Fall 11-2023

SUTMS - Unified Threat Management Framework for Home Networks

Asif Siddiqui

DAKOTA STATE UNIVERSITY

DISSERTATION APPROVAL FORM

This dissertation is approved as a credible and independent investigation by a candidate for the Doctor of Philosophy degree and is acceptable for meeting the dissertation requirements for this degree. Acceptance of this dissertation does not imply that the conclusions reached by the candidate are necessarily the conclusions of the major department or university.

Student Name: Asif Siddiqui Student ID: 101064025

Dissertation Title:
SUTMS - Unified Threat Management Framework For Home Networks

Graduate Office Verification: DocuSigned by: Brianna Mae Feldhaus F44C8D9E621C417... Date: 12/01/2023

Dissertation Chair/Co-Chair: DocuSigned by: Yong Wang 70AB505BC7B649E... Date: 12/01/2023
Print Name: Yong Wang

Dissertation Chair/Co-Chair: _____ Date: _____
Print Name: _____

Committee Member: DocuSigned by: Dr. Rimal, Bhaskar 91D1A4B81F74EF... Date: 12/02/2023
Print Name: Dr. Rimal, Bhaskar

Committee Member: DocuSigned by: Edward Dennis 5BEC844CFF91413... Date: 12/01/2023
Print Name: Edward Dennis

Committee Member: _____ Date: _____
Print Name: _____

Committee Member: _____ Date: _____
Print Name: _____

Submit Form Through Docusign Only
or to Office of Graduate Studies
Dakota State University

SUTMS - UNIFIED THREAT MANAGEMENT FRAMEWORK FOR HOME NETWORKS

A dissertation submitted to Dakota State University in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

in

Cyber Defense

November 17th, 2023

By

Asif Siddiqui

Dissertation Committee:

Dr. Yong Wang

Dr. Bhaskar Rimal

Dr. Edward M.Dennis

Beacom College of Computer and Cyber Sciences

DISSERTATION APPROVAL FORM

This dissertation is approved as a credible and independent investigation by a candidate for the Doctor of Philosophy in Cyber Operations degree and is acceptable for meeting the dissertation requirements for this degree. Acceptance of this dissertation does not imply that the conclusions reached by the candidate are necessarily the conclusions of the major department or university.

This page will be replaced by a signed page in the final version.

ACKNOWLEDGMENTS

I wish to express my heartfelt gratitude to Dr. Rimal Bhaskar for his unwavering dedication and invaluable support throughout the completion of my research. His exemplary leadership and resolute commitment to advancing research were pivotal to the success of my work.

I extend my sincere appreciation to my esteemed committee members, Dr. Yong Wang and Dr. Dennis Edward, for their meticulous review of my research and unwavering support. Their insights and guidance greatly contributed to shaping the quality of my work.

Lastly, I would like to acknowledge the pivotal role played by Dr.Martin Reisslein in the publication of my maiden journal in IEEE Communications Surveys and Tutorials (COMST). From the inception to the culmination of the publication process, Dr. Reisslein's guidance and contributions were instrumental.

ABSTRACT

Home networks were initially designed for web browsing and non-business critical applications. As infrastructure improved, internet broadband costs decreased, and home internet usage transferred to e-commerce and business-critical applications. Today's home computers host personnel identifiable information and financial data and act as a bridge to corporate networks via remote access technologies like VPN. The expansion of remote work and the transition to cloud computing have broadened the attack surface for potential threats. Home networks have become the extension of critical networks and services, hackers can get access to corporate data by compromising devices attacked to broadband routers. All these challenges depict the importance of home-based Unified Threat Management (UTM) systems. There is a need of unified threat management framework that is developed specifically for home and small networks to address emerging security challenges. In this research, the proposed Smart Unified Threat Management (SUTMS) framework serves as a comprehensive solution for implementing home network security, incorporating firewall, anti-bot, intrusion detection, and anomaly detection engines into a unified system. SUTMS is able to provide 99.99% accuracy with 56.83% memory improvements. IPS stands out as the most resource-intensive UTM service, SUTMS successfully reduces the performance overhead of IDS by integrating it with the flow detection module. The artifact employs flow analysis to identify network anomalies and categorizes encrypted traffic according to its abnormalities. SUTMS can be scaled by introducing optional functions, i.e., routing and smart logging (utilizing Apriori algorithms). The research also tackles one of the limitations identified by SUTMS through the introduction of a second artifact called Secure Centralized Management System (SCMS). SCMS is a lightweight asset management platform with built-in security intelligence that can seamlessly integrate with a cloud for real-time updates.

DECLARATION

I hereby certify that this dissertation constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions, or writings of another. I declare that the dissertation describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

Asif Siddiqui

Asif Siddiqui

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

1.1 Introduction

Home broadband routers are primarily designed for internet connectivity without considering any built-in security. Cyber attacks, remote work and cloud computing have made home networks more vulnerable than ever before. Barracuda Sentinel observed an increase of 667% of phishing attacks in 2020 [1], use of home computer for work is another reason of compromises as personnel computers lack proper security controls [2]. Lack of network security protection and increase of attack exposure introduces a need of security device that is capable of providing corporate grade security for small networks. In this chapter, we introduce an artifact that is designed to operate as a unified threat management system for small office and home networks. The artifact is named Smart Unified Threat Management System or SUTMS, its sole purpose is to protect against advanced attacks in a multi-layer approach. Traditional devices are restricted to Layer-3 and Layer-4 inspection due to the limitation on resource consumption, enhanced features like intrusion detection and anti-bot requires additional CPU cycles along with paid signature subscriptions.

SUTMS introduces advanced security features with real-time detection and blocking of malicious IP addresses, it can examine all IP-based communication, including the Internet of Things (IoT) and mobile computing devices. The system is comprised of four

core engines that work in collaboration with each other for optimization and inspection. The network intrusion detection engine is responsible for analyzing traffic for known vulnerabilities and defined signatures. The signature database gets updated on a regular basis as new vulnerabilities arise. The intrusion can be set to detect or prevent depending upon the mode of device operation i.e. intrusion detection system IDS or intrusion prevention system vice versa. IPS and Firewall engine allows or blocks traffic based on IP addresses, ports, and indicator of compromise IoC feeds. Flow detection engine serves as the backbone, and optimization, anomaly detection, and traffic identification are the key components of this layer. Finally, the routing engine routes the traffic to the desired destination, this component is optional, and routing can be outsourced to the existing infrastructure components like broadband routers, and access points. Wi-Fi, Domain Name Service DNS forwarding, and Dynamic Host Configuration Protocol service can also be configured. Running multiple security services on a single device with limited resources is a challenging task, however, using correlation services like flow detection can significantly reduce the burden of CPU intense processes like IDS/IPS. Since IDS are log-generation machines, therefore it is critical to eliminate false positives and only generate alerts for critical log messages. Logs can be locally collected or sent to a centralized log server via Syslog, it is recommended to integrate the logs into Security Incident Event Management SIEM systems in the case of corporate machines. There are various deployment approaches for SUTMS, the device can easily be implemented in the existing home infrastructure without any major changes, the various deployment scenarios are discussed later in this chapter. There is also room for expanding scanning to IoT networks like ZigBee, Z-wave, and Bluetooth low energy BLE networks, it may require procurement of additional hardware and software. The research conducted extends beyond the artifact and tackles one of the identified limitations, which is the management of smart devices. Chapter. 5 provides a comprehensive exploration of a management model.

1.2 Home Network Overview

Home networks consist of end-user computing devices, routers, and internet access, as shown in Fig. 1.1. Local area networks (LANs) and WiFi networks are comprised of IP devices that are dynamically configured for RFC 1918 addresses [3]. WiFi access is mostly protected by the WiFi Protected Access (WPA) protocol [4], [5]. WPA is designed to provide encryption and integrity and to resolve the security issues associated with Wire Equivalent Privacy (WEP) [6]. WPA/WPA2 strengthens network security using the Temporal Key Integrity Protocol (TKIP) and the AES Advanced Encryption Standard [7]. However, WPA is susceptible to brute-force attacks that exploit the 4-way handshake process of association [8].

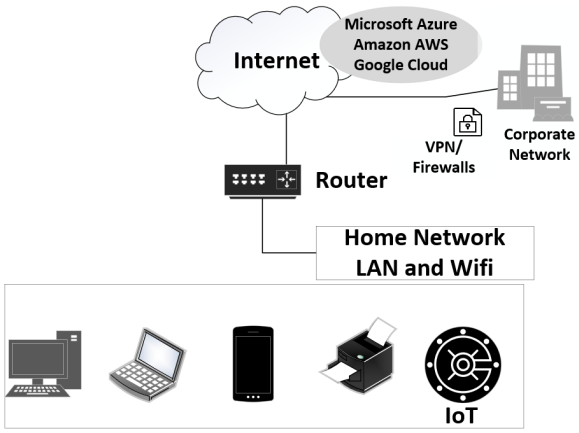


Figure 1.1: Illustration of typical topology of a home network: Common homework nodes, such as personal computer, laptop computer, mobile phone, and printer, as well as IoT nodes are connected via home network LAN or WiFi networks to a router, which provides network connectivity to the Internet, including cloud computing providers and corporate networks (typically reached via VPNs and firewalls).

