A Platform for Safer and Smarter Networks

Ibbad Hafeez

Master's Thesis
UNIVERSITY OF HELSINKI
Department of Computer Science

Helsinki, October 21, 2016

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Laitos — Institution — Department
Faculty of Science	Department of Computer Science

Tekijä — Författare — Author
Ibbad Hafeez
Työn nimi — Arbetets titel — Title
A Platform for Safer and Smarter Networks
Oppiaine — Läroämne — Subject
Computer Science

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidoantal — Number of pages
Master's Thesis	October 21, 2016	116 pages + 1 appendices

Tiivistelmä — Referat — Abstract

The number of devices connected to the Internet is growing exponentially. These devices include smartphones, tablets, workstations and Internet of Things devices, which offer a number of cost and time savings by automating routine tasks for the users. However, these devices also introduce a number of security and privacy concerns for the users. These devices are connected to small office/home-office (SOHO) and enterprise networks, where users have very little to no information about threats associated to these devices and how these devices can be managed properly to ensure user's privacy and data security. We proposed a new platform to automate the security and management of the networks providing connectivity to billions of connected devices. Our platform is low cost, scalable and easy to deploy system, which provides network security and management features as a service. It is consisted of two main components i.e. *Securebox* and *Security and Management Service* (SMS). *Securebox* is a newly designed Openflow enabled gateway residing in edge networks and is responsible for enforcing the security and management decisions provided by SMS. SMS runs a number of traffic analysis services to analyze user traffic on demand for Botnet, Spamnet, malware detection. SMS also supports to deploy on demand software based middleboxes for on demand analysis of user traffic in isolated environment. It handles the configuration update, load balancing and scalability of these middlebox deployments as well. In contrast to current state of the art, the proposed platform offloads the security and management tasks to an external entity, providing a number of advantages in terms of deployment, management, configuration updates and device security. We have tested this platform in real world scenarios. Evaluation results show that the platform can be efficiently deployed in traditional networks in an incremental manner. It also allows us to achieve similar user experience with security features embedded in the connectivity.

ACM Computing Classification System (CCS):

C.2.0 [**Computer-Communication Networks**]: General-Security and protection,
C.2.1 [**Computer-Communication Networks**]: Network communications and Wireless Communications,
C.2.3 [**Network Operations**]: Network management and Network monitoring,
C.4 [**Performance of Systems**]: Reliability, availability, and serviceability,

Avainsanat — Nyckelord — Keywords
Security, Privacy, IoT, SDN, cloud, small/medium/enterprise Networks, SOHO networks

Säilytyspaikka — Förvaringsställe — Where deposited
University of Helsinki E-Thesis: `https://ethesis.helsinki.fi`

Muita tietoja — Övriga uppgifter — Additional information

Contents

1 Introduction **1**
- 1.1 Internet of Things 1
- 1.2 Cybersecurity and IoT 3
  - 1.2.1 Data handling 4
  - 1.2.2 Cybersecurity in SOHO networks 5
  - 1.2.3 Cybersecurity in enterprise networks 6
- 1.3 Cyber security attacks in the wild 7
- 1.4 Improving cybersecurity 8
- 1.5 Network Management 9
  - 1.5.1 SOHO networks 10
  - 1.5.2 Enterprise networks 10
- 1.6 Overview 10

2 State of the Art **12**
- 2.1 Software-defined Networking 12
- 2.2 Related Work 13
  - 2.2.1 Home Network Security 13
  - 2.2.2 Home Network Management 15
  - 2.2.3 Software-defined Middleboxes 15
- 2.3 Commercial Solutions 17
  - 2.3.1 Google OnHub 17
  - 2.3.2 F-Secure Sense 18
  - 2.3.3 Qualcomm Smart Home Gateway 18
  - 2.3.4 Bitdefender Box 19
  - 2.3.5 Luma Wi-Fi Router 19
  - 2.3.6 Dojo Gateway 19
  - 2.3.7 Cujo Gateway 20
- 2.4 User Study 21
- 2.5 Open Questions 25

3 Securebox **27**
- 3.1 Design 27
  - 3.1.1 Portability 28
  - 3.1.2 Architecture 28
- 3.2 Policy Rules 30
- 3.3 Functioning 32
- 3.4 Implementation 35
  - 3.4.1 Hardware 35
  - 3.4.2 Software 36
- 3.5 Deployment Models 37
  - 3.5.1 Securebox as an AP 37
  - 3.5.2 Securebox as Super AP 38

3.5.3 Securebox as a Sensor 39

4 Security and Management Service **40**
4.1 Design 40
4.2 Functioning 42
4.3 Implementation 44
4.4 Deployment 46
4.4.1 Third Party Security Service Provider 47
4.4.2 ISP-based Deployment 47
4.4.3 Private Deployment 48
4.5 Policy Database Updates 48
4.6 Smartphone Application 49

5 Evaluation **51**
5.1 Latency 51
5.1.1 Web Browsing 53
5.1.2 VOIP Traffic 55
5.1.3 Skype VOIP Traffic 55
5.1.4 File Transfer Performance 57
5.1.5 Bittorrent Traffic 57
5.1.6 IPerf 58
5.1.7 Processing and Memory Overhead 58
5.2 Selective Isolation 59
5.3 Phishing Attack Prevention 60
5.4 Privacy 61
5.4.1 Metadata Sharing 62
5.4.2 Policy Database Updates 63
5.4.3 Privacy Supporting Deployment Models 64
5.4.4 Privacy-aware Data Sharing 65
5.5 Collaborative Approach for Network Security 65
5.6 Policy Database Updates 69
5.7 Quality of Service and Bandwidth Optimization 71
5.8 Dynamic Access Control 73
5.9 Cost Efficiency 74
5.10 Scalability 76
5.11 Fault Tolerance 77
5.12 Robustness 79
5.13 Infrastructure Security 79
5.14 Discussion 81

6 Features and Use Cases **83**
6.1 Device Discovery and Profiling 83
6.2 AP Management 84
6.3 Device Level Data Cap 85

6.4 Dynamic Traffic Analysis 85
6.5 Small and Home Office Networks 87
6.6 Enterprise Networks 89
6.7 Secure Wi-Fi Environments: Password free Wireless Networks with Trust Levels 90
6.7.1 Limitations of WPA 90
6.7.2 State of the Art 92
6.7.3 Proposed Solution 92
6.8 SWEN: Software-defined Wearable Networking 95
6.8.1 Design 95
6.8.2 Goals 95
6.8.3 Features 96
6.8.4 Deployment 97

7 Discussion 98
7.1 Limitations and Future Work 98
7.1.1 Latency 98
7.1.2 User Privacy 99
7.1.3 Fault Tolerance 99
7.1.4 Device Identification and Profiling 100

References 101

A Appendix 117
A.1 Libraries 117

1 Introduction

In recent times, the devices connected to our networks have become smarter but the underlying networks are using decades old approach for security and management to billions of these devices. Networking gear manufacturers and network managers have come up with solutions to deal with issues faced in network management, however, there is a need of improvement in network deployment architecture and technologies to deal with the existing challenges in terms of securely managing connected devices.

Recent advancements in technology have been the driving factor in the development of the new generation of smart portable devices including smart phones, smart watches, and tablet PCs to give some examples. Together they add up to more than 6.4 billion connected devices and this number is growing at a fast pace [107].

1.1 Internet of Things

Internet of Things (IoT) has recently gained huge popularity among consumers and estimates predict that there will be more than 20 billion connected devices by 2020 [106]. These devices promise to bring a number of time saving and comforting features to smart homes e.g. remotely opening door lock, checking ingredients from fridge etc. [21] IoT devices also promise to improve the industrial process automation, manufacturing and storage. Remote deployments of IoTs can be very useful in various sensing applications in marine, meteorology, seismic sciences etc. Various reports have estimated that IoT will add upto $10 – $15 trillion in the next decade with upto $6 trillion dollar spent in IoT infrastructure deployments in the next 5 years [68].

Figure 1 shows a typical smart home environment with a number of IoT connected devices. These devices collect the data from smart home and send this data to cloud-based applications, which provide different services to the users. These devices can be controlled using companion smartphone applications. The companion applications also allow users to access different functionalities offered by the web service collecting data from the IoT devices. Health monitoring and wearables are very common examples of IoT where the devices constantly collects the data about user's health e.g., heart rate, workouts, calories etc. This data can be accessed via smartphone applications and some web services provide suggestions to the user about improving their health, diet and workout plans [121]. New generation of wearables include connected clothes, connected shoes etc.

IoT devices typically consist of one or more sensors. These devices are designed to perform specific functions e.g., monitoring (preferably) using very few resources e.g., an IoT sensor running on battery power is expected to run for months before the battery dies. Due to the lack of resources, IoT devices typically run a very stripped down version of an operating system

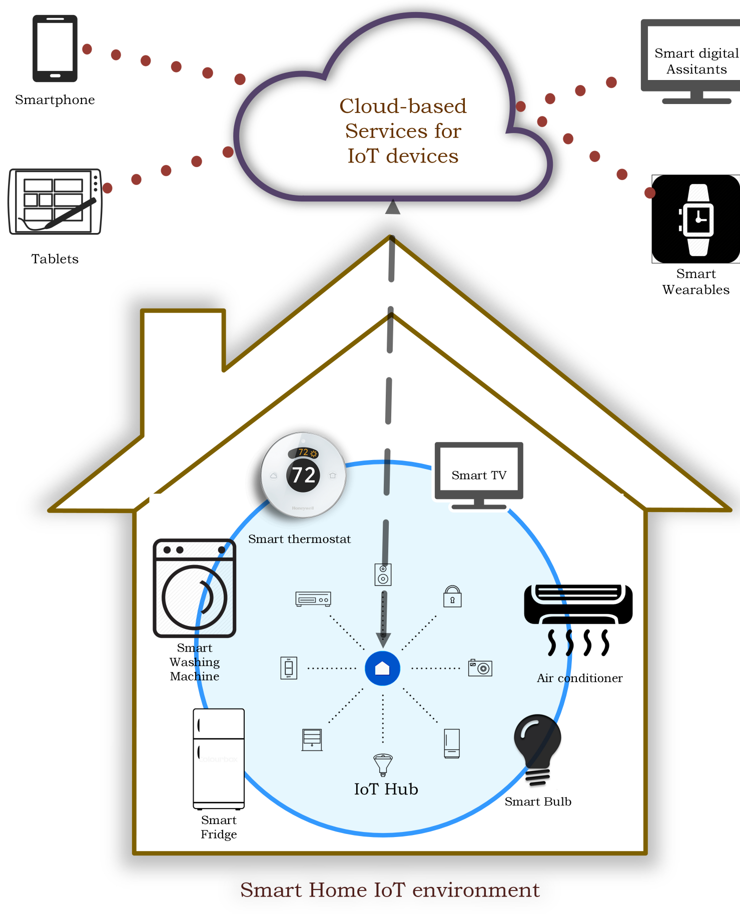


Figure 1: **Smart homes**. *A typical smart home environment has a number of connected devices collecting and sending user data to associated cloud based services. Users can control these devices via companion smart phone or web applications and data is used to provide suggestive services for users as well as improve device performance and automation.*

and in many cases they do not have an operating system at all. Due to limited hardware and energy sources, there is no *"Graphical User Interface"* (GUI) and very few other interfaces to communicate with the IoT device [6].

The function of IoT devices is mainly to collect data about their users and surroundings. IoT devices then send this data to services usually deployed in cloud environments, which in turn provide different kind of functionalities e.g., health monitoring, object tracking etc. [121, 130] These devices connect to these cloud-based services either directly or via an IoT hub. Since these devices do not have any resources to process or store this data locally, every IoT device requires constant connectivity for relaying the collected data to users or cloud based services.

IoT devices are generally developed by fast moving teams in large enterprises or independently working startup teams with limited resources. The development cycle for these devices is very tight with strict deadlines and the teams face a constant pressure of launching their products in the market as soon as possible (before any other manufacturer launches similar device) to get maximum customer base. Due to these constraints, there is little to no effort put into inherently securing device design and implementation of IoT devices [106].

Once the device is launched, there are no firmware updates or security patches made available for these devices. The diversity of manufacturers manufacturing IoT devices has made it harder to standardize the communication and development procedures for these devices. Lack of standardization further complicates development and support cycle. Since, IoT devices have a number of sensors constantly monitoring and collecting user related information, lack of secure design raises a number of security concerns for these devices [132, 131].

1.2 Cybersecurity and IoT

With the increasing number of connected devices, cyber security has become more important than ever. Large enterprises, governments and other institutions are spending more money in cyber security infrastructure than ever before. Studies have shown that the spending in cyber security has increased from \$3.5 billion (2005) to \$75 billion (2015) and is expected to increase upto \$170 billion by 2020. Careful predictions estimate upto \$1 trillion spent in cyber security in 2017-2021 period [68].

Every year cyber security causes \$350-500 billion losses out of which $\$150 - 160$ billion losses are suffered by individuals through credit card scams etc. [63] United States (US) and European union (EU) are frequent targets of these cyber crimes, which can cost more than 150000 jobs every year in each of these regions. With the growing popularity of IoT devices, cyber security has become a bigger problem than before and reports estimate the size of cyber security market will grow upto \$2 trillion dollars [67].

Security and privacy are important concerns for online users. With the recent popularity of e-commerce, cloud storage and cloud based services, network security and user privacy have become even more important. IoT and BYOD related security threats are fairly new to existing network security techniques and tools, which are mostly designed for large enterprise networks [80]. Therefore, we need to develop new techniques for securing these networks connecting large numbers of heterogeneous devices.

The cost of deploying and operating network security solutions, e.g., Firewall (FW), Deep Packet Inspection (DPI) is high. Therefore, these solutions are mainly adopted by large enterprises with sufficient resources to deploy and maintain them. Small enterprise and home users also need similar facilities, but do not have the resources. Our work in this article introduces the advantages of these sophisticated security and remote management solutions to all users with low cost.

1.2.1 Data handling

With all the possibilities and promises for smart future using IoT devices, there are some huge problems in terms of security. The biggest threats comes from the way IoT devices collect and manage user related data i.e. what kind of information is collected? How frequently is it collected? How is it stored? How and where is it processed? and a number of other questions. In normal practise, the best approach is to send only minimal data to web services [103, 108]. However, due to limited hardware, power resources and inefficient system design, most IoT devices upload all information collected from the users for *"just in case"* and *"future use"* purposes. Encryption is one of the possible solutions to protect user data. However, due to lack of power and hardware resources, nearly 70% of IoT devices do not encrypt their communications [105].

These approaches seriously affect the security and privacy of user's personal information. Typically, IoT devices are saved from many network attacks due to the presence of *"Network Address Translation"* (NAT) existing between user's internal network and the Internet. Also, there is little incentive in hacking IoT devices if they are few in number. However, both these incentives will soon be gone with deployment of `IPv6` across networks and ever increasing number of IoT devices in home and enterprises.

Smart phones and tablets also suffer from the same problem. These devices have a number of sensors and the applications can collect various kinds of information about the users to improve their services. If these services are breached, user's secret information including their identification and credit card information is accessible by rogue entities, causing serious security risks for the users.

Another important issue with IoT devices is the control system design. All data from IoT devices is either uploaded directly to cloud services or offloaded to IoT hub (via low power communication protocol) which then sends this data to the cloud services. In order for an attacker to get this data, he only needs to steal user's login credentials for cloud service or access to communication between IoT hub and the device. Snooping on device to hub communication is also an easy way to access user data because IoT devices do not encrypt these communications in most of the cases.

Stealing user credentials is also not very difficult for a moderately skilled attacker due to a number of loopholes in communication protocols being used [94, 33]. It is also known that average users do not make a serious effort for selecting a strong password and keeping it safe [66]. 80% of IoT devices do not force users to chose sufficiently strong and complex passwords [105]. With IoT devices, these passwords are going to become the key to user's home, bank accounts, health records etc. making the security issues with IoT security situation more complex and important.

1.2.2 Cybersecurity in SOHO networks

Small office and home office (SOHO) networks are a center piece to network security puzzle. These networks have a large number of connected devices. Gartner expects a typical home to have 500 connected devices by 2022 [35]. Home networks are typically the most insecure network deployments with no serious security mechanism to protect the connected devices. Most of the devices in home contain personal information about the user. Due to lack of security, these devices can easily be hijacked to compromise user privacy. With growing number of IoT devices, an attacker can cause a number of problems for a normal user, just by remotely controlling these devices e.g., playing inappropriate content on your smart TV or playing loud music at night to your connected speakers.

There are different kind of attacks happening on IoT devices and smart homes. Attackers mainly target home routers, setup boxes and IoT devices using factory default settings and security credentials. These devices can be used as agents for botnets, spam-nets, distributed denial of services (DDoS) attacks, Bitcoin mining etc. The compromised nodes can also be sold to adversary individual or agencies which can use them to spy on user activities or launch large scale ransom-ware, botnet and similar attacks [40, 50]. Some researchers have been able to trick IoT devices to spill out Wi-Fi passphrase of user network, giving them unwarranted access to all devices connected in the user network.

Wi-Fi based attacks are very crucial as these attacks does not require an attacker to physically trespass user premises to gain access to user devices. Recent research has shown that over 62.6% of home broadband networks use wireless connectivity for network setup [68] and this share is increasing. In typical cases, it is not difficult for an attacker to get the snoop Wi-Fi password [81, 119, 116]. Once attacker gets this password, it can connect to network and seamlessly communicate with other devices, possibly hacking or infecting them. There is no option to secure device to device (D2D) communications in Wi-Fi networks using typical gateways deployed in SOHO and IoT networks.

The purpose and methods of hacking are constantly evolving. Modern day hackers can use compromised IoT devices e.g., temperature, light sensors, electric meters etc. to find out whether a person is inside home or not. They can also hijack smart locks to ease break-ins without raising any alarms. Hackers can sell this information to burglars and help them carry out criminal activities more securely. Several news article have shown how burglars and thief are using technology to conduct their activities easily without alarming people around [126, 2].

Modern smart phones and IoT devices are equipped with a number of sensors and many of them are always-on. Therefore, an attacker can effectively use compromised devices to actively spy on user activities, movements [61, 1]. New generation of smart TV and virtual assistants e.g., Amazon Echo [28], Google Nest [124] etc. come with microphone and video cameras installed. An attacker can hijack these devices, using compromised user credentials or *"man-in-the-middle"* (MITM) etc. to get access to live audio and video feed from inside user home, which is a serious threat to user privacy [62].

1.2.3 Cybersecurity in enterprise networks

Enterprises are also expected to have large IoT installations for manufacturing, supply, storage units etc. IoT devices are used to improve automation in product development cycle. IoT sensors can be deployed across enterprises, sub offices and products to monitor the product functionality and detect any issues or faults. These IoT devices make it difficult for network management team to perfectly secure enterprise networks because on one hand they require connectivity to enterprise services but on the other hand, they can be physically accessed and used to breach in enterprise network.

A large installation of heterogeneous IoT devices from multiple vendors also makes it difficult to develop a uniform strategy for securing all these devices. Additionally, IoT devices do not provide inherent security neither do they allow users to install custom security applications e.g., anti-virus etc. There is insufficient authentication and authorization mechanism and insecure protocols are used for communication, making these devices easier to hack.

Enterprises are also joining *"Bring Your Own Device"* (BYOD) bandwagon, which allows employees to connect their insecure devices to enterprise network and use enterprise services to increase employee productivity. Previously, enterprise networks had tighter restrictions over what devices could be connected and it was easier to manage network but BYOD has made this task much complex. There are a number of heterogeneous devices used by employees and most of them are not secure because there can be malicious applications, malwares, trojans installed on these devices without user's knowledge.

Enterprise network managers and Chief Information Security Officers (CISOs) have been concerned over the way IoT and BYOD has changed the network security situation in enterprise and corporate networks. Majority of CISOs agree that IoT has made network management tasks more complex than it was before [80]. When a large number of unknown (employee's personal) mobile and IoT devices will be connected to enterprise networks, IoT cloud services will be collecting and processing a large amount of business critical data as well, which can lead to business losses for the enterprise.

1.3 Cyber security attacks in the wild

There have been various kinds of attacks happening affecting millions of connected users.

Hijacking home routers: Network attacks are a common place these days and their frequency is growing rapidly day by day [24]. Typically, attackers hijack home routers around the world and change their DNS servers to communicate with attacker controlled servers. Attackers can then redirect user traffic to malicious web pages for carrying out phishing, click fraud etc. attacks. Attackers can also use such hijacked networks and devices to use them for DDoS or botnets attacks. They can serve malicious web pages hosting malicious content in response of a legitimate request. This also leads to ransom-ware attacks which are very common these days.

It is common for attackers to login to home routers using manufacturer's default login credentials (e.g., username=*"admin"*, password=*"admin"*) as most of the users do not change login credentials after buying home gateways, routers or access points. Attackers can also use cross-site request forgery to gain access to local router's management interface, even it it is not exposed to the Internet.

***Hijacking setup boxes and NAS*:** Attacker have also targeted many device with embedded Linux such as DVR and NAS devices. These devices are mainly hacked for mining bitcoin and crypto-currency. Most of the attacks are launched as worms, where one infected device joins the network and infects other devices connected to the same network. With growing deployment of Wi-Fi networks, worms pose a serious threat to the devices.

***Webcam and CCTV hijacking*:** Webcam hijacking of laptops and other machines connected to home networks have become increasingly frequent these days. These hijacked webcams are used to record private videos of users which are later used for making ransom demands etc [78, 120]. Hijacking CCTVs also pose threat to home and enterprise security as a burglars can hack these cameras to assist them during burglaries. In later 2014, somebody setup a website showing live video stream from 4600 cameras around the world. The attacker scanned through Internet for connected cameras and used factory default username and password to get access to the live feed [59].

There are hundreds of thousands of cases where connected devices were hijacked by attacker or security researchers to show the extent of vulnerability and security risks associated to these devices [119, 81].

Network security is a big problem especially in SOHO networks because average users do not have the expertise nor the resource to manage the security of their networks. Typical network security equipment is expensive enough for a home or small business owner to use it for their network security.

Additionally, lack of awareness also leads to bad security practises for these users. As long as they are not first hand victim of hacking or ransom-ware etc., they do not consider the impact of these attacks. With so many IoT devices attached to user networks, these users (unknowingly) become part of online cyber crime as sophisticated hackers first hijack devices in Small and Home Office (SOHO) networks and use them to launch DDoS attacks against enterprise networks.

Recent research has shown that SOHO users are motivated for securing their networks and devices but it is too difficult for them to securely configure these devices due to the low level knobs available at typical (low-cost) network gateways [18]. Therefore, there is a constant demand for a solution which is easy to manage and operate for average users. One possible approach to secure these networks is to hire permanent experts or managers to securely operate the networks, but this is not feasible due to high costs of these experts.

A substitute approach requisites one expert managing more than one networks but neither is this approach scalable nor affordable. Additionally, this approach can bring a lot of inconsistencies in network configurations as each network can have a different set of requirements. Other approaches to make networks *"easier to manage"* still assumes that network will be managed by some dedicated managers or operators and SOHO networks still lack these dedicated managers.

1.4 Improving cybersecurity

One approach to deal with these issues is to release the burden of security and management of SOHO networks from their users by offloading the security mechanism to a third party service provider. Nick Feamster initially suggested this idea of a third party managing network security issues [31]. Such a system will allows users to operate their networks in *"plug and forget"* manner where they do not need to care about security and configuration updates for their network. For such a system, every network needs to have a programmable gateway which can constantly monitor the network and act accordingly. This gateway also collects network level statistics and user data to secure the traffic by filtering malicious flows in the network.

However, this approach will bring a number of new challenges [31]. Firstly, the programmable gateway should be smart enough to swiftly detect and block any malicious traffic flows in the network, requiring a robust mechanism for detection of such flows. The scale of data collected from these networks will be huge, containing lots of repetition and noise. The system should be efficient enough to filter out any data which negatively effects system analysis outcome. It needs smart algorithm and techniques to develop such systems. If the data is processing remotely, the system should be sensitive enough to immediately detect any anomaly and direct network devices to filter out the threat immediately. A lot of traffic we see is overlapping (due to flash crowds, viral content etc.), and analyzing this overlapping traffic from different source is a waste of resources.

Secondly, this approach raises a set of privacy challenges due to distributed data collection from SOHO networks, containing sensitive information e.g., user browsing history, login information etc. This information can be used to track user's online activities which raises some concerns for privacy advocates. Thirdly, the system should be able to heal itself i.e. a remote management service managing devices in SOHO networks should be able to correct the failures and misconfiguration it did previously. This feature would require a robust feedback loop in configuration update engine to learn and improve itself.

In this work, we look into the requirements from such a system which can provide network management and security as a service to SOHO and enterprise networks. We explored the challenges of building and deploying such a system in real world environment. We have also looked into various architectures for deploying such a system. Based on the literature survey and results of previous testing, we designed a system that is capable of offloading security and management tasks to a third party service provider. The service will work in a *plug and forget* manner where the network devices are managed via remote service without requiring significant user interaction required.

Our system offers a set of security and devices management services. It provides complete control to the users for managing the services used to manage and secure their networks. The users are able to classify their devices and individually decide what kind of services should be performed for each of these devices. The remote service will provide a set of services including network device management, user device management, user and device profiles, remote traffic analysis, scalable software defined middlebox deployments etc. We have also designed a smart gateway for networks which allows us to monitor and secure edge networks. This gateway also acts as a sensor which collects network meta-data and use it to improve overall network security across all connected networks.

1.5 Network Management

IoT, smart devices, personal computing devices need connectivity for their operations. An average user needs Internet access on all its devices for various reasons. Billions of these devices have already been connected to the networks which use the same technology and protocols from decades ago. However, looking forward to connect the next 30-40 billions devices to the same network definitely raises some concerns on how these networks will provide connectivity to all these devices and how will they coup with the management overhead.

New generation of networking gear comes with a number of features for

IoT connectivity and network management but they pace of improvement in network industry is slower than IoT and smart devices. Therefore, future networks deployment needs to be improved to provide efficient connectivity to these tens of billions of devices. As most of the devices use wireless connectivity, which already have a number of security issues, we need to the improve these networks to provide easily manageable inherent security to all connected devices as well [51].

1.5.1 SOHO networks

Networks in SOHO environments are poorly managed. These networks are setup using low cost routers or access points (AP) providing basic connectivity to all devices. The networking gear used in these networks do not offer enough control to the users over the device and network management. They provide basic security features e.g., MAC address filtering etc. The users of these networks mostly lack the expertise to use these low level knobs to perform any operation.

Since, SOHO networks connect the majority of users who suffer most individual losses in cyber crimes, they need to be well managed and well secured. These networks are expected to connect more than 50% of next 20 billion devices connected to the Internet. The users in this network also want a system that is easily manageable and is low cost to deploy and operate.

Typically, these users rely on the gateways provided by Internet Service Providers (ISP) to handle all the underlying functions and provide device connectivity. However, in order to easily manage and secure the users and devices connected to these networks, networking gear should support easy management by providing high level network information and function control knobs to the users.

1.5.2 Enterprise networks

Enterprises have large network deployments with a team of experts managing all these networks. With the popularity of IoT devices, the management of these networks has become troublesome [80]. Network managers express that the job of network management has become tougher then ever before because of the huge number of heterogeneous devices connected to enterprise networks. Also, the cost of deploying and managing enterprise setup is huge. Therefore, networks deployments should be improved to mitigate these cost and management issues.

1.6 Overview

In this report, we go through the state of the art from academic research to deal with these security and management issues in networks. Sect. 2 also discusses the latest products from industry to improve home gateways.

Sect. 3, 4 contains the details about design and architecture of Securebox and SMS respectively. We have developed a prototype for our system to evaluate its performance in real-world scenarios. Sect. 5 discusses this performance evaluation in detail for a number of factors. Sect. 6 present a number of use cases for our proposed system and explains how it can be advantageous to use this platform for future network deployments in different environments. We make a conclusive statement about this work Sect. 7. Sect. 7 also highlights some limitations in this work and gives a heading to lead any future works.

2 State of the Art

Improving network security has always been an active field in academic research. There have been a number of security protocols (i.e. supporting encryption etc.) proposed to achieve secure network communications. *Software defined Networking* (SDN) research has become popular in the last decade, opening new horizons for networking applications.

Researchers have also proposed several solutions for virutalizing middlebox functionality using software defined middleboxes for security and traffic analysis operations. These middleboxes can be deployed remotely and SDN is used to dynamically re-route traffic through them. We discuss all these research ideas and their applications in the following sections.

2.1 Software-defined Networking

Software-defined Networking (SDN) was initially proposed by Casado et al. in 2007 [16]. Their work was motivated by the complexities and difficulties faced in managing traditional computer networks. The closed nature of the traditional networks equipment makes the process of updating configurations slow, complex and expensive in terms of deployment and operational costs [32].

Casado et.al envisioned programmable switches and routers where routing and switching mechanism will become two separate planes i.e. *"control plane"* and *"data plane"*. *Control plane* is responsible for deciding how to handle the network traffic and *Data plane* is responsible for forwarding the traffic according to the routing/switching decisions made by control plane. SDN architecture consists of a controller (managing control plane operations) managing one or more switches (managing data plane operations).

Due to its flexible and programmable architecture, SDN has gained popularity in recent times. Many vendors have started supporting OpenFlow APIs in their network equipments. A number of controllers have been developed by both research community and industrial suppliers [29, 74, 42, 58, 114].

SDN promises to change the way traditional networks are managed by offering a flexible model that supports dynamic reconfiguration of network [16, 32]. Traditionally, SDN has been used in data center environments for traffic management in wide area networks (WANs) and virtualization platforms [55, 113]. However, programmers and researchers have used these controllers to develop different kind of network applications e.g. dynamic quality of service (QoS), access control [16, 71], load balancing [49, 118], security [83] etc. We believe that SDN can provide better security and remote management capabilities to SOHO networks. Previous research has also showcased some techniques for using SDN for dynamic re-routing of traffic through middleboxes deployed outside the network [37, 88, 99].

SDN has promised to revolutionize networking as we knew it and it has been successful in doing that by enabling interesting applications for network devices. However, there are some limitations to SDN design as well. SDN centralizes the control point of the network, which raises concerns of performance bottlenecks and attacks against SDN itself. Various studies have been performed on the resilience of this architecture against attacks and performance losses [12, 75, 77].

In this work, we intend to design a smart programmable gateway for networks. This gateway should be capable of monitoring the network and filtering out malicious traffic dynamically. SDN is a useful candidate to built such a gateway since it allows easy programmable interfaces to control the routing and switching in the network. The use of SDN does incur a performance penalty to the system, however, the advantages of the programmable interfaces are vastly beneficial.

Sect. 3.1 explains our gateway design, which uses SDN to dynamically filter and steer traffic in user network. (To the best of our knowledge) Our system is the first attempt to realize the utilization of SDN in SOHO networks. Previously, there have been several design proposals and prototypes to advocate the use of SDN in home networks [31] but we have realized a system, which uses service based model and SDN for automation of network security dynamics in a network. SDN provides a number of features including flexibility in programming the network inclemently and revoke any misconfiguration in the setup.

2.2 Related Work

Previously network security systems were designed for large enterprise and corporate networks as those customers have enough resources to invest in network security infrastructure and employ a team of professionals for managing their systems. With growth in network coverage areas and advancements in communication technologies e.g. 3G, 4G, LTE etc., more people are connected to Internet in the last decade compared to the total period before that. A vast majority of these connected devices are connected to the Internet via SOHO networks. The fact that these numbers will increase exponentially with the growing popularity of IoT devices, put SOHO networks in the center of the whole network security picture.

2.2.1 Home Network Security

As discussed before, SOHO networks are insecure and poorly managed. Similarly, IoT and BYOD trends are also compromising the security of enterprise networks as well. Therefore, recent research has been focused on how to secure these network with billions of connected devices. Nick Feamster initially highlighted the problem of smart home security and proposed that a system that outsources home network security to an outside entity [31].

Yiakoumis et al. also looked into the problem of security in home networks and their inflexibility for management [125]. They proposed a scheme to slice home network to deal with the common issues. Their scheme provides bandwidth and traffic isolation control, individual management, and ability to improve or modify the behavior of each slice.

