromise a smart device and gain access to a critical application (high threat), they can remotely alter or corrupt a manufacturing process (high threat), maliciously manipulate machines to produce undesirable results (high threat), or they can cause disruption or denial of process controls (high threat). Adversaries can target databases for the purpose of stealing intellectual property and industrial espionage (medium threat). Additionally, Bring Your Own Device (BYOD) also offers attackers opportunities to target smart industries through their employees (medium threat).

2.3.2.7 Smart Supply Chain Threat Model

Cyber threats in the smart supply chain can come from different threat actors with varying motivations. Adversaries may include criminal groups, third-party suppliers, chipmakers, business competitors, ransomware operators, industrial spies, disgruntled or rogue employees, innocent employees, hostile governments, malware authors, and hackers. Since data is a key driver of cybercrime, persons with malicious intent may access data through third-party organizations and compromise smart supply chain and cause different levels of damages (medium threat). They can also leverage the lack of proper authorization, accountability, and authentication in the cloud to gain access to targeted networks and, ultimately, their data, which may include sensitive information like customer contracts, credit card information, and other valuable data (high threat). Adversaries can inject malware in smart supply chain applications that can steal personal information, as well as allow for complete remote control of devices (high threat). In addition, innocent employees can, unintentionally, jeopardize supply chains as a result of negligence, or simple human errors (medium threat).

2.3.2.8 Smart Wearable Threat Model

The growth of wearable technology and the increasing number of users of smart wearable devices pose a series of security threats that open the doors for cybercriminals to target organizations and individuals via IoT devices such as smartwatches, smart headgears, fitness trackers, and smart clothing/accessories. Security threats in this domain usually originate from outside the network, however, disgruntled employees and innocent employees that use such devices may also constitute potential threats to an organization. Adversarial sources may include criminal groups, hackers, business competitors, indus-

trial spies, disgruntled employees, innocent employees, and malware authors. Malicious entities can easily access sensitive data from a company if they compromise communication links between smart wearable devices and a mobile device of an employee that has access to the corporate network (medium threat). They can also compromise cloud-based data centers and access sensitive financial information (high threat). Adversaries can use a malware-infected wearable device to infect other data sources or create back doors for malicious purposes (high threat). They can also intercept and manipulate data transmitted to or from wearable devices (low threat).

2.3.2.9 Smart Farming Threat Model

In smart farming, threats may likely originate from outside the network. Adversaries may include industrial spies, terrorist groups, criminal groups, hostile governments, ransomware operators, hackers, and spyware authors. An adversary can attack an automated fertilization system and change the percentage of chemical agents to produce toxic materials that will be very harmful to crops (high threat). Additionally, attackers can create the same effects by maliciously interchanging chemical agent A for agent B (high threat). An attacker can also infiltrate a sensor network of a greenhouse and compromise the sensors so as to report false data on important information, such as temperature or soil moisture, which will have negative effects on the greenhouse (high threat). Attackers can also disrupt GPS communication using GPS jammers, which may result in disruption of the operations of satellite-guided tractors (high threat). Since smart farm-management relies on different sensor information and analysis of past crop yields to produce planting and business plans, people with malicious intention can eavesdrop data over the wireless communication channel and steal some important farming or business secrets (medium threat).

2.3.3 Overview of IoT Cyber Threat Landscape

In the past, the threat landscape in information security was dominated by malware creators that were merely seeking attention. But in recent years, cybercrime has become a lucrative business through the use of ransomware and selling of prepackaged malware [84]. Consequently, the cyber threat landscape has dramatically changed with numerous capable hackers and cybercriminals everywhere. Additionally, in recent years, IoT is changing the overall cyber threat landscape even further with many devices that are being connected to the Internet by the day and generating massive amounts of data that must be stored, processed, and analyzed. Considering that this massive amounts of data are being generated and transmitted by numerous connected devices and smart apps that may have exploitable vulnerabilities, it is a natural progression for cybercriminals to shift their focus to attacking the IoT [26]. This is contributing to the constantly changing nature of IoT threat landscape, which continues to get more sophisticated, aggressive and destructive.

There is undeniable evidence that the dependence of people, organizations, and businesses on the IoT is expanding the attack surface for hackers and cybercriminals [85]. For instance, the adoption of IoT technology in many organizations has weakened the defenses of many companies by creating a host of new security loopholes for hackers and cybercriminals to exploit [84]. Over the last few years, for example, there has been a stream of data breaches in the headlines involving IoT devices, confirming the fact that cybercriminals are now shifting their focus from the traditional computer systems such as desktop computers to mobile devices, especially IoT devices and smart apps; and hence the need to bake security and privacy into the design and development of IoT devices and smart apps from the onset.

2.3.3.1 IoT is Fast Changing the Cyber Threat Landscape

There are two key fundamental defense strategies in cybersecurity. Firstly, to examine the network and strive to figure out in advance the possible sources of threats and try to prevent them. Secondly, to have a mechanism in place that can monitor or scan network traffic in real-time so as to detect possible leaks that may occur in compromised devices [86]. The aforementioned defense strategies are possible in the traditional risk management models since organizations own most, if not all, of the cyber assets and the data flowing through them. With the advent of the IoT, cloud computing, and other new ICT systems, however, most businesses today are being done outside the defensive fence of organizations [86]. In addition to this, the mobile and distributed nature of IoT technology is pushing organizations assets and data beyond their corporate defense systems, such that potential attackers are presented with a large attack surface. Thus, both organizations and individuals are now exposed to new types of Advanced Persistent Threat (APT) scenarios that current risk management schemes can not defend against [85].

In this era of IoT, the potential for cyber threat landscape growth is almost unbounded [26]. For instance, the resource-constraints, as well as the deployment nature of many IoT devices, make it a real challenge to identify seemingly independent malicious events from orchestrated activities or to apply certain security solutions at strategic locations in the network [31]. Given the large attack surface created by the number of connected devices [85], it is now not very difficult for cybercriminals and hackers to search for weak links in the cybersecurity chain using available tools, or search engines like Shodan. Essentially, exploiting a vulnerability in a single weak link in the security chain could open a doorway for attackers to have unauthorized access to sensitive data, including user sensitive data like personally identifiable information. Currently, when hackers want to attack an organization they do not have to target the well-protected servers, but rather they target ordinary devices that are relatively unmonitored but are connected to the network, such as smart coffee makers or smart fridges, and if they successfully break their way through such smart devices, they can eventually reach their main target.

2.3.3.2 IoT is Attracting the Attention of Malicious Attackers

Many IoT and embedded systems lack inherent defenses and there are virtually no firewalls to protect them. Instead, they are left wide open to cyberattacks, relying on simple password authentications. This is probably based on the assumption that IoT devices are not attractive targets to cybercriminals and hackers. However, the increase in number and sophistication of targeted attack campaigns against IoT devices that dominated the news headlines lately clearly proves that such an assumption is no longer valid [87]. The fact that many industries, including healthcare, power, hospitality, and banking are employing the use of IoT platforms and smart apps for financial services is enough incentive to attract the interest of financially motivated cybercriminals. Moreover, even if some IoT devices may not contain anything that is of high-value to attackers, the fact that such devices are typically embedded deep within enterprise networks makes them attractive targets. Additionally, the susceptibility to attack and capture of smart devices [26] compared to traditional computer systems that are adequately protected, coupled with the ignorance of many users with regards to the security of such devices open an avenue for cybercriminals and hackers to attack IoT systems.

2.3.3.3 IoT Threat Actors

In recent years, IoT has been an industry buzzword with hundreds of both positive and negative press releases, and most of the bad news headlines revolve around security incidents caused by different threat actors. In IT security, a threat actor, also called a malicious threat actor, is a person that uses one or more threat vectors to attack a target, resulting in an incident that has an adverse impact on (or has the potential to affect) the security of an individual or an enterprise. There are several threat actors targeting IoT with different intentions. Examples of threat vectors that could be used in the IoT threat scenario include fake websites, wireless unsecured hotspots, email links/attachments, mobile devices, malware, and Universal Serial Bus (USB) (removable) media. A threat target can be anything of value to the threat actor, such as a critical asset. Threat actors are generally grouped into two categories, namely external (outsider) and internal (insider) [88].

1. *IoT External Threat Actors*: External threat actors in the IoT are individual hackers, well-funded hackers, organized crime groups, ransomware operators, or government entities that seek to gain access to protected information by breaking through and taking over the profile of a trusted entity from outside the IoT network or enterprise [87]. This category of threat actors can launch either active or passive attacks. The attacks are active if the actors participate in the network or generate packets; and passive if they only eavesdrop or track the entities using the IoT network. Most IoT data breaches are typically caused by external threat actors and tend to be more severe in terms of negative impact.

2. *IoT Internal Threat Actors*: In the context of IoT, internal threat actors are people from within an IoT enterprise or private network, or people with legal access rights to an IoT network [88]. They include employees, former employees, business associates, contractors, smart home occupants, and smart city citizens, who have privileged information about the security practices of the organization or network. Such people may even know the usernames and passwords of some critical systems. Although malicious insider threats exist, many IoT users and enterprises do not consider these threats as very important ones. While several data breaches resulting from internal threat actors may be unintentional, an external person like a competitor could also provide financial incentives to motivate internal threat actors to sabotage an organization.

2.3.4 The Complexity of IoT Vulnerabilities

Recently, a plethora of vulnerabilities have been discovered in a variety of IoT consumer products ranging from smart implantable medical devices to connected home appliances. Yet as new devices are being connected to the Internet in their hundreds and thousands, many more of such vulnerabilities are still being discovered. For example, recent studies conducted by a number of researchers reveal alarming vulnerabilities in the IoT [89], especially in consumer IoT products [27].

The Open Web Application Security Project (OWASP) for IoT is designed to provide manufacturers [90], developers, and consumers of IoT devices with the necessary information about the security issues associated with the IoT, as well as to help them make better security decisions with regards to building, deployment or using IoT devices and smart apps. OWASP has identified the following as the most common vulnerabilities in IoT devices:

1. *Username Enumeration*: is when malicious actors use brute-force techniques to either guess or confirm valid users in a system by interacting with the authentication mechanism. The response "That user does not exist" allows a malicious person to collect valid usernames. On the other hand, if the username is valid and the password is not, and the server returns "The password is incorrect", the malicious actor can infer that the username is valid;
2. *Weak Passwords*: allow users to set short and simple account passwords, such as "123456" or "abcdef", or the device default passwords;
3. *Account Lockout*: lack of account lockout allows a malicious entity to keep sending authentication attempts even after 3 to 5 failed login attempts;
4. *Unencrypted Services*: attackers can eavesdrop on the network communication since network services are not properly encrypted;
5. *Lack of 2-Factor Authentication*: 2-Factor Authentication (2FA) adds an extra layer of security for an account using mechanisms like a security token or fingerprint scan-

ner, which ensures that users or entities are who they say they are. Hence lack of 2FA makes it easier for attackers to compromise a system;

6. *Poorly Implemented Encryption*: the implemented encryption is poorly configured or not properly updated, e.g. using Secure Socket Layer (SSL) v2;
7. *Update Sent Without Encryption*: Transport Layer Security (TLS) is not used when transmitting updates over the network or encrypting the update file itself;
8. *Update Location Writable*: storage location for update is writable, which allows malicious entities to modify and distribute the modified version to all users;
9. *Denial of Service*: authorized users or devices can be denied access to a service, or to a device;
10. *Removal of Storage Media*: since some devices are exposed in open fields, the storage media can be removed;
11. *No Manual Update Mechanism*: there is no provision that will enable users to manually force an update check for the device;
12. *Missing Update Mechanism*: no provision for device update;
13. *Firmware Version Display and/or Last Update Date*: device does not display current firmware version or last update date.

Essentially, what makes the vulnerability of IoT devices so complex is the fact that almost all communication in the IoT is wireless, therefore it is possible for attackers to eavesdrop on the communication between devices in the network [91]. In addition to this, many IoT devices have limited power, memory and processing capability, and are often battery operated. Such devices cannot handle large amounts of data [92]. Some of them, such as WSNs, are deployed in highly dynamic and harsh environments. Similarly, due to the embedded nature of some IoT devices, some are deployed in factory environments that are noisy with so much interference, and hence transmitted messages are often lost due to the lossy nature of such networks [93]. In the aforementioned scenarios, IoT devices are expected to operate for an extended period of time in an unattended manner; but this leaves them vulnerable to both logical and physical attacks.

The vulnerability issues of IoT are difficult to deal with because fundamentally most of the devices are not designed with security in mind [94]. There are inherent vulnerabilities in almost every aspect of IoT, including hardware, Operating System (OS), communication protocols, APIs, design of the architecture, etc. [95]. Aside from this, when vulnerabilities are found and fixed, the vulnerability patching process in the IoT is another big challenge. For instance, if a firmware update is required in order to patch the vulnerability, it will be a very difficult task, considering the number of devices that may be involved (which in the case of WSNs may be hundreds or thousands). In addition, not every user or IT

professional can be able to upgrade a custom firmware, since it requires extra time and effort.

2.3.5 Evolving Attack Strategies Against the IoT

Today, attackers are well-resourced, well-organized, and very methodical in approaching their targets for corporate or business espionage, as well as for financial, political, or ideological reasons. According to a report entitled “*IoT under fire*”, Kaspersky has detected more than 100 million attacks on smart devices in the first half of 2019 alone [96]. In addition, according to a recent Spiceworks security survey in the first quarter of 2016 involving about 200 IT professionals in North America and EMEA (Europe, the Middle East, and Africa), 49% of these IT pros pointed to IoT devices as the network-connected endpoints at highest risk in 2016 [97]. Recent research [98, 99] reveals that cybercriminals and malicious hackers are becoming more tactical in avoiding detection and concealing their paths. This is exacerbated by the legal barriers to investigating IoT based crimes which include lack of specific laws; lack of universal IoT regulatory frameworks; the difficulty of monitoring the adoption, correct implementation, and effectiveness of IoT security best practices and standards [38]; as well as the challenges IoT poses to digital forensics.

Over the years, attackers have discovered a number of ways to attack IoT devices. For instance, attackers can use the Universal Plug and Play (UPnP) protocols that enable devices to automatically find other connected devices on a network [100]. In addition, exploiting exposed APIs is another option for attackers [101]. Exposed APIs are major threats to organizations that use the IoT, and hence it is important for all API calls to be encrypted or authenticated. Moreover, the increasing trend towards BYOD to work is another loophole that attackers can exploit to attack an enterprise that is thought to be secured. Furthermore, attackers can also use Domain Name System (DNS) poisoning and hijacking to attack a router, gateway, or hub and take control of an IoT device [102]. A recent example is the attack incident involving Netgear routers [103], where some cybercriminals were able to bypass the embedded authentication process and changed the default DNS to an IP address of a malicious website.

Presently, IoT devices can be weaponized and used as *thingbots* to build up large botnets [28]. A good example of a botnet for attacking IoT devices is the LizardStresser created by a hacker group called Lizard Squad. This botnet was used to hijack IoT devices which were used to launch DDoS attack campaigns on banking, gaming, and government websites, among others. The number of cybercriminals that used the LizardStresser was significantly high, and the most disturbing issue is the fact that the botnet source code has been released to the public in 2015 [104]. Another issue is that even attackers that do not know how to technically exploit a vulnerability may be able to use the botnet, since all that is required is a device default administrative username and password, which many users and some IT professionals do not change; a report showed that 46% of consumers

and 30% of IT professionals do not change the default administrative password [105].

2.3.6 Attacks on the Privacy of Users

In the IoT paradigm, seamless Human-Device Interaction (HDI) is commonplace. An important attribute of HDI is the translation of human intentions into control commands of devices [106]. Today users interact with their IoT devices in diverse ways, for example, they are worn on the body, implanted in the body, interacted with through voice commands, just to mention a few. The interaction of these smart devices with the environment and with humans gives rise to the pervasive collection, processing, and dissemination of different kinds of information. This massive information obviously includes private and personal data that are collected by these devices with or without the consent of the users [107], which raises serious privacy concerns, thus making privacy a predominant issue in the IoT today [108].

While privacy can simply be defined as the appropriate use of data [109], in 1968, Westin defined privacy as “the right to select what personal information about me is known to what people [110].” and in [111], privacy is defined as “the faculty and right that a person has to define, preserve and control the boundaries that limit the extent to which the rest of society can interact with or intrude upon. At the same time, he or she retains full control over information generated by, and related to, him or her.” Although those definitions may not directly refer to electronic information, they are still valid today. Similarly, in the context of the IoT, in order to ensure privacy, it is expected that users should have the right to know and to control what data is collected about them, who maintains and uses the data, and for what purpose [107].

Privacy and security are distinct, but they are complementary properties for IoT services. For instance, one of the purposes of a security policy is to ensure data privacy. This implies that a successful attack on security may eventually result in an attack on privacy. Basically, information security is about preserving the Confidentiality, Integrity and Availability (CIA) of information. Confidentiality represents an intersection between security and privacy, forming a mutual relationship between the two [112]. While in security, confidentiality refers to the property that a trusted party must protect a data that has been collected from being released, in privacy (which is usually about a person), confidentiality is about making sure that such data is not collected, shared or monitored without the consent of the person involved.

There have been a number of successful attacks on the privacy of IoT users that made headlines in recent years [108]. For example, a US researcher Matt Jakubowski exploited vulnerabilities in the Wi-Fi enabled Barbie doll that allowed him to have access to the information system, account information, the audio files stored on the doll, and direct access to the microphone. Having access to these resources could enable a cybercriminal

to spy on children [108]. Another disturbing issue is that tracing the identity of a smart thing may lead to the automatic identification of the owner. This is because a transaction linked to the same identifier is usually traceable, which in turn shows that the owner is also identifiable. Additional information that IoT devices can reveal about their users includes location, interests, habits, and other personal information. In 2015 for example, some researchers at the Synack security firm analyzed 16 smart devices for home automation [113]. They observed that exploiting some vulnerability in those devices could reveal private information, such as user behaviors and activity patterns. It is therefore imperative for everyone with a stake in the IoT to take all the necessary measures to prevent malicious entities from having unauthorized access to private information of users.

2.4 IoT Fundamental Security Issues, Trade-offs, and Countermeasures

As the core technology behind smart devices and applications, which is connecting everything to the Internet, IoT should be very secure. However, as more smart devices and applications are being deployed in complex, uncontrolled, and often hostile environments, securing IoT systems presents several unique security and privacy challenges [114]. For example, the protection of the massive amounts of data that IoT devices and smart apps continue to generate has become an important issue in recent years. This is becoming more critical as IoT has become more pervasive and interwoven into our everyday lives. Consequently, many researchers in academia and industry believe that IoT is hardly safe for processing sensitive information [21, 115]. This section presents a few of the fundamental and root causes of IoT security and privacy issues, a number of security versus performance trade-offs, as well as provides possible countermeasures to some of the issues raised.

2.4.1 IoT Fundamental Security and Privacy Issues

This subsection examines the fundamental IoT security issues. It discusses some of the root causes of IoT security and privacy issues, namely IoT devices are not designed with security in mind; open debugging interfaces; inappropriate network configuration and use of default passwords by users; and lack of encryption of critical information before storage.

2.4.1.1 Smart Devices and Applications are not Designed with Security in Mind

IoT encompasses diverse technologies such as IIoT, CPS, WSNs, robotics, AI, machine learning, and big data analytics, and hence it is labeled as the major driving force behind the 4th industrial revolution. Consequently, its revolution is set to be even bigger and more pervasive than those of other emerging technologies. As IoT finds value in different

aspects of human endeavors, its market is rapidly growing especially as it continues to create real economic value [116]. Although the IoT market is fairly new, revolving around products and services, its potential is huge with annual global revenue expecting to reach trillions of dollars by 2025 [117, 118]. Therefore, new players including both established and start-up companies are taking advantage of the business opportunities in the IoT, and developing innovative IoT use cases as well as creating new business models.

Currently, IoT business opportunities are almost endless, especially as enterprises and individuals begin to reap the benefits of the IoT, making different companies venture into the business [16]. While some technology giants are desperately competing and rushing to be the first to launch a particular product into the emerging IoT market [119], they sometimes forget to give security the priority it deserves. Hence, in the quest to react to various business opportunities and challenges, these manufacturers leave their smart consumer products exposed to all sorts of digital intrusions. Similarly, some IoT start-up companies lack security expertise and as a result, they produce IoT devices and smart apps with exploitable security vulnerabilities [120, 121]. This introduces new exploitable attack surfaces that expand beyond the vulnerable devices or applications themselves into home and enterprise networks, the cloud, and the Internet. Consequently, IoT security breaches resulting from avoidable vulnerabilities continue to make headlines in cybersecurity news every day. This underscores the importance of building security into IoT devices and smart apps from the very beginning rather than leaving it as an afterthought.

2.4.1.2 Open Debugging Interfaces

The importance of building security into IoT devices at the outset rather than considering it as an afterthought was already highlighted in Subsubsection 2.4.1.1 above. One of the important approaches to implementing security by design is to make sure that the attack surface is minimized as much as possible. As such, there is a need for manufacturers of IoT devices and gateways to implement only the necessary interfaces and protocols that will enable an IoT device to perform its intended functionalities. Manufacturers should put a limit on the services of all interfaces on the device for debugging purposes, which, in most cases, may not be even needed by the user, and can allow hackers to have direct access into the local area network that the device is connected to.

Although leaving such interfaces may be indispensable for the manufacturer and researchers for development and testing purposes, a user may not even know that such interfaces exist throughout the lifespan of the device, hence he does not need them. In addition, these open debugging interfaces are potentially dangerous and present opportunities for malicious entities to hack the device or access important information. Moreover, through them, malicious attackers can remotely run some harmful code, such as viruses and spyware on the device [122]. For instance, in the research study conducted by Verocode [123], the research team discovered that some debugging interfaces were left open and unse-

cured on two of the devices they examined, through which a malicious attacker can run arbitrary code and harm the system. Even though some of these interfaces may be hidden, a malicious person can go to any length to discover them.

To sum it up, in order to implement workable security in the IoT, the concept of security by design should be given the priority it deserves so that unnecessary security flaws can be avoided. For example, it is important for manufacturers to include some mechanism in their design that will prevent even a legitimate user from running malicious code that can be dangerous to an IoT device or gateway.

2.4.1.3 Inappropriate Network Configuration

With the continued rise in the production of diverse IoT devices, new smart devices are being connected to the Internet by the day, and currently, the connected versions of almost every household appliances are available in the market. Additionally, as manufacturers are seriously competing to grab their share of the predicted \$3 trillion annual revenues in the IoT market by 2026 [124], the prices of these *things* are becoming cheaper every day. For these reasons, people are now using connected devices more than ever before. A recent survey carried out by a leading global data privacy management company, TRUSTe [125], showed that 35% of the U.S. and 41% of British domestic online consumers own at least one smart *thing* apart from their phone. The survey further revealed the popular devices in use, which include: smart TVs (20%), in-car navigation systems (12%), fitness bands (5%), and home alarm systems (4%). Even though the percentage and popularity of devices may differ, the results of this survey are likely to be true for other European Countries, China, and many other developing countries. However, security vulnerabilities have been discovered in a number of brands of most of the smart devices listed in the TRUSTe survey [126, 89].

While there have been many reports of IoT security attacks in recent years, unfortunately, it seems that many consumers are not aware of IoT security vulnerabilities and their associated implications on their privacy and security. This is evident from the manner in which some consumers install, configure, and use their smart devices. A study [126] shows that a significant portion of IoT security concerns revolves around consumer appliances for smart homes. Thus, a badly configured IoT device or home network can lead to a security breach. For example, the *do-it-yourself syndrome*, where some consumers use default passwords settings on their smart devices, as in the cases of the security camera incidents reported in [127, 128], has resulted in many attacks and privacy abuses. In some instances, attackers can hijack the compromised devices and turn them against their users, for example, they can turn them into spying machines to spy on their users [129].

Other issues are the use of weak passwords by users and the lack of good authentication features on smart devices [130]. Although there are devices that come with good password

authentication features that require users to create new passwords immediately after their first log on, some users use simple passwords that are easy to guess. On the other hand, the restrictions on some devices may not allow users that know the value of security to use long and complex passwords on their devices. For instance, some smart devices allow only minimal security configuration or customization. Furthermore, many of the devices do not have keyboards, and since all configurations must be done remotely, many users may not have the skills to do such configurations. More importantly, it is possible that some users are unaware that attackers usually look for poorly configured networks and devices to exploit. Therefore, it is important for vendors to find a way of educating consumers on current security best practices, which include changing passwords regularly and proper network configuration.

2.4.1.4 Lack of Encryption of Critical Information Before Storage

In Subsection 2.3.6, it was mentioned that some IoT devices and smart apps collect certain kind of personal information with or without the consent of users. Such information may include name, date of birth, address, zip code, email address, location, health information, social security number, and in some cases even credit card number. To show the level of concerns about privacy among smart *things* users, an online survey was conducted by TRUSTe in the U.S. on 2,000 users aged between 18 and 75, which revealed that 59% of them are aware that smart *things* can capture sensitive information about their personal activities [131]. 22% believed that the benefits and conveniences that come with the IoT innovation are worth sacrificing their privacy for, and surprisingly, 14% were comfortable with such companies collecting their personal information. The question now is not, if these devices are really collecting personal data from users, but, as rightly posed by HP [132]: “Do these devices really need to collect this personal information to function properly?” One of the obvious reasons that most companies would give for capturing personal user data is that they need such data in order to improve their products. Another reason could be to know the habits of their valuable customers so that they can serve them better by creating customized services for them.

Now that it is established that some IoT devices and smart apps collect sensitive personal information from their users, limiting the amount of data collected by these devices and applications is crucial. Of course, this is a difficult thing to do because of the possible business value of such data [107, 133]. However, as these devices store data locally on their memories and transmit it over a variety of networks, there are concerns that the data could be accessed or manipulated over the Internet [134]. This is because data stored on IoT devices constitutes a tempting target for attackers, either due to the value of the information it refers to or because of the simplicity involved in accessing such data [26]. Recent studies show that many IoT devices store and/or transmit data in plaintext [135, 134], making it easy for hackers to access confidential information, and hence the need for securing data at rest and in transit between IoT edge devices and back-end systems.

So far, encryption has been identified as the best way to protect sensitive information from being revealed to unauthorized entities [136, 137]. While encryption is widely believed to be the best approach to provide data confidentiality, implementing standard encryption algorithms on resource-constrained IoT devices presents some unique challenges for security experts. This is because most of those algorithms require extensive computational resources and large memory. Similarly, the use of TLS for securing communication in some IoT devices is not an option since it requires significant processing power and memory, which are usually scarce resources on resource-constrained devices.

Another issue is that both standard and lightweight encryption algorithms only protect data at rest or in transit, leaving sensitive information in plaintext and potentially vulnerable to disclosure during processing, especially by cloud providers. This raises questions about the privacy of users. It also raises concerns about the protection of trade secrets such as confidential business information that provides an enterprise a competitive edge. This problem can be overcome by employing Homomorphic encryption schemes, which allow the processing of encrypted data without knowing the private key [138], and generating an encrypted result which when decrypted provides the same result of processing performed on the decrypted data. Nonetheless, implementing Homomorphic encryption on resource-constrained and low-power IoT nodes is currently a major challenge [139].

2.4.2 Security versus Performance Trade-offs in IoT

IoT offers very appealing prospects for almost every area of human endeavor. However, while connecting every device to the Internet comes with added convenience, increased productivity, and comfort to mankind, it is imperative that users consider what they may be sacrificing from a security and privacy point of view. Sometimes the convenience that embracing new technology brings comes at a price, and in the case of the IoT, the price is the value of security and privacy that enterprises and individuals lose when they connect vulnerable IoT devices and applications to the Internet.

As highlighted in Subsection 1.1.1, managing security and privacy are among the major challenges facing the IoT, and there is always a tough trade-off between security and performance. Unfortunately, however, many people often prioritize ease and convenience over security and privacy [140, 141]. Users of unsecured IoT devices and smart apps knowingly or unknowingly give up varying amounts of privacy and security in order to enjoy the convenience of connected devices. However, data leaks and other negative effects of cyberattacks are often the results of such faulty decisions. One fundamental challenge facing IoT designers, manufacturers, and developers is that the more secure IoT devices or smart apps are, the more inconvenient it is [142, 141]. For manufacturers, security decision-making requires making the appropriate trade-offs. While developing secure IoT devices and smart apps may seem expensive, failure to observe security best prac-

tices can result in loss of consumer confidence, which can eventually cost a company a lot more in lost sales opportunities than the cost of implementing proper security [143, 144]. The following subsections briefly outline some key trade-offs that should be taken into consideration when designing IoT devices and smart apps.

2.4.2.1 Security versus Energy Trade-offs

Energy efficiency and security represent two fundamental challenges for the design and implementation of IoT systems; the two requirements nonetheless are at odds with each other. For example, while security, which usually drains energy resources, is needed to safeguard data and provide privacy, low-power consumption is needed for longevity of energy source [145]. This is because IoT edge devices are usually equipped with small batteries, which makes energy conservation a critical issue in the IoT. This issue, therefore, requires making intelligent and informed trade-offs. A security architecture based on configurability and adaptability was developed in [146] to address security versus energy trade-offs in the IoT.

2.4.2.2 Security versus Processing/Memory Capacity Trade-offs

When it comes to security implementation, there is always the question of performance and speed. This is because cryptographic schemes are expected to be portable and fast, but with little or no reduction in security performance [147]. On the contrary, most standard security solutions are computationally intensive and require a lot of memory space. Many IoT devices, however, are designed to have very small foot-print with extreme processing and memory constraints [148]. Hence the requirements of standard cryptographic algorithms are at variance with the requirements of most constrained IoT devices. The key to resolving such trade-offs is to know the specifications of the constrained devices and evaluate the requirements of potential lightweight candidate algorithms.

2.4.2.3 Security versus Cost Trade-offs

IoT devices, especially those at the sensing layer, such as RFID devices, sensor nodes, and cameras, are expected to be cheap and disposable [149], because thousands of such devices may be needed for some applications such as smart environmental monitoring. But there are complex trade-offs between security and hardware/software cost [150], which require careful business and security/privacy assessments. The fact is that the more resources are added to a device (e.g., memory, processing power, and longer battery life) the more expensive the device will be [151].

2.4.2.4 Security versus Convenience Trade-offs

Most smart devices, especially in the consumer domain such as household appliances, are designed to be easy to install and use [152]. This is because when people buy a new device,

they are usually eager to go home and start using it immediately. From the point of view of many users, it is actually more convenient to buy a device that is already configured with all settings in place. No wonder in order to satisfy such consumers, some devices come with default settings that can allow them to perform the basic functionality required by the user as soon as they are powered on. Most users do not bother to do any security configurations on their devices, even if it is just to enable wireless security settings, and hence some of these devices fall victim to cyberattacks.

In the case of some IoT devices, the matter is even worse since many of those devices may not have good user interfaces, and some must be configured remotely or interfaced with a laptop or desktop computer in order to be properly configured. Here, there is a trade-off between security and convenience, and people would often err on the side of convenience over privacy and security [153]. Although many consumers are unlikely to make the right trade-offs, manufacturers can help to improve security by making some basic security settings on their IoT products unavoidable. In addition, such IoT devices should not allow the use of weak passwords.

2.4.3 Possible IoT Security and Privacy Countermeasures

Following the security threats in IoT architecture presented in Subsection 2.1.2 and the fundamental IoT security and privacy issues highlighted in Section 2.4.1, this section provides some basic possible safeguards against a number of IoT security and privacy threats identified. Based on the types of threats, the countermeasures are classified into the following categories: countermeasures against threats associated with insecure web interfaces and network services; countermeasures against threats on routing protocols (sink-hole, selective forwarding, black hole, hello flood, wormhole, and Sybil attacks); countermeasures against threats on the information in transit; countermeasures against physical and environmental threats; countermeasure against GPS jamming; countermeasures against tag tracking and tag cloning; and countermeasures against inappropriate network configuration.

2.4.3.1 Countermeasures Against Threats Associated with Insecure Web Interfaces and Network Services

Today, IoT has become the major source of propagating DDoS attacks [26], and as a consequence, many big organizations have been seeing a large number of attempts to penetrate their databases in recent years. Before an IoT device is compromised and weaponized for any DDoS attack, standard ports like port 23 (Telnet), 2323 (many IoT devices use this port as an alternate port for Telnet), and 22 (Secure Shell (SSH)) will be bombarded recursively with authentication attempts to gain access to the device [154]. Therefore, countermeasures should include blocking TCP/IP ports and other nonessential services running on the smart device that can be used for probing and brute-forcing, thereby preventing

the device from accepting unsolicited commands from attackers via backdoors; changing default passwords on devices; choosing strong passwords; and ensuring that only necessary ports are exposed. The UPnP protocol on smart devices should also be disabled. Updating the operating system and applications running on smart devices with the latest bug fixes and patches is also a very important safeguard since software vulnerabilities are usually discovered after a product is released.

2.4.3.2 Countermeasures Against Threats on Routing Protocols

Protecting IoT networks against such threats is challenging due to the constrained nature of IoT devices [155]. Nevertheless, sinkhole attacks can be prevented using robust authentication schemes. Geographic routing and systematic rerouting can also reduce the chances of sinkhole attacks. A geographic routing protocol makes routing decisions based on the positions of the nodes and packet destination. Similarly, a possible countermeasure against wormhole attacks is to use geographic routing, also known as geographic forwarding, such that the next hop of a packet is the node closest to the destination node. Another way to protect IoT systems against wormhole attacks is by physical monitoring of edge devices, as well as using source routing to regularly monitor the network; a LEACH [156] based protocol can be used for such monitoring.

Monitoring the network regularly using source routing can prevent or reduce the chances of selective forwarding/blackhole attacks. This involves specifying which route some or all of the packets take through the network. Using multipath routing, it is also possible to protect against selective forwarding. The technique is about using multiple alternative paths through a network. For example, the multiple disjoint routing paths or diversity coding may be employed. One possible countermeasure against Hello flood in IoT systems is by using bidirectional authentication or by verifying the bi-directionality of a connection link. A countermeasure against Sybil attacks is to evaluate the identities of individual nodes in the network in order to assure that a particular ID is assigned to only one node, and no other node should have that same ID. Another defense method is the use of random key pre-distribution schemes [157].

2.4.3.3 Countermeasures Against Threats on Information in Transit

Threats that fall under this category include eavesdropping, MitM, malicious code, replay, traffic analysis, hijacking, hacking, masquerading, malicious data deletion, spoofing, and economic espionage. It was opted to present general countermeasures here due to the number of threats involved. When a smart device connects to a network, it should be authenticated prior to participating in any data exchange. This will ensure that the message originates from a legitimate source. Mutual authentication enables two IoT entities to prove their identity to each other, which protects against attacks. There are few lightweight cryptographic symmetric and asymmetric key algorithms that can be used for

this two-way authentication, including Secure Hash Algorithm (SHA-x) and Hash-based Message Authentication Code (HMAC) for symmetric keys, and the Elliptic Curve Digital Signature Algorithm (ECDSA) for asymmetric keys [158].

Another safeguard is secure boot using a cryptographically secure signing method, which helps to ensure that an IoT device only executes code generated by a trusted party or the Original Equipment Manufacturer (OEM) [159]. This prevents malicious entities from replacing device firmware with malicious code. There is a need to always encrypt data in transit between one device and another, as well as between the device and its service infrastructure (e.g., the Cloud). Another important safeguard is regular intrusion monitoring and analysis. This involves monitoring the overall system, including edge devices and connectivity traffic. By analyzing such data, it is possible to detect violations of company security policies or potential threats. If potential threats are detected, a range of measures can be taken in the context of the security policy of the organization. In some applications, the manufacturers, OEMs, or the Cloud service providers may opt to control the security of the system, and effectively manage all security aspects of the device during operation using different security features. For example, in the event of a security breach, the Over The Air (OTA) device keys replacement feature can be used to mitigate the cyber disaster and ensure minimal service disruption.

2.4.3.4 Countermeasures Against Physical/Environmental Threats

Threats in this class include node capture, tampering, and side channel. Countermeasures against node capture should include camouflaging the node, making it difficult to remove the storage medium, encrypting stored data at rest, making the device too compact to be disassembled, and protecting USB ports from malicious access. The device should also be equipped with capture perception, and the ability to delete the data when subjected to unanticipated stress conditions.

Safeguard against tampering should include frequent changing of the secret key, using proper key management techniques, and the ability to detect tampering within a very short time like 5-6 milliseconds without slowing device performance. One safeguard against side-channel is that hardware and software designers should obfuscate the timing, power consumption, and electromagnetic radiation information that can be gleaned from the System on Chip (SoC) or device hardware. Additionally, hardware designers should always choose side-channel resistant processors.

2.4.3.5 Countermeasure Against GPS Jamming

Essentially, GPS Jamming refers to the process of sending a fake and noisy radio signal in order to disrupt or override a GPS signal so as to report a fake location or make it difficult for GPS receivers to receive broadcast microwave signals from GPS satellites.

One countermeasure against this threat is to use the adaptive notch filter mitigating technique [160].

2.4.3.6 Countermeasures Against Tag Tracking and Tag Cloning

A countermeasure to tag tracking attacks is to ensure that the response of a targeted RFID tag to a malicious reader appears random and uniformly distributed. Similarly, a safeguard against tag cloning is to prevent the attacker from accessing the identifying data of a tag under attack. But the back-end server should be able to verify the response of the tag for authentication purposes, and the response must not reveal the identity data of the tag.

2.4.3.7 Countermeasures Against Inappropriate Network Configuration

Countermeasures against this threat include educating users about the importance of security, enforcing strong password policies, deactivating or blocking nonessential ports and services such as Telnet and SSH, disabling the UPnP protocol, and enabling logging of security events.

2.5 Brief Overview of IoT Hardware Development Platforms

The recent advancements in semiconductor technology and the trend toward miniaturization of semiconductor devices, coupled with the recent breakthroughs in embedded systems design that enabled the integration of high-level software and low-level electronics, have created a paradigm shift in embedded systems development. Consequently, there is a dramatic drop in the price of computing hardware; in addition, computer products and hardware are becoming smaller in size, easier to create, faster, and easier to integrate. This has led to the invention of microcontroller-based boards, SoCs, and SBCs, which in turn led to the current state of the art in IoT hardware development platforms, in which there is a variety of tiny, energy-efficient, and inexpensive IoT hardware platforms that can be used to design and develop tailored IoT devices.

An IoT hardware development platform is a small single electronic circuit board that combines microcontrollers and processors with other components such as memory and wireless connectivity chips in a variety of ready-to-build packages for IoT design, prototyping, and mass production. Such platforms play a vital role in the overall development of IoT. Essentially, IoT hardware development platforms provide designers, developers, researchers, hobbyists, or anyone with some level of experience to create interactive electronic objects with prebuilt and ready-to-program kits that enable them to focus more on their projects. Most IoT hardware platforms feature onboard sensors, while others have provision for connecting external sensors or a means of connecting additional sensors, as the case may be. Hardware platforms for IoT development can be programmed via

an external interface to another computer, or via a web-based Integrated Development Environment (IDE).

2.5.1 General Classification of IoT Hardware Development Platforms

IoT hardware development platforms or boards can generally be classified into two main types, namely, open-source and proprietary. The design details of open-source platforms are made publicly available for users to study, reproduce, and modify to suit their various purposes. These platforms are usually more suitable for prototyping. Most open-source platforms have active and supportive communities that make building projects for the IoT easier. On the other hand, proprietary IoT boards are typically designed to be used in end applications and users are not allowed to have access to the design details. Such boards are more suited for final production. IoT Hardware development platforms can be divided into three categories: microcontroller-based, SoCs, and SBCs.

2.5.1.1 Microcontroller-based IoT Hardware Development Platforms

A microcontroller-based IoT hardware development platform integrates a number of components such as a microcontroller; RAM; flash memory; internal clock; different kinds of Input/Output (I/O) interfaces such as Serial Peripheral Interface (SPI), Inter-Integrated Circuit (I2C), Inter-IC Sound (I2S), USB, and Universal Asynchronous Receiver/Transmitter (UART); and other supporting components, all built onto a single Printed Circuit Board (PCB). This type of platform provides all the necessary circuitry for some useful control tasks, and therefore they are used mainly for prototyping. A popular example is the Arduino *Uno*.

2.5.1.2 System on Chips

An SoC is an Integrated Circuit (IC) that integrates the necessary components of a computer into a single silicon chip [161]. Typically, an SoC contains a processor, a Graphics Processing Unit (GPU), memory, and circuitry for power management, along with some I/O peripherals, such as SPI, I2C, I2S, UART, USB, and Peripheral Component Interconnect (PCI). SoC devices usually have small form factor, consume less power, and are computationally excellent. Hence, they are used for building SBCs as well as for mass production of IoT devices and other embedded systems. An example is the Intel Curie module.

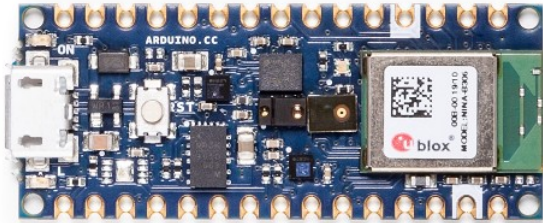
2.5.1.3 Single Board Computers

SBCs are complete computers on small PCBs with microprocessors, memory, power management circuitry, a GPU, real-world multimedia, and some I/O interfaces, such as USB, UART, High Definition Multimedia Interface (HDMI), and Ethernet, Wi-Fi or cellular data connection, which allows it to function as a computer. SBCs are usually used for IoT

prototyping, for educational purposes, and for use as embedded computer controllers. One example is the Raspberry Pi.

2.5.2 Key Features of IoT Hardware Development Platforms

Recent advances in microprocessor chip technology have reshaped the IoT hardware industry in profound ways [162, 163], resulting in a variety of IoT hardware development platforms. These hardware platforms come with several features that make them suitable for the development of IoT devices for different applications. Today there are several different IoT prototyping hardware platforms on the market, with new models still being released regularly. Figure 2.6 shows two examples of current IoT hardware development platforms. Although the basic functionality of these platforms is very similar, each platform comes with different features, capabilities, and limitations that make it ideal for certain applications. Thus, the choice of a hardware platform depends completely on the type of project. This subsection, however, focuses on a few key features and attributes of IoT hardware platforms, namely, processing and memory/storage capacity; power consumption, size, and cost; Operating Systems (OSes); connectivity and peripherals; and hardware security features.



(a) Arduino Nano 33 BLE (Image CC-SA-BY from Arduino.cc, 2020)



(b) Raspberry Pi 4 Model B: 8GB RAM (Image from Raspberrypi.org, 2020)

Figure 2.6: Examples of IoT hardware development platforms.

2.5.2.1 Processing and Memory/Storage Capacity

Among the many underlying features that determine an IoT hardware platform performance are processing speed and memory capacity. Like other computing hardware devices, the number and type of processors and their speed determine the processing capacity of IoT hardware. Similarly, the amount and type of memory a development board possess directly impacts its performance. Memory as well as processor performance requirements for connected devices depend largely on the type of sensing, processing, and communication needed for the target application. For instance, some devices are designed to perform a limited amount of processing on data sets like temperature or humidity, and others are designed to handle more complicated and bandwidth-intensive tasks such as video streaming or high-resolution sound. In view of the diversity of IoT applications, and especially as the IoT is maturing, with smart devices able to perform much more complex tasks without human intervention, there is a need for a greater diversity of chip configurations. This implies that the processor and memory requirements of some IoT hardware platforms need to be like those of computers and smartphones. Consequently, today there are IoT hardware platforms with a reasonable number of processors and sufficient memory; one example is the Raspberry Pi 4 Model B shown in Figure 2.6 (b).

2.5.2.2 Power Consumption, Size and Cost

A key requirement for IoT systems is the ability to consume very low power while maintaining an acceptable level of performance, which can enable them to run on batteries for long periods of time. Therefore, one of the most important things to consider when designing and deploying IoT devices for certain applications is battery life. This is because replacing batteries in the field is not an easy task and it is not economically viable, especially if the replacement will involve hundreds or even thousands of devices. How much battery energy is consumed by an IoT device depends significantly on the radio transmitter type, protocols used for communications, the sensors, and the processor type [164]. Some strategies for achieving ultra-low-power operation in IoT devices include, among others, employing low-power communication technologies and implementing low duty-cycle operations [165]. Examples of batteries commonly used in small IoT devices include lithium, nickel, and alkaline batteries. Most of these battery chemistries offer very low self-discharging characteristics, making them well suited for long service intervals.

There is a relationship that exists between the size and the cost of electronic devices: as devices become smaller, their prices usually increase due to high manufacturing costs. But when the technology matures and the process of miniaturization is fully automated, prices begin to decline [166]. Over the years, the idea of shrinking transistor sizes onto microcontrollers and computer processors to enhance performance as well as to reduce size and cost has become more complex following the twilight of Moore's law. However, in recent years researchers have developed a new process for stacking thin layers of semi-

conductor material with transistors while the performance of the transistors remains intact. The result is a landmark achievement that led to the development of monolithic 3D chips, which behave like a single device, having the same size as the two-dimensional(2D) chips, consuming less power, and generating less heat. The monolithic 3D chips have the advantages of cost-effective manufacturing and smaller footprint [167].

2.5.2.3 Operating Systems

An OS usually acts as a resource manager, making it essential for managing the limited resources on some resource-constrained devices of the IoT. From the hardware perspective, IoT devices can be divided into two major categories depending on their performance and capability: high-end devices and low-end devices [168]. Raspberry Pi and many other SBCs fall under the first category because they have adequate resources to run lightweight versions of the traditional General-Purpose Operating System (GPOS), also known as a High-Level Operating System (HLOS), such as Linux and Android. The second category consists of devices that are resource-constrained and cannot run HLOS, such as Arduino *Uno* and wireless sensor nodes. But since precise timing and timely execution are very crucial in many IoT use cases, such as in smart healthcare and industrial automation, a Real-Time Operating System (RTOS) based on a micro-kernel and designed for a very small memory footprint, as well as for energy efficiency, is best suited for such devices [168]. In contrast to computers and mobile devices like smartphones, there is a wide variety of open-source and commercial RTOSs for IoT devices. RTOSs for IoT applications are usually designed for real-time performance, as well as to run efficiently on small-form-factor and low-power devices.

2.5.2.4 Connectivity and Peripherals

In the IoT vision, things are expected to become part of the Internet and active participants in information, business, and social processes where every connected device is uniquely identified and accessible over the network. Consequently, reliable Internet connectivity is arguably the most important component of IoT. This underscores the need for IoT development boards to be Internet-enabled and have the necessary communication interfaces that will allow them to interact and communicate with each other, as well as to sense the environment. While the Internet of Space (IoS) [169], a high-data-rate suborbital-based communication network, is still on the horizon, mobile networks such as 3G, 4G, 5G as well as other wireless technologies like Wi-Fi will continue to play crucial roles in providing the IoT with Internet connectivity, at least in the near future.

Although many IoT applications like WSNs for simple environmental monitoring require only a limited number of interfaces for connecting a few sensors, there are other applications that will require several different I/O peripherals. Currently, most IoT hardware platforms come with several I/O peripherals, also known as on-chip peripherals, or provi-

sion to solder external peripherals. As the IoT takes shape and offers endless possibilities for organizations, businesses, and services, the need for a greater diversity of peripherals will exponentially rise to meet the ever-increasing demands.

2.5.2.5 Hardware Security Features

Embedding security into IoT hardware may offer an additional layer of protection to some IoT devices. This may help in achieving security by design in some IoT systems as software security alone has proven inadequate to protect some smart devices against many known threats, such as DoS, DDoS, malware, and tampering attacks [170]. For devices that are not too constrained in terms of resources, combining software and hardware security can provide better security. However, this will depend on the ability of IoT hardware makers to implement reliable security at the hardware level, which can be achieved by including an encryption chip, also known as a Security Controller (SC), in the hardware (e.g., TrustZone). The SC performs a defined set of cryptographic operations using cryptographic keys that are securely stored in the SC. These operations include identifying unauthorized access and detecting tampering. If a tampering or micro probing is detected, the chip creates a tamper response that results in an immediate zeroization. The SC should also be resistant against Differential Power Analysis (DPA) and other side-channel attacks. Other approaches include using attestation, which is a mechanism that enables software to verify the authenticity and integrity of hardware and software of a device; Physical Unclonable Functions (PUFs)-based authentication, which are circuit primitives that can derive secrets from physical characteristics of ICs; and by using Trusted Platform Module (TPM) and Trusted Execution Environment (TEE), a piece of hardware specifically designed to do crypto calculations and a secure area on a chip for the execution of trusted applications, respectively.

2.6 Conclusion

The various aspects of IoT security and privacy, as well as the range of different IoT security and privacy issues that were discussed in this chapter, were important not only to provide a concise perspective on the current state of cybersecurity in IoT but also to help organize and structure the framework that was to be developed later in Chapter 4. This is of particular importance as it will serve as a preparatory work usually needed prior to developing a solution in computer security, which will help to define the exact scope and boundaries of the security framework.

This chapter provided an overview of some IoT security and privacy concepts in the literature. It described a three-layer IoT architecture and discussed the security requirements as well as highlighted a number of security threats in the given architecture. The chapter also described nine application domains and their associated cyber assets and used them to discuss IoT system models and threat models. It also provided an overview of the

IoT threat landscape in which it discussed several aspects of IoT threats. Furthermore, this chapter discussed fundamental security and privacy issues and security versus performance trade-offs in the IoT. Additionally, it presented some possible countermeasures to the IoT security and privacy threats highlighted in the chapter. Finally, it provided a brief overview of IoT hardware development platforms.

The next chapter (i.e., Chapter 3) will attempt to explain some important concepts underlying the design and development of the security framework that is described in this thesis, which will be presented in Chapter 4.

Chapter 3

Basic Concepts of IoT-HarPSecA Framework

This chapter prepares the ground for answering the important question: "what exactly is IoT-HarPSecA framework?" In order to answer this question, it is crucial to discuss, first of all, the underlying concepts and principles behind the security framework, which is exactly what this chapter is devoted to. The chapter discusses security requirements in the IoT, IoT security and privacy best practices, and presents an overview of LWC and LWCA's upon which the design of the IoT-HarPSecA framework hinged. This chapter is mainly based on articles 1, 2, and 10 [45, 43] mentioned in Subsection 1.5.1.

3.1 Introduction

In chapter 2, the foundation for the need for a security framework that could facilitate the design and development of secure IoT devices and smart apps has been laid down. The chapter presented some of the most important concepts of IoT security and privacy, which gave important insights into IoT threat landscape, and provided an overview of the scope of IoT security and privacy issues and challenges. One of the important points that emerge from the state-of-the-art presented in Chapter 2 is the need for manufacturers of IoT systems, especially IoT start-up companies, to build security and privacy into their products from the beginning of their lifecycle and across their entire product building process (see Subsection 2.4.1.1). This will entail baking security and privacy into the stages of planning, designing, building, and testing of IoT products.

Essentially, the concept of the IoT-HarPSecA framework was born out of the need to address the aforementioned issue. However, because this chapter is dedicated to discussing the underlying concepts behind the design and development of each of the three components of the security framework, the discussion of the IoT-HarPSecA framework, which will provide the answer to the question posed at the beginning of this chapter, is reserved for the next chapter (i.e., Chapter 4).

Explaining the important principles, concepts and theory behind the IoT-HarPSecA security framework is an important point of this thesis. Therefore, this chapter is focused on describing the key concepts and principles underlying the design and implementation of the three components of the security framework. The next section, Section 3.2, presents discussions of security requirements in the IoT, which provides the basis upon which the first component of the security framework is built. Then, Section 3.3 discusses IoT security and privacy best practices, which form the background of the second component of

the framework. Moreover, Section 3.4 presents an overview of LWC and LWCAs, which serve as essential bedrocks of the third component of IoT-HarPSecA. Finally, Section 3.5 wraps up the chapter with a conclusion.