Home routers are by default configured to allow benign traffic that originated from the internal network (outbound) and deny all other traffic (inbound), as illustrated in the access control list in Figure 1.2. In particular, Fig. 1.2 represents the access list name "allowed-traffic" that permits RFC 1918 IP subnets (Lines 1, 2, 3) and denies everything else (Line 4). The maximum number of host that can be assigned addresses in a given IP

```
1 access-list allowed-traffic permit 192.168.0.0 255.255.0.0
    any
2 access-list allowed-traffic permit 10.0.0.0 255.0.0.0 any
3 access-list allowed-traffic permit 172.16.0.0 255.240.0.0 any
4 access-list denied traffic deny any any
```

Figure 1.2: Sample Access Control List

address space with n free bits can be calculated as [9]:

$$2^n - 2. \tag{1.1}$$

Two IP addresses are subtracted, i.e., -2, because one address represents the subnet and another address is the broadcast address; hence, these two addresses cannot be programmed on host machines. For example, the subnet 10.0.0.0 with subnet mask 255.0.0.0 (which is sometimes represented as 10.0.0.0/8) has an address space with 24 free bits. Essentially, the 255.0.0.0 subnet mask means that the first 8 bits (i.e., the 255) are already taken, leaving 24 bits (i.e., the 0.0.0) free to be used for host addresses; thus, host IP addresses can take on any value of the form 10.x.x.x, where x is any number from 1–254. Analogously, the subnet 192.168.0.0 with subnet mask 255.255.0.0 (i.e., subnet 192.168.0.0/16) has 16 free bits, and the subnet 172.16.0.0 with subnet mask 255.240.0.0 (i.e., subnet 172.16.0.0/12) has 20 free bits. Among these, the subnet 10.0.0.0 with subnet mask 255.0.0.0 address space has the highest number of host that can be addressed, i.e, up to $2^{24} - 2 = 16,777,214$ hosts. Home networks are primarily configured for the subnet 192.168.0.0 with subnet mask 255.255.0.0, which can accommodate up to $2^{16} - 2 = 65,534$ hosts. The number of hosts can be increased or decreased by altering the subnet mask, for example the subnet 10.0.0.0 with subnet mask 255.255.255.0 can only accommodate 254 hosts, compared to 16,777,214 hosts in the subnet 10.0.0.0 with subnet mask 255.0.0.0.

The sample access control lists in Fig. 1.2 can be interpreted as follows. Line 1 allows all the traffic originating from the subnet 192.168.0.0 with subnet mask 255.255.0.0, Line 2

allows all the traffic originating from the subnet 10.0.0.0 with subnet mask 255.0.0.0, and Line 3 allows all the traffic originating from subnet 172.16.0.0 with subnet mask 255.240.0.0, while Line 4 denies all the traffic that is not matched by Lines 1, 2, or 3. Since all three networks, i.e. 192.168.0.0/16, 10.0.0.0/8, and 172.16.0.0/12 are part of the RFC 1918 private IP address space, these IP addresses cannot be configured on the internet. Thus, the access control list in Fig. 1.2 allows traffic sourced from home networks (RFC 1918 IP addresses) as well as the corresponding return traffic; however, traffic that originated from the internet will be blocked as the source IP address will not be RFC 1918, and, therefore, internet-originated traffic is going to match Line 4 (deny access). The access lists are stateful and keep track of originated sessions; thus, the access list will allow the return traffic that is sourced from the internet because the return traffic is a reply to an initial packet that originated from an RFC 1918 IP address (i.e., a home network, Line 1, 2, or 3 IP address). In summary, the access lists in Fig.1.2 allows all the traffic originating from home networks (RFC 1918 address space) and the matching return traffic. Any inbound traffic originating from the internet is going to be blocked. The order of operation of access lists is first-match-first-out; therefore, it is critical to put the "deny" at the end.

The limited awareness of home users regarding the specific services, protocols, and destination IP addresses and URLs that they use makes it challenging to differentiate between different applications. For example, distinguishing web application traffic from other application services and protocols becomes an intricate task for users with limited knowledge about their network activities. This lack of awareness creates a so-called network visibility issue. The lack of network visibility typically leads to inadequate implementation of access control lists on routers. More specifically, when routers lack application awareness and categorization functionality, then access lists are usually created based on port numbers rather than application signatures. Moreover, independent port applications that use dynamic ports (which always change) or try to find an open port to communicate,

cannot be restricted by router port based inspection. Port-agnostic (port-independent) services, such as Remote Procedure Call (RPC) and torrents, use open ports to tunnel traffic and, therefore, cannot be blocked by traditional access control lists.

1.2.1 IoT-Based Home Networks

The general Internet growth, increased vendor competition, and user inclination towards modern technology have forced companies to develop the next generation of home devices with integrated rich features. IoT has made the vision of smart homes a reality. A home network scanning study that involved 83 million devices in 16 million homes revealed some interesting home network characteristics [10], see Table 1.1: 71% of homes in North America have IoT devices, compared to the global median rate of 40% [10]. The types of IoT home devices varied by region; whereby, America has a significantly higher number of media/TV devices than Asia. People's adoption and perception of IoT differ by region. 4.1% of South Asia homes have at least one surveillance device, and the surveillance devices in these homes account for 55% of the IoT devices in South Asia. In contrast, North America has a significantly lower proportion of surveillance devices. The extensive scanning results in [10] further indicate that IoT devices are more common in the homes in developed countries; whereby, media and gaming devices are the dominating device categories. Regions of the world where technology is still evolving are confined to critical services, such as surveillance, security, and work appliances.

The study [10] also found that roughly 400 vendors account for 99% of the IoT devices [10]. The wide variety of IoT vendors introduces novel interoperability, security, and scalability challenges to home networks. In particular, the wide variety of IoT vendors makes it difficult to manage vulnerabilities and support. For instance, when operating IoT devices from many vendors at home, one vendor may be running a vulnerable version of the Apache web server, and another vendor may be exposed to buffer overflow attacks. Overall, the lack of standardization when many vendors provide home IoT devices tends

to open new vulnerabilities or make it more challenging to manage the vulnerabilities.

IoT networks typically comprise sensors, communication protocol, gateway, and application [11].

- Sensors: Sensors have direct exposure to the outside physical world. The sensors can be an interface to communicate with humans (wearable devices) or home appliances (non-wearable devices). For instance, heart rate monitors and oxygen sensors are common types of wearable human sensors. Typical non-wearable device sensors are, for example, temperature sensors that are integrated into smart HVAC appliances to observe temperature changes [11].

- Communication Protocol: A protocol specifies the messages sent between IoT networks and the application interface. IoT networks depend on several communication protocols, such as ZigBee, Bluetooth Low Energy (LE), Z-Wave, and IPv6 over Low-Power Wireless Personal Area Networks (6LoWPAN). In addition, system health checks, status monitoring, updates, and management functionalities utilize the underlying WiFi and cellular networking technologies as well as a variety of IoT protocols.

- Gateway: A gateway provides an aggregation point of communication between IoT networks and IP networks (e.g., the Internet at large). Home users access IoT applications via IP devices, e.g., smartphones, laptops, tablets, that are attached to IP networks. Similarly, IoT devices need to communicate with the Internet. The gateway works as a middleman that has the ability to translate IoT addressing schemes to IP addresses and vice versa.

- Application Interface: The application interface is the front-end component of the IoT. The application interface provides rich features, such as dashboards, reporting, and configuration management in an easy-to-use graphical user interface. The applications can be hosted on-premises, in the cloud, or embedded within the IoT

Table 1.1: Proportion [%] of homes with at least one IoT device; proportions [%] of *homes* with at least one media/TV IoT device and at least one surveillance IoT device; as well as proportions [%] of IoT *devices* that are a media/TV device or a surveillance IoT device; by region [10].

Region	Homes IoT	Media/TV Homes	Surveil. Home	Media/TV Devices	Surveil. Devices
North America	71	42.8	3.9	44.9	3.7
South America	34.4	20.5	4.6	51.7	13.3
Eastern Europe	25.7	16.8	2.5	50.2	14.0
South Asia	8.7	2.5	4.1	16.6	54.5

device. However, cloud computing is often a preferred hosting choice, providing accessibility from anywhere.

1.3 Research Objectives

The main objective of this research is to develop a next generation security framework that is capable of mitigating traditional and advance attacks for small and home networks. The artifact produced should be easy to set up, manageable, cost-effective and dynamic in nature, equipped with enhanced security inspection and have a small form factor. Scalability, interoperability, and support of existing home infrastructure is considered as part of the built-in design. The proposed Smart Unified Threat Management System (SUTMS) framework is a light-weight implementation of enterprise Unified Threat Management UTM that is geared towards home networks, SUTMS assimilate firewall, IDS, antibot and anomaly detection in a single piece of hardware i.e. RaspberryPi. SUTMS takes advantage of automation and able to dynamically update access control lists based on Indicator of Compromise IoC feeds, combination of flow data with intrusion detection signatures provides a robust and efficient protection against malicious content. SUTMS accomplishes the task of running multiple services in a hardware with limited processing power by eliminating unnecessary functions from the modules. It is observed that IDS can quickly overload the box as the number of signature increases, SUTMS optimizes IDS

processing by only enabling applicable signatures. Another aspect of the research is to highlight key challenges and improvements that can be embedded in future work, it is inevitable to secure home networks and a compact Unified Threat Management (UTM) designed specifically for home networks is a step forward in the right direction. COVID and remote work has resulted in a paradigm shift, remote works has become a reality and organizations have kept the remote work permanent, therefore the research conducted directly influences present and future work environments.