Kim et al. proposed `Lithium` which provides event driven control for the home networks [57]. Lithium implements network policies based on four domains: time, user, history and traffic flow. It allows operators to define high level policies for the network and then translates these policies to low level configuration changes to achieve desired functionality in the network.

Xu et al. have developed a system to capture and analyze home network traffic [123]. They have captured real-world traffic traces from home networks, characterize it and apply *"principal component analysis"* (PCA) to understand temporal correlation between application ports. Using this system, they were able to identify the sources of unwanted traffic in typical home networks. This solution needs alot of training data and does not prevent from snooping or phishing attacks.

Tialong et al. have identified the key challenges in network security due to growing number of connected (IoT) devices [128]. They have argued that these networks need to be secured to ensure user data privacy and security. They have developed a system which monitors the home networks to identify any vulnerabilities. It sends this information to an IoT security service which directs the network device to take necessary actions to secure the network.

Zachariah et al. identified common issues with existing IoT gateway and advocate that these issues hinder efficient IoT deployed [129]. Therefore, they have proposed a system that can leverage Bluetooth (BLE) [47] connectivity to connect IoT device to the Internet. They use smart phones as a gateway for IoT devices to provide universal access to the Internet for BLE enabled devices. These smartphone act as BLE proxy for IoT devices.

However, one possible issue with this architecture is that it cannot provide all-the-time connectivity to the devices which are permanently installed e.g. temperature sensors etc. Secondly, it requires explicit permissions from smartphone owners to allow IoT devices to piggyback their networks, which may end up consuming reasonable amount of user data packages. Also, IoT devices may need to actively look for smartphones which may allow them IoT devices to use them as proxy to connect to Internet. With limited power resources, this active probing will result in battery drains. Additionally, IoT devices may need to upload sensitive information but a malicious user himself or a rogue application on the smart phone may steal this data, leading to privacy concerns. Sect. 6.8 presents an improved system architecture which can resolve a number of these issues.

2.2.2 Home Network Management

A majority of users in SOHO as well as enterprise networks do not have enough expertise and knowledge to manage their networks. They directly plug their device to the access point and do not put special efforts in security and optimization of their network. In order to improve and automate the management of these networks, there has been some research in recent decade.

Gharakheili et al. have proposed the use of cloud-based SDN deployment for managing home networks [38]. They have developed an architecture which allows remote SDN controller to manage and prioritize devices in user network. System prototype evaluation has shown that this architecture can improve user experience. However, the efficiency of this solution is greatly limited by the load on cloud-based SDN controller. Since, all the clients have to go through remote SDN portal which would affect user experience. Also, caching can be helpful in this work as it allows minimize latency for two similar requests.

Chetty et al. have developed a system which allows user to monitor traffic usage of their individual devices [18]. Their system uses an intelligent gateway which requires monitors the data usage on device level granularity. This system is useful for customer who have strict data usage limits on their internet packages. It also allows them to detect any malicious activities originating from devices, by observing the anomalies in device's data usage.

Bozkurt and Benson have developed a contextual router for home networks which improves the traffic prioritization in home networks [9]. This work includes the design goals for a management framework which optimizes network utility in networks with multiple network devices. This optimization can improve the page load times, reduce buffering events for various application, improving overall QoE.

SpaceHub is proposed by Meng et al. as an improved relay node for providing better connectivty and coverage to all devices in a smart home [64]. Spacehub listens to the Wi-Fi signals in surrounding environment and separates collided signals from the clean signals. It then relays the separated signals to intended destination without any prior knowledge [64].

2.2.3 Software-defined Middleboxes

With the advancements in SDN, extensive research has been done in virtualization of middleboxes [91]. Traditional middleboxes are specialized hardware equipments which need to be installed over the line in networks. These middleboxes are expensive to deploy and maintain and require manual configuration updates to block new threats. They also need to be periodically updated to increase the bandwidth and processing power. However, the can still become bottlenecks in many scenarios e.g. flash crowds.

Sherry et al. have initially proposed the idea of remotely deployed middleboxes for traffic anlaysis [100]. They claimed that using cloud resources to deploy middleboxes reduces the cost of deployment and improves the scalability. The have demonstrated that this concept can be useful for enterprise and corporate networks where middleboxes take a huge share of investment for network infrastructure. Gember-Jackobson et al. have also developed a implemented a framework for software-defined middlebox networking [36]. Deidtect, developed by Shanmugam et al., uses SDN for dynamic traffic steering through remotely deployed middleboxes [99]. This works envisions a system which can perform dynamic re-routing of user traffic through middleboxes on demand.

Yu et al. have proposed a system which allows users to remotely request other network to process their traffic to mitigate any threats [127]. It provides API to the user which allow him to direct other networks to analyze its traffic before delivering it to user network. They envision a scenario where ISPs provide some interfaces using which a user can direct the ISP to handle its traffic differently e.g. if a user finds some anomaly in its network, it can direct the ISP to process its traffic via a set of middleboxes before sending it to user network [5].

One of the key issues with this technique is that it requires user to actively monitor its network traffic and understand whether or not it is a security threat. Typical users are unable to perform this kind of monitoring and decision making for their network traffic. One possible approach is to automate the detection of malicious traffic in user network. Secondly, this system supposes that ISPs will provide interfaces to users allowing them to manipulate how their traffic is handled by ISP. This will be a strict requirement as ISPs have specially optimized internal traffic handling mechanism to provide best QoS and quality of experience (QoE) to their users.

Sherry et al. have also developed a system for performing DPI over encrypted traffic [101]. DPI is used by a number of middleboxes for analyzing network traffic. Recent efforts to improve privacy and security for users has led to most of the websites and Internet based services using encryption as default standard for communication. New generation of protocols e.g. HTTPS, TLS etc. encrypt all payload information during transit, therefore, making it harder to understand the origin of packet and what data is being communicated. It hinders the ability of most security system which analyze payload data to identify any malicious data served as response to user request. However, the method proposed by Sherry et al. uses a novel protocol and encryption scheme to realize DPI over encrypted traffic. This technique is applicable for long-lived HTTPS connections and its encryption schemes perform an order of magnitude better than existing cryptographic schemes [101].

Bremler-barr et al. have moved proposed architecture for a system which provides DPI as a service [10]. This technique scans the traffic only once and use it for all middleboxes by passing relevant results to corresponding middleboxes. This approach improves the scalability and robustness of middleboxes [10]. SDN can also benefits this scheme by allowing dynamic traffic steering between middleboxes for efficiently sharing scan results from DPI service.

Qazi et al. have proposed a framework `SIMPLE`, which uses SDN for dynamic traffic steering through set of middleboxes [88]. This system combines the advantages from middleboxes functioning using information from L4-L7 and combines it with L2/L3 functionality supported by SDN. SIMPLE shows that using the statistics from middlebox, SDN can very well integrate with current networking setup [88].

2.3 Commercial Solutions

Recently, a number of products have been launched to improve the user experience and security in smart home scenarios. These devices are developed to improve usability experience by providing companion smart phone applications. Some of these devices use public cloud services to tunnel all traffic through security services (similar to a VPN). The security service performs traffic analysis, destination filtering and anti virus protection for securing user traffic. We discuss some of these devices and their features in this section.

2.3.1 Google OnHub

Second iteration of Google *"onHub"* router was launched by Google in 2015 [39]. *Onhub* is designed to be a faster and stylish Wi-Fi router for home and office environments. Google *onHub* uses an array of directional antennas to ensure maximum coverage across entire home and office to support better data rates and bandwidth for all connected devices.

OnHub is especially designed to become part of interior decor so that it is not hidden behind the desk etc., as physical objects greatly affect the Wi-Fi coverage. *OnHub* is also equipped with `ZigBee` [92] antennas to support IoT connectivity. It also includes a microphone which allow users to setup voice commands for different device operations.

Google *OnHub* comes with a companion smart phone application which enables easy setup, monitoring and setting priorities for connected devices. This application is also connected to Google cloud for easy updates and notifications. *OnHub* is primarily designed for providing better coverage in home environments and does not offer any security related features as of now. The functionality of companion application is fairly limited to generating notifications and monitoring device status. With inclusion of Nest, Weave [19] and other IoT support, *OnHub* is expected to improve its support functionality as an IoT hub. Since, *Onhub* is mainly designed to provide improved connectivity across home, there are limited security features supported by the device, as of yet.

2.3.2 F-Secure Sense

Sense was initially launched by F-Secure in 2015 [30] and is currently in pre-order stages. F-Secure Sense is designed to be an improved gateway, which is convenient to setup and provide some security functionality as well. It does so by creating a secure network for all devices to connect and performs constant monitoring to detecting any threats in the network. Since the device is in developments stages, the final set of features is not available as of yet.

Sense uses a subscription based model for updating the endpoint and services. There is a companion mobile application available for improving user experience to control the device. It is expected to perform traffic analysis (if any) operations on the Sense device itself to protect user privacy. However, end point based analysis may limit the scope of analysis operations due to hardware constraints.

2.3.3 Qualcomm Smart Home Gateway

Qualcomm released their smart home gateway platform, which uses a Qualcomm Internet Processor (IPQ) to enable a robust smart home gateway. This gateway includes IPQ processor, Gigabit Wi-Fi from Qualcomm VIVE `802.11ac`, and Qualcomm StreamBoost technology to enhance user experience in connected smart homes [89]. The smart home gateway platform acts as always-on channel for carriers and digital content providers to support new applications and services including data, voice and video services. This gateway uses IPQ processing power to manage different complex and demanding applications. It improves network bandwidth management and provide useful analytics for application optimization. IPQ also enables gateway manufacturers to optimize content delivery and content caching on the edge.

Qualcomm smart home gateway also provides parental and access control security features for protecting the traffic inside user network. The gateway platform also enables third party to optimize their application performance for end users. However, these gateways are focused on the applications enabled by processing power available via IPQ processors. They do not provide any traffic analysis or other security features as of yet. Currently, there is no information or control available to the user about what processing is being done, what information from user network is being shared to third parties.

2.3.4 Bitdefender Box

Bitdefender BOX is a network security device for smart home from Bitdefender. This solution is a combination of hardware and cloud services to protect all user devices. Box can be installed in the home or carried along by the users to protect all their connected devices. It sets up a private line with Bitdefender cloud to secure user traffic by processing it through cloud services. `Box` can also be carried around by the user to get security connectivity *"on the go". Box* connects to available networks in open Wi-Fi environments maintains a private line to Bitdefender cloud (like a VPN) to securely channel all user traffic to the Internet.

Box promises to perform vulnerability assessment of user network by scanning the network and finding any connected devices which can lead to data theft or other malicious attacks. It also provides complete security for all device communications by routing it via Bitdefender cloud services. These services also notify the user if there is any malicious activity detected during the traffic analysis.

2.3.5 Luma Wi-Fi Router

Luma is a redesigned Wi-Fi router which is easy to install and configure for normal users [110]. It is designed to provide built-in security and content filtering services for IoT and other devices in user network. Luma provides a mobile application for controlling and managing the router itself. Luma router provides better coverage for home users by using adaptive band steering technique based on location and data load. However, it shows average performance during testing when compared with low price alternatives i.e. mid range Wi-Fi routers [112]. The efficiency of content filtering and built-in security techniques is also questionable as the device is expected to perform all these operations using limited resources available. This device also relies on user to setup security preferences which also limits the efficient use of the features available on the device.

2.3.6 Dojo Gateway

Dojo gateway was initially launched in 2015 [93]. Dojo gateway is designed to resolve the security issues in SOHO networks, which are littered with a number of connected devices. Dojo gateway is an easy to setup device which requires plugging in the base station into home router and installing a companion smartphone application. The base station then itself scans the whole network to find out connected devices and looks for any vulnerabilities. It monitors the traffic passing through gateway to detect any malicious activity. User is alerted about any suspicious activity by changing the status lights on pebble.

The design choice for Dojo allows users to place the pebble anywhere around home to get status updates, while the vulnerability assessment box (i.e. base station) should be placed next to router. The companion application also shows the detailed information about any potential security threats and give suggestions to the user about what can be done to avoid these threats. Dojo gateways manufacturer uses Dojo Security cloud to update vulnerability assessment box, to make sure that it can detect latest security issues.

Dojo gateway is a passive device which only scans the network for any vulnerabilities. It does not actively block any malicious activities in the network but relies on user to take necessary actions. The constant update of its vulnerability detection abilities via security cloud is a useful feature. The manufacturer says that Dojo does not collect any PII about the devices but it uses machine learning and collect meta data information to find new threats.

The device is expected to cost $199 but is not launched yet, therefore, there is no performance evaluation available (by the time of writing). It is handy to notify the user about threats, however, most users do not take actions or the low level knobs available on common routers makes it too tedious to perform configuration update task.

The manufacturer does not give any details about what machine learning techniques will be used and what kind of meta-data information from user devices will be collected for machine learning based threat analysis. There is no indication about whether these machine learning algorithms will be operated on the vulnerability detection modules installed in the network or in Dojo cloud. There is also no information about what kind of control is available to the user over the processing of the information collected from its network.

2.3.7 Cujo Gateway

Cujo gateway is the latest in the series of devices launched to protect home networks. It provides plug-n-play protection for all devices in the network including mobile, IoT devices etc. [25] Cujo gateway promises to protect user's financial and personal data, device integrity and offers features like parental and privacy control. Cujo campaign advertises that it can monitor home network and detect the threats in home network.

It does so by inspecting all the data coming and leaving from the network. It can detect and block viruses and malwares in the network and its ability to perform these functions is always improving by constantly adapting to block new threats. Cujo gateway uses *Cujo Cloud* to updates its threat detection services. This requires monthly or yearly subscription from Cujo cloud to update malware and threat detection mechanism.

Cujo gateway's campaign page does not give detailed information about device functioning and what kind of operations will be performed on the device. It also does not give any details about what kind of data will be collected and utilized for these analysis operations and what information will be transferred to the Cujo security cloud.

Most of the devices discussed here rely on constant monitoring of the network to detect different kind of threats, malwares or attacks in the network. Some of them e.g. Dojo gateway etc. uses remote services to update their ability to detect these threats. Others e.g. Bitdefender Box etc. reroute all user traffic through their security cloud to actively analyze and secure any traffic flowing inward/outward from the user network. Although, all these products provide a companion mobile application which notifies user about any network threats. However, they do not provide actual control to the users about how their network should be protected. There is also very limited support in terms of network management available with these devices. Table 1 shows a comparison between list of features offered by these products.

These solutions are designed mainly for smart homes and (currently) do not provide any scalability model from SOHO to enterprise networks. Currently available information does not give details about transparency and control over what kind of data is collected from user networks and how is it used. If the analysis is performed on edge device deployed in user network, as claimed by Dojo, Cujo etc., the efficiency of this analysis is dependent on the limited hardware and training data available on end devices. On the other hand, if the analysis is performed in service backend, there is no information about the extra delay and what kind of data from user network is used for this analysis.

2.4 User Study

Figure 2 shows the demographics of the user study we have conducted with 170+ participants from 25 different countries. The study was designed to get an understanding of how well users know the risks and threats associated with all the connected things they use in their daily life. The study also queried respondents about their willingness to use a system which can provide automated security and management of their networks. Figure 2 show the demographics of participants from our user study.

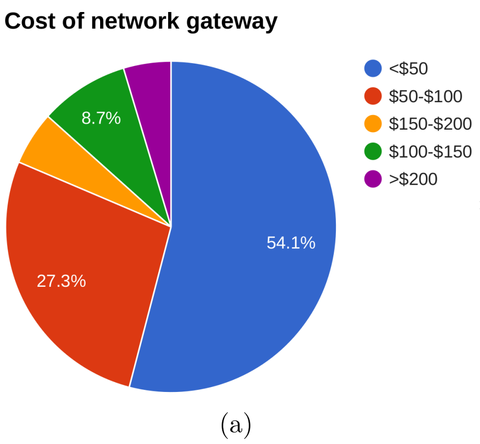
Our respondents belonged to different age groups with diverse occupation ranging from students to lawyers and doctors. The median education was Bachelors degree and modal age was $26 - 35$ years. Table 2 shows the compiled results from respondents of our survey.

This survey showed us that more than 70% of the people are using more than three connected devices in their daily life but a large majority i.e. $\geq 70\%$ are not aware of security and privacy risks associated to these IoT and smart devices. When questioned about the networking gear, 80% of the respondents told that they have spent no more than \$100 on their home

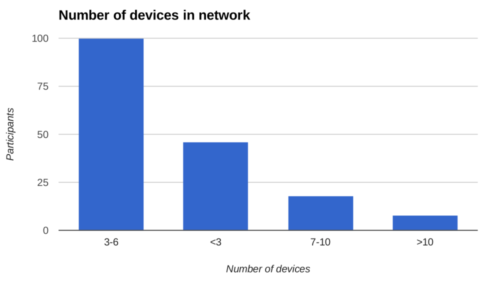
Table 1 Smart home gateway features. *Comparison of features offered by the latest generation of gateways designed to secure IoT and smart homes. Only Google onHub and Bitdefender Box are currently available in market (at the time of writing). *LIM: Limited information available, **NA: No information available*

Feature	**OnHub** [39]	**Sense** [30]	**Box** [7]	**Cujo** [25]	**dojo** [93]
Better network coverage	✓	✓	*LIM	*LIM	**NA
Smartphone application	✓	✓	✓	✓	✓
Prevent hacking	×	*LIM	✓	✓	*LIM
Virus protection	×	*LIM	✓	✓	×
Malware detection	×	✓	✓	✓	✓
Rule based protection	×	✓	×	✓	×
Deep Packet Inspection	×	**NA	**NA	✓	×
Machine learning	×	**NA	**NA	✓	×
Security features	×	✓	✓	✓	✓
Automated Management	✓	*LIM	×	✓	×
Automated Security	×	*LIM	×	*LIM	×
Device discovery	×	×	×	×	*LIM
Device profiling	×	×	×	×	×
D2D communication	×	×	×	×	×
IoT specialized	✓	*LIM	×	*LIM	*LIM
Price	$199	$99	$199	$99	**NA
Subscription	**NA	$8/M	$99/Y	$99/Y	$9/M

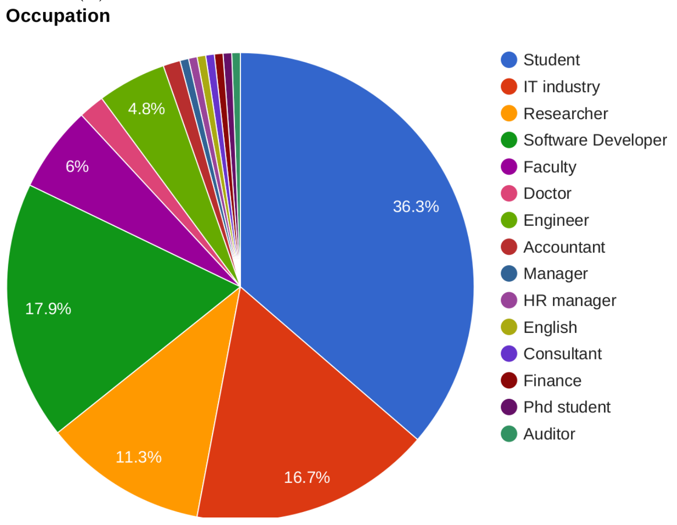
Disclaimer: The data is aggregated from the information provided by the manufacturers on official product pages.



(a)



(b)



(c)

Figure 2: **Demographics of user study**. *(a) Number of devices connected to respondent's network on average. (b) Average spending by respondents on network gateway. (c) Respondents included majority of students and professionals from various fields.*

gateways and although 60% of the people think that their gateways do not provide enough features, 71% have never updated their gateways since first installation as they find the exercise hard.

According to the survey statistics, 85.2% people think that typical network gateways should be more easier to operate and they should offer more features to control device level functionalities. Although current gateways do offer features e.g. mac address or IP filtering. operating on device level granularity, the complexity of managing these options makes them unattractive and less usable for average users. More than 50% of people showed their interest in a system which offers them to process their traffic using middleboxes and

Table 2 Survey responses. *User's response about limitations in traditional gateways and requirements from next generation network gateways and access points. (Scale: 1: Least Agree, 7: Most Agree)*

Scale	1	2	3	4	5	6	7
Typical gateways							
Lack of easy UI	13.9%	8.1%	11.6%	19.7%	14.5%	12.1%	20.2%
Lack of features	5.2%	6.9%	11.6%	28.3%	20.2%	13.9%	13.9%
Lack of information for user	8.7%	15%	16.8	20.2%	11%	15.6%	12.7%
Lack of support for IoT	3.5%	3.5%	9.2%	31.8%	21.4%	17.3%	13.3%
Requirements from gateway							
Should be easy to operate	1.2%	1.7%	11.6%	20.2%	16.8%	23.7%	24.9%
Should support device specific policies	1.2%	2.3%	6.9%	15%	15.6%	24.9%	34.1%
Should support middlebox analysis	1.2%	4%	4%	35.8%	15%	22%	17.9%
Should have better data visualization	1.2%	1.7%	63.4%	24.9%	26%	17.9%	22%
Should provide more feedback to user	2.9%	0.6%	4%	6.9%	12.1%	26.6%	46.8%

90.7% people wanted to have a better utilization of their network and device activity.

Our respondents explained that the biggest limitation in managing their gateways and APs is the difficulty in accessing and understanding the available features. Most of the respondents expressed their interest in a product that could automatically manage and secure their network. Survey results showed that people are interested in systems that provide them information about their device behavior and any malicious activities happening in their network. Table 3 shows that nearly 70% of the respondents do not have problem in sharing their network data with any service which provides them security services and 75% respondents said they will not have any trouble if remote service updates their network gateway.

Our survey shows us that people are interested in a solution which can automate network gateways and APs to provide better security with minimal user interaction required. It also showed that although people are serious about their data privacy and device security, they find it difficult to manage these function themselves. Therefore, they will be comfortable to share their network data with a trusted third party which uses this information to improve the security of their networks. The amount of money spent

Table 3 Privacy concerns. *Responses for user's privacy concerns about remote analysis of network data for providing automated network management. (Scale. 1: Least comfortable, 2: Most comfortable)*

Scale	1	2	3	4	5	6	7
Concerns							
User knowledge of ISP traffic analysis	32.9%	13.9%	11.6%	11%	8.1%	11%	11.6%
Comfort with ISP traffic analysis	12.1%	15%	15.6%	36.4%	8.7%	7.5%	4.6%
Comfort with service based model	4%	9.8%	9.2%	28.9%	30.1%	11.6%	6.4%
Comfort with remote analysis	5.8%	6.9%	13.3%	28.9%	25.4%	14.5%	5.2%
Willingness to replace existing gateways	6.9%	6.9%	12.1%	24.9%	20.8%	22.5%	5.8%

by average user on home gateways also gives us an upper bound over the preferred cost for our system.

2.5 Open Questions

Our literature survey shows that there are various problems with existing solutions. Academic researchers have proposed various techniques which are prototyped and tested to work well but these techniques have not been applied to real world problems and the efficiency of these ideas is yet to be evaluated at the scale of real world networks. Each of these solutions focus on a specific problems experienced in networks and there is little to no study over how these solutions will work in conjunction with each other. To the best of our knowledge, there has not been an all around solution which combines individual (focused) solutions to develop a platform to improve traditional networking gear, which is still using decades old design and protocols.

SDN has become increasingly popular in recent times. However, its practical deployments are still found only in data center environments [55]. SDN has yet to be adopted in wide scale network deployments in SOHO and enterprise networks. One of the barriers in wide scale adoption is cost of devices supporting SDN functionalities. Although, latest generation networking gear comes with SDN and Openflow (OF) support but only a few of these devices operate with vastly deployed network gear in traditional networks.

There have been many proposals to use SDN for redirecting traffic through remotely deployed middleboxes. There has also been some research recently to improve the efficiency of these software-defined middleboxes. However, there has not been any all around system which could allow large scale deployment supporting traffic analysis in remote middleboxes.

New generation of products from industry discussed in Sect. 2.3, which support to offload security and traffic analysis tasks to a remote service are also in their infancy. Different promising products have been introduced but they are not in production yet. These products promise a number of features, however, their efficiency is yet to be analyzed. The cost of these products is high and there is no clear scalability model available for them.

Although, the commercial solutions discussed above provide a number of security features and promise to use fancy machine learning based techniques. However, they do not give any mention of how to control device interactions in the network. In order to protect devices from being infected from a malicious device in the network, network gateways should be able to automatically detect and limit communications between all devices. The service should also be able to detect any suspicious device in the network and block it before it could infect other devices.

Our user study shows that there is a need of a new breed of networking gear and deployment architecture which allows networks to automate their management and security. With so many heterogenous devices connected to our networks, the task of network management has become increasingly tedious for security experts, let alone the common users. Therefore, we need an all around system which provides *"plug and forget"* model of security and management of networking gear so that we can secure all different kind of network environments ranging from SOHO to enterprise networks.

Based on these requirements, we have developed a platform which can provide features such as automated network management, automated network security, controlled device to device (D2D) communications, selective network isolation along with user control over his network. Our platform is designed to be low cost and easy to deploy in different networked environments. We have discussed the design of our platform components in Sect. 3 and 4. We have implemented a prototype of our platform to evaluate the performance of this system in real world deployments. Section 5 gives a detailed discussion over the performance achieved by our proposed system. During evaluation, we have also identified the areas for possible improvements of the system.

3 Securebox

Section 2 discussed the state of the art in network management and security from academic research. It also presented some of the new devices for securing SOHO environments using a service based model. Section 2 highlight some limitations of these solutions and presented open research questions which can be explored in order to improve network management and security situation.

Cost reduction and performance efficiency are among the two primary goals for developing the system. Therefore, the proposed system architecture is based of a service model in which user require minimal equipment to setup their networks and subscribe for the desired services on the go. New services can also be tested by users before actually deploying them in their networks. This approach will reduce the cost for the users to setup and operate their networks.

The proposed system consists of two primary components i.e. Securebox and Security and Management Service (SMS). Securebox is an improved smarter home gateway/AP which is deployed in the network. Here we discuss the design, architecture and implementation details of Securebox in detail and the design, architecture and implementation details for SMS are given in Sect. 4.

Securebox is one of the two key components of the proposed system. It is a lightweight, intelligent gateway or AP which replaces regular APs used in traditional network deployments. Securebox is a smart, low cost replacement for traditional access points which provide connectivity through wireless or wired medium. Securebox provides a better overview of network activities for the users and allows them to have a control over their networks.

It provides features such as dynamic traffic management, Quality of Service (QoS) control, dynamic access control, controlled device-to-device (D2D) communications etc. Securebox resides in the edge networks and connects to a service i.e. SMS for performing different operations e.g. traffic analysis, service mobility management etc. remotely.

3.1 Design

Securebox is conceived as a smart, lightweight, plug-n-play AP which removes any tedious setup procedure and handles most of the configuration and operation tasks by themselves. A typical AP used for network deployment in home or small office networks (SOHO) has a fairly straightforward setup procedure which does not require any extensive procedures from the users. However, these APs only provide low level control interfaces (we would call them *"knobs"*) which makes it difficult for an average user to make any changes in configuration.

In a basic setup, these APs allow users to easily connect their devices to the network but in case an average user wants to make any changes in operational configuration, these *knobs* make that task more painful. These APs only support a limited set of security options e.g. IP/ MAC blocking etc. and do not provide control over device activities by supporting device specific network policies, QoS preferences etc.

Securebox is designed to solve these problems by providing high level control knobs to their users and handles the low level configuration updates automatically. These knobs allow users to setup high level policies for their network e.g. parental control, data-cap for a device etc. and Securebox automatically translates these high level user preferences to low-level network configuration changes.

One of the strengths of Securebox design lies in the automation of configuration tasks as it does not rely on user's knowledge to handle network configuration. A majority of users in typical SOHO environments do not have enough expertise to handle their networks efficiently. They are unaware of the risks and threats associated with their network and device. Therefore, users are unable to take respective actions to block these threats. Adding to that limitation, legacy APs do not support enough features to provide full control over network operations, to the user.

Securebox also uses an interactive model to involve the users in the network management cycle. The system notifies the users about their overall security status of their devices and network. It also notifies the users about any possible threats to their network and devices, and suggests measures to mitigate any threats to user's privacy. Although, Securebox automatically updates network configuration to mitigate most of the threats, these suggestions and notifications help in increasing user's awareness about their security and privacy.

3.1.1 Portability

Portability is one of the important design motivation for making Securebox lightweight and small in size. Using small form factor PC e.g. Raspberry PI [111], Omega [23] etc., makes Securebox highly portable for personal use. Due to its small size, users will be able to carry Securebox and connect it to any available (insecure) Internet connection e.g. public Wi-Fi, hotel networks. Users can then connect their devices to the adhoc network by Securebox. This secure personal access point (S-PAP) setup will prevent malwares and spywares on the insecure network from infecting user devices. It also prevents illegal access and hijacking of user's devices connected to insecure networks.

3.1.2 Architecture

Figure 3 shows the internal architecture of Securebox. Securebox consists of an SDN controller for managing networks routing, switching and other

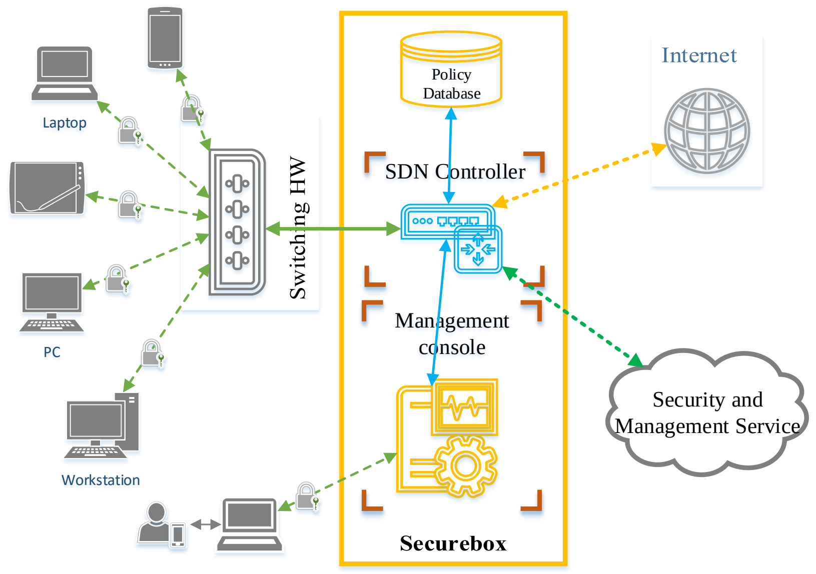


Figure 3: **Internal Architecture of Securebox**. *SDN controller manages the flows using network policies stored in local policy DB. The controller can request SMS to analyze traffic and respond with relevant network policies to be enforced in the network. User devices can be connected through wired/ wireless interface provided by Securebox and user can also set preferences using the local management console or web application.*

function, switching hardware for providing device connectivity, a management console for setting up user preferences, a policy database for caching network policies and a connection to SMS for support in Securebox operations.

At the core, Securebox uses SDN for enabling runtime network configuration and security policy updates. SDN controller provides flexibility to control network operations and configurations updates on the go. In Securebox, an SDN controller allows us to implement device and context specific policies in the network to improve QoS and security of the network. It also eases the remote administration of Securebox and push policy updates which are implemented without requiring interaction with the device. The use of SDN allows us to have a better control over network resource provisioning and implementing security policies at device level granularity.

Securebox also contains a lightweight database containing most frequently used network and security policies. This database serves as a cache of network policies used for managing network traffic at the edge. These network includes policies configured by user itself, policies received from SMS in response of traffic analysis request and policies received in policy database updates. Policy database makes Securebox a stateful AP which can retain its state in case of any interruptions. It helps Securebox to perform traffic filtering operations in cases where the SMS service is not available.

The switching hardware is responsible for providing connectivity to user devices. Client devices can be connected using wired or wireless interfaces. Securebox administrates all traffic coming from wired or wireless interfaces using SDN controller and policy database. Using SDN, Securebox can individually control these interface and connected devices with specific network policies for each interface.

Securebox is a lightweight gateway which offers a set of features e.g. automated configuration updates, traffic analysis etc. In order to provide these service, Securebox needs to maintain connectivity to SMS. SMS also provides resources for performing computationally intensive online/offline traffic analysis on user traffic. It also provides a set of other functions for the Securebox discussed in Sect.4 in detail. Securebox preferably should have constant connectivity to the SMS in order to keep its state updated, however, it can perform the basic set of operations of a normal AP e.g. network connectivity, security etc. in the absence of SMS connectivity as well.