3.2 Security Requirements in the IoT

In requirement engineering, also known as the requirement phase (which is the initial phase of a system development process), system requirements are generally divided into two major groups, namely Functional Requirements (FRs) and Non-Functional Requirements (NFRs) [171]. But since the degree of satisfaction that a system provides to its users is what generally determines the success of system design, FRs describe the functionality that meets the user requirements while NFRs impose restrictions on the functionality [172]. NFRs include usability, performance, security, and privacy [172]. The current practice in many IoT start-up companies is to treat the FRs as first-class requirements during the design process while some of the NFRs such as security and privacy are usually considered during implementation or as afterthoughts in the event of a cyberattack or a related cybersecurity incident [107].

As an important factor in measuring the security of an IoT system, security requirements, in the context of IoT, can be defined as a set of conditions that describe the requirements that need to be satisfied in order to achieve the security goals of an IoT system. Although security requirements are usually considered as NFRs, they constitute important features that prescribe what a system or an application needs in order to meet a particular security objective. Thus, identifying the security requirements of an IoT system is a major step towards providing proper security protection.

Ascertaining the security requirements of an IoT system is key to designing an appropriate security solution and the subsequent evaluation of that solution, which will entail checking whether or not those requirements are fulfilled, among other things. While there are international standards for identifying and defining security requirements for standard computer systems, the counterparts of such standards for the IoT are scarce, immature, and not widely adopted by manufacturers [38]. In the previous chapter, it was already mentioned that fitting security into the scarce resources of many IoT devices can be difficult. This implies that satisfying certain security requirements in some IoT systems is a huge challenge due to the limitations associated with resource-constrained IoT devices [173]. Nonetheless, it is still possible to identify the important security requirements of an IoT system and strive to meet at least those that are critical to the secure operation of the system.

3.2.1 Security Requirements per Domain

Generally, the primary goals of information security can be described in terms of three fundamental security goals (which are interchangeably referred to in the literature as security properties or security attributes): confidentiality, integrity, and availability, usually coded by CIA [174]. However, these security attributes have been broadened over time to include other security attributes, properties or requirements like authenticity, authorization, non-repudiation, accountability, reliability, user privacy, physical security, etc. In Subsection 2.1.1, the above mentioned security requirements and many others have been identified and defined in a three-layer IoT architecture.

Nonetheless, IoT systems are exposed to a variety of both internal and external threats which may differ from one domain to another, as highlighted in Subsection 2.3.2. In the same manner, IoT security requirements may differ across different application domains as some applications are more critical than others. This is because, in some application domains like smart healthcare, there may be stringent security and privacy requirements dictated by various regulatory bodies or agencies [175]. Additionally, because different IoT systems are typically made up of different components and end-nodes, and the fact that different organizations may use their IoT systems for different purposes, and hence, will have different security policies, security requirements may differ significantly across different domains of application.

The following subsections outline some of the most important security requirements for protecting each of the nine application domains presented in Subsection 2.2.1. The security requirements are presented in a holistic manner, approaching the issue from a system perspective rather than a layered approach. However, note that usage scenario and hardware capability can be considered to be among the most critical factors that will determine if more security requirements would be needed, or some of the security requirements presented herein are unnecessary.

3.2.1.1 Smart Home Security Requirements

The following are some of the most important security requirements that are required to provide a secure and trustworthy smart home services. It should be guaranteed that:

- information is protected from disclosure to unauthorized individuals or devices (data confidentiality, user privacy);
- no data are modified or deleted by unauthorized persons for any malicious reasons and any such action is traced back to the entity responsible (data integrity, accountability);
- only authorized devices and users are allowed to access any information on the network and that users cannot bypass permission checks and access information they

are not permitted to access (authenticity, authorization);

- the services of smart devices should be available to authorized entities whenever needed (availability, reliability);
- resilience of the connection between the core network and the edge devices is maintain even in the face of faults or other challenges (network resilience);
- ensuring that all RESTful APIs are authenticated and authorized, and that request timestamp is added to prevent basic replay attacks (Secure API);
- actors with malicious intent cannot have physical access to smart device in a smart home (physical security).

3.2.1.2 Smart Grid Security Requirements

In a typical smart grid, the most important security requirements should include the following:

- ensuring that information is protected from disclosure to unauthorized devices or individuals (data confidentiality, user privacy);
- ensuring that the accuracy, trustworthiness and consistency of information is maintained throughout the life cycle of a message (data integrity);
- access to information is only limited to authorized users and devices, and that devices should be able to verify the source of a message (authenticity);
- users, devices and applications are only allowed to have access to the resources needed for their normal operations (authorization);
- IoT end nodes at the sensing layer are able to adjust self to handle failures and avoid a single point of failure (device resilience);
- ensuring that authorized devices or persons are able to access information or service when needed (availability, reliability);
- ensuring that no party can deny participating in the transfer of a message and that any transaction can be traced back to individual entities involved (Non-Repudiation (NR), accountability);
- ensuring that smart grid devices are resistant to tampering and that any attempt to tamper with such devices is detected (tamper resistance and detection);
- network intrusions are detected, prevented, or mitigated (attack detection, prevention, or mitigation);

- ensuring that RESTful APIs for smart meter measurements are authenticated and authorized, and that request timestamp is added to prevent basic replay attacks (Secure API);
- the communications of smart grid devices are monitored to identify or detect abnormal communication behaviors that may be caused by malware (Anomaly detection);
- a smart grid network is proactively searched for threats that may be lurking undetected in the network (threat hunting);
- ensuring that resilience of the connection between the core network and the edge devices is maintain even in the face of faults or other challenges (network resilience);
- ensuring that all sensitive smart grid devices are physically secured (physical security).

3.2.1.3 Smart City Security Requirements

Given that a smart city is usually composed of multiple IoT subsystems or components, the security requirements of a typical smart city should include:

- ensuring that no authorized users or devices are denied access to any smart city service or information (availability, reliability);
- ensuring that access to any smart city services is limited to authorized users and devices only, and that the source of every message is verified and that messages are not altered (authenticity);
- ensuring that users and devices can only have access to the resources they are permitted to access (authorization);
- ensuring that no smart city information is disclosed to unauthorized users and devices (data confidentiality, user privacy);
- ensuring that no party can deny participating in any transaction and that any action is traceable to the entities involved (NR, accountability);
- ensuring that smart city devices at the sensing layer are able to adjust self to handle failures and avoid a single point of failure (device resilience);
- ensuring that the communications of smart city devices are monitored to identify or detect abnormal communication behaviors that may potentially be caused by malware (Anomaly detection);
- ensuring that no part of stored data or data in transit is modified or deleted (data integrity);
- ensuring that smart city devices can withstand significant tampering attempts and that any such attempts are detected (tamper resistance and detection);

- ensuring that smart city network is proactively searched for threats that may be lurking undetected in the network (threat hunting);
- ensuring that resilience of the connection between the smart city core network and the edge devices is maintain even in the face of faults or other challenges (network resilience);
- ensuring that all RESTful APIs are authenticated and authorized, and that request timestamp is added to prevent basic replay attacks (Secure API);
- ensuring that intrusions into smart city networks are detected, prevented, or mitigated (attack detection, prevention, or mitigation);
- ensuring that all sensitive smart city devices are physically secured (physical security).

3.2.1.4 Smart Transportation Security Requirements

The following are some of the important security requirements that should be guaranteed in smart transportation:

- data in transit should be protected such that no captured data should reveal any part of the message (data confidentiality, user privacy);
- the origin of every message should be verified and sensors on smart cars, smart trains and road/rail infrastructures should only respond to queries from authorized entities, so that no unauthorized parties are allowed to access or inject data (authenticity);
- users or devices should only be able to have access to the information or resource they are permitted to access (authorization);
- no transmitted or received data is allowed to be maliciously modified or deleted (data integrity);
- that smart transportation network is proactively searched for threats that may be lurking undetected in the network (threat hunting);
- that devices at the physical layer are able to adjust self to handle failures and avoid a single point of failure (device resilience);
- that APIs are authenticated and authorized, and that request timestamp is added to prevent basic replay attacks (Secure API);
- that the services of smart vehicles and infrastructure are always available to authorized entities (availability, reliability);
- that sensing devices and infrastructure are resistant to tampering and that any attempt to tamper with such devices are detected (tamper resistance and detection);

- that resilience of the connection between the smart transportation core network and the edge devices is maintain even in the face of faults or other challenges (network resilience);
- that every sensing devices, road infrastructure, or vehicles are physically protected from unauthorized entities (physical security);
- that no party can deny participating in any transaction and that any transaction can be traced back to the entities involved (NR, accountability);
- ensuring that network intrusions are detected, prevented, or mitigated (attack detection, prevention, or mitigation);
- that the communications of smart transportation devices are monitored in order to identify or detect abnormal communication behaviors that may potentially be caused by malware (Anomaly detection).

3.2.1.5 Smart Healthcare Security Requirements

In a typical smart healthcare system, the following are considered to be among the most critical security requirements:

- privacy of patients should be protected such that no medical records or any sensitive data are read by any unauthorized person or device (data confidentiality, user privacy);
- no unauthorized entities are allowed to have access to any smart medical devices or networks and that smart medical devices should be able to verify the source of a message (authenticity);
- an authorized entity can only have access to the information needed to perform its normal operations (authorization);
- no medical records or data are modified or deleted by unauthorized persons for any malicious reasons (data integrity);
- intrusions into a smart healthcare network are detected, prevented, or mitigated (attack detection, prevention, or mitigation);
- no authorized persons, such as patients or physicians are denied access to any medical device or a healthcare service (availability, reliability);
- smart healthcare devices at the edge layer should be able to adjust self to handle failures and avoid a single point of failure (device resilience);
- resilience of the connection between a smart healthcare core network and devices at the edge is maintain even in the face of faults or other challenges (network resilience);

- all RESTful APIs are authenticated and authorized, and that request timestamp is added to prevent basic replay attacks (Secure API);
- that smart healthcare network is proactively searched for threats that may be lurking undetected in the network (threat hunting);
- smart medical devices are not physically accessible to unauthorized users (physical security);
- communications of smart healthcare devices are monitored in order to identify or detect abnormal communication behaviors that may potentially be caused by malware (Anomaly detection);
- smart medical devices should include hardened enclosures to resist tampering and that any attempt to tamper with such devices is detected (tamper resistance and detection);
- any breach of patient confidentiality can be traced back to perpetrators and no party can deny participating in any transaction (accountability, NR).

3.2.1.6 Smart Manufacturing Security Requirements

The security requirements of a typical smart manufacturing system should include the following:

- only authorized entities are allowed to have access to smart devices or networks and smart equipment and devices should be able to verify the source of a message (authenticity);
- authorized employees, devices and applications can only have access to the information needed to perform their normal operations (authorization);
- smart manufacturing devices at the edge layer like sensors should be able to adjust self to handle failures and avoid a single point of failure (device resilience);
- all APIs are authenticated and authorized, and that request timestamp is added to prevent basic replay attacks (Secure API);
- information is only accessible to authorized staff and devices (data confidentiality);
- no data on transit, or on a storage device is maliciously modified or deleted (data integrity);
- no authorized staff or devices are denied access to any smart equipment or service (availability, reliability);
- communications of smart manufacturing devices are monitored in order to identify or detect abnormal communication behaviors that may be caused by malware (Anomaly detection);

- resilience of the connection between a smart manufacturing core network and sensors at the physical layer is maintain even in the face of faults or other challenges (network resilience);
- no staff or device can deny participating in any transaction and that any transaction can be traced back to a staff or an entity involved (NR, accountability);
- intrusions into a smart manufacturing network are detected, prevented, or mitigated (attack detection, prevention, or mitigation);
- smart manufacturing network is proactively searched for threats that may be lurking undetected in the network (threat hunting);
- smart equipment and smart devices should be tamper resistant and any unauthorized attempt to tamper with smart equipment or devices should be detected (tamper resistance and detection);
- no unauthorized staff or persons are allowed to physically have access to smart equipment or devices (physical security).

3.2.1.7 Smart Supply Chain Security Requirements

The most important security requirements for securing smart supply chain should include:

- ensuring that authorized entities are not denied access to any service or information (availability, reliability);
- ensuring that no information is accessible to unauthorized entities (data confidentiality, user privacy);
- making sure that no data on transit, or on a storage device is maliciously modified or deleted by any entity (data integrity);
- ensuring that unauthorized entities are not allowed to have access to any service and ensuring that messages are from the stated sender, and that they have not been modified along the line (authenticity);
- ensuring that entities can only have access to what they are permitted to access (authorization);
- ensuring resilience of the connection between a smart supply chain core network and sensors at the physical layer is maintain even in the face of faults or other challenges (network resilience);
- smart supply chain devices at the edge layer like sensors should be able to adjust self to handle failures and avoid a single point of failure (device resilience);

- ensuring that communications of smart supply chain devices are monitored in order to identify or detect abnormal communication behaviors that may be caused by malware (Anomaly detection);
- intrusions into a smart supply chain network are detected, prevented, or mitigated (attack detection, prevention, or mitigation);
- ensuring that smart supply chain devices are tamper resistant and any unauthorized attempt to tamper with smart equipment or devices are detected (tamper resistance and detection);
- ensuring that smart supply chain network is proactively searched for threats that may be lurking undetected in the network (threat hunting);
- ensuring that all RESTful APIs are authenticated and authorized, and that request timestamp is added to prevent basic replay attacks (Secure API);
- ensuring that no entity can deny participating in any transaction and that any transaction can be traced back to the entities involved (NR, accountability);
- ensuring that both exterior and interior perimeters of smart supply chain facilities are physically secured (physical security).

3.2.1.8 Smart Wearable Security Requirements

The security requirements of a typical smart wearable system (e.g., a fitness tracker) should include ensuring that:

- information is protected from disclosure to unauthorized devices or individuals and that privacy of users is protected such that no sensitive personal information are viewed by any unauthorized entity (data confidentiality, user privacy);
- no unauthorized entities are allowed to access any resource on smart wearable devices or networks and that messages should be verified to ensure that they come from the claimed sender, and that they have not been changed (authenticity);
- an authorized user is not denied access to any device or service (availability, reliability);
- entities can only have access to the data they are permitted to access (authorization)
- accuracy, trustworthiness and consistency of data in transit is maintained throughout the life cycle of a message (data integrity);
- smart wearable devices are not physically accessible to unauthorized users (physical security).

3.2.1.9 Smart Farming Security Requirements

The following security requirements are critical to securing a typical smart farming system, that:

- no unauthorized entities are allowed to have access to any smart farm equipment or automated machineries and that farm sensors should be able to verify the origin and credibility of every message (authenticity);
- entities can only have access to the information and resources necessary for their normal operations (authorization);
- there is no disclosure of information to unauthorized parties (data confidentiality);
- no part of data in transit is altered or deleted (data integrity);
- authorized staff, farm devices, or smart equipment can access information of a resource when needed (availability, reliability);
- farm devices are resistant against tampering and that any attempt to tamper with a smart farm device or equipment is detected (tamper resistance and detection);
- any transaction can be traced back to the entities involved (Accountability);
- ensuring that all RESTful APIs are authenticated and authorized, and that request timestamp is added to prevent basic replay attacks (Secure API);
- no farm devices or equipment are physically accessible to unauthorized users (physical security).

Table 3.1 summarizes the security requirements for the nine application domains outline above. To make Table 3.1 a bit compact, the word "smart" was omitted in the names of the application domains and the following abbreviations or words were used to represent the full names of some of the application domains: Transpt = transportation, Health = healthcare, Manufc = manufacturing, Supply = supply chain, Wear = wearable, and Farm = farming. Similarly, these abbreviations were used in some security requirements: Det = detection, Pre = prevention, Mitg = mitigation, and Resis = resistance.

3.2.1.10 Security Mechanisms for Achieving some Security Requirements

To achieve some of the security requirements outlined in Subsections 3.2.1.1 to 3.2.1.9, there exist several IoT security mechanisms and schemes that have been proposed by different researchers in academia, industry, and government agencies. A few of these security mechanisms and schemes are:

Table 3.1: Summary of security requirements for the nine application domains.

Security requirements	Application domains								
	Home	Grid	City	Transpt	Health	Manufc	Supply	Wear	Farm
Data Confidentiality	✓	✓	✓	✓	✓	✓	✓	✓	✓
Data Integrity	✓	✓	✓	✓	✓	✓	✓	✓	✓
Availability	✓	✓	✓	✓	✓	✓	✓	✓	✓
Authenticity	✓	✓	✓	✓	✓	✓	✓	✓	✓
Authorization	✓	✓	✓	✓	✓	✓	✓	✓	✓
Device Resilience		✓	✓	✓	✓	✓	✓		
Network Resilience	✓	✓	✓	✓	✓	✓	✓		
Anomaly Detection		✓	✓	✓	✓	✓	✓		
Secure API	✓	✓	✓	✓	✓	✓	✓		✓
User Privacy	✓	✓	✓	✓	✓		✓	✓	
Threat Hunting		✓	✓	✓	✓	✓	✓		
Accountability	✓	✓	✓	✓	✓	✓	✓		✓
NR (Non Repudiation)		✓	✓	✓	✓	✓	✓		
Attack Det, Pre, Mitg		✓	✓	✓	✓	✓	✓		
Tamper Resis, Det		✓	✓	✓	✓	✓	✓		✓
Reliability	✓	✓	✓	✓	✓	✓	✓	✓	✓
Physical Security	✓	✓	✓	✓	✓	✓	✓	✓	✓

- *Lightweight encryption schemes* are data protection mechanisms needed to prevent attackers from accessing data from IoT systems, and hence ensure data confidentiality. Examples include PRESENT, Piccolo, SIMON, and SPECK [176];
- *Access control*: In the context of IoT, access control refers to the security mechanism that monitors, logs, limits or permits the actions or operations of a legitimate user or device in an IoT system, as well as defining a limit for programs executing on behalf of the legitimate user [177, 178];
- *Intrusion detection and prevention systems* based on AI such as machine learning can be used to detect and analyze both inbound and outbound IoT network traffic for abnormal activities [179];
- *Lightweight authentication mechanisms for IoT and other Machine-to-Machine (M2M) communications* are lightweight authentication schemes characterized by low communication as well as low computation and storage overhead, and can be used to achieve mutual authentication and session key agreement [180, 181];
- *Tamper resistance and data tampering detection systems* are schemes for detecting device and/or data tampering or modification attacks aimed at causing disruption and outages in IoT systems [182, 183];
- *Efficient and fast anti-collision algorithms* are needed for RFID [184] and hybrid systems where there is integration of sensor nodes and RFID devices [185]. Anti-collision algorithms can mitigate interference and prevent collision at the reader when multiple tags transmit simultaneously;
- *Secure booting*: This is a security feature that enables a device to check every software using digital signature or checksums, including the operating system, when the

device is first powered on, or whenever it restarts [186]. This will ensure that only authorized software run on IoT devices;

- *Secure updates*: This is a security feature that ensures that IoT systems authenticate security patches from operators using digital signatures so that patches cannot be intercepted, extracted, and modified, thereby preventing updating interfaces from turning into security holes themselves [187];
- *Key management mechanisms* are necessary at the data link-layer to allow two remote nodes to negotiate security credentials, such as secret keys [188, 189, 190];
- *Securing physical location*: Refers to putting physical obstacles like locks and fencing in the way of people with malicious intent, as well as device hardening against accidents and environmental disasters [191].

3.3 IoT Security and Privacy Best Practices

In Section 1.2 of this thesis, lack of sufficient security experience on the part of some manufacturers was outlined as one of the key factors that make IoT security and privacy challenges unique. Similarly, in Section 2.4, the need for IoT manufacturers to adopt the concept of security by design was emphasized, which underscores the need for manufacturers to strictly follow IoT security and privacy regulations and guidelines based on industry best practices.

The term *best practices* can be defined as a generally accepted set of guidelines, techniques, ethics or de facto standards that represent the most efficient course of action to be taken in a given situation for the purpose of complying with certain legal, ethical or technical requirements, or for the purpose of maintaining certain standard quality. Thus, IoT security and privacy best practices refer to a set of good practices for privacy and security of IoT, otherwise known as security and privacy guidelines or standards. Such standards are usually based on certain benchmarking and accredited by international standard-setting bodies such as the International Organization for Standardization (ISO).

3.3.1 Challenges of Developing a Commonly Agreed-upon IoT Security and Privacy Best Practices

As the scale of cyberattacks on the IoT increases, all players with a stake in IoT are increasingly seeing the need for all IoT device manufacturers and smart apps companies to ensure alignment of their security and privacy policies with security and privacy regulations and guidelines based on industry best practices [192]. Accordingly, different industries, regulatory agencies, and policymakers are considering setting a baseline for IoT security and privacy that will ensure data protection, public safety, and service continuity. However,

considering the diversity of IoT application domains and the different industries involved, what this security baseline should include, and how to monitor the implementation is still subject to debate [38]. Another issue is the seeming ambiguity in the classification of the term *data*, which can include data about people, functional data, collections of data, and intellectual property of companies. Furthermore, different countries have different regulations and laws as to what constitutes *sensitive data*, and how data in transit and in storage must be protected.

The aforementioned issues can be attributed to the fact that at the time of writing this thesis, IoT security and privacy standards landscape is mostly controlled and dominated by de facto standards developed by a few government institutes and industry alliances across the IoT ecosystem [38]. For example, a number of industry giants such as IBM (IBM Watson IoT), AT&T (American Telephone & Telegraph), Trusted Computing Group (TCG), and Cisco are deploying their different IoT security best practices [193]. Although in recent years there appears to be a significant degree of convergence towards common baseline specifications for IoT security and privacy, there exists limited collaboration between industry alliances, academic institutes, and government agencies, which is mainly due to considerable competition among industry giants and between some of these initiatives [194]. As a result, there exist many different IoT security and privacy guidelines and standards with a significant number of contradictions and duplication. This may be a problem for many IoT start-up companies since they may not know exactly the most appropriate one to adopt. Therefore, there is an immediate need to harmonize the existing IoT security and privacy guidelines and standards.

3.3.2 Attempts to Develop More Widely Accepted Security and Privacy Best Practices for the IoT

In response to the need for a concerted effort to develop more widely accepted good practices for security and privacy of IoT, a number of governing bodies and regulatory agencies have in the past decade enacted laws as well as developed different formal regulations and standards pertaining to IoT security and privacy [193]. However, enforcing compliance with many of these laws, regulations, and standards in the entire IoT industry sectors in all jurisdictions across the world is not possible. This is because many of the laws and regulations are still specific to particular geographic regions or application domains [193, 38]. The following subsections outline a few laws, regulations, and standards as they pertain to security and privacy in the IoT ecosystem.

3.3.2.1 IoT Laws and Regulations

In recent years, there have been concerted efforts around the world to ensure security and privacy as well as accelerate development across the IoT ecosystem through several policy actions [193]. The European Union (EU) suite of policy actions, for example, in-

cludes the creation of a number of alliances, regulatory documents, centers of expertise, and pilot projects. A notable example that is worthy of mention is the General Data Protection Regulation (GDPR) [195], a regulation that is aimed at protecting the privacy and personal information of users. While this regulation is not specifically targeted at IoT, the fact that IoT relies on the exchange, processing, and storage of huge amounts of data make IoT security and privacy issues to fall under the GDPR.

Focusing on regulations that target some IoT industry sectors, in November 2018, the European Union Agency for Network and Information Security (ENISA) has proposed *Good Practices for Security of the IoT in the context of Industry 4.0 and Smart Manufacturing* [196]. The results of this study is intended to be used across the EU member states to raise awareness about IoT threats and risks as well as to serve as a reference guide for security and safety in Industry 4.0 and Smart Manufacturing. Notable contributions of this study include: definitions of relevant terminologies; providing a comprehensive taxonomy of cyber assets across the Industry 4.0 and manufacturing value chain; providing a detailed Industry 4.0 taxonomy of threats based on related attack scenarios and risks; mapping of identified threats to cyber assets; and a detailed enumeration of security measures related to Industry 4.0 and smart manufacturing and mapping them against the above mentioned threats. The study also outlines security measures in three dimensions, namely policies, organizational, and technical measures.

Additionally, in November 2019, the ENISA has released two reports: *ENISA Good Practices for Security of Smart Cars* [197], which defines good practices for security of smart cars encompassing connected, semi-autonomous, and autonomous vehicles; and *Good Practices for Security of IoT - Secure Software Development Lifecycle* [198], with a particular focus on smart apps development. The first report is aimed at promoting cybersecurity for smart cars, and it is intended to raise awareness about threats and risks, as well as to serve as a reference guide for security and safety in this application domain. This study was conducted in response to the rapid pace of technological advancements in the area of smart cars, which pose serious security and privacy risks and potentially expand the attack surface of the IoT. Some major contributions of this study include: providing a detailed asset/threat taxonomy for smart cars; providing good practices that can improve the cybersecurity landscape in the domain of smart cars; and mapping of existing legislative, standardization, and policy initiatives so as to foster harmonization. The second report is based on the ENISA Baseline Security Recommendation study that focused on establishing secure development best practices across the IoT ecosystem. The purpose and contributions of this report are similar to the previous reports but with a focus on software development. Given that securing IoT software development is a fundamental building block for IoT security and privacy, the report provided security guidelines that may foster secure development of IoT products and services throughout their lifetime.

In the United States (US), California is the only state so far that has passed state laws regarding the future of cybersecurity. The first law, California Assembly Bill 1950 [199], which does not specifically target IoT, requires all businesses and their suppliers to maintain an appropriate level of cybersecurity. The second law, SB-327 [200], is the law that specifically targets the IoT industry. The law requires that manufacturers of IoT devices that will be connected to the Internet in California must equip their smart devices with appropriate security features. Such features must prevent unauthorized access, data exposure, and unauthorized data modification. The security features should also force users to set their own unique passwords during the initial setup of new smart devices, which can prevent attackers from guessing default passwords.

In the United Kingdom (UK), the government is considering a law that would require smart devices to be sold with labels that would indicate the vulnerabilities of those devices or security measures taken to protect IoT devices against cyberattacks. However, as at the time of writing this thesis, no IoT specific security laws have been passed yet. Nonetheless, the UK government has set up a National Center of Excellence for IoT Systems Cybersecurity, dubbed Privacy, Ethics, Trust, Reliability, Acceptability, and Security (PETRAS) [201]. The PETRAS is aimed at providing a national capability that will allow the UK to become a leader in IoT and associated systems security.

3.3.2.2 IoT Standards

As at the time of writing this thesis, the ISO has released five sets of standards that span across all aspects of cybersecurity. These standards are wide in scope and encompass security and privacy in cyberspace in general, and hence IoT can be considered to be an implicit part of the standards:

1. The ISO/International Electrotechnical Commission (IEC) 27001 standard is among the ISO/IEC 27000 family of standards designed to help organizations to keep information assets secure [202]. ISO/IEC 27001 specifies best practices for Information Security Management Systems (ISMSs) for the protection and preservation of information under the CIA triad model. The standard also provides a set of controls that organizations can apply based on expected risks;
2. The ISO/IEC 27032 standard specifies how to improve the state of cybersecurity [203]. It draws out unique aspects of activities and their dependencies on other security domains such as information security, network security, and Internet security. Although the controls recommended in this standard are not as prescriptive as those specified in the ISO/IEC 27001, this standard specifies attack vectors, as well as provided guidelines for safeguarding information beyond the borders of organizations;
3. The ISO/IEC 27035 is the standard for incident management [204], as it specifies a structured approach for detecting, responding, reporting, and assessing information security incidents. Cyber resilience is a crucial aspect of incident management,

therefore, this standard includes a planned approach to update security policies in order to strengthen existing security controls following the analysis of an event;

4. The ISO/IEC 27031 standard describes the concepts of ICT readiness for business continuity [205]. The standard also describes a framework that specifies performance criteria, design, and implementation for improving ICT readiness for business continuity of an organization;
5. The ISO/IEC 22301 is the standard for Business Continuity Management Systems (BCMSs) that targets all organizations irrespective of type, size, nature, and complexity [206]. The standard specifies protection against attacks and how to maintain continuity by specifying aspects of cyber resilience from disruptive incidents.

While there are no ISO standards that specifically focused on the IoT, in June 2019, the US National Institute of Standards and Technology (NIST) has released a standard, *NISTIR 8228* [207], for managing IoT cybersecurity and privacy risks. This document is intended to help federal agencies and other organizations better manage the cybersecurity and privacy risks related to their IoT devices. Although not a formal set of regulations, the document outlines how such organizations should manage the security and privacy of their IoT devices throughout the entire lifecycles of the devices. The document is intended to precede a series of IoT standards that would be released in the near future.

3.3.3 Security and Privacy Best Practices for IoT Manufacturers and Developers

This subsection outlines a series of security and privacy best practices for manufacturers and developers [208, 196, 197], which should be adhered to where possible.

3.3.3.1 Secure Boot

The process of secure boot is critical to the integrity of an IoT device. This is because the execution of a trusted boot sequence can prevent the execution of unauthorized code when the device is turned on. It can also provide the necessary degree of trust that other security features on the device may likely operate properly. The following are some best practices for IoT device secure boot:

- Use any of the following ways to ensure secure boot: digitally signed binaries, secure and trusted boot loaders, boot file encryption, or security microprocessors;
- To store important data, use tamper-resistant hardware-based storage such as TPM, microcontroller security subsystem, or Secure Access Module (SAM);
- Always ensure that each stage of boot code is trusted and valid immediately before running it;

- Check that only valid and expected hardware are present at each stage of the boot sequence;
- Ensure that no stage in the boot sequence is skipped;
- Ensure that there is a provision for safe remediation that checks the validity of firmware image in case of device failure or compromise.

3.3.3.2 Secure OS

Many IoT devices are based on basic OSes that are incapable of addressing specialized security requirements, therefore, protecting these OSes is very critical to the security of such devices. The following are a few best practices for IoT device secure OS:

- Ensure that the OS of an IoT device consists of only the necessary components such as libraries, packages, and modules that are needed for the functionality of the device;
- Make sure that IoT devices are only shipped with the latest and stable versions of OSes;
- Ensure that by default, smart devices are shipped with the proper OS security configuration in place;
- It must be ensured that OSes on IoT can boot securely;
- Continue to provide updates of OS components throughout the lifecycle of smart devices;
- Ensure that permissions are properly set such that users or apps cannot write to the root file system; and be sure that all protocols, services, and ports that are not needed are disabled.

3.3.3.3 Management of Security Credentials

Having access to security credentials such as passwords, encryption keys, and digital certificates can allow an attacker to easily gain unauthorized access to an IoT system or network. Below are a few best practices for IoT security credential management:

- Each IoT device should be uniquely identifiable using tamper-resistant factory-set hardware;
- Ensure the use of good password management techniques, such as encrypting passwords sent across a network, preventing the use of blank or default passwords, and permitting the use of non-alphanumerics along with digits and letters;
- Make sure there is a robust and secure password reset mechanism in place;

- Ensure that stored passwords are hashed and uniquely salted;
- Ensure that security credentials such as encryption keys are securely stored in TPM, SAM, or in a trusted key store;
- Endeavor to use a 2FA for accessing sensitive information.

3.3.3.4 Encryption Algorithm

While cryptographic algorithms can consume a significant amount of battery power and computing resources such as memory and Central Processing Unit (CPU) time, it is recommended to use the strongest lightweight cryptographic algorithms available for encrypting sensitive data for IoT applications. For more details on this subject, see Section 3.4, however, a few best practices for IoT encryption are presented below:

- Ensure that only the most recent versions of industry-standard lightweight ciphers are used; do not use algorithms and protocols that are not vetted by the cryptographic community;
- Be very careful when selecting and implementing cryptographic algorithms because a cryptographic algorithm is only as strong as how it is implemented;
- Use the appropriate level of encryption in proportion to the sensitivity of data being processed;
- Ensure that encryption keys are securely stored in TPM, SAM, or in a trusted key store;
- Avoid using insecure protocols such as File Transfer Protocol (FTP) and Telnet, which relies on clear-text usernames and passwords for authentication;
- Avoid using the same keys or global keys when implementing encryption on IoT devices. Each device should have its own unique key;
- Ensure that certificates are properly validated against the hostnames whom they are meant for;
- To reduce the risk of compromising many servers, avoid using wildcard certificates unless there is a business need for it. Only store sensitive data that you need;
- If a password is being used to protect keys then the password strength should be sufficient for the strength of the keys it is protecting;
- Use cryptographically strong random numbers for cryptographic applications such as the generation of keys, nonces, and salts in certain signature schemes;
- Make provisions for securely remote replacement of encryption keys.

3.3.3.5 IoT App Security

It was already mentioned that smart apps can intercept or capture a variety of personal information, which can provide attackers with the opportunity to have access to such information. The following are some best practices that can help developers to bake security into the design of their smart apps:

- Ensure that smart apps operate with the lowest level of privilege and not as root. They should also be given access to only the necessary resources needed for their normal operations;
- Isolate smart apps from one another using secure computing mode (seccomp), a Linux kernel facility that restricts how a process makes system calls. Other methods of isolation can include containerization and virtual machines;
- Bake security into every stage of smart apps development lifecycle, and document the stages of the security design;
- Use secure coding practice, a coding technique that guards against accidental introduction of security vulnerabilities. Also, avoid buffer overflow, use only strong lightweight encryption ciphers and secure protocols, and validate input data before processing;
- Handle errors carefully and ensure that error logs and other messages do not reveal sensitive information;
- Do not use default usernames and passwords, and force users to choose reasonable passwords that will be difficult for attackers to guess;
- Do not deploy debug versions of code and ensure that code comments, compilers, and other superfluous files that can allow attackers to reverse engineer the code are not included;
- Ensure that OS and libraries are the most recent and stable versions.

3.3.3.6 IoT Physical Security

Many IoT edge devices are often deployed in open fields or environments where it is virtually impossible to protect them physically. As a result, a lot of these devices are exposed to various forms of physical attacks such as tampering and side-channel attacks. The following are a few best practices for ensuring physical security in the IoT:

- Disable every Test Access Port (TAP) like Joint Test Action Group (JTAG) which can introduce hardware security risks. This is because a functioning JTAG port can serve as a backdoor through which attackers can introduce a false input/output signal. Most chipmakers, however, do provide a way to disable the JTAG before the software is finally ready for release;

- In the same vein, disable all interfaces installed for test or administration purposes, or make them physically inaccessible;
- Use secure protocols and strong access control mechanisms to protect administration ports left for remote managements that are deemed necessary for normal operations;
- If possible, protect IoT device circuitry from tampering using resin encapsulation, epoxy chips to the circuit board, etc.;
- Ensure that smart devices that will be deployed in an open environment are securely protected using strong casing;
- Use processor intellectual property that includes side-channel resistant capabilities. Careful design can also obfuscate the information that SoC timing, power consumption, and electromagnetic radiation can inadvertently reveal.

3.3.4 Security and Privacy Best Practices for Enterprises and Individual IoT Users

While manufacturers and developers have very significant roles to play in ensuring security in the IoT, enterprises and individual users should also be active participants in the security of their information [33]. Connecting new IoT endpoints to corporate, city, home, and personal networks can create potential entry points for hackers and cybercriminals that must be blocked. Although enterprises may have comprehensive security policies, the actions of individual employees play a big role in helping the organizations to implement those policies. For example, the malicious intent, carelessness, or mistake of a single employee could lead to a serious data breach [209]. The following are some of the best practices for enterprises and individual IoT users (some of them are commonplace nowadays, yet included for the sake of completeness):

- Never use a device default password and always use a strong password that should be a mixture of upper and lowercase letters, numbers, and special characters;
- Keep your smart devices updated: always check manufacturers websites regularly to see if new patches have been released;
- Isolate IoT devices on separate networks and avoid connecting them on the same network with mission-critical devices;
- If possible, always use 2FA for authentication to add another layer of security;
- Turn off all default features that you do not need. For example, disable the UPnP protocol, a protocol that allows networked devices to discover other devices on the network, which can allow attackers to track IoT devices;

- Beware of phishing emails and avoid pop-ups and unknown links, which could lead to a security breach;
- Be suspicious of any official-looking email that asks for your password, financial or personal information.

3.4 An Overview of Lightweight Cryptography and Lightweight Cryptographic Algorithms

Pervasive computing technologies are enabling all sorts of *things* to connect to the Internet, some of which are characterized by limited energy as well as computational and storage resources. What usually comes to fore when implementing security in such constrained environments are memory requirements, circuit area, and energy drain of the primitive to be implemented. Therefore, as highlighted in Section 1.2 and Subsections 2.1.2.1, 2.4.1.4, and 3.3.3.4, standard algorithms are often prohibitively expensive for implementation in such environments.

While there are no strict criteria for classifying a cryptographic algorithm as lightweight, LWCAs must fulfill certain criteria. They should: (1) provide an acceptable level of security, (2) consume less CPU time, (3) have extremely low memory requirements, (4) consume very low power, and (5) occupy a very small circuit area on hardware. Lightweight cryptography can, therefore, be defined as specialized cryptography with good performance, tailored for devices that feature low computational capabilities, less memory, and/or small area footprint. The following subsections discuss different types of LWCAs, LWC design and implementation considerations, and LWC for IoT.

3.4.1 Types of Lightweight Cryptographic Algorithms

The demand for secure, efficient, low-energy, and yet implementable cryptographic primitives that can fit the most constrained environments is rapidly increasing. Thus, in recent years, significant progress has been made in addressing this fundamental issue through different LWC research initiatives [210, 176, 211]. Today, there are many types of lightweight symmetric ciphers as well as a few of their asymmetric counterparts. Lightweight symmetric ciphers comprise of block and stream ciphers. Other LWCAs include hash functions, Message Authentication Code (MAC), and Authenticated Encryption (AE) algorithms.

3.4.1.1 Lightweight Block Ciphers

Block ciphers are the most versatile of the symmetric ciphers, and based on algorithm structure, they can be classified into two basic categories, namely, Substitution-Permutation Network (SPN) and Feistel Network (FN).

The inputs to the SPN are each block of the plaintext and the key, however, many round keys are derived from the main key which are used in the operations [212]. To produce the ciphertext block, SPN applies several rounds of substitution boxes (S-boxes) as well as permutation boxes (P-boxes) in an alternating manner, also called substitution and permutation functions, to the plaintext and the round keys [213]. The substitution (i.e., replacement of certain number of bits with other according to some rules) and permutation (i.e., manipulation of the order of bits using some algorithm) functions provide confusion and diffusion, respectively. While diffusion prevents attackers from deducing any key by creating complex statistical relationship between ciphertext and plaintext, confusion creates intricate relationship between the encryption key bits and ciphertexts, making deduction of key bits extremely difficult [214].

These operations are typically achieved through Exclusive-OR (XOR) and bitwise rotation, which are efficient and easy to implement in hardware. To decrypt an encrypted message using SPN, the encryption rounds must be reversed by reversing the S-boxes/P-boxes, and also the order of the application of the round keys must be reversed [212]. The structure of SPN provides good resistance against most cryptographic attacks. It also has more inherent parallelism, and hence algorithms based on SPN can be faster in software on high-end devices with multiple processing units than their FN counterparts. However, low-end devices with few processing units cannot benefit from the inherent parallelism.

Like in SPN, the plaintext in FN is processed in a number of rounds to get the ciphertext, and the number of rounds required determines the number of round keys (i.e., subkeys) which must be generated from the master key. However, in Feistel structure, the plaintext input block is divided into left (L) and right (R) halves, and an encryption function, f , is applied only to the R half [213]. For example, in the first round, the function takes the round key K_i and R_{i-1} as inputs and produce the output $f(K_i, R_{i-1})$. This output is then XORed with L_{i-1} , where $i = 1, 2, 3, \dots, n$. The last step of each round is the permutation step which simply swaps L and R , such that the R for the next round would be the L for the current round. Similarly, the L for the next round would be the R for the current round. The process of decryption in FN is almost identical to that of the encryption, the only difference being the reversal of the round keys used in the encryption [215].

Some advantages that make the FN a better choice for lightweight cipher design are that the encryption and decryption functions are almost the same, and that the round function is only applied to half the input block. This implies that FN performs less computation than the SPN, and hence ciphers based on the FN can be implemented with nearly half the size of code or circuitry needed in SPN [215]. Nevertheless, the disadvantage of the scheme for lightweight applications is that it requires many number of rounds to make the algorithm complex and hence secure, which translates to more time to encrypt and decrypt.

The strength of FN depends on the following parameters: number of rounds - the more the number of rounds, the more security; subkey generation algorithm - the complex the algorithm, the more difficult for cryptanalysis to generate the secret key; and round function - the complex the function, the greater the resistance to cryptanalysis. Other parameters are: block size - the larger the block size, the more the security; and key size - the larger the key size, the more the security. The design of FN depends on the number of rounds, subkey generation algorithm, and the round function used.

One example of both conventional and lightweight ciphers based on the SPN are Advanced Encryption Standard (AES) and PRESENT, respectively. Similarly, an example of both conventional and lightweight ciphers based on the FN are Data Encryption Standard (DES) and CLEFIA, respectively.

3.4.1.2 Lightweight Stream Ciphers

Lightweight stream ciphers are symmetric-key ciphers that encrypt one bit or one byte of plaintext at a time. The design of stream ciphers is similar to the ideal cipher, also known as One-Time Pad (OTP), the only cipher that is *perfectly secure* and fully immune to brute force attacks [216]. This is because the key used in OTP is truly random, and it is at least as long as the message. Stream ciphers use an infinite keystream of unpredictable (in feasible time) pseudorandom bits generated from a secret key (K), the only part that should be truly random in this algorithm, and an initialization vector (IV). The keystream generator is typically based on Linear Feedback Shift Registers (LFSRs) and Non-linear Feedback Shift Registers (NFSRs) [217]. To encrypt a message P , 1 bit of the keystream is XORed with 1 bit of the plaintext to generate 1 bit of the ciphertext. For instance, given 1 bit of plaintext P_i , 1 bit of ciphertext C_i , and 1 bit of keystream K_{s_i} , encryption and decryption can be represented as $C_i = E_{K_{s_i}}(P_i) = P_i + K_{s_i} \bmod 2$ and $P_i = E_{K_{s_i}}(C_i) = C_i + K_{s_i} \bmod 2$, respectively. In the previous mathematical expressions, E represents an encryption/decryption function, which in the case of stream ciphers is modulo 2, i.e., $P_i, C_i, K_{s_i} \in \{0, 1\}$.

Modulo 2 addition is used since it is equivalent to the XOR operation. From the second mathematical expression it can be seen that the decryption process is very simple, because it is the same XOR operation. It is very important that the key be reproducible by the communicating parties. The security of this scheme depends on the keystream, hence the keystream should be very random. A good statistical property for ciphertext is that for a perfectly random keystream bit K_{s_i} , i.e., $P(K_{s_i} = 0) = P(K_{s_i} = 1) = 0.5$, a ciphertext output bit C_i should have a 50% chance of being 0 or 1.

Stream ciphers can be divided into synchronous and asynchronous stream ciphers [218]. In synchronous stream ciphers, the keystream depends only on the key. However, in

asynchronous stream ciphers the keystream depends both on the key and the ciphertext. Lightweight stream ciphers are more compact, and are designed to minimize space and computation time, and thus faster in hardware and software compared to block ciphers. One example of both conventional and lightweight stream ciphers are RC4 and chacha20, respectively.

3.4.1.3 Lightweight Cryptographic Hash Functions

A hash function is any function that takes an input string of arbitrary-length and outputs a fixed-size of alphanumeric string, usually called digest, hash fingerprints, or hash values. Formally, it can be defined as an efficient and deterministic algorithm $H : P \rightarrow D$ that maps a given input from a finite arbitrary-length message space $P = \{0, 1\}^*$ to an output within a finite fixed-length space $D = \{0, 1\}^n$ (i.e., a finite set of bit strings). Hence, $H : \{0, 1\}^* \rightarrow \{0, 1\}^n$, where $*$ represents an arbitrary input size, and n the size of hash value [219]. On the other hand, a cryptographic hash function is a special type that in addition must have certain properties in order to be considered secure and suitable for cryptography. The ideal cryptographic hash function properties are: (1) *Pre-Image Resistance*: given a digest d , it should be computationally infeasible to find a message p such that $d = H(p)$; (2) *Second Pre-Image Resistance*: given a digest d and a message p such that $d = H(p)$, it should be computationally infeasible to find another message p' (where $p' \neq p$), such that the two messages hash to the same digest, i.e., $d = H(p') = H(p)$; and (3) *Collision Resistant*: it should be computationally infeasible to find two different messages with the same hash, i.e., computationally infeasible to find p, p' such that $H(p) = H(p')$ [219].

Most modern hash functions are designed by constructing a collision resistant compression function that takes a fixed-length input and outputs a shorter fixed-length output, and then extending its input domain by applying some composition scheme iteratively [220]. Hence, the two fundamental building blocks are: a compression function and domain extender. A compression function can be derived from a block cipher or a permutation, or designed from scratch. Popular methods of constructing compression functions from traditional block ciphers include Davies-Meyer, Matyas-Meyer-Oseas, and Miyaguchi-Preneel.

However, since hash functions are expected to take arbitrary-length inputs, there is a need to securely extend the domain of the compression function, which can be achieved by using composition principles or iterative techniques. Thus, a domain extender (also called a composition scheme) is a generic construction that transforms a given compression function with fixed-length input into a function with arbitrary-length input [221]. The iterative structures in the composition schemes allow for a sequential message processing. Today, most of the iterative techniques used in cryptographic hash function designs are based on the Merkle-Damgård (MD) or Sponge constructions. Many standard cryptographic

hash functions are based on the MD construction (or its variants since there have been successful attacks on the MD construction) iterating a Davies-Meyer as their underlying compression function.

Typically, hash function footprint is largely determined by the number of internal state bits (as well as the key schedule for block cipher based designs), which dominates the total area requirements of hash functions. For example, the internal states for SHA-3 is 1600 bits [222]. Thus, a major challenge in designing lightweight hash functions is how to strike a perfect balance between security and memory requirements. One option is to build a domain extender on top of a lightweight block cipher using Davies-Meyer or other constructions. But it has been shown in [223] that this is not an optimal solution for reducing the footprint. However, it has been demonstrated convincingly in [224, 225] that by using permutation-based Sponge construction, the state size can be reduced to almost halve.

Consequently, the design of lightweight cryptographic hash functions is mostly based on the Sponge construction [226]. The sponge construction depends on a specific-length permutation (or transformation) as well as a strict padding rule to ensure mapping variable-length input to variable-length output. A sponge function takes a binary string of any length as input, and returns a binary string of any desired length, i.e., it takes input element of \mathbb{Z}_2^* and returns an element of \mathbb{Z}_2^n , where n is a user-specified value.

Cryptographic hash functions are major targets for cryptanalysts as they constitute essential building blocks for many security applications [227]. For example, they are used to achieve some security goals, such as authenticity; they are also used for password protection, digital time stamping, certificate revocation management, etc. One example of both conventional and lightweight cryptographic hash functions are SHA-3 and Keccak, respectively. The cryptographic hash functions discussed so far belong to the *Keyless* category, which produces an output that depends only on the input message. The other category is known as *Keyed* cryptographic hash function, which produces an output that depends both on the input message and a key, also referred to as MAC.

3.4.1.4 Lightweight Message Authentication Code (MAC)

Like the *Keyless* hash functions, MACs are non-invertible functions that are used to produce a fixed-length output from an arbitrary-length input message. As highlighted above, the main difference between the *Keyless* hash functions and MACs (i.e., *Keyed* hash functions) is that a MAC also takes a secret key K as an input in addition to the message P to be authenticated. Using the same notations as in the case of the *Keyless* version, a MAC can be defined formally as a deterministic algorithm $H : P \times K \rightarrow D$ that maps a given input from a finite arbitrary-length domain space P and a given key from a finite fixed-length key space K to an output in a finite fixed-length space D , where $K = \{0, 1\}^k$. Hence,

$$H : \{0, 1\}^* \times \{0, 1\}^k \rightarrow \{0, 1\}^n.$$

MAC functions are used as cryptographic checksums for authentication and message integrity checks [228]. The scheme consists of a tag-generation and tag-verification algorithms. For example, assuming the sender and receiver have securely shared a secret key K , the sender uses the tag-generation algorithm to compute a tag t (also called checksum) for a message P (using the secret key K) and sends both P and t to the receiver. Upon reception of the message (which may have been tampered with en route), the receiver runs the tag-verification algorithm to check the message-tag pair using the same secret key. If the check is successful, the algorithm returns 1, and the receiver accepts the message; otherwise it returns 0, and the message is rejected.

Formalizing the above descriptions: (1) let a key-generation algorithm GEN take a security parameter 1^n and produce a uniformly distributed secret key K , where $|K| \geq n$, i.e., $K \leftarrow GEN(1^n)$; (2) a given tag-generation algorithm MAC takes as an input a secret key K and a message P , and returns a tag t such that $t \leftarrow MAC_k(P)$; and (3) a given tag-verification algorithm VER takes a secret key K , a message P and a tag t , and returns 1 if $t = MAC_k(P)$, otherwise it returns 0 [228]. Note, however, that the tag-verification algorithm simply runs the tag-generation algorithm on the received message and the secret key of the receiver, and compares the newly generated tag with the one received.

In the asymmetric key cryptography setting, the MAC counterpart is known as a digital signature, which in addition to message integrity verification, also verifies the identity of the sender. A digital signature also has two algorithms: a signature generation algorithm and a verification algorithm [229]. Thus, the sender signs the message with his/her private key, and the receiver verifies the message using the public key of the sender.

In the case of MAC, since all communicating parties have the secret key, any party can produce and verify a tag, meaning that a given tagged message could have been produced by any of the parties. On the contrary, in the case of digital signatures, since every party has a unique public and secret key pair, once a party signs a message with his/her private key, it remains bound to the party, since (presumably) no one else knows his/her secret key [229]. Hence, apart from authentication and message integrity, digital signatures also provide another important security service known as non-repudiation - a property that ensures that a participant cannot later deny taking part in a particular action.

There are different lightweight approaches to deal with authentication and corrupted message issues in the IoT. For example, embedding digital certificates, signed by the manufacturer, into devices can solve authentication and key exchange problems in some IoT applications [230]. However, there are many devices that cannot cope with this approach. This is because digital certificates rely on public key cryptography, which requires more mem-

ory due to longer keys, and may be computationally expensive for some devices [231]. A lighter alternative is lightweight Cipher-based Message Authentication Code (CMAC) [232]. Lightweight CMAC algorithms can be built using block ciphers, provided these underlying block ciphers are lightweight themselves. This is demonstrated in [233], where the lightweight block ciphers SIMON, SPECK, and LEA were used as the underlying primitives for a lightweight CMAC. One example of both conventional and lightweight MAC algorithms are CBC-MAC-DES and SipHash, respectively.

3.4.1.5 Lightweight Authenticated Encryption (AE)

Arguably, encryption (used for providing confidentiality for a message) is the most widely known security concept, but it lacks the mechanism to check the authenticity of the message. However, in practice, the two security goals (i.e., confidentiality and authenticity in the sense of data origin authentication) are often desirable [234, 235], since encryption that is not accompanied by message authentication has limited value in some applications. To achieve these two security goals, two separate algorithms are usually implemented. However, implementing two security algorithms on some resource-constrained IoT devices can drain the limited resources available such as energy source, code space or silicon area, and hence the need for lightweight AE algorithms. An AE is a symmetric key cryptographic primitive that simultaneously provides message confidentiality and authenticity under a single key [235].