1.4 Research Questions

In this research, the goal is to answer following questions:-

- **1. How can we justify the need of unified threat management system for home networks?**

It is critical to build a business case for any solution, importance of home networks, targeted attacks and technological advancements discussed in Chapter 1 provides a sufficient evidence for smart defense solution.

- **2. How to develop a artifact that is able to run resource intense processes efficiently and effectively ?**

Firewall, IDS, Antibot and Anomoly detection are all run in parallel, SUTMS optimizes CPU and memory by simplifying configurations, and without impacting the system accuracy. Chapter 3 and Chapter 4 provides in depth analysis on SUTMS architecture and evaluation of large datasets.

- **3. How to make the solution dynamic and distinct from previous attempts?**

Integration of STIX/TAXII and Flow data into inspection engine automates prevention and detection capabilities at various layers. SUTMS is also able to detect anomalies and

perform basic SSL inspection (without decrypting traffic) by using NTOP. Chapter 2 explains the latest research conducted on UTM components, it encourages the development of differentiating solution like SUTMS.

- **4. How can we effectively handle the management challenges posed by a diverse range of home devices?**

Introducing the SCMS centralized management platform, proficient in automated and manual device detection, alongside their corresponding risk assessments. Chapter. 5 provides a detailed exposition of the design, implementation, and evaluation of this management model.

1.5 Home Network Security Challenges

Home routers are often the target of attacks. Home router attacks can be categorized as server-side attacks or as client-side attacks. Server-side attacks compromise router functionality, including routing, wireless access, and web services. Authentication attacks and SQL injection are the most commonly used server-side attacks. Client-side attacks are directed toward WiFi users, IoT networks, and connected machines. Session hijacking, IoT vulnerabilities, and weak IoT device authentication are the most prevalent client-side attacks.

Home routers are usually managed by a web-based front-end, indicating the presence of a web server running on the router itself. Routers that are provided by ISPs may also include vulnerabilities. For example, it was discovered that one of the most popular Asymmetric Digital Subscriber Line (ADSL) routers provided by a Greek ISP contained two zero-day vulnerabilities [12]: the router web interface was vulnerable to operating system (OS) command line injection and stored cross-site scripting (XSS) attacks [12]. The vulnerabilities can allow the attacker to gain shell access to the router and further infiltrate into the network by turning routers into zombie machines. Generally, the more

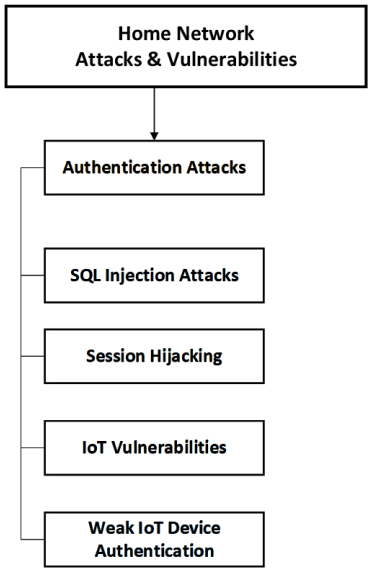


Figure 1.3: Taxonomy of home network attacks and vulnerabilities.

machines a home network has, the more chances it offers for implanting backdoors and bots. Once the attacker gains access to a home network, the malicious payload can remain in the network without being detected for a long time, i.e., the malicious payload can remain persistent with stealth capabilities, due to a lack of advanced security controls. Home network attacks and vulnerabilities can be categorized according to the taxonomy presented in Fig. 1.3.

1.5.1 Authentication Attacks

A password alone is no longer a secure way of authentication. Hence, a password is usually combined with other factors, e.g., what you have or know, often defined as multi-factor authentication (MFA). The more factors there are, the more secure the authentication will be. Adversaries have to compromise all the factors to gain access. Therefore, critical online services rely on MFA for secure communication [13].

Unfortunately, home routers have very weak authentication. Out of twenty-nine of the home routers studied by consumer reports, twenty routers did not allow changing the username and have no protection against password brute-force attacks [14]. Also,

most routers had a very weak password and lacked password complexity [14]. Even top brands had vulnerabilities. Fifty-three critical vulnerabilities were discovered on D-Link and Netgear home routers [15]. Attackers were able to create a botnet of 420,000 routers, modems, and other network-interconnected devices [16]. Furthermore, the router web interface only verifies credentials for the login page and not for the configuration scripts [16].

1.5.2 SQL Injection Attacks

Router databases do not have much valuable information from an attacker's point of view compared to commercial and government database systems, since the router databases usually do not contain Personal Identifiable Information (PII). However, hackers can use SQL injections to exploit buffer overflows and pivot into connected systems, which may give adversaries access to sensitive data [17]. Therefore, SQL queries should require validation to prevent attackers from gaining remote control of devices via SQL injection.

1.5.3 Session Hijacking

Routers can also contribute to client-side attacks. One of the most common attacks is session ID hijacking (also referred to as cookie hijacking). As most routers transmit cookies in clear text [12], an attacker can capture cookies using packet sniffing techniques and replay them, impersonating a legitimate user. Session hijacking can be used for access elevation, data exfiltration (i.e., unauthorized removal of data from a device), unauthorized access, and access to other systems. Therefore, sessions should be encrypted and authenticated to avoid session hijacking attacks.

Table 1.2: UPnP vulnerabilities and attacks [18].

Vulnerability	Adversary	Attack
Absence of verification in advertisements	Service device	Advertisement forgery/flooding
Absence of verification in discovery	Control point	Discovery reply/flooding/spoofing
Lack of authentication	Control point	Malicious action invocation
Lack of integrity check	Control point	Event forgery and flooding

1.5.4 IoT Vulnerabilities - UPnP Protocol

Absence of device verification, lack of authentication, and no integrity checks classify Universal Plug and Play (UPnP) as a highly insecure protocol with fundamental security flaws, as summarized in Table 1.2 [18]–[20]. Exploitation of UPnP vulnerabilities can result in resource exhaustion, Distributed Denial of Service (DDoS), and redirection to Command and Control botnets. Similarly, a lack of authentication and integrity checks can result in data exfiltration. Software Development Kits (SDKs), e.g., MiniUPnp [21] and LibUPnP [22], are some of the lightweight UPnP implementations for IoT. These SDKs have several security flaws. For instance, the CVE-2020-15893 vulnerability allows an attacker to inject a malicious payload in one of the discovery fields of UPnP packets [18].

1.5.5 Weak IoT Device Authentication

The implementation of clear-text protocols, such as FTP and Telnet, in the IoT software stack makes the IoT software stack an easy target for session hijacking and man-in-middle attacks. The study [10] found that 7.1% of IoT devices support FTP and Telnet protocols; whereby, 17% of these FTP-supporting devices had default or weak credentials, see Table 1.3. Security-unaware protocols and default credentials significantly increase the chances

Table 1.3: Percentages of used weak IoT FTP and Telnet credentials.

Weak FTP Credentials	%	Weak Telnet Credentials	%
admin/admin	88.3	admin/admin	35.6
admin/	5.9	root/xc3511	16.0
Administrator/	1.4	Vodafone/Vodafone	10.4
root/	0.7	admin/1234	7.5
admin/password	0.3	admin/4321	1.8

of IoT network compromises. Table 1.3 gives a worrisome picture of 88% and 36 % of IoT devices that are running FTP and Telnet services with a default username and password of admin/admin [18]. A similar behavior is also observed for device management access via clear-text HTTP. The existence of legacy services with default passwords indicates that security is not a focus for IoT vendors. FTP, Telnet, and HTTP can be enhanced by replacing the services with SFTP, SSH, and HTTPS. Simple scripts can be embedded in software modules to force the users to change default credentials.

1.6 Home Network Standards

Home area network (HAN) relies on modem, Ethernet, WiFi, switches, routers, to establish physical, data link, and network layer connectivity. The standards that govern the HAN are based on underlying network technologies.

1.6.1 Wireless Networks - IEEE 802.11

WiFi is the preferred and dominant media of communication for home networks. IEEE 802.11 is the standard published in 1997 governing access point (AP) based and ad-hoc networks [23]. The standard focuses on Physical PHY and Media Access control MAC Layers. Electrical, network, and frame characteristics are explained in detail. The original standard allowed 1-2 Mbps of speed at a 2.4GHz frequency band utilizing Frequency Hopping Spread Spectrum (FHSS), Direct Sequence Spread Spectrum (DSSS), and in-

frared (IR) technology [23]. Several amendments were made to accommodate demand and growth in technology. Table 1.4 enlists amendments made to IEEE 802.11 from 1997 to 2009 [23]. It can be inferred that every amendment improved the data transfer rate. Early versions of the IEEE 802.11 include security features referred to as Wired Equivalent Privacy. In 2004, 802.11i amendment was introduced to overcome the flaws of WEP and improvements to WiFi Protected Access (WPA) [23]. The IEEE 802.11i variant was designed to implement robust security features into the existing IEEE 802.11 standard.

Table 1.4: IEEE 802.11 amendments.

Amendments	Details
IEEE 802.11 (1997).	1 and 2 Mbps at 2.4 GHz with FHSS, DSSS and IR PHY.
IEEE 802.11b (1999).	1, 2, 5.5 and 11 Mbps at 2.4 GHz with CCK PHY.
IEEE 802.11a (1999).	6, 9, 12, 18, 24, 36, 48 and 54 Mbps at 5 GHz and with OFDM PHY.
IEEE 802.11g (2003).	6, 9, 12, 18, 24, 36, 48 and 54 Mbps at 2.4 GHz and with OFDM PHY.
IEEE 802.11n (2009)	HT capabilities with MIMO (bit rates up to 600 Mbps) at 2.4 and 5 GHz.
IEEE 802.11i (2004)	Security enhancements: Temporal Key Integrity Protocol (TKIP) and Counter Mode (CTR) with Cipher-Block Chaining Message Authentication Code (CBC-MAC) Protocol (CCMP) encryption methods and Robust Security Network Association (RSNA) protocol (authentication through 802.1x and Extensible Authentication Protocol (EAP).