3.2 Policy Rules

A network policy or security enforcement rule is a rule used for traffic filtering on Securebox. A sample enforcement rule a.k.a policy rule, as shown in Fig. 4, consists of a number of parameters for matching to type of traffic and setting up priorities for handling the matching traffic. Policy rules are also used to define QoS, security and other management services implemented by the network. In the proposed system, most of the policies are sent to Securebox by SMS, which develops these policies by taking input from a number of analysis services running in SMS. However, users can also add custom policies in Securebox to override the automatically provided security and other services.

Each policy has an associated *"time-to-live"* (`ttl`), priority and profile variable. `ttl` is used to revoke the policies from the local cache. If a policy is not used during `ttl`, it is removed from the local policy database to limit the size of cache in Securebox. Whenever, a policy is used by Securebox for some matching traffic, its ttl is reset. Priority value is used to resolve conflict between two similar policies e.g. if a user and SMS both inject a similar policy to Securebox, priority decides which policy should be implemented for traffic management. These rules can be grouped together based on their profiles i.e. *"Regular", "Trusted", "Untrusted"* and *"Strict"*. The scope of policy rules defines the type of traffic on which these rules should be applied.

In the proposed system, user policies have higher priorities compared to the policies injected by SMS. Policy profile is used for bundling relevant policies depending on their context and preferences. When a new profile is setup on Securebox, a set of policies specific to that profile are sent to

```
{
  "_title": "Sample enforcement rule",
  "id": 12345,
  "name": "Policy1",
  "profile": 4,
  "source_mac": ["13-73-74-7E-A9-C2",
                                  "BC-D5-8C-FA-4F-54"],
  "destination_mac": [],
  "source_ip": ["214.128.120.44", "10.20.51.123"],
  "destination_ip": ["10.0.25.44", "192.168.0.100"],
  "source_port": [],
  "destination_port": [],
  "devices": [],
  "scope"= 1,
  "priority"= 1234,
  "ttl"= 360,
  "decision"= True,
  "hash" = "pNnZYYehEl"
}
```

Figure 4: **Sample enforcement rule (i.e. network policy)**.*These network policies are cached by Securebox. They can be specified for device(s) (using device id(s) or MAC address(es)). Each policy has an associated time to live and it is removed from cache if remains unused for the specified* `ttl`*. Network policies can be grouped together based on their profiles where profile can be 1: Regular, 2: Trusted, 3: Untrusted, 4: Strict. Similarly, scope of network policy states whether it is applied to Network local traffic or Global traffic only or both i.e Any.*

the Securebox from SMS (or retrieved from local cache of Securebox). For example, if a user changes Securebox profile from *"normal"* to *"strict"*, Securebox requests for policies corresponding with *strict* profile e.g. *"a policy which would block access to any social media website"*. When the profile is changed back to *normal*, policies belonging to *strict* profile are removed from network.

Figure 5 shows the device information received from SMS when a new device is connected to the network. This information is cached to be used later for identifying the connected devices and enforcing device preferences in the network. QoS and network access can be specified for each devices based on its profile such that *Strict* profile will allow the device to either communicate with some (untrusted) devices in the network with no Internet access. *Restricted* profile allows the devices to communicate with a limited set of devices (within the same trust level) with limited Internet access (i.e. allowed to communicate with specific services). *Trusted* profile will allow the device to communicate with any other (trusted) device in the network while

```
{
  "_title": "Sample device profile",
  "id": 123,
  "name": "Device1",
  "device_profile": 2,
  "mac": "13-73-74-7E-A9-C2",
  "isolation": 3,
  "subset": [121, 111, 132],
  "decision"= True
}
```

Figure 5: **Device profile received from SMS**.*This information is stored locally by Securebox for subsequent uses. SMS can update device profile and isolation levels as well as (re)specify the subset of devices with whom device can communicate. The isolation levels can be defined using device profiles such that* 1*: Strict,* 2*: Restricted,* 3*: Trusted.*

having unrestricted access to the Internet. Unknown devices are assigned *Strict* or *Restricted* profile depending on user preferences.

3.3 Functioning

In a typical network deployment, users install a Securebox in their network and connect the devices to it using wired or wireless interface. Securebox can be pre-configured to use a specific SMS or the user can subscribe to any SMS and configure its Securebox to use that service. User subscription from SMS will include user preference about what services should be used for traffic analysis, how frequently should user receive policy updates, what kind of security user wants for its networks etc. A user can register multiple Secureboxes to itself and have different preferences for connectivity and services on each of these Secureboxes. SMS can maintain all these preferences with user profile to ensure similar set of services are provided for the user across all Securebox deployments.

Figure 6 shows the overview of Securebox operations, follow algorithm 1 for processing incoming connection request.

1. Securebox intercepts any new traffic connection to/from any device in the network and extracts metadata information from this request. This metadata information includes 5 tuple information including source destination IP and MAC addresses and port numbers. Header information can also be used to setup more fine grained network policies.

2. Using this information, Securebox looks for any matching policy for the requested connection in local policy database. If a matching policy is found, Securebox retrieves the policy, extracts the required action to

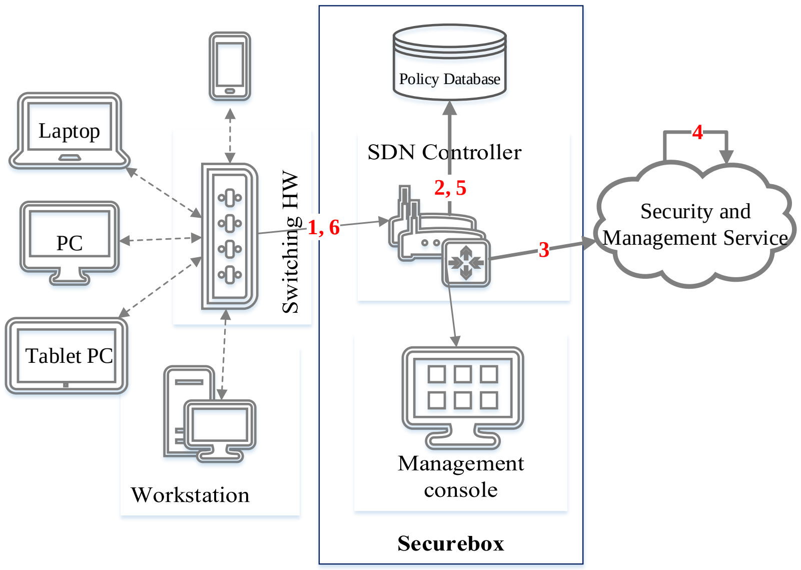


Figure 6: **Internal functioning of Securebox**. *Securebox first checks its local policy database to find a matching policy for the given flow. If no matching policy is found, it requests SMS to anlayze the connection request to identify it as a safe or suspicious flow. Securebox caches the response and updates network configuration to enforce required network state for given flow.*

allow or block the connection request, translates it to respective OpenFlow rule (OFRule) and installs it on OpenFlow switches (OFSwitch) from where the request was originated.

3. In case no matching policy for the given connection request is found in local policy database, Securebox sends this metadata analysis request to SMS.

4. SMS receives the request, retrieves the user profile to obtain user traffic analysis preferences and performs required operations to analyze the incoming request, explained in detail in Sect. 4.2. This analysis can be performed using middleboxes and services maintained by SMS or inside middleboxes leased by the user for dedicated traffic analysis.

5. Once the analysis is done, SMS responds to the request with a network (security) policy which contains the decision to whether allow or block this traffic. Securebox receives this policy and caches it in local policy database. When the Securebox receives similar connection request in future, it will address those requests locally as it will contain matching

network policies.

6. Securebox then adds corresponding OFRules in the network (i.e. OF-Switch).

Algorithm 1 Algorithm for processing incoming flow request in Securebox

```
initialization
while $traffic\_flow\_request$ do
    $metadata \leftarrow extractMetadata(traffic\_flow)$
    if $matchingPolicy \leftarrow policy\_exists(metadata)$ then
        $policy\_decision \leftarrow getDecision(matchingPolicy)$
        $generateOFRule(matchingPolicy)$
        $insertFlow(OF\_switch, traffic\_flow\_request)$
        $updateLog(event)$
    else
        $policy \leftarrow getSecurityPolicy(metadata)$
        $generateOFRule(matchingPolicy)$
        $insertFlow(OF\_switch, traffic\_flow\_request)$
        $updatePolicyDB(policy)$
        $updateLog(event)$
    end
end
```

Securebox can request SMS for traffic analysis in two different modes. In first case, Securebox only sends metadata (5-tuple) information for analyzing whether or not it is a safe connection request and client should be allowed to communicate with these source/destinations. SMS can anlayze this information using malware and threat analysis services, blacklisted server database, malicious traffic signatures etc. to decide if it is a safe source/destination to connect. Once Securebox gets the response, it enforces the appropriate policy in the network.

In the other mode, Securebox can request SMS to provision a dedicated middlebox i.e. Firewall, IDS, IPS etc. for analyzing user traffic online. In this case, SMS will retrieve user preferences and provision a middlebox on request. Securebox will then reroute all its traffic through that middlebox to the destination and perform live traffic analysis. SMS can also send policy to Securebox to reroute user traffic through the middlebox in case it finds a connection request suspicious. As soon as SMS detects malicious activity in the traffic being analyzed, it will push policy update to Securebox for blocking the respective source/ destination.

3.4 Implementation

We have implemented two prototypes for the proposed system. The first bare-bone prototype for the system was demonstrated at ACM MobiCom 2015 [44] workshop and Cloud Security Services (CloSe) Workshop 2015 [45]. The second (improved) prototype will be demonstrated at ACM SEC 2016 [46].

3.4.1 Hardware

We have implemented these two prototypes using two different hardware devices. It allowed us to verify Securebox's design independence on the underlying hardware resources. The first prototype version used FitPC3 machine to run Securebox whereas the second prototype version will use Raspberry PI 2 (Model B) for acting as a Securebox. Table 4 gives a comparison between hardware configuration of both these devices.

Table 4 Comparison of hardware resources available with Raspberry-Pi and Fit-PC3

	Raspberry PI 2 (Model B)	**Fit-PC3 pro Linux**
CPU	900Mhz Quad-Core	1.6 Ghz Dual Core
Memory	1 GB	4 GB
Storage	SD Card	320 GB
Ethernet	1	5
Wireless	None	802.11 b/g/n
USB	4	6
HDMI	Yes	Yes
Price	\$35	\$533

For system prototype, we have chosen `Floodlight SDN controller` [74]. Floodlight controller is an open-source `JAVA` based SDN controller which provides a useful set of built-in features and allows us to write and add custom modules for the controller with ease.

There are several other open source SDN controllers available providing a set of features e.g. NOX [42], FortNOX [84], Flow Visor [102] etc. All these controllers are developed by the research teams and have very little software life cycle (SLC) support, developer community support and documentation available. Therefore, it is relatively hard to incorporate new modules in these controllers. These controllers also suffer in performance when deployed in large scale live environments.

Open Daylight is industry standard open-source SDN controller developed and maintained by Linux Foundation [60]. Open Daylight is developed and maintained by experienced development team and is performance optimized for large scale deployments. There is a SLC support and developer community available as well. However, the amount of efforts required to develop new modules in Open Daylight controller is fairly high as it is designed mainly for production use. It contains a number of features that were unnecessary for the current prototype system and requires more hardware resources for operation. Since, our system is aimed to work on a low cost small form factor-based PC with minimal hardware requirements, Open Daylight would not be a suitable choice in this work.

We have chosen to use Floodlight controller for our system prototype development because of its lightweight and modular design including all essential features. There is a nice developer community and SLC support available for Floodlight as well which greatly helps during development process. The modular design of Floodlight controller eases the addition and removal of any extension module to the controller. As shown in our prototype system, Floodlight controller can show decent performance on a small factor PC e.g. Raspberry PI [111] as well. The built-in modules in Floodlight controller have also come handy during the development of this work.

3.4.2 Software

We have written a custom module i.e. Securebox Module, in `Floodlight` controller with $\geq$ 5000 lines of JAVA code. Some open-source libraries used in development of Securebox module are listed in Appendix A.1. This module handles device and Securebox profiles, security policies, communication with SMS via RestAPI. Our custom module can be enabled or disabled via `RestAPI` calls to the controller.

Policy database is maintained in memory as a hashtable and periodic snapshots of this database are stored locally in a JSON file for storing state of the controller. In future, local policy database can be implemented using lightweight database e.g. `SQLite` [4] or `NeDB` [17] etc. In order to limit the memory footprint of policy database, each policy has an associated retention period i.e. `ttl`. If a policy is not used during this retention period, it is discarded from the memory and this period is renewed every time policy is accessed.

The prototype system uses `Open vSwitch`[1] (OvS) for managing network interfaces on the AP [82]. The Securebox module in `Floodlight` controller is responsible for injecting any OFRules in OvS, which are formulated on the basis of network policies. When OvS sees a new connection request, it requests Floodlight controller to install an OFRule for this request. Securebox module intercepts this request to perform analysis operations and install a corresponding OFRule based on matching network policy.

The management console for Securebox is developed as a set of RestAPIs. These APIs allows users to receive information about their network state and

[1]http://openvswitch.org/

devices as well as configure the network. They are also used to develop web and mobile applications for controlling the network functions and devices. Securebox also offers some management features via SMS. User subscribe to services and setup high level security preferences in their SMS profile and those policies are later translated to network policies and pushed to all Securebox registered to the user.

Our proposed system provides a companion smartphone application for users to monitor device connected to the Securebox at any given moment. This smartphone application also gives notifications to the user about any possible threats to the system. The smartphone application is discussed in detail in Sect. 4.6.

All communications between Securebox and SMS are done over RestAPI using `HTTPS`. In order to be able to communicate with SMS each Securebox must be registered by a legitimate client which is a valid subscriber of SMS. SMS allows every user to setup their preferences for each of their registered Secureboxes individually. SMS maintains the common and individual states of these Secureboxes and push any updates to the Securebox via RestAPI calls. All communications between SMS and Securebox are secured using standard secure communication protocols i.e. `HTTPS` and `TLS`. Section 4 explains in detail what measures are taken in order to protect these communicatons and prevent any attacks on SMS.

Securebox maintains a set of policies which can be specific to a device and context as well. For a new connection request, Securebox identifies the set of policies matching to 6-tuple information extracted from the connection request metadata. If the set includes more than one policies, then Securebox anlayzes the device and context of connection request to narrow down the selection to a single policy using longest match rule. This matching policy is then translated into OFRule which is installed in OFSwitches.

3.5 Deployment Models

Figure 3 shows that Securebox is designed to be a lightweight access point which contains interface which allow users to attach different devices. On the other hand, Securebox functionality is implemented as a generic software component which can be run on any computing hardware e.g. a workstation, laptop or mid-range AP.

3.5.1 Securebox as an AP

Using this model, Securebox can be deployed as an AP where SDN controller and OFSwitch are running on the same node where devices are being connected, as show in Fig 7a. Small form factor PC e.g. Raspberry PI [111], Edison board [54] or OpenWRT enabled APs e.g. ASUS WL500g Premium[2]

[2]https://www.asus.com/Networking/WL500gP_V2/

can be used for this deployment models by replacing their stock firmware with modified controller software and OFSwitch [48].

This deployment model provides discrete control over network operations and device activity with more detailed footprint available for device functionality. It provides central view of all connected device across all APs and supports better security and resource management across the network. On the other hand, this model requires replacement of legacy APs and gateways already installed in the network.

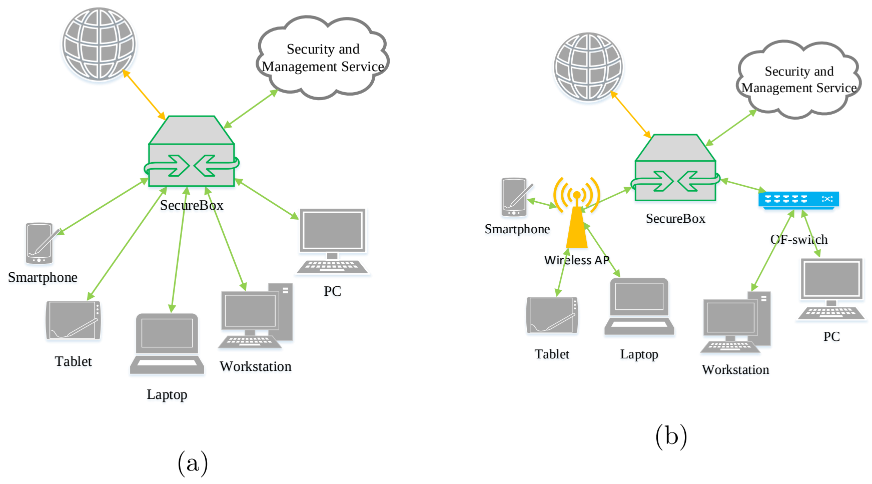


Figure 7: **Deployment models for Securebox in SOHO or enterprise networks.** *(a) Securebox deployed as an AP where it can provide interface to connect the devices automatically. (b) Securebox deployed as a super AP where it allows devices to connect directly, as well as it can manage other legacy gateways and APs in the network.*

3.5.2 Securebox as Super AP

This deployment model allows us to use legacy AP and gateways installed in the network already and follow an incremental approach for updating the network. In this model, Secureboxes as AP are deployed only at vantage points and the edge APs are then connected to this *"Super AP"*, as shown in Fig. 7b. All the traffic from old APs will go through Securebox. This approach supports incremental deployment and reduces the cost of deployment as legacy networking gear is kept in use. Securebox can be run as software functionality on server machines and traffic is proxied through these servers. Securebox application will monitor and analyze any traffic passing through this machine and block any malicious traffic in the network.

Although this installation allows us to deploy Securebox over existing network infrastructure, it has some limitations as we lose fine grained control over the network and connected devices. This deployment adds a layer of abstraction between edge clients and Securebox, therefore Securebox cannot see all the traffic between legacy APs. This issue allows spreading of malware among edge clients before it can be detected by Securebox. Also, this model greatly limits Securebox's control over the network for implementing device level network policies which can cause compromises in network security and QoS.

3.5.3 Securebox as a Sensor

Securebox deployed in the edge networks also acts as a sensor to collect network data. This data can be used for different kind of analysis e.g. malware, botnet, spam net, network threat analysis etc. This data can also be used to improve the QoS and Quality of Experience (QoE) for the users by having in depth information about the devices connected in their networks and their utilization pattern. Internet service providers (ISPs) run various kind of analysis over user traffic to constantly improve QoS for the users but they do not have enough knowledge about the devices connected to the user network and their usage pattern. Securebox can provide more information to the ISPs about ground level statistics from user network to help them optimize network efficiency and provide more specialized services to their subscribers.

We have discussed the design, architecture, implementation and deployment details of Securebox. Securebox is an improved gateway to support automated network management and security services. It uses SMS to provide these services. Securebox is easy to deploy and operate as it supports user with high level interfaces to setup their preferences and automates all underlying configuration tasks. Securebox design uses SDN for its ability to flexibly control network. The improved design allows us to deploy Securebox using various hardware platforms and achieve similar performance. The performance evaluation for Securebox is discussed in detail in Sect. 5.

4 Security and Management Service

SMS provides service backend for the proposed platform. SMS provides support to Securebox for performing security, management, traffic analysis and other tasks. SMS decouples resource intensive and management tasks from Securebox in order to reduce cost and resource requirement of Securebox. As discussed before, SMS can be deployed locally and it can also be provided by a third party services provider just like an anti-virus service provider. Users can subscriber to this service and configure their Secureboxes to communicate with this service to recieve security and network configuration updates, as well as device management etc.

4.1 Design

Figure 8 shows the internal architecture of SMS. *"Central server"* is the focal point of SMS as it is responsible for handling all communications from Securebox. It is also responsible for intercommunication and management of all internal components. Since, SMS will be deployed in cloud or cluster environment for better scalability, central server can be distributed across a number of servers to remove a single point of failure in the system. SMS also maintains a state-aware replica of central server. This backup central server will replace central server if it goes down due to an attack or hardware failure.

"Certification Authority" is responsible for maintaining certificates for clients and Secureboxes. The communication between Securebox and SMS may contain user's personally identifiable information (PII) e.g. web browsing history etc. Therefore, all communications are signed and encrypted using state of the art web technologies to protect against MITM attacks. Certification authority also administrates licenses for third party services which use network data collected by SMS. It can revoke certificates for any Securebox or third party service as soon as the system detects anomalous behavior from an actor in question.

All the data collected from Secureboxes, middleboxes analysis etc. is stored in *"central database"*. This data is utilized by malware, threat analysis and other services. SMS also uses this data to provide user and device management services. The data can also be used by third party services providers e.g. ISPs, security analysis companies, research teams for analysis purposes etc. The central database also acts as a feedback loop for virtual middleboxes and other analysis services operated by SMS to improve their threat/anomaly detection abilities.

The results of the analysis services are used to update Securebox configurations and middlebox configurations. The feedback loops allow the system to improve itself and become more robust in detection of new threats and malwares. This mechanism also provides a self-healing mechanism for the

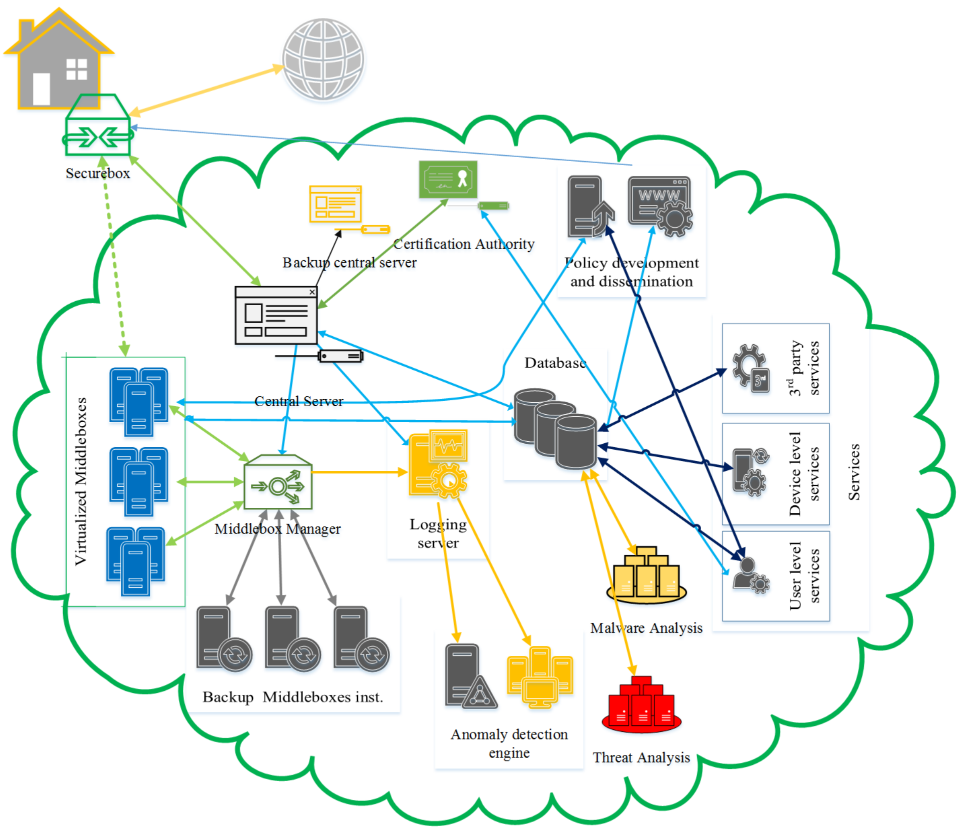


Figure 8: **Internal Architecture of SMS**. *Central server receives all requests from Secureboxes and administrates all services. Certification authority is responsible for maintaining security certificates for registered devices and Secureboxes. Middlebox manager is responsible for managing middleboxes and updating database and service logs. A number of threat and malware detection, device and user management etc. services are also provided by SMS to support context and device based network service depending on user preferences.*

system allowing it to recover from any misconfiguration, which harm overall system performance and increased false positives in threat detection services.

SMS runs a *"Logging Service"* to maintain logs from all the services across the system. These logs are analyzed by *"anomaly detection service"* and human experts to detect any misbehaving Securebox or third party services. Depending on the level of anomaly detected, the devices are either completely blocked by the SMS or moved to quarantine phase where they are actively monitored before marking them safe or infected. The anomaly detection service also uses feedback loop to improve its confidence in anomaly detection and reduce false positives.

"Middlebox Manager" is responsible for managing the deployment of

middleboxes e.g. FW, IDS, IPS, DPI etc. SMS maintains two sets of operational middleboxes (OPMs) at any given interval. The first set of OPMs is used by SMS for running generic traffic analysis for various clients. These middleboxes can run analyze traffic from more than one sources and the number of these middleboxes is dependent on the amount of traffic being processed.

The other set of OPMs include dedicated middleboxes which are leased to users/clients for analyzing traffic only from one source. Users can setup their preferences with SMS such that their traffic is always analyzed in isolated middleboxes. This set will ensure the privacy and no interruption when user traffic is analyzed. These middleboxes can be controlled and configured by the user according to their own preferences. SMS will support scaling up and bringing down the number of these middleboxes depending on the amount of traffic being analyzed.

SMS also maintains a set of backup middleboxes. The backups instances are state aware replicas of OPMs, responsible for traffic analysis operations. If an OPM goes down (due to hardware, software exceptions), the backup instance (live replica) replaces it to continue traffic analysis operations. SMS maintains one or more backup instances of an OPM depending upon the criticality of middleboxes e.g. a middlebox processing latency sensitive data can have more than one live replicas and a middlebox performing analysis on general request can have only one backup instance running.

SMS can also use these replicas to estimate the functional integrity of a middlebox by comparing the result of analysis from OPM and its replicas e.g. if an OPM is not detecting any threats or attacks but both its replicas are flagging the traffic as malicious, it should raise a warning. Such anomalies may be caused due to configuration mismatch etc. In case of such warnings, *"middlebox manager"* should promote a replica to replace OP and launch another replica for the node, if needed.

SMS also provides a number of user, device and Securebox management services as well. Users can update their service subscriptions, change device and Securebox level preferences using the services. SMS also provides a web application and companion smartphone application for managing all these services and getting notifications.

4.2 Functioning

Since every Securebox is registered against a user, all requests from the Securebox are handled according to user preferences. Algorithm 2 shows the operations performed in handling a new request from Securebox. As an example, if a Securebox requests SMS to analyze a connection request from CCTV camera at home to connect to an unknown server, SMS would check the destination server among the list of known trusted servers. If the destination server is not found in the list, SMS will send a policy to

Securebox to either block the connection or reroute the traffic via a middlebox for inspection. The preventive measure taken by SMS depend on severity of issue and user preference. In case, SMS decides to block the connection, it will send a policy to Securebox directing it to *"drop any traffic from CCTV which does not go to (specific) well known servers on the Internet"*.

Algorithm 2 Algorithm for processing incoming requests in SMS.

```
bootstrap system, services
launch middleboxes
while incoming_analysis_requests do
    $info \leftarrow extractInfo(incoming\_request)$
    if policy_exists(incoming_request) then
        $sec\_policy \leftarrow getPolicy(incoming\_request)$
        $sendToClient(sec\_policy, incoming\_request\_id)$
        $updateLog(event)$
    else
        $user\_profile \leftarrow getUserProfile(incoming\_request)$
        $sec\_policy \leftarrow analyzeRequest(user\_profile,$
        $incoming\_request)$
        $sendToUser(sec\_policy, incoming\_request\_id)$
        $storePolicy(sec\_policy)$
        $updateLog(event)$
    end
end
```

Figure 9 shows the internal functioning of SMS when it receives a new traffic analysis request from one of the Secureboxes. The sequence of operations performed in SMS to address the analysis request are explained below.

1. User devices initiates a new connection request to *www.some-random-website.com.*

2. Securebox extracts metadata information from connection request and checks its local policy database to find a matching policy.

3. When no matching policy is not found in local policy database, Securebox requests SMS central server to analyze the traffic.

4. Central server requests certification authority to verify incoming request, user subscription and provide user profile containing user preferences.

5. Certification authority obtains user subscription information and preference from SMS services and responds to central server's request to verify incoming request.

6. Central server requests middlebox manager to deploy a middlebox for user traffic analysis (based on user preference) and update system logs. Central server can also check policy database to find if there is some matching security policy already available. If found, the security policy is sent to Securebox, without requiring to analyze the traffic in a middlebox.

   It also depends on user preferences if user wants all its traffic to be analyzed in every case or it only needs a security policy for the traffic. One of the issues with generic security policies is that it may not satisfy all user's preferences e.g. a generic policy will restrict any connections to any unverified 3rd party app store but a specific user may want to connect to a specific unverified store, considering all the risks and warnings provided by Securebox.

7. Middlebox manager sets up a middlebox for user, sends connection parameter to central server and updates system logs.

8. Central server sends this connection information for middlebox to the Securebox.

9. Securebox uses this middlebox to analyze the traffic.

10. Secureox can use this middlebox for the lease time and can extend lease time as well. The middlebox will be taken down if it is not used, or lease time expires.

11. User device connects to requested website.

12. Securebox caches the policy to address similar requests in future,

13. Middlebox stores the analysis request and related information in database.

14. Middlebox sends analysis data to policy dissemination module to be included in next security policy update.

15. The data for threat and malware analysis services is updated.

16. User and device level services and information is updated.

4.3 Implementation

In this work, core SMS is implemented using Python `Flask`[3] framework [41] with ≈ 27000 lines of `Python` code and ≈ 3000 lines of HTML, JS, CSS code

[3]http://flask.pocoo.org/

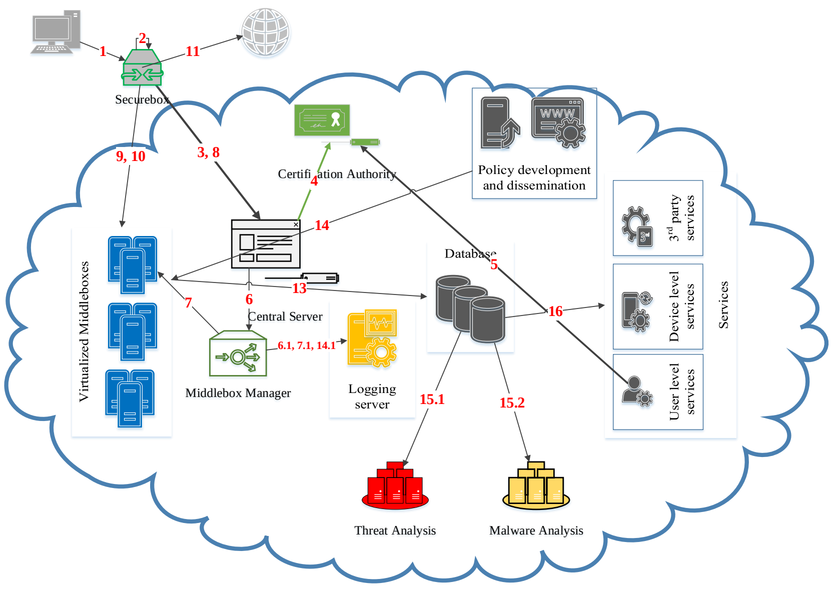


Figure 9: **Request processing flow in SMS.** *All flow processing requests received by central server are verified by certification authority. Depending on user preferences and subscriptions, SMS repsonds to these requests with a network policy to be enforced in the network for given flow. These requests can also be responded with connection information of a virtual middlebox where client can process its traffic. All information is logged and analysis results are stored in database, which are later used to generate policy database updates. These logs are used for anomaly detection engine as well as threat and malware analysis services.*

for web management console using `Jinja2`[4] template engine [96]. A list of other libraries during the development of SMS are available in Appendix A.1. All the libraries used are open source libraries freely available with `Flask` framework.

`Python` was chosen as the development language for this work due to its fast learning curve and wide scale applications. It offers different frameworks to integrate services including web applications and machine learning tools, therefore, it will be more suitable choice during the extension of this system. It is also handy for third party tools as well.