Authenticated Encryption with Associated Data (AEAD) additionally provides the ability to check the integrity and authenticity of additional information called Associated Data (AD), such as packet header which is a public data that travels alongside the ciphertext and must be authenticated with the ciphertext [236]. An AEAD scheme takes plaintext P , initialization vector IV or nonce, and AD (which we refer to here as a header H) as arguments. P is encrypted by the underlying cipher using a secret key K , producing a ciphertext C . C is authenticated along with H yielding a tag T , truncated to a suitable length. The output, which consists of C appended to T , $C||T$, can be decrypted and verified with the same secret key K used in the encryption process [235]. Note that in order to decrypt correctly, AD must be sent along with tagged-ciphertext. To minimize waste of resources, especially in the case IoT, ciphertext authentication is preferred over plaintext authentication, so that invalid messages can be discarded earlier by simply checking the authentication tag without having to decrypt the whole message.

Generic composition of encryption and MAC algorithms is the classical method for achieving data confidentiality and authenticity. In the past few years, for example, the variants of generic composition schemes *Encrypt-and-MAC*, *MAC-then-Encrypt*, and *Encrypt-then-MAC* [237] were used in the old versions of these important security protocols, namely SSH, TLS, and IPsec, respectively. Besides being non-optimal and inefficient (since the scheme needs to process the message twice), this approach is difficult and may be prone

to implementation errors [236]. Therefore, several efficient and secure AE designs have been proposed in recent years, including Galois/Counter Mode (GCM), Counter-with-CBC-MAC (CCM), and Offset Codebook Mode (OCB). In addition, the sponge functions proposed lately promise to be useful building blocks for resource-efficient types of AE algorithms. One example of conventional and lightweight AE algorithms are OCB and ACORN, respectively.

3.4.2 Lightweight Cryptography Design and Implementation Considerations

Lightweight cryptography targets a diverse range of IoT devices in different applications domains, including implantable medical devices, wearables, vehicle security systems, and smart meters. The diversity of IoT devices ranges from a simple standalone sensor device enabling telemetry (i.e., gathering data from remote places) to a complex machine participating in an industrial process. Typically, different IoT devices utilize different communications protocols, and are powered by a variety of energy sources, such as disposable or rechargeable batteries, or renewable energy sources. Accordingly, the security requirements and the manner in which security is designed and implemented in these devices differ drastically [238]. For instance, while there are IoT devices that can handle any lightweight security algorithm, the extremely resource-constrained devices can only afford to devote a small portion of their circuit area and/or computing power to security. Thus, an algorithm that can take up much of the scarce space on such devices is not a viable candidate.

It is important to note that providing security is not the main purpose of IoT devices [176]. As a matter of fact, security is an overhead on top of the actual functionality of a device, and if not properly designed and/or implemented, it may even hinder the intended operations of the device or application. Therefore, the design and implementation of LWCAs deserve special attention, especially considering that optimizing the three design goals (i.e., security, cost, and performance) in the IoT setting is very difficult [210].

Notwithstanding the implementation difficulties, LWCAs can be implemented both in software and hardware. There are, however, important considerations to take into account when implementing LWCAs: (1) block cipher choice: depending on the application, a very short key and/or a very short block algorithms may not provide the desired level of security. This is because the cryptanalytic strength of most symmetric key ciphers lies in their key length, therefore, a very short key length may increase the probability of key-related attacks. Similarly, while block size does not play an important role in the security of a cipher, the security of a mode of operation (since block ciphers are often used in modes of operation) depends on the block size of the cipher in addition to the security of the underlying cipher. In Cipher Block Chaining (CBC) mode, for example, a very short block size can cause collision problems. For instance, if block size is n bits, then according to

the birthday paradox, there may be a collision after about $2^{n/2}$ block encryptions. Hence, if $n = 32$, there may be collision after about 2^{16} block encryptions [239]; (2) good security and good performance are at odds with each other. Although this trade-off is not peculiar to LWC, it is especially more critical in this context because of the aforementioned constraints related to many IoT devices; (3) the choice of an implementation option should be based on hardware capabilities, as well as on the application; and (4) any implementation option should target an optimum use of resources.

The following subsections provide detailed discussions on both software and hardware design and implementation considerations, as they often have different and sometimes conflicting properties and demands. For example, while bit permutation is a trivial task in hardware which can be easily achieved by simple re-wiring, in software it is usually a very difficult task to perform. In contrast, implementing a substitution table is not a trivial task in hardware, but it can be easily implemented in software [240]. Thus, lightwightness in software does not necessarily mean lightwightness in hardware and vice versa.

3.4.2.1 Software Considerations

The primary design goals for software implementation of LWC are to minimize memory requirements, maximize throughput, and to optimize power consumption of a cipher. The memory mainly consists of Random Access Memory (RAM) and Read-Only Memory (ROM). Analyzing memory usage is fundamental, as it provides useful information on the implementability of a security algorithm in a constrained environment.

Methods of cipher implementation vary according to the device and the application. Where speed is not so important, an algorithm can be implemented in a high-level programming language, such as C, C++, and Java. But many cryptographic engineers prefer the portability of C code to other high-level programming languages [241], and the fact that it allows the programmer to have a lot of control over memory usage. As a result, the C implementations of many cryptographic algorithms are readily available. For higher speed applications, however, an assembly language, or machine language (which is very tedious, error-prone, and rarely used today) can be used.

The core performance metrics often used for evaluating software design and implementation of LWCAs are code size (ROM), RAM usage, execution time, and energy consumption [241, 242]. The design and implementation of IoT devices typically require a careful control of the limited resources available, including the scarce memory. By leveraging some special features in device memory and C/C++ programming, data can be allocated in memory in such a way that would allow one to manage and control certain allocated data properties, such as size, scope, location, access, and lifetime. This will greatly help in optimizing the design and implementation of cryptographic algorithms, since knowing how data is allocated in physical Microcontroller Unit (MCU) memory is of utmost im-

portance in optimizing performance and speed. Therefore, as a background to memory usage, a high-level description of some aspects of memory layout of embedded C programs is provided in the following paragraphs.

Most MCUs today are based on the Harvard architecture, and unlike in the Von-Neumann architecture [243], the program memory and data memory on the Harvard architecture are separate memories, as shown in Figure 3.1. The compiler produces relocatable blocks of code ¹ and data called segments, which are allocated into memory in a variety of ways to conform to different system configurations. These memories are also known as the program and data segments, which can be simply referred to as the flash and RAM, respectively. These blocks of memories map compiled segments of a program to a physical address space through the linker file. The program memory, which is non-volatile, stores the compiled program and a small part of the data. It is further divided into sub-segments, which includes the `.text`, `.const` (or `.rodata` in some architectures), and `.cinit/.pinit` sub-segments. The text section (which is usually the largest) stores the written program, including the main function, user defined functions, and standard library code. The const/read-only section stores constant variable data, which is difficult to overwrite at runtime. Its size depends on software implementation. The compiler on a Harvard architecture based MCU uses special instructions to access constant variables. The `.cinit/.pinit` (the names depend on architecture, compiler, or C/C++ standard) are the initialization sections where initialized data values are stored (e.g., global initialized variables), and are loaded into the data memory at start up. Their sizes also depend on software implementation.

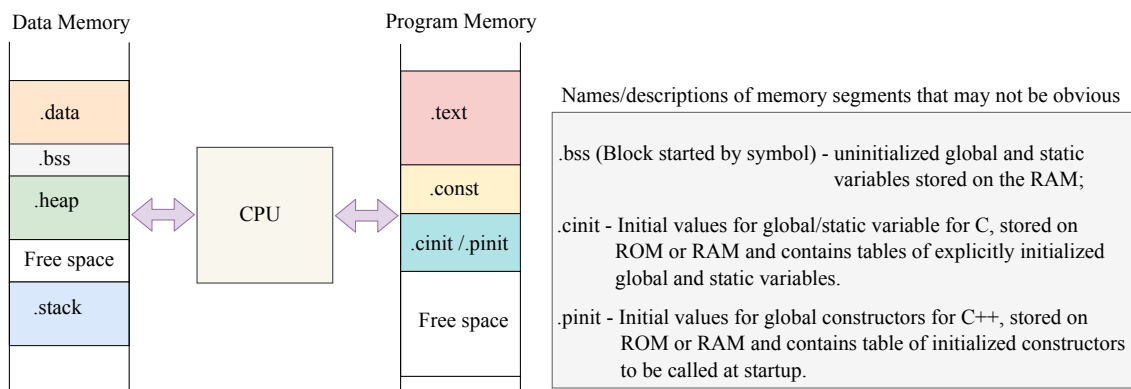


Figure 3.1: A high-level representation of a typical Harvard based MCU memory segments (adapted from [43]).

The data memory, which is usually volatile, stores the program operands like the variables on which the program executable operates on. Its content changes regularly throughout the program execution as data is loaded into the CPU registers, and the results are stored

¹Code whose execution address can be changed, and often used for dynamic linking and loading. Note that during the link process, the linker corrects all address references to the appropriate execution values.

back into the memory. The data memory is also divided into sub-segments, which include `.data`, `.bss`, the heap, and the stack. The size of each sub-segment depends on how the program is written. The data section stores non-zero initialized global and static data (including static local variables). The data in this section is allocated at compile time, and will persist in memory until the end of the program. The `.bss` region stores zero initialized and uninitialized global and static data. The heap is used to store dynamically allocated data. While a heap space is reserved at compile time, data is allocated at runtime, and lifetime of data is directly managed by the programmer (usually longer than a function, but less than the program), hence memory can be used over and over again. The `malloc`, `calloc`, `realloc`, and `free` functions are used to allocate data, reallocate data, and free a heap memory space, respectively [244]. Note that it is very important to free the heap memory that is no longer needed by the program. The stack stores most local variables and temporary data. Like the heap, memory space is reserved at compile time but data allocation is done at runtime, and data will only exist for the length of a function or block of code, meaning that the memory can be reused by other functions. Hence, the stack shrinks and grows as the program runs. However, writing a program that exhibits a large data size/type (such as many local variables, input parameters, return data, nested interrupts, and nested subroutines) can potentially overflow the allocated stack region [245]. This is because reserved space on the stack is of specific size determined at compile time.

Having highlighted the memory layout of embedded C programs, below we outline the four aforementioned performance metrics:

- *Code Size*: The code size is the binary program size representing the program footprint stored in the flash memory (program memory), which is usually measured in bytes. The total code size is obtained by adding the `.text` section of the program memory and the `.data` section of the data memory (since the `.cint/.pint` sections of the program memory are loaded into data memory at runtime). Sometimes the size of the `.bss` is not considered, for example, in the Fair Evaluation of Lightweight Cryptographic Systems (FELICS) framework [241], the `.bss` section is not considered since the framework forbids the use of global uninitialized variables;
- *RAM Usage*: The RAM usage is divided into `.data` usage and stack usage. The `.data` may contain the data to encrypt, round keys, master key, or initialization vectors [241]. However, the RAM is not for storing large fixed data such as Look-Up Tables (LUTs). Large data like LUTs should be allocated on the flash to save the valuable space on the RAM. The stack may contain local variables, return addresses after interrupts, and subroutine calls;
- *Execution Time*: Although the speed of an algorithm on a MCU is related to its performance [242], achieving high speed at the cost of high overhead due to complex key schedule would be counterproductive. Therefore, latency must be taken into consideration as well [246]. In addition, there is need to find acceptable ways to

enhance performance, as well as ways for evaluating the swiftness of execution time of an algorithm. While there is no standard speed measurement procedure that cuts across all hardware platforms, a general approach to determining execution time of an algorithm would be to calculate the absolute difference between the system timer number of cycles at the end and the beginning of a given operation;

- *Energy Consumption:* Despite the technological advancements in battery technology, there is no commensurate improvement in energy storage capacity, especially for very small footprint devices [247]. As a result, energy consumption still remains a major constraint in the IoT. While transmission overhead is a major source of energy drain, intensive computations, such as execution of cryptographic algorithms can also cause excessive energy drain in resource-constrained devices. Thus, energy efficiency should be a key consideration in the design of usable security solutions for this space. To determine the magnitude of consumed energy in a device, there is a need to obtain the power (i.e., the rate at which energy is consumed). Thus, power consumption can be calculated by multiplying the device current consumption by the operating voltage, i.e.,

$$P_{(W)} = I_{(A)} \times V_{(v)}, \quad (3.1)$$

where P is power, I is current, and V is voltage. Hence, energy consumption (E) is equal to the product of the average power consumption and the operation time (T), i.e.,

$$E_{(J)} = P_{(W)} \times T_{(s)}. \quad (3.2)$$

Therefore, to optimize energy consumption, designers must strive to minimize power consumption and/or reduce execution time.

3.4.2.2 Hardware Considerations

While software implementations of lightweight cryptosystems are inexpensive (since they utilize existing system resources), more flexible, easy to deploy, as well as easy to upgrade and update, they are relatively slower in comparison to their hardware counterparts. They also provide a very low level of protection for crypto variables such as crypto-keys. The hardware implementations, on the other hand, are much faster [217], typically by several orders of magnitude. Additionally, many hardware security solutions are tamper-resistant, and hence compromising cryptographic security parameters such as retrieving secret keys would be much harder. In fact, in some hardware solutions, such attempts will trigger the module to delete its internal memory. Another important aspect is that hardware implementation allows designers to implement the exact desired functionality

without redundant components [248], and hence they are typically considered more suitable for ultra-constrained IoT devices.

Hardware security can be achieved using hardware-assisted MCUs, dedicated hardware, or onboard implementation, as in the case of some IoT devices, and most especially the highly constrained devices such as RFID tags [223]. Such devices have very limited silicon area and very little energy available. In particular, the passive RFID tags are so constrained that they do not have any software-programmable processor. This implies that hardware implementation is the only feasible option available for implementing security in such devices. In addition, they do not even have a battery and must harvest power from the radio waves transmitted from a reader [248], and therefore designers must cope with this tight power budget.

Although optimizing all of the three aforementioned design goals at once is a very difficult task, even in the case of lightweight hardware implementation, designers should provide better trade-offs in this space. Particularly, by reducing the implementation cost of LWCAs as much as possible, while at the same time not losing track of the appropriate level of security. The key performance metrics that describe the efficiency of hardware implementations of cryptographic primitives are circuit area, energy consumption (which has already been outlined in the previous subsection), throughput, and latency [176]:

- *Circuit Area*: refers to a measure of physical circuit area needed to implement a primitive. In digital electronics, this is basically a function of the number of fundamental building blocks called logic gates that have two inputs and one output, such as NAND and NOR gates. The smaller the area, the better;
- *Throughput*: is a measure of the average units of data processed in each clock cycle, and in this context it is usually expressed in bytes per cycle. The higher it is, the better;
- *Latency*: refers to the time taken by a circuit to process an input data (a byte or block of data) and produce the corresponding output (a byte or block of data). The lower it is, the better.

Unlike software implementations that are coded using traditional programming languages like C and C++, a hardware implementation of LWC is best designed and programmed using a Hardware Description Language (HDL). An HDL is a language used to describe the structure and the behavior of digital electronic hardware designs. Two examples of HDLs are Very High Speed Integrated Circuit HDL (VHDL) and Verilog [249], which are used to map a circuit logic description to logic blocks. HDL designs can be implemented on a variety of hardware development environments, however, Application Specific Integrated Circuits (ASICs) and Field Programmable Gate Arrays (FPGAs) are currently the most popular platforms used for hardware implementations of cryptographic algorithms.

An ASIC is a custom manufactured microchip or IC specially designed for a particular purpose, for example, customized for a specific functionality required by the end product of one client. Modern ASICs can contain over 100 million logic gates. Since ASICs are designed only for a needed application, they are often more efficient in performance, very small in size (which can help to shrink the size of a product), and hence consume very little energy. They also provide very strong access control to cryptographic keys; and if carefully designed, the unit cost of ASICs can be considerably less [243]. However, the downside of this platform is that its initial development cost (which includes physical layout and subsequent fabrication in a semiconductor foundry) can be very high. Moreover, the process must be carefully monitored to ensure that the final design meets the given design requirements, and that the device operates well in real-world applications.

The FPGA, on the other hand, is an IC that allows users to create their own digital circuits using the onboard array of Configurable Logic Blocks (CLBs) connected via programmable interconnects. The logic blocks of most FPGAs contain memory elements, which can be complete blocks of memory or simple flip-flops. Using an HDL, users can configure and reconfigure the device, if need be, to a desired functionality or application requirements. Similarly, the FPGAs on circuit boards are usually programmed by the manufacturer, however, if there is need for update or upgrade, such FPGAs can be reprogrammed to effect desired changes. This is a key feature that distinguishes FPGAs from ASICs [250], which are custom manufactured only for specific applications. Reduced cost (FPGA tools are cheap or free) and shorter time of development, as well as simpler design cycle are other features that distinguish FPGAs from ASICs. In FPGAs, layout (physical design), fabrication, and verification of physical defects are not required.

Furthermore, considering that FPGAs are highly flexible, reconfigurable, and can be dynamically reprogrammed in the field, an insignificant error in the design of an algorithm would not invalidate the device, since a software patch could be released to fix the bug. While access control to crypto-keys in the FPGAs is weaker than in the ASICs, it is usually stronger than in software implementations.

However, FPGAs are typically slower than ASICs, larger in size, and hence consume more power [249], and are more costly (i.e., in terms of unit cost). These drawbacks are mainly due to the programmable routing interconnect, accounting for almost 90% of the total circuit area [250]. Although recent developments in the FPGAs are narrowing down the gap between them and the ASICs (now attaining up to a maximum frequency of 500MHz), most of the enhanced features of FPGAs concern functionality and not performance. Therefore, FPGAs are still less efficient than ASICs.

Although circuit area has been highlighted as a major performance metric in hardware implementation of lightweight cryptographic primitives, there is no accurate and acceptable

way to measure or estimate circuit area across different platforms. For example, an ASIC design mapped to two different technology libraries would give two different areas. This is also the case when two different families of FPGAs are compared. Consequently, both the academia and the industry have adopted a technology-independent method for determining circuit area in their quest to solve the problem in the ASIC space. The measurement standard is known as Gate Equivalent (GE), which can be obtained by dividing the total area A of a design (technology-dependent) in μm^2 by the area of the smallest two-input NAND gate N_A (technology-dependent) available in the particular technology [249], i.e.,

$$GE = \frac{A}{N_A}. \quad (3.3)$$

Despite the existence of a few works on area estimation model for the FPGAs [251], there is still no accurate uniform area estimation model for these devices. For example, the top two FPGA companies, Xilinx and Altera, have different measurement standards. The measurement standards for the Xilinx FPGAs are CLBs or Slices. One CLB element contains a pair of slices, and at a minimum, each slice is composed of four LUTs and eight storage elements (flip-flops). In the case of the Altera FPGAs, the measurement standards are Logic Array Blocks (LAB), Adaptive LogicModules (ALMs) or Logic Elements (LEs) [252]. Above all, converting from ASIC GEs to FPGA number of slices is a major challenge as ASIC designs do not map to the same FPGA area due to differences in technology and development processes. Despite attempts by some researchers to measure the gap between FPGAs and ASICs [253], there is still no easy and acceptable way to find the equivalent gate count of FPGAs.

As a vital component of microchip design reuse, the Intellectual Property core (IP core) is key to the continued advancement of the Electronic Design Automation (EDA), a trend that has revolutionized the electronics industry. Today, IP cores are essential building blocks of hardware designs using ASICs and FPGAs. Thus, the use of existing LWC algorithm IP cores that have been properly designed can facilitate the implementation of lightweight hardware primitives [254]. However, such IP cores must be those whose security is mature, having undergone a sufficient number of tests and reviews.

3.4.3 Lightweight Cryptography for IoT

In response to the constant need for secure and high-performance LWCAs suitable for IoT applications, research and development of LWC have been on the rise in recent years. As a consequent, quite a number of LWCAs have been standardized by different international and national organizations. For example, in 2012, the ISO has specified two block ciphers suitable for LWC as the ISO/IEC (29192-2P:2012), namely PRESENT and CLEFIA [255]. Similarly, in 2013, the Research Directorate of the US National Security Agency (NSA) proposed two block ciphers, SIMON and SPECK, specifically designed for

Table 3.2: A summary of the cryptographic algorithms evaluated by CRYPTREC.

Block Ciphers	Block/Key size (bits)	Stream Ciphers	Key size (bits)	Hash Functions	Message Authentication Code	Key size/Out length (bits)	Authenticated Encryption
CLEFIA	128/128, 128/192 128/256	ChaCha20	256	Keccak	SipHash	128/64	ACORN
LED	64/64, 64/128	Enocoro	80, 128	PHOTON			Ascon
PRINCE	64/128	Grain v1	80, 128	QUARK			AES-JAMBU
PRESENT	64/80, 64/128	MICKEY 2.0	80	SPONGENT			AES-OTR
Piccolo	64/80, 64/128	Trivium	80				CLOC and SILC
TWINE	64/80, 64/128						Deoxys
SIMON	32/64, 48/72, 48/96, 64/96, 64/128, 96/96, 96/144, 128/128, 128/192, 128/256,						Joltik
SPECK	32/64, 48/72, 48/96, 64/96, 64/128, 96/96, 96/144, 128/128, 128/192, 128/256,						Ketje
Midori	64/128, 128/128						Minalpher
							OCB
							PRIMATES

very constrained platforms [256]. In addition, the international cryptographic community has, over the years, made significant efforts in the development of other LWCAs, which include ciphers such as TWINE, Piccolo, PRINCE, and LED [257].

In an analogous manner, a lightweight cryptographic technology guideline has been provided by the Cryptography Research and Evaluation Committee (CRYPTREC) in Japan [176]. The guideline contains the results of a comprehensive study that reviewed different LWCAs that have potential applications in resource-constrained environments. Therefore, many of the cryptographic algorithms evaluated by CRYPTREC have been used in the security framework that will be presented in the next chapter (i.e., Chapter 4). Table 3.2 presents a summary of the cryptographic algorithms presented by CRYPTREC. Comparing the performance of the lightweight cryptographic algorithms presented in Table 3.2 is not the focus of this thesis, and hence the table only presents the algorithms and shows a few parameters of the block and stream ciphers, as well as a few parameters of the MAC algorithm. For more details on the remaining parameters of these and other algorithms see [176]. Note that Table 3.2 shows SipHash as the only MAC algorithm because according to [176], it is the only mature lightweight MAC algorithm that has undergone a sufficient number of reviews at the time of their study.

3.5 Conclusion

In this chapter, the important principles and concepts of the IoT-HarPSecA framework have been described to serve as the foundation for a formal presentation of the security framework in the next chapter. Since the IoT-HarPSecA framework is composed of three major components that will be described formally in Chapter 4, this chapter has focused on each aspect of these components, namely security requirements, security and privacy best practices, and lightweight cryptographic algorithms.

After describing some concepts of security requirements in IoT, this chapter outlined the security requirements for each of the nine application domains presented in Section 2.2. It further presented some challenges of developing commonly agreed regulations and guidelines based on industry best practices and pointed out some attempts to develop more widely accepted security best practices for the IoT. The chapter also presented a number of security and privacy best practices for IoT manufacturers and developers and outlined a few security and privacy best practices for enterprises as well as for individual IoT users. Finally, this chapter provided an overview of LWC and LWCA and discussed different types of LWCA. It also presented LWC design and implementation considerations, focusing on both software and hardware considerations.

Now that this chapter has described the underlying concepts of the IoT-HarPSecA from a theoretical perspective, the next chapter will focus on answering the question presented at the beginning of this chapter. The chapter will delve into the design and implementation of each of the components of the security framework.

Chapter 4

The IoT-HarPSecA Framework

This chapter will attempt to answer the question posed at the beginning of Chapter 3 of this thesis. The chapter delves deeply into the design and implementation of the three components of the IoT-HarPSecA (IoT Hardware Platform Security Advisor) framework. This chapter is mainly based on publications 2 and 4 [43, 50] mentioned in Subsection 1.5.1.

4.1 Introduction

According to [258], an IoT framework can be defined as a conceptual structure made up of guiding rules, protocols, or standards intended to simplify the design and implementation of IoT devices and smart apps. In this chapter, the IoT-HarPSecA framework will be presented, and in line with the above definition, the main goal of the IoT-HarPSecA framework is to facilitate the design and implementation of secure IoT devices and smart apps. To address the problem of implementing poorly secured IoT devices and applications that has been highlighted in Section 1.2 and Subsection 2.4.1.1 of this thesis, the security framework is designed to provide the necessary technical guidance to non-security experts actively involved in a variety of IoT and IIoT product development processes. This is achieved by implementing a tool for user interaction that realizes the functionalities of the individual components of the security framework, which is expected to help users in implementing the concept of security-by-design during the design and development processes of IoT systems.

IoT-HarPSecA offers three functionality features, namely security requirement elicitation, generation of a set of security best practice guidelines for secure development, and above all, a feature that recommends specific LWCAAs for both software and hardware implementations. Accordingly, IoT-HarPSecA is composed of three main components: (1) the SRE (Security Requirements Elicitation) component, (2) the SBPG (Security Best Practice Guidelines) component, and (3) the LWCAR (Lightweight Cryptographic Algorithms Recommendation) component, each of them servicing each of the aforementioned features, respectively. At the time this thesis was written and to the best of the knowledge of the author, there were no existing IoT security frameworks that seem to focus on addressing these issues.

As previously mentioned, IoT-HarPSecA is composed of three main components, each having a different functionality feature; however, a user can only select one functionality at a time, as shown in Figure 4.1. IoT-HarPSecA is modular in design, and the modularity

in the design of each of the components allows for easy integration of new functionalities; it also allows for easy upgrading of existing functionalities. In the following sections, a summary of the design and description of the modules in each of the components of the IoT-HarPSecA will be presented. Section 4.2 presents the design and description of the SRE component. Section 4.3 presents the design and description of the SBPG component. Then, Section 4.4 focuses on the design and description of the LWCAR component. Furthermore, Section 4.5 provides a summary of the implementation of the tool that can be used to interact with the IoT-HarPSecA framework. Finally, Section 4.6 concludes the chapter.

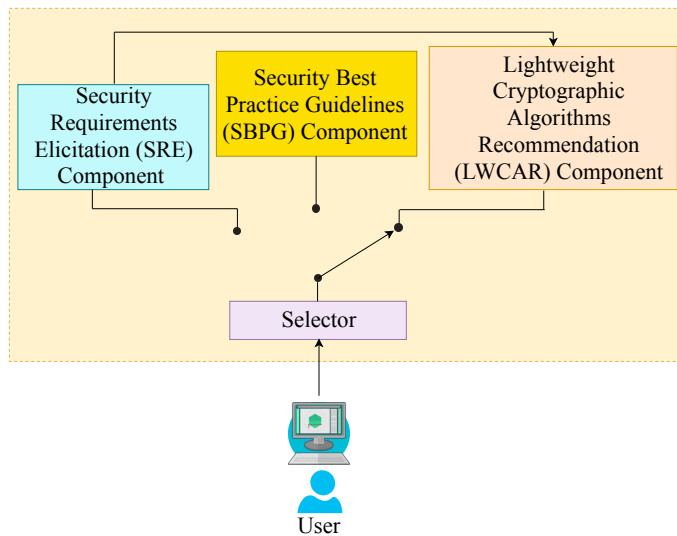


Figure 4.1: IoT-HarPSecA system components (adapted from [43]).

4.2 Design and Description of the Security Requirements Elicitation Component

The SRE component of the IoT-HarPSecA framework is responsible for generating a set of security requirements for a given IoT system based on user inputs, as highlighted in Section 4.1. A high-level depiction of the SRE component architecture is shown in Figure 4.2. The architecture consists of five functional modules, namely the user interface, security requirements generator, storage and updates, admin interface, and the output interface.

4.2.1 User Interface

This serves as a conduit between the security framework and the users. Depending on how the IoT-HarPSecA tool is implemented, the user interface can be a Command Line Interface (CLI) or a Graphical User Interface (GUI). But at the time of writing this thesis, it was a CLI. Upon registration and login, a user is taken to the *main menu* consisting of

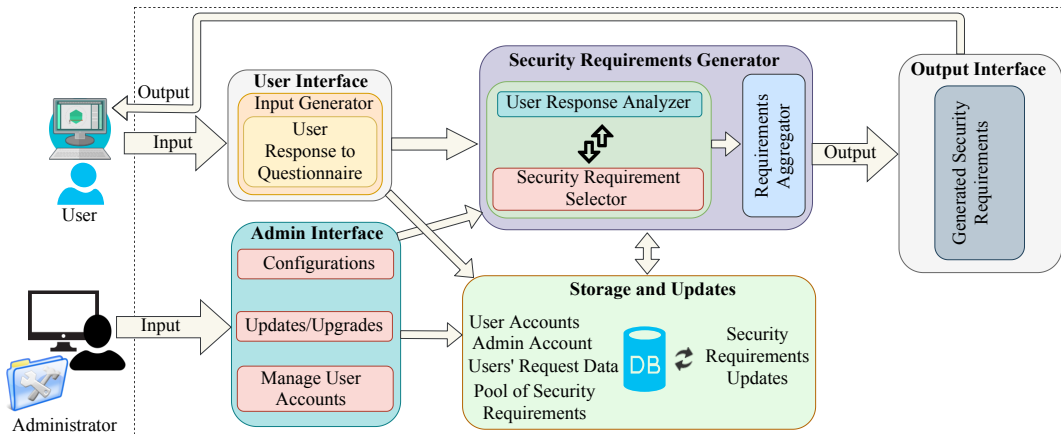


Figure 4.2: A high-level architecture of the SRE component of the IoT-HarPSecA (adapted from [43]).

14 options: options 1 to 4 are for the SRE component, 5 to 8 for the SBPG component, 9 to 12 for the LWCAR component, etc., as shown in Figure 4.3. By selecting option 1, a user is taken to the SRE component console window to make a request. However, prior to making a request, the user must choose a unique request Identification (ID) starting with an *R* which represents security requirements and followed by four numeric digits. The rationale for the choice of four digits is to significantly reduce the probability of two users choosing the same request IDs. This is important because an error will occur when inserting a new request in the MySQL database if the request ID specified in the insert query has already been used by another user. Therefore, the use of four digits provides the user with 10,000 possible combinations (i.e., from 0 - 9) to choose the four digits of his/her request ID from. After entering a unique request ID, the user is prompted to select an application domain, he/she is also asked to select the system development phase (which allows users to select whether the IoT system is under development, or it is an existing IoT system). This enables the tool to appropriately manage the way the questions are rendered.

Finally, the tool unfolds the questions in the questionnaire one after the other. The questionnaire consists of 16 questions, which include:

- *Will the system have a user?*
- *Will it have a user login?*
- *Will it store any user information?*
- *Will it store any other information?*
- *What type of information will it store? (normal, sensitive, or critical)*
- *Will it send data to a cloud?*
- *Can someone with bad intention have access to it?*

```

*****
WELCOME TO IoT-HarPSecA MAIN MENU

What would you like to do?

1. Make A New Security Requirement Elicitation Request
2. Display/Modify Your Security Requirement Elicitation Request
3. Process Your Security Requirement Elicitation Request
4. Delete Your Security Requirement Elicitation Request
5. Request A New Security Best Practice Guidelines for Secure Development
6. Display/Modify Your Security Best Practice Guidelines Request
7. Process Your Security Best Practice Guidelines Request
8. Delete Your Security Best Practice Guidelines Request
9. Make A New Lightweight Cryptographic Algorithms Recommendation Request
10. Display/Modify Your Lightweight Cryptographic Algorithms Recommendation Request
11. Process Your Lightweight Cryptographic Algorithms Recommendation Request
12. Delete Your Lightweight Cryptographic Algorithms Recommendation Request
13. Return to Login Menu
14. Exit

Select Your Option (1-14):

```

Figure 4.3: IoT-HarPSecA main menu.

Although the questionnaire consists of 16 questions, the total number of questions that can be presented to a user depends on his/her response to some of the previous questions. For example, if the answer of a user to the first question presented above is *no*, he/she will not be presented with the second question. The input generator extracts the required data from the user response to the questionnaire and feeds it to the security requirements generator. It also stores the extracted data in the database to allow the user to process the request at a later time.

4.2.2 Security Requirements Generator

This functional module consists of three components: user response analyzer, security requirements selector, and requirements aggregator. The user response analyzer evaluates the input data from the input generator and extracts specific information from every answer provided by the user, such as *yes*, *no*, *sensitive information* or *critical information*, etc. This information is used by the security requirement selector to select an appropriate security requirement from a pool of security requirements in the database. The requirement aggregator collects the individual security requirements for a given user, compiles them into a set of security requirements, and then pushes them to the output interface.

4.2.3 Storage and Updates

The storage and updates module is a repository which basically consists of a MySQL database. Since this module is similar in the three components, it is only described in this

```

*****
WELCOME TO IoT-HarPSecA ADMIN MENU

What would you like to do?

1. Display/Delete a User Registration
2. Delete a Security Requirements Elicitation Request
3. Delete a Security Best Practice Guidelines Request
4. Delete a Lightweight Cryptographic Algorithms Recommendation Request
5. Display, Add, Update, or Delete Lightweight Cryptographic Algorithms
6. Allow one more Admin user
7. Go to Registration/Login Menu and enter Main Menu as a user
8. Exit

Select Your Option (1-8):

```

Figure 4.4: IoT-HarPSecA admin menu.

subsection and the items stored in each instance of the module for each of the three components are mentioned. This module is where the user and admin accounts as well as the SRE, SBPG, and LWCAR components requests of users are stored. The database also stores the pool of security requirements for the SRE component and the collection of security best practice guidelines for the SBPG component. Other items stored in the database are the security requirements, security mechanisms, and the LWCA's used in the LWCAR component. These resources are entered manually by the administrator, and whenever the need arises they can be updated by the administrator.

4.2.4 Admin Interface

This can also be referred to as the administrator control panel. The *admin menu* of the IoT-HarPSecA tool consisting of eight options is shown in Figure 4.4. Since this module is the same for the SRE, SBPG, and LWCAR components, it will only be described herein. While the tool has provision for the creation of admin accounts, it is not designed with a default admin account and a hard-coded password. The administrator has to register as admin, and he/she will automatically be given the username *Admin1*. Usually, there is only one administrator, however, if need be, *Admin1* can authorize the creation of a second admin account as deemed necessary, which can have any username. To protect the passwords of both users and admin, a salted password hashing using the SHA256 hashing algorithm is employed in the IoT-HarPSecA framework.

Only authorized administrators can log into the admin interface, and once logged in, an administrator is trusted and assumed to be non-hostile, and hence will be able to perform all administrative tasks. With administrative privileges, an administrator can add, edit, or delete some records in the database. For example, an administrator can manage user accounts, perform updates on lightweight security algorithms, as well as delete SRE, SBPG,

and LWCAR requests. However, besides the ability to completely delete user registrations and user requests, an administrator cannot manipulate any user request, nor can he/she influence the decision of IoT-HarPSecA.

4.2.5 Output Interface

Each component of the IoT-HarPSecA framework features an output interface module with some similarities in their description. Therefore, the module is only described in this subsection, but a description of the output in each instance of the module for each of the components is provided here. For each component, the output interface simply communicates the output to the user. For example, the output interface module of the SRE component outputs the generated security requirements to the user. Similarly, the output interface module of the SBPG component produces a report consisting of a set of security best practice guidelines. In the same vein, the output interface module of the LWCAR component displays a table consisting of the user security requirements, their corresponding recommended security mechanisms and the lightweight security algorithms that provide them.

The output modules of the three IoT-HarPSecA framework components are printed both on the screen and to a text file. In all cases, the text file outputs are formatted in markdown. While the output module of the SBPG component prints only a summary of security best practice guidelines on the screen, the detailed result is printed to a text file upon request of a user. In an analogous manner, when the result of the LWCAR component is displayed on the screen, users are asked if they would like to see a detailed result. If a user selects yes, a detailed result will be displayed on the screen, which will include brief descriptions of the LWCAs and a few references where the user could read more about the algorithms, or even find full implementation details. In an instance where a user had used the SRE tool, the definitions and brief explanations on how to achieve the remaining security requirements generated by the SRE component, for which no LWCAs can be recommended, are also provided in the detailed result.

4.3 Design and Description of the Security Best Practice Guidelines Component

Like other aspects of information security, IoT security can never be guaranteed because new vulnerabilities are being discovered every day. This necessitates the need for regular review of security policies and practices relating to different operating environments and specific use cases, and hence the need for security best practice guidelines for different IoT systems and use cases. IoT security best practice guidelines essentially provide direction and guidance to designers and developers on how to appropriately secure IoT systems and products. However, the designer or developer is left to decide how best to

follow and implement them. Thus, the SBPG component generates a set of security best practice guidelines for secure development of IoT systems based on user inputs.

Figure 4.5 shows a high-level depiction of the SBPG component architecture. Like the SRE component, the architecture of the SBPG component also consists of five functional modules, namely the user interface, report generator, storage and updates, admin interface, and the output interface. While the architectures of the SRE and SBPG components look similar, most of their modules are different in design.

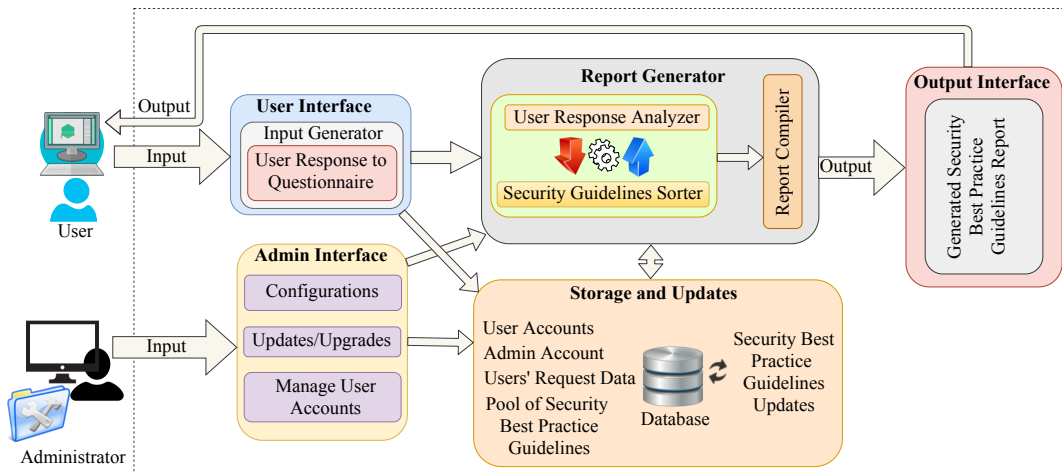


Figure 4.5: A high-level architecture of the SBPG component of the IoT-HarPSecA (adapted from [43]).

4.3.1 User Interface

Selecting option 5 in the *main menu* shown in Figure 4.3 takes a user to the SBPG component console window, where he/she can make a request after choosing a unique request ID starting with a *B*, which represents security best practices and followed by four numeric digits. Upon entering a request ID, a user is prompted to select the phase of system development and architecture(s) that best describes his/her IoT system, after which the questions in the questionnaire are presented.

Coincidentally, this questionnaire is also composed of 16 questions, including:

- *Will the system have a provision for user registration?*
- *Who will register users?*
- *Will the system allow users to enter any input?*
- *Will the system store user information?*
- *What type of authentication will be implemented?*
- *Will it store data in a database?*

- *Will it allow file upload?*
- *Will it generate a log file?*

As in the previous case, the number of questions presented to a user depends entirely on the response of the user to the previous questions. For instance, if the user response to the first question presented above is *no*, the second question will be skipped. The input generator extracts important information from the user response, which constitutes the user input that is fed to both the report generator and the storage and updates modules.

4.3.2 Report Generator

This module consists of three components: user response analyzer, security guidelines sorter, and report compiler. The user response analyzer is responsible for assessing and processing the input data from the input generator, which in conjunction with the security guidelines sorter selects the most appropriate guideline from a collection of predefined security guidelines stored in the database. The report compiler arranges the individual security best practice guidelines for a given user, compiles them into a report, and then sends the report to the output interface.

4.4 Design and Description of the Lightweight Cryptographic Algorithms Recommendation Component

As highlighted in Section 4.1, the LWCAR component of the IoT-HarPSecA framework is intended to facilitate the selection of secure LWCAs. Although selecting LWCAs can be a difficult task for non-security experts, it is even much harder to implement a security algorithm properly and accurately in either software or hardware. To make matters worse, implementation vulnerabilities may not be discovered early enough until an organization or a device is attacked. Thus, accurate implementation of security algorithms is extremely critical in cryptography, and hence it is imperative that users implement the recommended algorithms properly and accurately. As a consequence, the following important assumption, which is the core rationale behind the concept of the design of the LWCAR component of the IoT-HarPSecA framework is made: *All users of the LWCAR component are capable of implementing the recommended algorithms properly and accurately.*

Figure 4.6 depicts a high-level architecture of the LWCAR component of the IoT-HarPSecA framework. It consists of six functional modules, namely user interface, filtering and query processing, security manager, storage and updates, admin interface, and output interface.

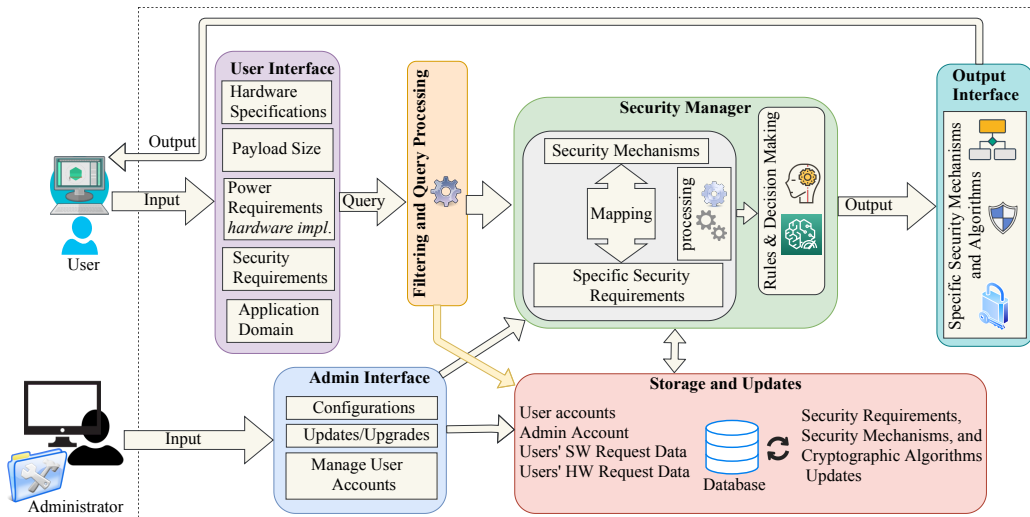


Figure 4.6: A high-level architecture of the LWCAR component of the IoT-HarPSecA (adapted from [43]).

4.4.1 User Interface

A user will be taken to the LWCAR component console window by selecting option 9 in the *main menu* (see Figure 4.3). He/she can make a request after choosing a unique request ID starting with *S* or *H* (depending on whether the request is for software or hardware implementation) followed by four numeric digits. The user will then be prompted to select the development phase of the IoT system (i.e., whether it is under development, or it is an existing system). The inputs expected from a user are (1) hardware specifications, (2) payload size, (3) power requirements (for hardware implementations only), (4) security requirements, and (5) application domain:

(1) *Hardware specifications*: For software implementations, and for IoT systems under development, users are prompted to select their hardware platform type (Advanced Virtual RISC (AVR)¹, Mixed-Signal Processor (MSP), Advanced RISC Machines (ARM), Programmable Interface Controllers (PIC), SBC, or other). They are also prompted to enter their flash memory size, RAM size, and processor frequency. For existing systems, a user is expected to provide flash memory and RAM sizes for each security requirement. However, for hardware implementation, the hardware platform options are only two, namely ASIC and FPGA. Additionally, users are expected to provide the GE and throughput for each security requirement; but this task is simplified for the user since the tool displays a range of values that serve as a guide for the user;

(2) *Message payload size*: Users are prompted to select a payload size: small (1-128 bytes), average (129-256 bytes), large (> 256 bytes), continuous (e.g., audio and video), or unknown;

¹RISC is an acronym for Reduced Instruction Set Computer.

(3) *Power requirements*: This is only applicable to the hardware implementation option, where users are prompted to select between low-power and ultra low-power options;

(4) *Security requirements*: The security requirements options that users must select from are (i) data confidentiality, (ii) integrity, (iii) authentication, (iv) user privacy, (v) non-repudiation, and (vi) confidentiality plus authenticity. Users are also asked if they have used the SRE component of the tool, and if the answer is yes, they are asked if they would want to import their generated security requirements, or they would want to select the security requirements manually. If they want to import their security requirements, they are asked to enter the request ID they used for the SRE request. Upon entering the SRE request ID and pressing enter, the aforementioned security requirements found in the generated list will automatically be imported. Users that have not used the SRE component tool must select their security requirements manually;

(5) *Application domain*: There is a long list of application domains users can choose from, including smart home, smart healthcare, industrial automation, smart retail, etc. Additionally, users can use the *other* option to specify their application domain if not found in the list.

All user inputs are fed into the filtering and query processing module for input screening and preprocessing. Figures 4.7 and 4.8 present the workflow of the software and hardware implementation requests, respectively.

4.4.2 Filtering and Query Processing

This Module manages user entries and performs the important task of screening for errors and invalidating unwanted entries from a user request. It basically acts as a preprocessor that ensures that input data is carefully screened to avoid misleading results. It assesses user requests and ensures that entries conform to the data format specified in the LWCAR component. Entries that do not conform to the prevailing standard are rejected, and users are asked to re-enter their requests again. When valid inputs are entered, the preprocessed user requests are fed into the security manager for further processing; the same requests are also stored in the database for later processing.

4.4.3 Security Manager

This module is central to the design and subsequent operation of the LWCAR component of the IoT-HarPSecA framework as it acts as the decision maker. It starts by determining user request type (i.e., whether it is a software or hardware implementation request). For software implementation requests, the *decision maker* checks the capability of the hardware (e.g., MCU flash and RAM sizes). However, if the hardware is an SBC, it only checks these parameters to ensure that they are typical of an SBC, and if not, it generates

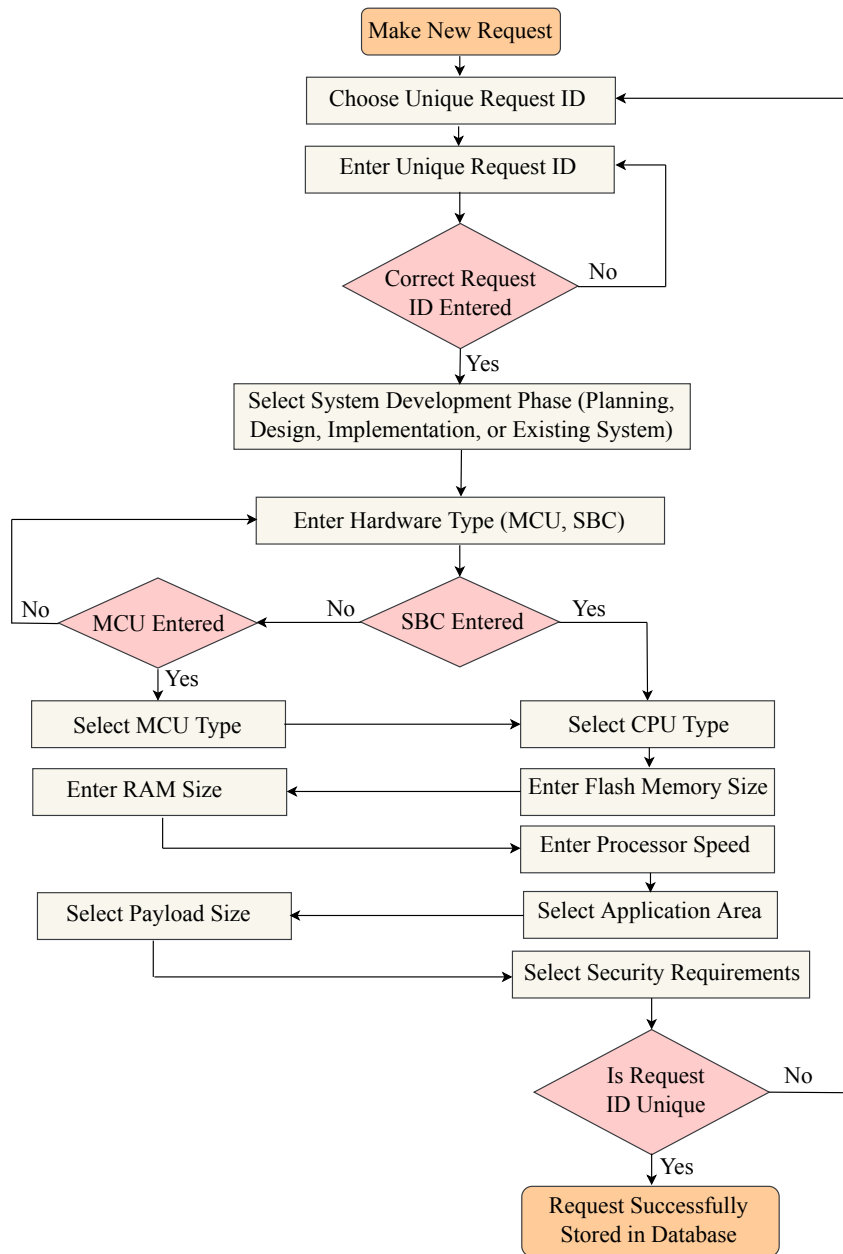


Figure 4.7: Software implementation request workflow for the LWCAR component (adapted from [43]).

an error message to alert the user. This is because most, if not all, SBCs can run standard security algorithms. Moreover, for both software and hardware implementation requests, the sensitivity of the application domain is key to selecting the appropriate cryptographic algorithms. For example, only the most efficient algorithms are selected for critical application areas, including healthcare, smart elderly monitoring, banking, retail, smart grid, and other sensitive application domains.