1.6.2 LPWAN - IEEE 802.15

Low Power Wide Area Network (LPWAN) IEEE 802.15 is another standard that targets the connectivity of low-power devices like IoT. LPWAN operates on license exempted frequency bands and transmits payload at a very low bit rate [24]. The low data bit

rate makes IEEE 802.15 an efficient choice for IoT networks. IEEE 802.15.4y amendment integrated Advanced Encryption Standard (AES)-256 and security extensions to addresses previous security concerns [25].

1.6.3 Home LAN Standards - IEEE 802.3

In 1985, official standard for Ethernet was published by IEEE known as IEEE 802.3, specifying carrier sense multiple access with collision detection (CSMA/CD) method (a medium access control MAC Ethernet mechanism to sense transmission and prevent collisions) [26]. The original scope was 10 Mbps transmission on a coaxial medium operating in a bus topology. Amendments were made in later years to include fiber optics, twisted-pair cabling, and medium attachment unit MAU. The two prominent advancements to the standards were introduced in 1997 and 1998, i.e., 802.3x, 802.3z. Full duplex, flow control, and switching operations were incorporated in 802.3x and 802.3z set as the specification for gigabit Ethernet 1000BASE-T [26]. The most commonly used standard in home Ethernet networks is 1000BASE-T and continues to evolve as new technologies emerge. Home wide area networks (HWAN) relies on broadband technologies of Digital Subscriber Line (DSL) and cable internet. Bellcore Laboratories invented DSL in 1980 [27]. The benefit of DSL is that it uses existing home network infrastructure utilizing existing cabling. The same wiring can be used for data, voice, and television services. Variants of DSL often referred to as xDSL. ADSL is the most commonly deployed technology used by home users for internet service. ADSL can achieve download speeds of 12 Mbps and upload speeds up to 1.5 Mbps [27].

1.6.4 IoT Standards - Zigbee, Zwave, Bluetooth and Thread

IEEE 801.15 variants such as 802.15.4 and 802.15.1 are specifically designed for low power devices like IoT. Table 1.5 enlists IoT relevant standards [28]. ZigBee and Thread are built on 802.15.4, whereas 802.15.1 concentrates on Bluetooth LE. ZigBee was created

by the ZigBee alliance specifically designed for low power with long battery life devices capable of transmitting data at lower rates [29]. ZigBee is suitable for networks with a high number of nodes that require communication status, health checks, configuration changes at regular intervals. Typologies supported by ZigBee are a star, mesh, and tree [30][31]. Z-Wave is another low data rate IoT protocol developed by Zensys that supports a fewer number of nodes per network [29]. Z-Wave is appropriate for home networks as it only supports 232 nodes, whereas ZigBee is an ideal candidate for commercial IoT networks up to 65,000 nodes. Bluetooth LE 802.15.1 is designed as a convenience standard, as it allows a wide variety of home devices to connect to the user interface. Bluetooth is a preferred media to access IoT devices via smartphones, tablets, and other wearable gadgets due to the default integration of Bluetooth functionality into end-user devices. Bluetooth LE allows faster data transfer rates than ZigBee and Z-Wave. Thread is another wireless protocol designed for smartphones supporting IPV6 using 6LoWPAN [28]. 6LoWPAN is an IP-based IoT standard developed by Internet Engineering Task Force (IETF) [29].

Table 1.5: IoT standards.

Characteristics	ZigBee	Wi-Fi	Thread	Z-Wave	Bluetooth LE
IEEE Standard	802.15.4	802.11	802.15.4	N/A	802.15.1
Frequency band	2.4 GHz	2.4/5 GHz	2.4 GHz	900MHz	2.4 GHz
Nominal range	100m	150m	30m	30m	10m
Data Rate	250Kbps	1Gbps	250Kbps	100Kbps	1Mbps
Number of Nodes per Network	65000	250/Access Point	300	232	one-to-many

IoT standards reveal that a home IoT network is comprised of multiple protocols running at the same time. IoT sensors may be utilizing Zig-Bee or Z-Wave for health monitoring and data synchronization that are managed/configured by smartphones with integrated Bluetooth LE chips. The coexistence of multiple standards makes interoper-

ability extremely critical for the viability of any smart home solution.

Chapter 2

Literature Review

2.1 Unified Threat Management Overview

To study UTM, it is critical to understand each integrated feature because every inspection engine has unique requirements. Fig. 2.1 illustrates the core components of UTM. The customization of services is what makes the UTM a functioning device. Resource allocation varies component by component. For example, IDS requires more CPU cycles compared to a firewall, and dedicating resources to each module requires an in-depth analysis of various components that form UTM. Karahan and Berat [32] designed a light weight firewall on Raspberry Pi that is able to inspect ports, user groups and malicious traffic .Easy to use interface and compact design makes it a perfect candidate for home users. Cruz, Goyzueta, and Cahuana [33] introduces a $209 USD intrusion detection and prevention systems that can be managed remotely, Snort engine was utilized for attack detection according to signature matching. Protection Internet of Things via IDS PITI designed specifically for IoT networks, PITI combines Snort detection with anomaly detection [34] on a Raspberry Pi hardware. Alarms and sounds are generated as anomalies are detected by PITI [34]. Tirumala, Nepal, and Ray [35] conducted research to evaluate performance of defensive controls on Raspberry Pi, the study was mainly geared towards volumetric traffic testing across network interfaces, it was observed that built in ethernet interface performed much better than built in WiFi interface. Malware propagation via

malicious URL bypasses traditional security controls, Fauzi and Abdullah [36] introduces Rasberry Pi based phishing detection technique called Netbits. Netbits was able to identify malicious content using traffic flows, the study can be useful in mitigating botnet communication. Wireless networks can be brought down using Denial of Service DoS attacks, Kismet intrusion detection system is able to detect 10 DoS attacks with a average attack detection rate of 3.42 seconds [37]. Integration of artificial intelligence algorithms in intrusion detection system improves device accuracy, Siddharthan and Thangavel [38] proposes an algorithm for IoT devices using ensemble learning (EL). The algorithm achieved 99.99% of accuracy with lower feature selection complications [38]. Training datasets in machine learning is the most crucial step to attain accuracy, training fewer and relevant features is the key in designing an efficient algorithm. Algorithms can be deployed to detect botnet infection, BotStop (IoT based botnet detection engine) make use of only seven features from network packets and able to achieve greater than 99.99% of accuracy [39].

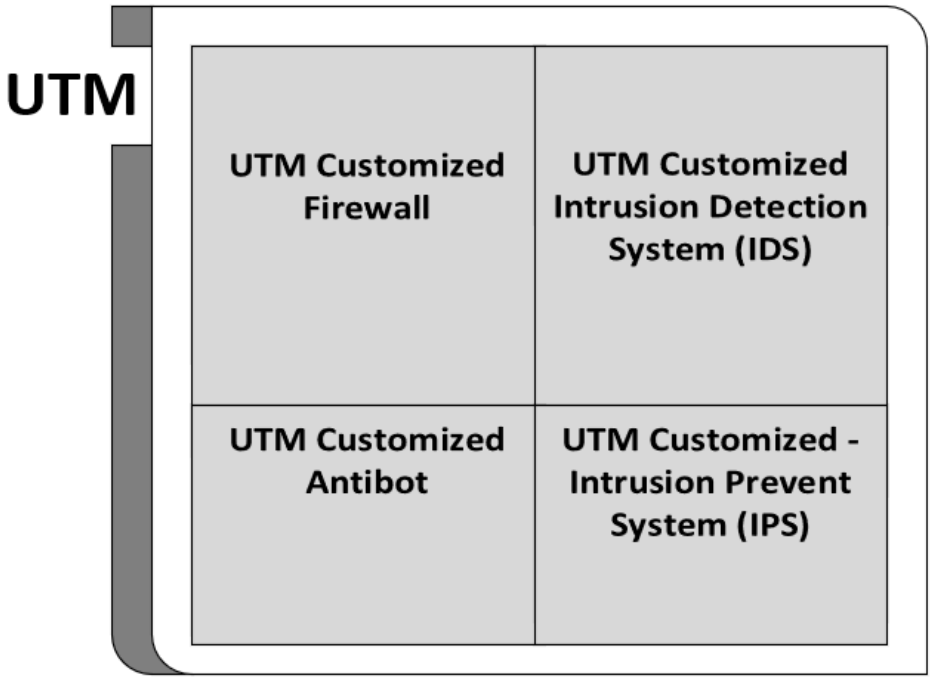


Figure 2.1: Core components of UTM.

The research carried out was focused towards IDS, Firewall, URL filtering and antibot on low resource devices like Raspberry Pi. The study conducted was limited to individual UTM components rather than development of a single cohesive device that is capable of inspecting IP, port, bot communication, malicious content and anomalies. The proposed SUTMS solution is a multi-dimension system that introduces advance inspection along with efficient resource usage.

The study of UTM systems is attracting increased attention for enterprise and small networks due to the high cost-efficiency of running multiple security services on a single device. A UTM system combines several security services, i.e., commonly traditional firewall controls, IDS, and antibot protection.