`Django` [13] is also a popular choice for web development in Python but Flask is better suited for small to medium scale web applications as it is faster and simpler to develop. It includes most of the features offered by `Django` and also has developer community support. We expect central database to contain huge amount of data collected from various sources, therefore, we

[4]http://jinja.pocoo.org/docs/dev/

used NOSQL database `MongoDB` [20] for setting up central database.

The modules responsible for device, policy and user management are loosely coupled with each other which allows us to easily replace any component from the application. Each modules offers a web-API for easy communication with other modules and external services. However, all modules use the central database for storing and retrieving any data.

All requests are handles by the core SMS service a.k.a. central server. Every incoming request is authenticated by *"authorization module"*. Authorization module is also responsible for user profile and session management. Any request from a Securebox that is not registered to any user will be discarded. *"User management module"* is responsible for managing user profile and preferences. It contains information about all devices and Secureboxes registered by the user. This information is used for analyzing every request recieved by the SMS.

"Device management service" is responsible for registering and maintaining device information. Users can register their devices and specify preferences using this service. *"Securebox management service"* maintains Securebox registrations and operations. It keeps track of Securebox context, what devices are connected currently and previously at respective Securebox and user preferences for this Securebox. This module is also responsible for pushing policy updates for Secureboxes. Both device and Securebox management service modules are responsible for storing and pushing QoS preferences for devices to Secureboxes in user network.

"Policy management service" is responsible for maintaining security and network policy database. This module communicates with traffic analysis services to obtain result of traffic analysis on user's request, formulates those results in form of network policies and send them to Securebox. This module maintains a huge database for policies developed as a result of analysis performed on request of all Secureboxes. This database is used by traffic analysis module to find any matching policies for incoming analysis request.

Securebox can communicate to *"traffic analysis module"* either directly or via central server. This module obtains user preferences from user, Securebox and device management modules. It also consults policy management modules as well to retrieve any matching policies for user request. Securebox request is responded in form of a policy which is developed by combining context information, any matching policies and result of traffic analysis. A copy of this policy is also stored in the policy database.

4.4 Deployment

For our prototype deployment, we are using `Kubernetes`[5] cluster [11] set up in our university laboratory to deploy SMS, where each service is deployed

[5]http://kubernetes.io/

in `Docker`[6] container [53, 122]. `Kubernetes` is an open-source management platform for containerized applications. `Kubernetes`is an easy to deploy system with built-in load balancing, fault tolerance, service discovery and other services. It decouples application and management plan to make system management easier and improve scalability. SMS is able communicate with `Kubernetes` via RestAPI and request to launch new Docker containers on demand.

`Docker` containers are selected for SMS deployment due to its modular structure. As mentioned before, all services are loosely coupled from each other and can be deployed independently. This deployment model improves the scalability by allowing system to launch more docker instances for a specific module depending on system load. It also ensures SMS operations are not halted when one of the components goes down. The proposed SMS architecture offers three different deployment models.

4.4.1 Third Party Security Service Provider

In this model, a user gets a pre-configured Securebox from third party service provider. It also offers flexibility for users to add their own configuration preferences to the Securebox. Service provider can run different kind of traffic analysis on network traffic based on user preferences and subscription. Service provider can also share the data collected by Secureboxes and analysis services to other services providers. This data is valuable for the device (e.g. IoT, network etc.) manufacturers, service providers (e.g. Netflix [7]), Internet Service Providers (ISPs) etc. to help them improve their products and services. This data can be used to develop new technologies with built-in security features offering better user experience and QoS.

4.4.2 ISP-based Deployment

ISPs can also deploy SMS to provide network management services to their customers. Typically, ISPs provide a home gateway installed at user premises, which can also run Securebox functionality. ISPs do various kind of analysis on user traffic to improve QoS and QoE for the end users. All those analysis can be combined with analysis performed by SMS to provide better security and network services to the end users.

With Securebox, ISP will also have a better view of devices and their usage inside user networks. ISP's adoption of proposed system will be useful for both customers and ISP. Following this model, customers would not need to install a new gateway and ISP can get valuable information about the user networks to offer distinguished and personalized services. It will also

[6]https://www.docker.com/
[7]www.netflix.com

save the cost of deployment and operation for both customers and ISPs and improve ISP operations.

4.4.3 Private Deployment

Private deployment model is useful for research and enterprise-scale deployments since it provides a complete control over the infrastructure. In this model, a client (e.g. enterprise) deploys its own SMS which is responsible for managing all the Secureboxes. This model provides a central control interface to monitor and operate the network. All the traffic is analyzed in a centrally managed infrastructure where personalized traffic analysis techniques can be applied to the data. Private deployment model removes any privacy concerns since the network information is not shared to third party to any external entity, which is desired in enterprise scenarios.

In traditional networks, it is possible to deploy middleboxes centrally at a gateway location and traffic from different establishments is routed through the centrally deployed middleboxes. However, these gateways frequently become bottlenecks, resulting in bad user experience. Such deployments offer very little flexibility for configuration, management in operational networks.

However, the proposed system will offer more flexibility by enabling network managers to classify the traffic and change the middleboxes on the fly. It will greatly improve fault tolerance by significantly reducing downtime of middleboxes. It also improves scalability of infrastructure during peak access periods without compromising user experience and network operations.

4.5 Policy Database Updates

Policy management module periodically generates policy database updates. It follows a proactive approach to push most frequently requested (within a specific period) network (esp. security) policies to other Secureboxes. These policy updates are very helpful in reducing traffic analysis load and can reduce latency experienced by users by many folds. These updates are useful in handling flash crowd scenarios where all users are trying to access some unknown popular content.

In such scenario, all Secureboxes will request SMS to analyze traffic to this previously unknown destination and overload *"traffic analysis service"* possibly causing service disruptions and more latency. Pushing policy for such traffic to the Securebox will result in Securebox handling new connection requests locally, reducing latency as well as load on *traffic analysis service*. This approach is also useful for disseminating security policies in case of newly discovered threats or malwares.

In general scenario, a new threat can infect many networks by the time it is patched in the network due to slow configuration update procedure and manual human intervention required in update procedure. Using policy database updates, security policies to block these threats can be immediately sent to all Secureboxes and readily enforced on the network. There are several performance advantages of using policy database updates as discussed in Sect. 5.

4.6 Smartphone Application

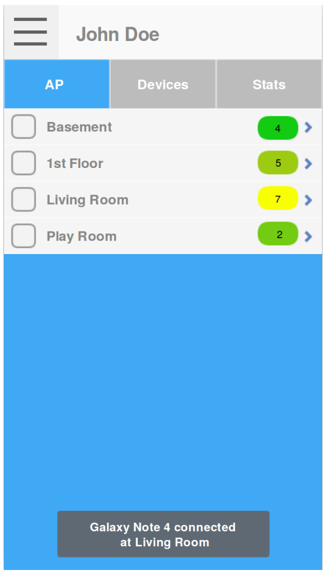
The proposed system provides easy to use interface via web and smart phone applications. Smart phone applications are particularly useful for users since user can easily operate them via hand-held devices. These applications can also provide alerts, notifications and suggestions to the users about how their network activity. Smart sphone applications follow a simplistic design which provides device information and its security profile in easy to understand manner for the user.

These applications provide simple controls for setting up security preferences e.g. allowing/blocking devices, setting up parental control etc. with single action. Power users can use both web and mobile applications for setting up specific security preferences for their devices and network. Smartphone application also provides security rating and suggestions for the users to help them protect their privacy and device security.

Figure 10 shows the design of smartphone application. Figure 10a shows the list of APs (i.e. Secureboxes) registered by the user. It gives the security rating for the AP in color coded format (green showing best) and the number of devices connected those AP. Figure 10b shows the detailed view of the AP. It shows the name of devices connected to AP and their security rating in color coded format. It also gives option to manually update security policies at the given AP. Figure 10c shows the list of device registered for the user itself and their security rating.

Figure 10d shows a detailed view giving detailed information about device including device IP address, security rating, trust level. It shows the AP where device is connected and `uPSK` assigned to the device. The use of `uPSK` is discussed in detail in Sect. 6.7. Smart phone application also lists a set of risk or threats related to the device with a color code representing the criticality of threat. The application can provide more details of these threats, possible causes of these threats and techniques useful to prevent these threats.

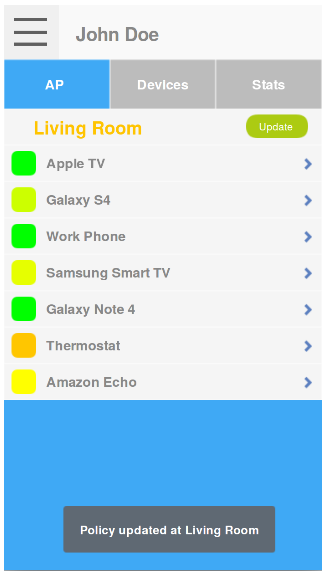
SMS is a service which support Securebox during operations by providing network policies, device, user, mobility management services etc. It also supports software-defined middlebox deployments to analyze user traffic. Section 4.4 presents different deployment models for SMS. The evaluation of system prototype shows that using SMS in the proposed platform architecture provides many fold benefits in terms of network services provided to the users and cost of overall deployment.



Basement 4
1st Floor 5
Living Room 7
Play Room 2

Galaxy Note 4 connected at Living Room

(a) List of APs deployed at user home showing the number of devices connected to each AP.

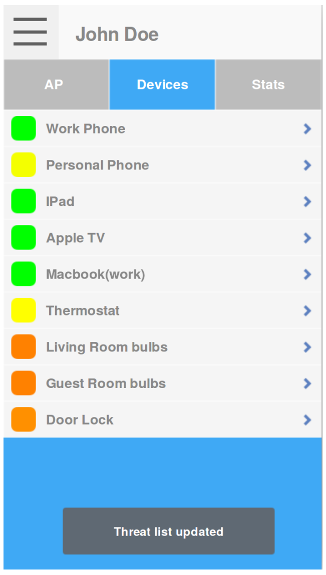


Living Room

Apple TV
Galaxy S4
Work Phone
Samsung Smart TV
Galaxy Note 4
Thermostat
Amazon Echo

Policy updated at Living Room

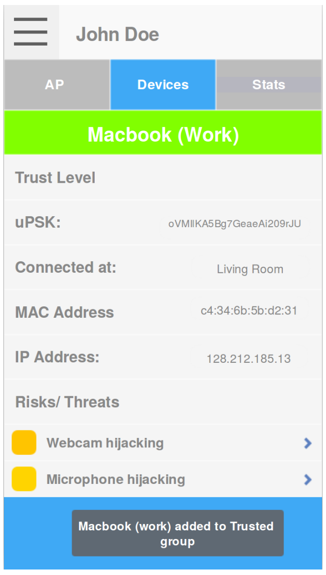
(b) What devices are connected at an AP. The markers show trust rating for each device (Green:best, Red:worst)



Work Phone
Personal Phone
IPad
Apple TV
Macbook(work)
Thermostat
Living Room bulbs
Guest Room bulbs
Door Lock

Threat list updated

(c) List of trusted devices registered by the user. User can block any device from accessing the network.



Macbook (Work)

Trust Level
uPSK: oVMIlKA5Bg7GeaeAi209rJU
Connected at: Living Room
MAC Address c4:34:6b:5b:d2:31
IP Address: 128.212.185.13
Risks/ Threats
Webcam hijacking
Microphone hijacking

Macbook (work) added to Trusted group

(d) Detailed device information including `uPSK` and threats related to the device.

Figure 10: **Companion mobile application**. *The mobile application is used for setting up user preferences, registering new devices and getting feedback about data usage and other network activities. SMS can also generate notifications for the user about any threats or vulnerabilities associated to their devices.*

5 Evaluation

Sect. 3 and Sect. 4 discuss the design and implementation of Securebox and SMS in detail. We present a detailed discussion for performance of proposed system across a set of variables. We have used Raspberry PI 2 as a Securebox and simplified version of SNORT [95] as IDS instance and a Firewall service, during these experiments. The analysis service including FW and IP filtering service was deployed in Docker containers in Kubernetes cluster co-located with SMS.

Stopping criteria The testing was performed by running i iterations for each experiments until standard deviation converges i.e. the standard deviation does not change for x latest iterations where x can be any number of choice ($x = 10$ in our experiments).

5.1 Latency

Latency is a crucial metric for businesses, services and end users. Reports show that higher latencies result in significant drop in traffic for services and businesses [79]. We have evaluated the increase in latency experienced by the user when using Securebox. We compare these latencies with baseline cases where no Securebox is used in the networks.

In Eq. 1, L is the total latency experienced with Securebox setup where bl is the *"baseline latency"* experienced when clients want to communicate with an online server/website. $\sum_{i=1}^{n} l_i$ is the total latency added during the process of analyzing the connection request using SMS. In case there is a policy available in local policy database, this latency will be $\leq 80\mu$ seconds at $229.54 MB/s$ using a usual Micro SD storage card,

$$L = \sum_{i=1}^{n} l_i + bl. \tag{1}$$

In Eq. 1, l_i is the latency induced by each step of connection request analysis process e.g. connectivity to central server, request verification, database lookup, middlebox provision etc. and the overall latency is the sum of all these latencies,

$$\begin{aligned} L &= \sum_{i=1}^{k} l_i + \sum_{j=k}^{n} l_j, & (2) \\ L &= l_1 + l_2 + l_3 + ... + l_k + \sum_{j=k}^{n} l_j. & (3) \end{aligned}$$

Equation 3 opens up left hand side of Eq. 1 where $l_i,\ i \epsilon \{1, ..., k\}$, $k \geq 2$ are the latencies which can be bounded e.g. l_1 is the latency between

Securebox and central server, l_2 is the time taken for request verification using certification authority, l_3 is the time taken by database lookup operation for relevant security policies.

In Eq. 3, $l_j,\ where\ j \epsilon \{k, ..., n\}$ are the latencies due to middlebox operations. These latencies vary according to user preferences e.g. latency for user who does not process traffic using any middlebox will be lower than the latency for the user who prefers to process connection requests in FW, IDS and malware analysis service,

$$L = \lceil C \rceil + \sum_{j=k}^{n} l_j. \tag{4}$$

Equation 4 combines all latencies which can be bounded into a single variable C. With the current prototype setup, we try to minimize C by changing SMS design and operations. Policy database updates and local policy database in Securebox also helps in minimizing the overall latency, given in Eq. 1, by reducing redundancy of traffic analysis requests.

In order to evaluate the latency experienced by the user, we have tested different scenarios which include common user routines e.g. web browsing, VOIP, file transfer etc. The results show that our proposed system only introduces marginal increase in latency as experienced during the connection initiation stages. Later on, Securebox is able to achieve similar latency as experienced using traditional network gateways or APs. All these results were obtained when with empty local policy database so that actual latency experienced due to remote analysis can be measured.

The proposed system is expected to increase the latency experienced by the user as it performs traffic analysis using remotely deployed services and middleboxes. Sect. 4 explains a number of improvements made in system design to reduce the latency incurred by the system. In order to minimize latency overhead, Securebox sends only metadata information (i.e. 5 tuple) from a new connection request sent to SMS for the first time Securebox sees the connection request. Any subsequent similar connection requests will be addressed using the local copy of the network policy received in response to the first request e.g. when a user accesses a website Youtube[8] for the first time and Securebox does not have a matching policy available in local database, it will request SMS for analyzing this request. SMS will respond to this request with a policy for allowing connections to well known Youtube servers.

Securebox will store this policy locally and use it to address any subsequent connection request for Youtube. This approach results in adding latency only at the start of loading a web page content, which does not affect user experience. In our test-bed, the proposed system only increased page load time for Youtube from **3.53s** to **3.81s** for the first time.

[8]https://wwww.youtube.com

These design approaches also help to control the utilization of uplink bandwidth available to the user and minimize the number of requests made to SMS for traffic analysis. These design choices and policy database updates significantly lower the latency overhead offered by our system.

5.1.1 Web Browsing

Web browsing is most frequent use case for Internet among majority of users. With more and more data being published on web forums, web browsing has become integral part of a user's everyday routine. However, majority of users access only a handful of websites. If the user experiences abnormal increase in time taken to load websites, it will negatively impact user experience. Therefore, Securebox should not significantly increase the overall latency experienced by the user.

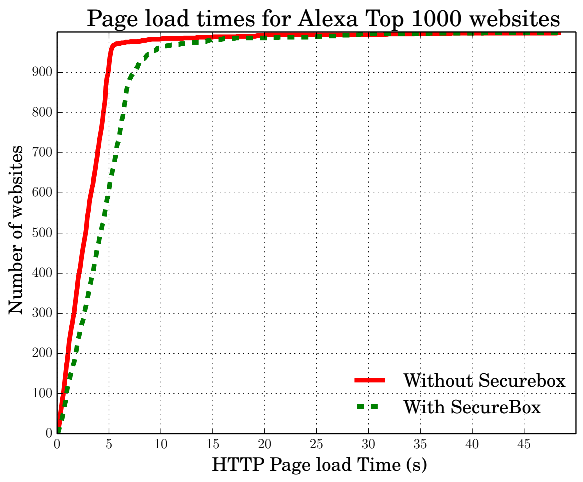


Figure 11: **CDF for HTTP page load times**. *The test was performed for Alexa top 1000 websites (as of 31/07/2016). Securebox setup is able to achieve similar performance as achieved by using traditional networking setup.*

Our system evaluation shows that Securebox is successfully able to minimize the increase in latency experienced by the user. Figure 11 shows that the increase in time taken to load popular websites when using Securebox is negligible.

We have tested the page load latencies for top 1000 websites (as of 31/07/2016) ranked by Alexa[9]. The latency overhead of Securebox is negligible in user experience because this increase in latency is experienced only for

[9]http://www.alexa.com/topsites

the first time website is accessed. For any subsequent requests to the same website, user will experience no added latency.

It is also common that websites load handful content from third party resources. Most of the times these third party resources are similar for a number of websites. Therefore, the local policy database (i.e. cache) comes handy in further reducing the latency because the network security policies for common third party resources are already available in cache e.g. if both websites A.com and B.com load same content from resources.abc.xyz.com as well. When user will load A.com, Securebox will also get security policies for resources.abc.xyz.com which can be used when accessing B.com, therefore minimizing the latency experienced when loading B.com.

The approach to individually analyze every data source for loading one page also comes handy in blocking malicious content which is loaded in background as the user visits a compromised page. Securebox will load the content for webpage from legitimate sources but will not load any (hidden) malicious content from untrusted sources.

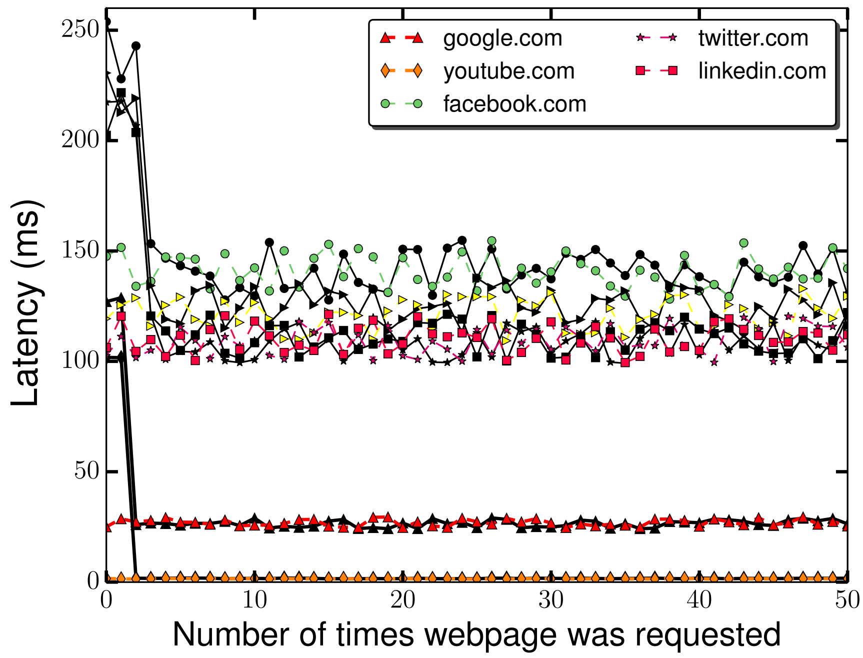


Figure 12: **Latency experienced for web browsing**. *The colored lines show latency experience while using traditional gateway and black lines showing latency experienced while using Securebox. Initially, securebox setup introduces high latency due to remote analysis but for future iterations the latency achieved is same as typical network setup because Securebox has network policies matching to these flows in local policy database.*

Policy database updates also help in minimizing the latency as each policy update will contain policy rules for recently popular content. When a user tries to access this trending content, these request will be handled locally using policy database and no latency, due to remote analysis, will be added.

Figure 12 shows the latency experienced for loading popular websites. The results show that Securebox introduces some delay in the beginning when the web page is requested for the first time, as the request is anlayzed by SMS. After initial request, Securebox achieves similar latency for loading these websites, as experienced in scenario where Securebox is not used, because it has relevant policies in local policy database. The user will experience initial spike in latency if the security policy (relevant to the web page) is removed from the database due to ttl-expiry.

5.1.2 VOIP Traffic

Voice Over IP (VOIP) applications are latency sensitive applications in which user experience is suffered most due to any latency in connection. Therefore, we evaluate the performance achieved by using Securebox for VOIP traffic. We have compared the difference in jitter experienced by user with and without using our system. Results[10] presented in table 5 show that there is no significant loss of performance due to the use of Securebox.

Table 5 shows that Securebox setup introduces no significant jitter in VOIP operations and provides consistent uplink and downlink bandwidth for the application. Both Raspberry PI (R-PI) and fitPC versions of Securebox achieve *"mean opinion score"* (MOST) of 4 or above, which shows that the proposed system can support jitter free good quality VOIP traffic.

These results also show that Securebox design allows us to achieve almost similar performance using both fitPC and R-PI based versions, although fitPC3 has better hardware resources. As the Securebox offloads most of the computational intensive activity to SMS, it does not need more hardware resources for better performance.

5.1.3 Skype VOIP Traffic

Jitter is an important factor in VOIP traffic, therefore, we evaluate the jitter experienced for Skype[11] calling using Securebox as network gateway. Skype is popular VOIP client and is frequently used in SOHO and enterprise environments.

Figure 13 shows three different scenarios where two users, Alice and Bob, try to make a Skype call with and without using Securebox as network gateway. The first scenario is a *"baseline scenario"* representing current

[10]These tests were performed using an online service hosted at http://voiptest.8x8.com/
[11]https://www.skype.com

Table 5 VOIP Performance. *In comparison with typical networking setup i.e. without Securebox, we can achieve similar performance (with excellent quality) for VOIP traffic with Securebox setup.*
C: Client and S: Server.

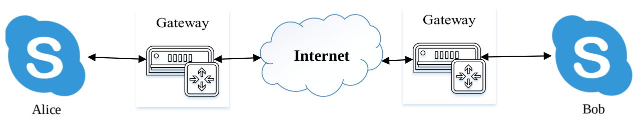
Parameter	**No Securebox** Mean (± StDev)	**Securebox (fitPC)** Mean (± StDev)	**Securebox (R-Pi)** Mean (± StDev)
Download (Mbps)	13.1 (± 0.8)	12.905 (± 0.7)	12.6 (± 0.7)
Upload (Mbps)	2.153 (± 0.1)	1.783 (± 0.1)	1.69 (± 0.1)
Download Consistency	80% (± 1%)	78% (± 2%)	78% (± 2%)
Upload Consistency	86% (± 2%)	83% (± 1.6%)	82% (± 1.5%)
Download BW (Mbps)	18.5 (± 1.3%)	17.7 (± 1.5%)	17.3 (± 0.9%)
Jitter (in ms) (S $\rightarrow$ C)	3.3 (± 2.1)	5.4 (± 2.1)	5.8 (± 1.9)
Jitter (in ms)(C $\rightarrow$ S)	5.8 (± 0.8)	5.6 (± 1.2)	5.2 (± 0.9)
Packet loss (C $\leftarrow$ S)	0% (± 0%)	0% (± 0%)	0.5% (± 0.5%)
Packet loss (S $\leftarrow$ C)	0% (± 0%)	0% (± 0%)	0% (± 0%)
MOS Score	4.3 (± 0.1)	4.1 (± 0.2)	4 (± 0.2)

networks where neither of the users is using Securebox as network gateway, see Fig. 13a.

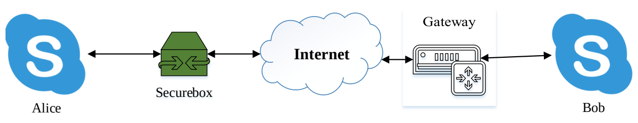
Table 6 Comparison of performance achieved for Skype call quality. *The jitter experienced for skype calling does not vary significantly with and without using Securebox, therefore, user gets uninterrupted VOIP calling experience.*

Jitter (ms)	**Mean**	**Q1 (25%)**	**Q2 (50%)**	**Q3 (75%)**	**95th percentile**	**MOS**
Baseline	10.2	9.486	10.2034	10.899	11.480	4.2
Scenario B	10.4	9.679	10.394	10.434	11.169	4.1
Scenario C	10.6	9.972	10.601	11.336	11.445	4.1

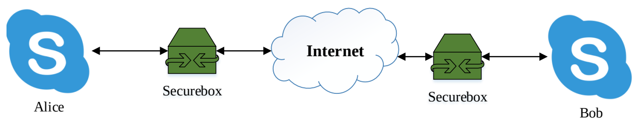
Table 6 compares the jitter experienced in different scenarios shown in Fig. 13. In scenario B and C, users experience slightly increased jitter due to analysis of initial connection requests in SMS. However, during the rest of call, jitter is approximately similar as experienced with traditional network setup. It is because Securebox has all security policies in local policy database.Average MOS score achieved in both scenario B and C is ≥ 4, showing that the user experience and video quality was undisturbed [115].



(a) **Baseline scenario**, where both users use traditional gateways installed in their network.



(b) **Scenario B**, where only one of the users is using Securebox as network gateway.



(c) **Scenario C**, where both users are using Securebox as network gateway.

Figure 13: **Skype test scenarios**

5.1.4 File Transfer Performance

File transfer over HTTP and FTP protocol is also an important use case for Internet users. Compared to Bit-torrent, these file transfers are performed over direct connections to a server and mostly these servers co-host similar content. Therefore, only a few connection requests are analyzed by Securebox and local policy database, once again, helps in lowering delay in file transfer. Figure 14 also shows that Securebox increases the file transfer time by only a negligible amount.

Table 7 File size (in MB) for FTP/HTTP performance testing.

File1	File2	File3	File4	File5	File6	File7	File8
154	205	269	996	133	818	952	478

5.1.5 Bittorrent Traffic

Peer to Peer (P2P) traffic is an interesting use case for Securebox, as P2P clients makes parallel connection to multiple sources. These connections are repeatedly updated and some of these sources may not be secure to connect. As Securebox analyzes new connection requests, in worst case scenario, the

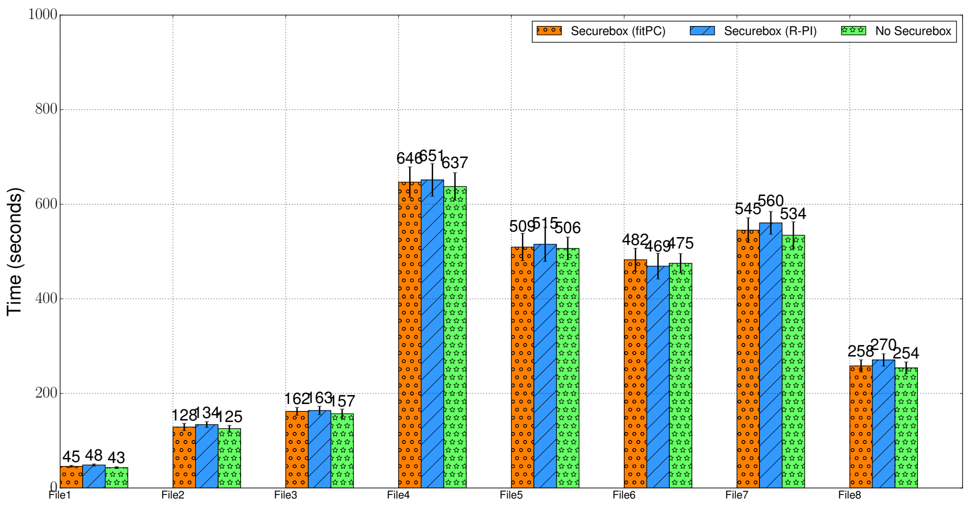


Figure 14: **Performance for HTTP/FTP protocol**. Comparison of time taken for file transfer using HTTP/FTP protocol while using Securebox as network gateway.

number of these connections requests to be analyzed can be significantly high when using P2P traffic, causing high delays in file transfer.

Figure 15 shows the comparison between time taken for downloading a file using Bit-torrent protocol [85] with and without using Securebox. The results show that Securebox does not substantially increase download times. Additionally, Securebox secures client from connecting to any data sources which may be compromised or serve malicious content.

5.1.6 IPerf

Figure 16 shows the performance difference experienced for bandwidth testing when using Securebox. These tests are performed using two IPERF servers. Server 1[12] is hosted by FUNET cloud providing connectivity to test bed and Server 2[13] is hosted in Amazon EC2. The results show that there is ±3% impact in bandwidth and throughtput achieved when using Securebox. Once again, the performance of Securebox is similar for R-PI and fitPC based versions.

5.1.7 Processing and Memory Overhead

Securebox allows user to setup context based network preferences on per device granularity. Setting up these network preferences may require a large number of network policies to be used for traffic filtering. Based on these

[12]iperf.funet.fi
[13]iperf.scottlinux.com

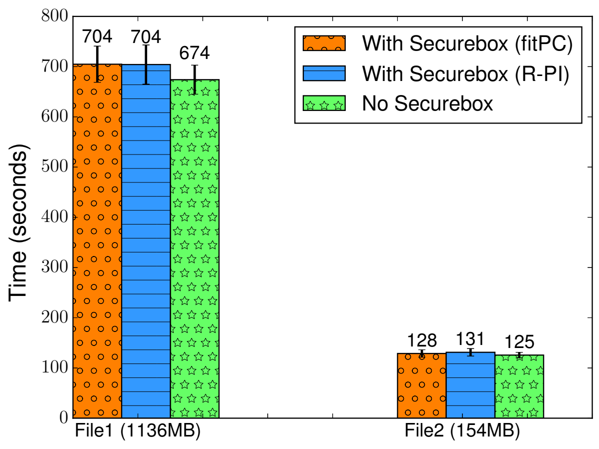


Figure 15: **Performance for Bittorrent traffic**. *Comparison of time taken for file transfer over Bittorrent protocol with and without using Securebox shows that Securebox does not introduces substantial delay as the data is exchanged with several other nodes (i.e., several parallel connections). It has added benefit of protecting user from untrusted servers serving malicious content.*

preferences, there will be only one matching network policy which should be enforced in the network for any given flow. In order to minimize the time required to find this matching policy, we store network policies in a hash table structure so that matching policy search takes constant time. Figure 17a shows that the latency experienced during communication among $D1$, $D2$, $D3$ is not affected by the number of policies used for filtering network traffic.

We have also measured the affect of total number of concurrent flows in the network over the latency experienced in the network, see Fig. 18a. Figure 18b shows that the CPU utilization for Securebox only increases marginally with the number of concurrent flows in the network. Figure 17 and 18 shows that Securebox does not require high processing or memory resource for carrying out network traffic filtering operations. Therefore, it can be deployed using limited hardware e.g. Raspberry PI etc.

5.2 Selective Isolation

Securebox provides support for selectively isolating traffic from different devices in the network. It is an essential feature for mitigating security threats in IoT or public networks with hundreds of untrusted devices connected to the network. Figure 19 shows how selective network isolation can limit D2D communications between untrusted (potentially malicious) and trusted devices in the network.