The security manager analyzes a user request and makes decisions based on a set of rules (rule base) and some encoded security metrics, namely, security mechanisms, security requirements, and problem-solving knowledge of human experts, serving as the knowledge

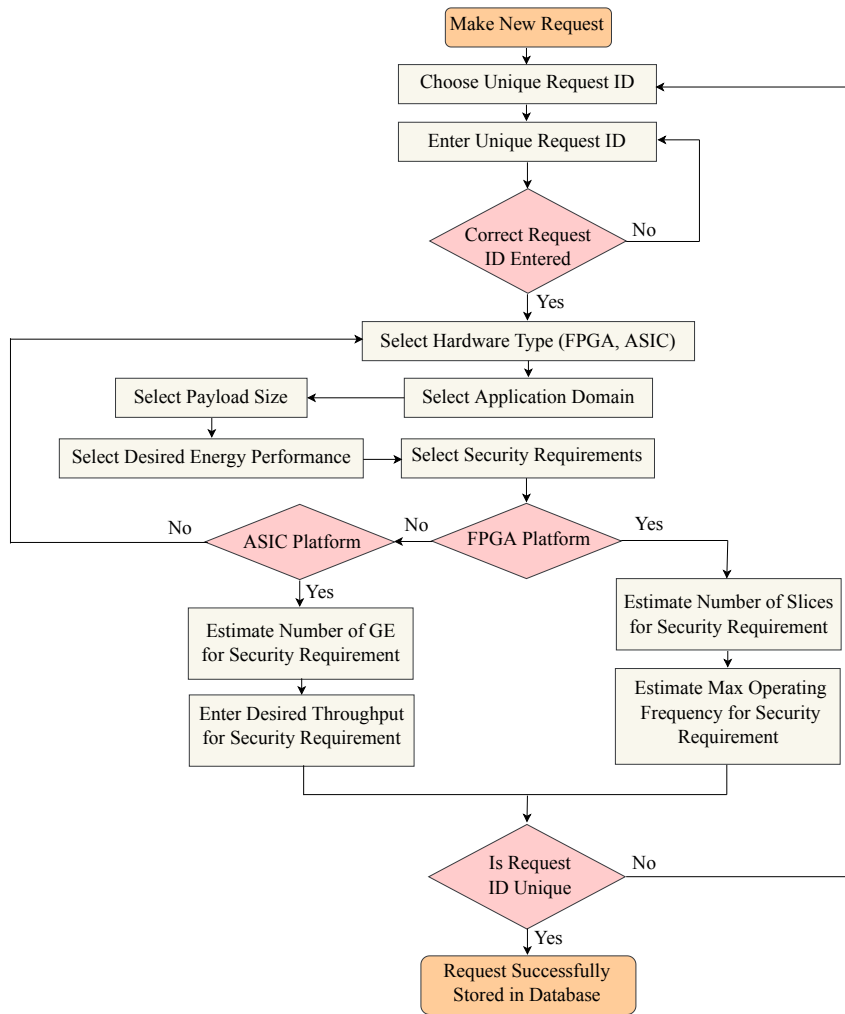


Figure 4.8: Hardware implementation request workflow for the LWCAR component (adapted from [43]).

base for the IoT-HarPSecA framework. The inference engine, which in the context of the IoT-HarPSecA is a component of the knowledge base, applies the rules to a given user request and evaluates relevant facts in the knowledge base. Finally, it recommends appropriate LWCAs from the pool of algorithms in the database consisting of several secure LWCAs. The key factors that play important roles in the decision-making process include the hardware capability, sensitivity of application domain, the message payload type, and power requirement (i.e., in the case of hardware requests).

The above mentioned factors play important roles in defining the rule base, which is a set of rules that govern the decision of the security manager. The rule base serves as the knowledge representation, which is based on the knowledge of human experts. Each rule consists of *if-then* statements that mimic the reasoning of human experts in solving the selection problem involving LWCAs. The rules that are applied to software implementation requests are outlined below:

1. IF RAM capacity is very large AND flash memory is very large THEN hardware is

very capable;

2. IF RAM capacity is sufficient AND flash memory is sufficient enough THEN hardware is capable;
3. IF the message payload type is small OR average OR large AND hardware is very capable OR capable THEN select an appropriate block cipher;
4. IF the message payload type is continuous OR unknown AND hardware is very capable OR capable THEN select an appropriate stream cipher;
5. IF application domain is sensitive AND hardware is very capable THEN select the most secure algorithm with large block size and/or large key size;
6. IF application domain is sensitive AND hardware is capable THEN select a secure algorithm with reasonable block size and/or reasonable key size;
7. IF hardware is very constrained THEN select a very lightweight algorithm that meets or most nearly meets this condition;
8. IF hardware is constrained THEN select a lightweight algorithm that meets this condition.

The rule base works on a top-down principle, where the first rule is applied first. If the premises for rule 1 are not met, rule 2 is applied, and if the premises for rule 2 are not satisfied, the remainder of the rules are not applied since, by implication, the hardware is not capable.

For example, before selecting a LWCA for a software implementation request, the security manager checks the available RAM as well as the storage space on the flash memory of the hardware in question by applying rule 1 or rule 2. Accordingly, it recommends a cipher with the appropriate block size and/or key size based on the sensitivity of the application domain (e.g., smart healthcare is considered a sensitive domain in the IoT-HarPSECA framework) by applying rule 3 or rule 4, that is, of course, if the hardware can support it. The message payload type is used to determine the type of encryption algorithm to be recommended (i.e., using rule 5 or rule 6), for example, the security manager recommends a stream cipher for a request with a continuous or unknown message payload size.

Similarly, the rules that are applied to hardware implementation requests are as follows:

1. IF the message payload type is small OR average OR large THEN select an appropriate block cipher;
2. IF the message payload type is continuous OR unknown THEN select an appropriate stream cipher;

3. IF circuit area is small AND throughput is high THEN select an algorithm that meets or most nearly meets these conditions;
4. IF circuit area is not too small AND throughput is moderate THEN select an algorithm that meets these conditions;
5. IF application domain is sensitive THEN select a secure algorithm with reasonable block size and/or reasonable key size;
6. IF power requirement is low-power THEN select an energy efficient algorithm;
7. IF power requirement is ultra-low-power THEN select a very energy efficient algorithm.

As in the case of the software implementation rules, the rule base for the hardware implementation works based on a top-down strategy, where the first rule is acted upon first.

Essentially, the SRE component tool generates many security requirements, and when users import their generated security requirements from the SRE component tool to the LWCAR component tool, the whole list is imported into the security manager. At the time of writing this thesis, only data confidentiality, integrity, authentication, user privacy, and non-repudiation are considered for algorithm recommendation. Nonetheless, modularity will potentially enable the integration of all security requirements. Where applicable, the security manager provides some insights into how users can meet the remaining security requirements for which no lightweight cryptographic algorithms can be recommended. Additionally, since the generated security requirements do not include confidentiality plus authenticity, the decision-maker scans the generated lists for confidentiality (and/or privacy) and integrity. If found, users are advised to consider returning to the *main menu* in order to select option 10 and modify their requests (see Figure 4.3) by including confidentiality plus authenticity, whose security mechanism is authenticated encryption, which can provide both confidentiality and/or privacy as well as authenticity.

4.5 IoT-HarPSecA Tool Implementation

This Section presents the implementation of the tool that allows users to interact with the IoT-HarPSecA framework. The IoT-HarPSecA framework tool is built to realize the functionality of the security framework, and hence it is also composed of the three components of the framework first introduced in Section 4.1. Thus, the IoT-HarPSecA framework tool is also made up of the SRE, SBPG, and LWCAR components. The console application is implemented in C++, and it can run on different operating systems such as Windows and Linux. The flexibility in the implementation allows for fine-tuning of some parameters in order to improve the performance of the tool; it also enables the addition of new features, such as the integration of new metrics. A MySQL database is used for maintaining all information in the IoT-HarPSecA framework tool such as user requests, user registration

data, and the security metrics mentioned in Subsection 4.4.3.

The SRE, SBPG, and LWCAR component tools are collectively referred to as the framework tool. Nonetheless, each component tool can function as a standalone tool. While each component tool is independent of the others, a user that had previously used the SRE component tool can import the generated security requirements into the LWCAR component tool as highlighted in the last paragraph of Subsection 4.4.3. This relationship is depicted in Figure 4.1 by an arrow that connects the SRE and LWCAR component tools. The complete source code, which is released under the Apache License, Version 2.0 (SPDX-License-Identifier: Apache-2.0), is available on Github and can be accessed via https://github.com/mgsamaila/IoT-HarPSecA_Tool; it is also available on the **SECUR IoT ESIGN** project Github page: <https://github.com/SECURIoTESIGN/SECURIoTESIGN>.

The IoT-HarPSecA framework tool consists of four dialogue-oriented console menus: the login menu, registration/login menu, main menu, and admin menu, each having different options from which a user or an administrator can choose from. Both users and administrator(s) must register and be authenticated before logging in. Their accounts details are securely stored in two different MySQL database tables. In particular, the stored passwords are protected with a salted SHA256 hash as mentioned in Subsection 4.2.4. The screenshots of the last two menus of the tool have already been shown in Figure 4.3 and Figure 4.4, respectively. The following subsections summarize the implementations of the components of the framework tool.

4.5.1 Implementations of the SRE and SBPG Components of the Tool

Algorithm 1 summarizes the implementation flow for the SRE component tool. It describes the procedure for making a request, as well as outlines the procedure and main stages of the questionnaire design. In Algorithm 1, it is assumed that the IoT system is in its early phase of development, and hence the use of *Will* in the questionnaire.

Algorithm 1 Summary of the main stages of the SRE tool implementation.

```
1: Enter a request ID;
2: Select application domain;
3: Select system development phase.
4: Will the system have a user? (yes1 or no1);
5: if yes1 then
6:   Will the system have a user login? (yes2 or no2);
7: if yes1 then
8:   Will it store any user information? (yes3 or no3);
9: if yes3 then
10:  Will it store any other information? (yes4 or no4);
11: else
12:  Will it store any information? (yes5 or no5);
13: if yes3 or yes5 then
14:  What type of information? (normal, sensitive, or critical);
15: if yes3 or yes4 or yes5 then
16:  Will the information be sent to an entity? (yes6 or no6);
17: Will it be connected to the Internet? (yes7 or no7);
18: Will it send data to a cloud? (yes8 or no8);
19: Will it store data in a database? (yes9 or no9);
20: Will it receive regular updates? (yes10 or no10);
21: Will it use third-party software? (yes11 or no11);
22: Is there possibility of eavesdropping? (yes12 or no12);
23: Could messages sent between system components be captured and re-
    played? (yes13 or no13);
24: Can someone try to impersonate a user? (yes14 or no14);
25: Can someone with bad intentions gain physical access to the system?
    (yes15 or no15);
```

The inputs in Algorithm 1 are application domain, phase of system development, and response to the questionnaire; and the output is a set of security requirements.

The description of the SBPG component tool implementation can be summarized as in Algorithm 2. The algorithm describes the procedure for making a request and represents the stages of the SBPG component tool questionnaire design. In Algorithm 2, it is assumed that the IoT system is an existing system, and hence the use of *Does* in the questions.

Algorithm 2 Summary of the main stages of the SBPG tool implementation.

- 1: Enter a request ID;
 - 2: Select system development phase;
 - 3: Choose IoT system architecture.
 - 4: Does the system have a user? (*yes₁* or *no₁*);
 - 5: **if** *yes₁* **then**
 - 6: Does it have a provision for user registration? (*yes₂* or *no₂*);
 - 7: **if** *yes₂* **then**
 - 8: Who register users? (*admin, users themselves*);
 - 9: **if** *yes₂* **then**
 - 10: Is there user login? (*yes₃* or *no₃*);
 - 11: **if** *yes₁* **then**
 - 12: Does it allow users to enter any input? (*yes₄* or *no₄*);
 - 13: **if** *yes₁* or *yes₄* **then**
 - 14: Does it store user information? (*yes₅* or *no₅*);
 - 15: **if** *yes₅* **then**
 - 16: Does it store any other information? (*yes₆* or *no₆*);
 - 17: **else**
 - 18: Does it store any information? (*yes₇* or *no₇*);
 - 19: **if** *yes₅* or *yes₇* **then**
 - 20: What type of information? (*normal, sensitive, or critical*);
 - 21: What is the current authentication type? (*no authentication, username and password, etc.*);
 - 22: Does it store data in a database? (*yes₈* or *no₈*);
 - 23: What is the type of data storage? (*SQL, NoSQL, Local storage, etc.*);
 - 24: What type of database is used? (*SQL server, MySQL, SQLite, etc.*);
 - 25: What programming language is use? (*C/C ++, Java, Ruby, Python, PHP, Javascript, etc.*);
 - 26: Does it allow file uploads? (*yes₉* or *no₉*);
 - 27: Does it generate a log file? (*yes₁₀* or *no₁₀*);
-

Similarly, the inputs in Algorithm 2 are phase of system development, IoT system architecture, and response to the questionnaire; and the output is a set of security best practice guidelines.

4.5.2 Implementation of the LWCAR Component Tool

The procedure for making a request for both software and hardware implementations have already been depicted in Figures 4.7 and 4.8 in Subsection 4.4.1, respectively. Therefore, Algorithm 3 only summarizes the procedure for processing a request in the LWCAR component tool of the IoT-HarPSecA framework.

Algorithm 3 Summary of the main stages of the implementation of the request processing aspect of the LWCAR tool.

```
1: Begin
2: Enter a request ID;
3: Check request type (for software or hardware implementation);
4: while RAM and flash memory sizes are okay do
5:   Check sensitivity of application domain (sensitive, not sensitive);
6:   Check message payload type (small, average, large, continuous,
   unknown);
7:   if request is for hardware implementation then
8:     Check power requirement (low power, ultra low power);
9:   Select the right security mechanism;
10: Recommend the most appropriate LWCA;
11: End
```

The inputs in Algorithm 3 are hardware specifications, payload size, power requirement, security requirements, and application domain; and the outputs are security mechanisms and recommended LWCAs.

Most of the metric parameters used in the implementation of the LWCAR component tool are obtained from the CRYPTREC Cryptographic Technology Guideline [176] as well as from the FELICS framework [241]. In the implementation, kilobyte (kB) is used as the unit of memory measurement for the software implementation requests due to the limited amount of memory on the resource-constrained MCU devices. Although SBCs have a reasonable amount of memory, the kB is still used for them as well for uniformity purposes; and Megahertz (MHz) is used as the unit of processor clock frequency measurement for both MCUs and SBCs for the same reasons. Two metrics are used for the implementation of each of the hardware request options available: circuit area and throughput for the ASICs, and number of slices as well as the maximum operating frequency for the FPGAs. As discussed in Subsection 3.4.2.2 of Chapter 3, GEs and number of slices are used as the unit of circuit area measurements for the ASIC and FPGA platforms, respectively. For the second metric, in each case, Kilobits Per Second (Kbps) and MHz are adopted as the units for throughput and maximum operating frequency, respectively.

The implementation also includes some error detection mechanisms that warn users of obviously erroneous or inconsistent data entry. For instance, the tool warns users that their hardware specifications are not typical of SBCs if they mistakenly select a SBC instead of a MCU. It also generates a warning when a MCU capability is too constrained for the algorithms in the database, as well as alerts a user when an invalid request ID is entered. Furthermore, if a user enters a non-unique request ID, the IoT-HarPSecA tool notifies the user that the entered request ID already exists in the database. In such a situation, the user is asked to press enter to go back to the main menu and repeat the process with a unique request ID.

4.6 Conclusion

This chapter has attempted to provide an answer to the question posed at the beginning of Chapter 3 by providing a detailed description of the three components of the IoT-HarPSecA framework. The chapter has discussed the design and description of the SRE component. It also presented the design and description of the SBPG component, as well as provided a detailed design and description of the LWCAR component. Finally, this chapter discussed the implementation of the three components of the IoT-HarPSecA framework tool that allows users to interact with the security framework, which may help them in the implementation of the concept of security-by-design during the design and development processes of IoT systems.

Now that the design and implementation of the framework and the mode of operation of its three components have been described, the discussion may move on to the description of the tests that were conducted to evaluate its functionality and usability. The next chapter focuses mainly on this aspect of the work.

Chapter 5

Functionality and Usability Tests, Results and Evaluation

This chapter presents a number of functionality and usability tests performed on the SRE, SBPG, and LWCAR components of the IoT-HarPSecA framework tool. The analysis of the results obtained serves as the basis for evaluating the performance and usability of the IoT-HarPSecA framework. This chapter is largely based on publications 2, 4, 9 and 10 [43, 50, 51] mentioned in Subsection 1.5.1.

5.1 Introduction

In chapter 4, detailed design and description of the three components of the IoT-HarPSecA framework have been presented; the chapter also discussed the implementation of the tool that can be used to interact with the security framework, namely the IoT-HarPSecA framework tool. This chapter is intended to showcase the potential capabilities of the IoT-HarPSecA framework by presenting some common usage scenarios for the SRE, SBPG, and LWCAR components of the security framework. The chapter is, most importantly, aimed at discussing the performance evaluation of the framework. In the context of this thesis, performance refers to the usefulness or functionality of the framework. Thus, this chapter focuses on analyzing and evaluating the functionality and overall user experience of the security framework by way of assessing its efficacy and usability. The performance evaluation is an important step towards validating the effectiveness of the IoT-HarPSecA framework. The different tests carried out will also help to verify whether or not the IoT-HarPSecA framework tool functions correctly in conformity with the design objectives. For simplicity, the IoT-HarPSecA framework tool will hereafter be referred to as the tool in most instances, and the three components of the tool will individually be referred to as the SRE tool, SBPG tool, and LWCAR tool, respectively.

Although many subjects have participated in the tests for evaluating the performance of the SRE and SBPG tools, this chapter presents only three test results of the SRE tool and the summaries of three test results of the SBPG tool. This is because the results generated by the SRE and SBPG tools are somewhat large, especially in the case of the SBPG tool. Thus, only the summaries of the test results of the SBPG tool are presented in this chapter because a full-length result generated by the SBPG tool can run to several pages. Nonetheless, an example of a full-length result generated by the SBPG tool is presented in Appendix A. On the other side of the spectrum, while only the summaries of four test results and one full-length test result of the LWCAR tool are presented in this chapter, the

whole test results have been summarized in a single table due to the compact nature of the result summary produced by the LWCAR tool.

The LWCAR component is the main component of the IoT-HarPSecA framework, and hence the LWCAR tool was the first to be developed. Thus, the performance evaluation tests for the LWCAR tool were carried out much earlier and the results have already been published in [43]. The performance evaluation tests for the SRE and SBPG tools were conducted at the same time, one after the other. Therefore, to keep track of the two test requests for a given subject (i.e., the SRE and SBPG tools test requests for a particular subject), each subject was asked to use the same request IDs for the two requests, with the only difference being the first letters of the request IDs, which are *R* and *B* for SRE and SBPG tools test requests, respectively, as already discussed in subsections 4.2.1 and 4.3.1 (e.g., R1234 and B1234). A preliminary evaluation of the SRE and SBPG tools of the IoT-HarPSecA framework has been presented at the 2020 IEEE Global Internet of Things Summit (GIoTS) [51]. Furthermore, at the time of writing this thesis, an article that discusses, in more detail, the performance evaluation of the SRE and SBPG tools has been submitted to the Elsevier Journal of Computer Networks for publication (i.e., article 10 in the first list of publications in Subsection 1.5.1).

The following sections present the functionality and usability tests, results, and performance evaluation of each component of the tool. More precisely, Sections 5.2, 5.3 and 5.4 discuss each of the above-mentioned subjects for the SRE, SBPG and LWCAR tools, respectively. Finally, Section 5.5 wraps up the chapter with a conclusion.

5.2 Test, Results, and Performance Evaluation of the SRE Tool

Performance and usability testing are typically done to determine compliance with certain performance goals and requirements, as well as to evaluate or assess user experience which focuses more on the aspect of user interaction with a given system under test. In addition, a statistical analysis of results from performance and usability testing can help to identify bottlenecks in a system. This section presents the test that is used to evaluate the performance and usability of the SRE tool. The evaluation is based on two performance metrics that will be presented in subsection 5.2.1.1.

5.2.1 SRE Tool Test Setup

The process of setting up the performance and usability test involves creating a testing task similar to a typical real-world usage for volunteer subjects to complete using the SRE tool. Based on the intended functionality of the SRE component of the IoT-HarPSecA framework described in Section 4.2, a test scenario that specifically targets people actively

involved in the design and implementation of IoT systems is created. Note, however, that because of its versatility, the SRE tool can also be adapted for some standard computer systems. A test scenario that demonstrates the interaction of a user with the SRE tool is briefly described in the following subsection.

5.2.1.1 SRE Tool Test Scenario

Despite the fact that there is only one test case for the SRE tool, the test scenario is tied to the two major requirements needed to be verified, namely functional accuracy and usability. From the perspective of users, a real-world scenario and use case of the SRE tool would simply be the elicitation of security requirements for a given IoT system. Thus, in this scenario, it is considered that a user is making a request to know the security requirements of his/her IoT system which may be in its early phase of development (i.e., conception, planning, design) or an existing IoT system.

In this test case, potential users (referred to in this chapter as volunteer subjects or simply subjects) were approached to perform the test, which is aimed at verifying whether or not the tool can effectively perform its intended task in real-world scenarios in accordance with the design specifications. The subjects were not given a priori indication or hints of what system parameters or application domain they need to use (they were free to choose the system parameters and scenarios they were more acquainted with). Aside from a brief introduction of the SRE tool given by the author at the beginning, subjects were allowed to carry out the test with minimal intervention. The following performance metrics are used to assess the performance of the SRE tool:

1. security requirements elicitation accuracy of the SRE tool;
2. simplicity and ease of use of the SRE tool.

5.2.2 Results Overview of the SRE Tool Test

The test was performed on a Windows 10 laptop running the tool. A total number of 24 subjects, consisting of both males and females in the age range of 18–59 years, participated in the test. Out of the 24 subjects that participated in the test, 13 were computer engineers, 6 developers, and 5 electronic engineers. At the end of the test, subjects were asked to complete an evaluation form (a paper form), where they provided the following information: request ID number, age range, level of security experience, a field of expertise, proficiency level, and their opinion on the simplicity of use of the tool. The evaluation form does not require subjects to provide any sensitive information such as name and specific age. This is to encourage more subjects to participate in the test and to protect their privacy.

Table 5.1 presents the data in the performance evaluation form as completed by the 24 subjects. Apart from the aforementioned information provided by the subjects, the last

Table 5.1: SRE tool performance evaluation form as completed by subjects.

S/No.	Request ID	Age Range	Security Experience	Field of Expertise	Proficiency Level	Simplicity of Use	Request Completion Time (s)
1.	R1111	A	B (7.5)	D	B (7.5)	A (10.0)	76.8
2.	R2345	C	B (7.5)	C	C (5.0)	A (10.0)	134.0
3.	R4444	A	D (2.5)	D	C (5.0)	A (10.0)	117.3
4.	R1234	A	D (2.5)	D	C (5.0)	A (10.0)	80.5
5.	R6548	A	D (2.5)	C	B (7.5)	B (7.5)	105.2
6.	R0601	D	D (2.5)	C	C (5.0)	B (7.5)	129.5
7.	R7788	C	D (2.5)	C	A (10.0)	B (7.5)	96.5
8.	R5432	A	E (0.0)	D	C (5.0)	B (7.5)	139.1
9.	R2278	B	E (0.0)	D	C (5.0)	B (7.5)	141.2
10.	R9128	A	D (2.5)	C	C (5.0)	B (7.5)	123.3
11.	R6666	A	D (2.5)	C	B (7.5)	A (10.0)	88.8
12.	R1115	A	C (5.0)	D	A (10.0)	A (10.0)	145.5
13.	R1995	B	D (2.5)	E	C (5.0)	B (7.5)	115.3
14.	R4321	B	B (7.5)	C	B (7.5)	B (7.5)	148.8
15.	R1235	C	E (0.0)	E	B (7.5)	C (5.0)	153.2
16.	R8374	A	B (7.5)	C	A (10.0)	A (10.0)	122.0
17.	R4040	C	C (5.0)	E	B (7.5)	A (10.0)	103.9
18.	R8789	D	D (2.5)	C	C (5.0)	A (10.0)	108.7
19.	R1008	B	D (2.5)	C	A (10.0)	A (10.0)	96.9
20.	R5555	C	E (0.0)	E	B (7.5)	B (7.5)	135.4
21.	R1287	D	D (2.5)	C	B (7.5)	A (10.0)	81.7
22.	R5461	B	D (2.5)	C	B (7.5)	B (7.5)	93.2
23.	R0011	D	E (0.0)	E	A (10.0)	A (10.0)	79.4
24.	R8126	A	D (2.5)	C	C (5.0)	A (10.0)	96.3

Table 5.2: SRE request data of subjects as stored in the database.

Reqst_ID	state	Domain	anyUsr	anyUsrLogin	holdUsrInfo	storeAnyInfo	sensitivOfInfo	infoSent2E
R1111	Off	City	No			Yes	Normal	Yes
R2345	Off	Toy	Yes	Yes	Yes	Yes	Normal	Yes
R4444	On	Home	Yes	Yes	Yes	No	Sensitive	Yes
R1234	Off	Wearable	Yes	Yes	Yes	Yes	Normal	Yes
R6548	Off	Pet	Yes	Yes	Yes	Yes	Normal	No
R0601	Off	Healthcare	Yes	Yes	Yes	Yes	Normal	Yes
R7788	Off	Healthcare	Yes	Yes	Yes	Yes	Sensitive	No
R5432	Off	Grid	Yes	Yes	Yes	Yes	Sensitive	Yes
R2278	Off	Healthcare	Yes	Yes	Yes	Yes	Sensitive	Yes
R9128	Off	Environment	No			Yes	Critical	Yes
R6666	Off	AI	No			Yes	Normal	No
R1115	Off	Agriculture	Yes	Yes	Yes	Yes	Normal	Yes
R1995	Off	Toy	Yes	Yes	No	Yes	Normal	Yes
R4321	Off	Environmental	Yes	No	Yes	No	Sensitive	Yes
R1235	Off	Retail	Yes	Yes	No	Yes	Normal	Yes
R8374	Off	Healthcare	Yes	Yes	Yes	No	Sensitive	Yes
R4040	Off	Grid	Yes	Yes	No	Yes	Normal	No
R8789	Off	City	Yes	Yes	Yes	Yes	Critical	Yes
R1008	Off	Elderly	Yes	Yes	Yes	Yes	Critical	Yes
R5555	Off	Connected_Car	Yes	Yes	Yes	Yes	Normal	No
R1287	Off	Healthcare	Yes	Yes	Yes	Yes	Critical	Yes
R5461	Off	Grid	Yes	Yes	No	No		
R0011	On	Home	No			Yes	Critical	Yes
R8126	Off	City	Yes	Yes	Yes	Yes	Sensitive	Yes

Table 5.3: Continuation of SRE request data of subjects as stored in the database.

connected	dataSent-2Cloud	data-StoredInDb	regula-Update	use3rd-PrtySfw	possibl-OfEvesdrop	possibl-,OfCapt_Resent	Possibl-OfImpersonatUsr	possibl-OfPhysicAcces
Yes	Yes	Yes	Yes	No	Yes	Yes		Yes
Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes
Yes	Yes	Yes	Yes	No	No	No	Yes	Yes
Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes	No	Yes	Yes	Yes	Yes	Yes	No	No
Yes	Yes	Yes	No	No	No	No	Yes	Yes
Yes	No	No	Yes	No	Yes	No	Yes	No
Yes	Yes	Yes	No	No	No	Yes	Yes	Yes
Yes	Yes	Yes	Yes	Yes	No	No		Yes
Yes	No	Yes	No	Yes	No	No		Yes
Yes	Yes	No	Yes	No	Yes	Yes	Yes	Yes
Yes	No	No	No	No	Yes	No	No	Yes
Yes	Yes	Yes	Yes	No	No	Yes	Yes	No
Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	No
Yes	Yes	Yes	No	No	No	No	Yes	No
Yes	No	No	No	No	Yes	Yes	Yes	No
Yes	Yes	No	No	Yes	Yes	No	No	No
Yes	No	No	Yes	Yes	No	No	Yes	No
Yes	Yes	No	No	No	No	No	Yes	Yes
Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes	No	Yes	Yes	No	No	No		Yes
Yes	No	No	No	No	Yes	Yes	Yes	Yes

parameter required in the evaluation form is the elapsed time for each request, which is obtained from the tool. Tables 5.2 and 5.3 show the SRE requests data of the subjects stored in the database. Table 5.3 is a continuation of Table 5.2. But the actual screenshots of requests data of the subjects as stored in the MySQL database table can be found in Figure B.1 in Appendix B. The columns names in Tables 5.2 and 5.3 (or in Figure B.1) are variables in the C++ code implementation of the IoT-HarPSecA framework tool, which store user response to the questions presented (see Subsection 4.2.1). Table 5.4 provides more clarity on the designations of the 18 columns in the SRE user request database table in Tables 5.2 and 5.3, or Figure B.1.

To simplify the form filling process for subjects, the following assessment format was adopted. Age range: 18-29, 30-39, 40-49, 50-59 (A, B, C, D); Security experience: an expert, very experienced, adequate, a little, none (A, B, C, D, E); field of expertise: developer, computer engineer, electronics engineer (D, C, E); proficiency level: an expert, very proficient, proficient, a little, none (A, B, C, D, E); and simplicity of making a request: very easy, easy, average, difficult, very difficult (A, B, C, D, E). The security expertise, proficiency level, and the simplicity of use metrics can be easily converted to a 0-to-10 numeric rating scale (i.e., 10.0, 7.5, 5.0, 2.5, 0.0), as shown in Table 5.1. The data obtained from scaling the three metrics are used to present the SRE tool performance evaluation form data graphically in Figures 5.1, 5.2, and 5.3, which are more intuitive and interpretable than the data in the evaluation form presented in Table 5.1.

As it was previously mentioned in Section 5.1, only three SRE test results will be presented

Table 5.4: Description of the designations of the 18 columns in Tables 5.2 and 5.3.

S/No.	Column Designation	Brief Description
1.	Reqst_ID	User request ID
2.	state	System development phase: <i>Off</i> means early stage of development and <i>On</i> means it is an existing IoT system
3.	Domain	Application domain
4.	anyUsr	Will, or does the system have a user?
5.	anyUsrLogin	will there be, or is there a user login?
6.	holdUsrInfo	Will, or does the system store user information? information
7.	storeAnyInfo	Will, or does it store any other information?
8.	sensitivOfInfo	Sensitivity of information stored
9.	infoSent2E	Will it send, or does it send information to other entities?
10.	connected	Will it be, or is it connected to the Internet?
11.	dataSent2Cloud	Will it send, or does it send data to any cloud platform?
12.	dataStoredInDb	Will it store, or does it store data in a database
13.	regulaUpdate	Will it receive, or does it receive regular updates?
14.	use3rdPrtySfw	Will it use, or does it use third party software?
15.	possiblOfEvesdrop	Is there a possibility of eavesdropping attacks?
16.	possiblOfCap_Resent	Is there a possibility of replay attacks?
17.	possiblOfImpersonatUsr	Is there a possibility of attackers impersonating a user?
18.	possiblOfPhysicalAcces	Is there a possibility that malicious entities can have physical access to the system?

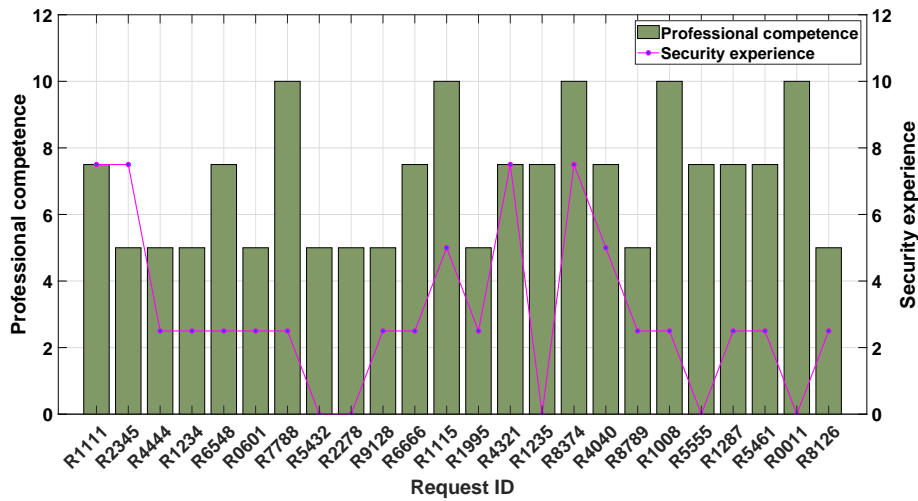
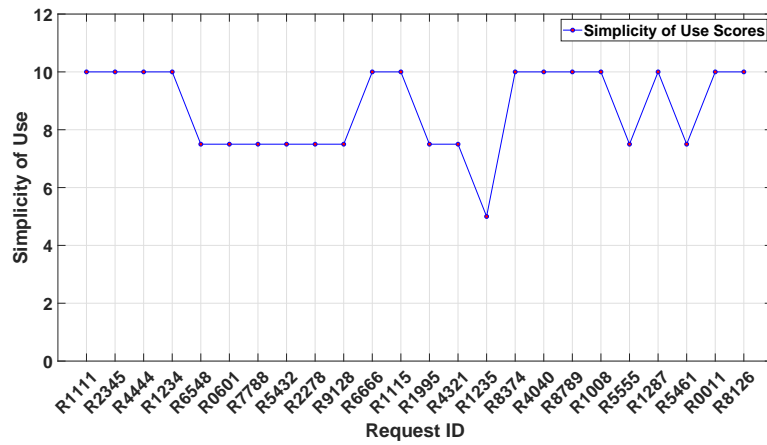
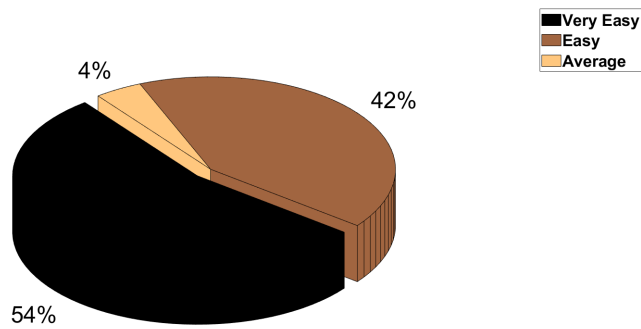


Figure 5.1: Professional competence and security experience of the 24 subjects.

in this chapter due to space constraints. Hence, the results of the SRE test for subjects with request IDs R1995, R5432, and R1287 are presented in Figures 5.4, 5.5, and 5.6, respectively. Figures 5.4, 5.5, and 5.6 consist of 6, 11, and 15 security requirements, respectively. The three results are selected from the 24 test results not only to represent the minimum, average, and the maximum number of security requirements obtained from the test conducted on the SRE tool but also to represent one test result from each group of volunteer subjects, representing the three areas of expertise of the subjects that participated in the test (see Table 5.1). The evaluation of these results is presented in Subsection 5.2.3.1.



(a) SRE tool simplicity of use scores of individual subjects



(b) Percentage representation of the simplicity of use of the SRE tool

Figure 5.2: Simplicity and ease of use of the SRE tool.

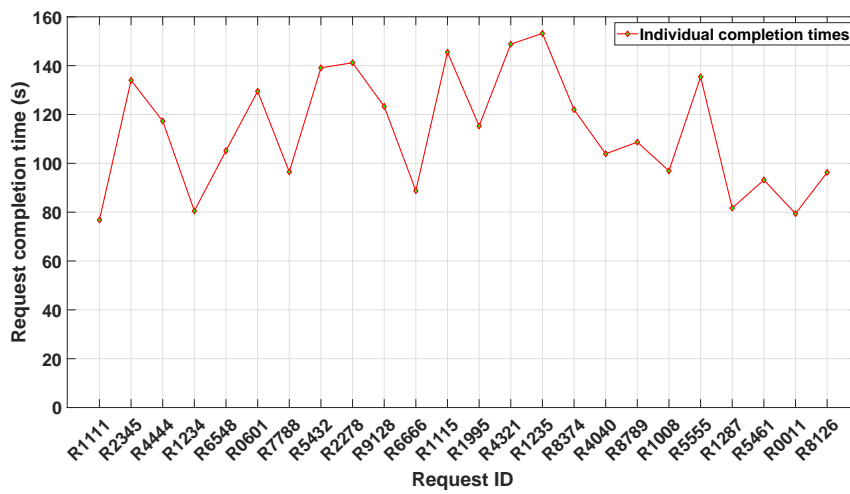


Figure 5.3: Request completion times of subjects for the SRE tool test.

```
*****
THE SECURITY REQUIREMENTS FOR THE IoT SYSTEM OF THE USER WITH REQUEST ID No.: R1995
```

SECURITY REQUIREMENT	DESCRIPTION
Authentication	This is the assurance that a message is from the source it claims to be from.
Confidentiality	This is the property that ensures that information is not disclosed or made available to any unauthorized entity.
Integrity	Is the property of safeguarding the correctness, consistency, and trustworthiness of data over its entire life cycle in an IoT system.
Availability	Refers to the property which ensures that an IoT device or system is accessible and usable upon demand by authorized entities.
Confinement	Ensures that even if an entity is hijacked or corrupted, the spreading of the effects of the attack is as confined as possible.
Physical Security	Refers to the security measures designed to deny unauthorized physical access to IoT devices or systems, and to protect them from damage or tampering.

Press Enter to return to the Main Menu

Figure 5.4: Final results of SRE request for subject with request ID R1995.

```
*****
THE SECURITY REQUIREMENTS FOR THE IoT SYSTEM OF THE USER WITH REQUEST ID No.: R5432
```

SECURITY REQUIREMENT	DESCRIPTION
Authentication	This is the assurance that a message is from the source it claims to be from.
Privacy	Refers to users control over the disclosure of their personal information, meaning that only the users should decide whether they want to share their data or not.
Confidentiality	This is the property that ensures that information is not disclosed or made available to any unauthorized entity.
Integrity	Is the property of safeguarding the correctness, consistency, and trustworthiness of data over its entire life cycle in an IoT system.
Availability	Refers to the property which ensures that an IoT device or system is accessible and usable upon demand by authorized entities.
Authorization	Refers to the property that determines whether the user or device has rights/privileges to access a resource, or issue commands.
Forgery Resistance	This is the propriety that ensures that data shared between entities and updates cannot be forged by a third party trying to damage or harm the system or its users.
Non-Repudiation	Refers to the security property that ensures that the transfer of messages or credentials between 2 IoT entities is undeniable.
Confinement	Ensures that even if an entity is hijacked or corrupted, the spreading of the effects of the attack is as confined as possible.
Accountability	This is the property that ensures that every action can be traced back to a single user or device.
Reliability	Is the property that guarantees consistent intended behavior of an IoT system.

Press Enter to return to the Main Menu

Figure 5.5: Final results of SRE request for subject with request ID R5432.

 THE SECURITY REQUIREMENTS FOR THE IoT SYSTEM OF THE USER WITH REQUEST ID No.: R1287

SECURITY REQUIREMENT	DESCRIPTION
Authentication	This is the assurance that a message is from the source it claims to be from.
Privacy	Refers to users control over the disclosure of their personal information, meaning that only the users should decide whether they want to share their data or not.
Confidentiality	This is the property that ensures that information is not disclosed or made available to any unauthorized entity.
Integrity	Is the property of safeguarding the correctness, consistency, and trustworthiness of data over its entire life cycle in an IoT system.
Availability	Refers to the property which ensures that an IoT device or system is accessible and usable upon demand by authorized entities.
Physical Security	Refers to the security measures designed to deny unauthorized physical access to IoT devices or systems, and to protect them from damage or tampering.
Authorization	Refers to the property that determines whether the user or device has rights/privileges to access a resource, or issue commands.
Forgery Resistance	This is the propriety that ensures that data shared between entities and updates cannot be forged by a third party trying to damage or harm the system or its users.
Non-Repudiation	Refers to the security property that ensures that the transfer of messages or credentials between 2 IoT entities is undeniable.
Confinement	Ensures that even if an entity is hijacked or corrupted, the spreading of the effects of the attack is as confined as possible.
Accountability	This is the property that ensures that every action can be traced back to a single user or device.
Reliability	Is the property that guarantees consistent intended behavior of an IoT system.
Counterfeit Resistance	Is the property that ensures effective validation of software such that any fake or maliciously modified software is rejected.
Data Freshness	Ensures that data is the most recent, and that old messages cannot be replayed.
Tamper Detection	Ensures all devices are physically secured, such that any tampering attempt is detected.

Press Enter to return to the Main Menu

Figure 5.6: Final results of SRE request for subject with request ID R1287.

5.2.3 Evaluation and Discussion of the SRE Tool Test Results

This subsection evaluates the performance of the SRE tool. The following evaluation and discussion is based on the two performance metrics mentioned in Subsection 5.2.1.1, namely *security requirements elicitation accuracy of the SRE tool* and *simplicity and ease of use of the SRE tool*. While the evaluation based on the first performance metric focuses only on the three selected results presented in Subsection 5.2.2 above, the evaluation in light of the second performance metric covers the entire 24 entries in the performance evaluation form presented in Table 5.1.

5.2.3.1 Security Requirements Elicitation Accuracy of SRE Tool

This evaluation is focused on the results of the SRE test for subjects with request IDs R1995, R5432, and R1287, as highlighted in Subsection 5.2.2. Depending on a user request, the SRE tool can generate a long list of security requirements. Therefore, for the sake of conciseness, the following evaluation may not cover each and every one of the security requirements that appear in each of the three results under consideration, especially if it has already been discussed in a previous result, unless if there is a need to prove a point. It is also worth mentioning that while 4 of the 24 subjects are very experienced in information security and 2 have adequate security experience (see Table 5.1), the evaluation focuses only on the results of the subjects with little or no security experience.

Arguably, the SRE tool produces an accurate list of security requirements based on user input. This claim can be validated by carrying out a careful examination and a comparative analysis of the requests of those subjects with the aforementioned request IDs (see Tables 5.2 and 5.3) and their corresponding results presented in Figures 5.4, 5.5, and 5.6, respectively. The requests of the subjects under consideration can be summarized as in Tables 5.5, 5.6, and 5.7. Note that in some cases a positive response to not just one, but a number of questions can trigger the need for the inclusion of a particular security requirement in the generated security requirements result for a given user. For example, a positive response to either of the questions: *Will it send data to an entity?* and *Is there a possibility of eavesdropping?* can trigger the need for the inclusion of *Confidentiality* in the list of security requirements. Similarly, the following questions and corresponding user responses can trigger the need for the inclusion of *Forgery Resistance* in the list of security requirements: *What type of information will the system store?* - *Sensitive*, *Will it send data to a cloud?* - *Yes*, *Will it send data to a database?* - *Yes*, and *Will it receive regular updates?* - *Yes*. Consequently, in order to avoid the inclusion of a security requirement multiple times, only the first instance is featured in the generated list of security requirements.

In the request summarized in Table 5.5, the subject with request ID R1995 specified that the IoT device to be designed will have users and a user login which necessitates the need

for users to verify or prove that they are indeed the entities they claim to be, and hence the need for authentication (i.e., the first security requirement in Figure 5.4) which verifies the authenticity of entities. The subject also indicated that the device will share its data with other entities, which may allow attackers to eavesdrop on communications between the device and other entities; attackers can also attempt to modify data in transit between the device and other entities. These necessitate the need for confidentiality which restricts unauthorized access to information and integrity which is concerned with the assurance of the trustworthiness, origin, and correctness of data, respectively (i.e., appearing as the second and third security requirements in Figure 5.4). Moreover, Table 5.5 shows that the device will store some information, and as an IoT device that will be connected to the Internet, the smart toy can be rendered inaccessible to authorized users through DoS attacks, or through MitM attacks by intercepting and destroying the messages which could terminate the communication between the device and other entities and thus causing availability issues. Thus, availability (i.e., the fourth security requirement in Figure 5.4), which ensures accessibility and usability of a system upon demand, is an important security requirement that needs to be met. In addition, attackers can use MitM attacks to hijack or corrupt the device, which underscores the need for confinement (i.e., the fifth security requirement in Figure 5.4), a security requirement that limits the spread of the effect of attacks on other devices. Furthermore, it can be seen from Table 5.5 that someone with malicious intentions can physically access the device. Hence, there is a need for physical security measures that can prevent unauthorized physical access to the device (i.e., the last security requirement in Figure 5.4). Note that the question and user response in item 15 of Table 5.5 has already been taken care of previously as the second security requirement in Figure 5.4. Finally, it is worth noting that smart toy is not considered as a sensitive application domain in the IoT-HarPSecA framework, therefore, security requirements such as non-repudiation, accountability, tamper detection, and reliability are not considered necessary, and hence the SRE tool did not include them in the result shown in Figure 5.4.

Table 5.6, which summarizes the request of the subject with request ID R5432, shows that the smart app will capture and store user information, raising concerns about user privacy. Therefore, privacy is an essential security requirement that is needed to protect the rights of users to the privacy of their personal information, which is the second security requirement in Figure 5.5. Table 5.6 also reveals that the smart app will have users and a user login. But note that the smart grid is considered as a sensitive application domain in the IoT-HarPSecA framework. Thus, based on the sensitivity of the application area, apart from the need for user authentication, there is also a need for users to be assigned a set of permissions, rights, or privileges. Privilege level assignment defines what exactly users can do on a system based on the security policy of an organization, usually specified by the system administrator, which is exactly what authorization (the sixth security requirement in Figure 5.5) provides. Based on the sensitivity of the application domain and

S/No.	Parameter/Presented Question	User Response
1.	Request ID	R1995
2.	Application Domain	Smart Toy
3.	System development phase	Planning
4.	Will the system have a user?	Yes
5.	Will it have a user login?	Yes
6.	Will it store user information?	No
7.	Will it store any other information?	Yes
8.	Type of information	Normal
9.	Will it send data to an entity?	Yes
10.	Will it be on the Internet?	Yes
11.	Will it send data to a cloud?	No
12.	Will it send data to a database?	No
13.	Will it receive regular updates?	No
14.	Will it use third-party software?	No
15.	Is there possibility of eavesdropping?	Yes
16.	Can a message be replayed?	No
17.	Can a user be impersonated?	No
18.	Can someone access it physically?	Yes

Table 5.5: A summary of the SRE request for subject with request ID R1995.

S/No.	Parameter/Presented Question	User Response
1.	Request ID	R5432
2.	Application Domain	Smart Grid
3.	System development phase	Planning
4.	Will the system have a user?	Yes
5.	Will it have a user login?	Yes
6.	Will it store user information?	Yes
7.	Will it store any other information?	Yes
8.	Type of information	Sensitive
9.	Will it send data to an entity?	Yes
10.	Will it be on the Internet?	Yes
11.	Will it send data to a cloud?	No
12.	Will it send data to a database?	No
13.	Will it receive regular updates?	Yes
14.	Will it use third-party software?	No
15.	Is there possibility of eavesdropping?	Yes
16.	Can a message be replayed?	No
17.	Can a user be impersonated?	Yes
18.	Can someone access it physically?	No

Table 5.6: A summary of the SRE request for subject with request ID R5432.

coupled with the fact that Table 5.6 shows that the smart app will send data to other entities, will be connected to the Internet, and will receive regular updates, forgery resistance (i.e., the seventh security requirement in Figure 5.5) is a vital security requirement that can ensure that the app is resistant against message and update forgery that can cause the system to malfunction or to be corrupted. In the same vein, the sensitivity of the application domain necessitates the inclusion of non-repudiation and accountability in the list of security requirements which ensure that entities cannot deny taking part in any action and that every action is traceable to a single entity, respectively (i.e., the eighth and tenth security requirements in Figure 5.5). In addition, the domain sensitivity parameter also leads to the inclusion of reliability, which guarantees consistent intended behavior of a system, which happens to be the last security requirement in Figure 5.5. Note that the questions and user responses in items 15 and 17 of Table 5.6, which would have triggered the inclusion of *Confidentiality* and *Authentication*, respectively, have already been discussed in the previous paragraph (i.e., when dealing with the case of the subject with request ID R1995).

Looking at the request of the subject with request ID R1287, it can be seen that the subject answered *Yes* to all the questions with the *Yes* or *No* options, except for the question in item 11 of Table 5.7. This, coupled with the fact that smart healthcare is considered a critical application domain in the IoT-HarPSecA framework, made the SRE tool to generate up to 15 security requirements, as shown in Figure 5.6. According to Table 5.7, the device will use a third-party software, and hence the need to guard against the use of maliciously modified software that may harm the system, which is what the counterfeit resistance security requirement tries to achieve (i.e., the thirteenth security requirement in Figure 5.6). Furthermore, Table 5.7 shows that attackers can capture and replay a message. Consequently, since the system will have a user login, and that data will be sent to other entities and to a database, there is a need for the data freshness security require-

Table 5.7: A summary of the SRE request for subject with request ID R1287.

S/No.	Parameter/Presented Question	User Response
1.	Request ID	R1287
2.	Application Domain	Smart Healthcare
3.	System development phase	Planning
4.	Will the system have a user?	Yes
5.	Will it have a user login?	Yes
6.	Will it store user information?	Yes
7.	Will it store any other information?	Yes
8.	Type of information	Critical
9.	Will it send data to an entity?	Yes
10.	Will it be on the Internet?	Yes
11.	Will it send data to a cloud?	No
12.	Will it send data to a database?	Yes
13.	Will it receive regular updates?	Yes
14.	Will it use third-party software?	Yes
15.	Is there possibility of eavesdropping?	Yes
16.	Can a message be replayed?	Yes
17.	Can a user be impersonated?	Yes
18.	Can someone access it physically?	Yes

ment which will ensure that data is the most recent and thereby protect the system from replay attacks; data freshness is the fourteenth security requirement in Figure 5.6. Finally, Table 5.7 shows that someone with malicious intentions can physically access the device. However, due to the sensitivity of the application domain, aside from the need for taking physical security measures, there is also a need to detect any active attempt to compromise the integrity of the device or the data associated with it, and this is what the tamper detection security requirement is intended to meet, which is the last security requirement in Figure 5.6. Note that the questions and user responses in Table 5.7 that are not discussed here have already been discussed in either of the two previous cases above (i.e., in the case of the subject with request ID R1995 or the case of the subject with request ID R5432).

5.2.3.2 Simplicity and Ease of Use of the SRE Tool

In software performance analysis, efficient performance plays a more important and fundamental role in determining the usability of a tool compared to its simplicity of use. However, although simplicity is not the best parameter for measuring usability, an easy-to-use user interface is a desirable feature for many users. While there are many ways to evaluate the simplicity of a tool, a good approach is to allow potential users themselves to try the tool and provide their feedback. Accordingly, a performance evaluation form has been designed which the 24 subjects have completed as mentioned in Subsection 5.2.2; and as it can be seen in Table 5.1, simplicity of use is among the evaluation criteria.

Users will usually have different perceptions of simplicity irrespective of the type of interface the tool has. The user interface of the IoT-HarPsecA framework tool is currently not a GUI, but it is arguably easy to use. Some useful notifications that can guide users as they

interact with the different components of the tool to ease task execution have been provided. Regarding the ease of use of the SRE and SBPG tools, the logical flow and sequential order in which the questions are administered, as well as the simple multiple-choice questions format implemented in the questionnaires allow users to answer the questions easily. However, the focus here is on the SRE tool.

The two criteria used for evaluating the simplicity and ease of use of the SRE tool are shown in Table 5.1, namely simplicity of use and request completion time in seconds. Figure 5.1 depicts the professional competence and security experience of the subjects. In Figure 5.1, 13 subjects have a little security experience and 5 subjects have no security experience at all (i.e., 2.5 and 0, respectively, on the numeric rating scale). Although the author claims that the SRE tool is easy to use, in Figure 5.2(b), 54% of the 24 subjects went to the extreme by arguing that the tool is very easy to use; 42% agree with the author that the tool is easy to use, and 4% indicate that the simplicity of usage of the tool is average. Figure 5.2(a) shows the simplicity scores of individual subjects. In addition, Figure 5.3 shows the request completion time for each subject, however, comparing Figure 5.2(a) with Figure 5.3 reveals no significant correlation between the simplicity of use and request completion time. For instance, the request completion time for the subject with request ID R1115 is 145.5 seconds but the subject indicated that the tool is very easy to use. On the other side of the spectrum, the request completion time for the subject with request ID R5461 is 93.2 seconds which is considerably less than 145.5, however, the subject indicated that the tool is easy to use.

5.3 Test, Results, and Performance Evaluation of the SBPG Tool

This Section presents the test that is used to evaluate the performance and usability of the SBPG tool. As it was highlighted in Section 5.1, both the SRE and SBPG tools tests were conducted about the same time. Like the SRE tool evaluation, the evaluation of the SBPG tool is also based on two performance metrics that will be presented in Subsection 5.3.1.1.