2.1.1 General Threat and Security Reviews

Home networks, including the home-based IoT networks, face numerous security threats [40]–[47]. The growing deployment of smart home technologies further exacerbates the security threats for home networks [48]–[50].

Bout, Loscri, and Gallais [51] surveyed misuse cases of Artificial Intelligence (AI), and adversaries that can craft sophisticated attacks using Machine Learning (ML). ML-based attacks are more effective and challenging to detect. More specifically, Bout, Loscri, and Gallais [51] surveyed attack generation techniques using data analysis, data generation, behavioral deduction, and behavioral diversion. Moreover, the survey by Bout, Loscri, and Gallais [51] touches on the performance impact of supervised, unsupervised, and reinforcement learning. The Bout, Loscri, and Gallais [51] survey is unique because it covers ML from a hacker's point of view; similar to the ways that AI can be utilized for defensive controls, AI can also be utilized for crafting ML-based attacks.

Cloud-hosted email solutions, such as Office365, have introduced newer challenges in terms of phishing. Phishing techniques are deployed as a gateway for command and

control (C&C) bot communication and ransomware. A C&C bot infrastructure consists of compromised machines (bots) and centralized C&C servers. An attacker uses C&C servers to manage and administer bots. Attacks can be orchestrated and adversaries can spread malicious payload by utilizing bots and by sending instructions anonymously via C&C servers. Abdul, Maham, Liu, *et al.* [52] surveyed AI based strategies for tackling phishing, as well as deep learning, hybrid learning, and scenario-based ML techniques for countering phishing attacks. The survey taxonomy of Abdul, Maham, Liu, *et al.* [52] is based on communication media, including email, websites, applications, target devices, attack techniques, and countermeasures. The survey of the phishing research in [52] can be beneficial for developing future antibot techniques, since successful phishing campaigns can lead to bot networks.

Zaman, Alhazmi, Aseeri, *et al.* [53] surveyed artificial intelligence-based strategies to counter Internet of Things (IoT) cyber threats. The survey [53] targets IoT protocols deployed at various layers and discusses associated threats. The layered approach gives readers a perspective of implementing defense in depth at various levels of processing. The surveyed AI methods were primarily rule-based, i.e., Fuzzy Logic (FL) and Neuro-Fuzzy System (NFS).

2.1.2 Firewall Reviews

The general research and development area of firewalls has been covered in a few surveys with different topical focus areas. Firewall configuration has been the focus in [54], [55]. Best practices for firewalls have been surveyed in [56], while the recent survey [57] focuses on firewall operation based on signatures and machine learning.

2.1.3 IDS Reviews

Intrusion Detection Systems (IDS) [58]–[61] are crucial in detecting intrusions and mitigating threats before they become security breaches and have been covered in several

surveys. Jinisha and Jerine [62] explore IDS mechanisms on Wireless Sensor Networks (WSNs). WSNs are critical for IoT, surveillance, and industrial use due to their ability to operate with limited power and hardware resources. However, the limited power and hardware resources also make WSNs an easy target for system availability attacks. Jinisha and Jerine [62] categorized attacks based on networks, applications, and data-link layer; the surveyed detection methods were trusted management, fault tolerance, knowledge-based, feature selection, and deep learning. Recent attack sophistication has reduced the detection capabilities of standalone IDS systems. Therefore, the survey by Li, Meng, and Kwok [63] covered distributed and collaborative IDS. The survey Franco, Aris, Canberk, *et al.* [64] observed that accuracy, management, and scalability can be improved by IDS that collaborate in a distributed environment. The survey conducted by Franco, Aris, Canberk, *et al.* [64] on Honeypots and Honey nets gives insights into their relationships with firewalls, IDS, and other UTM components. Recent IDS surveys have mainly focused on machine learning [65]–[75], as well as anomaly detection [76] and the fog computing environment [77].

2.1.4 Antibot Reviews

Relatively few surveys have covered antibot methods. The general characteristics of botnets and their detection have been surveyed in [78], [79]. Strategies for detecting malicious bots that use social media platforms or peer-to-peer networks to spread and communicate have been surveyed in [80], [81].

2.1.5 Limited UTM Reviews

Overview of Existing UTM Surveys

The related existing overview articles that claim to cover the overall UTM services are very limited; they either are performance centric, e.g., Gharat, Vidhate, and Barve [82]

and Qi, Yang, Xu, *et al.* [83], or geared towards firewalls, tools/methods, and attack-specific (ransomware), e.g., Liang and Kim [84], Miloslavskaya [85], and Kapoor, Gupta, Gupta, *et al.* [86]. Christopoulos [87] surveyed UTM capabilities and features; however, only the firewall component is covered in detail. The related surveys focused on IoT [88] and on industrial control systems [89] and emphasize the importance of controls on low-processing devices. Latesh Kumar and Leena [90] surveyed the UTM limitations arising from the limitations of machine learning. The improved UTM inspection capabilities achieved by integration of anomaly detection have been covered in [91]. Recent network security trends, such as Software Access Service Edge (SASE) and Software Defined Networks (SDN) are covered by Nazareth, Choi, and Ngo-Ye [92] and Yurekten and Demirci [93], respectively. The COVID-19 pandemic has changed the security posture, leading to adapted strategies for building a secure network environment during the pandemic, as surveyed by Akiyama, Haruta, Tamai, *et al.* [94].

Chapter 3

Research Methodology

This chapter explains the research method chosen to conduct this research, research methodology formalize research into well-defined process and procedures. The research is sequentially followed with inter related steps, the ultimate goal is to analyze, validate data that can be scientifically proven. The purpose of this research is to propose a framework for securing home networks including IP based IoT devices, the proposed system is called Smart Home Unified Threat Management System (SUTMS). The research is dedicated to resolve the security problems home networks are facing, artifact is validated and assessed in context of the problem. Home networks have become critical due to remote work, cloud computing and increase of IoT based home appliances. The absolute choice is design science for this research, design science is the study of artifact in context of problem, artifact will be designed, analyzed and evaluated [95]. Adopting design science methodology ensures that the problem is highlighted and the proposed solution is developed and validated. The artifact produced in this research will follow design science guidelines. Hevner and Chatterjee [96] categories design science into seven guidelines. The guidelines are:-

i) Design science must produce an artifact, artifact can be in the form of algorithm, method, software, construct, device, framework, or instantiation.

ii) The developed artifact produced has to be in the context of problem resolution, improvement.

iii) Evaluation is the important part of design science; the solution will be benchmarked against existing standards or acceptable evaluation methods.

iv) Successful research will have verifiable and relevant contributions.

v) Rigorous methods must be applied for the construction and evaluation of artifact.

vi) Artifact will be searched to meet the acceptable results in accordance with the problem resolution.

vii) The research should be communicated to the technological aware and management oriented audience.

Hevner and Chatterjee [96] guidelines set the foundation of design science research, Peffers proposes Design Science Research Methodology DSRM [97], Peffers model closely aligns with the research done for SUTMS framework. In this research Peffers model is used, it is consisted of six steps. Those steps are: problem identification & motivation, define objective of the solution, design & development, demonstration, evaluation, and contribution.

The process is detailed, but easy to apply and follow through from the problem identification and artifact. Peffer's six step design methodology is represented in Fig. 3.1[97].

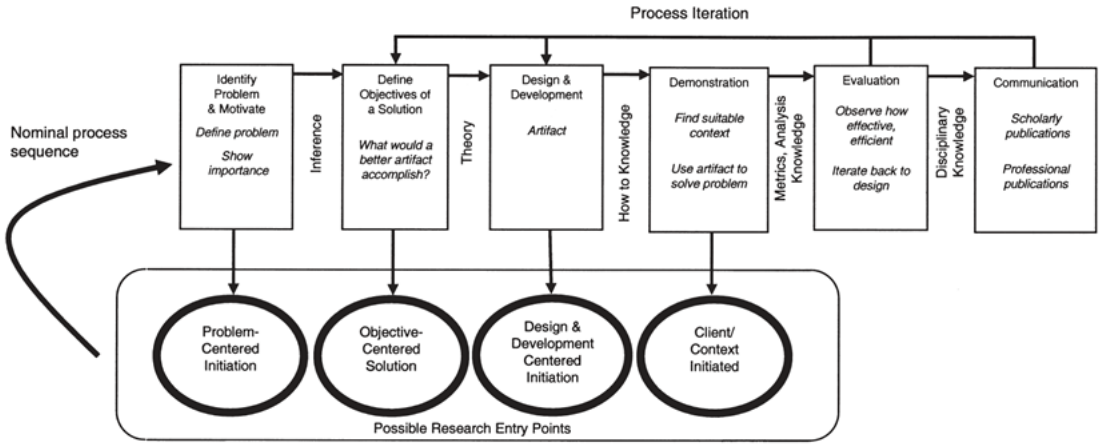


Figure 1. DSRM Process Model

Figure 3.1: Peffer's six step design science methodology [97].