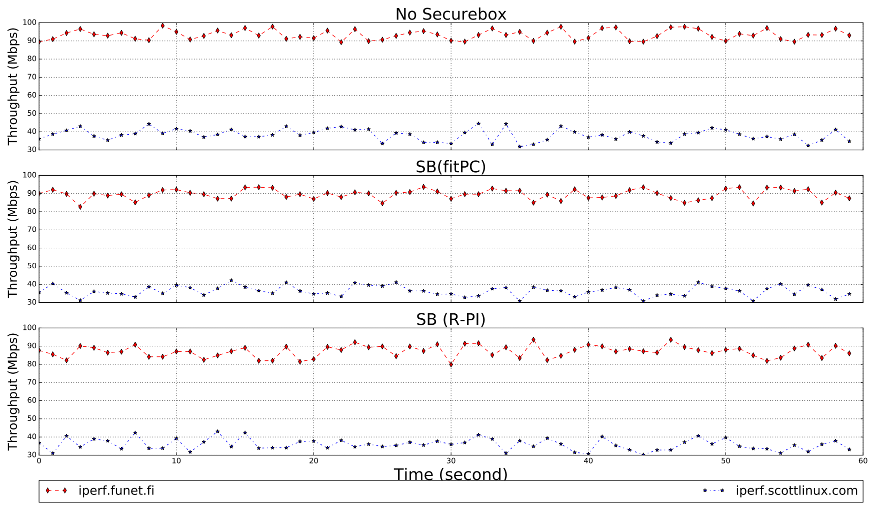


Figure 16: **IPerf bandwidth testing**, *"iperf.funet.fi" is hosted by FUNET, Finland and "iperf.scottlinux.com" is hosted at Amazon EC2.*

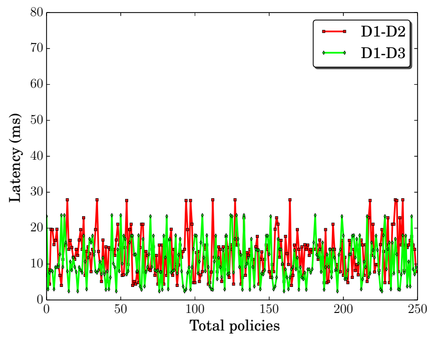
5.3 Phishing Attack Prevention

Phishing attacks are very common these days, where attackers setup fake webpages closely resembling to legitimate websites [26, 3]. Users are directed to these pages through DNS injection attacks and compromised routers. When a user connects to these malicious websites, attackers infect their machines by serving malicious scripts running on the webpage [61]. These websites also serve malicious content to the user which can infect their devices [40, 14]. Attackers can also steal user passwords, credit card and other PII through these fake websites [65].

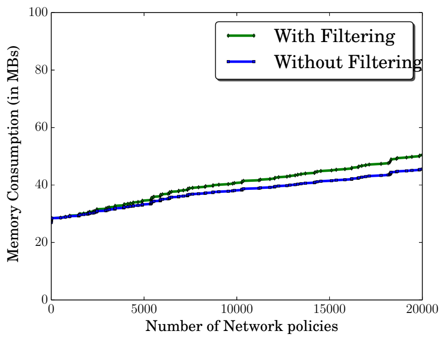
One critical advantage of Securebox is to block such attacks by dynamically inspecting the destinations where user is connected. Securebox prevents any attempts for connecting a user to a fake destination appearing as untrusted. This features also prevents user from downloading malicious content from untrusted sources (i.e. other than official content provider) e.g. if a user wishes to a download some content or install a smartphone application and a google search redirects it to a 3rd party app store instead of legitimate source e.g. Apple App Store[14] or Google Playstore[15], Securebox would block this attempt and notify the user that an attempt to (possibly) download malicious content has been blocked. This information will help users to avoid illegal third party marketplaces serving malicious content.

Figure 20 also shows one such scenario where a user tries to connect to an online portal for shopping sports accessories. Figure 20a shows a legitimate page for the online shopping portal and Fig. 20b shows the malicious page

[14]https://itunes.apple.com/en/app/apple-store/
[15]https://play.google.com/store



(a) Effect of total number of filtering policies over network latency.



(b) Memory utilization relative to number of policies used for traffic filtering.

Figure 17: **Overhead for filtering policies.** *(a): The latency experienced by the user is not related to the number of network policies used for setting up traffic filtering or QoS in the network. (b): The overhead of storing network policies does not increase the memory footprint of Securebox.*

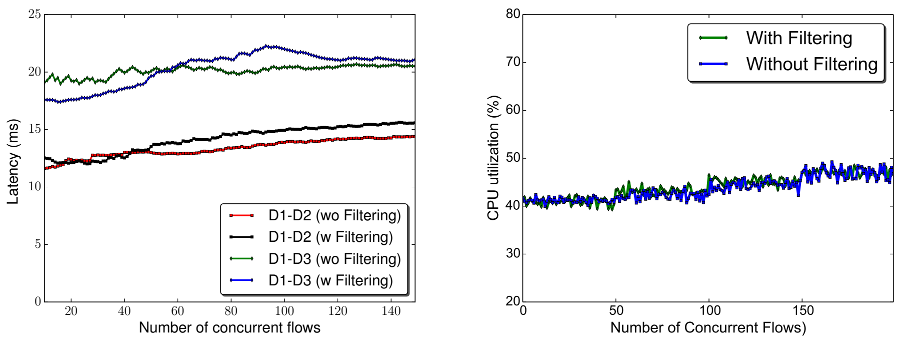
served by attacker. Both these pages look almost similar except for the web address (highlighted in top left corner). An average user will be unable to identify this difference as malicious page and becomes a victim of attack.

On the other hand, if user is using our proposed system, the Securebox will detect that the response for user requests is delivered by an untrusted source. It will block this attempt and notify the user about possible phishing attack, as shown in Fig. 20c. This notification will also be delivered via smartphone application. Such notifications will also help users to understand online security threats and improve their ability to become a victim in such attacks.

5.4 Privacy

Privacy is one of the most important concerns raised by the proposed system as the system uses offsite traffic analysis services. This requires sending user browsing information including source/destination IP addresses which may reveal user activity. Our proposed model is based on the trust relation between the service provider and user. It is the same model used by VPN, anti-virus or any other services where user trusts the service provider for maintaining user's data and privacy.

In case of VPN, user traffic is rerouted through service provider network who provide secreacy to user traffic from unwanted sniffers but VPN service provider can look at user browsing activity. Similarly, anti-viruses collect data from user's machines and send it to their cloud services for their analysis operations etc. Recent research has shown that it is difficult to provide complete accountability and management services while offering complete



(a) Effect of total number of network flows over latency. (b) CPU utilization relative to number of concurrent flows in the network.

Figure 18: **Overhead for filtering policies.** *(a): The latency experienced by the user is not significantly affected by the total number of concurrent flows in the network. (b): The increase in number of concurrent flows does not significantly impact the CPU utilization for Securebox as well.*

anonymity of the users [31].

During this work, when we asked respondents in our user study about *"How comfortable would you be in sharing your network information to get network security and management services?"*, a majority ($\geq 60\%$) of respondents said that they would be comfortable with their data being analyzed by a service provider for better QoS, security and automated network management. The functioning of SMS is similar to an ISP which also performed various kind of analysis on traffic to improve QoS, however, SMS give more transparent control to the user over what kind of analysis are performed on user traffic.

5.4.1 Metadata Sharing

The proposed system design tries to minimize the privacy concerns about the system. Securebox only send metadata information to SMS for traffic analysis request. This contains only header level information (no payload information) which can be seen throughout the packet's route to its destination. Therefore, SMS services would not track session lengths, payload information etc. from this data. The choice of sending only metadata information also helps in reducing latency and efficiently utilizing uplink bandwidth from the user.

If a user wants to analyze all incoming/ outgoing traffic through some middlebox, his/her traffic is directly channelled through a dedicated middlebox as per user preference. SMS will be responsible for updating configuration for this middlebox but no user related information is extracted from the middlebox to ensure user privacy. User can chose to opt-out from contributing any threat information detected by that middlebox into threat database. It

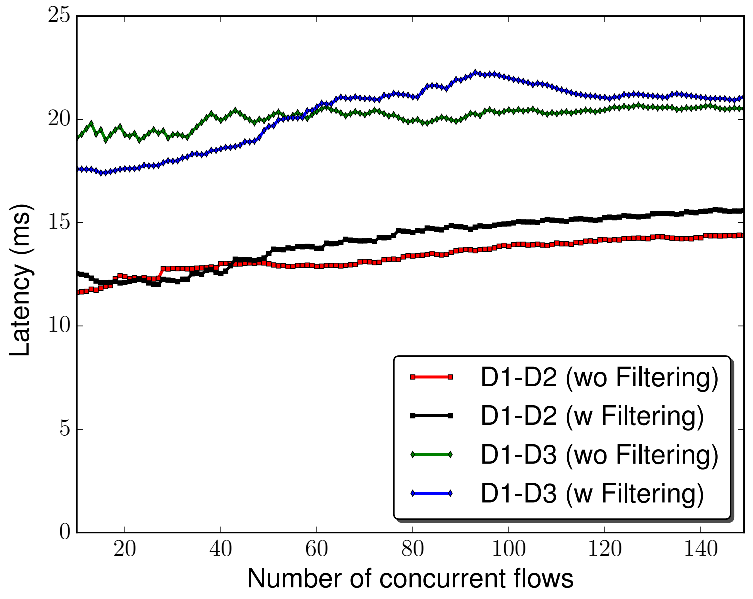


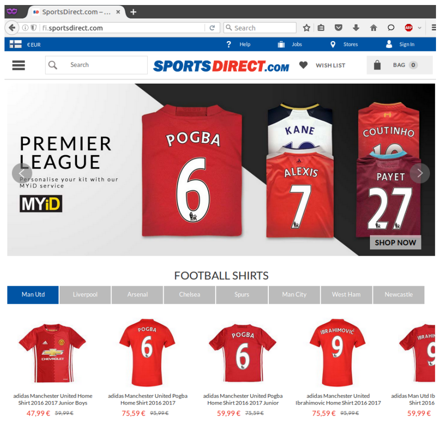
Figure 19: **Selective isolation**. *Securebox is able to enforce device specific network isolation by dynamically restricting communications between untrusted (D4, D5 in this case) and trusted devices, to prevent potentially malicious devices from infecting other device(s) in the network.*

shows that the system is designed to make an optimal trade-off between user privacy and usable security and provide maximum control to users over how their data is used.

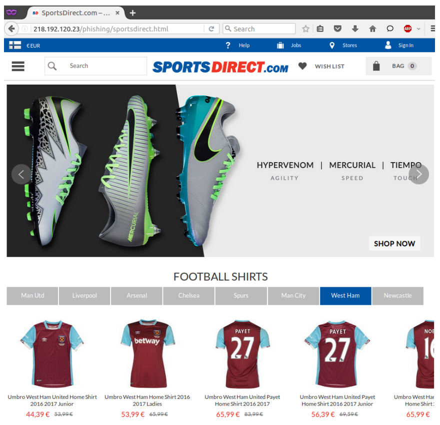
5.4.2 Policy Database Updates

Policy database updates is another useful feature for protection of user activity privacy. The updates contain policies for frequently analyzed connection requests from all Securebox deployment. Zipf's law probability distribution [86] suggests that majority of user traffic should be directed to only a handful of websites. SMS can easily detect most frequently visited websites by its users and it can send relevant security policies for these destinations in single policy database update. It will allow Securebox to handle majority of connection requests locally by Securebox.

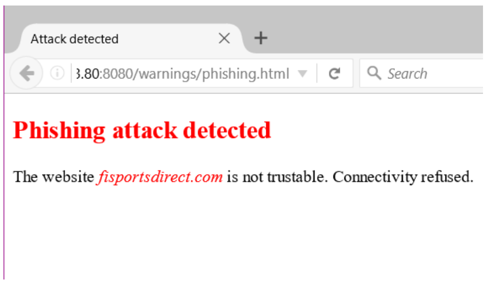
When majority of traffic is being handled locally by Securebox, the chances of tracking user activity via analysis requests from SMS are significantly minimized. Therefore, privacy concerns are lowered when there is only a minor percentage of user's traffic is analyzed remotely.



(a) Original webpage for online shopping portal requested by the user


218.192.120.23/phishing/sportsdirect.html
SPORTSDIRECT.com
HYPERVENOM | MERCURIAL | TIEMPO
FOOTBALL SHIRTS
SHOP NOW

(b) Malicious page served by an attacker.


Attack detected

3.80:8080/warnings/phishing.html

Phishing attack detected

The website *fisportsdirect.com* is not trustable. Connectivity refused.

(c) Securebox blocking attempt of phishing attack via malicious page of legitimate shopping portal.

Figure 20: **Securebox preventing phishing attack**. *Although the malicious page served by an attacker is visually similar to original web page, Securebox is able to identify it as malicious using the server information hosting this page. It can therefore block the attack and notify the user about possible phishing attack.*

5.4.3 Privacy Supporting Deployment Models

Our proposed system design offers multiple deployment models to deal with privacy challenges. Typically, enterprises are very sensitive about what data is being shared from their network, as it can be of sensitive nature. To address these concerns, SMS can be deployed by the enterprise locally. This choice will provide a number of advantages to enterprise as it can minimize the latency by many fold using a local optimized setup. It will also provide more control to the enterprise to run dedicated services for their traffic analysis and (re)implement these services based on their own preference at any time.

Enterprises usually maintain a data center to process their customer and business data. SMS can be deployed using the resources available in this data center. It will also reduce deployment and operational cost. Enterprises can meanwhile use third party services to improve the efficiency of their in-house SMS for detecting latest threats and attacks. The prototype SMS is also designed to run on commodity PCs, therefore, an average user can also run lightweight SMS on his own machine inside his SOHO network.

5.4.4 Privacy-aware Data Sharing

SMS design includes that user data is used for improving overall security and it can be shared with third party services for research and analysis purposes. However, users can opt-out of this data usage/sharing program by paying a subscription fee for any services they use. On the other hand, service provider can offer free services to the users/ subscribers who allows the service provider to use and share their data for third party analysis etc.

5.5 Collaborative Approach for Network Security

Lack of collaboration is the one of the important problems with network security currently. As mentioned before, lack of collaboration between security teams can help adversaries to use similar attack mechanism to successfully launch attacks against a number of organizations and the security teams in each of those organizations have to individually detect and block these attacks.

Although network security solutions have constantly improved over the last two decades, lack of collaboration between network security teams requires security experts to make repeated efforts to detect similar attacks. These attacks often remain undetected for a significant period of time which is enough for attackers to infect networks and cause damages [27].

The collaboration between network security teams is greatly limited due to organizational and legal reasons. However, enhancing these collaborations can greatly help in improving the overall network security situation. Security community has long acknowledged the need for a mechanism for sharing network attack related information and there exists an IETF working group developing protocols for sharing information about network attacks [52].

Improving the collaboration between networks to improve the overall network security situation is one of the most important design goals of the proposed system. In general, the compromised devices in SOHO networks can be used in DDoS, spam attacks launched against enterprises and consumers. The information obtained from these networks can be useful to block DDoS attacks launched against enterprise networks. ISPs can use the information about compromised devices to prevent them from becoming part of any DDoS or spam net in the first place as well.

The proposed system is built on the idea to use the information obtained

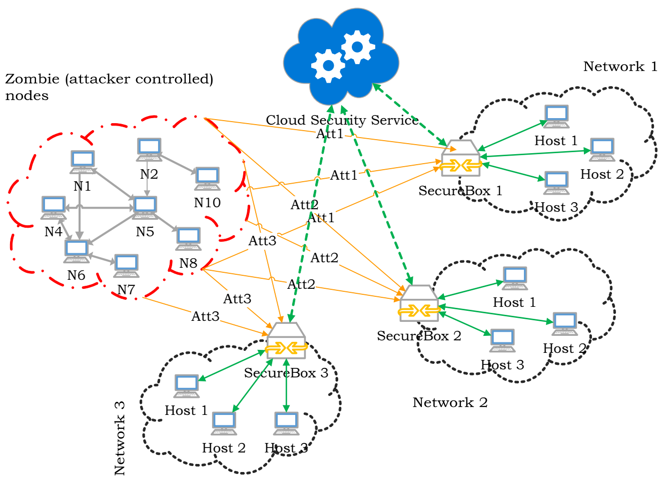


Figure 21: **Network attack simulation environment setup.** *We use a set of zombie nodes and a set of interconnected networks with user devices. Zombie nodes are used for attack each of these networks one after another. The attack scheme followed for each network is similar as well.*

from the network segments to protect the whole network. It gives us a platform for improved sharing of network information to promptly detect and block rogue network nodes or segments.

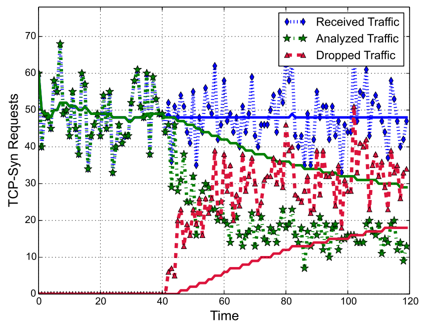
Figure 21 shows the layout of a typical network deployment using Securebox setup. Each network segment can represent an enterprise, SOHO or any organization's network where each Securebox, acting as a gateway, is also connected to external (cloud-based) security service i.e. SMS.

Figure 21 also shows a number of zombie nodes which are connected to various other networks. Such nodes can be controlled by an attacker to work in unison for launching an attack [40]. These zombie nodes are used to launch various attacks against these network segments.

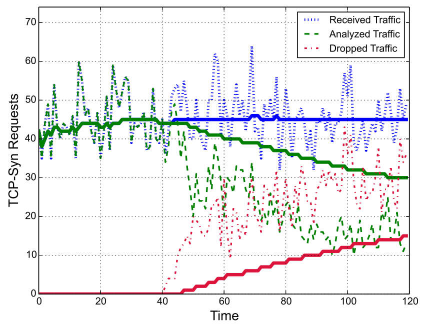
Typically, an attacker can use same set of nodes to launch the attacks against any of these three network segments and in each network, network security teams would need to first identify and block the attacks in their network. As explained before, the time taken by network security team to identify (if possibly) all these infections and quarantine them will be enough for attacker to cause significant damages.

Figure 22 shows the traffic analysis situation for the three network segments in a situation where all three network are attacked using similar mechanism and no information about these attacks is shared among networks.

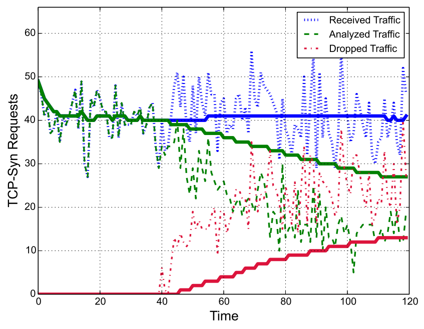
Figure 22a shows that all the traffic received is initially analyzed. As the SMS analyzes the incoming traffic and detects an attack, it directs Securebox to drop the traffic. Figure 22a shows that initially the traffic being dropped was zero and as soon as the attacks are identified, the volume of



(a) Traffic trace from Network 1.



(b) Traffic trace from Network 2.



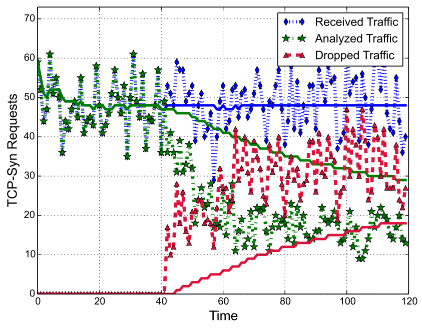
(c) Traffic trace from Network 3.

Figure 22: **Network traffic traces from three network segments.** *With no collaboration in place, traffic from each network needs to be processed separately to identify the attacks. It is a slow approach, taking long time before identifying a threat and enforcing mitigation policies in the network.*

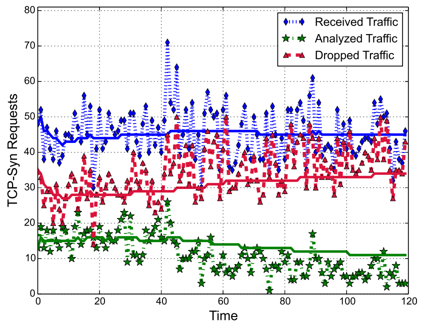
traffic dropped increases. The volume of traffic being analyzed also decreases because once SMS pushes the policy for dropping the traffic (from specific source/destination) to the Securebox, any traffic matching this policy is dropped directly. Similar situation happened in other two network segments as show in Fig. 22b and 22c.

Although the SMS blocks any attacks as soon as they are discovered, this approach has some disadvantages. First of all, SMS receives similar anlaysis requests from different networks and wastes resource to (re)analyzing these requests. Also, these requests utilize uplink bandwidth from the user. Although, we keep uplink bandwidth utilization to minimum by only sending necessary information for anlaysis, however it uses some resources nonetheless. Lastly, attacker might have infected the network already before the attack was detected (port scan attack takes some time to detect as it is uses legitimate request to make connections to various ports).

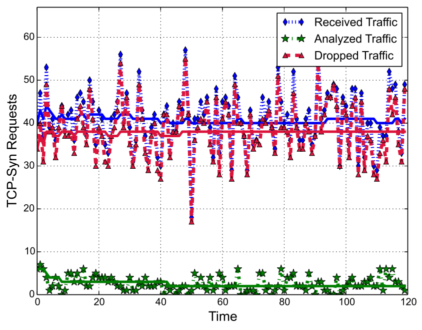
In order to resolve these issues, we implemented collaborative scheme to



(a) Traffic trace from Network 1.



(b) Traffic trace from Network 2.



(c) Traffic trace from Network 3.

Figure 23: **Performance gain by using collaborative approach for network security**. *As the network information is shared to other networks after the attack is identified in Network 1, other networks immediately start to drop traffic from (identified) malicious nodes. Therefore, the amount of traffic processed for Network 2,3 is substantially lesser than Network 1.*

spread attack prevention information across networks using policy database updates. The updates are periodically generated and contains network policies to block recently discovered network attacks. These policies are cached by Secureboxes locally and used to block any malicious traffic without needing to request SMS for analyzing this traffic. It lowers the resource usage at SMS and further minimizes the uplink bandwidth utilized by Securebox for analyzing traffic.

A critical advantage of using this technique is that Securebox blocks the malicious traffic as soon as it is first received at the gateway without waiting for traffic analysis. Policy database updates also improve Securebox efficiency by allowing it to handle most of the incoming traffic locally, which in turn minimizes the overall cost as analysis cost is directly proportional to the volume and type of analysis performed on user traffic.

Figure 23 shows the performance gain by using this approach. When attacker launches an attack on Network 1, all traffic is analyzed (just like previously) until the attack is discovered, leading to increase in dropped traffic, see Fig. 23a. After the detection of attack in Network 1, SMS generated a policy database update for all Secureboxes which was cached by Securebox 2 and 3. This update included policies developed to block any malicious traffic received from attack (zombie) nodes (identified in previous attacks).

When attacker launched similar attack on Network 2, Securebox already had policies to block any traffic from malicious nodes. Therefore, the volume of traffic dropped is significantly higher as compared to Network 1 from the beginning. Network 2 also analyzed traffic coming from other attack nodes which were not detected during attack on Network 1 due to their recessive activity. SMS combines the information from attacks on Network 1 and Network 2 to identify new nodes participating in the attack. Figure 23b shows that the traffic analyzed by Securebox 2 is significantly lower which proves the advantage of using collaborative approach.

After the attack on Network 2, SMS generates another policy database update containing policies to block newly discovered attack nodes. Therefore, as shown in Fig. 23c, Securebox 3 is able to handle almost all traffic locally with very little traffic analyzed by SMS because Securebox 3 had relevant security policies available in local policy database. Figure 23 shows that collaborative approach of sharing network attack information used by proposed improves efficiency and robustness of proposed system.

5.6 Policy Database Updates

Figure 24 shows the trend of analysis requests when there is no information sharing of attacks via policy database updates. When an attacker launches similar attack against all three network segments at *Event 1, 2, 3* respectively, each of these attacks is blocked after individually analyzing the traffic and detecting an attack. The analysis engine is required to performing same analysis repeatedly to detect same attack, resulting in inefficient resource utilization.

Figure 25 shows the advantageous of our proposed approach of using policy database updates to share attack information. When the attacker launches an attack against network 1 at *"event 1"*, it is analyzed by SMS, detected as an attack and blocked at *"event 1a"*. At *"event 2"*, SMS publishes a policy database update for all subscriber networks, which includes the security policy to block attacks similar to the one detected in *"event 1a"*.

These policies are cached by all Secureboxes and when attacker launches similar attacks against networks 3 and 4 at *event 3, 4*. These attacks are readily blocked using cached policies, without requiring to analyze the traffic. Network 1 also blocks repeated any attempts of some attacks launched from same set of *"zombie"* nodes

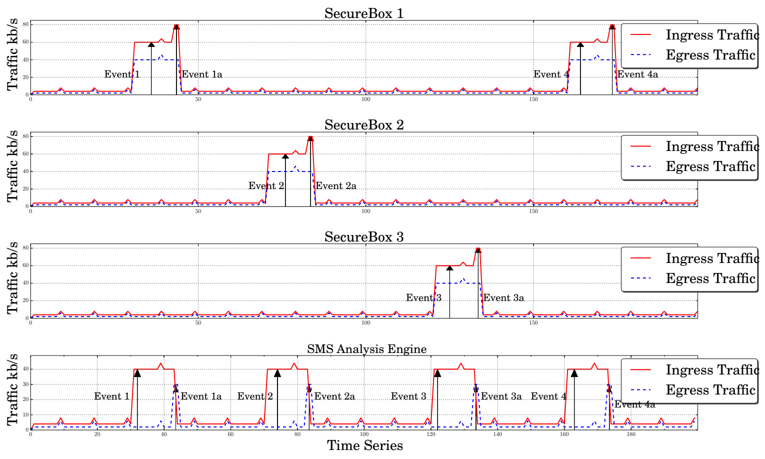


Figure 24: **System with no policy database updates.** *When no policy updates for sharing network information, system needs to individually analyze traffic and identify network attacks.*

Issues

This approach can result in blocking many legitimate nodes which somehow become a part of zombie network. In order to prevent the blocking of such nodes permanently, `priority` value and `ttl` counter is associated with each of the policies in policy database update, as discussed in Sect. 3.2. Using `ttl`, unnecessary policies are revoked from Securebox.

Since zombie nodes are legitimate nodes working under the influence of an attacker, it is possible that these policy updates can lead to unintended loss of connectivity with legitimate nodes e.g. if Alice and Bob are living in same dorm and routinely share project work, media files with each other. If Alice's computer is compromised by a virus and has become part of a botnet, outside the knowledge of Alice herself. Bob's Securebox received an update, marking Alice's computer as part of botnet and prohibiting any communication between Alice's and Bob's computers. In future, if Bob or Alice want to share any files directly with each other, Bob's Securebox would deny such connections.

In such scenario, Bob's Securebox will generate a notification for Bob that there is a connection request to/from Bob's computer to a malicious machine (i.e. part of botnet), which has been blocked. Such notifications will inform users about the devices which have been infected. Bob can share this information with Alice who could then take necessary actions to disinfect her PC. These notifications can also help users in securing their devices and take precautionary measures to ensure the protection of their device in future through various means e.g. installing anti virus, not connecting (possibly)

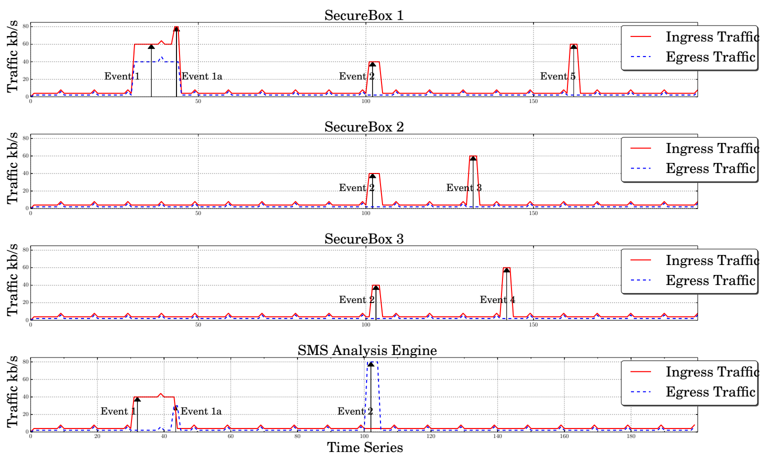


Figure 25: **System with policy database updates.** *Security policies are disseminated to all networks at Event 2, after attack on network segment 1 is identified i.e., Event 1. Therefore, attacks on network segment 2, 3 Event 3, 4 and subsequent attacks on network segment 1 Event 5 are immediately blocked without needing to analyze the traffic.*

malicious USBs, etc.

5.7 Quality of Service and Bandwidth Optimization

Securebox is designed to provide easy management of network resources based on device prioritization. It also users to set high level preferences to prioritize traffic from some of the devices in the network, to enhance user experience. For this purpose, Securebox provides mobile and web application for users to setup these preferences. SMS will ensure that all Secureboxes register to the user have policies dictating latest user preferences at any time.

We evaluate the functioning of these features which allows users to prioritize devices so that Securebox can automatically change network bandwidth available to these device when connected to the network.

Figure 26 shows a typical network where all devices get equal share of the available bandwidth. This distribution can affect user experience e.g. a user is more concerned about video quality on live stream and he can compromise on background file synchronization tasks for better quality of streaming video. In such cases, although users can manually configure file server to move file synchronization to (specific) night hours to get better web browsing experience during leisure time but this manual management is too complex for typical users and becomes more tedious as the number of devices grow.

Therefore, Securebox allows user to setup device priority by a single action

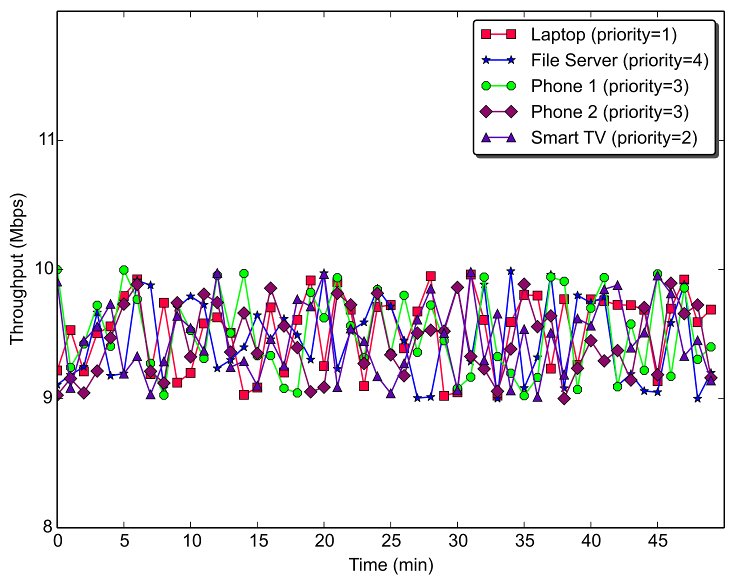


Figure 26: All devices share equal bandwidth in typical network.

at any time. Our proposed system will use these priorities to synchornize users preferences on its registered Secureboxes to allocate bandwidth to each device connected. Securebox summarizes all low level configuration changes into a simple user action which sets the priority for any device and network can adapt to those changes.

Figure 27 shows the dynamic bandwidth allocation depending on device priorities. The measurements were taken while simulating a typical home network where multiple devices are connected and being used simultaneously and user have different priority for each of these devices.

Figure 27 shows that initially smart TV's's bandwidth was increased as it started streaming HD video. At $timestamp = 15$, user starts web browsing on Phone 1 which increases bandwidth allocated to Phone 1. Meanwhile, the bandwidth for file server decreases. At $timestamp = 25$, user starts using his laptop for web browsing/video streaming. Since, the laptop has highest priority, the bandwidth for smart TV is reduced and laptop gets largest chunk of available bandwidth. Smart TV streaming ends at $timestamp = 30$ and user stops web and video streaming on laptop at $timestamp = 45$. When no device is not being used, equal bandwidth is allocated to all these devices again.