5.3.1 SBPG Tool Test Setup

In a similar manner to the case of the SRE tool test setup, setting up the performance and usability test of the SBPG tool involves creating a real-world-like testing scenario based on the intended functionality of the SBPG component of the IoT-HarPSECA framework described in Section 4.3. The targeted subjects are people that are actively involved in the design and implementation of IoT systems. A test that mimics a real-world interaction of a user with the SBPG tool is briefly described in the following subsection and the evaluation of the result summaries is presented in Subsection 5.3.3.

5.3.1.1 SBPG Tool Test Scenario

As in the case of the SRE tool test scenario, this test is also based on the two major criteria needed to be assessed, namely functional accuracy and usability. *Test scenario description:* consider a typical use case for making an SBPG request where a user is trying to ascertain the security best practice guidelines for designing and developing a secure IoT device or a smart app. The system can be in its early phase of development (i.e., conception, planning, design) or an existing IoT system.

The group of subjects that participated in this test is the same group that participated in the SRE tool test, and as in the case of the SRE tool test, every subject was given a brief description of the SBPG tool functionality prior to the commencement of the test. The following performance metrics are used to assess the performance of the SBPG tool:

1. security best practices accuracy of the SBPG tool;
2. simplicity and ease of use of the SBPG tool.

5.3.2 Results Overview of the SBPG Tool Test

The basic information about the composition of the subjects that conducted the test, as well as the information required from the subjects while completing the evaluation form and the rationale behind the design of the evaluation form in such a manner, were already provided in Subsection 5.2.2. Consequently, this section simply presents the data in the performance evaluation form as completed by the 24 subjects as shown in Table 5.8, as well as presents the summaries of test results of three subjects.

The entries in the first six columns of Table 5.8 are the same with those of Table 5.1 since the same group of subjects participated in both SRE and SBPG tools tests. However, the entries in the last two columns, namely simplicity of use and request completion time may be different. Tables 5.9, 5.10, and 5.11 show the data of the requests of the subjects stored in the database. Tables 5.10 and 5.11 are continuation of Table 5.9. However, the screenshots of the data of the actual requests of the subjects as stored in the MySQL database table can be found in Figure B.2 in Appendix B. Table 5.12 provides more clarity on the designations of the 30 columns in Tables 5.9, 5.10, and 5.11 (or Figure B.2) that are unique to the column designation descriptions provided in Table 5.4. The numerical scaling data extracted from the performance evaluation form presented in Table 5.8 are used to represent the information graphically, which are apparently more readable, as shown in Figures 5.7 and 5.8. Note that the chart representing the professional competence and security experience of individual subjects is already presented in Figure 5.1 in Subsection 5.2.2 since the same group of subjects performed the tests on the two tools (i.e., SRE and SBPG tools).

It was mentioned in Section 5.1 that only the summaries of three SBPG test results will be presented in this chapter due to space constraints. The selected results are for sub-

Table 5.8: SBPG tool performance evaluation form as completed by subjects.

S/No.	Request ID	Age Range	Security Experience	Field of Expertise	Proficiency Level	Simplicity of Use	Request Completion Time (s)
1.	B1111	A	B (7.5)	D	B (7.5)	A (10.0)	72.6
2.	B2345	C	B (7.5)	C	C (5.0)	A (10.0)	172.9
3.	B4444	A	D (2.5)	D	C (5.0)	A (10.0)	125.8
4.	B1234	A	D (2.5)	D	C (5.0)	A (10.0)	76.6
5.	B6548	A	D (2.5)	C	B (7.5)	B (7.5)	51.9
6.	B0601	D	D (2.5)	C	C (5.0)	A (10.0)	100.7
7.	B7788	C	D (2.5)	C	A (10.0)	B (7.5)	93.1
8.	B5432	A	E (0.0)	D	C (5.0)	B (7.5)	98.3
9.	B2278	B	E (0.0)	D	C (5.0)	C (5.0)	136.4
10.	B9128	A	D (2.5)	C	C (5.0)	B (7.5)	151.6
11.	B6666	A	D (2.5)	C	B (7.5)	A (10.0)	85.9
12.	B1115	A	C (5.0)	D	A (10.0)	A (10.0)	149.0
13.	B1995	B	D (2.5)	E	C (5.0)	B (7.5)	187.1
14.	B4321	B	B (7.5)	C	B (7.5)	B (7.5)	163.8
15.	B1235	C	E (0.0)	E	B (7.5)	B (7.5)	126.4
16.	B8374	A	B (7.5)	C	A (10.0)	A (10.0)	99.2
17.	B4040	C	C (5.0)	E	B (7.5)	A (10.0)	80.1
18.	B8789	D	D (2.5)	C	C (5.0)	B (7.5)	89.6
19.	B1008	B	D (2.5)	C	A (10.0)	B (7.5)	90.7
20.	B5555	C	E (0.0)	E	B (7.5)	B (7.5)	103.5
21.	B1287	D	D (2.5)	C	B (7.5)	A (10.0)	98.1
22.	B5461	B	D (2.5)	C	B (7.5)	A (10.0)	89.7
23.	B0011	D	E (0.0)	E	A (10.0)	B (7.5)	105.2
24.	B8126	A	D (2.5)	C	C (5.0)	A (10.0)	92.1

jects with request IDs B5555, B2278, and B7788. The rationale for selecting the three result summaries is as explained in the last paragraph of Subsection 5.2.2. These results consist of 5, 9, and 14 summaries of security best practice guidelines for subjects with the above-mentioned request IDs, respectively. While the result summary of the SBPG test for the subject with request ID B5555 is presented in Figure 5.9, the result summaries of the SBPG test for subjects with request IDs B2278 and B7788 are presented in Figures C.1 and C.2 in Appendix C, respectively, due to their large size. Also note that the extended version of the SBPG test result (i.e., full-length result) for subject with request ID B5555 is shown in Figure A.1 in Appendix A as mentioned in Section 5.1. The evaluation of the result summaries is presented in Subsection 5.3.3.1.

5.3.3 Evaluation and Discussion of the SBPG Tool Test Results

This subsection evaluates the performance of the SBPG tool, and as in the case of SRE tool evaluation, this evaluation and discussion is based on the two performance metrics mentioned in Subsection 5.3.1.1, namely *security best practices accuracy of SBPG tool* and *simplicity and ease of use of SBPG tool*. Although the evaluation based on the first performance metric centered only on the three selected result summaries presented in Subsection 5.3.2, the evaluation in regard to the second performance metric focuses on the entire 24 entries in the performance evaluation form presented in Table 5.8.

Table 5.9: SBPG request data of subjects as stored in the database.

Request_ID	Status	Struct1	Struct2	Struct3	Struct4	Struct5	Struct6	Struct7	Struct8	Struct9	Struct10	Struct11
B1111	NotYet	Device_Mgmt	Data_Collect									
B2345	NotYet	Device_Mgmt	Data_Collect									
B4444	Exist		Data_Collect									
B1234	NotYet									Web		
B6548	NotYet						Data_Process	Analytics				
B0601	NotYet						Data_Process					
B7788	Exist				API Service					Web		
B5432	NotYet									Web		
B2278	NotYet	Device_Mgmt				Data_Mgmt	Data_Process					
B9128	NotYet									Web		
B6666	NotYet	Device_Mgmt	Data_Collect					Analytics				
B1115	NotYet	Device_Mgmt	Data_Collect			Data_Mgmt	Data_Process	Analytics	Cloud Service			
B1995	NotYet	Device_Mgmt										
B4321	NotYet	Device_Mgmt										
B1235	NotYet						Data_Process					
B8374	NotYet					Data_Mgmt						
B4040	NotYet	Device_Mgmt										
B8789	Exist									Web		
B1008	NotYet	Device_Mgmt										
B5555	NotYet	Device_Mgmt										
B1287	NotYet									Web		
B5461	NotYet						Data_Process					
B0011	Exist	Device_Mgmt										
B8126	NotYet									Web		

Table 5.10: Continuation of SBPG request data of subjects.

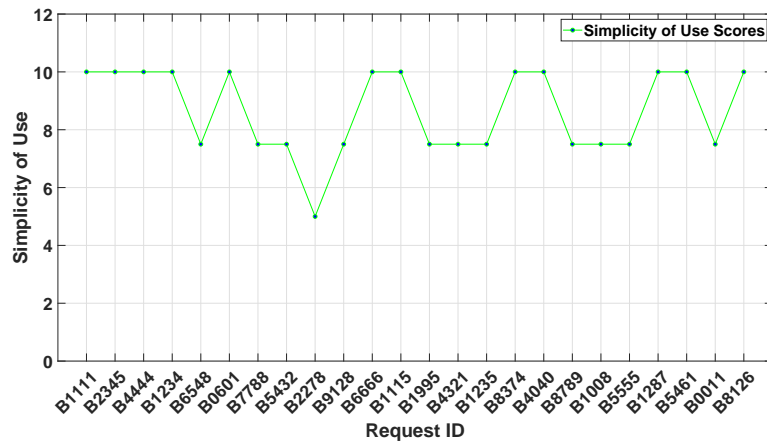
any-Usr	usr-Regist	typeOf-Regist	anyUsr-Login	usr-Input	hold-UsrInfo	store-AnyInfo	sensitiv-OfInfo	typeOfAUTH	useDb	typeOf-DataStorg
No						Yes	Normal		Yes	SQL
Yes	Yes	Users	Yes	Yes	Yes	Yes	Normal	usernam _passw	Yes	SQL
Yes	No			Yes	Yes	No	Sensitive	usernam _passw	Yes	NoSQL
Yes	Yes	Admin	Yes	Yes	Yes	Yes	Critical	2Factor _AUTH	Yes	SQL
Yes	Yes	Admin	Yes	Yes	Yes	Yes	Sensitive	usernam _passw	Yes	SQL
Yes	Yes	Users	Yes	Yes	Yes	Yes	Normal	usernam _passw	Yes	SQL
Yes	Yes	Admin	Yes	Yes	Yes	Yes	Critical	2Factor _AUTH	Yes	SQL
No						Yes	Sensitive		Yes	SQL
Yes	Yes	Admin	Yes	Yes	Yes	Yes	Sensitive	usernam _passw	Yes	Distr _Storage
Yes	Yes	Users	Yes	No	Yes	No	Normal	SociNet _Email	Yes	SQL
No						Yes	Normal		Yes	SQL
Yes	Yes	Users	Yes	No	Yes	Yes	Normal	usernam _passw	Yes	SQL
No						Yes	Normal		Yes	Local _Storage
Yes	No							usernam _passw	Yes	Local _Storage
Yes	Yes	Users	Yes	Yes	Yes	Yes	Normal	usernam _passw	Yes	Distr _Storage
Yes	Yes	Users	Yes	No	Yes	No	Sensitive	MultFact _AUTH	Yes	SQL
Yes	No			Yes	No	Yes	Normal	No _AUTH	No	
Yes	Yes	Users	Yes	Yes	No	Yes	Normal	usernam _passw	No	
Yes	Yes	Users	Yes	Yes	Yes	No	Normal	No _AUTH	Yes	SQL
No						Yes	Normal		No	
Yes	Yes	Users	Yes	Yes	No	No		No _AUTH	No	
Yes	Yes	Users	Yes	Yes	No	Yes	Sensitive	usernam _passw	Yes	SQL
Yes	Yes	Users	Yes	Yes	No	Yes	Normal	usernam _passw	No	
Yes	Yes	Users	Yes	Yes	No	Yes	Critical	usernam _passw	Yes	SQL

Table 5.11: Continuation of SBPG request data of subjects.

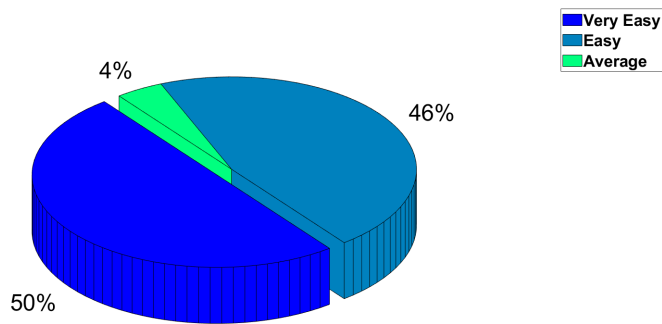
typeOfDb	progrm1	progrm2	progrm3	progrm4	progrm5	progrm6	progrm7	progrm8	file-Upload	sys-Log
MySQL						Python			No	Yes
SQLite						Python			Yes	Yes
SQL_Server				JavaSc		Python			Yes	Yes
MySQL		C/C++			PHP				No	Yes
MySQL					PHP	Python			Yes	Yes
Post-greSQL						Python			Yes	Yes
SQLite	C#				PHP				No	Yes
SQL-Server				JavaSc					Yes	Yes
SQL-Server		C/C++		JavaSc					Yes	No
Post-greSQL		C/C++		JavaSc		Python			Yes	Yes
SQL-Server						Python			No	Yes
MySQL						Python			Yes	Yes
MySQL		C/C++				Python			No	Yes
SQL-Server		C/C++				Python			No	Yes
SQL-Server	C#		Java						Yes	Yes
SQL-Server			Java						No	Yes
		C/C++							No	No
			Java						No	Yes
SQL-Server			Java						Yes	Yes
SQL-Server		C/C++							No	Yes
						Python			No	Yes
MySQL						Python			Yes	No
		C/C++							Yes	No
SQLite			Java	JavaSc					Yes	Yes

Table 5.12: Description of the designations of the 30 unique columns in Tables 5.9, 5.10, and 5.11.

S/No.	Column Designation	Brief Description
1.	Request_ID	User request ID
2.	Status	System development phase: <i>NotYet</i> means early stage of development and <i>Exist</i> means it is an existing IoT system
3.	Struct1	System Architecture or Purpose (SAoP) – <i>Device Management</i>
4.	Struct2	SAoP – <i>Data Collection</i>
5.	Struct3	SAoP – <i>Connectivity Management</i>
6.	Struct4	SAoP – <i>API Services</i>
7.	Struct5	SAoP – <i>Data Management</i>
8.	Struct6	SAoP – <i>Data Processing</i>
9.	Struct7	SAoP – <i>Big Data Analytics/Advanced Analytics</i>
10.	Struct8	SAoP – <i>Data Center and Cloud Services</i>
11.	Struct9	SAoP – <i>Web Services/Web apps</i>
12.	Struct10	SAoP – <i>Embedded Systems</i>
13.	Struct11	SAoP – <i>Other</i>
14.	usrRegist	Will there be user registration?
15.	typeOfRegist	Type of user registration
16.	usrInput	Will users provide any input?
17.	typeOfAUTH	Type of authentication
18.	useDb	Will is system use a database
19.	typeOfDataStorg	Type of data storage
20.	typeOfDb	Type of database
21.	progrm1	Programming Language(s) to be Used (PLtbU) – <i>C#</i>
22.	progrm2	PLtbU – <i>C/C++</i>
23.	progrm3	PLtbU – <i>Java</i>
24.	progrm4	PLtbU – <i>JavaScript</i>
25.	progrm5	PLtbU – <i>PHP</i>
26.	progrm6	PLtbU – <i>Python</i>
27.	progrm7	PLtbU – <i>Ruby</i>
28.	progrm8	PLtbU – <i>Other</i>
29.	fileUpload	Will the system allow file uploads?
30.	sysLog	Will there be system logs?



(a) SBPG tool simplicity of use scores of individual subjects



(b) Percentage representation of the simplicity of use of the SBPG tool

Figure 5.7: Simplicity and ease of use of the SBPG tool.

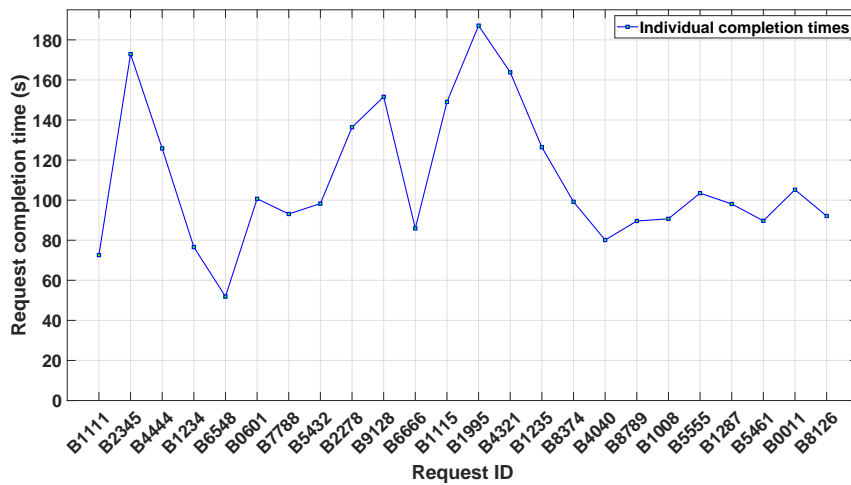


Figure 5.8: Request completion times of subjects for the SBPG tool test.

 THE SUMMARY OF SECURITY BEST PRACTICES FOR USER WITH REQUEST ID No.: B5555

S/No	SECURITY BEST PRACTICES
1	Strong device authentication is necessary to ensure that connected devices can be trusted to be what they claim to be. Hence, where applicable adopt strong password authentication; where possible, implement two factor authentication. Ensure secure boot, use tamper-resistant hardware-based storage like TPM, ensure that each stage of boot code is trusted before running it, and ensure that no boot sequence is skipped. Ensure that devices are shipped with latest and stable versions of OSes, and with proper OS security configuration. Also ensured that OSes can boot securely; use good password management techniques. Do not use the same keys when implementing encryption on many IoT devices; bake security into every stage of smart apps development lifecycle; and disable every port and interfaces that were installed on IoT devices for testing purposes.
2	Ensure that any newline characters in system log files are appropriately handled to prevent log forging; and ensure that any logged HTML characters are appropriately encoded to prevent XSS when viewing logs.
3	Do not use algorithms and protocols that are not vetted by the cryptographic community; ensure that certificates are properly validated against the hostnames whom they are meant for. To reduce the risk of compromising many servers, avoid using wildcard certificates unless there is a business need for it. Store only the sensitive data that you need. If a password is being used to protect keys then the password strength should be sufficient for the strength of the keys it is protecting. Use cryptographically strong random numbers for cryptographic parameters like keys.
4	Ensure that IoT and IIoT data both in-flight and at-rest is encrypted, and be very careful when selecting and implementing cryptographic algorithms because a cryptographic algorithm is only as strong as how it is implemented. Avoid using insecure protocols such as File Transfer Protocol (FTP) and Telnet because of lack of encryption, and their reliance on clear-text usernames and passwords for authentication.
5	Ensure that smart devices that will be deployed in open environments are securely protected using strong casing. If possible, protect IoT device circuitry from tampering using resin encapsulation and epoxy resin, etc.

Press Enter to process the detailed version of best practices

Figure 5.9: Final results of SBPG request for subject with request ID B5555.

Table 5.13: A summary of the SBPG request for subject with request ID No. B5555.

S/No.	Parameter/Presented Question	User Response
1.	Request ID	B5555
2.	IoT system architecture or purpose	Device Management
3.	System development phase	Early Phase
4.	Will the system have users?	No
5.	Will it store any kind of information?	Yes
6.	Type of information	Normal
7.	Will it store data in a database?	No
8.	What programming language(s) will be used to implement it?	C/C++
9.	Will it allow file uploads?	No
10.	Will it generate log files?	Yes

5.3.3.1 Security Best Practices Accuracy of SBPG Tool

This evaluation focuses on the result summaries of the SBPG test for subjects with request IDs B5555, B2278, and B7788, as highlighted in 5.3.2. To avoid unnecessary repetition, the following evaluation may not cover each and every one of the security best practices that appear in each of the three results under consideration, especially if it has been discussed in a previous result summary, unless if it is meant to clarify or highlight some key points.

The author of this thesis claims that the SBPG tool arguably generates an accurate list of security best practice guidelines based on user input. This claim can be substantiated or not through a careful examination and comparative analysis of the requests of those subjects with the aforementioned request IDs (see Tables 5.9, 5.10, and 5.11) and their corresponding results presented in Figures 5.9, C.1, and C.2, respectively. The requests of the subjects under consideration can be summarized as in Tables 5.13, 5.14, and 5.15. However, it is important to note that the order in which the security best practices appear in Figures 5.9, C.1, and C.2 does not necessarily follow the order of the appearance of the questions in the summarized requests in Tables 5.13, 5.14, and 5.15.

Table 5.13 summarizes the request of the subject with request ID B5555 which reveals the system architecture of the IoT system to be designed, namely device management. This implies that the IoT system may be an IoT device, a smart sensor, actuator, or a smart gateway that is in its early stage of development, which could be the conception, planning, or design stage. Comparing the request summary in Table 5.13 with those in Tables 5.14 and 5.15 reveals that five questions were skipped over in the request of the subject with request ID B5555, implying that those questions were not administered to the subject. This is because the subject gave a negative answer to the question that asks whether or not the system will have users.

Based on the information in Table 5.13 which shows that the device will not store any sensitive data and that no data will be sent to a database, it may seem unnecessary to secure the device. However, if every component of an IoT network is properly secured and

Table 5.14: A summary of the SBPG request for subject with request ID No. B2278.

S/No.	Parameter/Presented Question	User Response
1.	Request ID	B2278
2.	IoT system architecture or purpose	Device management Data management Data processing
3.	System development phase	Early phase
4.	Will the system have users?	Yes
5.	Will there be provision for user registration?	Yes
6.	Who will register users?	Admin
7.	Will the system have a user login?	Yes
8.	Will it allow users to enter any input?	Yes
9.	Will it store user information?	Yes
10.	Will it store any other information?	Yes
11.	Type of information	Sensitive
12.	What type of authentication will be implemented?	username and password
13.	Will it store data in a database?	Yes
14.	What will be the type of data storage?	Distributed Storage
15.	What type of database will be used?	SQL Server
16.	What programming language(s) will be used to implement it?	C/C++ JavaSc
17.	Will it allow file uploads?	Yes
18.	Will it generate log files?	No

tightened but one component is not, then the whole network may be vulnerable to attacks. This is because the security level of an IoT system is as good as the security level of the weakest link in the network. Moreover, attackers always look for the least resistant path to their target. Consequently, there is a need to take some basic measures to secure the device. Thus, a few general security best practices for the design of secure IoT devices have been provided in items 1, 3, 4, and 5 of Figure 5.9. Additionally, in Table 5.13, the subject specified that the system will generate log files, and hence a security best practice for implementing log files is provided in item 2 of Figure 5.9.

In Table 5.14, the summary of the request of the subject with request ID B2278 shows that the subject has selected device management, data management, and data processing as the functionalities of the smart app, which is in its early stage of development. The table also shows that the subject answered positively to the question that asks whether or not the system will have a user. Therefore, the five questions skipped in Table 5.13 are shown in Table 5.14 and as it can be seen, the subject answered *Yes* to all those questions (where applicable) and to the others with the *Yes* or *No* options, except for the question in item 18 where the subject answered *No*.

The first security best practice in Figure C.1 is in response to the answer to the question in item 6 of Table 5.14 where the subject answered that *Admin* will register users. The second security best practice is in response to the answer to the question in item 18 of Table 5.14 where the subject indicated that *file uploads* will be allowed. The third security best practice in Figure C.1 is in response to the answers to the questions in items 8 and 12 of Table 5.14 in which the subject specified that users will be allowed to *enter inputs* and

Table 5.15: A summary of the SBPG request for subject with request ID No. B7788.

S/No.	Parameter/Presented Question	User Response
1.	Request ID	B7788
2.	IoT system architecture or purpose	API Service Web Application
3.	System development phase	Existing System
4.	Does the system have users?	Yes
5.	Does it have provision for user registration?	Yes
6.	Who registers users?	Admin
7.	Does the system have a user login?	Yes
8.	Does it allow users to enter any input?	Yes
9.	Does it store user information?	Yes
10.	Does it store any other information?	Yes
11.	Type of information	Critical
12.	What type of authentication will be implemented?	2 Factor Authentication
13.	Does it store data in a database?	Yes
14.	What is the type of data storage?	SQL
15.	What type of database used?	SQLite
16.	What programming language(s) was used in implementing it?	C#
17.	Does it allow file uploads?	No
18.	Will it generate log files?	Yes

that users will use *username and password* for authentication, respectively. The fourth and ninth security best practices in Figure C.1 are in response to the question in item 2 of Table 5.14, where the subject included *device management* in the system functionalities which actually has to do with IoT *devices* as in the case of the request of the subject with request ID B5555 discussed above. The fifth security best practice is in response to the answers to the questions in items 13, 14, and 15 of Table 5.14 which are concerned with data storage in *database*. Furthermore, the sixth security best practice in Figure C.1 is in response to the answer to the question in item 12 of Table 5.14, where the subject specified that users will use *username and password* for authentication. While the seventh and eighth security best practices in Figure C.1 are in response to the answers to questions 9, 10, and 11 in Table 5.14, the two security best practices are part of the general security best practices for the development of secure IoT devices and smart apps.

Table 5.15 summarizes the request of the subject with request ID B7788 which shows that the subject has selected API service and web application as the functionalities of the existing IoT system, resulting in the generation of the second and third security best practices in Figure C.2, respectively. In addition, the sixth, eighth, twelfth, thirteenth, and fourteenth security best practices in Figure C.2 are also generated as a result of the second system functionality which is *web application*.

Like the case of the subject with request ID B2278, Table 5.15 also shows that the subject responded positively to the question that asks whether or not the system will have a user. Consequently, the five questions skipped in Table 5.13 are also shown in Table 5.15 and as it can be seen, the subject answered *Yes* to all of those questions (where applicable) and to the others with the *Yes* or *No* options, except for the question in item 17 where the subject

answered *No*.

5.3.3.2 Simplicity and Ease of Use of SBPG Tool

In order to evaluate the simplicity of use of the SBPG tool, a performance evaluation form has been designed, which the 24 subjects have completed, as in the case of the SRE tool mentioned in Subsection 5.3.2. In a similar manner to the case of the SRE tool, the two criteria used for evaluating the simplicity and ease of use of the SBPG tool are the two rightmost columns of Table 5.8, namely *simplicity of use* and *request completion time* in seconds. The professional competence and security experience of the subjects is as depicted in Figure 5.1, already presented in Subsection 5.2.3.2.

Like in the case of the SRE tool, although the claim of the author is that the SBPG tool is easy to use, in Figure 5.7(b), 50% of the 24 subjects stated that the tool is very easy to use; 46% agree with the author that the tool is easy to use, and 4% indicate that the simplicity of usage of the tool is average. Figure 5.7(a) shows the simplicity scores of individual subjects. Furthermore, Figure 5.8 shows the request completion time for each subject, but comparing Figure 5.7(a) with Figure 5.8 shows that there is no noticeable correlation between the simplicity of use and request completion time. For example, the request completion time for the subject with request ID B2345 is 172.9 seconds, and yet the subject indicated that the tool is very easy to use. On the other hand, the request completion time for the subject with request ID B2278 is 136.4 seconds which is significantly less than 172.9, however, the subject indicated that the simplicity of use is average.

5.4 Tests, Results, and Performance Evaluation of the LWCAR Tool

This section presents a series of tests aimed at evaluating the performance and usability of the LWCAR tool. As mentioned in Section 5.1, the LWCAR component is the main component of the IoT-HarPSecA framework and the evaluation of the LWCAR tool was carried out much earlier than the evaluation of the SRE and SBPG tools. The LWCAR tool is evaluated based on three performance metrics that are presented in Subsection 5.4.1.1.

5.4.1 LWCAR Tool Test Setup

In order to test the functionality of the LWCAR tool as well as to ensure that it is intuitive to use, four test scenarios were created: two for software implementation requests, and two for hardware implementation requests. For resource-constrained IoT platforms, the number of security requirements is usually kept as few as possible. But for the less constrained platforms, such as SBCs, a user may have as many security requirements as

he/she wants. Thus, for the software implementation use case, there are two application scenarios based on the type of platform: (1) a scenario where the user security requirements are not more than three, and (2) a scenario where a user has more security requirements. The two types of hardware used in the hardware implementation request test scenarios are the ASICs and FPGAs, which currently are the only platforms supported for hardware implementation requests in the LWCAR tool. The aforementioned tests are designed to be carried out by a group of subjects. Moreover, as highlighted in Subsection 4.4.1, users that have used the SRE tool can import their security requirements into the LWCAR tool. Thus, in addition to the four test scenarios mentioned above, a standalone test scenario has been created to demonstrate and validate this functionality, which happens to be the fifth test scenario.

5.4.1.1 LWCAR Tool Test Scenarios

The above mentioned test scenarios are briefly described below:

1. **Software Implementation Test Scenario with MCUs:** In this scenario, a user designing an IoT system using an MCU (or a smart app that will run on an MCU) wants to know the specific LWCAs for software implementation that will satisfy the security requirements of the IoT system he/she wants to design;
2. **Software Implementation Test Scenario with SBCs:** This scenario also consists of a software implementation request, in which a user building a smart device using a SBC (or a smart app that will run on a SBC) hardware is requesting to know the right LWCAs that will meet the security requirements of his/her IoT system;
3. **Hardware Implementation Test Scenario with ASICs:** In this scenario, a user is requesting to know the appropriate LWCAs that will provide the security mechanisms needed to fulfill the security requirements of the new IoT system he/she is building using an ASIC;
4. **Hardware Implementation Test Scenario with FPGAs:** In this hardware implementation request scenario, a user is using an FPGA hardware to design an IoT system and needs to know the LWCAs for hardware implementation that may be used to achieve the security requirements of the IoT system he/she is designing;
5. **A Test Scenario that Demonstrates Importing Security Requirements from the SRE tool into the LWCAR tool:** In this test scenario, a user first used the SRE tool to generate the security requirements of the system he/she wants to design and then imports the generated security requirements into the LWCAR tool as part of his/her inputs.

The usability tests involved inviting unbiased participants (i.e., subjects that had no previous knowledge of the tool). These subjects were invited sometimes in an ad hoc manner,

Table 5.16: LWCAR tool performance evaluation form as completed by subjects.

S/No.	Request ID No.	Age Range	Security Experience	Field of Expertise	Proficiency Level	Simplicity of Use	Request Completion Time (s)
1.	H1598	C	D (2.5)	E	A (10.0)	B (7.5)	116.79
2.	S8888	A	C (5.0)	D	C (5.0)	B (7.5)	75.40
3.	S3833	A	C (5.0)	D	C (5.0)	B (7.5)	83.10
4.	S6868	A	D (2.5)	D	C (5.0)	B (7.5)	81.13
5.	S6789	A	B (7.5)	D	A (10.0)	A (10.0)	29.57
6.	H3456	A	D (2.5)	C	B (7.5)	C (5.0)	114.51
7.	S6000	C	C (5.0)	D	C (5.0)	C (5.0)	86.53
8.	S4219	B	E (0.0)	D	C (5.0)	B (7.5)	77.64
9.	H8791	B	E (0.0)	C	C (5.0)	B (7.5)	108.67
10.	S4452	C	D (2.5)	D	A (10.0)	B (7.5)	73.23
11.	H2648	A	E (0.0)	E	C (5.0)	B (7.5)	115.80
12.	S1682	B	D (2.5)	D	A (10.0)	B (7.5)	79.11
13.	S6001	C	C (5.0)	D	C (5.0)	C (5.0)	56.36
14.	H9850	B	D (2.5)	E	C (5.0)	B (7.5)	118.80
15.	H5942	C	D (2.5)	C	B (7.5)	B (7.5)	111.23
16.	S0001	B	A (10.0)	D	A (10.0)	B (7.5)	67.31
17.	S7788	C	D (2.5)	D	A (10.0)	B (7.5)	82.19

Table 5.17: Description of the designations of the 14 columns in Table 5.22.

S/No.	Column Designation	Brief Description
1.	Request_ID	User request ID
2.	Hardware_type	Hardware type
3.	CPU	Hardware CPU
4.	FlashMem_size	Flash memory size
5.	Ram_size	RAM size
6.	Clock_speed	CPU clock speed
7.	Applic_Domain	Application domain
8.	Payload_size	Message payload size
9.	Req_1	First security requirement
10.	Req_2	Second security requirement
11.	Req_3	Third security requirement
12.	Req_4	Fourth security requirement
13.	Req_5	Fifth security requirement
14.	Req_6	Sixth security requirement

by taking the tool to scientific events and asking people to participate. Besides a brief introduction of the tool at the beginning, subjects were allowed to independently carry out the tests, and they were allowed to select the scenarios they wanted to test for.

To assess the performance and usability of the LWCAR tool, the following performance metrics are used:

1. comparison of the decision-making process of the LWCAR tool versus the decision-making process of human experts;
2. decision accuracy of the LWCAR tool;
3. simplicity and ease of use of the LWCAR tool.

Table 5.18: Description of the designations of the 13 unique columns in Tables 5.23 and 5.24.

S/No.	Column Designation	Brief Description
1.	Energy	Energy requirement
2.	cct_area_1	Circuit area for the first security requirement
3.	tp_1	throughput for the first security requirement
4.	cct_area_2	Circuit area for the second security requirement
5.	tp_2	throughput for the second security requirement
6.	cct_area_3	Circuit area for the third security requirement
7.	tp_3	throughput for the third security requirement
8.	cct_area_4	Circuit area for the fourth security requirement
9.	tp_4	throughput for the fourth security requirement
10.	cct_area_5	Circuit area for the fifth security requirement
11.	tp_5	throughput for the fifth security requirement
12.	cct_area_6	Circuit area for the sixth security requirement
13.	tp_6	throughput for the sixth security requirement

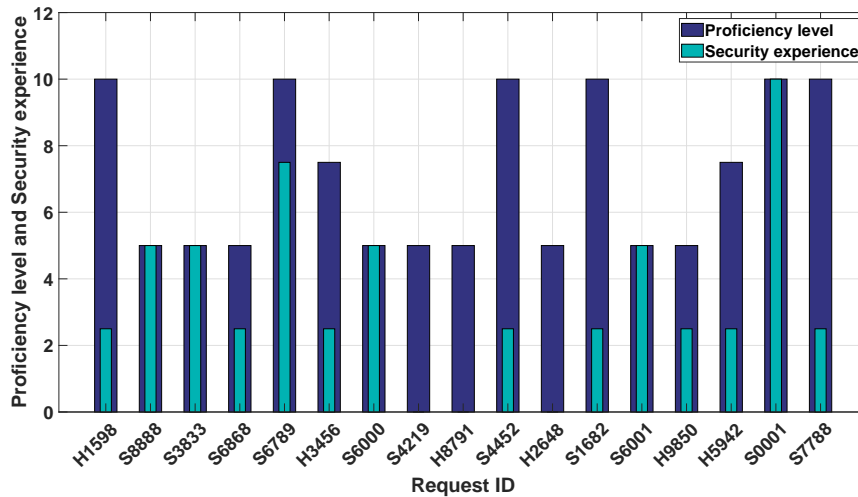


Figure 5.10: Professional competence and security experience of the 17 subjects.

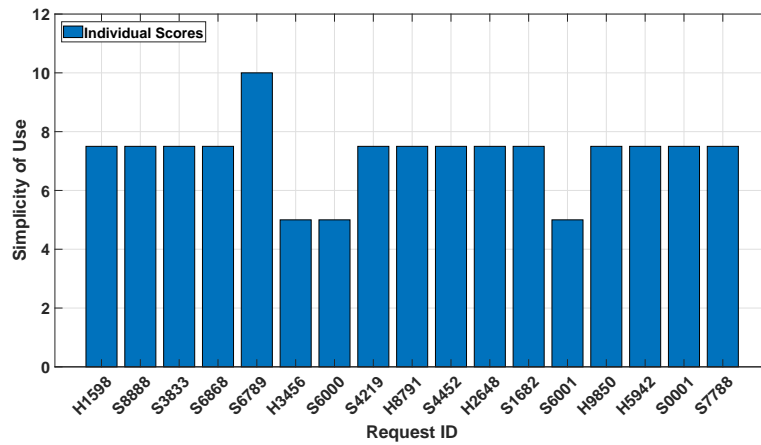
5.4.2 Results Overview of the LWCAR Tool Tests

The LWCAR tool tests were conducted on a Windows 10 desktop computer running the IoT-HarPSecA framework tool. Most of the procedures and the composition of the subjects are similar to those described in Subsections 5.2.2 and 5.3.2, except for the number of the subjects involved. In this case, 17 subjects consisting of both males and females with ages ranging between 18 and 49 years participated in the first four tests (i.e., 11 developers, 3 computer engineers, and 3 electronics engineers). The fifth test was performed by another six subjects (different from the subjects mentioned above). But their SRE and LWCAR tool tests request data are not presented here since they are very similar to those already shown in Tables 5.2-5.3 and Tables 5.22-5.24, respectively (see Figures B.4 and B.5 in Appendix B for the actual screenshots of their requests data as stored in the MySQL database). However, the summaries of the request data for one of the six subjects that are selected for the purpose of this evaluation are shown in Tables 5.20 and 5.21. Further details on this are provided in the third and the last paragraphs of this subsection.

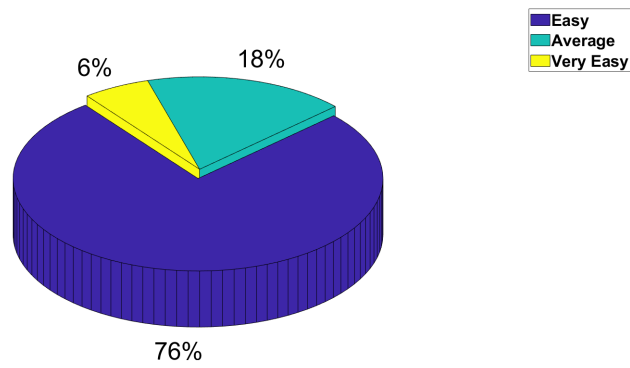
Table 5.19: Summary of the final results of the LWCAR tool tests for all subjects.

S/No.	Request ID No.	User Security Requirements	Security Mechanisms	Security Algorithms
1.	H1598	- Data Confidentiality - Authentication - Confidentiality & authenticity	Encryption MAC Authenticated encryption	*No matching algo found! *No matching algo found! ACORN
2.	S8888	- Data confidentiality/User privacy - Message Integrity - Authentication - Non-repudiation	Encryption Hash function MAC Digital signature	Clefiat128/192 PHOTON-256/32/32 SipHash-128 *No matching algo found!
3.	S3833	-Message Integrity - Non-repudiation - Confidentiality & authenticity	Hash function Digital signature Authenticated encryption	PHOTON-128/16/16 *No matching algo found! CLOC-AES
4.	S6868	- Data confidentiality/User privacy - Message Integrity - Authentication - Non-repudiation	Encryption Hash function MAC Digital signature	Clefiat128/192 PHOTON-256/32/32 SipHash-128 *No matching algo found!
5.	S6789	- Data confidentiality/User privacy - Non-repudiation	Encryption Digital signature	ChaCha20-256 *No matching algo found!
6.	H3456	- Data confidentiality/User privacy - Confidentiality & authenticity	Encryption Authenticated encryption	Grain_v1-128 ACORN
7.	S6000	- Message Integrity - Authentication	Hash function MAC	PHOTON-128/16/16 *No matching algo found!
8.	S4219	- Data confidentiality/User privacy - Message Integrity - Authentication - Confidentiality & authenticity	Encryption Hash function MAC Authenticated encryption	SPECK64/96 PHOTON-80/20/16 *No matching algo found! CLOC-TWINE
9.	H8791	- Message Integrity - Confidentiality & authenticity	Hash function Authenticated encryption	Keccak-f[100] Deoxys
10.	S4452	- Message Integrity - Non-repudiation - Confidentiality & authenticity	Hash function Digital signature Authenticated encryption	PHOTON-128/16/16 *No matching algo found! CLOC-AES
11.	H2648	- Data Confidentiality - Message Integrity - Authentication - Confidentiality & authenticity	Encryption Hash function MAC Authenticated encryption	*No matching algo found! *No matching algo found! *No matching algo found! Ascon
12.	S1682	- Data Confidentiality - Message Integrity - Authentication - Confidentiality & authenticity	Encryption Hash function MAC Authenticated encryption	SPECK64/128 PHOTON-128/16/16 *No matching algo found! CLOC-AES
13.	S6001	- Message Integrity	Hash function	PHOTON-80/20/16
14.	H9850	- Data Confidentiality - Message Integrity - Authentication - Confidentiality & authenticity	Encryption Hash function MAC Authenticated encryption	SIMON64/96 SPONGENT-128/128/8 *No matching algo found! SILC-AES
15.	H5942	- Message Integrity - Confidentiality & authenticity	Hash function Authenticated encryption	PHOTON-80/20/16 Deoxys
16.	S0001	- Data confidentiality/User privacy - Message Integrity	Encryption Hash function	Clefiat128/256 PHOTON-224/32/32
17.	S7788	- Data confidentiality/User privacy - Message Integrity	Encryption Hash function	SPECK64/128 PHOTON-128/16/16

* No matching algo found! Means: No algorithm matching the security requirement is found.



(a) LWCAR tool simplicity of use scores of individual subjects



(b) Percentage representation of the simplicity of use of the LWCAR tool

Figure 5.11: Simplicity and ease of use of the LWCAR tool.

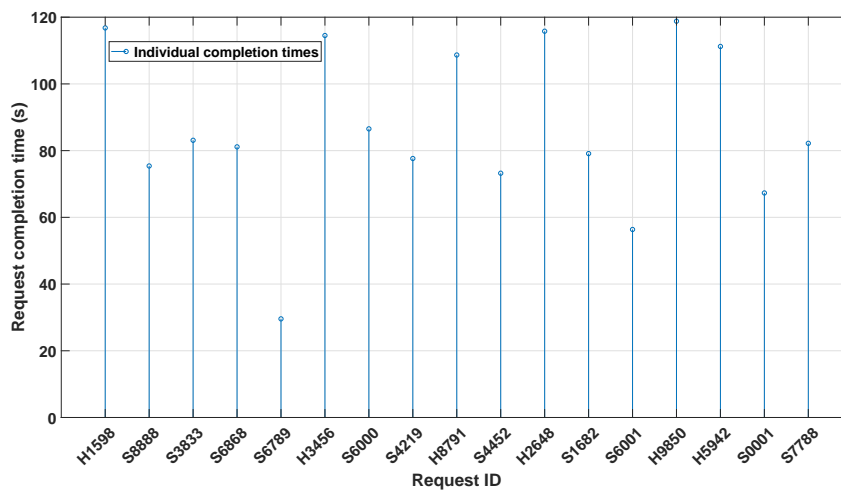


Figure 5.12: Request completion times of subjects for the LWCAR tool test.

This section presents the data in the performance evaluation form as completed by the 17

S/No.	Parameter/Presented Question	User Response
1.	Request ID	R2143
2.	Application Domain	Smart Home
3.	System development phase	Planning
4.	Will the system have a user?	Yes
5.	Will it have a user login?	Yes
6.	Will it store user information?	Yes
7.	Will it store any other information?	Yes
8.	Type of information	Sensitive
9.	Will it send data to an entity?	Yes
10.	Will it be on the Internet?	Yes
11.	Will it send data to a cloud?	Yes
12.	Will it send data to a database?	Yes
13.	Will it receive regular updates?	Yes
14.	Will it use third-party software?	Yes
15.	Is there possibility of eavesdropping?	Yes
16.	Can a message be replayed?	Yes
17.	Can a user be impersonated?	Yes
18.	Can someone access it physically?	Yes

Table 5.20: A summary of the SRE request for subject with request ID R2143.

S/No.	Parameter	User Response
1.	Request ID	S2143
2.	System development phase	Planning
3.	Enter hardware type	MCU
4.	Select MCU type	AVR
5.	Select CPU type	8-bit
6.	Enter flash memory size (in KB)	128
7.	Enter RAM size (in KB)	4
8.	Enter processor speed (in MHz)	16
9.	Select application domain	Smart Home
10.	Select payload size	small
11.	Import security requirements from SRE component tool?	Yes

Table 5.21: A summary of the LWCAR request for subject with request ID S2143.

Table 5.22: LWCAR software implementation requests data as stored in the database.

Request ID	Hardware type	CPU	FlashMem_size	Ram_size	Clock_speed	Applic_Domain	Payload_size	Req_1	Req_2	Req_3	Req_4	Req_5	Req_6
S0001	SBC	32	8000000	4096000	2000	Home	small	CONF	INTG		PRIV		
S1682	ARM	32	1024	96.00	72	City	small	CONF	INTG	AUTH			COAU
S3833	ELEC	32	1024	128	120	Health-care	average		INTG			NONR	COAU
S4219	RL78	16	64	4	16	Retail	small	CONF	INTG	AUTH	PRIV		COAU
S4452	TENS	32	16384	520	240	Grid	small		INTG			NONR	COAU
S6000	AVR	8	136	6	16	Connected_Car	small		INTG	AUTH			
S6001	AVR	8	128	4	16	Pet	small		INTG				
S6789	SBC	64	32000000	2048000	900	Home	Continuous	CONF			PRIV	NONR	
S6868	SBC	64	8388608	1048576	900	Agriculture	small	CONF	INTG	AUTH	PRIV	NONR	
S7788	PIC	32	256	1024	16	Home	small	CONF	INTG		PRIV		
S8888	SBC	64	8388608	1048576	1200	Financial	average	CONF	INTG	AUTH	PRIV	NONR	

subjects, as shown in Table 5.16. It also presents the request data of the subjects stored in the database as depicted in Tables 5.22-5.24. But the screenshots of the actual request data of the subjects as stored in the MySQL database table is presented in Figure B.3 in Appendix B. Table 5.17 provides more clarity on the designations of the 14 columns in Table 5.22. Similarly, the designations of unique columns in Tables 5.23 and 5.24 are clarified in Table 5.18. As in Subsections 5.2.2 and 5.3.2, the numerical values obtained from scaling the security expertise, proficiency level, and simplicity of use metrics are used to present the data graphically in Figures 5.10, 5.11, and 5.12, which are more intuitive and interpretable than the data in the evaluation form presented in Table 5.16. Figures 5.13 and 5.14 present the final results of the processed software and hardware requests, respectively, for only four subjects. Nonetheless, the summaries of the results for all the 17 subjects that participated in the first four tests are presented in Table 5.19. Note that because of space constraints it is only the basic results that are presented in Figures 5.13 and 5.14. The result of the fifth test scenario represents an example of a detailed result produced by the LWCAR tool.

 FINAL RESULTS FOR THE USER WITH REQUEST ID No.: S4219

YOUR SECURITY REQUIREMENTS AND RECOMMENDED SECURITY MECHANISMS AND SECURITY ALGORITHMS ARE:

SECURITY REQUIREMENT(S)	SECURITY MECHANISM(S)	SECURITY ALGORITHM(S)
Data Confidentiality/User Privacy	Encryption	SPECK64/96
Message Integrity	Hash Function	PHOTON-80/20/16
Authentication	Message Authentication Code	*No matching Algo found!
Confidentiality & Authenticity	Authenticated Encryption	CLOC-TWINE

*No algorithm matching the security requirement is found!

Would you like to see a detailed report on the recommended algorithms?:

1. Yes
2. No, thank you

Select Your Option (1-2): 2

WARNING! LIMITED RESOURCES
 Implementing all algorithms may have negative impact on performance.

Press Enter to return to MAIN MENU

(a) Final result of software request for subject with request ID S4219

 FINAL RESULTS FOR THE USER WITH REQUEST ID No.: S8888

YOUR SECURITY REQUIREMENTS AND RECOMMENDED SECURITY MECHANISMS AND SECURITY ALGORITHMS ARE:

SECURITY REQUIREMENT(S)	SECURITY MECHANISM(S)	SECURITY ALGORITHM(S)
Data Confidentiality/User Privacy	Encryption	ClefiA128/192
Message Integrity	Hash Function	PHOTON-256/32/32
Authentication	Message Authentication Code	SipHash-128
Non-repudiation	Digital Signature	*No matching Algo found!

*No algorithm matching the security requirement is found!

Would you like to see a detailed report on the recommended algorithms?:

1. Yes
2. No, thank you

Select Your Option (1-2): 2

Press Enter to return to MAIN MENU

(b) Final result of software request for subject with request ID S8888

Figure 5.13: Final software results for subjects with request IDs S4219 and S8888.

FINAL RESULTS FOR THE USER WITH REQUEST ID No.: H1598

YOUR SECURITY REQUIREMENTS AND RECOMMENDED SECURITY MECHANISMS AND SECURITY ALGORITHMS ARE:

SECURITY REQUIREMENT(S)	SECURITY MECHANISM(S)	SECURITY ALGORITHM(S)
Data Confidentiality	Encryption	*No matching Algo found!
Authentication	Message Authentication Code	*No matching Algo found!
Confidentiality & Authenticity	Authenticated Encryption	ACORN

*No algorithm matching the security requirement is found!

Would you like to see a detailed report on the recommended algorithms?:

1. Yes
2. No, thank you

Select Your Option (1-2): 2

Press Enter to return to MAIN MENU

(a) Final result of hardware request for subject with request ID H1598

FINAL RESULTS FOR THE USER WITH REQUEST ID No.: H3456

YOUR SECURITY REQUIREMENTS AND RECOMMENDED SECURITY MECHANISMS AND SECURITY ALGORITHMS ARE:

SECURITY REQUIREMENT(S)	SECURITY MECHANISM(S)	SECURITY ALGORITHM(S)
Data Confidentiality/User Privacy	Encryption	Grain_v1-128
Confidentiality & Authenticity	Authenticated Encryption	ACORN

Would you like to see a detailed report on the recommended algorithms?:

1. Yes
2. No, thank you

Select Your Option (1-2): 2

Press Enter to return to MAIN MENU

(b) Final result of hardware request for subject with request ID H3456

Figure 5.14: Final hardware results for subjects with request IDs H1598 and H3456.

Table 5.23: LWCAR hardware implementation requests data as stored in the database.

Request_ID	Hardware_type	Applic_Domain	Payload_size	Energy	Req_1	cct_area_1	tp_1	Req_2	cct_area_2	tp_2
H1598	FPGA	Home	small	Ultra-low	CONF	1022	100		0	0
H2648	FPGA	Health-care	small	Ultra-low	CONF	578	230	INTG	734	158
H3456	ASIC	Connected_Car	Continuous	Low	CONF	1450	200		0	0
H5942	ASIC	Agriculture	average	Low		0	0	INTG	757	1
H8791	ASIC	Home	small	Low		0	0	INTG	1254	2.3
H9850	ASIC	Home	small	Low	CONF	825	5.1	INTG	1398	16

Table 5.24: Continuation of LWCAR hardware implementation requests data as stored in the database.

Req_3	cct_area_3	tp_3	Req_4	cct_area_4	tp_4	Req_5	cct_area_5	tp_5	Req_6	cct_area_6	tp_6
AUTH	270	3.5		0	0		0	0	COAU	137	240
AUTH	356	280		0	0		0	0	COAU	415	310
	0	0	PRIV	1500	200		0	0	COAU	3150	4.3
	0	0		0	0		0	0	COAU	2879	4.9
	0	0		0	0		0	0	COAU	2863	4.9
AUTH	748	4.3		0	0		0	0	COAU	3200	5.8

In the case of the fifth test scenario, out of the six results for the six subjects that participated in the SRE tool test, the result of the subject with request ID R2143 was selected as highlighted in the first paragraph. The rationale for selecting this result is based on the response of the subject to the questionnaire. In particular, the response of this subject to all of the questions with the *Yes* or *No* options were *Yes* as shown in the summary of the request presented in Table 5.20. As a result, the SRE tool generated a long list of security requirements, which represents the maximum number of security requirements the SRE tool can currently generate, as shown in Figure D.1 in Appendix D.