3.0.1 Problem Identification and Motivation

Home networks doesn't have a corporate grade security, it is because they are originally designed for home users to access internet and non-business applications. As internet infrastructure has improved, internet broadband cost has gone down and availability has also improved. Comcast is offering internet service at $9.99 plus tax with no price increase to families in need [98]. Affordable internet service has contributed to the rise of ecommerce, cloud computing, remote work and IoT home devices. In 2017 Chinese ecommerce giant Alibaba reported rise of revenue by 56% to $7.4 billion [99]. It is expected that cloud computing industry will grow from $371.4 in 2020 to billion $832.1 billion in 2025 [100]. Remote work in 2020 have shown improved productivity according to surveys, more than 1 out of 3 HR leaders survey expected 40% of employee will be remote permanently post pandemic [101]. The number of IoT devices are expected to reach 43 billion by 2023, which is a threefold increase from 2018 [102]. It is obvious from the above data that consumers are using home networks to access ecommerce and corporate applications hosted in the cloud due to ease of access and rise of remote work. Corporations are hosting business applications on cloud providers network like Microsoft Azure, Amazon AWS and Google Cloud Platform, which homes users access with/without connecting to corporate networks therefore it depends on home networks security. Homes are equipped more and more with smart IoT based appliances and slowly legacy appliance will phase out. All those factors make the home networks critical from availability and security perspective. From adversaries point of view, compromising home networks can not only reward with Personnel Identifiable Information PII, but can also lead to access to corporate data or networks. The trend leads to the next generation of breaches using home networks. IoT devices have increased the attack surface, hackers have many options to get into home networks and IoT devices are the weakest of those. Similarly, home IoT devices can also be used as bots for DDoS attacks, crypto mining. Home networks are facing numerous

unseen threats i.e., no exposure to home networks in terms of visibility, increase of attack surface due to easy adoption of IoT devices and no adequate network protection. Home networks not only lack security, network visibility is also a problem. Most of the home users are unaware of traffic, applications, and websites used in their homes. The only protection home networks have is anti-virus software on computers and stateless firewall inspection on internet router. Smart IoT home devices also contribute to a bigger percentage of traffic, which is mostly left unprotected. There is a major gap in home networks security that has to be addressed, which motivates us for a proposed security framework for home networks. The proposed solution provides a simplified version of corporate grade security to home networks.

3.0.2 Objective of a Solution

The research proposes a solution that mitigates the security risks associated with home devices and smart IoT appliances.

- The solution must be a unified artifact that provides security along with basic connectivity. The solution can replace existing home routers by incorporating basic WiFi and routing functionality. Home users can connect their laptops, printers, desktops, smart phone and IoT devices to SUTMS via security WiFi. SUTMS can route the traffic to internet after processing it through inspection engines. There is a flexibility to deploy our solution in home networks with or without existing WiFi routers, which makes the solution scalable and easy to deploy with user's choice of network.

- One of the challenges home users have is lack of technical skills, by keeping this in mind our proposed solution requires basic configurations equivalent to home routers. Most of the services are run as part of a startup script and doesn't require user interaction. The solution can be administered via a web front end and users doesn't

need understanding of backend Ubuntu operating system.

- Visibility is an issue when it comes to home networks, users are unaware of applications, services, protocols, and traffic in use. Traffic patterns and behavior are very useful in detecting anomalies and machine learning algorithms executions. SUTMS improves visibility by incorporating flow traffic into a unified threat management platform.

- The proposed solution includes a stateful inspection engine capable of analyzing traffic in real time. Firewall rules can be customized based on IP addresses, protocols, and logs can be generated for allowed or denied traffic.

- Advanced attacks can be blocked by integrating intrusion detection and prevention capabilities, SUTMS has a Suricata engine to detect/prevent malicious traffic based on signatures.

- Firewall and Intrusion Detection System (IDS) inspection is not sufficient to block malicious traffic, keeping that in mind, the solution includes dynamic antibot capabilities. Antibot functionality is built on Indicators of Compromise (IoC's) ingestion that gets updated on regular basis and traffic to threat actor can automatically be blocked. SUTMS security capabilities is enhanced greatly with the addition of dynamic blocking of IoC's feeds.

- Firewall, IDS, Antibot configurations can be tailored towards home networks to deliver robust security without impacting the performance and efficiency of home networks. The solution is going to be tested thoroughly to automate and customize configurations based on user acceptance and performance impact to network throughput should not be significant.

3.0.3 Design and Development

Peffer's design and development phases focuses on artifact's interworking, in this research the developed artifact is the instantiation of framework of securing home networks. The design phase of this research is comprised of topology diagrams, scenarios, and software/hardware used. The design simulates a typical home network with the detailed information about network ingress and egress points. Solution design determines the attack surface for home networks and strategic deployment of artifact to reduce attack surface, Firewall, IDS, Antibot and Flow collection engines are part of solution design. The development phase of this research is the transformation of solution design into the actual working artifact. Design functions and capabilities identified in design phase are incorporated into an artifact in the form of software code. The artifact should be developed according to the design and Peffer's research methodology. Detailed technical information about design and development of artifact is included in the "SUTMS - Smart Unified Threat Management System" Chapter 4.

3.0.4 Demonstration

Peffer's demonstration phase is about the actual implementation of artifact in the context of a problem. In this research, artifact is implemented in a simulated home network environment as defined in solution design. The deployment replicates home networks in a controlled environment, traffic gets routed and inspected via artifact. Demonstration of artifact ensures that the solution is practically capable of addressing a real world problem. Solution demonstration is explained in detail in Chapter 5 "SUTMS Testing and Evaluation".

3.0.5 Evaluation

Solution evaluation, testing, and validation is the most important part of our research, without proper evaluation the artifact will have no integrity. Evaluation requires acceptable evaluation criteria based on recognized standards. The artifact in this research is tested against existing solutions and acceptable thresholds. Impact on network and system performance is evaluated and tuned accordingly. The evaluation and testing of SUTMS is based on two criteria, i.e., stress testing of the product under heavy load and solution efficiency in mitigating security threats. Detailed artifact evaluation is included in chapter 5 "SUTMS Testing and Evaluation"

3.0.6 Contribution

SUTMS framework sets the foundation for the next generation of affordable unified threat management devices for home networks. The research contributes to the field of cybersecurity for IP based devices, including IoT's and smart phones. Artifact goes through a series of design, development, and evaluation stages to come up with a final product, which addresses the security problems home networks are facing. COVID-19 has significantly increased remote work therefore it is a dire need of implementing corporate grade security for home networks, the research conducted addresses the present and future challenges in Chapter 6.

Chapter 4

SUTMS - Smart Unified Threat Management System

4.1 Design and Architecture

SUTMS is consisted of three core engines and one optional component, Fig.4.1 is the high-level graphical representation of SUTMS components and their interaction to come up with the final processing decision. The required modules are Intrusion, flow detection and Firewall, it is a security device therefore routing is categorized as optional. Dynamic routing protocols are currently not supported, as the solution is geared towards small networks. Flow detection engine correlates data with Intrusion module to improve signature efficacy, reduce CPU usage and memory consumption. Collaboration among flow detection and IDS engines results in signature optimization. The resulting logs can be feed to collectors for reporting, analytics, and troubleshooting. Suricata & NTOP are deployed as Intrusion and Flow detection modules respectively. Automated IoC feeds via STIX/TAXII turns the Iptables firewall into an antibot engine, black listed IP's can be blocked dynamically and without any user intervention. SUTMS takes care of the formatting, conversion and data duplication. The traffic flows are classified as egress and ingress, Fig.4.2 and Fig.4.3 depicts the egress traffic and ingress flow to the internet respectively, there are three main components of traffic flow:

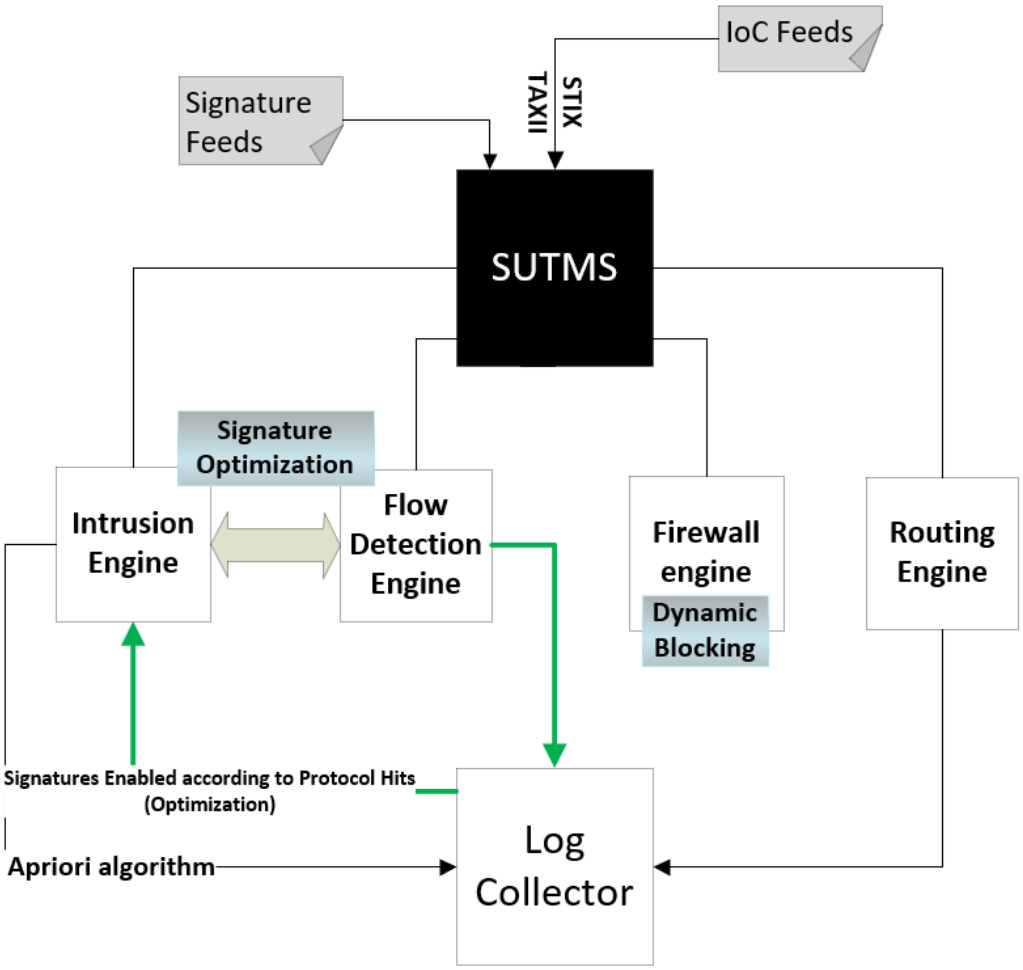


Figure 4.1: SUTMS High-Level Design

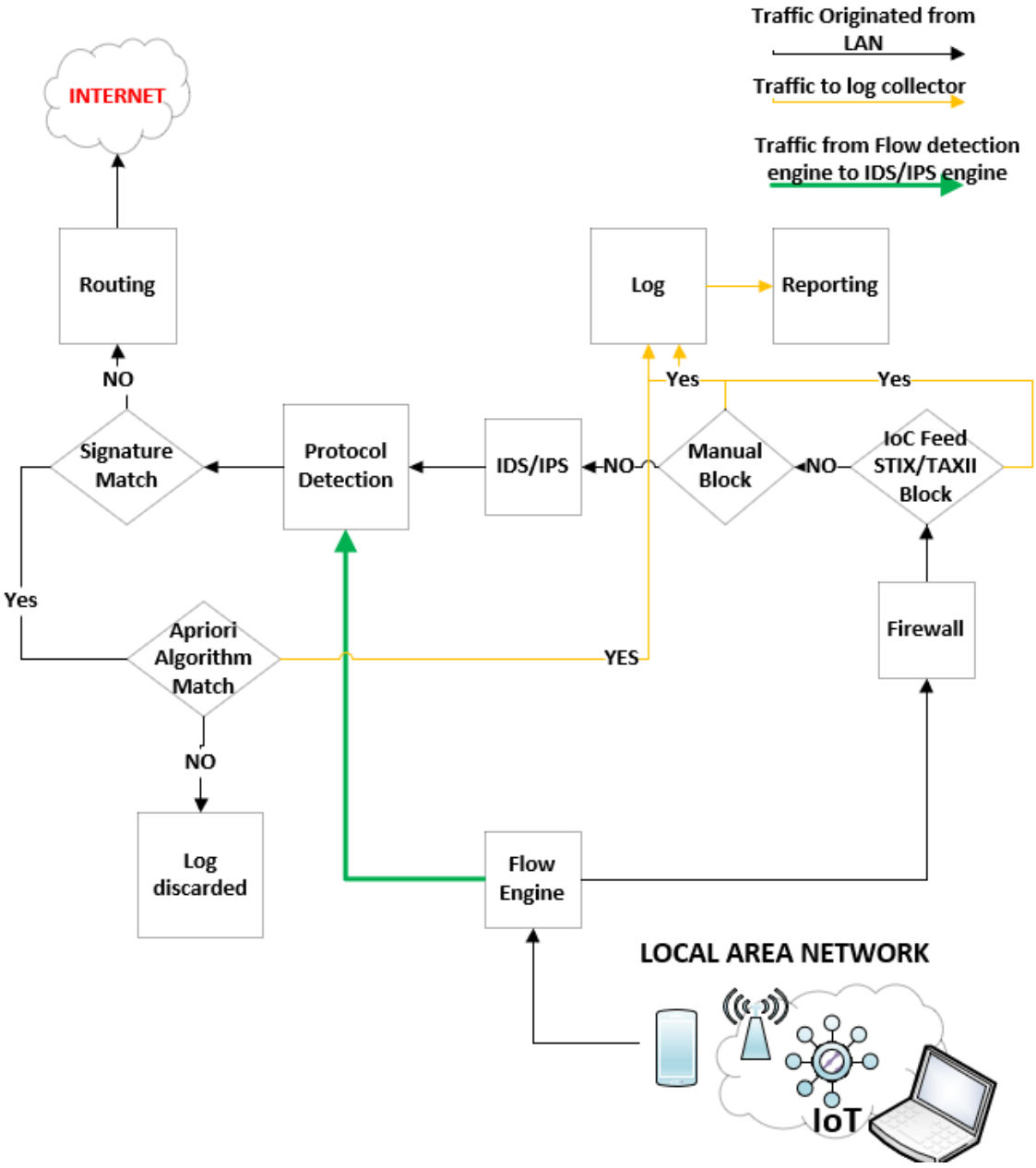


Figure 4.2: Flow of outbound traffic via SUTMS.

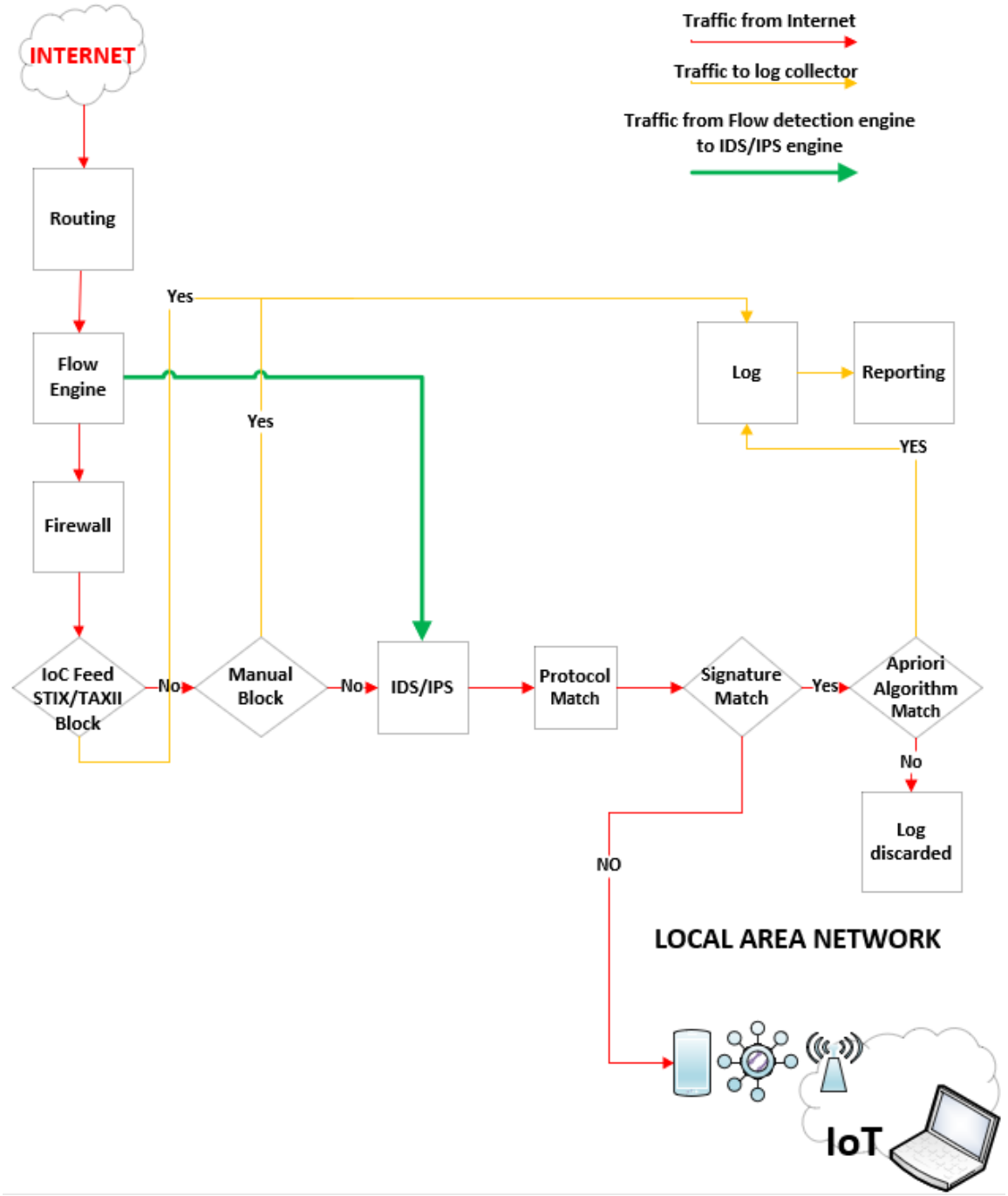


Figure 4.3: Flow of Inbound traffic via SUTMS.

- Traffic originated from the local area network, which includes IP traffic from home computing devices like laptops, desktops, smart devices, and IoT.

- Traffic destined to log collectors. It is an optional component and can be integrated with

security log correlations tools like SIEM. Similarly, logs can be used for reporting, trending, and data analytics.

- The flow collection engine detects the protocols used and forwards the output to IDS to only enable the rule sets relevant to protocols. Ingress traffic from the internet is processed according to Fig.4.3, the steps are similar to egress traffic except it is executed in the reverse direction. Raspberry Pi is based on the following open-source services.

a) Ubunto 22.04 LTS running on Vilros Rasberry Pi 4.

b) Suricata version 6.0.0 is selected as an intrusion detection engine, it provides real-time scanning and detection/prevention of malicious traffic.

c) Linux iptables provides firewall services.

d) STIX/TAXII feeds from Anomali is integrated into Anti-bot capabilities.

e) Ntopng probe for protocol and anomaly detection.

f) Rasberry Pi routing and access point service.

g) Apriori algorithm can be applied to log outputs for fine-tuning and noise elimination.

h) Elasticsearch and logstash are used for log aggregation.