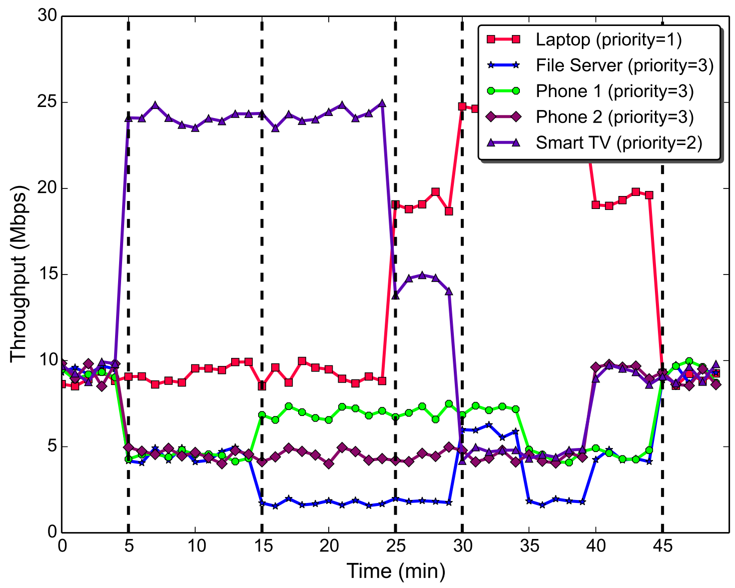


Figure 27: **Dynamic bandwidth allocation** using Securebox, where traffic from individual devices get bandwidth share depending on user preferences and priorities.

5.8 Dynamic Access Control

Sect. 3.1 discusses the ability of Securebox to dynamically restrict device communications in the network. Securebox can automatically allow/block device from communicating in the network, based on their activity and user preferences. Securebox also allows client to allow/block network access for any device using web/ mobile application. Figure 28 show one such example where device access is barred from communication in user network.

Figure 28 shows that initially all devices are allowed to communicate in the network but device 2 and 3 are blocked at $timestamp = 10$ and $timestamp = 30$ respectively, due to suspicious activity or user preference. Blocking these devices increased the network bandwidth share available to other three connected device. At $timestamp = 40$, device 1 is quarantined and again allowed access to the network, changing the bandwidth available to each device.

Securebox can also block devices from communicating to online servers depending on the security policies received from SMS. These security policies depend on the analysis performed by SMS, user preferences, context of Securebox. Figure 29 shows one such scenario where web access is blocked on user device due to some malicious activity or user preference. The device traffic shows the network access for the device is immediately lost as soon as

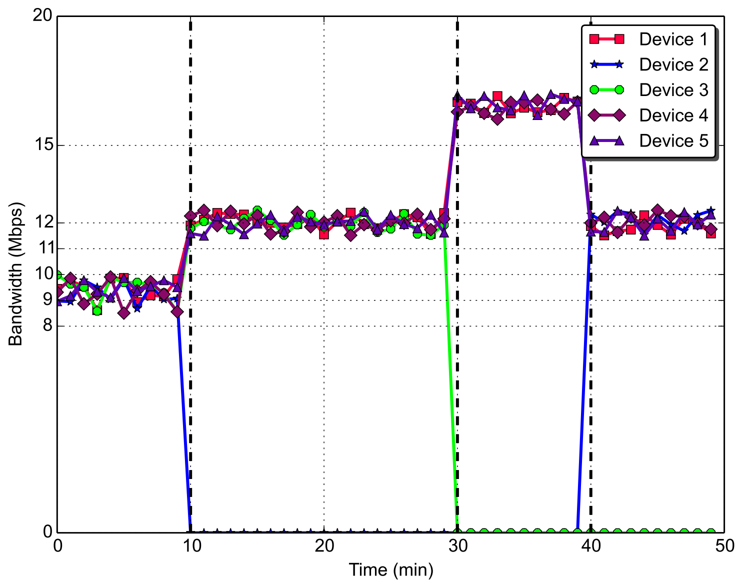


Figure 28: **Selective network isolation** using Securebox showing network access to individual devices can be blocked and released dynamically (due to security threats or user preferences).

the Securebox restricts device's access to the network. Network connectivity is restored once the device is quarantined and Securebox reconfigures the network policies.

Device and network connectivity restriction tasks are handled automatically by Securebox using SMS intelligence. Securebox also provides easy interface for users via mobile and web application to perform these tasks.

5.9 Cost Efficiency

Typically network middlebox costs from hundreds to thousands of dollars a piece, depending upon supported bandwidth, functionalities and processing capabilities. After initial installation, these middleboxes require a team of experts to configure and manage these middleboxes. The life cycle for each middleboxes is around 3-5 years after which they need to be upgraded.

With current advancement in technology and ever increasing bandwidth available, the update cycle is now reduced. These upgrades come at cost of thousands of dollars in terms of hardware upgrade as well as the trainings required for network security to upgrade their networks and operate new devices. Update cycles are also very slow due to number of requisition and planning phases involved.

Large enterprises and big organizations have enough resources to handle

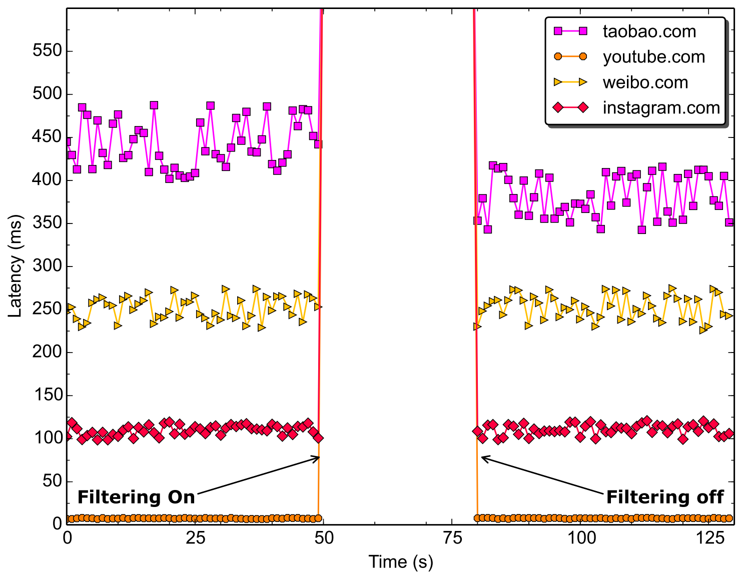


Figure 29: **Dynamic web(content) filtering**. *Securebox automatically blocks communications to untrusted remote destinations to prevent users from connecting to servers serving malicious content. It can also help in controlling phishing attacks by allowing users to connect to trusted services only.*

deployment and operational cost of specialized network hardware. However, small business and home users lack the resources to do similar arrangements (even at lower scale) for their own networks. Our system has therefore proposed a cost efficient way of resolving these models. Our setup only requires low cost ($\leq$ \$50) piece of hardware to be installed as network gateway.

If a service provider provides SMS subscription $\approx$ \$10 a month (average cost for Bitdefender Box [7], F-Secure Sense [30], Dojo [93], Cujo [25] etc.), user will spend $\approx$ \$600 over the course of 5 years to use the traffic analysis and management service. Users can also lower the cost of subscription if they operate SMS locally.

On the other hand, a basic firewall enabled router will cost $\geq$ \$300 with no software updates, management service, SLC support and frequent update cycle. The manual configuration overhead for such firewall greatly undermine their potential use in SOHO networks. A solution from Radware i.e. Radware Defense Flow [90] (similar to proposed system) for DDoS protection would cost $\geq$ \$10000. This solution is also designed for large scale enteprise deployed and has strong privacy related concerns.

Enterprises have a number of middleboxes of various kinds deployed in their networks which may cost more in hundreds of thousands of dollars [100]. These middleboxes are typically deployed at ingress/egress points of enterprise sub offices and network team can only administrate them at the granularity of enterprise sub offices.

This cost can be greatly reduced by using our proposed system as it requires a Securebox at each vantage point. SMS deployment would require experts to maintain and improve the system but the number of human resource required to maintain this setup will be greatly reduced as it can be operated (automatically) through a central management console.

The configuration update cycle will be smooth and less expensive as there is little to no human intervention required for configuration update of end devices. Hardware upgrades are also cheaper as the Securebox can be easily replaced and costs only a few dollars. The options available in small form factor PC market are always improving and prices are getting lowered, therefore new Secureboxes might cost less and be more efficient.

Enterprises can then deploy SMS in their own perimeter using public/private cloud. SMS service can also be deployed using enterprise servers instead of dedicated infrastructure. SMS can be readily updated to support new analysis and threat mitigation techniques. These updates does not require physical upgrade of network infrastructure i.e. Secureboxes, hence reducing the cost of upgrade by many folds.

5.10 Scalability

Flash crowds are common place these days and these situations can result in middleboxes becoming a bottleneck for the network [70]. These bottlenecks can cause serious business damages to the organizations. In order to minimize such instances of middleboxes becoming bottlenecks, extra middleboxes are deployed in the network. However, a great majority ($\geq$ 50%) of those middleboxes are under utilized at any given time of normal operations. This results in wastage of resources. Small and medium enterprises may not have enough resources to deploy such standby middleboxes for dealing with high traffic scenarios.

In order to deal with these issues, our proposed system offers a scalable model for traffic analysis without making middleboxes as network bottleneck. Using cloud or server-farm resources and virtual middleboxes, the system can automatically launch and take down middlebox instances. The number of instances of middlebox running at any given time depends on amount of traffic being analyzed. Using information from the global view of network activity, SMS can also predict the flash crowds and deploy more middleboxes for traffic analysis.

This scalable approach can also significantly cut down the operational cost of the network management setup because resources used for traffic analysis at any given instance are bounded by volume of traffic anlayzed. It also deals well with flash crowds scenarios where more middleboxes are readily deployed to analyze extra volume of traffic.

Figure 30 shows the three instances running for performing traffic analysis on incoming requests. Initially, all requests are handled by instance I but when the CPU load crosses pre-set threshold of 80% CPU usage, new analysis request are redirected to instance II. Therefore, the load is shared between two instances of analysis service. The CPU load for instance I, II again reaches threshold at *"Event 2"*. At this point, new traffic requests are redirected to instance III. These results show that system can scale efficiently to coup with the load of incoming traffic analysis requests.

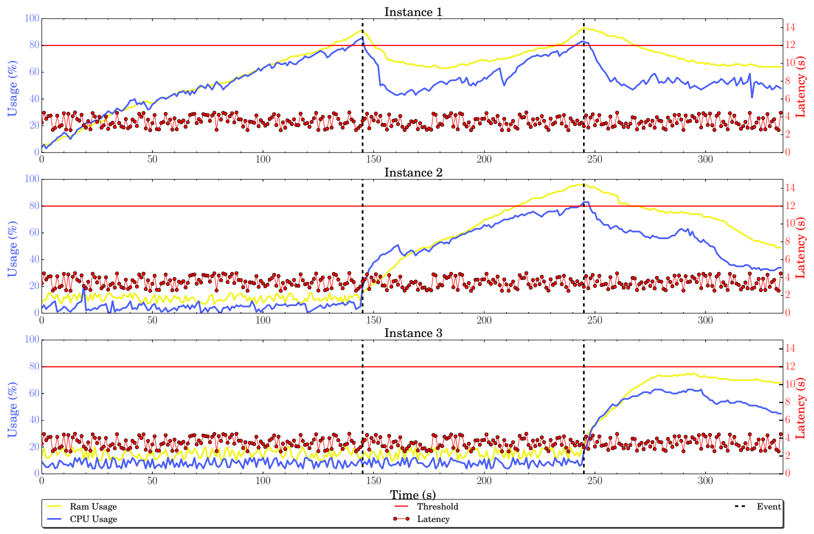


Figure 30: **Scalability in middlebox instances**, *Event 1: Instance 1 reaches threshold, so analysis load is (partly) redirected to Instance 2. Event 2: Both Instance I, II crossed threshold (80% CPU utilization), requests for analysis (partly) redirected to Instance III.*

For SOHO and small business users, our system's scalability can save business and upgrade costs i.e. if a SOHO user upgrades his network bandwidth, SMS will automatically scale network analysis services for the user. Sherry et al. also support our claim that using cloud resources to deploy middleboxes can reduce costs and improve scalability for enterprise security infrastructure [100].

5.11 Fault Tolerance

Sect. 4.1 discusses the backup instances maintained by SMS for OPMs. These instances replace OPMs if it goes down during traffic analysis. When there are more than one replicas running for an OPM, any of these replicas is promoted to OPM. If OPM recovers later, it starts acting as a replica, otherwise, a new replica is launched by SMS *"middlebox manager"*.

Figure 31 shows a master (i.e. OPM) and backup node. When master goes down, the backup node replaces it to become master and perform traffic analysis. Later when master recovers from failure, it synchronizes itself with current master and starts acting as a backup node. If the master does not recover, a new backup node is launched for current master.

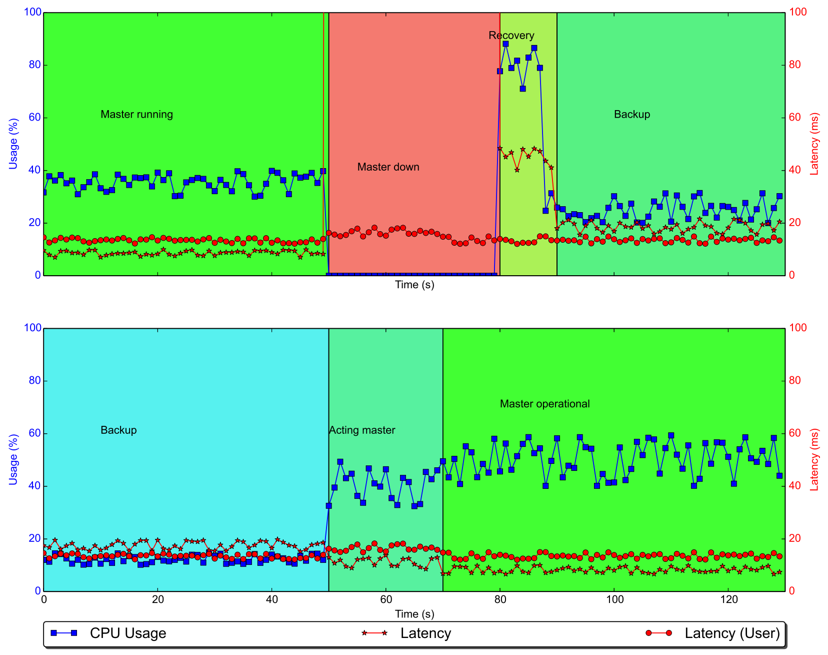


Figure 31: **Fault tolerance in middlebox operations using backup replica instances**. *When a master middlebox goes down during operations, SMS elevates its backup replica to become master and handle traffic analysis operations. If the master comes back up online later, it starts to serve as backup replica to new master, otherwise SMS launches a new backup instance for the new master.*

During this evaluation, we expect that running redundant copies of analysis services does not incur extra cost over the whole system. However, this assumption may not hold valid in scenarios with individual services running for hundreds of thousands of users. To address fault tolerance and scalability in such scenarios, we need to incorporate state of the art solutions to maintain state-aware replicas in the given system.

5.12 Robustness

The proposed system design allows it to improve its efficiency for network management and threat detection using the data collected by different networks via Secureboxes. A number of analysis services coupled with user's preferences and device profiling allow the system to identify the devices in the network and setup corresponding network configurations. However, the system should be robust enough to scale back its configuration changes from the system.

The system relies on number of analysis services for its operations and there is always a probability of false positives in the analysis results. In order to deal with false positives, a team of human experts can audit the configuration changes proposed by the system and remove any policies which result in deteriorating network performance. This external audit is also used as a system feedback to help it improve itself over time.

The system relies on individual services for threat, malware, botnet etc. detection. The techniques used for these purposes use machine learning which can also lead to false positives in the system [104, 87, 56, 98]. Such techniques have been constantly improving in the recent times to minimize false positives and we expect such technique to further improve in future.

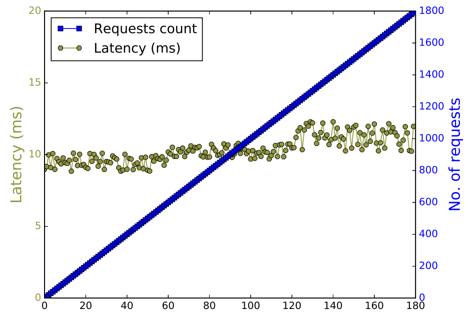
Although the system favors to minimize false positives using feedback from system, human experts, end users etc. but false negatives are even more important for providing better security. An attacker needs only one weak link to infect the whole system therefore the techniques achieving lower false negatives at the cost of false positives can be preferred because it is okay to block a link as precaution instead of serving a malicious link which would infect the user's devices and network. Therefore, bloom filters based malware and threat detection techniques can be particularly useful.

5.13 Infrastructure Security

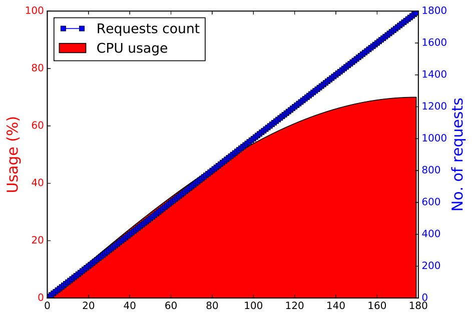
The proposed system moves the security and network functionality to a central entity and it is clear that a central entity is susceptible to different kind of attacks e.g. DDoS etc. The system also collects information from various networks segments and disseminate security policies based on this information. Therefore, rogue users can craft special analysis request for the system which trick system into generating policy updates directing all Secureboxes to block all traffic to/from the network.

Figure 8 shows a number of components in SMS architecture to deal with any attacks against SMS. Central server receives all the requests from Securebox, making it a primary target for DDoS attacks. In order to increase its resilience to these attacks, SMS maintains a state-aware replica of *central server* which would replace central server immediately if it goes down. The service can also use a distributed central server making it more efficient and resilient to attacks. Virtual middlebox manager is responsible for inspecting middlebox instance utilization and incoming traffic load. In case, a middlebox is over utilized, new instances are launched to share the traffic analysis load.

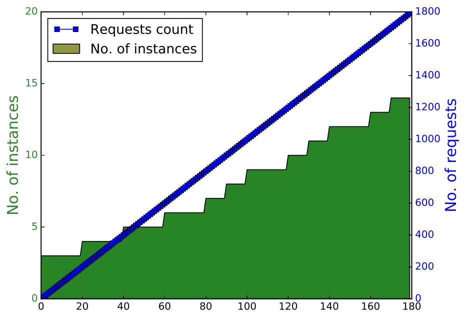
The *certification authority* is responsible for maintaining Securebox and user certificates. Every incoming request is verified by the certification authority to make sure that requests from only legitimate sources are processed by the system. The certification authority receives information about user subscription and preferences from user, device management services to limit the scope of traffic analysis request.



(a)



(b)



(c)

Figure 32: **Simulated DDoS attack by flooding traffic anlaysis requests to SMS.** *(a) Latency experienced during the simulated attack on SMS remains constant showing that end user does not experience significant delay due to service outage during an attack. (b) Average CPU usage for all instances shows that SMS scales well in case of increasing traffic analysis load to maintain the latency experienced by the end user. (c) As the traffic analysis load increases, SMS launches new instances to keep CPU usage inbound and prevent service outage due to overloaded instances analyzing incoming requests.*

Anomaly detection engine shown in Fig. 8 inspects operational logs for the whole system to detect any anomalous behavior in traffic analysis request or system components. These logs are also analyzed by human experts and the information combined from automated and human analysis is combined to block any misbehaving system components or Securebox.

In order to prevent spread of rumour-based policy updates by rogue (acting like SMS) Secureboxes in the system, a Securebox only accepts policy database updates certified by SMS. In order to prevent man-in-the-middle attacks, all communication between Securebox and SMS is encrypted using the state of the art used for web traffic encryption.

We have also tested our system performance against a simulated DDoS attack which is launched by bombarding the system with a number of analysis requests. Figure 32 shows the performance hit taken by the system in case of a DDoS attack. We have monitored system latency and CPU usage to evaluate the ability of system to tolerate such attacks. Figure 32a shows that the system is able to maintain only a slight increase in latency during the simulated DDoS attack. It shows the proposed system has ability to maintain latency in bounds, during a reasonably sized attacks.

Figure 32b shows the internal statistics about traffic analysis instances. It shows that the CPU overhead increased linearly with the number of incoming requests. Figure 32c shows that SMS also increased the number of new instances as the computational load increases. It would prevent overloading of instances analyzing incoming requests. Therefore, the system was able to divide the incoming traffic load across all these instances to bound the CPU usage load and latency experienced by the user.

5.14 Discussion

The results presented here are obtained using the data collected from the prototype system in university laboratory setup. During the experimentation, we tried to simulate the behavior of an average house hold user to get better estimate of system performance in terms of user experiences. However, these results can vary in real world deployment setups depending on the mode of deployment. For example, the communication latency between Securebox and SMS is very low $\approx 1.56(\pm 0.1)ms$ compared to real world setups where SMS is deployed by a remote service. During evaluation, we have tried to model the number of simultaneous traffic flows in the networks equivalent to what we expect in a common house hold with 10-15 devices. The performance of system can be affected by the increase in number of simultaneous traffic flows in the network. The layout of network was also static compared to public Wi-Fi or enterprise networks where devices join/leave frequently. We have not evaluated the performance hit for the system in such scenarios. Similarly, the system was also not evaluated for the out-of-band devices communications where IoT devices can directly influence other devices to perform an action e.g. *"upon sensing an increase in room temperature by* *temperature sensor, alarm rings and sounds of alarm results in activation of fire safety system"*. Currently, we have a performed limited evaluation on securing D2D interactions within user network. We expect that this area will be explored to evaluate system performance in securing D2D interactions with single level or multilevel cross-device dependencies.

6 Features and Use Cases

The proposed system design is motivated by the need to developed a unified platform which can be deployed across a number of different scenario ranging from SOHO to enterprise network. The flexibility in deployment of proposed system allows the system to get a comprehensive view of disjoint network segments and use this global view to improve security situation in all those individual network segments. In this chapter, we discuss various features and use cases of our proposed platform including SOHO and enterprise scenarios. We highlight the advantages and limitations of using the proposed system in each of these scenarios.

6.1 Device Discovery and Profiling

The proposed system uses device discovery and identification mechanism for improving network security and management. With growing popularity of IoT and smart devices, typical networks are expected to connect a number of devices. These devices can be specialized to perform different activities and average users may not be able to characterize these devices based on their network activity. Sect. 1 explains how these devices can be can be vulnerable and raise different security issues.

In order to deal with the security issues, the proposed system uses a dynamic access control mechanism. Securebox acts as sensor and enforcer of this dynamic control mechanism and SMS acts as the control plane deciding what access control should be applied. In order to minimize the solution's reliance on human efforts, our system automates the task of device identification and security profiling. However, the automated services are always take suggestive or preventive measures only and their actions can be overridden by user's choice.

Device discovery mechanism allows the system to identify any devices connected to the Securebox. Once the device is identified, Securebox can obtain device specific policies from SMS and enforce them in the network. Device discovery mechanism is particularly useful to secure networks where unknown devices are frequently connected to the network. Using this mechanism, Securebox can actively prevent any attacks due to vulnerable device connected to the network.

SMS maintains a database of all identified devices and related security threats. This database is maintained by combing information from third parties, vulnerability databases and human experts. This database also contains devices related activity fingerprints, user registrations and data from device manufacturers, which is used to identify the devices when they are connected to the network. This information is also updated periodically using new device registrations and other data.

Securebox can use a variety of methods for device identification. The proposed system requires users to register their personal devices with SMS for using device and context specific services at different Securebox. Device registration requires a user to give some information about the device and user preferences. This information is stored along with user profile and later used to identify the device when it is connected at different Secureboxes.

For the devices which are not registered to the Securebox e.g. in an enterprise guest networks, where unknown device connected to the network. Securebox can use device activity to recognize the devices. For this purpose, SMS needs to maintain a database of device activity signatures. These signatures can also be obtained from user's registered devices and user anonymity can be ensured by removing any user related information from these signatures. When a device connects to the network, Securebox analyzes its network activity to obtain a signature for device activity, which is used by SMS to identify the device and enforce required policies to the network.

SMS also allows users to setup device profiles which is a quick and handy way to set up device preferences. Each of these profiles include a set of preferences e.g. *"parental control"* profile will include all preference which are required for parental control setup. Users can dynamically set these profiles on any of the devices. As soon as the device profile is updated, SMS will generate a policy database update for Securebox where the device is currently connected so that device activity is limited by the profiles preferences.

Device profile reduces the effort to individually setup and update policies for each of the user registered devices. This is very useful in enterprise scenarios, where the number of device is huge and network management team can easily update or limit the access to set of devices by updating device profile preferences.

6.2 AP Management

The proposed system design support automated access point management services. The system provides allows user to specify security and management preferences for their registered Secureboxes. The system also allows the user to setup securebox profiles, context specific policies for different secureboxes and SMS ensures that these preferences are implemented at all Secureboxes.

When users deploys a new Securebox in their network, they register it in their user profile and setup its preferences (if any). SMS uses these preferences to setup initial policies on the Securebox and all subsequent operations are performed based on these preferences e.g. if the network manager sets up a policy that *"all traffic from Secureboxes deployed in meeting room should be analyzed by Firewall and DPI"*, SMS will setup the policy database update for Securebox such that its traffic should be redirected via the specified middleboxes. Similarly, if a user specifies that any Securebox at their home should serve active parental control for all devices registered as kid's devices, SMS will use device information and generate a policy update which makes sure that all traffic from specific (kid's) devices should adhere to parental control guidelines.

This feature allows the system to setup context specific preferences (policies) at Securebox. SMS also updates the management policies at all Secureboxes based on the analysis services. SMS also monitors the traffic analysis requests and other activity from a Securebox and actively blocks any misbehaving Secureboxes to prevent any attacks against SMS itself.

6.3 Device Level Data Cap

With increasing utilization of video content over the Internet and fixed data plans, it becomes frequent for data users to over utilize their monthly bills resulting in overcharging for extra data used. Our system also supports data cap management for individual devices. It allows users to setup data caps for each of their devices to make sure that their monthly data plan usage does not cross the limits. Securebox collects the data used by the device and sends it to SMS where data usage record is maintained on per devices basis.

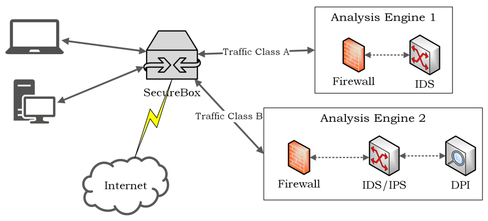
User gets notifications about the device data usage once it crosses pre defined limits and takes any measures based on user preferences e.g. *"Block a device if it exceeds its quota for data usage"*. The system allows to implement different QoS for devices depending on their data usage e.g. *"if the device uses* 70% *of its quota, it's bandwidth should be slashed by* 30%*"*. Since the information is stored in SMS, similar QoS can be experienced at all Secureboxes connected to same SMS.

The information about per device data usage is also useful for the user to understand device utilization pattern. Research has shown that users appreciate to get information about data usage of their individual devices and it helps them handle their data plans better [18]. Some users are also able to find anomalies in device and application functionality due to unusual data usage patterns e.g. *"Based on the data usage, a user can identify if its TV is streaming any content to some external server at night times (when TV is not being used)"*. Similarly, this feature helps users to identify any malicious applications on mobile devices or malicious IoT devices by looking at their data usages.

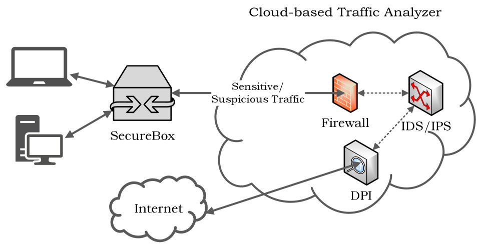
6.4 Dynamic Traffic Analysis

The proposed system allows the user to request traffic analysis by combining different services in desired sequence. SMS can itself process the traffic through a series of middleboxes if required. This feature is similar to service chaining concept where for bundling different application services together to develop a dedicated chain of services for processing tasks.

These middlebox and traffic analysis services are chained together according to context based preferences from the user. It can be used to manage the traffic from two different devices coming through the same Securebox (AP) e.g. traffic from enterprise's own registered devices connected to conference room Securebox will be processed by middleboxes processing enterprise middleboxes but traffic from a guest device connected to same conference room Securebox should be passed through firewall first and then processed through a separate set of middleboxes.



(a) Two different classes of traffic are processing in separate analysis engines with specific traffic analysis chain.



(b) All traffic from Securebox is re-routed via SMS based middlebox deployment. The string of middleboxes for traffic processing can be dynamically updated depending on traffic context and user preferences.

Figure 33: **Subscriber control over traffic analysis operations**. *A user can dynamically choose (modify) the services being used for traffic analysis and steer traffic through one or more (different) middleboxes.*

Figure 33a shows the case where user has configured that all traffic from IoT devices (including smart TV, smart fridge, etc.) should be classified as class A and rest of traffic as class B. Based on user preferences, class A traffic should be handled by analysis engine 1 consisting of Firewall and IDS service. Whereas, traffic from class B (i.e. traffic from user smartphone, PC, tablet) should be handled by analysis engine 2 which processes this traffic through a Firewall, IDS and DPI service.

The user can also set the context based traffic processing for situations such that traffic should only be processed in DPI if suspicious behavior is detected for any device. Dynamic service chaining allows user to save costs

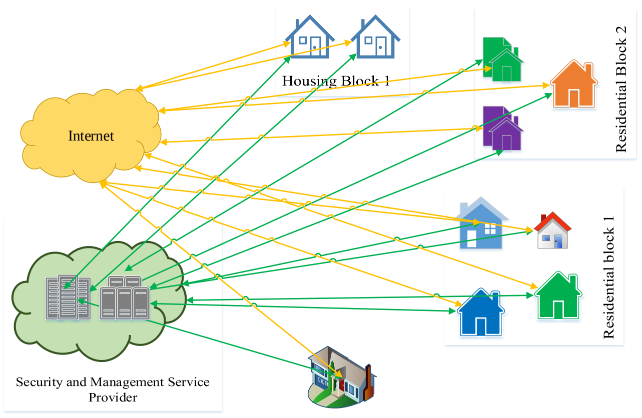


Figure 34: **Securebox deployment in SOHO networks**. *Each home can have one or most Secureboxes installed with each Securebox using SMS services subscribed from some third party service provider.*

of traffic analysis. It also provides more control over how the subscribed services are used to control traffic processing and analysis.

Figure 33a shows the scenario where user traffic is tunnelled through a set of middleboxes before going to the Internet. It is common place scenario for network deployments where in-line middleboxes process all incoming and outgoing traffic at gateway points. Using service chaining, users can manually or automatically change the sequence of middleboxes used in this chain for traffic processing.

This is particularly useful for enterprise scenarios where automated setup can update the string of middleboxes processing the traffic. Similar concepts have been previously introduced, allowing the user to redirect their traffic through remotely deployed middleboxes but those ideas require user involvement to identify malicious activity and configure their network to re route their traffic. Our system provides automation to this task to make the re-routing and dynamic middleboxes deployment easier and faster.

6.5 Small and Home Office Networks

SOHO networks are very crucial in overall network security scenario as discussed in Sect. 1. Securing these vulnerable networks with millions of connected devices was one of the primary motivations to build Securebox platform. The deployment is fairly easy as the user needs to install a Securebox as a gateway to the network or as an AP. These Secureboxes are either obtained from a service provider as part of subscription bundle or purchased as a separate device to be used with any third party SMS.

If Securebox is obtained as part of subscription bundle for SMS, it is already configured and user only needs to install it in their network and Securebox should be able to communicate with SMS. User can then use the mobile or web application to set up their security, management and device preferences. Securebox will receive policy database update as part of bootstrap operations and request SMS for any kind of anlaysis operations required. The data collected from user network is also sent to SMS based on user subscription and preferences.

This setup provides automated network management for the user. Using SMS device identification services, Securebox can detect what devices are connected to the network and implement any network policies specific to those devices. SMS also maintains a database for device related threats and security issues. When a device is detected in user network, SMS generates email or mobile app notifications for the user about the device related threats and any suggestions to mitigate these threats. This information enables users to protect their data and privacy.

Using virtual middleboxes to analyze traffic and automated configuration update, the system provides enterprise grade security for SOHO networks at a fraction of that cost. These traffic analysis service also prevent users to connect to any malicious servers which host malicious content resulting in compromising user's device to become part of a botnet, spam net. Ransomware have caused losses of millions of dollar for home users and it is very crucial to protect home users from these ransom-ware and trojans [8]. By actively checking all sources and destination of user traffic, Securebox greatly limits phishing and ransom-ware attacks as it prevents user from connecting to any illegitimate servers.