In the same vein, using a similar request ID as in the previous case, but replacing the *R* with an *S* (i.e., S2143) as described in Subsection 4.4.1, the subject used the LWCAR tool to ascertain the LWCAs for software implementation of his IoT system. When prompted, the subject decided to import his security requirements that were generated by the SRE tool as mentioned in the previous paragraph (see Figure D.1 in Appendix D). The summary of the LWCAR tool request for the subject is presented in Table 5.21. The LWCAR tool can produce both a short version of results (see Figure 5.13 and Figure 5.14) as well as a detailed report on results. Note that a detailed report can be quite long, making it difficult to fit into a single page, especially if a user imports his/her security requirements from the SRE tool. For example, the detailed report on the result of the request of the subject with request ID S2143 is presented in Figure D.2 in Appendix D.

5.4.3 Evaluation and Discussion of the LWCAR Tool Test Results

This subsection presents the evaluation of the performance of the LWCAR tool. It focuses on the analysis and discussion of the results presented in Subsection 5.4.2 in light of the three performance metrics mentioned in Subsection 5.4.1.1.

5.4.3.1 Comparison of the Decision-Making Process of LWCAR Tool Versus Decision-Making Process of Human Experts

Different experts in the same field would usually approach a given problem in quite different ways and they often have different opinions for making a decision about the same issue. Therefore, modeling expert decision-making process can be a challenging task. For the purpose of this evaluation, the author attempts to develop an expert knowledge representation, in the form of a selection procedure, that reflects how experts use their IoT and LWC knowledge to select an appropriate LWCA for a given security requirement [259, 246]. This is achieved by exploring the reasoning underlying the decision-making process of human experts in the fields of LWC and IoT. This selection procedure is compared with the decision-making process of the LWCAR tool to see how IoT-HarPSecA measures up to the decision-making process of human experts [241, 176].

A concise procedure for selecting a lightweight security algorithm for IoT applications, which a typical human expert would follow is presented below:

1. Given an IoT device or application development project and a particular hardware platform, a human expert would start by considering the environment in which the IoT device or application will operate;
2. Consider the sensitivity of the application domain;
3. Determine the necessary security requirements needed to secure the device or application;
4. Consider the message payload size that the device or application will be sending/receiving;
5. Consider the hardware specifications on which the *candidate* algorithm will run: for software implementation, the hardware specifications include hardware type (e.g., MCU or SBC), CPU, RAM size, flash memory size, and CPU clock speed. For hardware implementation, the specifications include type of implementation platform (e.g., FPGA or ASIC), circuit area, throughput (or number of slices and maximum operating frequency), and energy requirement;
6. Consider the desired security level (32-bit, 64-bit, or 128-bit);
7. Consider the security level, key sizes (and block sizes in case of a block cipher) of potential algorithms;
8. Consider the energy requirements of potential algorithms, keeping in mind the power specifications of target hardware;

9. Integrate this information to make judgments that underlie the algorithm selection decisions.

The above procedure arguably mimics the intuitive judgment of a typical human expert in the fields of LWC and IoT. While the sequence of the steps may vary more or less, most of the steps in the above procedure are similar to those in the LWCAR tool decision-making process already described in Subsection 4.4.3. Unlike human experts, computers and computer programs are predictable and may not provide very detailed explanations and rational reasons for a particular decision because they are logical machines or systems that deal with zeros and ones (0s and 1s) rather than literal information. Nonetheless, based on user inputs, the LWCAR tool is capable of providing a detailed report that can contain some useful information that may help users during LWCAs implementations, such as brief descriptions of the algorithms, where to obtain a detailed implementation guide, and additional references for further study (see Figure D.2).

The LWCAR tool can also offer some suggestions about the choice of security requirements that may help users to optimize the implementation of the algorithms. For example, if the security requirements of a user include confidentiality/privacy and integrity, the tool will advise the user to consider returning to the *main menu* in order to modify his/her security requirements by including confidentiality & authenticity since the mechanism that provides confidentiality & authenticity is the authenticated encryption. This is because as discussed in Subsection 3.4.1.5, an authenticated encryption algorithm is capable of providing message integrity and message origin authentication in addition to protecting data confidentiality and/or user privacy (see the suggestion section in Figure D.2).

It also provides some warning messages that can guide the user when making a request and during implementation. For example, in Figure 5.13(a) and Figure D.2, a warning message was printed due to the hardware limitations and the number of user security requirements. The *No matching Algo found! warning message will also be printed if a user hardware capabilities (e.g., RAM, and/or flash memory) do not match the specifications of the lightest algorithm (in the database) corresponding to the user security requirement in question.

5.4.3.2 Decision Accuracy of the LWCAR Tool

Although the author of this thesis does not claim that the LWCAR tool accurately mimics the reasoning process of human experts, arguably the tool produces good and useful decisions about its inputs. This can be verified by comparing the user requests presented in Tables 5.22-5.24 and Table 5.21 with the corresponding final results in Figures 5.13, Figure 5.14, and Table 5.19, as well as the detailed report presented in Figure D.2.

Take for example the software implementation requests for the subject with request ID S4219 in Table 5.22 consisting of the following information: hardware - MCU (RL78),

CPU - 16-bit, flash memory size - 64kB, RAM size - 4kB, CPU clock speed - 16MHz, application domain - smart retail (sensitive), payload size - small, security requirements - data confidentiality, message integrity, authentication, user privacy, and confidentiality & authenticity. Based on the given security requirements, it can be seen from Figure 5.13(a) that the tool accurately provided the security mechanisms needed (i.e., encryption, hash function, MAC, and authenticated encryption). Similarly, based on the hardware specifications, the sensitivity of the application domain, and the message payload size, the tool recommended the appropriate LWCA, as discussed in Subsection 4.4.3. For example, for data confidentiality and user privacy, it recommended the lightweight block cipher SPECK (SPECK64/96) with a block size of 64 and the key length of 96, which takes less CPU time and occupies small space on the ROM. Similarly, for message integrity, it recommended PHOTON-80/20/16 from the PHOTON lightweight hash function family specially designed for very constrained devices. PHOTON-80/20/16 has a hash output size of 80 bits, and input and output bitrate of 20 and 16, respectively. In the case of authentication, however, the tool printed `*No matching algo found!` because there is no algorithm matching the security requirement. This will be discussed in more detail in the following paragraphs. Furthermore, for confidentiality & authenticity, the tool recommended the AEAD algorithm CLOC-TWINE, which is optimized for short inputs or small message payloads.

In the case of hardware implementation requests, the hardware platform for the subject with request ID H1598 whose result is shown in Figure 5.14(a) is FPGA, and his security requirements are data confidentiality, authentication, and data confidentiality & authenticity, as shown in Tables 5.23 and 5.24. Although the tool recommended a lightweight authenticated encryption algorithm (ACORN) for the third security requirement, it printed `*No matching algo found!` for the first two security requirements. The reasons will be explained in the following paragraphs. On the other hand, the hardware platform for the subject with request ID H3456 is ASIC, and his security requirements are data confidentiality, user privacy, and confidentiality & authenticity (see Tables 5.23 and 5.24). But in this case, the LWCA tool recommended LWCA for all of the security requirements, as shown in Figure 5.14(b), for the reasons that will be explained in the next paragraph.

Studies show that most research on hardware implementation performance of LWCA mainly target ASIC implementations, for which exhaustive reviews and evaluations have been conducted [176]. Therefore, in the lightweight block cipher, stream cipher, and hash function performance evaluations presented by CRYPTREC, only GEs, cycles/blocks (for block ciphers) as well as throughput are provided, and no number of slices and maximum operating frequency for FPGAs were provided. Nonetheless, a sufficient number of evaluations have been conducted in the case of lightweight authenticated encryption algorithms for the FPGAs. Therefore, CRYPTREC has provided the number of slices and maximum operating frequency along with the GEs and throughput for the lightweight authenticated

encryption algorithms they presented.

This is reflected in the design and implementation of the hardware request aspect of the LWCAR tool described in Section 4.4. Consequently, if the security requirements of a user with hardware implementation request using an FPGA platform includes data confidentiality (and/or user privacy), or integrity, the tool will print `No algorithm matching the security requirement is found!`. This can be seen in the case of the results of the requests of the subjects with the request IDs H1598 and H2648 in Figure 5.14(a) and in Table 5.19, respectively.

Now, regarding the algorithm for authentication, aside from the fact that SipHash is the only lightweight MAC algorithm that has undergone a sufficient number of evaluations as mentioned in Subsection 3.4.3, the algorithm has quite a limited use in the implementation of the LWCAR tool. The reason being that the MAC algorithm is optimized only for software implementation, explaining why the tool printed `*No matching Algo found!` in the final results for the subject with request ID H1598 as shown in Figure 5.14(a). In addition, the SipHash algorithm works well only on 64-bit processors, which explains why the tool did not recommend the algorithm in the final results for the subject with request ID S4219 as shown in Figure 5.13(a) since the CPU of the MCU is 16-bit (see Table 5.22). However, looking at the final results for the subjects with request IDs S6868 (see Table 5.19) and S8888 (see Figure 5.13(b)), it can be seen that the same algorithm has been recommended since in both cases the hardware use 64-bit processors, as shown in Table 5.22.

In spite of the limitations mentioned above, the subject with request ID S4219 has the option to implement the recommended AE algorithm, namely CLOC-TWINE, which can provide data confidentiality and authenticity. Similarly, the first two security requirements (i.e., data confidentiality and authentication) in the request of the subject with request ID H1598 can be achieved by implementing the recommended AE algorithm (i.e., ACORN), which is capable of providing both confidentiality and authenticity.

In an analogous manner to the case of the SipHash algorithm, the author has not yet included any lightweight digital signature algorithm in the database, which currently contains 97 different secure LWCAs. This is because to the best of the knowledge of the author, there is no standardized or widely used lightweight digital signature algorithm that has undergone a sufficient number of reviews to warrant inclusion in the database. Therefore, if a user security requirements include non-repudiation, the tool will print: `No algorithm matching the security requirement is found!`, as shown in Figure 5.13(b) and in Table 5.19.

5.4.3.3 Simplicity and Ease of Use of LWCAR Tool

This evaluation covers only the first four tests performed by the 17 subjects as highlighted in Subsection 5.4.2. To this effect, the data for this evaluation is obtained from Table 5.16 and Figures 5.10, 5.11, and 5.12.

As in the case of the SRE and SBPG tools, the author claims that the LWCAR tool is easy to use. This can be justified by the provision of a number of useful notifications that can guide users as they interact with the tool to ease task execution. For example, when making a hardware request, users are provided with the range of values of GEs, throughput, number of slices, and maximum operating frequencies for the available algorithms to prevent them from entering values outside the allowable range defined by the designers of the algorithms.

As depicted in Tables 5.1 and 5.8, Table 5.16 shows the two criteria used for evaluating the simplicity and ease of use of the LWCAR tool, namely simplicity of use and request completion time. Figure 5.10 shows the professional competence and security experience of the subjects. The figure shows that 8 subjects have a little security experience, while 3 subjects have no security experience at all (i.e., 2.5 and 0.0, respectively, on the numeric rating scale). Yet in Figure 5.11(b), 76% of the 17 subjects agree that the tool is easy to use, 6% indicates that it is very easy, and 18% say it is average. Figure 5.11(a) shows the simplicity scores of individual subjects. Finally, Figure 5.12 shows the request completion time for each subject. From Figure 5.12, it can be seen that the time taken to make a hardware request is longer. This is due to the fact that for each security requirement, a user needs to enter the number of GEs and throughput for ASICs, or the number of slices and maximum operating frequency for FPGAs, but such information is not needed when making a software implementation request.

5.5 Conclusion

In light of the foregoing discussion, the IoT-HarPSecA framework can arguably be said to be capable of assisting non-experts in the security field to design and develop more secure IoT systems. The components of the framework tool, which are mostly text-based, have been designed to accurately capture the intent or requirements of users in the context of their functionalities. For example, the questionnaires devised for the SRE and SBPG component tools have been designed to better capture, from the request of a user, the essential components needed to generate appropriate security requirements and security best practice guidelines, respectively. Similarly, the LWCAR component tool uses user security requirements, application domain, and other specifications from a user input to recommend security mechanisms and the appropriate lightweight cryptographic algorithms that can provide them. The results show that these tools can help IoT system architects and developers with little or no security expertise to develop more secure IoT

systems. The tools can particularly facilitate the process of embedding security in the IoT product development lifecycle.

This chapter described a number of real-world-like test scenarios used in the evaluation of the performance and usability of the three components of the IoT-HarPSecA framework. The tests were actually performed on the SRE, SBPG, and LWCAR tools of the IoT-HarPSecA framework. The tests on the first two tools were performed within the same period of time by 24 subjects consisting of 13 computer engineers, 6 developers, and 5 electronic engineers. There was only one test scenario for each of the two tools. In a similar manner, a series of tests were conducted on the LWCAR tool. The first four tests were performed by 17 subjects consisting of 11 developers, 3 computer engineers, and 3 electronics engineers. Furthermore, there were only 6 subjects that participated in the fifth test conducted on the LWCAR tool.

Because of space constraints, only three results of the SRE tool test and three result summaries of the SBPG tool test were presented in this chapter. Similarly, in the case of the LWCAR tool tests, only four results (i.e., two for software implementation requests and two for hardware implementation requests) were presented from the first four tests performed by the 17 subjects due to space constraints. But the whole results of the 17 subjects were summarized in a table. Regarding the fifth test conducted on the LWCAR tool, only one result was presented for the same reason. Finally, this chapter has discussed the evaluations of all of these results. The next chapter presents the main conclusions of this thesis and outlines some future research directions.

Chapter 6

Conclusions and Future Work

This chapter presents the main conclusions of the research work described in this thesis. It begins by summarizing the main contributions highlighted in Section 1.5, and then evolves to the assessment of the outcome, or measuring the achievement of the research objectives defined in Subsection 1.2.1 to ascertain whether or not the research objectives are met. It also determines whether or not the research work described in this thesis is consistent with the thesis statement presented in Section 1.3, after which it presents the final conclusions. Finally, it wraps up the thesis with a few research limitations and also provides directions for further research on the subject.

6.1 Summary of Main Scientific Contributions

The scientific contributions resulting from this work are recapitulated below in accordance with their appearance in Chapters 2, 3, 4, and 5 of this Ph.D. thesis. Chapter 2 provided a general survey on the current state of IoT security and privacy. The chapter is based on a previously-published survey paper [45], a conference paper [47], and four book chapters [52, 48, 46, 49]. The contributions in this chapter include a detailed and comprehensive survey of the state-of-the-art in IoT security and privacy, including a detailed description of different aspects of IoT security and privacy and other aspects of IoT in general; description of several application domains and identification of cyber assets per domain; description of threat and system models per domain; identification of several security threats in IoT architecture as well as in different IoT industry domains, and the provision of various possible countermeasures to address the identified security threats; and a brief overview of IoT hardware development platforms.

Chapter 3, which is based on articles 1 [45], 2 [43], and 10 enumerated in Subsection 1.5.1, discussed the key concepts and principles underlying the design and implementation of the three components of the IoT-HarPSecA framework. A major contribution in this chapter is the identification of security requirements in the nine IoT application domains described in Chapter 2. Other important contributions include an extensive study of IoT security and privacy best practices, including an in-depth study of challenges of developing a commonly agreed security and privacy best practices for IoT, as well as discussion on various attempts to develop more widely accepted security and privacy best practices for the IoT; exploration of some important aspects of LWC; and the presentation of software/hardware LWC design and implementation considerations.

Chapter 4 delves into the design and implementation of the SRE, SBPG and LWCAR components of the IoT-HarPSecA framework. Essentially, the description and design of these three components as well as the implementation of their respective tools are the major scientific contributions in this chapter. Articles 2 [43] and 4 [50] mentioned in Subsection 1.5.1 formed the basis for this chapter.

Chapter 5 focused on the performance evaluation of the three IoT-HarPSecA framework component tools. The chapter is based on articles 2 [43], 4 [50], 9 [51] and 10 enumerated in Subsection 1.5.1. The main scientific contributions in this chapter are the development of a human expert knowledge representation in the form of decision-making procedures of experts in the fields of IoT and LWC, and a detailed evaluation of the performance and usability of the SRE, SBPG and LWCAR tools of the IoT-HarPSecA framework.

6.2 Assessing the Achievement of Research Objectives and Validating Thesis Statement

The objectives underlying the research work described in this thesis have been presented at the beginning of this thesis in Subsection 1.2.1 and it is now claimed that those research objectives have been successfully achieved. To substantiate the claim that those research objectives are successfully accomplished within the scope of this Ph.D. research work, especially through the design and development of the IoT-HarPSecA framework, the research objectives are recapitulated below and analyzed based on the work presented in this thesis in order to ascertain whether or not they are successfully achieved.

The two initial research objectives that formed the basis of this research work are the following:

1. The first research objective was to explore security and privacy issues in the IoT by providing some insights into the research being done in this area, including current challenges and requirements, as well as to highlight the need to bake security and privacy into the design and development of IoT devices and smart apps from the very beginning;
2. In line with the concept of security-by-design, the second objective of this work was to propose and implement a prototype of a security advisor which might function as a security framework intended to facilitate the design and development of secure IoT devices and smart apps, as well as to evaluate the performance and usability of the security framework.

Starting with the first objective, the extensive survey of different aspects of IoT security and privacy presented in Chapter 2 (for example, see Subsections 2.3.3, 2.3.4, 2.3.5 and

2.3.6) and the discussion of important concepts of security and privacy provided in Chapter 3 of this thesis provide evidence that the first research objective has been successfully accomplished.

To validate the accomplishment of the second research objective, there is a need to divide the research objective into two parts. While the first part of the objective concerns the description of the design and implementation of the framework later known as IoT-HarPSecA, the second part has to do with the intended functionality of the security framework as well as its usability. Regarding the first part of the objective, considering the topics covered in Chapter 4, particularly in Sections 4.2, 4.3, and 4.4, it can be seen that the design and description of the three components of the IoT-HarPSecA framework, as well as the implementation of the three framework tools (see Section 4.5), have been extensively discussed, and thus this aspect of the research objective can be said to have been successfully achieved.

Validating the accomplishment of the second part of the second research objective would normally require restating the intended functionalities of the SRE, SBPG, and LWCAR components of the IoT-HarPSecA framework mentioned in Sections 4.2, 4.3, and 4.4, respectively. Additionally, it would require referring to the results of the various tests presented in Chapter 5 of this thesis. However, in order to avoid unnecessary repetition and to put it concisely, this aspect is summarized in a table that swiftly presents the achievement of the second part of the objective by using one request and its corresponding result from the functionality and usability tests and results for each of the three IoT-HarPSecA framework tools presented in Chapter 5.

Table 6.1 maps the request summaries of the selected subjects to their corresponding result summaries which were consolidated to produce a table that can serve as a useful tool for validating the second part of the research objective. For quick reference, lines are used in the table rows of the last two columns to ease the mapping and matching of individual requests to their corresponding results. The five columns of Table 6.1 consist of the name of an IoT-HarPSecA framework component, summary of component functionality, subject request ID, request summaries, and summary of results. Note that in order to keep the table as concise as possible, only the questions and corresponding user responses that resulted in a particular result, in the case of the SRE and SBPG component tools, are featured in Table 6.1. To see the full request summaries, refer to Tables 5.5 and 5.13, respectively.

Arguably, the successful design and development of the IoT-HarPSecA framework, the demonstration of the functionalities of the SRE, SBPG, and LWCAR tools of the framework, as well as the evaluation of the results of the performance and usability tests conducted on the framework tool presented in this thesis strongly corroborate the thesis state-

ment put forward in Section 1.3. Although the security framework developed within the scope of this study is not a cybersecurity silver bullet that can deal with every IoT security issue, the author of this thesis claims that IoT-HarPSecA framework can help users with little or no expertise to develop secure IoT systems.

6.3 Final Conclusions

In this era of digital transformation where everything is turning smart, security and privacy are becoming more widely recognized as being among the key factors affecting consumer trust in the IoT, and thus holding back wider adoption of the IoT technology. The security and privacy risks that come with the deployment of IoT devices and smart apps are high that they cause for immediate concern. Consequently, there is a need for heightened efforts to ensure security and privacy in the IoT ecosystem. However, ensuring appropriate security and privacy in the IoT presents unique challenges due to the constraints associated with many IoT devices. Therefore, the wider adoption of IoT requires effective and robust security and privacy solutions to handle these challenges. Integrating security right from the conception, planning and design stages up to production is key to overcoming most of these challenges.

To this end, this thesis has presented an effort towards the research and development of an efficient and easy-to-use security framework that can provide support to designers, engineers, and developers that are actively involved in the design and development of IoT devices and smart apps, particularly those without adequate security experience. For example, the IoT-HarPSecA framework can be employed by designers, engineers, or developers in IoT start-up companies in the early stages of IoT system development. The security framework can also be used to facilitate the implementation of security in existing IoT systems.

This thesis has presented a detailed description, design, and implementation of each of the three components of the IoT-HarPSecA framework, namely the SRE component, SBPG component, and LWCAR component. The SRE component can be used to elicit security requirements for an IoT system. Similarly, the SBPG component can be employed to generate security best practice guidelines that can be used for implementing security in a given IoT system. In the same vein, the LWCAR component can facilitate the selection of suitable LWCAs for software or hardware implementation in IoT systems. In addition, this thesis has presented a number of functionality and usability tests conducted on the three IoT-HarPSecA framework tools, where 24 and 17 different subjects conducted some tests for evaluating the performance and simplicity of use of the SRE/SBPG and LWCAR tools, respectively, in different usage scenarios (not forgetting the 6 subjects that conducted a test on the LWCAR tool). Finally, the test results and the evaluation that followed proved that the IoT-HarPSecA framework can efficiently facilitate the design and

development of secure IoT systems.

6.4 Research Limitations and Future Work

Although the research work described in this doctoral thesis has been successfully carried out with all research objectives fully accomplished, a number of challenges and research limitations, some of which could lay a firm foundation for future work, have been identified. There are two notable non-technical challenges the author of this thesis has encountered during the testing process of the IoT-HarPsecA framework tools. The first challenge was the shortage of highly skilled volunteer testers. The second challenge, which is related to the previous one was behavioral in nature, where some highly skilled experts did not want to participate in the tests; and lack of free time was the most common reason they gave for not volunteering to participate.

The aforementioned challenges, which have impacted the number of subjects that carried out the various tests reported in Chapter 5, are definitely due to the fact that there are no IoT companies or related start-up companies in Covilhã and its surrounding areas, where the author carried out the research work. Nonetheless, the author was able to visit a company that is into IoT-related services during the course of this work and the invaluable feedback and insightful suggestions of the staff of that company, especially the security team who tested and evaluated the three tools, have greatly helped the author to improve the user experience of the IoT-HarPsecA framework tool.

There are few limitations of this work that are of a technical nature which the author has observed. Note, however, that these limitations do not invalidate the results presented in Chapter 5, nor do they invalidate the functionality of the IoT-HarPsecA framework. These limitations are, for the most part, related to the LWCAR component of the IoT-HarPsecA framework tool, mainly because there are very few or no well-vetted LWCAAs for certain security mechanisms. Most of the following limitations are imposed by the fact that the author did not want to include algorithms that have not undergone enough evaluation by the lightweight cryptography community:

- Currently, there are no lightweight digital signature algorithms in the IoT-HarPsecA framework database, as highlighted in the last paragraph of Subsection 5.4.3.2, which results in printing `No algorithm matching the security requirement is found!` if the security requirements of a user include non-repudiation;
- Similarly, at present there is only one well-vetted lightweight MAC algorithm in the IoT-HarPsecA framework database (i.e., *SipHash*) as mentioned in Subsection 3.4.3 and Subsection 5.4.3.2. Hence, due to the limitations of this algorithm (already discussed in the sixth paragraph of Subsection 5.4.3.2), oftentimes user requests with

authentication security requirement returns `No algorithm matching the security requirement is found!` as in the previous case;

- In the same vein, for hardware implementation requests using the FPGA hardware platform, the LWCAR tool prints `No algorithm matching the security requirement is found!` if user requests include data confidentiality (and/or user privacy) and integrity for the reason explained in the fourth paragraph of Subsection 5.4.3.2;
- While the SRE tool of the IoT-HarPSecA framework can elicit up to 15 different security requirements, there are still a number of security requirements that are being considered and hence yet to be included in the database, including robust and resilient management, secure routing, DDoS protection, as well as insider attack detection.

The aforementioned research limitations could serve as a basis for future work. For example, the first three research limitations reveal that further research is needed in the area of lightweight cryptography, especially in the development of more secure, more energy-efficient, and implementable LWCAs for digital signature and MAC. In addition, more research work is needed to evaluate the implementation of various LWCAs on the FPGA hardware platforms, as well as the development of new LWCAs tailored for implementation on the FPGA hardware platforms.

Future work may be underway to improve and extend the features of the IoT-HarPSecA framework tool as well as to include as many IoT security requirements and IoT security best practice guidelines as possible, such as the three security requirements mentioned in the fourth research limitation outlined above. Additionally, while the individual tools are currently somewhat integrated (e.g., the output of the SRE tool can be fed as input to the LWCAR tool), there is still work to be done in the integration of these tools to produce the best-combined output as possible. Incorporation of artificial intelligence techniques into the IoT-HarPSecA framework tool is also being considered, with the view to making the tools better at understanding requests and arriving at conclusions with greater accuracy. Moreover, the author is planning to develop a GUI for the IoT-HarPSecA framework tool, which could deliver a better user experience.

Furthermore, future iterations of the tool might be web-based with a means to receive inputs from the community with the view to improving its performance over time. In addition, the framework and components developed thus far will be part of a larger set of tools that will be developed within the scope of the **SECUR IOT ESIGN** project. Accordingly, future work is underway to develop and integrate other tools, including a tool for automatic attack modeling and a tool for semi-automatic generation of software tests. These tools are expected to provide a solid basis for an even more complete framework in the future.

Table 6.1: Validation of second part of the second research objective.

IoT-HarPSec Component	Summary of Functionality	Subject Request ID	Summary of Request and Corresponding Response	Summary of Results
SRE	Elicitation of security requirements	R1995	Will the system have a user? -Yes; Will it have a user login? -Yes	Authentication
			Will it send data to an entity? - Yes; Is there possibility of eavesdropping? - Yes	Confidentiality, Integrity
			Will it store any other information? - Yes; What type of information will it store? - Normal; Will it be on the Internet? - Yes	Availability
			Will it send data to an entity? - Yes; Will it be on the Internet? - Yes	Confinement
			Can someone with bad intentions gain physical access to the system? - Yes	Physical security
SBPG	Generation of security best practice guidelines	B5555	Will it store any kind of information? - Yes; What type of information will it store? - Normal	Ensure secure boot; use tamper-resistant hardware-based storage like TPM; use good password management techniques; ensure physical security for devices that will be deployed open environments; etc.
			Will it generate log files? - Yes	Ensure that any newline characters in system log files are appropriately handled to prevent log forging; ensure that HTML characters are appropriately encoded to prevent XSS when viewing logs.
LWCAR	Facilitates the selection of LWCAs	S4219	Security Requirement Data Confidentiality/User Privacy	Mechanism - Algorithm Encryption - SPECK64/96
			Message Integrity	Hash Function - PHOTON-80/20/16
			Authentication	MAC - *No matching Algo found!
			Confidentiality & Authenticity	Authenticated Encryption - CLOC-TWINE

Bibliography

- [1] N. Kumar and D. P. Vidyarthi, "A Green Routing Algorithm for IoT-Enabled Software Defined Wireless Sensor Network," *IEEE Sensors Journal*, vol. 18, no. 22, pp. 9449–9460, Nov 2018. xiii, 1
- [2] M. Mercaldi, A. D’Oria, D. Murru, Hai-Ning Liang, K. L. Man, Eng Gee Lim, V. Hahnov, and M. Alexander, "Internet of Things: A practical Implementation Based on a Wireless Sensor Network Approach," in *Proceedings of the IEEE East-West Design Test Symposium (EWDTS 2013)*, Sep. 2013, pp. 1–4. xiii, 1
- [3] J. Patel, P. Trivedi, and D. Patel, "A Performance Analysis of "Light Fidelity" and "Internet of Things" It’s Application," in *Proceedings of the IEEE International Conference on Transforming Engineering Education (ICTEE)*, Dec 2017, pp. 1–4. xiii, 1
- [4] A. H. Ngu, M. Gutierrez, V. Metsis, S. Nepal, and Q. Z. Sheng, "IoT Middleware: A Survey on Issues and Enabling Technologies," *IEEE Internet of Things Journal*, vol. 4, no. 1, pp. 1–20, Feb 2017. xiii, 1
- [5] B. Cyr, J. Mahmood, and U. Guin, "Low-Cost and Secure Firmware Obfuscation Method for Protecting Electronic Systems From Cloning," *IEEE Internet of Things Journal*, vol. 6, no. 2, pp. 3700–3711, April 2019. xiii, 1
- [6] S. Lokuhitige and S. Brown, "Forecasting Maturity of IoT Technologies in Top 5 Countries Using Bibliometrics and Patent Analysis," in *Proceedings of the IEEE International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC)*, Oct 2017, pp. 338–341. xiii, 1
- [7] S. Oh and Y. Kim, "Development of IoT Security Component for Interoperability," in *Proceedings of the 13th International Computer Engineering Conference (ICENCO)*, Dec 2017, pp. 41–44. xiv, 1
- [8] M. Ramadan, "Industry 4.0: Development of Smart Sunroof Ambient Light Manufacturing System for Automotive Industry," in *Proceedings of the Advances in Science and Engineering Technology International Conferences (ASET)*, March 2019, pp. 1–5. xiv, 2
- [9] H. Truong, "Integrated Analytics for IIoT Predictive Maintenance Using IoT Big Data Cloud Systems," in *Proceedings of the IEEE International Conference on Industrial Internet (ICII)*, Oct 2018, pp. 109–118. xiv, 2
- [10] S. S. Arumugam, R. Badrinath, A. H. Herranz, J. Höller, C. R. B. Azevedo, B. Xiao, and V. Tudor, "Accelerating Industrial IoT Application Deployment through Reusable AI Components," in *Proceedings of the Global Internet of Things Summit (GIIoTS)*, June 2019, pp. 1–4. xiv, 2

- [11] C. Petrov. (2019) Internet of Things Statistics 2019 [The Rise Of IoT]. [Online]. Available: <https://techjury.net/stats-about/internet-of-things-statistics/> xiv, 2
- [12] Y. Kawamoto, H. Nishiyama, N. Kato, N. Yoshimura, and S. Yamamoto, "Internet of Things (IoT): Present State and Future Prospects," *IEICE Transactions*, vol. 97-D, pp. 2568–2575, 2014. xiv, 2
- [13] D. Miorandi, S. Sicari, F. D. Pellegrini, and I. Chlamtac, "Internet of Things: Vision, Applications and Research Challenges," *Ad Hoc Networks*, vol. 10, no. 7, pp. 1497 – 1516, 2012. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1570870512000674> xiv, 2
- [14] P. Annamalai, J. Bapat, and D. Das, "Emerging Access Technologies and Open Challenges in 5G IoT: From Physical Layer Perspective," in *Proceedings of the IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, Dec 2018, pp. 1–6. xiv, 2
- [15] N. Wang, P. Wang, A. Alipour-Fanid, L. Jiao, and K. Zeng, "Physical Layer Security of 5G Wireless Networks for IoT: Challenges and Opportunities," *IEEE Internet of Things Journal*, pp. 1–1, 2019. xiv, 2
- [16] K. Routh and T. Pal, "A Survey on Technological, Business and Societal Aspects of Internet of Things by Q3, 2017," in *Proceedings of the 3rd International Conference On Internet of Things: Smart Innovation and Usages (IoT-SIU)*, Feb 2018, pp. 1–4. xiv, 2, 42
- [17] W. Wang, S. He, L. Sun, T. Jiang, and Q. Zhang, "Cross-Technology Communications for Heterogeneous IoT Devices Through Artificial Doppler Shifts," *IEEE Transactions on Wireless Communications*, vol. 18, no. 2, pp. 796–806, Feb 2019. xv, 3
- [18] V. R. Konduru and M. R. Bharamagoudra, "Challenges and Solutions of Interoperability on IoT: How Far have we Come in Resolving the IoT Interoperability Issues," in *Proceedings of the IEEE International Conference On Smart Technologies For Smart Nation (SmartTechCon)*, Aug 2017, pp. 572–576. xv, 3
- [19] M. Gharbieh, H. ElSawy, A. Bader, and M. Alouini, "Spatiotemporal Stochastic Modeling of IoT Enabled Cellular Networks: Scalability and Stability Analysis," *IEEE Transactions on Communications*, vol. 65, no. 8, pp. 3585–3600, Aug 2017. xv, 3
- [20] J. Spehar, A. Fuks, M. Vauclair, M. Meijer, J. van Beek, and B. Shao, "Power Challenges Caused by IOT Edge Nodes: Securing and Sensing Our World," in *Proceedings of the 2019 31st International Symposium on Power Semiconductor Devices and ICs (ISPSD)*, May 2019, pp. 17–22. xv, 3

- [21] M. Frustaci, P. Pace, G. Aloï, and G. Fortino, “Evaluating Critical Security Issues of the IoT World: Present and Future Challenges,” *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2483–2495, Aug 2018. xvi, 3, 41
- [22] D. Moongilan, “5G Internet of Things (IOT) Near and Far-Fields and Regulatory Compliance Intricacies,” in *Proceedings of the 2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, April 2019, pp. 894–898. xvi, 3
- [23] F. B. Saghezchi, G. Mantas, J. Ribeiro, M. Al-Rawi, S. Mumtaz, and J. Rodriguez, “Towards a Secure Network Architecture for Smart Grids in 5G Era,” in *Proceedings of the 13th IEEE International Wireless Communications and Mobile Computing Conference (IWCMC)*, June 2017, pp. 121–126. xvi, 3
- [24] C. Li and B. Palanisamy, “Privacy in Internet of Things: From Principles to Technologies,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 488–505, Feb 2019. xvi, 4
- [25] S. Bhatt, F. Patwa, and R. Sandhu, “An Access Control Framework for Cloud-Enabled Wearable Internet of Things,” in *Proceedings of the 2017 IEEE 3rd International Conference on Collaboration and Internet Computing (CIC)*, Oct 2017, pp. 328–338. xvi, 4
- [26] R. Vishwakarma and A. K. Jain, “A Honeypot with Machine Learning based Detection Framework for Defending IoT based Botnet DDoS Attacks,” in *Proceedings of the IEEE International Symposium on 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, April 2019, pp. 1019–1024. xvii, 4, 34, 35, 36, 44, 47
- [27] F. Meneghello, M. Calore, D. Zucchetto, M. Polese, and A. Zanella, “IoT: Internet of Threats? A Survey of Practical Security Vulnerabilities in Real IoT Devices,” *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8182–8201, Oct 2019. xvii, 4, 37
- [28] T. Mahjabin, Y. Xiao, T. Li, and C. L. P. Chen, “Load Distributed and Benign-Bot Mitigation Methods for IoT DNS Flood Attacks,” *IEEE Internet of Things Journal*, pp. 1–1, 2019. xvii, 5, 39
- [29] V. Hassija, V. Chamola, V. Saxena, D. Jain, P. Goyal, and B. Sikdar, “A Survey on IoT Security: Application Areas, Security Threats, and Solution Architectures,” *IEEE Access*, vol. 7, pp. 82 721–82 743, 2019. xvii, 5
- [30] F. Assaderaghi, G. Chindalore, B. Ibrahim, H. de Jong, M. Joye, S. Nassar, W. Steinbauer, M. Wagner, and T. Wille, “Privacy and Security: Key Requirements for Sustainable IoT Growth,” in *Proceedings of the 2017 IEEE Symposium on VLSI Technology*, June 2017, pp. T8–T13. xvii, 5

- [31] S. Shapsough, F. Aloul, and I. A. Zualkernan, “Securing Low-Resource Edge Devices for IoT Systems,” in *Proceedings of the 2018 International Symposium in Sensing and Instrumentation in IoT Era (ISSI)*, Sep. 2018, pp. 1–4. xvii, 5, 35
- [32] J. O’Raw, D. Lavery, and D. J. Morrow, “Securing the Industrial Internet of Things for Critical Infrastructure (IIoT-CI),” in *Proceedings of the 2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, April 2019, pp. 70–75. xviii, 5
- [33] Y. I. Khan and M. U. Ndubuaku, “Ontology-Based Automation of Security Guidelines for Smart Homes,” in *Proceedings of the IEEE 4th World Forum on Internet of Things (WF-IoT)*, Feb 2018, pp. 35–40. xviii, xx, 5, 7, 77
- [34] J. B. Martinez, “Medical Device Security in the IoT Age,” in *Proceedings of the 2018 9th IEEE Annual Ubiquitous Computing, Electronics Mobile Communication Conference (UEMCON)*, Nov 2018, pp. 128–134. xviii, 5
- [35] A. Pillai, M. Sindhu, and K. V. Lakshmy, “Securing Firmware in Internet of Things using Blockchain,” in *Proceedings of the 2019 5th International Conference on Advanced Computing Communication Systems (ICACCS)*, March 2019, pp. 329–334. xviii, 5
- [36] S. Oh and Y. Kim, “Security Requirements Analysis for the IoT,” in *Proceedings of the 2017 International Conference on Platform Technology and Service (PlatCon)*, Feb 2017, pp. 1–6. xviii, 6
- [37] M. Miettinen, S. Marchal, I. Hafeez, T. Frassetto, N. Asokan, A. Sadeghi, and S. Tarkoma, “IoT Sentinel Demo: Automated Device-Type Identification for Security Enforcement in IoT,” in *Proceedings of the 2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, June 2017, pp. 2511–2514. xviii, 6
- [38] I. Brass, L. Tanczer, M. Carr, M. Elsdén, and J. Blackstock, “Standardising a Moving Target: The Development and Evolution of IoT Security Standards,” in *Proceedings of the 2018 Living in the Internet of Things: Cybersecurity of the IoT Conference*, March 2018, pp. 1–9. xviii, 6, 39, 58, 70
- [39] N. Neshenko, M. Husák, E. Bou-Harb, P. Čeleda, S. Al-Mulla, and C. Fachkha, “Data-Driven Intelligence for Characterizing Internet-Scale IoT Exploitations,” in *Proceedings of the 2018 IEEE Globecom Workshops (GC Wkshps)*, Dec 2018, pp. 1–7. xix, 6
- [40] J. Lindley, P. Coulton, and R. Cooper, “Informed by Design,” in *Proceedings of the Living in the Internet of Things: Cybersecurity of the IoT - 2018 Conference*, 2018, pp. 1–12. xix, 6
- [41] J. M. Blythe and S. D. Johnson, “The Consumer Security Index for IoT: A Protocol for Developing an Index to Improve Consumer Decision Making and to Incentivize

- Greater Security Provision in IoT Devices,” in *Proceedings of the Living in the Internet of Things: Cybersecurity of the IoT - 2018 Conference*, 2018, pp. 1–7. xix, 6
- [42] S. Rizvi, A. Kurtz, J. Pfeffer, and M. Rizvi, “Securing the Internet of Things (IoT): A Security Taxonomy for IoT,” in *Proceedings of the 2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (Trust-Com/BigDataSE)*, 2018, pp. 163–168. xix, 7
- [43] M. G. Samaila, J. B. F. Sequeiros, T. Simões, M. M. Freire, and P. R. M. Inácio, “IoT-HarPSecA: A Framework and Roadmap for Secure Design and Development of Devices and Applications in the IoT Space,” *IEEE Access*, vol. 8, pp. 16 462–16 494, 2020. xx, xxii, xxiii, xxv, xxvi, xxxix, 7, 10, 11, 57, 87, 95, 96, 97, 101, 103, 105, 106, 115, 116, 157, 158
- [44] C. A. de Souza, J. N. Correa, M. M. Oliveira, A. Aagaard, and M. Presser, “IoT Driven Business Model Innovation and Sustainability: a literature review and a case Study in Brazil,” in *Proceedings of the 2019 Global IoT Summit (GIoTS)*, 2019, pp. 1–6. xx, 7
- [45] M. G. Samaila, M. Neto, D. A. B. Fernandes, M. M. Freire, and P. R. M. Inácio, “Challenges of Securing Internet of Things Devices: A Survey,” *Security and Privacy*, vol. 1, no. 2, p. e20, 2018. [Online]. Available: <https://onlinelibrary.wiley.com/doi/abs/10.1002/spy2.20> xxi, xxii, xxiii, xxv, xxxix, 9, 10, 15, 16, 22, 24, 28, 57, 157
- [46] M. G. Samaila, J. B. F. Sequeiros, A. F. P. P. Correia, M. M. Freire, and P. R. M. Inacio, *A Quick Perspective on the Current State of IoT Security: A Survey*. CRC Press, 2017, ch. 21, pp. 431–464. xxii, xxiv, xxv, 9, 11, 15, 157
- [47] M. G. Samaila, J. B. F. Sequeiros, M. M. Freire, and P. R. M. Inácio, “Security Threats and Possible Countermeasures in IoT Applications Covering Different Industry Domains,” in *Proceedings of the 13th International Conference on Availability, Reliability and Security*, ser. ARES 2018. New York, NY, USA: ACM, 2018, pp. 16:1–16:9. [Online]. Available: <http://doi.acm.org/10.1145/3230833.3232800> xxii, xxiii, xxv, xxxix, 9, 11, 15, 20, 157
- [48] M. G. Samaila, J. B. F. Sequeiros, A. F. P. P. Correia, M. M. Freire, and P. R. M. Inácio, “IoT Hardware Development Platforms: Past, Present, and Future,” in *Internet of Things: Challenges, Advances, and Applications*, Q. F. Hassan, A. ur Rehman Khan, and S. A. Madani, Eds. CRC Press, 2018, pp. 107–139. xxii, xxiv, xxv, 10, 11, 15, 157
- [49] M. Meruje, M. G. Samaila, V. N. L., M. M. Freire, and P. R. M. Inácio, “A Tutorial Introduction to IoT Design and Prototyping with Examples,” in *Internet of things*

A to Z: Technologies and Applications, Q. F. Hassan, Ed. Wiley-IEEE Press, 2018, pp. 153–190. xxii, xxiv, xxv, 10, 12, 15, 157

- [50] M. G. Samaila, M. Z. V. José, J. B. F. Sequeiros, M. M. Freire, and P. R. M. Inácio, “IoT-HarPSecA: A Framework for Facilitating the Design and Development of Secure IoT Devices,” in *Proceedings of the 14th International Conference on Availability, Reliability and Security*, ser. ARES '19. New York, NY, USA: ACM, 2019, pp. 72:1–72:7. [Online]. Available: <http://doi.acm.org/10.1145/3339252.3340514> xxiii, xxvi, 10, 11, 95, 115, 158
- [51] M. G. Samaila, C. Lopes, E. Aires, J. B. F. Sequeiros, T. Simões, M. M. Freire, and P. R. M. Inácio, “A Preliminary Evaluation of the SRE and SBPG Components of the IoT-HarPSecA Framework,” in *Proceedings of the 2020 Global Internet of Things Summit (GloTS)*. IEEE, 2020, pp. 1–7. xxiii, xxiv, xxvi, 10, 12, 115, 116, 158
- [52] M. G. Samaila, M. Neto, D. A. B. Fernandes, M. M. Freire, and P. R. M. Inacio, “Security Challenges of the Internet of Things,” in *Beyond the Internet of Things: Everything Interconnected*, J. M. Batalla, G. Mastorakis, C. X. Mavromoustakis, and E. Pallis, Eds. Springer International Publishing, 2017, pp. 53–82. xxiv, xxv, 9, 11, 15, 157
- [53] J. B. F. Sequeiros, F. T. Chimuco, M. G. Samaila, M. M. Freire, and P. R. M. Inácio, “Attack and System Modeling Applied to IoT, Cloud, and Mobile Ecosystems: Embedding Security by Design,” *ACM Computing Surveys*, vol. 53, no. 2, Mar. 2020. [Online]. Available: <https://doi.org/10.1145/3376123> xxv, 12
- [54] L. Theodosiou and J. Green, “Emerging Challenges in Using Health Information from the Internet,” *Advances in Psychiatric Treatment*, vol. 9, no. 5, p. 387–396, 2003. 2
- [55] Z. C. Fluhr and P. T. Porter, “Advanced Mobile Phone Service: Control Architecture,” *The Bell System Technical Journal*, vol. 58, no. 1, pp. 43–69, Jan 1979. 2
- [56] R. Rahdar, J. T. Stracener, and E. V. Olinick, “A Systems Engineering Approach to Improving the Accuracy of Mobile Station Location Estimation,” *IEEE Systems Journal*, vol. 8, no. 1, pp. 14–22, March 2014. 2
- [57] A. Zanella, N. Bui, A. Castellani, L. Vangelista, and M. Zorzi, “Internet of Things for Smart Cities,” *IEEE Internet of Things Journal*, vol. 1, no. 1, pp. 22–32, Feb 2014. 15
- [58] M. Zhang, F. Sun, and X. Cheng, “Architecture of Internet of Things and Its Key Technology Integration Based-On RFID,” in *Proceedings of the IEEE International Symposium on 2012 Fifth International Symposium on Computational Intelligence and Design*, vol. 1, Oct 2012, pp. 294–297. 15

- [59] J. S. Kumar and D. R. Patel, “a survey on internet of things: Security and privacy issues,” *International Journal of Computer Applications*, vol. 90, no. 11, pp. 0975–8887, 2014. 15
- [60] K. Zhao and L. Ge, “A Survey on the Internet of Things Security,” in *Proceedings of the IEEE International Symposium on 2013 Ninth International Conference on Computational Intelligence and Security*, Dec 2013, pp. 663–667. 15
- [61] H. Suo, J. Wan, C. Zou, and J. Liu, “Security in the Internet of Things: A Review,” in *Proceedings of the 2012 International Conference on Computer Science and Electronics Engineering*, vol. 3, March 2012, pp. 648–651. 15
- [62] J. Liu, X. Li, X. Chen, Y. Zhen, and L. Zeng, “Applications of Internet of Things on smart Grid in China,” in *Proceedings of the 13th International Conference on Advanced Communication Technology (ICACT2011)*, Feb 2011, pp. 13–17. 15
- [63] G. Gan, Z. Lu, and J. Jiang, “Internet of Things Security Analysis,” in *Proceedings of the 2011 International Conference on Internet Technology and Applications*, Aug 2011, pp. 1–4. 15, 16
- [64] Q. Jing, A. V. Vasilakos, J. Wan, J. Lu, and D. Qiu, “Security of the Internet of Things: Perspectives and Challenges,” *Wireless Networks*, vol. 20, no. 8, pp. 2481–2501, 2014. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1570870512000674> 15
- [65] Q. Ou, Y. Zhen, X. Li, Y. Zhang, and L. Zeng, “Application of Internet of Things in Smart Grid Power Transmission,” in *Proceedings of the 2012 Third FTRA International Conference on Mobile, Ubiquitous, and Intelligent Computing*, June 2012, pp. 96–100. 16
- [66] R. Ravindran, J. Yomas, and J. E. Sebastian, “IoT : A Review On Security Issues And Measures,” *Engineering Science and Technology: An International Journal (ESTIJ)*, vol. 5, no. 6, pp. 348–351, 2015. 16
- [67] H. HaddadPajouh, A. Dehghantanha, R. M. Parizi, M. Aledhari, and H. Karimipour, “A Survey on Internet of Things Security: Requirements, Challenges, and Solutions,” *ELSEVIER Internet of Things*, p. 100129, 2019. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S2542660519302288> 17
- [68] S. Nastic, H. Truong, and S. Dustdar, “Data and Control Points: A Programming Model for Resource-constrained IoT Cloud Edge Devices,” in *Proceedings of the 2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, Oct 2017, pp. 3535–3540. 19
- [69] P. Varga, S. Plosz, G. Soos, and C. Hegedus, “Security Threats and Issues in Automation IoT,” in *Proceedings of the IEEE International Symposium on 2017 IEEE*

- 13th International Workshop on Factory Communication Systems (WFCS), May 2017, pp. 1–6. 20, 21
- [70] X. Yi, M. Zhou, and J. Liu, “Design of Smart Home Control System by Internet of Things Based on ZigBee,” in *Proceedings of the IEEE International Symposium on 2016 IEEE 11th Conference on Industrial Electronics and Applications (ICIEA)*, June 2016, pp. 128–133. 22
- [71] Q. Ou, Y. Zhen, X. Li, Y. Zhang, and L. Zeng, “Application of Internet of Things in Smart Grid Power Transmission,” in *Proceedings of the 2012 Third FTRA International Conference on Mobile, Ubiquitous, and Intelligent Computing*, June 2012, pp. 96–100. 22
- [72] W. Yuan, Y. Li, and C. Liu, “Integrated Intelligence of Long Bridge Disaster Prevention and Mitigation by Focusing on Bearing Performance,” in *Proceedings of the IEEE International Symposium on IEEE International Conference on Smart City and Systems Engineering (ICSCSE)*, Nov 2016, pp. 54–58. 23
- [73] Zhu Yongjun, Zhu Xueli, Zhu Shuxian, and Guo shenghui, “Intelligent Transportation System Based on Internet of Things,” in *Proceedings of the IEEE International Symposium on IEEE World Automation Congress 2012*, June 2012, pp. 1–3. 23
- [74] E. Union. (2010, Jul.) Directive 2010/40/Eu Of The European Parliament And Of The Council. [Online]. Available: <https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32010L0040&from=EN> 23
- [75] M. P. R. S. Kiran, P. Rajalakshmi, K. Bharadwaj, and A. Acharyya, “Adaptive Rule Engine Based IoT Enabled Remote Health Care Data Acquisition and Smart Transmission System,” in *Proceedings of the IEEE World Forum on Internet of Things (WF-IoT)*, March 2014, pp. 253–258. 25
- [76] Y. J. Qu, X. G. Ming, S. Q. Qiu, Z. W. Liu, X. Y. Zhang, and Z. T. Hou, “A Framework for Smart Manufacturing Systems Based on the Stakeholders’ Value,” in *Proceedings of the IEEE International Conference on Advanced Manufacturing (ICAM)*, Nov 2018, pp. 239–242. 25
- [77] V. Fore, A. Khanna, R. Tomar, and A. Mishra, “Intelligent Supply Chain Management System,” in *Proceedings of the IEEE International Conference on Advances in Computing and Communication Engineering (ICACCE)*, Nov 2016, pp. 296–302. 25
- [78] S. Yuvaraj and M. Sangeetha, “Smart Supply Chain Management Using Internet of things (IoT) and Low Power Wireless Communication Systems,” in *Proceedings of the IEEE International Symposium on IEEE International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, March 2016, pp. 555–558. 25