Studies have shown that typical users care about the protection of their devices and privacy of their data [references]. However, typical networking gear provide little or no feedback to users about the safety situation of their network. Anti-virus applications also provide low level information about the threats detected and blocked. Our system intends to raise user awareness about their security and privacy by providing them high level threat information in human understandable language e.g. *"Webcam in smart TV can be hijacked to stream your video to remote destination"*.

This kind of feedback in form of mobile notifications keep users updated about any threats in the network. A notification alerting user e.g. *"Downloading any content from this site may infect your device"* when it tries to contact an untrustable site hosting some content will help user understand the risks of downloading content from third party website. If an application tricks user to download some content from a source other than legitimate publisher, the system prevents such redirections and notifies the user about misbehaving application so that user stop using it. This information is also used to alert other users of the service to stop using misbehaving (potentially malicious) application.

6.6 Enterprise Networks

The proposed system decouples the enforcement plane from control plane to increase its deployment flexibility to enterprise and large organizational networks with multiple offices. Traditionally, large organizations and enterprises with multiple sub offices deploy middleboxes at each sub office and the traffic is directed to organization's central network before going out on Internet. This setup requires huge deployment costs as it requires middlebox deployments at remote sites and sub-offices.

Additionally, network management team has to maintain both these remote middleboxes and central network which serves as gateway for organization's Intranet to the Internet. In case of any issues with network, management team has to individually update each of these Secureboxes at all sites, which can become a tedious and time consuming. Research has shown that manual configuration updates by human experts can often lead to security loopholes [34]. Any one of these loopholes can grant attackers access to organization's internal network, which can cause serious damage to organization/enterprise.

Our proposed system tries to cut down costs for enterprise network deployment by replacing middleboxes with lightweight Secureboxes, see Fig. 8. Figure 35 shows that each department, subdivision is connected to enterprise network via Securebox. All these secureboxes are controlled by SMS which is either deployed by enterprise itself or it is subscribed from a third party service provider. In-house deployment will give more control to enterprise over how the Secureboxes are configured, what information is collected and how network is operated?

These Secureboxes provide more detailed view and discrete control over the network as they can identify end devices connected to the network. It also provides central control over all vantage points in the network. The automated configuration updates from central management platform will minimize inconsistencies and delay in configuration updates across the network. Using in house SMS deployment for Securebox management, the networking team can easily deploy and test new traffic analysis services in the enterprise without the need to update/upgrade any of the Secureboxes.

Our proposed system requires less human personal to operate the system by automating most of routine management and configuration tasks. It also assists security experts by automating log analysis and anomaly detection mechanism. It also improves the scalability of middlebox deployments and prevent them from becoming bottlenecks in flash crowd scenarios. These improvement have been discussed in detail in Sect. 5.

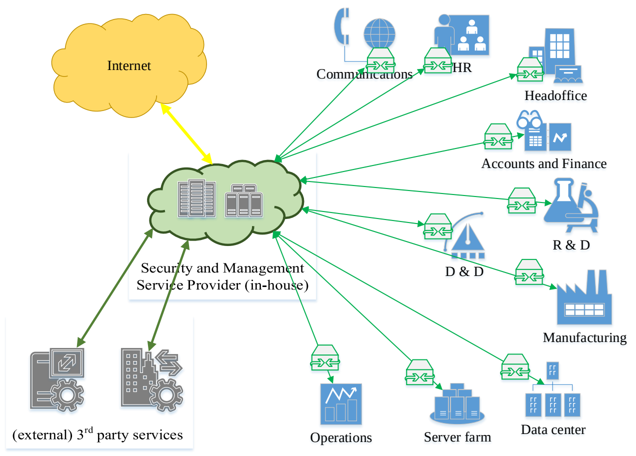


Figure 35: **Securebox deployment in enterprise networks**. *Enterprise network management team can use external SMS for managing these secure-boxes (provided by hardware vendor or third party service provider). The can also deploy SMS locally to have more control over network services and network data collection.*

6.7 Secure Wi-Fi Environments: Password free Wireless Networks with Trust Levels

With growing popularity of smart, mobile and IoT devices, wireless has emerged as the primary mode of connectivity for such devices. In order to connect to a user's network, these devices typically need to authenticate to the network's Wi-Fi access point. These wireless APs typically use `WPA-PSK` based authentication, which requires the devices to be authenticated using pre-shared keys (PSKs), as shown in Fig. 36. `WPA-PSK` based authentication is widely popular choice for setting up Wi-Fi networks because it does not assume any security associations between users and device manufacturers and is relatively to setup.

6.7.1 Limitations of WPA

The primary draw back of this technique is that authentication reliability is dependent on the confidentiality of `PSK`, as any entity with access to `PSK` will be able to authenticate itself with AP. Currently, the number of devices associate to Wi-Fi networks is relatively small for typical SOHO networks. With the emergence of IoT devices, the number of connected devices will grow by an order of magnitude. These devices are developed by fast moving teams in large enterprises or independent start-up teams who have limited resources and hard deadlines to launch their devices.

Therefore, security get little to no consideration in product design and

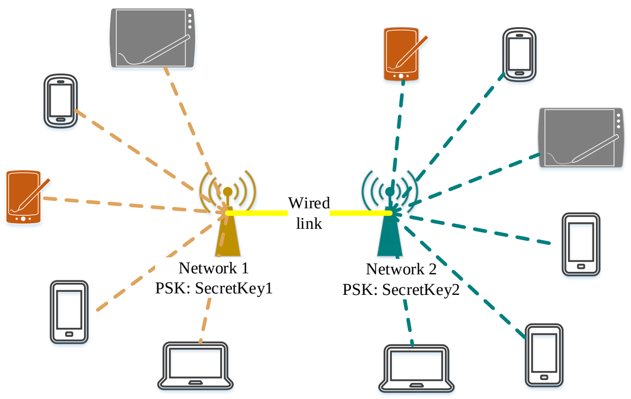


Figure 36: **Typical WPA-PSK based wireless network setup**. In these setups, each device is authenticated using same PSK.

development process. Mostly, these devices do not have a device life cycle support, which leads to likelihood of having many IoT devices with a number of security vulnerability and no support cycle to provide security updates or software patches [72]. Attackers and hackers can exploit these vulnerabilities obtain the `PSK`, this compromising the security of device authentication [81, 116, 119].

Using one PSKs for setting up Wi-Fi networks has many other issues undermining the reliability of using `PSK` as network security parameters e.g. if an employee leaves the enterprise, the `PSK` needs to be updated, which means re-associating all devices with new `PSK`. It can be a tedious task to perform in large networks and become more troublesome if happens frequently. Similarly, the PSKs typically used to setup SOHO networks are not very strong and can easily be broken by an attacker [109]. Since Wi-Fi connectivity is not bounded by physical parameters, attacker can get authenticated connectivity to the targeted network without need to break into target premises. Once attacker devices authenticate to target network using compromised `PSK`, it can snoop on other device activity, infect other devices and perform a number of other attacks [22].

The compromise of `PSK` confidentiality can have adversely affect the security of targeted network, as an attacker is able to join the network and attack other devices in the network. Therefore, we require a Wi-Fi deployment framework which resolves the security issues raised by compromised PSKs. `IEEE 802.1X` system allows Wi-Fi networks to setup a `RADIUS` server for managing client and device authentication. Such a setup uses user credentials to allow connectivity for a device once it joins the network. However, this technique requires setting up authentication and authorization servers in the network and may not work with typical APs deployed in SOHO environments.

6.7.2 State of the Art

Another technique to mitigate this problem is to user device specific PSKs to connect different devices to the network. Ruckus networks have proposed a patented scheme to use device specific Dynamic PSK (DPSK) [97]. This setup generates dynamic keys for each connected device and updates them automatically on user devices. However, this technique only works with proprietary hardware from Ruckus networks. Also, it requires to setup a authentication service and an application on smart-phone or laptops to operate. Therefore, its usability in greatly limited in IoT case and typical SOHO networks.

Aerohive networks have also proposed a similar scheme of using private PSKs (`PPSK`) [73]. This scheme allows the network manager to generate a set of PSKs which are used for associating devices (one device per key). This technique also works only with proprietary Aerohive APs and Hive Manager. Both these technique provide support to revoke key if the key or device is compromised, but they do not provide support to setup dynamic access control based on device behavior. Therefore, an attacker can still successfully impersonate any device and use the compromised key to authenticate with the network and infect other devices.

6.7.3 Proposed Solution

Using our proposed system, we provide a solution for network deployments which can resolve security issues raised by compromised PSKs (including DPSK, PPSK). Our system utilizes PSKs and other techniques for device identification and dynamic access control to limit device connectivity in the network. In most cases, IoT devices have only a specific functionality which requires only limited network interactions for operations e.g. a smart kettle requires to connect to smartphone for getting coffee making instructions but it does not need to connect to any other device in the network e.g. smart fridge or smart TV.

Similarly, a smart doorbell only needs to be able to connect to smartphone application but does not need any connectivity with smart kettle or any other device inside home. Using this information, our system can limit the device interactions with other devices in the network.

In order to identify the device, we can use a number of parameters including device registration, device activity fingerprinting and device specific unique PSKs. Device registrations are performed by users when they first connect a device to the network. Based on user preferences, no unregistered device should have access to any device in the network (or not network access at all). When the device is registered, a unique PSK (`uPSK`) is generated for the device to connect to the network, see Fig. 37. Different techniques to identify a device and its profile are discussed in detail in Sect. 6.1. Device

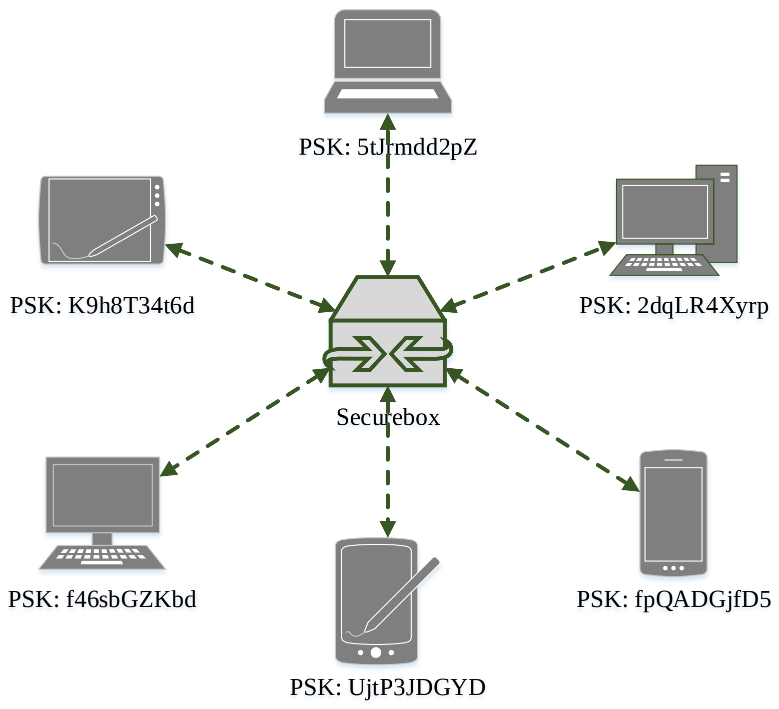


Figure 37: **Securebox based network deployment using uPSK**. *uPSK based scheme uses unique keys for each of the device. These keys are used to identify the device and limit device access to the network as well.*

specific `uPSK` will also become on of the parameters to identify the device.

When Securebox identifies a device, it requests SMS to send policy database update including device specific security policies in the network. These policies limit the device interactions to any other devices and limits its access to the Internet e.g. if user sets up a CCTV camera at his home, SMS will send policy database which would direct Securebox to *"not allow any IoT device to connect to CCTV camera from user network"* and *"donot allow any connections to CCTV camera to/from Internet"*. User can specify the file server to record video feed from CCTV camera and that policy will have higher priority then SMS's injected policies. Therefore, CCTV will be able to connect to user file server but not able to connect to any other server on the Internet.

Dynamic access control greatly limits the attackers ability to infect other devices in the network, even if it gets access to `uPSK`. Firstly, the attacker will not be able to authenticate to the network because Securebox can detect duplicate authenticate connection request using same uPSK and consider it as anomaly. Securebox will raise an alarm and notify the user about

Table 8 Feature comparison of device specific key based solution.

Features	**DPSK** [97]	**PPSK** [73]	**uPSK**
Device specific keys	✓	✓	✓
Proprietary technique	✓	✓	✗
Require proprietary hardware	✓	✓	✗
Secure D2D communication	✗	✗	✓
Auto update of keys	Limited	✗	Limited
Device profiling	✗	✗	✓
Device vulnerability assessment	✗	✗	✓
Support keyless wireless environments	✗	✗	✓
Supports software-defined Wi-Fi	✗	✗	✓

possible key compromise and replace compromised `uPSK`. The new `uPSK` is automatically updated on destination device if the device supports running software application e.g. smartphone, laptops or desktops. Otherwise, user can update this key manually on IoT devices.

Secondly, if the attacker takes down the device from the network and use compromised `uPSK` for that device to authenticate to the network, Securebox will be able to distinguish attacker's device from the device whose `uPSK` is compromised. Once again, Securebox will detect the anomaly and `uPSK` will be updated.

Thirdly, if the attacker is able to impersonate the device well enough to trick device identification mechanism, the access granted to attacker's device will be greatly limited due to dynamic access control. This limited connectivity will prevent attacker from infecting other devices in the network. If the attacker tries to perform aggressive network attack, Securebox will detect this activity, block the device from network and notify the user.

Table 8 provides a comparison of features offered by Aerohive PSK and Ruckus DPSK based technique with our proposed `uPSK` technique. It shows that our proposed technique provides a number of additional features to secure D2D communications within user networks. These features are helpful in cases where an infected device or an attacker tries to connect to user network and infect other devices in the network.

In some cases, Wi-Fi networks can be setup without requiring any `PSK`. Such networks will provide open connectivity to any device without requiring prior authentication. Once the device is connected to the network, Securebox runs a security estimation for the device. Combining security estimation along with device registration information, Securebox will generate a trust index for the device. The trust index will be used to setup device specific policies in the network. These policies will dictate the network access level for the device. The proposed system can support this mechanism as it allows Securebox to implement device and context specific policies on Securebox.

6.8 SWEN: Software-defined Wearable Networking

Wearables are becoming increasingly popular in recent times. These low-power devices can be used for activity tracking, fitness tracking, health monitoring [132, 15, 43, 76]. Recent research has shown promises to bring connected clothes and accessories for users [103]. These wearable collect user information and relay this information to user's smartphone or tablet via BLE. A companion application in user's smartphone relays this information to cloud-based services which provides various kind of suggestions and feedback to the user.

The information collected by these middleboxes is sensitive because it can be related to user health or his activity. Since, these devices can be paired with mobile applications, an attacker can trick user's wearable to leak monitoring data [117, 69]. Therefore, we propose a new approach *"Software-defined Wearable Networks"* (SWEN) for securing communications to/from these wearables.

6.8.1 Design

SWEN leverages service based design from our proposed system for securing user networks. It uses a software application to setup an adhoc network using BLE, Wi-Fi, etc. for wearable devices to communicate securely. SWEN uses contextual information to improve the security of the system. Figure 38 shows the high level design of SWEN where *"SWENBox"* resides on user's smartphone or a lightweight (similar to wearable) component. It provides the user interface to manage device and security preferences. It uses application logic and contextual information to support secure interactions between devices. The contextual information can be collected from various sensors available at the smartphone or wearable device.

The system also uses an external security which provides security services for wearable. This security services is responsible for management of device profiles, users information and trusted D2D interactions. It also provides control to users to setup custom preferences over how their wearable should be managed and secured. The remote service allows us to provide different kind of security services for these wearables.

6.8.2 Goals

The goals of designing SWEN is to establish trust between these wearable devices using collaborative sensing. This *"big trust from little things"* is useful for sensing applications. The system is designed to support secure pairing between devices and handling devices association in order to prevent

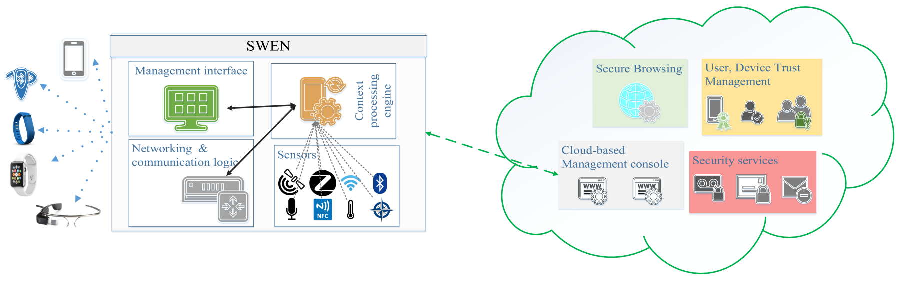

SWEN
Management interface
Context processing engine
Networking & communication logic
Sensors
Secure Browsing
User, Device Trust Management
Cloud-based Management console
Security services

Figure 38: **Architecture of SWEN**. *SWEN can be deployed as smartphone application or using a wearable whereas the remote service can be deployed by an external service provider.*

an attacker with communicating with user wearables. The system achieves this by monitoring all communications among wearables and other devices within adhoc network. This adhoc networks provides a medium for seamless and secure D2D communication among devices that are authenticated by the remote security services.

SWEN design also prevents relay and MITM attacks by allowing devices to communicate within trusted networks. Since the wearables will only communicate with/via SWENbox using hardware specific certificates, an attacker can not pretend to be SWENbox for any wearable device. Therefore, SWEN is able to achieve the goal of protecting user data privacy and secure sensing with wearables.

6.8.3 Features

SWEN offers a number of features in order to achieve these goals. It uses software-defined networking logic with isolation and second factor authentication (for devices) support by using contextual information of communicating devices. The trust between communicating devices can be achieved by using remote analytics service which supports context based trust assessment of wearable and other devices. SWEN also supports *"contextual fencing"* to provide context based access and services for any devices. *Contextual fencing* combines user preferences, security and trust assessment, and context information of the device to manage access for device communication and other services.

SWEN is useful to secure the interactions between trusted and untrusted (unknown) IoT devices by offering selective isolation for D2D communications. This kind of isolation is akin to the selective isolation provided by Securebox for D2D communications within one network.

6.8.4 Deployment

SWEN is designed as a software module which can leverage on the sensor information of host device and work with a remote service to provide various services. It is hardware independent and can be deployed as a smartphone application or a wearable device with reasonable power resources e.g. Omega2 [23]. As a smartphone application, SWEN will be able to use information from different sensors and applications (for wearable) to better administrate user devices. It will also be easy to deploy as nearly every wearable user carries a smartphone. On the other hand, deploying it on independent hardware will give it more control over functionality (unbounded by underlying OS) and provide enhanced features. The external security service can be deployed by the user using his personal machine or it can be provided by an external security service provider. Individual deployments will offer more control over service functionality and modules. On the other hand, third party deployments will increase the efficiency of service offering better analytics, trust services.

7 Discussion

We have explained design and architecture of Securebox and SMS in Sect. 3 and Sect. 4 respectively. The thorough evaluation of the system, give in Sect. 5, shows that the design level choices for the proposed system had a significant impact on system performance and efficiency.

System evaluation shows that service based model for providing security as a service alternative to deploying in-line middleboxes in the network is a practical solution. It offers many folds advantage including cost efficiency, deployment efficiency, scalability etc. This model provides enterprise grade security to SOHO users. It is very important to secure these users as they form a significant percentage of victims of cyber crimes [63]. We have developed this system as a software based services so that it can be easily modified to deploy in different kinds of environments.

The proposed system can be deployed as a smartphone or PC application, acting as a software firewall. This application will inspect all the traffic to/from the device to secure the device connectivity. The dynamic design of SMS allows the service to support different client applications, may it be a hardware-based securebox or software application, as it is designed to provide intelligence to the software/hardware enforcers.

The choice of using policy database updates has various advantages. It significantly reduces the latency overhead and improves users experience with the proposed system. It also allows to minimize the overhead on SMS for (re)processing similar traffic request. The ability of system to support software based virtual middleboxes is very useful for all users as it alleviates the need of deploying, maintaining and constantly updating those middleboxes. The system also supports a number of user and device level services. These services allow the system and user to control D2D communications which is very handy feature for security in Wi-Fi or IoT environments. All these advantages and their impact on user experience are discussed in detail in previous sections.

7.1 Limitations and Future Work

Besides all these advantages of proposed system, there are some limitations as well. The key limitations of the system (for user's experience) are increased latency and user privacy.

7.1.1 Latency

Sect. 5.1 shows that the primary source of increase in this latency is the time taken to analyze user's traffic in software based middleboxes. Therefore, the latency experienced by the user is directly affected by his choice for traffic analysis. However, software based middleboxes can be improved to further decrease latency. Sect. 2 presents various promising solutions for virtual middleboxes, which can be further improved to increase their efficiency to become comparable with traditional hardware based middleboxes. Our work leverages software middlebox work from existing research and focuses on system design to decrease overall latency by minimizing the interactions between Securebox and SMS.

The latency offered by our system can be further improved by improving the communication model between Securebox and SMS. We use a central database in SMS to combine all analysis results, which are then relayed to Secureboxes via policy database updates or used to respond to user queries instead of (re)analyzing similar traffic. The design of this database can be further improve to decrease lookup times. The design of software based middleboxes can be improved to make them more efficient for large scale collaborative traffic analysis operations.

7.1.2 User Privacy

Since the system does remote analysis of user traffic, privacy advocates claim this to be a breach in privacy as the user's internet activity can be monitored by the external service providing security and management of the network. The proposed system uses trust-based model to mitigate these concerns, just like any other service e.g. Gmail [16] or anti virus services. All services collect information about the user but they have explicit agreement with the subscribers to use this information. We propose the same model for deployment, where user explicitly agrees on terms of data collection and corresponding services. The proposed system also offers dedicated middleboxes, which allow users to analyze their traffic in isolated environments, without sharing information with the system. In order to further improve privacy preserving analysis techniques can be designed. These techniques will allow the system to share user traffic information with traffic analysis services without leaking any information about the user himself.

7.1.3 Fault Tolerance

Figure 31 shows that our proposed system can provide fault tolerance using backup replicas. However, maintaining these state aware replicas can be an interesting problems. There are two different ways to address this problem. One solution is to analyze the traffic simultaneously in both master and backup instances. This approach will increase the system load for processing the traffic but also increase the ability of system to detect inconsistent middleboxes. Another solution is to share state information from master to replicas but the efficiency of this approach is limited by the frequency of information shared between the master and replicas. Currently, we use

[16]https://www.gmail.com

use `Kubernetes` fault tolerance and scalability mechanism to assist fault tolerance in proposed system but this problem can be further investigated to improve system fault tolerance in real world deployments.

7.1.4 Device Identification and Profiling

The proposed system uses device identification and profiling mechanism to automatically detect any devices connected to the network. Once these devices are connected, Securebox can automatically install relevant security policies in the network, depending on the type of vulnerabilities for the devices. This mechanism needs two different kinds of services i.e. *"device discovery assistance service"* (DDAS) and *"device vulnerability database"* (DVD).

There can be different techniques used to discover new devices. The device registration information can be used to detect the devices which are registered by users with SMS. However, in order to detect (new) devices, their network activity fingerprinting, hardware information can be used. DDAS supports these device discovery techniques by providing support services and related information for automatic classification of devices.

DVD on the other hand stores vulnerability information about the devices. It collects this information from various resources e.g. `CVE` publishing systems, security researchers, testbeds. This information is then used to profile these vulnerabilities and design security policies to mitigate any threats to user privacy and security due to these vulnerabilities.

Both DDAS and DVD require robust mechanisms for their functionality and raise many interesting questions about how to improve the design and efficiency of these services? how to profile IoT vulnerabilities? how to automatically create security policies for a vulnerability? All these research questions can be explored in the future work and their solution can be valuable contributions for IoT and network security. The proposed system can also use external services and data sources to improve systems ability to prevent any attacks to user privacy or network due to device vulnerability.

Our current prototype is a proof of concept model showing that the system works in the real world environment. Based on our evaluation, the system provides an improved, cost efficient and scalable way to improve network security situation in SOHO and enterprise networks. This system provides a long due alternative for traditional network deployment techniques.

However, it needs to be improved in order to deal with challenges of real world scale deployments. In this section, we have discussed some of the limitations of current prototype and proposed directions for some of the future work. The proposed system is the first attempt to realize network management and security as a service model and it is expected to offer various interesting research problems when deployed at large scale.

References

[1] Abhimanyu: *Hackers can remotely activate your smart vibrator and find out how often you use it.* http://thenextweb.com/gadgets/2016/08/10/hackers-can-remotely-activate-vibrator-find-often-use/, 2016. [Online; accessed 10-August-2016].

[2] Abhimanyu: *This is why I'm still wary of the Internet of Things.* http://thenextweb.com/gadgets/2016/07/27/this-is-why-im-still-wary-of-the-internet-of-things/, 2016. [Online; accessed 10-August-2016].

[3] Alam, Safwan and El-Khatib, Khalil: *Phishing susceptibility detection through social media analytics.* In *Proceedings of the 9th International Conference on Security of Information and Networks*, SIN '16, pages 61–64, New York, NY, USA, 2016. ACM, ISBN 978-1-4503-4764-8. http://doi.acm.org/10.1145/2947626.2947637.

[4] Allen, Grant and Owens, Mike: *The Definitive Guide to SQLite.* Apress, Berkely, CA, USA, 2nd edition, 2010, ISBN 1430232250, 9781430232254.

[5] Alwabel, Abdulla, Yu, Minlan, Zhang, Ying, and Mirkovic, Jelena: *SENSS: Observe and Control Your Own Traffic in the Internet.* In *Proceedings of the 2014 ACM Conference on SIGCOMM*, SIGCOMM '14, pages 349–350, New York, NY, USA, 2014. ACM, ISBN 978-1-4503-2836-4. http://doi.acm.org/10.1145/2619239.2631459.

[6] Atzori, Luigi, Iera, Antonio, and Morabito, Giacomo: *The internet of things: A survey.* Comput. Netw., 54(15):2787–2805, October 2010, ISSN 1389-1286. http://dx.doi.org/10.1016/j.comnet.2010.05.010.

[7] Bit Defender: *Bitdefender BOX - IoT Security Solution For All Connected Devices.* www.bitdefender.com/box/, 2016. [Online; accessed 1-August-2016].

[8] Blake, Andrew: *Ransomware infections led to $1.6 million in losses during 2015: Report.* http://www.washingtontimes.com/news/2016/may/25/ransomware-infections-led-16-million-losses-during/. [Online; accessed 1-August-2016].

[9] Bozkurt, Ilker Nadi and Benson, Theophilus: *Contextual router: Advancing experience oriented networking to the home.* In *Proceedings of the Symposium on SDN Research*, SOSR '16, pages 15:1–15:7, New York, NY, USA, 2016. ACM, ISBN 978-1-4503-4211-7. http://doi.acm.org/10.1145/2890955.2890972.

[10] Bremler-Barr, Anat, Harchol, Yotam, Hay, David, and Koral, Yaron: *Deep packet inspection as a service.* In *Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies*, CoNEXT '14, pages 271–282, New York, NY, USA, 2014. ACM, ISBN 978-1-4503-3279-8. `http://doi.acm.org/10.1145/2674005.2674984`.

[11] Brewer, Eric A.: *Kubernetes and the path to cloud native.* In *Proceedings of the Sixth ACM Symposium on Cloud Computing*, SoCC '15, pages 167–167, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3651-2. `http://doi.acm.org/10.1145/2806777.2809955`.

[12] Brooks, Michael and Yang, Baijian: *A man-in-the-middle attack against opendaylight sdn controller.* In *Proceedings of the 4th Annual ACM Conference on Research in Information Technology*, RIIT '15, pages 45–49, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3836-3. `http://doi.acm.org/10.1145/2808062.2808073`.

[13] Burch, Carl: *Django, a web framework using python: Tutorial presentation.* J. Comput. Sci. Coll., 25(5):154–155, May 2010, ISSN 1937-4771. `http://dl.acm.org/citation.cfm?id=1747137.1747166`.

[14] Cara McGoogan: *Fake version of Pokemon Go infects Android and takes over phone.* `http://www.telegraph.co.uk/technology/2016/07/11/fake-version-of-pokemon-go-infects-android-and-takes-over-phone/`, 2016. [Online; accessed 26-July-2016].

[15] Carbonaro, N., Anania, G., Mura, G. D., Tesconi, M., Tognetti, A., Zupone, G., and De Rossi, Danilo: *Wearable biomonitoring system for stress management: A preliminary study on robust ecg signal processing.* In *Proceedings of the 2011 IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks*, WOWMOM '11, pages 1–6, Washington, DC, USA, 2011. IEEE Computer Society, ISBN 978-1-4577-0352-2. `http://dx.doi.org/10.1109/WoWMoM.2011.5986192`.

[16] Casado, Martin, Freedman, Michael J., Pettit, Justin, Luo, Jianying, McKeown, Nick, and Shenker, Scott: *Ethane: Taking control of the enterprise.* In *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '07, pages 1–12, New York, NY, USA, 2007. ACM, ISBN 978-1-59593-713-1. `http://doi.acm.org/10.1145/1282380.1282382`.

[17] Chatriot, Louis: *NeDB: a lightweight Javascript database using MongoDB's API.* `http://blog.mongodb.org/post/55693224724/nedb-a-`

lightweight-javascript-database-using, 2016. [Online; accessed 1-September-2016].

[18] Chetty, Marshini, Kim, Hyojoon, Sundaresan, Srikanth, Burnett, Sam, Feamster, Nick, and Edwards, W. Keith: *ucap: An internet data management tool for the home.* In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 3093–3102, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3145-6. http://doi.acm.org/10.1145/2702123.2702218.

[19] Chi, Pei Yu (Peggy) and Li, Yang: *Weave: Scripting cross-device wearable interaction.* In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 3923–3932, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3145-6. http://doi.acm.org/10.1145/2702123.2702451.

[20] Chodorow, Kristina and Dirolf, Michael: *MongoDB: The Definitive Guide.* O'Reilly Media, Inc., 1st edition, 2010, ISBN 1449381561, 9781449381561.

[21] Clark, Bryan: *Buying a smart lock might be a dumb investment.* http://thenextweb.com/insider/2016/08/09/buying-a-smart-lock-might-be-a-dumb-investment, 2016. [Online; accessed 10-August-2016].

[22] Condra, Geremy: *A plea for incremental work in iot security.* In *Proceedings of the 5th International Workshop on Trustworthy Embedded Devices*, TrustED '15, pages 39–39, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3828-8. http://doi.acm.org/10.1145/2808414.2808424.

[23] Corporation, Onion: *Omega2.* https://onion.io/, 2016. [Online; accessed 29-August-2016].

[24] Corporation, Symantec: *Internet security threat report.* Technical Report 7, Symantec Corporation, April 2016. https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf.

[25] Cujo: *The Smart Way To Fight Hacking.* https://www.getcujo.com/internet-of-things-a-gateway-for-hackers/, 2016. [Online; accessed 1-September-2016].

[26] Dhamija, Rachna and Tygar, J. D.: *The battle against phishing: Dynamic security skins.* In *Proceedings of the 2005 Symposium on Usable Privacy and Security*, SOUPS '05, pages 77–88, New York, NY, USA, 2005. ACM, ISBN 1-59593-178-3. http://doi.acm.org/10.1145/1073001.1073009.

[27] Dockrill, Peter: *Project Sauron: Scientists just found an advanced form of malware that's been hiding for at least 5 years.* `http://www.sciencealert.com/scientists-just-found-an-advanced-form-of-malware-that-s-been-hiding-for-at-least-5-years`, 2016. [Online; accessed 12-August-2016].

[28] Ecko, Jack and Alexa: *Amazon Echo: Amazon Echo User Guide.* CreateSpace Independent Publishing Platform, USA, 2015, ISBN 1511916494, 9781511916493.

[29] Erickson, David: *The beacon openflow controller.* In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, HotSDN '13, pages 13–18, New York, NY, USA, 2013. ACM, ISBN 978-1-4503-2178-5. `http://doi.acm.org/10.1145/2491185.2491189`.