- [79] W. Sun, J. Liu, and H. Zhang, “When Smart Wearables Meet Intelligent Vehicles: Challenges and Future Directions,” *IEEE Wireless Communications*, vol. 24, no. 3, pp. 58–65, June 2017. 26
- [80] F. Viani, M. Bertolli, and A. Polo, “Low-Cost Wireless System for Agrochemical Dosage Reduction in Precision Farming,” *IEEE Sensors Journal*, vol. 17, no. 1, pp. 5–6, Jan 2017. 26
- [81] S. Jo, D. Park, H. Park, and S. Kim, “Smart Livestock Farms Using Digital Twin: Feasibility Study,” in *Proceedings of the IEEE International Conference on Information and Communication Technology Convergence (ICTC)*, Oct 2018, pp. 1461–1463. 26
- [82] G. Sabaliauskaite and S. Adepu, “Integrating Six-Step Model with Information Flow Diagrams for Comprehensive Analysis of Cyber-Physical System Safety and Security,” in *Proceedings of the IEEE 18th International Symposium on High Assurance Systems Engineering (HASE)*, Jan 2017, pp. 41–48. 26
- [83] H. Tyagi, “Distributed Computing with Privacy,” in *Proceedings of the IEEE International Symposium on Information Theory Proceedings*, July 2012, pp. 1157–1161. 26
- [84] P. Radanliev, D. D. Roure, S. Cannady, R. Montalvo, R. Nicolescu, and M. Huth, “Economic Impact of IoT Cyber Risk - Analysing Past and Present to Predict the Future Developments in IoT Risk Analysis and IoT Cyber Insurance,” *IET Conference Proceedings*, pp. 3 (9 pp.)–3 (9 pp.)(1), January 2018. [Online]. Available: <https://digital-library.theiet.org/content/conferences/10.1049/cp.2018.0003> 34, 35
- [85] M. Ammar, M. Washha, and B. Crispo, “WISE: Lightweight Intelligent Swarm Attestation Scheme for IoT (The Verifier’s Perspective),” in *Proceedings of the 2018 14th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Oct 2018, pp. 1–8. 35
- [86] C. Onwubiko, “Cyber Security Operations Centre: Security Monitoring for Protecting Business and Supporting Cyber Defense Strategy,” in *Proceedings of the IEEE International Conference on Cyber Situational Awareness, Data Analytics and Assessment (CyberSA)*, June 2015, pp. 1–10. 35
- [87] Y. Seralathan, T. T. Oh, S. Jadhav, J. Myers, J. P. Jeong, Y. H. Kim, and J. N. Kim, “IoT Security Vulnerability: A Case Study of a Web Camera,” in *Proceedings of the IEEE 20th International Conference on Advanced Communication Technology (ICACT)*, Feb 2018, pp. 172–177. 36
- [88] J. R. C. Nurse, A. Erola, I. Agrafiotis, M. Goldsmith, and S. Creese, “Smart Insiders: Exploring the Threat from Insiders Using the Internet-of-Things,” in *Proceed-*

- ings of the IEEE International Workshop on Secure Internet of Things (SIoT)*, Sep. 2015, pp. 5–14. 36, 37
- [89] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, “Demystifying IoT Security: An Exhaustive Survey on IoT Vulnerabilities and a First Empirical Look on Internet-Scale IoT Exploitations,” *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2702–2733, thirdquarter 2019. 37, 43
- [90] OWASP. (2016, Aug.) Internet of Things Project. [Online]. Available: https://www.owasp.org/index.php/OWASP_Internet_of_Things_Project?utm_source=datafloq&utm_medium=ref&utm_campaign=datafloq#tab=IoT_Vulnerabilities 37
- [91] N. Kumar, J. Madhuri, and M. Channe Gowda, “Review on Security and Privacy Concerns in Internet of Things,” in *Proceedings of the 2017 International Conference on IoT and Application (ICIOT)*, May 2017, pp. 1–5. 38
- [92] T. Chan, Y. Ren, Y. Tseng, and J. Chen, “Multi-Slot Allocation Protocols for Massive IoT Devices With Small-Size Uploading Data,” *IEEE Wireless Communications Letters*, vol. 8, no. 2, pp. 448–451, April 2019. 38
- [93] A. Chaudhary, S. K. Peddoju, and K. Kadarla, “Study of Internet-of-Things Messaging Protocols Used for Exchanging Data with External Sources,” in *Proceedings of the 2017 IEEE 14th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*, Oct 2017, pp. 666–671. 38
- [94] M. Khurram, H. Kumar, A. Chandak, V. Sarwade, N. Arora, and T. Quach, “Enhancing Connected Car Adoption: Security and Over the Air Update Framework,” in *Proceedings of the IEEE 3rd World Forum on Internet of Things (WF-IoT)*, Dec 2016, pp. 194–198. 38
- [95] S. Shiaeles, N. Kolokotronis, and E. Bellini, “IoT Vulnerability Data Crawling and Analysis,” in *Proceedings of the IEEE World Congress on Services (SERVICES)*, vol. 2642-939X, July 2019, pp. 78–83. 38
- [96] Kaspersky. (2019) IoT under fire: Kaspersky Detects more than 100 Million Attacks on Smart Devices in H1 2019. Kaspersky. [Online]. Available: https://www.kaspersky.com/about/press-releases/2019_iot-under-fire-kaspersky-detects-more-than-100-million-attacks-on-smart-devices-in-h1-2019 39
- [97] SpiceWorks. (2016, Aug.) Battling the Big Network Security Hack. [Online]. Available: <https://www.spiceworks.com/it-articles/it-security/> 39
- [98] A. Djenna and D. Eddine Saïdouni, “Cyber Attacks Classification in IoT-Based-Healthcare Infrastructure,” in *Proceedings of the 2018 IEEE 2nd Cyber Security in Networking Conference (CSNet)*, 2018, pp. 1–4. 39

- [99] S. Sen and C. Jayawardena, "Analysis of Cyber-Attack in Big Data IoT and Cyber-Physical Systems - A Technical Approach to Cybersecurity Modeling," in *Proceedings of the 2019 IEEE 5th International Conference for Convergence in Technology (I2CT)*, 2019, pp. 1–7. 39
- [100] S. Esnaashari, I. Welch, and P. Komisarczuk, "Determining Home Users' Vulnerability to Universal Plug and Play (UPnP) Attacks," in *Proceedings of the IEEE 27th International Conference on Advanced Information Networking and Applications Workshops*, March 2013, pp. 725–729. 39
- [101] P. Solapurkar, "Building Secure Healthcare Services Using OAuth 2.0 and JSON Web Token in IoT Cloud Scenario," in *Proceedings of the IEEE 2nd International Conference on Contemporary Computing and Informatics (IC3I)*, Dec 2016, pp. 99–104. 39
- [102] S. Yamaguchi and M. S. B. A. Malek, "A Model Checking-Based Analysis Method of Cyber Attack in IoT System by Agent-Oriented Petri Net," in *Proceedings of the IEEE International Symposium on 2018 IEEE 7th Global Conference on Consumer Electronics (GCCE)*, Oct 2018, pp. 581–584. 39
- [103] K. Jain. (2015, Oct.) Critical Netgear Router Exploit Allows anyone to Hack You Remotely. [Online]. Available: <http://thehackernews.com/2015/10/netgear-router-hack.html> 39
- [104] S. News. (2016) Botnet Uses IoT Devices to Power Massive DDoS Attacks. [Online]. Available: <http://www.securityweek.com/botnet-uses-iotdevices-power-massive-ddos-attacks> 39
- [105] Tripwire. (2016) Tripwire VERT Research: SOHO Wireless Router (In)Security. [Online]. Available: <http://www.tripwire.com/register/soho-wireless-routerinsecurity/showMeta/2/> 40
- [106] E. Rubio-Drosdov, D. Díaz-Sánchez, F. Almenárez, P. Arias-Cabarcos, and A. Marín, "Seamless Human-Device Interaction in the Internet of Things," *IEEE Transactions on Consumer Electronics*, vol. 63, no. 4, pp. 490–498, November 2017. 40
- [107] W. Z. Khan, M. Y. Aalsalem, M. K. Khan, and Q. Arshad, "Data and Privacy: Getting Consumers to Trust Products Enabled by the Internet of Things," *IEEE Consumer Electronics Magazine*, vol. 8, no. 2, pp. 35–38, March 2019. 40, 44, 58
- [108] S. Imtiaz, R. Sadre, and V. Vlassov, "On the Case of Privacy in the IoT Ecosystem: A Survey," in *Proceedings of the 2019 International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*, July 2019, pp. 1015–1024. 40, 41

- [109] Security-Marathon. (2016, May) The Difference between Data Privacy and Data Security. [Online]. Available: <http://www.security-marathon.be/?p=2007> 40
- [110] A. F. Westin, "Privacy and Freedom," *Washington & Lee Law Review*, vol. 25, pp. 167–170, 1968. [Online]. Available: <https://scholarlycommons.law.wlu.edu/wlulr/vol25/iss1/20> 40
- [111] K. Renaud and D. Gálvez-Cruz, "Privacy: Aspects, Definitions and a Multi-Faceted Privacy Preservation Approach," in *Proceedings of the IEEE Information Security for South Africa*, Aug 2010, pp. 1–8. 40
- [112] W. AL-mawee, "Privacy and Security Issues in IoT Healthcare Applications for the Disabled Users: a Survey," Master's thesis, Western Michigan University, 2015. [Online]. Available: https://scholarworks.wmich.edu/cgi/viewcontent.cgi?article=1661&context=masters_theses 40
- [113] P. Paganini. (2015, Sep.) How Hackers Violate Privacy and Security of the Smart Home. [Online]. Available: <https://resources.infosecinstitute.com/how-hackers-violate-privacy-and-security-of-the-smart-home/#gref> 41
- [114] S. Sen Gupta, M. Shad Khan, and T. Sethi, "Latest Trends in Security, Privacy and Trust in IoT," in *Proceedings of the IEEE 3rd International conference on Electronics, Communication and Aerospace Technology (ICECA)*, June 2019, pp. 382–385. 41
- [115] Y. Yang, L. Wu, G. Yin, L. Li, and H. Zhao, "A Survey on Security and Privacy Issues in Internet-of-Things," *IEEE Internet of Things Journal*, vol. 4, no. 5, pp. 1250–1258, Oct 2017. 41
- [116] M. I. Hossain, L. Lin, and J. Markendahl, "A Comparative Study of IoT-Communication Systems Cost Structure: Initial Findings of Radio Access Networks Cost," in *Proceedings of the 2018 11th CMI International Conference: Prospects and Challenges Towards Developing a Digital Economy within the EU*, Nov 2018, pp. 49–55. 42
- [117] L. Columbus. (2018, Dec.) 2018 Roundup of Internet of Things Forecasts and Market Estimates. [Online]. Available: <https://www.forbes.com/sites/louiscolombus/2018/12/13/2018-roundup-of-internet-of-things-forecasts-and-market-estimates/#b20f8777d838> 42
- [118] A. O. Joseph, J. Raffety, P. Morrow, L. Zhiwei, C. Nugent, S. McClean, and G. Duca-tel, "Securing Self-organizing IoT Ecosystem: A Distributed Ledger Technology Approach," in *Proceedings of the 2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, April 2019, pp. 809–814. 42

- [119] A. K. Ray and A. Bagwari, "Study of Smart Home Communication Protocol's and Security Privacy Aspects," in *Proceedings of the 7th International Conference on Communication Systems and Network Technologies (CSNT)*, Nov 2017, pp. 240–245. 42
- [120] A. Kruschke. (2018, Feb.) Hidden IoT Security Issues Pose a Huge Threat to Your Network. [Online]. Available: <https://www.arcserve.com/insights/iot-security-issues/> 42
- [121] Z. Kazemi, A. Papadimitriou, I. Souvatzoglou, E. Aerabi, M. M. Ahmed, D. Hely, and V. Beroulle, "On a Low Cost Fault Injection Framework for Security Assessment of Cyber-Physical Systems: Clock Glitch Attacks," in *Proceedings of the IEEE 4th International Verification and Security Workshop (IVSW)*, July 2019, pp. 7–12. 42
- [122] B. Vignau, R. Khoury, and S. Hallé, "10 Years of IoT Malware: A Feature-Based Taxonomy," in *Proceedings of the IEEE International Symposium on IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, July 2019, pp. 458–465. 42
- [123] Veracode. (2015, Apr.) Veracode Study Reveals the Internet of Things Poses Cybersecurity Risk. [Online]. Available: <https://www.veracode.com/veracode-study-reveals-internet-things-poses-cybersecurity-risk> 42
- [124] P. Newman. (2019, Jan.) IoT Report: How Internet of Things Technology Growth is Reaching Mainstream Companies and Consumers. [Online]. Available: <https://www.businessinsider.com/internet-of-things-report> 43
- [125] TRUSTe. (2015, Jan.) Majority of Consumers Want to Own the Personal Data Collected from their Smart Devices: Survey. [Online]. Available: <https://www.trustarc.com/blog/2015/01/05/majority-consumers-want-own-personal-data-survey/> 43
- [126] R. Williams, E. McMahon, S. Samtani, M. Patton, and H. Chen, "Identifying Vulnerabilities of Consumer Internet of Things (IoT) Devices: A Scalable Approach," in *Proceedings of the IEEE International Symposium on 2017 IEEE International Conference on Intelligence and Security Informatics (ISI)*, July 2017, pp. 179–181. 43
- [127] M. Smith. (2014) Peeping into 73,000 Unsecured Security Cameras Thanks to Default Passwords. NetworkWorld. [Online]. Available: <http://www.networkworld.com/article/2844283/microsoft-subnet/peeping-into-73-000-unsecured-security-cam-Eras-thanks-todefaut-passwords.html> 43

- [128] K. Zetter. (2012) Popular Surveillance Cameras Open to Hackers, Researcher Says. WIRED,. [Online]. Available: <http://www.wired.com/2012/05/cctv-hack/> 43
- [129] B. Lagesse, K. Wu, J. Shorb, and Z. Zhu, “Detecting Spies in IoT Systems using Cyber-Physical Correlation,” in *Proceedings of the 2018 IEEE International Conference on Pervasive Computing and Communications Workshops (PerCom Workshops)*, March 2018, pp. 185–190. 43
- [130] C. Bradley, S. El-Tawab, and M. H. Heydari, “Security Analysis of an IoT System Used for Indoor Localization in Healthcare Facilities,” in *Proceedings of the 2018 Systems and Information Engineering Design Symposium (SIEDS)*, April 2018, pp. 147–152. 43
- [131] TRUSTe. (2014) 59 % of U.S. Internet Users know Smart Devices can Collect Information about their Personal Activities. TRUSTe. [Online]. Available: <https://www.prnewswire.com/news-releases/59-of-us-internet-users-know-smart-devices-can-collect-information-about-their-personal-activities-261073901.html> 44
- [132] H. Packard. (2015) HP Study Reveals 70 Percent of Internet of Things Devices Vulnerable to Attack. Hewlett Packard. [Online]. Available: <https://www8.hp.com/us/en/hp-news/press-release.html?id=1744676> 44
- [133] M. Rozenfeld, K. Albrecht, and K. Michael, “The Value of Privacy: Safeguarding your Information in the Age of the Internet of Everything,” *The Institute (IEEE)*, 2014. [Online]. Available: <https://works.bepress.com/kmichael/449/> 44
- [134] Y. Chandu, K. S. R. Kumar, N. V. Prabhukhanolkar, A. N. Anish, and S. Rawal, “Design and Implementation of Hybrid Encryption for Security of IoT Data,” in *Proceedings of the 2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon)*, Aug 2017, pp. 1228–1231. 44
- [135] C. Lin, C. Tsai, and C. Kao, “Lower Power Data Transport Protection for Internet of Things (IoT),” in *Proceedings of the 2017 IEEE Conference on Dependable and Secure Computing*, Aug 2017, pp. 468–470. 44
- [136] G. Singh and Supriya, “Modified Vigenere Encryption Algorithm and Its Hybrid Implementation with Base64 and AES,” in *Proceedings of the IEEE 2nd International Conference on Advanced Computing, Networking and Security*, Dec 2013, pp. 232–237. 45
- [137] A. Fragkiadakis, P. Charalampidis, S. Papadakis, and E. Tragos, “Experiences with Deploying Compressive Sensing and Matrix Completion Techniques in IoT Devices,” in *Proceedings of the IEEE 19th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, Dec 2014, pp. 213–217. 45

- [138] D. Tourky, M. ElKawkagy, and A. Keshk, "Homomorphic Encryption the "Holy Grail" of Cryptography," in *Proceedings of the 2016 2nd IEEE International Conference on Computer and Communications (ICCC)*, Oct 2016, pp. 196–201. 45
- [139] I. Yoon, N. Cao, A. Amaravati, and A. Raychowdhury, "A 55nm 50nJ/Encode 13nJ/Decode Homomorphic Encryption Crypto-Engine for IoT Nodes to Enable Secure Computation on Encrypted Data," in *Proceedings of the IEEE International Symposium on 2019 IEEE Custom Integrated Circuits Conference (CICC)*, April 2019, pp. 1–4. 45
- [140] S. Donaldson. (2013, Apr.) Security Tradeoffs - a Culture of Convenience. [Online]. Available: <https://www.securityweek.com/security-tradeoffs-culture-convenience> 45
- [141] P. W. Dowd and J. T. McHenry, "Network Security: It's Time to Take it Seriously," *Computer*, vol. 31, no. 9, pp. 24–28, Sep. 1998. 45
- [142] W. Bai, C. Lynton, C. Papamanthou, and M. L. Mazurek, "Understanding User Tradeoffs for Search in Encrypted Communication," in *Proceedings of the IEEE European Symposium on Security and Privacy (EuroS P)*, April 2018, pp. 258–272. 45
- [143] M. Krishna Kagita, "Security and Privacy Issues for Business Intelligence in IoT," in *Proceedings of the IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3)*, Jan 2019, pp. 206–212. 46
- [144] J. Payne, K. Budhraj, and A. Kundu, "How Secure is your IoT Network?" in *Proceedings of the IEEE International Congress on Internet of Things (ICIOT)*, July 2019, pp. 181–188. 46
- [145] P. S. Munoz, N. Tran, B. Craig, B. Dezfouli, and Y. Liu, "Analyzing the Resource Utilization of AES Encryption on IoT Devices," in *Proceedings of the 2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, Nov 2018, pp. 1200–1207. 46
- [146] S. Ray, T. Hoque, A. Basak, and S. Bhunia, "The Power Play: Security-Energy Trade-Offs in the IoT Regime," in *Proceedings of the 2016 IEEE 34th International Conference on Computer Design (ICCD)*, Oct 2016, pp. 690–693. 46
- [147] R. Roman, P. Najera, and J. Lopez, "Securing the Internet of Things," *Computer*, vol. 44, no. 9, pp. 51–58, Sep. 2011. 46
- [148] N. Khan, N. Sakib, I. Jerin, S. Quader, and A. Chakrabarty, "Performance Analysis of Security Algorithms for IoT Devices," in *Proceedings of the 2017 IEEE Region 10 Humanitarian Technology Conference (R10-HTC)*, Dec 2017, pp. 130–133. 46

- [149] S. J. Johnston, M. Apetroaie-Cristea, M. Scott, and S. J. Cox, “Applicability of Commodity, Low Cost, Single Board Computers for Internet of Things Devices,” in *Proceedings of the 2016 IEEE 3rd World Forum on Internet of Things (WF-IoT)*, Dec 2016, pp. 141–146. 46
- [150] C. Ou, F. Hsu, and C. Lai, “Keep Rogue IoT Away: IoT Detector Based on Diversified TLS Negotiation,” in *Proceedings of the 2019 IEEE Intl Conf on Dependable, Autonomic and Secure Computing, Intl Conf on Pervasive Intelligence and Computing, Intl Conf on Cloud and Big Data Computing, Intl Conf on Cyber Science and Technology Congress (DASC/PiCom/CBDCCom/CyberSciTech)*, Aug 2019, pp. 548–555. 46
- [151] P. Hao, X. Wang, and W. Shen, “A Collaborative PHY-Aided Technique for End-to-End IoT Device Authentication,” *IEEE Access*, vol. 6, pp. 42 279–42 293, 2018. 46
- [152] N. Ericsson, T. Lennvall, J. Åkerberg, and M. Björkman, “A Flexible Communication Stack Design for Time Sensitive Embedded Systems,” in *Proceedings of the 2017 IEEE International Conference on Industrial Technology (ICIT)*, March 2017, pp. 1112–1117. 46
- [153] C. Scott, D. Wynne, and C. Boonthum-Denecke, “Examining the Privacy of Login Credentials using Web-Based Single Sign-on - Are We Giving Up Security and Privacy for Convenience?” in *Proceedings of the 2016 Cybersecurity Symposium (CYBERSEC)*, April 2016, pp. 74–79. 47
- [154] G. Kambourakis, C. Koliass, and A. Stavrou, “The Mirai Botnet and the IoT Zombie Armies,” in *Proceedings of the MILCOM 2017 - 2017 IEEE Military Communications Conference (MILCOM)*, Oct 2017, pp. 267–272. 47
- [155] B. D. Patel and A. D. Patel, “A Trust Based Solution for Detection of Network Layer Attacks in Sensor Networks,” in *Proceedings of the International Conference on Micro-Electronics and Telecommunication Engineering (ICMETE)*, Sept 2016, pp. 121–126. 48
- [156] A. Das, Rishikesh, and P. N. Astya, “A Relative Survey of Various LEACH based Routing Protocols in Wireless Sensor Networks,” in *Proceedings of the International Conference on Computing, Communication and Automation (ICCCA)*, May 2017, pp. 630–636. 48
- [157] H. Chan, A. Perrig, and D. Song, “Random Key Predistribution Schemes for Sensor Networks,” in *Proceedings of the Symposium on Security and Privacy, 2003.*, May 2003, pp. 197–213. 48
- [158] T. Yalçın, “Compact ECDSA Engine for IoT Applications,” *Electronics Letters*, vol. 52, no. 15, pp. 1310–1312, 2016. 49

- [159] Y. Liu, J. Briones, R. Zhou, and N. Magotra, “Study of Secure Boot with a FPGA-based IoT Device,” in *Proceedings of the IEEE 60th International Midwest Symposium on Circuits and Systems (MWSCAS)*, Aug 2017, pp. 1053–1056. 49
- [160] D. Borio, C. O’Driscoll, and J. Fortuny, “GNSS Jammers: Effects and Countermeasures,” in *Proceedings of the 6th ESA Workshop on Satellite Navigation Technologies (Navitec 2012) European Workshop on GNSS Signals and Signal Processing*, Dec 2012, pp. 1–7. 50
- [161] R. Saleh, S. Wilton, S. Mirabbasi, A. Hu, M. Greenstreet, G. Lemieux, P. P. Pande, C. Grecu, and A. Ivanov, “System-on-Chip: Reuse and Integration,” *Proceedings of the IEEE*, vol. 94, no. 6, pp. 1050–1069, 2006. 51
- [162] P. Magarshack, “Breakthrough Technologies and Reference Designs for New IoT Applications,” in *Proceedings of the 2015 Symposium on VLSI Circuits (VLSI Circuits)*, 2015, pp. C42–C43. 52
- [163] J. Cai, M. Takemoto, and H. Nakajo, “Implementation of DNN on a RISC-V Open Source Microprocessor for IoT Devices,” in *Proceedings of the 2018 IEEE 7th Global Conference on Consumer Electronics (GCCE)*, 2018, pp. 295–299. 52
- [164] V. Vujovic, S. Jokic, and M. Maksimovic, “Power Efficiency Analysis in Internet of Things Sensor Nodes,” in *Proceedings of the IEEE International Symposium on 2nd International Electronic Conference on Sensors and Applications session Sensor Networks*, 2015, pp. 1–6. 53
- [165] A. Frøyttlog, M. A. Haglund, L. R. Cenkeramaddi, T. Jordbru, R. A. Kjellby, and B. Beferull-Lozano, “Design and Implementation of a Long-Range Low-Power Wake-up Radio for IoT Devices,” in *Proceedings of the 2019 IEEE 5th World Forum on Internet of Things (WF-IoT)*, 2019, pp. 247–250. 53
- [166] Jinbum Kim, J. A. Noquil, Teik keng Tan, Chung-Lin Wu, and Seung-Yong Choi, “Multi-flip Chip on Lead Frame Overmolded IC Package: a Novel Packaging Design to Achieve High Performance and Cost Effective Module Package,” in *Proceedings of the Electronic Components and Technology, 2005. ECTC ’05.*, 2005, pp. 1819–1821 Vol. 2. 53
- [167] K. Li, F. Hsueh, C. Shen, J. Shieh, H. Chen, W. Huang, H. Chiu, C. Yang, T. Hsieh, B. Chen, W. Chen, K. Hsu, M. Chang, and W. Yeh, “FinFET-based Monolithic 3D+ with RRAM Array and Computing in Memory SRAM for Intelligent IoT Chip Application,” in *Proceedings of the 2018 IEEE SOI-3D-Subthreshold Microelectronics Technology Unified Conference (S3S)*, 2018, pp. 1–3. 54
- [168] O. Hahm, E. Baccelli, H. Petersen, and N. Tsiftes, “Operating Systems for Low-End Devices in the Internet of Things: A Survey,” *IEEE Internet of Things Journal*, vol. 3, no. 5, pp. 720–734, 2016. 54

- [169] S. Raman, R. Weigel, and T. Lee, “The Internet of Space (IoS): A Future Backbone for the Internet of Things?” in *IEEE Internet of Things*, 2016. 54
- [170] I. Tudosa, F. Picariello, E. Balestrieri, L. De Vito, and F. Lamonaca, “Hardware Security in IoT Era: the Role of Measurements and Instrumentation,” in *Proceedings of the 2019 II Workshop on Metrology for Industry 4.0 and IoT (MetroInd4.0 IoT)*, 2019, pp. 285–290. 55
- [171] S. M. Sundharam, “Modeling of High Level System Requirements- An IoT Case-Study,” in *Proceedings of the IEEE International Conference on Vision Towards Emerging Trends in Communication and Networking (ViTECoN)*, March 2019, pp. 1–7. 58
- [172] S. Ullah, M. Iqbal, and A. M. Khan, “A Survey on Issues in Non-Functional Requirements Elicitation,” in *Proceedings of the IEEE International Symposium on IEEE International Conference on Computer Networks and Information Technology*, July 2011, pp. 333–340. 58
- [173] A. Assiri and H. Almagwashi, “IoT Security and Privacy Issues,” in *Proceedings of the IEEE 1st International Conference on Computer Applications Information Security (ICCAIS)*, April 2018, pp. 1–5. 58
- [174] G. Cybenko and J. Hughes, “No Free Lunch in Cyber Security,” in *Proceedings of the First ACM Workshop on Moving Target Defense*, ser. MTD ’14. New York, NY, USA: ACM, 2014, pp. 1–12. [Online]. Available: <http://doi.acm.org/10.1145/2663474.2663475> 59
- [175] D. Minoli, K. Sohraby, and J. Kouns, “IoT Security (IoTSec) Considerations, Requirements, and Architectures,” in *Proceedings of the 214th IEEE Annual Consumer Communications Networking Conference (CCNC)*, Jan 2017, pp. 1006–1007. 59
- [176] CRYPTREC, “Cryptographic Technology Guideline (Lightweight Cryptography),” *CRYPTREC Lightweight Cryptography Working Group*, vol. 1, 2017. 68, 78, 85, 90, 93, 112, 150, 152
- [177] H. Liu, D. Han, and D. Li, “Fabric-iot: A Blockchain-Based Access Control System in IoT,” *IEEE Access*, vol. 8, pp. 18 207–18 218, 2020. 68
- [178] M. U. Aftab, Y. Munir, A. Oluwasanmi, Z. Qin, M. H. Aziz, Zakria, N. T. Son, and V. D. Tran, “A Hybrid Access Control Model With Dynamic COI for Secure Localization of Satellite and IoT-Based Vehicles,” *IEEE Access*, vol. 8, pp. 24 196–24 208, 2020. 68
- [179] N. Chaabouni, M. Mosbah, A. Zemmari, and C. Sauvignac, “A OneM2M Intrusion Detection and Prevention System based on Edge Machine Learning,” in *Proceed-*

ings of the NOMS 2020 IEEE/IFIP Network Operations and Management Symposium, 2020, pp. 1–7. 68

- [180] A. Esfahani, G. Mantas, R. Maticsek, F. B. Saghezchi, J. Rodriguez, A. Bicaku, S. Maksuti, M. G. Tauber, C. Schmittner, and J. Bastos, “A Lightweight Authentication Mechanism for M2M Communications in Industrial IoT Environment,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 288–296, 2019. 68
- [181] S. Banerjee, V. Odelu, A. K. Das, J. Srinivas, N. Kumar, S. Chattopadhyay, and K. R. Choo, “A Provably Secure and Lightweight Anonymous User Authenticated Session Key Exchange Scheme for Internet of Things Deployment,” *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8739–8752, 2019. 68
- [182] K. Shibagaki, Y. Nozaki, and M. Yoshikawa, “Tamper Resistance Evaluation of Noise Based Countermeasure for IoT Devices,” in *Proceedings of the 2018 IEEE International Meeting for Future of Electron Devices, Kansai (IMFEDK)*, 2018, pp. 1–2. 68
- [183] M. N. Aman, B. Sikdar, K. C. Chua, and A. Ali, “Low Power Data Integrity in IoT Systems,” *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 3102–3113, 2018. 68
- [184] J. Wang, D. Wang, Y. Zhao, and T. Korhonen, “Fast Anti-Collision Algorithms in RFID Systems,” in *Proceedings of the IEEE International Symposium on International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies (UBICOMM’07)*, Nov 2007, pp. 75–80. 68
- [185] A. W. Nagpurkar and S. K. Jaiswal, “Anti-Collision in WSN and RFID Network Integration,” *International Journal of Science and Research (IJSR)*, vol. 4, no. 4, pp. 3008 – 3012, 2015. [Online]. Available: <https://pdfs.semanticscholar.org/9bf6/dbde07565b908e04305100bd8be03487c498.pdf> 68
- [186] L. Guan, C. Cao, P. Liu, X. Xing, X. Ge, S. Zhang, M. Yu, and T. Jaeger, “Building a Trustworthy Execution Environment to Defeat Exploits from both Cyber Space and Physical Space for ARM,” *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 3, pp. 438–453, 2019. 69
- [187] K. Zandberg, K. Schleiser, F. Acosta, H. Tschofenig, and E. Baccelli, “Secure Firmware Updates for Constrained IoT Devices Using Open Standards: A Reality Check,” *IEEE Access*, vol. 7, pp. 71 907–71 920, 2019. 69
- [188] S. Raza and R. M. Magnússon, “TinyIKE: Lightweight IKEv2 for Internet of Things,” *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 856–866, 2019. 69
- [189] H. Chien, “Group-Oriented Range-Bound Key Agreement for Internet of Things Scenarios,” *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 1890–1903, 2018. 69

- [190] R. Roman, C. Alcaraz, J. Lopez, and N. Sklavos, “Key Management Systems for Sensor Networks in the Context of the Internet of Things,” *Computers & Electrical Engineering*, vol. 37, no. 2, pp. 147 – 159, 2011, modern Trends in Applied Security: Architectures, Implementations and Applications. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0045790611000176> 69
- [191] A. A. Waleed, V. Kharchenko, D. Uzun, and O. Solovyov, “IoT-based Physical Security Systems: Structures and PSMECA Analysis,” in *Proceedings of the 2017 9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, vol. 2, 2017, pp. 870–873. 69
- [192] Z. Ren, X. Liu, R. Ye, and T. Zhang, “Security and Privacy on Internet of Things,” in *Proceedings of the 27th IEEE International Conference on Electronics Information and Emergency Communication (ICEIEC)*, July 2017, pp. 140–144. 69
- [193] I. Makhdoom, M. Abolhasan, J. Lipman, R. P. Liu, and W. Ni, “Anatomy of Threats to the Internet of Things,” *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1636–1675, Secondquarter 2019. 70
- [194] M. V. Staalduinen and Y. Joshi. (2019, Sep.) The IoT Security Landscape: Adoption and Harmonization of Security Solutions for the Internet of Things. TNO-Innovation for Life. [Online]. Available: <https://www.csa.gov.sg/news/press-releases/> 70
- [195] E. Parliament and C. of the European Union. (2016, May) The General Data Protection Regulation (GDPR) (EU) 2016/679. [Online]. Available: <https://gdpr-info.eu/> 71
- [196] ENISA. (2018, Nov.) Good Practices for Security of Internet of Things in the Context of Smart Manufacturing. [Online]. Available: <https://www.enisa.europa.eu/publications/good-practices-for-security-of-iot> 71, 73
- [197] ——. (2019, Nov.) ENISA Good Practices for Security of Smart Cars. [Online]. Available: <https://www.enisa.europa.eu/publications/enisa-good-practices-for-security-of-smart-cars> 71, 73
- [198] ——. (2019, Nov.) Good Practices for Security of IoT - Secure Software Development Lifecycle. [Online]. Available: <https://www.enisa.europa.eu/publications/good-practices-for-security-of-iot-1> 71
- [199] C. Legislature. (2018, Apr.) California Assembly Bill No. 1950 and Senate Bill 1386 Report. [Online]. Available: https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180AB1950 72

- [200] California Legislature. (2018, Sep.) Senate Bill No. 327. [Online]. Available: https://leginfo.legislature.ca.gov/faces/billTextClient.xhtml?bill_id=201720180SB327 72
- [201] U. Government. (2019) National Centre of Excellence for IoT Systems Cybersecurity, dubbed PETRAS. [Online]. Available: <https://www.petrashub.org/launch-of-the-the-petras-national-centre-of-excellence-for-iot-systems-cybersecurity/> 72
- [202] ISO/IEC. (2018) The ISO/IEC 27000 Family of Standards. [Online]. Available: <https://www.iso.org/standard/73906.html> 72
- [203] ——. (2012, Jul.) ISO/IEC 27032:2012 Information Technology: Security Techniques — Guidelines for Cybersecurity. [Online]. Available: <https://www.iso.org/standard/44375.html> 72
- [204] ——. (2011, Sep.) ISO/IEC 27035:2011 Information Technology: Security Techniques — Information Security Incident Management. [Online]. Available: <https://www.iso.org/standard/44379.html> 72
- [205] ——. (2011) ISO/IEC WD 27031 [ISO/IEC WD 27031] Information Technology: Guidelines for ICT Readiness for Business Continuity. [Online]. Available: <https://www.iso.org/standard/78771.html> 73
- [206] ISO. (2012, May) ISO 22301:2012 Societal Security: Business Continuity Management Systems — Requirements. [Online]. Available: <https://www.iso.org/standard/50038.html> 73
- [207] NIST. (2019, Jun.) NISTIR 8228: Considerations for Managing Internet of Things (IoT) Cybersecurity and Privacy Risks. [Online]. Available: <https://nvlpubs.nist.gov/nistpubs/ir/2019/NIST.IR.8228.pdf> 73
- [208] IoT Security Foundation. (2019, Nov.) Security Design Best Practice Guides Release 2. [Online]. Available: <https://www.iotsecurityfoundation.org/best-practice-guidelines/> 73
- [209] N. Ghosh, S. Chandra, V. Sachidananda, and Y. Elovici, “SoftAuthZ: A Context-Aware, Behavior-Based Authorization Framework for Home IoT,” *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 10 773–10 785, Dec 2019. 77
- [210] T. Eisenbarth, S. Kumar, C. Paar, A. Poschmann, and L. Uhsadel, “A Survey of Lightweight-Cryptography Implementations,” *IEEE Design Test of Computers*, vol. 24, no. 6, pp. 522–533, Nov 2007. 78, 85
- [211] F. Farahmand, W. Diehl, A. Abdulgadir, J. Kaps, and K. Gaj, “Improved Lightweight Implementations of CAESAR Authenticated Ciphers,” in *Proceedings*

of the 2018 IEEE 26th Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM), April 2018, pp. 29–36. 78

- [212] K. Lisickiy, V. Dolgov, and I. Lisickaya, “Block Cipher with Improved Dynamic Indicators of the Condition of a Random Substitution,” in *Proceedings of the 2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S T)*, Oct 2017, pp. 391–395. 79
- [213] R. Girija and H. Singh, “A New Substitution-Permutation Network Cipher using Walsh Hadamard Transform,” in *Proceedings of the 2017 International Conference on Computing and Communication Technologies for Smart Nation (IC3TSN)*, Oct 2017, pp. 168–172. 79
- [214] K. Kumar.V.G, A. Poojary, C. S. Rai, and H. R. Nagesh, “Implementation of Lightweight Cryptographic Algorithms in FPGA,” in *Proceedings of the 2017 International Conference on Circuits, Controls, and Communications (CCUBE)*, Dec 2017, pp. 232–235. 79
- [215] M. A. Philip and Vaithyanathan, “A Survey on Lightweight Ciphers for IoT Devices,” in *Proceedings of the 2017 International Conference on Technological Advancements in Power and Energy (TAP Energy)*, Dec 2017, pp. 1–4. 79
- [216] M. A. Mokhtar, S. N. Gobran, and E. A. El-Badawy, “Colored Image Encryption Algorithm Using DNA Code and Chaos Theory,” in *Proceedings of the 2014 International Conference on Computer and Communication Engineering*, Sep. 2014, pp. 12–15. 80
- [217] D. Upadhyay, T. Shah, and P. Sharma, “Cryptanalysis of Hardware based Stream Ciphers and Implementation of GSM Stream Cipher to Propose a Novel Approach for Designing n-bit LFSR Stream Cipher,” in *Proceedings of the IEEE International Symposium on 2015 19th International Symposium on VLSI Design and Test*, June 2015, pp. 1–6. 80, 89
- [218] A. Masmoudi, W. Puech, and M. S. Bouhlef, “A Generalized Continued Fraction-based Asynchronous Stream Cipher for Image Protection,” in *Proceedings of the 2009 17th European Signal Processing Conference*, Aug 2009, pp. 1829–1833. 80
- [219] S. C. Ramanna and P. Sarkar, “On Quantifying the Resistance of Concrete Hash Functions to Generic Multicollision Attacks,” *IEEE Transactions on Information Theory*, vol. 57, no. 7, pp. 4798–4816, July 2011. 81
- [220] L. Li, G. Liu, S. Xie, and L. Xue, “A Sufficient Condition for Constructing some XOR-based Iterative Target Collision Resistant Hash Functions,” in *Proceedings of the 2010 International Conference On Computer Design and Applications*, vol. 1, June 2010, pp. V1–168–V1–172. 81

- [221] M. Daubignard, P. Fouque, and Y. Lakhnech, “Generic Indifferentiability Proofs of Hash Designs,” in *Proceedings of the 2012 IEEE 25th Computer Security Foundations Symposium*, June 2012, pp. 340–353. 81
- [222] P. Luo, Y. Fei, L. Zhang, and A. A. Ding, “Differential Fault Analysis of SHA3-224 and SHA3-256,” in *Proceedings of the 2016 Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC)*, Aug 2016, pp. 4–15. 82
- [223] A. Bogdanov, G. Leander, C. Paar, A. Poschmann, M. Robshaw, and Y. Seurin, “Hash Functions and RFID Tags: Mind the Gap,” in *Proceedings of the Cryptographic Hardware and Embedded Systems – CHES 2008*. Springer, 2008, pp. 283–299. 82, 90
- [224] G. Bertoni, J. Daemen, M. Peeters, and G. Assche, “On the Indifferentiability of the Sponge Construction,” in *Lecture Notes in Computer Science – EUROCRYPT 2008*. Springer, 2008, pp. 181–197. 82
- [225] B. G. D. J. P. M., and V. A. G., “Sponge-Based Pseudo-Random Number Generators,” in *Proceedings of the Cryptographic Hardware and Embedded Systems – CHES 2010*. Springer, 2010. 82
- [226] A. Akhimullah and S. Hirose, “Lightweight Hashing Using Lesamnta-LW Compression Function Mode and MDP Domain Extension,” in *Proceedings of the 2016 Fourth International Symposium on Computing and Networking (CANDAR)*, Nov 2016, pp. 590–596. 82
- [227] P. Gauravaram, “Security Analysis of salt||password Hashes,” in *Proceedings of the 2012 IEEE International Conference on Advanced Computer Science Applications and Technologies (ACSAT)*, Nov 2012, pp. 25–30. 82
- [228] X. Wu, Z. Yang, C. Ling, and X. Xia, “Artificial-Noise-Aided Message Authentication Codes with Information-Theoretic Security,” *IEEE Transactions on Information Forensics and Security*, vol. 11, no. 6, pp. 1278–1290, June 2016. 83
- [229] S. R. Subramanya and B. K. Yi, “Digital Signatures,” *IEEE Potentials*, vol. 25, no. 2, pp. 5–8, March 2006. 83
- [230] A. B. Rabbiah, K. K. Ramakrishnan, E. Liri, and K. Kar, “A Lightweight Authentication and Key Exchange Protocol for IoT,” in *Proceedings of the Workshop on Decentralized IoT Security and Standards (DISS)*, San Diego, CA, USA, Feb. 2018. 83
- [231] S. A. Salami, J. Baek, K. Salah, and E. Damiani, “Lightweight Encryption for Smart Home,” in *Proceedings of the 2016 IEEE 11th International Conference on Availability, Reliability and Security (ARES)*, Aug 2016, pp. 382–388. 84

- [232] I. B. Dhaou, T. N. Gia, P. Liljeberg, and H. Tenhunen, “Low-latency Hardware Architecture for Cipher-based Message Authentication Code,” in *Proceedings of the 2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2017, pp. 1–4. 84
- [233] S. Shin, M. Kim, and T. Kwon, “Experimental Performance Analysis of Lightweight Block Ciphers and Message Authentication Codes for Wireless Sensor Networks,” *International Journal of Distributed Sensor Networks*, vol. 13, pp. 711–722, 2017. 84
- [234] F. D. Santis, A. Schauer, and G. Sigl, “ChaCha20-Poly1305 Authenticated Encryption for High-speed Embedded IoT Applications,” in *Proceedings of the Design, Automation Test in Europe Conference Exhibition (DATE)*, 2017, March 2017, pp. 692–697. 84
- [235] M. A. Simplicio, B. T. de Oliveira, P. S. L. M. Barreto, C. B. Margi, T. C. M. B. Carvalho, and M. Naslund, “Comparison of Authenticated-Encryption schemes in Wireless Sensor Networks,” in *Proceedings of the 2011 IEEE 36th Conference on Local Computer Networks*, Oct 2011, pp. 450–457. 84
- [236] S. Kumar, J. Haj-Yahya, and A. Chattopadhyay, “Efficient Hardware Accelerator for NORX Authenticated Encryption,” in *Proceedings of the 2018 IEEE International Symposium on Circuits and Systems (ISCAS)*, May 2018, pp. 1–5. 84, 85
- [237] L. Bossuet, N. Datta, C. Mancillas-López, and M. Nandi, “ELmD: A Pipelineable Authenticated Encryption and its Hardware Implementation,” *IEEE Transactions on Computers*, vol. 65, no. 11, pp. 3318–3331, Nov 2016. 84
- [238] U. M. Mbanaso and G. A. Chukwudebe, “Requirement Analysis of IoT Security in Distributed Systems,” in *Proceedings of the 2017 IEEE 3rd International Conference on Electro-Technology for National Development (NIGERCON)*, Nov 2017, pp. 777–781. 85
- [239] W. J. Buchanan, S. Li, and R. Asif, “Lightweight Cryptography Methods,” *Journal of Cyber Security Technology*, vol. 1, pp. 187–201, March 2018. 86
- [240] D. Schinianakis, “Alternative Security Options in the 5G and IoT Era,” *IEEE Circuits and Systems Magazine*, vol. 17, no. 4, pp. 6–28, Fourthquarter 2017. 86
- [241] D. Dinu, Y. Le Corre, D. Khovratovich, L. Perrin, J. Großschädl, and A. Biryukov, “Triathlon of Lightweight Block Ciphers for the Internet of Things,” *Journal of Cryptographic Engineering*, 07 2018. 86, 88, 112, 150
- [242] M. Sharafi, F. Fotouhi-Ghazvini, M. Shirali, and M. Ghassemian, “A Low Power Cryptography Solution Based on Chaos Theory in Wireless Sensor Nodes,” *IEEE Access*, vol. 7, pp. 8737–8753, 2019. 86, 88

- [243] D. Shin and H. Yoo, "The Heterogeneous Deep Neural Network Processor with a Non-von Neumann Architecture," *Proceedings of the IEEE*, pp. 1–16, 2019. 87, 91
- [244] L. Semeria, K. Sato, and G. D. Micheli, "Synthesis of Hardware Models in C With Pointers and Complex Data Structures," *IEEE TRANSACTIONS ON VERYLARGE SCALE INTEGRATION (VLSI) SYSTEMS*, vol. 9, no. 6, pp. 743–756, December 2001. 88
- [245] S. C. Kim, H. Kim, J. Song, and P. Mah, "A Dynamic Stack Allocating Method in Multi-Threaded Operating Systems for Wireless Sensor Network Platforms," in *Proceedings of the 2007 IEEE International Symposium on Consumer Electronics*, June 2007, pp. 1–6. 88
- [246] S. Kotel, F. Sbiaa, M. Zeghid, M. Machhout, A. Baganne, and R. Tourki, "Performance Evaluation and Design Considerations of Lightweight Block Cipher for Low-cost Embedded Devices," in *Proceedings of the 2016 IEEE/ACS 13th International Conference of Computer Systems and Applications (AICCSA)*, Nov 2016, pp. 1–7. 88, 150
- [247] S. Sen, J. Koo, and S. Bagchi, "TRIFECTA: Security, Energy Efficiency, and Communication Capacity Comparison for Wireless IoT Devices," *IEEE Internet Computing*, vol. 22, no. 1, pp. 74–81, Jan 2018. 89
- [248] H. Manifavas, G. Hatzivasilis, K. Fysarakis, and K. Rantos, "Lightweight Cryptography for Embedded Systems - A Comparative Analysis," in *Proceedings of the Data Privacy Management and Autonomous Spontaneous Security: 8th International Workshop, DPM 2013, and 6th International Workshop, SETOP 2013*, 09 2013. 90
- [249] R. Portella, "Balancing Energy, Security and Circuit Area in Lightweight Cryptographic Hardware Design," Ph.D. dissertation, PSL Research University, 2016. 90, 91, 92
- [250] U. Farooq, Z. Marrakchi, and H. Mehrez, "FPGA Architectures: An Overview," in *Tree-based Heterogeneous FPGA Architectures*. Springer, New York, NY, 2012, pp. 7–48. 91
- [251] R. Singh and A. Rajawat, "Accurate Area Estimation Model for FPGA Based Implementation," *IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)*, vol. 6, no. 4, pp. 26–32, Aug. 2016. 92
- [252] B. J. Baldwin, "Hardware Design of Cryptographic Accelerators," Ph.D. dissertation, University College Cork, 2013. 92
- [253] I. Kuon and J. Rose, "Measuring the Gap Between FPGAs and ASICs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 2, pp. 203–215, Feb 2007. 92

- [254] Y. A. Abbas, R. Jidin, N. Jami, and M. R. Z'aba, "Securing Electrical Substation's Wireless Messaging with a Lightweight Crypto-Algorithm IP Core," in *Proceedings of the 2014 IEEE International Conference on Power and Energy (PECon)*, Dec 2014, pp. 159–163. 92
- [255] G. Bansod, N. Raval, and N. Pisharoty, "Implementation of a New Lightweight Encryption Design for Embedded Security," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 1, pp. 142–151, Jan 2015. 92
- [256] R. Beaulieu, S. Treatman-Clark, D. Shors, B. Weeks, J. Smith, and L. Wingers, "The SIMON and SPECK Lightweight Block Ciphers," in *Proceedings of the 2015 52nd ACM/EDAC/IEEE Design Automation Conference (DAC)*, June 2015, pp. 1–6. 93
- [257] B. J. Mohd, T. Hayajneh, and A. V. Vasilakos, "A Survey on Lightweight Block Ciphers for Low-Resource Devices: Comparative Study and Open Issues," *Journal of Network and Computer Applications*, vol. 58, pp. 73 – 93, 2015. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1084804515002076> 93
- [258] M. Ammar, G. Russello, and B. Crispo, "Internet of Things: A Survey on the Security of IoT Frameworks," *Journal of Information Security and Applications*, vol. 38, pp. 8 – 27, 2018. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S2214212617302934> 95
- [259] S. Sallam and B. D. Beheshti, "A Survey on Lightweight Cryptographic Algorithms," in *Proceedings of the TENCON 2018 - 2018 IEEE Region 10 Conference*, Oct 2018, pp. 1784–1789. 150

Appendix A

Full-Length SBPG Result for Subject with Request ID B5555

Given the different constraints associated with IoT systems and the nature of their deployments, securing IoT systems against cyber threats is particularly difficult. Moreover, new cybersecurity threats that arise from IoT technology are increasing daily and cyber-criminals are constantly looking for new exploits and strategies to defraud organizations and individual IoT users. Hence, there is a need to ensure strong security for IoT and IIoT. For this reason, IoT makers and developers must be encouraged to adopt security best practices at the early stage of IoT product design. The term security best practices has already been defined and explained in Section 3.3.

Therefore, the purpose of this appendix is to present the full-length result or extended version of security best practice guidelines produced by the SBPG component tool of the IoT-HarPSecA framework mentioned in Section 5.1 and Subsection 5.3.2. While the author does not claim that the SBPG tool produces a holistic security best practice guidelines that can serve as a bulletproof IoT security baseline, adhering to these security best practices could enable IoT makers and developers to achieve a certain level of security controls that would reduce common IoT security risks. Figure A.1 presents the extended version of the result summary of the SBPG test for the subject with request ID B5555 which was presented in Figure 5.9 in Subsection 5.3.2. This SBPG test result is selected because it is considerably shorter than the others.

The Full-Length Best Practice Guidelines for Secure Development of IoT Systems for the User with Request ID Number: B5555

General IoT Security Best Practices

Internet of Things (IoT) devices can be divided into three main categories, namely:

- Sensors, responsible for gathering data
- Actuators, which effect actions
- Gateways, which act as communication hubs and may also implement some automation logic.

Each of these devices can be stand-alone or they are embedded in a larger product. These devices may also be complemented by a web application or mobile device app and cloud-based service. IoT devices, services, software, and the communication channels that connect them, are at risk of attack by a variety of malicious parties.

Each new IoT endpoint connected to a network constitutes a potential entry point for cybercriminals that must be protected. Therefore, each new IoT endpoint will need to be identified and added to the asset inventory, and such endpoints need to be monitored for their health and safety irrespective of their size. Malicious intent usually takes advantage of poor design, however, even unintentional leakage of data due to ineffective security controls can also bring significant consequences to consumers and organizations. Thus, it is very important to design IoT devices, smart apps, and services with security designed in mind.

By leveraging vulnerabilities in the IoT, cybercriminals can wreak havoc across the entire Internet by using DDoS to take down major sites, online services, gaming networks, and other important websites. Below are some important security best practices that can be employed to improve IoT security.

Authentication

Despite the challenges of interoperability between heterogeneous IoT platforms, device authentication is a necessity for most IoT use cases. Strong device authentication is necessary to ensure that connected devices can be trusted to be what the purport to be. Therefore, the ability to authenticate other devices, applications, and humans is a critical feature for IoT systems. More details on IoT authentication best practice guidelines are provided in the next section.

Data Classification

As IoT systems generate massive amounts of data, it is important to protect all types of data from growing threats across the IoT ecosystem while complying with privacy requirements. This is because the data may contain sensitive information, such as personally identifying information about employees or customers, social security numbers, financial account information like credit card details, and/or protected health information. A few data classification best practices listed below.

- Define a data classification scheme and document it;
- Assess every item of data stored, processed, transmitted or received by a device and apply a data classification rating to it;
- Ensure the security design protects every data item and collections of items against unauthorised viewing, changing or deletion, to at least its classification rating or higher.

Physical Security

IoT devices are liable to be tampered with by attackers in a way not intended by the manufacturer. The potential threat actors that can gain physical access and cause physical damage to IoT systems include disgruntled employees who, for some reason, want to harm the organization or an outsider who has managed to bypass insufficient physical security measures. Some important physical security best practice guidelines include the following.

- Any interface used for administration or test purposes during development should be removed from a production device, disabled or made physically inaccessible;
- All test access points on production units must be disabled or locked, for example by blowing on-chip fuses to disable JTAG;
- If a production device must have an administration port, ensure it has effective access controls, e.g. strong credential management, restricted ports, secure protocols etc;
- Make the device circuitry physically inaccessible to tampering, e.g. epoxy chips to circuit board, resin encapsulation, hiding data and address lines under these components etc, <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&number=4249896>, <https://www.abchimie.com/en/resins-potting/>;
- Provide secure protective casing and mounting options for deployment of devices in exposed locations;
- For high-security deployments, consider design measures such as active masking or shielding to protect against side-channel attacks.