[30] F-Secure: *F-Secure SENSE: Smart security for your smart lifestyle.* `https://sense.f-secure.com/`, 2016. [Online; accessed 1-August-2016].

[31] Feamster, Nick: *Outsourcing Home Network Security.* In *Proceedings of the 2010 ACM SIGCOMM Workshop on Home Networks*, HomeNets '10, pages 37–42, New York, NY, USA, 2010. ACM, ISBN 978-1-4503-0198-5. `http://doi.acm.org/10.1145/1851307.1851317`.

[32] Feamster, Nick, Rexford, Jennifer, and Zegura, Ellen: *The road to sdn: An intellectual history of programmable networks.* SIGCOMM Comput. Commun. Rev., 44(2):87–98, April 2014, ISSN 0146-4833. `http://doi.acm.org/10.1145/2602204.2602219`.

[33] Florêncio, Dinei, Herley, Cormac, and Coskun, Baris: *Do strong web passwords accomplish anything?* In *Proceedings of the 2Nd USENIX Workshop on Hot Topics in Security*, HOTSEC'07, pages 10:1–10:6, Berkeley, CA, USA, 2007. USENIX Association. `http://dl.acm.org/citation.cfm?id=1361419.1361429`.

[34] Fran Howarth: *The Role of Human Error in Successful Security Attacks.* `https://securityintelligence.com/the-role-of-human-error-in-successful-security-attacks/`, 2016. [Online; accessed 1-August-2016].

[35] Gartner: *Gartner Says a Typical Family Home Could Contain More Than 500 Smart Devices by 2022.* `http://www.gartner.com/newsroom/id/2839717`, 2014. [Online; accessed 5-August-2016].

[36] Gember, Aaron, Grandl, Robert, Khalid, Junaid, and Akella, Aditya: *Design and implementation of a framework for software-defined middlebox networking.* SIGCOMM Comput. Commun. Rev., 43(4):467–468, August 2013, ISSN 0146-4833. `http://doi.acm.org/10.1145/2534169.2491686`.

[37] Gember, Aaron, Prabhu, Prathmesh, Ghadiyali, Zainab, and Akella, Aditya: *Toward software-defined middlebox networking.* In *Proceedings of the 11th ACM Workshop on Hot Topics in Networks*, HotNets-XI, pages 7–12, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1776-4. `http://doi.acm.org/10.1145/2390231.2390233`.

[38] Gharakheili, H. H., Bass, J., Exton, L., and Sivaraman, V.: *Personalizing the home network experience using cloud-based sdn.* In *World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on a*, pages 1–6, June 2014.

[39] Google: *OnHub from Google. A better way to Wi-Fi.* `https://on.google.com/hub/`, 2016. [Online; accessed 21-August-2016].

[40] Grace Williams: *HummingBad: Chinese malware infects 10 million Android devices, experts warn.* `http://www.foxnews.com/tech/2016/07/05/hummingbad-chinese-malware-infects-10-million-android-devices-experts-warn.html`, 2016. [Online; accessed 26-July-2016].

[41] Grinberg, Miguel: *Flask Web Development: Developing Web Applications with Python.* O'Reilly Media, Inc., 1st edition, 2014, ISBN 1449372627, 9781449372620.

[42] Gude, Natasha, Koponen, Teemu, Pettit, Justin, Pfaff, Ben, Casado, Martín, McKeown, Nick, and Shenker, Scott: *Nox: Towards an operating system for networks.* SIGCOMM Comput. Commun. Rev., 38(3):105–110, July 2008, ISSN 0146-4833. `http://doi.acm.org/10.1145/1384609.1384625`.

[43] Ha, Kiryong, Chen, Zhuo, Hu, Wenlu, Richter, Wolfgang, Pillai, Padmanabhan, and Satyanarayanan, Mahadev: *Towards wearable cognitive assistance.* In *Proceedings of the 12th Annual International Conference on Mobile Systems, Applications, and Services*, MobiSys '14, pages 68–81, New York, NY, USA, 2014. ACM, ISBN 978-1-4503-2793-0. `http://doi.acm.org/10.1145/2594368.2594383`.

[44] Hafeez, Ibbad, Ding, Aaron Yi, Suomalainen, Lauri, Hätönen, Seppo, Niemi, Valtteri, and Tarkoma, Sasu: *Demo: Cloud-based security as a service for smart iot environments.* In *Proceedings*

of the 2015 Workshop on Wireless of the Students, by the Students, for the Students, S3 '15, pages 20–20, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3701-4. http://doi.acm.org/10.1145/2801694.2802140.

[45] Hafeez, Ibbad, Ding, Aaron Yi, Suomalainen, Lauri, Hätönen, Seppo, Niemi, Valtteri, and Tarkoma, Sasu: *Secure Smart Home with SDN based Cloud Service.* In *Close Workshop*, CloSe Project Workshop, 2015. https://wiki.aalto.fi/display/CloSeProject/CloSe+Project+Workshop.

[46] Hafeez, Ibbad, Ding, Aaron Yi, Suomalainen, Lauri, and Tarkoma, Sasu: *Demo: Securebox- a platform to safeguard network edge.* In *SEC, 2016*, Washington DC, USA, 2016.

[47] Hansen, Christopher J.: *Internetworking with bluetooth low energy.* GetMobile: Mobile Comp. and Comm., 19(2):34–38, August 2015, ISSN 2375-0529. http://doi.acm.org/10.1145/2817761.2817774.

[48] Hätönen, Seppo, Savolainen, Petri, Rao, Ashwin, Flinck, Hannu, and Tarkoma, Sasu: *Off-the-Shelf Software-defined Wi-Fi Networks.* In *Proceedings of the ACM SIGCOMM 2016 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '16, New York, NY, USA, 2016. ACM.

[49] Heller, Brandon, Sherwood, Rob, and McKeown, Nick: *The controller placement problem.* In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, HotSDN '12, pages 7–12, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1477-0. http://doi.acm.org/10.1145/2342441.2342444.

[50] Hughes, Matthew: *Thermostats can now get infected with ransomware, because 2016.* http://thenextweb.com/gadgets/2016/08/08/thermostats-can-now-get-infected-with-ransomware-because-2016/, 2016. [Online; accessed 10-August-2016].

[51] IDC: *Worldwide WLAN Market Shows Steady Growth in the First Quarter of 2016.* www.idc.com/getdoc.jsp?containerId=prUS41338416, 2016. [Online; accessed 31-August-2016].

[52] IETF: *Managed Incident Lightweight Exchange(mile).* https://datatracker.ietf.org/group/mile/charter/, 2016. [Online; accessed 1-August-2016].

[53] Inc, Docker: *Docker: Build, Ship, Run.* https://www.docker.com/, 2016. [Online; accessed 1-August-2016].

[54] Intel: *Intel®Edison Board.* https://software.intel.com/en-us/iot/hardware/edison, 2016. [Online; accessed 1-August-2016].

[55] Jain, Sushant, Kumar, Alok, Mandal, Subhasree, Ong, Joon, Poutievski, Leon, Singh, Arjun, Venkata, Subbaiah, Wanderer, Jim, Zhou, Junlan, Zhu, Min, Zolla, Jon, Hölzle, Urs, Stuart, Stephen, and Vahdat, Amin: *B4: Experience with a globally-deployed software defined wan.* SIGCOMM Comput. Commun. Rev., 43(4):3–14, August 2013, ISSN 0146-4833. http://doi.acm.org/10.1145/2534169.2486019.

[56] Jakobsson, Markus and Juels, Ari: *Server-side detection of malware infection.* In *Proceedings of the 2009 Workshop on New Security Paradigms Workshop*, NSPW '09, pages 11–22, New York, NY, USA, 2009. ACM, ISBN 978-1-60558-845-2. http://doi.acm.org/10.1145/1719030.1719033.

[57] Kim, Hyojoon, Voellmy, Andreas, Burnett, Sam, Feamster, Nick, and Clark, Russ: *Lithium: Event-Driven Network Control.* Technical report, Georgia Tech, June 2012. https://smartech.gatech.edu/handle/1853/43377.

[58] Koponen, Teemu, Casado, Martin, Gude, Natasha, Stribling, Jeremy, Poutievski, Leon, Zhu, Min, Ramanathan, Rajiv, Iwata, Yuichiro, Inoue, Hiroaki, Hama, Takayuki, and Shenker, Scott: *Onix: A distributed control platform for large-scale production networks.* In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, OSDI'10, pages 351–364, Berkeley, CA, USA, 2010. USENIX Association. http://dl.acm.org/citation.cfm?id=1924943.1924968.

[59] Kottasova, Ivana: *Russian website streams thousands of private webcams.* http://money.cnn.com/2014/11/20/technology/security/hacked-web-cameras-russia/, 2016. [Online; accessed 23-August-2016].

[60] Linux Foundation: *Open DayLight*, 2016. https://www.opendaylight.org/, [Online; accessed 1-August-2016].

[61] Lookout and Lab, Citizen: *Sophisticated, persistent mobile attack against high-value targets on iOS.* https://blog.lookout.com/blog/2016/08/25/trident-pegasus/, 2016. [Online; accessed 31-August-2016].

[62] Matyszczyk, Chris: *Samsung's warning: Our Smart TVs record your living room chatter.* http://www.cnet.com/news/samsungs-warning-our-smart-tvs-record-your-living-room-chatter/, 2016. [Online; accessed 31-August-2016].

[63] McAfee: *Net Losses: Estimating the Global Cost of Cybercrime.* Technical report, Center for Strategic and International Studies, 2014. [Online; accessed 23-August-2016].

[64] Meng, Meng, You, Lizhao, Tan, Kun, Zhang, Jiansong, and Wang, Wenjie: *Spacehub: A smart relay system for smart home.* In *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*, HotNets-XIV, pages 8:1–8:7, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-4047-2. `http://doi.acm.org/10.1145/2834050.2834073`.

[65] Mix: *Malicious KickassTorrents clone wants to steal your credit card details.* `http://thenextweb.com/insider/2016/08/15/popular-kickasstorrents-clone-torrent-site-wants-to-steal-your-credit-card-details/`, 2016. [Online; accessed 21-August-2016].

[66] Morgan: *Announcing Our Worst Passwords of 2015.* `https://www.teamsid.com/worst-passwords-2015/`, 2016. [Online; accessed 1-August-2016].

[67] Morgan, Steve: *Cyber Crime Costs Projected To Reach $2 Trillion by 2019.* `http://www.forbes.com/sites/stevemorgan/2016/01/17/cyber-crime-costs-projected-to-reach-2-trillion-by-2019`, 2016. [Online; accessed 23-August-2016].

[68] Morgan, Steve: *Cybersecurity market report q3- 2016.* Technical report, Cybersecurity Ventures, 2016. `http://cybersecurityventures.com/cybersecurity-market-report/`.

[69] Motti, Vivian Genaro and Caine, Kelly: *Users' Privacy Concerns About Wearables*, pages 231–244. Springer Berlin Heidelberg, Berlin, Heidelberg, 2015, ISBN 978-3-662-48051-9. `http://dx.doi.org/10.1007/978-3-662-48051-9_17`.

[70] Nathan Francis: *Is 'POKEMON' Down? Here's How To Check If The Popular Game's Servers Are Down Again?* `http://www.inquisitr.com/3339056/is-pokemon-go-down-heres-how-to-check-if-the-popular-games-servers-are-down-again/`, 2016. [Online; accessed 14-August-2016].

[71] Nayak, Ankur Kumar, Reimers, Alex, Feamster, Nick, and Clark, Russ: *Resonance: Dynamic access control for enterprise networks.* In *Proceedings of the 1st ACM Workshop on Research on Enterprise Networking*, WREN '09, pages 11–18, New York, NY, USA, 2009. ACM, ISBN 978-1-60558-443-0. `http://doi.acm.org/10.1145/1592681.1592684`.

[72] Neagle, Colin: *Smart home hacking is easier than you think.* http://www.networkworld.com/article/2905053/security0/smart-home-hacking-is-easier-than-you-think.html, 2015. [Online; accessed 1-August-2016].

[73] Networks, Aerohive: *Aerohive private psk.* Technical report, Aerohive Networks, solution brief, 2011.

[74] Networks, Big Switch: *Project Floodlight: Floodlight Open SDN Controller.* http://www.projectfloodlight.org/floodlight/, 2016. [Online; accessed 1-August-2016].

[75] Newman, Harvey, Mughal, Azher, Kcira, Dorian, Legrand, Iosif, Voicu, Ramiro, and Bunn, Julian: *High speed scientific data transfers using software defined networking.* In *Proceedings of the Second Workshop on Innovating the Network for Data-Intensive Science*, INDIS '15, pages 2:1–2:9, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-4002-1. http://doi.acm.org/10.1145/2830318.2830320.

[76] Nguyen, Le Nguyen Ngu, Rodríguez-Martín, Daniel, Català, Andreu, Pérez-López, Carlos, Samà, Albert, and Cavallaro, Andrea: *Basketball activity recognition using wearable inertial measurement units.* In *Proceedings of the XVI International Conference on Human Computer Interaction*, Interacciòn '15, pages 60:1–60:6, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3463-1. http://doi.acm.org/10.1145/2829875.2829930.

[77] Nikolich, Anita: *Sdn research challenges and opportunities.* In *Proceedings of the Sixth ACM Conference on Data and Application Security and Privacy*, CODASPY '16, pages 254–254, New York, NY, USA, 2016. ACM, ISBN 978-1-4503-3935-3. http://doi.acm.org/10.1145/2857705.2857730.

[78] Norton: *Are Hackers Using Your Webcam to Watch You?* http://us.norton.com/internetsecurity-malware-webcam-hacking.html, 2016. [Online; accessed 23-August-2016].

[79] Noticon Network Intelligence: *Understanding the impact of network latency on Service Providers' business.* http://www.noction.com/blog/does_latency_really_matter, 2012. [Online; accessed 26-July-2016].

[80] Oltisik, Jon: *The Internet of Things: A CISO and Network Security Perspective.* Technical report, Enterprise Strategy Group, October 2014. http://www.cisco.com/c/dam/en_us/solutions/industries/docs/energy/network-security-perspective.pdf.

[81] Pauli, Darren: *Connected kettles boil over, spill Wi-Fi passwords over London.* http://www.theregister.co.uk/2015/10/19/bods_brew_ikettle_20_hack_plot_vulnerable_london_pots/, 2015. [Online; accessed 1-August-2016].

[82] Pfaff, Ben, Pettit, Justin, Koponen, Teemu, Jackson, Ethan J., Zhou, Andy, Rajahalme, Jarno, Gross, Jesse, Wang, Alex, Stringer, Jonathan, Shelar, Pravin, Amidon, Keith, and Casado, Martín: *The design and implementation of open vswitch.* In *Proceedings of the 12th USENIX Conference on Networked Systems Design and Implementation*, NSDI'15, pages 117–130, Berkeley, CA, USA, 2015. USENIX Association, ISBN 978-1-931971-218. http://dl.acm.org/citation.cfm?id=2789770.2789779.

[83] Porras, Philip, Shin, Seungwon, Yegneswaran, Vinod, Fong, Martin, Tyson, Mabry, and Gu, Guofei: *A security enforcement kernel for openflow networks.* In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, HotSDN '12, pages 121–126, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1477-0. http://doi.acm.org/10.1145/2342441.2342466.

[84] Porras, Philip, Shin, Seungwon, Yegneswaran, Vinod, Fong, Martin, Tyson, Mabry, and Gu, Guofei: *A security enforcement kernel for openflow networks.* In *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, HotSDN '12, pages 121–126, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1477-0. http://doi.acm.org/10.1145/2342441.2342466.

[85] Pouwelse, Johan, Garbacki, Pawel, Epema, Dick, and Sips, Henk: *The bittorrent p2p file-sharing system: Measurements and analysis.* In *Proceedings of the 4th International Conference on Peer-to-Peer Systems*, IPTPS'05, pages 205–216, Berlin, Heidelberg, 2005. Springer-Verlag, ISBN 3-540-29068-0, 978-3-540-29068-1. http://dx.doi.org/10.1007/11558989_19.

[86] Powers, David M. W.: *Applications and explanations of zipf's law.* In *Proceedings of the Joint Conferences on New Methods in Language Processing and Computational Natural Language Learning*, NeMLaP3/CoNLL '98, pages 151–160, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics, ISBN 0-7258-0634-6. http://dl.acm.org/citation.cfm?id=1603899.1603924.

[87] Preda, Mila Dalla, Christodorescu, Mihai, Jha, Somesh, and Debray, Saumya: *A semantics-based approach to malware detection.* SIGPLAN Not., 42(1):377–388, January 2007, ISSN 0362-1340. http://doi.acm.org/10.1145/1190215.1190270.

[88] Qazi, Zafar Ayyub, Tu, Cheng Chun, Chiang, Luis, Miao, Rui, Sekar, Vyas, and Yu, Minlan: *Simple-fying middlebox policy enforcement using sdn.* In *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, SIGCOMM '13, pages 27–38, New York, NY, USA, 2013. ACM, ISBN 978-1-4503-2056-6. `http://doi.acm.org/10.1145/2486001.2486022`.

[89] Qualcomm: *Smart Home.* `https://www.qualcomm.com/products/smart-home`, 2016. [Online; accessed 1-August-2016].

[90] Radware: *DefenseFlow NetFlow and SDN based DDoS Attack Defense.* `http://www.radware.com/Products/DefenseFlow/`, 2016. [Online; accessed 1-August-2016].

[91] Rao, Ashwin, Sherry, Justine, Legout, Arnaud, Krishnamurthy, Arvind, Dabbous, Walid, and Choffnes, David: *Meddle: Middleboxes for increased transparency and control of mobile traffic.* In *Proceedings of the 2012 ACM Conference on CoNEXT Student Workshop*, CoNEXT Student '12, pages 65–66, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1779-5. `http://doi.acm.org/10.1145/2413247.2413286`.

[92] Rivero, Armando, Costante, Gianluca, Bonizzoni, Edoardo, Puiatti, Alessandro, and Förster, Anna: *Interconnecting zigbee and bluetooth networks with blupzi.* In *Proceedings of the 12th ACM Conference on Embedded Network Sensor Systems*, SenSys '14, pages 314–315, New York, NY, USA, 2014. ACM, ISBN 978-1-4503-3143-2. `http://doi.acm.org/10.1145/2668332.2668373`.

[93] Rober Baldwin: *The Dojo gateway secures your smart TV and other home devices.* `https://www.engadget.com/2015/11/19/dojo-secure-network-gateway/`, 2016. [Online; accessed 1-September-2016].

[94] Robyns, Pieter, Bonné, Bram, Quax, Peter, and Lamotte, Wim: *Short paper: Exploiting wpa2-enterprise vendor implementation weaknesses through challenge response oracles.* In *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless & Mobile Networks*, WiSec '14, pages 189–194, New York, NY, USA, 2014. ACM, ISBN 978-1-4503-2972-9. `http://doi.acm.org/10.1145/2627393.2627411`.

[95] Roesch, Martin: *Snort - lightweight intrusion detection for networks.* In *Proceedings of the 13th USENIX Conference on System Administration*, LISA '99, pages 229–238, Berkeley, CA, USA, 1999. USENIX Association. `http://dl.acm.org/citation.cfm?id=1039834.1039864`.

[96] Ronacher, A: *Jinja2 The Python Template Engine.* `http://jinja.pocoo.org`, 2016. [Online; accessed 1-August-2016].

[97] Ruckus: *Ruckus Wireless Awarded Wi-Fi Security Patent for Dynamic Authentication and Encryption.* `https://www.ruckuswireless.com/press/releases/20100524-dynamic-psk-patent`, 2010. [Online; accessed 1-August-2016].

[98] Sami, Ashkan, Yadegari, Babak, Rahimi, Hossein, Peiravian, Naser, Hashemi, Sattar, and Hamze, Ali: *Malware detection based on mining api calls.* In *Proceedings of the 2010 ACM Symposium on Applied Computing*, SAC '10, pages 1020–1025, New York, NY, USA, 2010. ACM, ISBN 978-1-60558-639-7. `http://doi.acm.org/10.1145/1774088.1774303`.

[99] Shanmugam, Praveen Kumar, Subramanyam, Naveen Dasa, Breen, Joe, Roach, Corey, and Merwe, Jacobus Van der: *DEIDtect: Towards Distributed Elastic Intrusion Detection.* In *Proceedings of the 2014 ACM SIGCOMM Workshop on Distributed Cloud Computing*, DCC '14, pages 17–24, New York, NY, USA, 2014. ACM, ISBN 978-1-4503-2992-7. `http://doi.acm.org/10.1145/2627566.2627579`.

[100] Sherry, Justine, Hasan, Shaddi, Scott, Colin, Krishnamurthy, Arvind, Ratnasamy, Sylvia, and Sekar, Vyas: *Making Middleboxes Someone else's Problem: Network Processing As a Cloud Service.* In *Proceedings of the ACM SIGCOMM 2012 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, SIGCOMM '12, pages 13–24, New York, NY, USA, 2012. ACM, ISBN 978-1-4503-1419-0. `http://doi.acm.org/10.1145/2342356.2342359`.

[101] Sherry, Justine, Lan, Chang, Popa, Raluca Ada, and Ratnasamy, Sylvia: *Blindbox: Deep packet inspection over encrypted traffic.* In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, SIGCOMM '15, pages 213–226, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3542-3. `http://doi.acm.org/10.1145/2785956.2787502`.

[102] Sherwood, Rob, Gibb, Glen, Yap, Kok Kiong, Appenzeller, Guido, Casado, Martin, McKeown, Nick, and Parulkar, Guru: *Can the production network be the testbed?* In *Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation*, OSDI'10, pages 365–378, Berkeley, CA, USA, 2010. USENIX Association. `http://dl.acm.org/citation.cfm?id=1924943.1924969`.

[103] Sicari, S., Rizzardi, A., Grieco, L.A., and Coen-Porisini, A.: *Security, privacy and trust in internet of things.* Comput. Netw., 76(C):146–164, January 2015, ISSN 1389-1286. `http://dx.doi.org/10.1016/j.comnet.2014.11.008`.

[104] Siddiqui, Muazzam, Wang, Morgan C., and Lee, Joohan: *A survey of data mining techniques for malware detection using file features.* In *Proceedings of the 46th Annual Southeast Regional Conference on XX*, ACM-SE 46, pages 509–510, New York, NY, USA, 2008. ACM, ISBN 978-1-60558-105-7. `http://doi.acm.org/10.1145/1593105.1593239`.

[105] Smith, Craig and Miessker, Daniel: *Internet of Things HP Security Research Study.* Technical report, HP, June 2014. `http://www8.hp.com/h20195/V2/GetPDF.aspx/4AA5-4759ENW.pdf`.

[106] Ståhlberg, Mikä: *Smart homes: Opportunities and risks.* In *Internet of Things-Finland Magazine*, Digile Internet of Things, pages 60–63, Helsinki, Finland, 2015. Digile. `http://www.internetofthings.fi/extras/IoTMagazine2015.pdf`.

[107] Statistica: *Statistics and facts about Smartphone.* `http://www.statista.com/topics/840/smartphones/`, 2015. [Online; accessed 1-August-2016].

[108] Still, Jeremiah D.: *Cybersecurity needs you!* interactions, 23(3):54–58, April 2016, ISSN 1072-5520. `http://doi.acm.org/10.1145/2899383`.

[109] Tews, Erik and Beck, Martin: *Practical attacks against wep and wpa.* In *Proceedings of the Second ACM Conference on Wireless Network Security*, WiSec '09, pages 79–86, New York, NY, USA, 2009. ACM, ISBN 978-1-60558-460-7. `http://doi.acm.org/10.1145/1514274.1514286`.

[110] Tim Moynihan: *Luma Makes Your Home Wi-Fi Smart, Secure, and Somehow Fun.* `http://www.wired.com/2016/05/luma-makes-wi-fi-fun-smart-secure-time/`, 2016. [Online; accessed 1-August-2016].

[111] Trapp, Brian: *Raspberry pi: The perfect home server.* Linux J., 2013(229), May 2013, ISSN 1075-3583. `http://dl.acm.org/citation.cfm?id=2492119.2492120`.

[112] UK, PCMag: *Luma Home WiFi System.* `http://uk.pcmag.com/luma-home-wifi-system/82488/review/luma-home-wifi-system`, 2016. [Online; accessed 3-August-2016].

[113] VMware: *It's time to virtualize the network.* Technical report, VMware, 2015. http://www.vmware.com/radius/wp-content/uploads/2015/10/89033_v2_NSX_2_FlexControl_AAG_UPDATE11-copy.pdf.

[114] Voellmy, Andreas and Hudak, Paul: *Nettle: Taking the sting out of programming network routers.* In *Proceedings of the 13th International Conference on Practical Aspects of Declarative Languages*, PADL'11, pages 235–249, Berlin, Heidelberg, 2011. Springer-Verlag, ISBN 978-3-642-18377-5. http://dl.acm.org/citation.cfm?id=1946313.1946339.

[115] Voip-info.org: *Call Quality Metrics.* http://www.voip-info.org/wiki/view/Call+Quality+Metrics, 2016. [Online; accessed 1-September-2016].

[116] Wakefield, Jane: *Smart LED light bulbs leak Wi-Fi passwords.* http://www.bbc.com/news/technology-28208905, 2014. [Online; accessed 1-August-2016].

[117] Wang, Chen, Guo, Xiaonan, Wang, Yan, Chen, Yingying, and Liu, Bo: *Friend or foe?: Your wearable devices reveal your personal pin.* In *Proceedings of the 11th ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '16, pages 189–200, New York, NY, USA, 2016. ACM, ISBN 978-1-4503-4233-9. http://doi.acm.org/10.1145/2897845.2897847.

[118] Wang, Richard, Butnariu, Dana, and Rexford, Jennifer: *Openflow-based server load balancing gone wild.* In *Proceedings of the 11th USENIX Conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services*, Hot-ICE'11, pages 12–12, Berkeley, CA, USA, 2011. USENIX Association. http://dl.acm.org/citation.cfm?id=1972422.1972438.

[119] Williams, Owen: *Someone could have stolen your Wi-Fi password from this Internet of Things doorbell.* http://thenextweb.com/gadgets/2016/01/12/now-someone-can-steal-your-wi-fi-password-from-your-doorbell/, 2016. [Online; accessed 1-July-2016].

[120] Wire, Tribune Media: *Webcam Nightmare: Mom Finds Young Daughters' Bedroom on Live Streaming App.* http://wnep.com/2016/08/11/webcam-nightmare-mom-finds-young-daughters-bedroom-on-live-streaming-app/, 2016. [Online; accessed 23-August-2016].

[121] Wu, K. and Wu, X.: *A wireless mobile monitoring system for home healthcare and community medical services.* In *2007 1st International*

Conference on Bioinformatics and Biomedical Engineering, pages 1190–1193, July 2007.

[122] Xavier, M. G., Neves, M. V., Rossi, F. D., Ferreto, T. C., Lange, T., and Rose, C. A. F. De: *Performance evaluation of container-based virtualization for high performance computing environments.* In *2013 21st Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*, pages 233–240, Feb 2013.

[123] Xu, Kuai, Wang, Feng, Gu, Lin, Gao, Jianhua, and Jin, Yaohui: *Characterizing home network traffic: an inside view.* Personal and Ubiquitous Computing, 18(4):967–975, 2014, ISSN 1617-4917. `http://dx.doi.org/10.1007/s00779-013-0711-x`.

[124] Yang, Rayoung and Newman, Mark W.: *Learning from a learning thermostat: Lessons for intelligent systems for the home.* In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing*, UbiComp '13, pages 93–102, New York, NY, USA, 2013. ACM, ISBN 978-1-4503-1770-2. `http://doi.acm.org/10.1145/2493432.2493489`.

[125] Yiakoumis, Yiannis, Yap, Kok Kiong, Katti, Sachin, Parulkar, Guru, and McKeown, Nick: *Slicing home networks.* In *Proceedings of the 2Nd ACM SIGCOMM Workshop on Home Networks*, HomeNets '11, pages 1–6, New York, NY, USA, 2011. ACM, ISBN 978-1-4503-0798-7. `http://doi.acm.org/10.1145/2018567.2018569`.

[126] YLE: *Parking at the airport could tip off burglars: here's how to stop them.* `http://yle.fi/uutiset/parking_at_the_airport_could_tip_off_burglarsheres_how_to_stop_them/9056642`, 2016. [Online; accessed 3-August-2016].

[127] Yu, Minlan, Zhang, Ying, Mirkovic, Jelena, and Alwabel, Abdulla: *SENSS: Software Defined Security Service.* In *Presented as part of the Open Networking Summit 2014 (ONS 2014)*, Santa Clara, CA, 2014. USENIX. `https://www.usenix.org/conference/ons2014/technical-sessions/presentation/yu`.

[128] Yu, Tianlong, Sekar, Vyas, Seshan, Srinivasan, Agarwal, Yuvraj, and Xu, Chenren: *Handling a trillion (unfixable) flaws on a billion devices: Rethinking network security for the internet-of-things.* In *Proceedings of the 14th ACM Workshop on Hot Topics in Networks*, HotNets-XIV, pages 5:1–5:7, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-4047-2. `http://doi.acm.org/10.1145/2834050.2834095`.

[129] Zachariah, Thomas, Klugman, Noah, Campbell, Bradford, Adkins, Joshua, Jackson, Neal, and Dutta, Prabal: *The internet of things has a gateway problem.* In *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, HotMobile '15, pages 27–32, New York, NY, USA, 2015. ACM, ISBN 978-1-4503-3391-7. `http://doi.acm.org/10.1145/2699343.2699344`.

[130] Zhang, K., Yang, K., Liang, X., Su, Z., Shen, X., and Luo, H. H.: *Security and privacy for mobile healthcare networks: from a quality of protection perspective.* IEEE Wireless Communications, 22(4):104–112, August 2015, ISSN 1536-1284.

[131] Zhang, K., Yang, K., Liang, X., Su, Z., Shen, X., and Luo, H. H.: *Security and privacy for mobile healthcare networks: from a quality of protection perspective.* IEEE Wireless Communications, 22(4):104–112, August 2015, ISSN 1536-1284.

[132] Zhou, J., Cao, Z., Dong, X., and Lin, X.: *Security and privacy in cloud-assisted wireless wearable communications: Challenges, solutions, and future directions.* IEEE Wireless Communications, 22(2):136–144, April 2015, ISSN 1536-1284.

A Appendix

A.1 Libraries

List of libraries used in the development of proposed system.

Libraries used in SMS development

Library	**Version**	**License**
Flask	0.1.0	Flask License
Dominate	2.1.16	GNU v3 General Public License
ForgeryPy	0.1	ForgeryPy License
Visitor	0.1.2	PSF or ZPL
Argparse	1.2.1	Python Licence
Flask-HTTPauth	3.1.1	MIT License
Flask-Login	0.3.2	MIT License
Flask-Moment	0.5.1	MIT License
Mongoengine	0.10.6	MIT License
Flask-bootstrap	3.3.5.7	BSD Licence
Flask-Script	2.0.5	BSD License
Flask-WTF	0.12	BSD License
Jinja2	2.8	BSD Licence
MarkupSafe	0.23	BSD Licence
WTForms	2.1	BSD Licence
Werkzeug	0.11.5	BSD Licence
Enum34	1.1.2	BSD Licence
Flask-mongoengine	0.7.5	BSD Licence
Itsdangerous	0.24	BSD Licence
Netaddr	0.7.18	BSD Licence
Mock	3.3	BSD Licence
Python-dateutil	2.5.3	Simplified BSD
Coverage	4.0.3	Apache v2.0 License
Pymongo	3.2	Apache v2.0 License
Selenium	2.53.1	Apache v2.0 License
Requests	2.10.0	Apache v2.0 License
Kubernetes	1.3.3	Apache v2.0 License
Docker	1.12.0	Apache v2.0 License
Requests Mock	1.0.0	Apache v2.0 License
MongoDB	3.0.8	Apache v2.0 License

Libraries used in Securebox development

Library	**Version**	**License**
Floodlight SDN controller	1.0 or later	Apache v2.0 License
Open vSwitch	2.1 or later	Apache v2.0 Licence
Gson	2.6.2	Apache v2.0 License
Hostapd	na	BSD License