Device Secure Boot

Arguably, IoT security starts with a secure boot because implementing a secure boot process is critical to IoT device integrity throughout its lifecycle. This is in view of the fact that a compromised boot process allows hackers to inject malware or entirely replace the firmware, and hence leaving the entirety of a connected system vulnerable. Additionally, a secure boot process also makes other security features possible by providing a necessary degree of trust. Thus, a secure boot process is critical to extending a root of trust throughout an entire system. Some IoT secure boot best practices are listed below.

- Where possible, ensure that the ROM-based secure boot function is always used. Use a multi-stage bootloader initiated by a minimal amount of read-only code;
- Where possible, use a hardware-based tamper-resistant capability (e.g. a microcontroller security subsystem, Secure Access Module (SAM) or Trusted Platform Module (TPM)) to store crucial data items and run the trusted authentication/cryptographic functions required for the boot process. Its limited secure storage capacity must hold the read-only first stage of the bootloader and all other data required to verify the authenticity of firmware;
- Ensure that each stage of boot code is valid and trusted immediately before running that code. Validating code immediately before its use can reduce the risk of attacks;
- At each stage of the boot sequence, wherever possible, check that only the expected hardware is present and matches the stage's configuration parameters;
- Do not boot the next stage of device functionality until the previous stage has been successfully booted;
- Ensure failures at any stage of the boot sequence fail gracefully into a secure state, to ensure no unauthorised access is gained to underlying systems, code or data. Any code run must have been previously authenticated;
- Always check for the existence of a new firmware update. If present, verify it is genuine, update the system to use the new firmware and restart the system;
- Verify all external services such as power supply, DNS, NTP etc. are operating correctly.

Secure Operating System

Many IoT devices are based on common operating systems that are incapable of addressing specialized security requirements. Consequently, there are many ways in which threat agents can infiltrate such operating systems. Hardening the operating system helps protect against this by using the latest software, removing all unnecessary access rights and functions, and limiting visibility of the system. Below are a few secure operating system best practices.

- Include in the operating system (OS) only those components (libraries, modules, packages etc.) that are required to support the functions of the device;

- Shipment should include the latest stable OS component versions available;
- Devices should be designed and shipped with the most secure configuration in place;
- Continue to update OS components to the latest stable versions throughout the lifetime of a deployed device;
- Disable all ports, protocols and services that are not used;
- Set permissions so users/applications cannot write to the root file system;
- If required, accounts for ordinary users/applications must have minimum access rights to perform the necessary functions. Separate administrator accounts (if required) will have greater rights of access. Do not run anything as root unless genuinely unavoidable;
- Ensure all files and directories are given the minimum access rights to perform the required functions;
- Consider implementing an encrypted file system.

Application Security

Software is the core of every IoT system and service, enabling its functionality and providing value-added features. Some examples of important aspects where smart applications provide essence to IoT include firmware of devices, implementations of communication protocols and stacks, operating systems for IoT products, Application Programming Interfaces (APIs) supporting interoperability and connectivity of different IoT services, IoT device drivers, backend IoT cloud and virtualization software, and software implementing different IoT service functionalities. The following are some smart apps security best practices.

- Applications must be operated at the lowest privilege level possible, not as root. Applications must only have access to those resources they need;
- Applications should be isolated from each other. For example, use sandboxing techniques such as virtual machines, containerisation, Secure Computing Mode (seccomp), etc;
- Ensure compliance with in-country data processing regulations;
- Ensure all errors are handled gracefully and any messages produced do not reveal any sensitive information;
- Never hard-code credentials into an application. Credentials must be stored separately in secure trusted storage and must be updateable in a way that ensures security is maintained;
- Remove all default user accounts and passwords;
- Use the most recent stable version of the operating system and libraries;
- Never deploy debug versions of code. The distribution should not include compilers, files containing developer comments, sample code, etc;
- Consider the impact on the application/system if network connectivity is lost. Aim to maintain normal functionality and security wherever possible.

Credential Management

To interact with IoT devices securely, communication protocols between devices need integrity and confidentiality protection, otherwise messages can be eavesdropped or modified in transit. However, one big challenge remains, namely, key protection, management, and storage solution. Security credentials have to be available on these devices for any communication security protocol to provide their service. Consequently, this leads to an additional requirement, a way to securely manage and store credentials in the IoT. A credential typically consists of keying material, algorithm-specific parameters, and a list of entities the credentials can be used with. Each credential also has an identifier associated with it and a lifetime. Some IoT Security best practices for protecting device credentials and secret keys include:

- Create unique credentials and crypto keys for all connected devices and keep those credentials cryptographically secure;
- Ensure mutual authentication of the network cloud management platform by the IoT device, and of the IoT device by the cloud management platform;
- Ensure that attackers cannot access credentials and private keys. This is the foundation of end-to-end IoT security;
- A device should be uniquely identifiable by means of a factory-set tamper resistant hardware identifier if possible;
- Use good password management techniques, for example no blank or simple passwords allowed, never send passwords across a network (wired or wireless) in clear text, and employ a secure password reset process;
- Each password stored for authenticating credentials must use an industry standard hash function, along with a unique salt value that is not obvious (for example, not a username);

- Store credentials or encryption keys in a Secure Access Module (SAM), Trusted Platform Module (TPM), Hardware Security Module (HSM) or trusted key store if possible;
- Aim to use 2-factor authentication for accessing sensitive data if possible;
- Ensure a trusted & reliable time source is available where authentication methods require this, e.g. for digital certificates;
- A certificate used to identify a device must be unique and only used to identify that one device. Do not reuse the certificate across multiple devices;
- A "factory reset" function must fully remove all user data/credentials stored on a device.

Network Connections

As IoT networks are becoming tempting prey for cybercriminals, hence securing an IoT infrastructure requires a rigorous security-in-depth strategy. This strategy requires securing data on devices, in the cloud, and protect data integrity while in transit over the public internet. A few network connection best practices include the following.

- Activate only those network interfaces that are required (wired, wireless - including Bluetooth etc.);
- Run only those services on the network that are required;
- Open up only those network ports that are required;
- Run a correctly configured software firewall on the device if possible;
- Always use secure protocols, e.g. HTTPS, SFTP;
- Never exchange credentials in clear text or over weak solutions such as HTTP Basic Authentication;
- Authenticate every incoming connection to ensure it comes from a legitimate source;
- Authenticate the destination before sending sensitive data.

Buffer Overflow Protection

Most IoT devices are constrained in terms of memory and computational capacity and their hardware is small in size. As a result, a lot of IoT source code tends to be written in C or C++. But these languages are particularly prone to buffer overflow vulnerabilities and memory leaks. A stack buffer overflow bug is caused when a program writes more data to a buffer located on the stack than what is actually allocated for that buffer. In most cases, this results in the corruption of adjacent data on the stack, which could lead to program crashes, incorrect operation, or security issues. An effective remedy is buffer overflow protection which checks accesses to each allocated block of memory and ensures that they do not go beyond the actually allocated space, and tagging. This ensures that memory allocated for storing data cannot contain executable code.

FireWalls

While commercial firewall solutions for the constrained IoT devices are expensive and not open-source [1], [2], [3], [4], the Netfilter can be used on the less constrained devices, such as the single board computers [5].

More details can be found in the following links:

<https://www.enisa.europa.eu/publications/good-practices-for-security-of-iot-1> <https://www.enisa.europa.eu/news/enisa-news/how-to-implement-security-by-design-for-iot> <https://www.enisa.europa.eu/publications/good-practices-for-security-of-iot> <https://www.iotsecurityfoundation.org/wp-content/uploads/2019/03/Best-Practice-Guides-Release-1.2.1.pdf>

https://www.owasp.org/index.php/IoT_Security_Guidance

https://www.owasp.org/index.php/IoT_Framework_Assessment

1. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6289292>
2. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6583680>
3. https://link.springer.com/chapter/10.1007/978-3-642-14478-3_42
4. <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=5940923>
5. <https://www.hindawi.com/journals/scn/2018/9291506/>

Authentication

Authentication

In the context of IoT, authentication is the process of verifying that a device, application, or an entity is who it claims to be. Authentication is commonly performed by submitting a username or ID and one or more items of private information that only a given device, app or user should know. Authentication is an important component of IoT security. The use of strong user authentication and access control mechanisms ensure that only authorized users can gain access to IoT networks, services, and data. Therefore, passwords must be sophisticated enough to resist educated guessing of passwords.

In addition, users and organizations should use two-factor authentication (2FA), which in addition to entering a password the system requires users to use another authentication factor such as a random code generated which can be via SMS text messaging or other means. Where applicable, a Role-based access control (RBAC) which restricts network access based on the roles of individual users can facilitate the authorization of data access within an IoT network. Based on the foregoing discussion, the following are general security best practices:

- IoT systems should adopt strong password where authentication is needed;
- Enforce the minimum length of passwords requirement (i.e., users should provide long enough passwords, a minimum of 8 to 10 characters), also enforce password complexity rules: at least 1 uppercase character, 1 lowercase character, 1 digit, and 1 special character;
- Where possible, IoT systems should include role-based access control for multi-user environments;
- There should be secured password recovery mechanisms in place;
- Where possible, IoT systems should implement two factor authentication;
- IoT systems should enforce the use of password expiration as well as periodic password change policy;
- At the initial setup stage, all IoT systems should enforce mandatory default password change;
- IoT systems should provide an option for changing privileged account username in addition to the option for changing privileged account password;
- There should be an account lockout mechanism to prevent against brute force attacks;
- Where possible, IoT systems should have mechanisms that can protect devices and smart apps against bots which can cause account lockout DoS attacks. An example of a security measure that can be used is a CAPTCHA (Completely Automated Public Turing test to tell Computers and Humans Apart).

User ID and session IDs

- Make sure your usernames/user IDs are case insensitive and unique;
- Rotate session IDs after a successful login;
- Don not expose the session ID on the URL.

User password

- Longer passwords provide a greater combination of characters and consequently make it more difficult for an attacker to guess;
- Minimum length of the passwords should be enforced by the application;
- Passwords shorter than 10 characters are considered to be weak;
- Maximum password length should not be set too low, as it will prevent users from creating passphrases;
- Typical maximum length is 128 characters;
- On IoT devices that need to be secured, think of forcing the user to change the password as part of the installation process.

Password Complexity

Applications should enforce password complexity rules to discourage easy to guess passwords. Password mechanisms should allow virtually any character the user can type to be part of their password, including the space character. Passwords should, obviously, be case sensitive in order to increase their complexity.

The password change mechanism should require a minimum level of complexity that makes sense for the application and its user population.

- Password must meet at least 3 out of the following 4 complexity rules;
- at least 1 uppercase character (A-Z);
- at least 1 lowercase character (a-z);
- at least 1 digit (0-9);
- at least 1 special character (punctuation) space included ;
- at least 10 characters;
- at most 128 characters;
- not more than 2 identical characters in a row (e.g. 111 not allowed).

Require Re-authentication for Sensitive Features

In order to mitigate Cross-Site Request Forgery (CSRF) and session hijacking, it is important to require the current credentials for an account before updating sensitive account information such as the user's password, user's email, or before sensitive transactions, such as shipping a purchase to a new address.

Authentication and Error Messages

Incorrectly implemented error messages in the case of authentication functionality can be used for the purposes of user ID and password enumeration. An application should respond (both HTTP and HTML) in a generic manner. An application should respond with a generic error message regardless of whether the user ID or password was incorrect. It should also give no indication to the status of an existing account.

Incorrect Response Examples :

- Login for User foo: invalid password;
- Login failed, invalid user ID;
- Login failed; account disabled;
- Login failed; this user is not active.

Correct Response Example :

- Login failed; Invalid userID or password

More details can be found in the following links:

<https://www.iotsecurityfoundation.org/best-practice-guide-articles/physical-security/>

<https://www.enisa.europa.eu/publications/good-practices-for-security-of-iot-1> <https://www.enisa.europa.eu/news/enisa-news/how-to-implement-security-by-design-for-iot>

https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Authentication_Cheat_Sheet.md

Logging and Error Handling

Logging

Logging data in IoT systems can provide a substantial benefit, especially since devices will be communicating with each other on a regular basis with little or no human intervention. For example, such a plethora of real-time data that IoT devices generated can be used for debugging and monitoring purposes. Some IoT logging best practices are listed below.

- Ensure all logged data comply with prevailing data protection regulations;
- Run the logging function in its own operating system process, separate from other functions;
- Store log files in their own partition, separate from other system files;
- Set log file maximum size and rotate logs.
- Where logging capacity is limited, just log start-up and shutdown parameters, login/access attempts and anything unexpected;
- Restrict access rights to log files to the minimum required to function;
- If logging to a central repository, send log data over a secure channel if the logs carry sensitive data and/or protection against tampering of logs must be assured;
- Implement log "levels" so that lightweight logging can be the standard approach, but with the option to run more detailed logging when required;
- Monitor and analyse logs regularly to extract valuable information and insight;
- Passwords and other secret information should not ever be displayed in logs.

Purpose of logging

Application logging should be always be included for security events. Application logs are invaluable data for:

- Identifying security incidents;
- Monitoring policy violations;
- Establishing baselines;
- Assisting non-repudiation controls;
- Providing information about problems and unusual conditions;
- Contributing additional application-specific data for incident investigation which is lacking in other log sources;
- Helping defend against vulnerability identification and exploitation through attack detection.

Each log entry needs to include sufficient information for the intended subsequent monitoring and analysis. It could be full content data, but is more likely to be an extract or just summary properties. The application logs must record "when, where, who and what" for each event.

Where to record event data

- When using the file system, it is preferable to use a separate partition than those used by the operating system, other application files and user generated content;
- For file-based logs, apply strict permissions concerning which users can access the directories, and the permissions of files within the directories;
- In web applications, the logs should not be exposed in web-accessible locations, and if done so, should have restricted access and be configured with a plain text MIME type (not HTML);
- When using a database, it is preferable to utilize a separate database account that is only used for writing log data and which has very restrictive database , table, function and command permissions;
- Use standard formats over secure protocols to record and send event data, or log files, to other systems e.g. Common Log File System (CLFS) or Common Event Format (CEF) over syslog; standard formats facilitate integration with centralised logging services.

Which events to log

- Input validation failures e.g. protocol violations, unacceptable encodings, invalid parameter names and values;
- Output validation failures e.g. database record set mismatch, invalid data encoding;
- Authentication successes and failures;
- Authorization (access control) failures;
- Session management failures e.g. cookie session identification value modification;
- Application errors and system events e.g. syntax and runtime errors, connectivity problems, performance issues, third party service error messages, file system errors, file upload virus detection, configuration changes;
- Application and related systems start-ups and shut-downs, and logging initialization (starting, stopping or pausing);

- Use of higher-risk functionality e.g. network connections, addition or deletion of users, changes to privileges, assigning users to tokens, adding or deleting tokens, use of systems administrative privileges, access by application administrators, all actions by users with administrative privileges, access to payment cardholder data, use of data encrypting keys, key changes, creation and deletion of system-level objects, data import and export including screen-based reports, submission of user-generated content - especially file uploads.

Data to exclude

- Application source code;
- Session identification values (consider replacing with a hashed value if needed to track session specific events);
- Access tokens;
- Sensitive personal data and some forms of personally identifiable information (PII) e.g. health, government identifiers, vulnerable people;
- Authentication passwords;
- Database connection strings;
- Encryption keys and other master secrets;
- Bank account or payment card holder data;
- Data of a higher security classification than the logging system is allowed to store;
- Commercially-sensitive information;
- Information it is illegal to collect in the relevant jurisdictions;
- Information a user has opted out of collection, or not consented to e.g. use of do not track, or where consent to collect has expired.

Error Handling

User Facing Error Messages

Error messages displayed to the user should not contain system, diagnostic or debug information.

Formatting Error Messages

Error messages are often logged to text files or files viewed within a web browser.

- text based log files: Ensure any newline characters (%0A%0C) are appropriately handled to prevent log forging;
- web based log files: Ensure any logged html characters are appropriately encoded to prevent XSS when viewing logs.

Recommended Error Handling Design

- Log necessary error data to a system log file;
- Display a generic error message to the user;
- If necessary provide an error code to the user which maps to the error data in the logfile. A user reporting an error can provide this code to help diagnose issue.

More details can be found in the following links:

https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Logging_Cheat_Sheet.md

Cryptography

When implementing security in constrained environments such as IoT, what usually comes to fore are memory requirements, circuit area, and energy drain of the primitive to be implemented. Therefore, standard cryptographic algorithms are usually prohibitively expensive for implementation in such environments, and hence the need for

lightweight cryptography algorithms. Lightweight cryptography can be defined as specialized cryptography tailored for devices that feature low computational capabilities, less memory and/or small area footprint. Lightweight cryptographic algorithms target a very wide variety of resource-constrained devices such as IoT end nodes and RFID tags, and like the conventional cryptographic algorithms, lightweight cryptography algorithms can be implemented on both hardware and software.

Encryption

Encryption is an effective mechanism used to provide confidentiality and maintain the integrity of the information. However, small size, limited computational capability, limited memory, and power resources of IoT devices make it difficult to use the resource-intensive traditional encryption algorithms for information security in the IoT. Nonetheless, lightweight cryptography enables the application of secure encryption on IoT devices with limited resources. Some lightweight encryption best practices are:

- Always apply the appropriate level of encryption commensurate with the classification of data being processed;
- Always use industry-standard cypher suites, use the strongest algorithms and always use the most recent version of an encryption protocol;
- When configuring a secure connection, if an encryption protocol offers a negotiable selection of algorithms, remove weaker options so they cannot be selected for use in a downgrade attack;
- Store encryption keys in a Secure Access Module (SAM), Trusted Platform Module (TPM), Hardware Security Module (HSM) or trusted key store if possible;
- Do not use insecure protocols, e.g. FTP, Telnet;
- It should be possible to securely replace encryption keys remotely;
- When implementing public/private key cryptography, use unique keys per device and avoid using global keys;
- A device's private key should be generated by that device or supplied by an associated secure credential solution, e.g. smart card. It should remain on that device and never be shared/visible to elsewhere.

Secure Cryptographic Storage Design:

- All protocols and algorithms for authentication and secure communication should be well vetted by the cryptographic community;
- Ensure that certificates are properly validated against the hostnames whom they are meant for;
- Avoid using wildcard certificates unless there is a business need for it;
- Maintain a cryptographic standard to ensure that the developer community knows about the approved ciphersuits for network security protocols, algorithms, permitted use, cryptoperiods and Key Management;
- Only store sensitive data that you need .

Lightweight cryptographic algorithms selection and implementation

- Do not use new and unproven lightweight cryptographic algorithms. Instead, use widely accepted algorithms and widely accepted implementations;
- Do not use broken lightweight cryptographic algorithms;
- To avoid implementation vulnerabilities, ensure that lightweight cryptographic algorithms are implemented properly and accurately ;
- Always determine the security requirements of an IoT system before selecting lightweight cryptographic algorithms;
- Consider the message payload size that the device will be sending/receiving;
- Consider the hardware specifications on which the candidate algorithm will run: important things to consider for software implementation includes hardware specifications (e.g., RAM size and flash memory size); the specifications to consider for hardware implementation includes circuit area, throughput (or number of slices and maximum operating frequency), and energy requirement;
- Based on a given use case, consider the desired security level (i.e., 32-bit, 64-bit, or 128-bit);
- Determine the security level, key sizes (and block sizes in case of a block cipher) of potential algorithms;
- Consider the energy requirements of potential algorithms, keeping in mind the power specifications of target hardware;

- If a password is being used to protect keys then the password strength should be sufficient for the strength of the keys it is protecting;
- Do not use ECB mode for encrypting lots of data (the other modes are better because they chain the blocks of data together to improve the data security).

Use strong random numbers

- Ensure that all random numbers, especially those used for cryptographic parameters (keys, IV's, MAC tags), random file names, random GUIDs, and random strings are generated in a cryptographically strong fashion;
- Ensure that random algorithms are seeded with sufficient entropy;
- Tools like NIST RNG Test tool can be used to comprehensively assess the quality of a Random Number Generator by reading e.g. 128MB of data from the RNG source and then assessing its randomness properties with the tool.

The following libraries are considered weak random numbers generators and should not be used:

C library: random(), rand(), use getrandom(2) instead

Java library: java.util.Random() instead use java.security.SecureRandom instead

For secure random number generation, refer to NIST SP 800-90A. CTR-DRBG, HASH-DRBG, HMAC-DRBG are recommended

More details can be found in the following links:

https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Cryptographic_Storage_Cheat_Sheet.md
<https://www.cryptrec.go.jp/report/cryptrec-gl-2003-2016en.pdf> <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8957116>

Figure A.1: Full-Length best practice guidelines for subject with request ID B5555.

Appendix B

Actual Screenshots of Request Data of Subjects as Stored in the MySQL Database

This appendix presents the actual screenshots of the three MySQL database tables of the SRE, SBPG, and LWCAR tools requests data presented in Tables 5.2-5.3, Tables 5.9-5.11, and Tables 5.22-5.24, and mentioned in Subsections 5.2.2, 5.3.2, and 5.4.2, respectively. It also presents the screenshots of the MySQL database tables of the SRE and LWCAR tools requests data for the six subjects mention in Subsection 5.4.2 and Section 5.5.

+ Options

Reqst_ID	state	Domain	anyUsr	anyUsrLogin	holdUsrInfo	storeAnyInfo	sensitivOfInfo	infoSent2E	connected	dataSent2Cloud
R1111	Off	City	No			Yes	Normal	Yes	Yes	Yes
R2345	Off	Toy	Yes	Yes	Yes	Yes	Normal	Yes	Yes	Yes
R4444	On	Home	Yes	Yes	Yes	No	Sensitive	Yes	Yes	Yes
R1234	Off	Wearable	Yes	Yes	Yes	Yes	Normal	Yes	Yes	Yes
R6548	Off	Pet	Yes	Yes	Yes	Yes	Normal	No	Yes	Yes
R0601	Off	Healthcare	Yes	Yes	Yes	Yes	Normal	Yes	Yes	No
R7788	Off	Healthcare	Yes	Yes	Yes	Yes	Sensitive	No	Yes	Yes
R5432	Off	Grid	Yes	Yes	Yes	Yes	Sensitive	Yes	Yes	No
R2278	Off	Healthcare	Yes	Yes	Yes	Yes	Sensitive	Yes	Yes	Yes
R9128	Off	Environmental	No			Yes	Critical	Yes	Yes	Yes
R6666	Off	AI	No			Yes	Normal	No	Yes	No
R1115	Off	Agriculture	Yes	Yes	Yes	Yes	Normal	Yes	Yes	Yes
R1995	Off	Toy	Yes	Yes	No	Yes	Normal	Yes	Yes	No
R4321	Off	Environmental	Yes	No	Yes	No	Sensitive	Yes	Yes	Yes
R1235	Off	Retail	Yes	Yes	No	Yes	Normal	Yes	Yes	Yes
R8374	Off	Healthcare	Yes	Yes	Yes	No	Sensitive	Yes	Yes	Yes
R4040	Off	Grid	Yes	Yes	No	Yes	Normal	No	Yes	No
R8789	Off	City	Yes	Yes	Yes	Yes	Critical	Yes	Yes	Yes
R1008	Off	Elderly	Yes	Yes	Yes	Yes	Critical	Yes	Yes	No
R5555	Off	Connected_Car	Yes	Yes	Yes	Yes	Normal	No	Yes	Yes
R1287	Off	Healthcare	Yes	Yes	Yes	Yes	Critical	Yes	Yes	No
R5461	Off	Grid	Yes	Yes	No	No			Yes	No
R0011	On	Home	No			Yes	Critical	Yes	Yes	No
R8126	Off	City	Yes	Yes	Yes	Yes	Sensitive	Yes	Yes	No

(a) SRE request data

dataStoredInDb	regulaUpdate	use3rdPrtySfw	possibitOfEvesdrop	possibitOfCapt_Resent	possibitOfimpersonatUsr	possibitOfPhysicalAcces
Yes	Yes	No	Yes	Yes		Yes
Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes	Yes	Yes	Yes	Yes	No	Yes
Yes	Yes	No	No	No	Yes	Yes
Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes	Yes	Yes	Yes	Yes	No	No
Yes	No	No	No	No	Yes	Yes
No	Yes	No	Yes	No	Yes	No
Yes	No	No	No	Yes	Yes	Yes
Yes	Yes	Yes	No	No		Yes
Yes	No	Yes	No	No		Yes
No	Yes	No	Yes	Yes	Yes	Yes
No	No	No	Yes	No	No	Yes
Yes	Yes	No	No	Yes	Yes	No
Yes	Yes	Yes	Yes	Yes	No	No
Yes	No	No	No	No	Yes	No
No	No	No	Yes	Yes	Yes	No
No	No	Yes	Yes	No	No	No
No	Yes	Yes	No	No	Yes	No
No	No	No	No	No	Yes	Yes
Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes	Yes	Yes	Yes	Yes	Yes	Yes
Yes	Yes	No	No	No		Yes
No	No	No	Yes	Yes	Yes	Yes

(b) Continuation of SRE request data

Figure B.1: Screenshots of SRE request data of subjects as stored in the database.

+ Options												
Request_ID	Status	Struct1	Struct2	Struct3	Struct4	Struct5	Struct6	Struct7	Struct8	Struct9	Struct10	Struct11
B1111	NotYet	Device_Mgmt	Data_Collect									
B2435	NotYet	Device_Mgmt	Data_Collect									
B4444	Exist		Data_Collect									
B1234	NotYet									Web		
B6548	NotYet						Data_Process	Analytics				
B0601	NotYet						Data_Process					
B7788	Exist				API_Service					Web		
B5432	NotYet									Web		
B2278	NotYet	Device_Mgmt			Data_Mgmt	Data_Process						
B9128	NotYet									Web		
B6666	NotYet	Device_Mgmt	Data_Collect					Analytics				
B1115	NotYet	Device_Mgmt	Data_Collect			Data_Mgmt	Data_Process	Analytics	Cloud_Service			
B1995	NotYet	Device_Mgmt										
B4321	NotYet	Device_Mgmt										
B1235	NotYet						Data_Process					
B8374	NotYet					Data_Mgmt						
B4040	NotYet	Device_Mgmt										
B8789	Exist									Web		
B1008	NotYet	Device_Mgmt										
B5555	NotYet	Device_Mgmt										
B1287	NotYet									Web		
B5461	NotYet						Data_Process					
B0011	Exist	Device_Mgmt										
B8126	NotYet									Web		

(a) SBPG request data

anyUsr	usrRegist	typeOfRegist	anyUsrLogin	usrInput	holdUsrInfo	storeAnyInfo	sensitivOfinfo	typeOfAUTH	useDb	typeOfDataStorg
No						Yes	Normal		Yes	SQL
Yes	Yes	Users	Yes	Yes	Yes	Yes	Normal	usernam_passw	Yes	SQL
Yes	No			Yes	Yes	No	Sensitive	usernam_passw	Yes	NoSQL
Yes	Yes	Admin	Yes	Yes	Yes	Yes	Critical	2Factor_AUTH	Yes	SQL
Yes	Yes	Admin	Yes	Yes	Yes	Yes	Sensitive	usernam_passw	Yes	SQL
Yes	Yes	Users	Yes	Yes	Yes	Yes	Normal	usernam_passw	Yes	SQL
Yes	Yes	Admin	Yes	Yes	Yes	Yes	Critical	2Factor_AUTH	Yes	SQL
No						Yes	Sensitive		Yes	SQL
Yes	Yes	Admin	Yes	Yes	Yes	Yes	Sensitive	usernam_passw	Yes	Distr_Storage
Yes	Yes	Users	Yes	No	Yes	No	Normal	SociNet_Email	Yes	SQL
No						Yes	Normal		Yes	SQL
Yes	Yes	Users	Yes	No	Yes	Yes	Normal	usernam_passw	Yes	SQL
No						Yes	Normal		Yes	Local_Storage
Yes	No			No	Yes	Yes	Sensitive	usernam_passw	Yes	Local_Storage
Yes	Yes	Users	Yes	Yes	Yes	Yes	Normal	usernam_passw	Yes	Distr_Storage
Yes	Yes	Users	Yes	No	Yes	No	Sensitive	MultiFact_AUTH	Yes	SQL
Yes	No			Yes	No	Yes	Normal	No_AUTH	No	
Yes	Yes	Users	Yes	Yes	No	Yes	Normal	usernam_passw	No	
Yes	Yes	Users	Yes	Yes	Yes	No	Normal	No_AUTH	Yes	SQL
No						Yes	Normal		No	
Yes	Yes	Users	Yes	Yes	No	No		No_AUTH	No	
Yes	Yes	Users	Yes	Yes	No	Yes	Sensitive	usernam_passw	Yes	SQL
Yes	Yes	Users	Yes	Yes	No	Yes	Normal	usernam_passw	No	
Yes	Yes	Users	Yes	Yes	No	Yes	Critical	usernam_passw	Yes	SQL

(b) Continuation of SBPG request data

typeOfDb	progmr1	progmr2	progmr3	progmr4	progmr5	progmr6	progmr7	progmr8	fileUpload	sysLog
MySQL						Python			No	Yes
SQLite						Python			Yes	Yes
SQL_Server				JavaSc		Python			Yes	Yes
MySQL		C/C++			PHP				No	Yes
MySQL					PHP	Python			Yes	Yes
PostgreSQL						Python			Yes	Yes
SQLite	C#								No	Yes
SQL_Server				JavaSc					Yes	Yes
SQL_Server		C/C++		JavaSc					Yes	No
PostgreSQL		C/C++		JavaSc		Python			Yes	Yes
SQL_Server						Python			No	Yes
MySQL						Python			Yes	Yes
MySQL		C/C++				Python			No	Yes
SQL_Server		C/C++				Python			No	Yes
SQL_Server	C#		Java						Yes	Yes
SQL_Server			Java						No	Yes
		C/C++							No	No
			Java						No	Yes
SQL_Server			Java						Yes	Yes
SQL_Server		C/C++							No	Yes
						Python			No	Yes
MySQL						Python			Yes	No
		C/C++							Yes	No
SQLite			Java	JavaSc					Yes	Yes

(c) Continuation of SBPG request data

Figure B.2: Screenshots of SBPG request data of subjects as stored in the database.

Request_ID	Hardware_type	CPU	FlashMem_size	Ram_size	Clock_speed	Applic_Domain	Payload_size	Req_1	Req_2	Req_3	Req_4	Req_5	Req_6
S0001	SBC	32	8000000.00	4096000.00	2000.00	Home	small	CONF	INTG		PRIV		
S1682	ARM	32	1024.00	96.00	72.00	City	small	CONF	INTG	AUTH			COAU
S3833	ELEC	32	1024.00	128.00	120.00	Healthcare	average		INTG				NONR
S4219	RL78	16	64.00	4.00	16.00	Retail	small	CONF	INTG	AUTH	PRIV		COAU
S4452	TENS	32	16384.00	520.00	240.00	Grid	small		INTG				NONR
S6000	AVR	8	136.00	6.00	16.00	Connected_Car	small		INTG	AUTH			
S6001	AVR	8	128.00	4.00	16.00	Pet	small		INTG				
S6789	SBC	64	32000000.00	2048000.00	900.00	Home	continuous	CONF			PRIV		NONR
S6868	SBC	64	8388608.00	1048576.00	900.00	Agriculture	small	CONF	INTG	AUTH	PRIV		NONR
S7788	PIC	32	256.00	1024.00	16.00	Home	small	CONF	INTG		PRIV		
S8888	SBC	64	8388608.00	1048576.00	1200.00	Financial	average	CONF	INTG	AUTH	PRIV		NONR

(a) Software implementation requests data

Request_ID	Hardware_type	Applic_Domain	Payload_size	Energy	Req_1	cct_area_1	tp_1	Req_2	cct_area_2	tp_2
H1598	FPGA	Home	small	Ultra-low	CONF	1022.0	100.0		0.0	0.0
H2648	FPGA	Healthcare	small	Ultra-low	CONF	578.0	230.0	INTG	734.0	158.0
H3456	ASIC	Connected_Car	continuous	Low	CONF	1450.0	200.0		0.0	0.0
H5942	ASIC	Agriculture	average	Low		0.0	0.0	INTG	757.0	1.0
H8791	ASIC	Home	small	Low		0.0	0.0	INTG	1254.0	2.3
H9850	ASIC	Home	small	Low	CONF	825.0	5.1	INTG	1398.0	16.0

(b) Hardware implementation requests data

Req_3	cct_area_3	tp_3	Req_4	cct_area_4	tp_4	Req_5	cct_area_5	tp_5	Req_6	cct_area_6	tp_6
AUTH	270.0	3.5		0.0	0.0		0.0	0.0	COAU	137.0	240.0
AUTH	356.0	280.0		0.0	0.0		0.0	0.0	COAU	415.0	310.0
	0.0	0.0	PRIV	1500.0	200.0		0.0	0.0	COAU	3150.0	4.3
	0.0	0.0		0.0	0.0		0.0	0.0	COAU	2879.0	4.9
	0.0	0.0		0.0	0.0		0.0	0.0	COAU	2863.0	4.9
AUTH	748.0	4.3		0.0	0.0		0.0	0.0	COAU	3200.0	5.8

(c) Continuation of hardware implementation requests data

Figure B.3: Software/hardware implementation requests data as stored in the database.

Reqst_ID	state	Domain	anyUsr	anyUsrLogin	holdUsrInfo	storeAnyInfo	sensitivOfInfo	infoSent2E	connected
R0718	Off	Retail	Yes	Yes	Yes	Yes	Sensitive	No	Yes
R1413	Off	Home	Yes	Yes	No	No			Yes
R2143	Off	Home	Yes	Yes	Yes	Yes	Sensitive	Yes	Yes
R2513	On	Grid	No	No	No	Yes	Normal	Yes	Yes
R4021	Off	Home	Yes	No	No	Yes	Critical	No	Yes
R6317	Off	City	No	Yes	Yes	Yes	Normal	Yes	Yes

(a) SRE request data

dataSent2Cloud	dataStoredInDb	regulaUpdate	use3rdPrtySfw	possibltOfEvesdrop	possibltOfCapt_Resent	possibltOfImpersonUsr	possibltOfPhysicalAcces
No	No	Yes	No	No	Yes	No	No
No	No	No	No	No	No	Yes	Yes
Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
No	No	No	No	No	No	Yes	No
No	No	No	No	Yes	No	Yes	No
No	Yes	Yes	Yes	No	No	Yes	Yes

(b) Continuation of SRE request data

Figure B.4: Screenshots of SRE request data for six subjects as stored in database.

	Request_ID	Hardware_type	CPU	FlashMem_size	Ram_size	Clock_speed	Applic_Domain	Payload_size	Req_1	Req_2	Req_3	Req_4	Req_5	Req_6
<input type="checkbox"/>	S0718	SBC	64	8000000.00	2048000.00	900.00	Retail	small	CONF	INTG	AUTH	PRIV	NONR	
<input type="checkbox"/>	S1413	ARM	32	1024.00	96.00	72.00	Home	average			AUTH		NONR	
<input type="checkbox"/>	S2143	AVR	8	128.00	4.00	16.00	Home	small	CONF	INTG	AUTH	PRIV	NONR	
<input type="checkbox"/>	S4021	SBC	64	8388608.00	1048576.00	1200.00	Home	small	CONF	INTG	AUTH	PRIV	NONR	

(a) Software implementation requests data

	Request_ID	Hardware_type	Applic_Domain	Payload_size	Energy	Req_1	cct_area_1	tp_1
<input type="checkbox"/>	H2513	FPGA	Grid	small	Low	CONF	722.0	2.0
<input type="checkbox"/>	H6317	ASIC	City	small	Ultra-low	CONF	820.0	4.1

(b) Hardware implementation requests data

Req_2	cct_area_2	tp_2	Req_3	cct_area_3	tp_3	Req_4	cct_area_4	tp_4	Req_5	cct_area_5	tp_5	Req_6	cct_area_6	tp_6
INTG	600.0	3.0	AUTH	847.0	3.5	PRIV	520.0	4.0		0.0	0.0	COAU	135.0	1.0
INTG	750.0	1.0	AUTH	36254.0	2541.0	PRIV	810.0	5.0		0.0	0.0	COAU	2860.0	5.0

(c) Continuation of hardware implementation requests data

Figure B.5: Software/hardware implementation requests data for six subjects.

Appendix C

Result Summaries of SBPG Test for Subjects with Request IDs B2278 and B7788

This appendix presents the result summaries of the SBPG tool test for the subjects with request IDs B2278 and B7788, which were discussed in Subsection 5.3.2.

 THE SUMMARY OF SECURITY BEST PRACTICES FOR USER WITH REQUEST ID No. : B2278

S/No	SECURITY BEST PRACTICES
1	There should be a strong access control policy that should limit unauthorized access to certain capabilities of the smart device or application.
2	Use input validation to ensure that an uploaded filename uses an expected extension type; also ensure that an uploaded file is not larger than the defined maximum file size.
3	Use positive (white-list) input validation to allow inputs that are considered valid. In addition, validate input length, format, characters, and range of numeric data.
4	Ensure secure boot, use tamper-resistant hardware-based storage like TPM, ensure that each stage of boot code is trusted before running it, and ensure that no boot sequence is skipped. Ensure that devices are shipped with latest and stable versions of OSes, and with proper OS security configuration. Also ensured that OSes can boot securely; use good password management techniques. Don't use the same keys when implementing encryption on many IoT devices; bake security into every stage of smart apps development lifecycle; and disable every port and interfaces that were installed on IoT devices for testing purposes.
5	Use parameterized database queries with bound, typed parameters, and parameterized stored proced- ures in the database. Avoid using shared database accounts between different web sites or appli- cations. Ensure that all software components such as libraries, plug-ins, and database server software are up-to-date with the latest security patches.
6	Enforce the minimum length of passwords requirement (i.e., users should provide long enough passwords, a minimum of 8 to 10 characters), also enforce password complexity rules: at least 1 uppercase character, 1 lowercase character, 1 digit, and 1 special character. To mitigate Cross-Site Request Forgery (CSRF) and session hijacking, require re-authentication using the current account credentials before updating sensitive account information like user password.
7	Do not use algorithms and protocols that are not vetted by the cryptographic community; ensure that certificates are properly validated against the hostnames whom they are meant for. To reduce the risk of compromising many servers, avoid using wildcard certificates unless there is a busin- less need for it. Store only the sensitive data that you need. If a password is being used to protect keys then the password strength should be sufficient for the strength of the keys it is protecting. Use cryptographically strong random numbers for cryptographic parameters like keys.
8	Ensure that IoT and IIoT data both in-flight and at-rest is encrypted, and be very careful when selecting and implementing cryptographic algorithms because a cryptographic algorithm is only as strong as how it is implemented. Avoid using insecure protocols such as File Transfer Protocol (FTP) and Telnet because of lack of encryption, and their reliance on clear-text usernames and passwords for authentication.
9	Ensure that smart devices that will be deployed in open environments are securely protected using strong casing. If possible, protect IoT device circuitry from tampering using resin encapsulation and epoxy resin, etc.

Press Enter to process the detailed version of best practices

Figure C.1: Final results of SBPG request for subject with request ID B2278.

 THE SUMMARY OF SECURITY BEST PRACTICES FOR USER WITH REQUEST ID No.: B7788

S/No	SECURITY BEST PRACTICES
1	There should be a strong access control policy that should limit unauthorized access to certain capabilities of the smart device or application.
2	API security should be strengthened by encrypting data using TLS, and by establishing trusted identities and control access to services/resources using tokens assigned to those identities.
3	Do not allow any untrusted inputs; do not permit any JavaScript code from an untrusted source to run; create an HTML sanitization engine for validating untrusted HTML input.
4	Use positive (white-list) input validation to allow inputs that are considered valid. In addition, validate input length, format, characters, and range of numeric data.
5	Ensure that any newline characters in system log files are appropriately handled to prevent log forging; and ensure that any logged HTML characters are appropriately encoded to prevent XSS when viewing logs.
6	Session ID should be long and random enough to prevent brute force attacks, and ID name should not provide unnecessary details about the purpose and meaning of the ID. Authenticated sessions should timeout after determined period of inactivity - 15 minutes is recommended.
7	Use parameterized database queries with bound, typed parameters, and parameterized stored procedures in the database. Avoid using shared database accounts between different web sites or applications. Ensure that all software components such as libraries, plug-ins, and database server software are up-to-date with the latest security patches.
8	All communication with and between web services containing sensitive features, an authenticated session, or transfer of sensitive data must be encrypted using well configured TLS. Use client certificate authentication using TLS. A web service should authorize its clients whether they have access to the method in question. Ensure that access to administration and management functions within the web service application is limited to web service administrators.
9	Enforce the minimum length of passwords requirement (i.e., users should provide long enough passwords, a minimum of 8 to 10 characters), also enforce password complexity rules: at least 1 uppercase character, 1 lowercase character, 1 digit, and 1 special character. To mitigate Cross-Site Request Forgery (CSRF) and session hijacking, require re-authentication using the current account credentials before updating sensitive account information like user password.
10	Do not use algorithms and protocols that are not vetted by the cryptographic community; ensure that certificates are properly validated against the hostnames whom they are meant for. To reduce the risk of compromising many servers, avoid using wildcard certificates unless there is a business need for it. Store only the sensitive data that you need. If a password is being used to protect keys then the password strength should be sufficient for the strength of the keys it is protecting. Use cryptographically strong random numbers for cryptographic parameters like keys.
11	Ensure that IoT and IIoT data both in-flight and at-rest is encrypted, and be very careful when selecting and implementing cryptographic algorithms because a cryptographic algorithm is only as strong as how it is implemented. Avoid using insecure protocols such as File Transfer Protocol (FTP) and Telnet because of lack of encryption, and their reliance on clear-text usernames and passwords for authentication.
12	Ensure that smart apps operate with the lowest level of privilege and not as root. They should also be given access to only the necessary resources needed for their normal operations.
13	Do not deploy debug versions of code and ensure that code comments, compilers and other superfluous files that can allow attackers to reverse engineer the code are not included.
14	Handle errors carefully and ensure that error logs and other messages do not reveal sensitive information.

Press Enter to process the detailed version of best practices

Figure C.2: Final results of SBPG request for subject with request ID B7788.

Appendix D

Maximum Number of Security Requirements and Full Report on LWCAR Result

This appendix shows the security requirements for the IoT system of the subject with request ID R2143, which represents the maximum number of security requirements that the SRE tool of the IoT-HarPsecA framework can generate, as discussed in Subsection 5.4.2. It also presents the full report on the LWCAR tool test result for the subject with request ID S2143 discussed in Subsection 5.4.2.

 THE SECURITY REQUIREMENTS FOR THE IoT SYSTEM OF THE USER WITH REQUEST ID No.: R2143

SECURITY REQUIREMENT	DESCRIPTION
Authentication	This is the assurance that a message is from the source it claims to be from.
Privacy	Refers to users control over the disclosure of their personal information, meaning that only the users should decide whether they want to share their data or not.
Confidentiality	This is the property that ensures that information is not disclosed or made available to any unauthorized entity.
Integrity	Is the property of safeguarding the correctness, consistency, and trustworthiness of data over its entire life cycle in an IoT system.
Availability	Refers to the property which ensures that an IoT device or system is accessible and usable upon demand by authorized entities.
Physical Security	Refers to the security measures designed to deny unauthorized physical access to IoT devices or systems, and to protect them from damage or tampering.
Authorization	Refers to the property that determines whether the user or device has rights/privileges to access a resource, or issue commands.
Forgery Resistance	This is the propriety that ensures that data shared between entities and updates cannot be forged by a third party trying to damage or harm the system or its users.
Non-Repudiation	Refers to the security property that ensures that the transfer of messages or credentials between 2 IoT entities is undeniable.
Confinement	Ensures that even if an entity is hijacked or corrupted, the spreading of the effects of the attack is as confined as possible.
Accountability	This is the property that ensures that every action can be traced back to a single user or device.
Reliability	Is the property that guarantees consistent intended behavior of an IoT system.
Counterfeit Resistance	Is the property that ensures effective validation of software such that any fake or maliciously modified software is rejected.
Data Freshness	Ensures that data is the most recent, and that old messages cannot be replayed.
Tamper Detection	Ensures all devices are physically secured, such that any tampering attempt is detected.

Press Enter to return to the Main Menu

Figure D.1: Security requirements for the IoT system of subject with request ID R2143

FINAL RESULTS FOR USER WITH REQUEST ID No.: S2143

YOUR SECURITY REQUIREMENTS AND RECOMMENDED SECURITY MECHANISMS AND SECURITY ALGORITHMS ARE:

SECURITY REQUIREMENT(S)	SECURITY MECHANISM(S)	SECURITY ALGORITHM(S)
Data Confidentiality/User Privacy	Encryption	SPECK64/96
Message Integrity	Hash Function	PHOTON-80/20/16
Authentication	Message Authentication Code	*No matching Algo found!
Non-repudiation	Digital Signature	*No matching Algo found!

*No algorithm matching the security requirement is found!

A DETAILED REPORT ON THE RESULTS OF THE REQUEST OF USER WITH REQUEST ID No.: S2143

This report consists of brief information about the recommended Lightweight Cryptographic Algorithms, and where applicable, it provides information on where to find the algorithms. The report also includes some security measures needed to be taken in order to meet the remaining security requirements that were generated by the Security Requirement Elicitation tool to which no security mechanisms and algorithms have been recommended.

THE RECOMMENDED ALGORITHMS

1. SPECK64/96:- SPECK is a family of lightweight block ciphers designed by the United States National Security Agency (NSA) to provide security in constrained environments where memory, storage space, and computational capabilities are limited. SPECK has been optimized for performance in software implementation. SPECK supports a variety of block and key sizes (i.e., block/key - 32/64, 48/72, 48/96, 64/96, 64/128, 96/96, 96/144, 128/128, 128/192, 128/256). Complete details about SPECK can be found at <https://eprint.iacr.org/2013/404.pdf>. Additionally, the C and Python implementations can be found at https://github.com/inmcm/Simon_Speck_Ciphers.

2. PHOTON-80/20/16:- PHOTON is a family of lightweight hash functions that come in five different flavors with the following digest sizes: 80, 128, 160, 224 and 256 bits. PHOTON can be represented in this format: PHOTON-n/r/r', where n is the output length in bits, r represents the input block length, and r' is the output block length. Thus, the five variants are PHOTON-80/20/16, PHOTON-128/16/16, PHOTON-160/36/36, PHOTON-224/32/32, and PHOTON-256/32/32. The lightweight hash function is suitable for extremely constrained devices such as passive RFID tags. Although it is optimized for hardware implementation, PHOTON can equally be implemented in software. More details about PHOTON can be found at <https://eprint.iacr.org/2011/609.pdf> and <https://pdfs.semanticscholar.org/63d1/7f64d7a7d5b1bcd199c2569334e7194e40e1.pdf>.

A SUGGESTION

* Authenticated Encryption:- Since your security requirements include confidentiality and/or privacy as well as integrity, you may consider returning to the main menu to select option 10 in order to modify your request by including 'Confidentiality & Authenticity' in your security requirements. This is because the mechanism that provides 'Confidentiality & Authenticity' is the authenticated encryption, which can provide message integrity and message origin authentication in addition to protecting data confidentiality and/or user privacy.

THE OTHER USER SECURITY REQUIREMENTS

1. Availability:- Refers to the property which ensures that an IoT device or system is accessible and usable upon demand by authorized entities. One way to ensure availability of IoT systems is to ensure that the systems run efficiently for a long time without breaking down, such as preventing power outages.

This often relies on routine maintenance of the devices and network, which includes regular updates, upgrades, and keeping the systems working reliably. Investing in cloud services for backup also ensures data availability.

2. Physical Security:- Refers to the security measures designed to deny unauthorized physical access to IoT devices or systems, and to protect them from damage or tampering. Where possible, the first step in providing physical security is to prevent unauthorized persons from having physical access to IoT systems. Another line of defense is to make the device casing extremely difficult to open, and if forcefully opened by a curious attacker, the device should be rendered permanently disabled. An important option is to remove every physical, radio, or optical ports that were installed for development purposes. Similarly, every unnecessary test point, such as pins and circuit tracks, should be removed or disabled.

3. Authorization:- Authorization is usually coupled with authentication, it refers to the property that determines whether a user or device has rights/privileges to access a resource, or to issue commands. While access control mechanisms are necessary and crucial for authorization in the IoT, there are no standardized open-source access control schemes for resource-constrained IoT devices.

4. Forgery Resistance:-This is the propriety that ensures that the data shared between entities cannot be forged by a third party trying to damage or harm the system or its users. Digital signature can provide forgery resistance, however, resource-constrained devices in the IoT may not be able to handle the computational overhead associated with digital signature due to the large real numbers required for the signature and the verification processes. Although many researchers have proposed lightweight digital signature schemes, implementing digital signature on resource-constrained devices is still a challenge. Another option is to use two-factor authentication, which provides a two-layer protection against unauthorized access.

5. Confinement:-Ensures that even if a device is attacked or corrupted, the effects of the attack is contained or confined only to the device. One way to limit the extent of damage resulting from an attack on IoT systems is to isolate IoT devices on a separate virtual LAN. This setup will prevent an attacker from monitoring the totality of network traffic, or launching attacks across the entire enterprise if one IoT device is compromised. The bottom line is this: you should not connect IoT devices on the same network with mission-critical systems.

6. Accountability:-This is the property that ensures that every action can be traced back to a single user or device. While Blockchain technology and emerging data provenance methods that track and record the flow of data end-to-end between components hold real promise, accountability in the IoT is still a major challenge. For more details on this subject, see <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8423131>.

7. Reliability:-Is the property that guarantees consistent intended behavior of an IoT system. A systematic approach to design that can help to ensure reliable IoT systems will entail incorporating some form of error or fault detection mechanisms. Such mechanisms can, if possible, correct the corruption or fault, or isolate the source of the error and report it to a recovery mechanism that can switch to a redundant replacement component or system. Regular testing, backing up, disaster recovery plans, and redundancy will help to improve the reliability of an IoT system.

8. Data Freshness:-Ensures that data is the most recent, and that old messages cannot be replayed. Data freshness is a critical requirement in the IoT, especially for critical automated decision-making processes (for example, in the smart healthcare and self-driving cars scenarios) where every millisecond counts. Nonetheless, ensuring data freshness in the IoT is still an active research topic. However, the ZigBee wireless technology that operates on top of the IEEE 802.15.4 standard provides data freshness in addition to authentication, message integrity, and encryption.

9. Counterfeit Resistance:-Is the property that ensures effective validation of software such that any fake or maliciously modified software is rejected. Software counterfeiting can be prevented by providing a means by which software can be validated. This is to ensure that third-party software with hidden vulnerabilities, as well as fake and maliciously modified software is detected and rejected.

10. Tamper Detection:-Ensures that every active attempt to compromise the integrity of an IoT system or the data associated with an IoT system is detected. The tamper detection design can be implemented using a suite of sensors, each designed to sense a particular physical penetration type. Other parameters that could be sensed include input voltage variations, input frequency variations, gamma rays, and x-rays. The detection of a threat may enable the device to initiate appropriate defensive actions, which may ultimately result in the deletion of any useful data.

WARNING! LIMITED RESOURCES

Implementing all algorithms may have negative impact on performance.

Press Enter to return to MAIN MENU

Figure D.2: A detailed report on the result of the request of subject with request ID S2143.