4.2 Deployment Strategies

The artifact is implemented, evaluated, and analyzed on Rasberry Pi 4 with 8 GB of Ram, 32Gb of SD Card memory storage, and equipped with Quad core Cortex-A72 1.5 GHz processor [103]. Pi 4 is one of the latest models with multiple connectivity interfaces, i.e., Wi-Fi, Gigabit Ethernet, and Bluetooth. Different interfaces allow it to act as a mode of communication for a variety of computing devices, including IoT. SUTMS is a multipurpose compact device that can be deployed according to the network specifications, home network depends on Internet Service Provider and home user hardware requirements.

The device is proposed by considering various possible scenarios and implementations according to the existing home network infrastructure.

- Inline deployment with an access point

Fig. 4.4 represent the inline implementation, all the traffic will be forced through SUTMS. The inline deployment eliminates the need for an access point as SUTMS will be responsible for Wi-Fi access along with the advance inspection. Home networks where the ISP handoff is not ethernet will require an external modem as currently, RasberryPi doesn't have a built-in modem, Fig. 4.5 illustrates the scenario with an external modem.

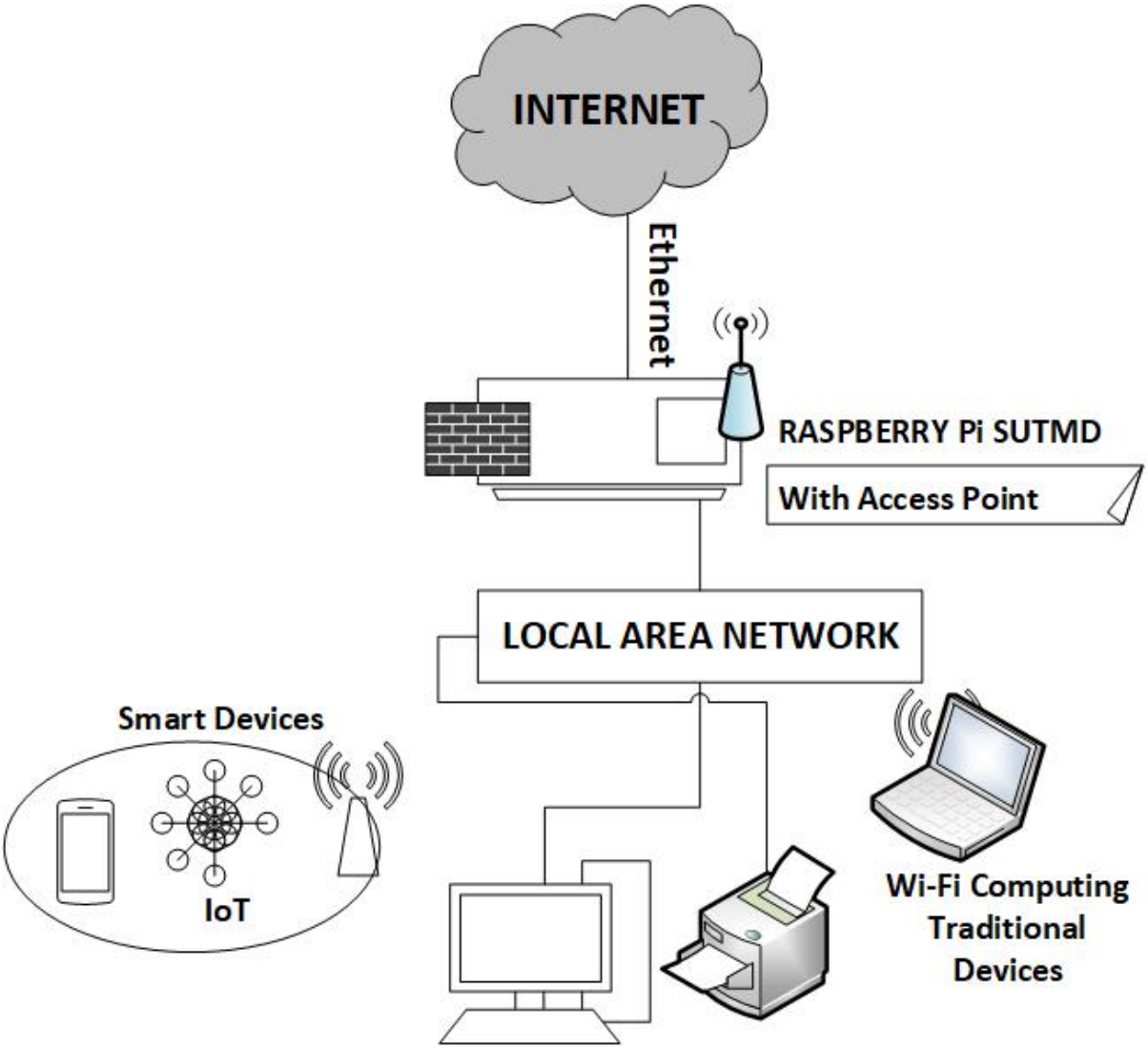


Figure 4.4: Inline Implementation of SUTMS.

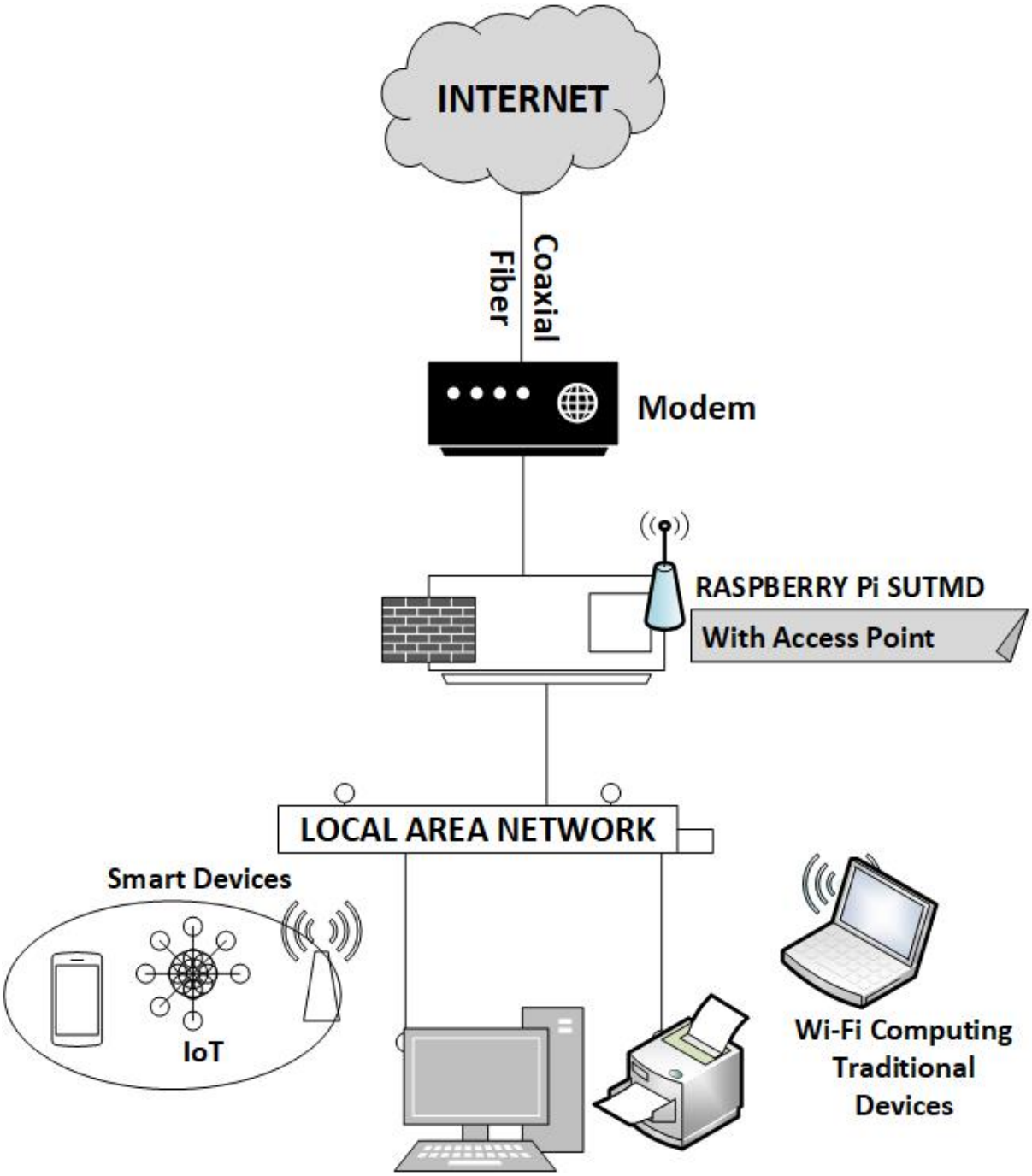


Figure 4.5: Inline Implementation of SUTMS with External Modem.

- Inline deployment without an access point

This is the most common form of deployment as most of the home networks already have an access point, we can leverage existing network infrastructure and install SUTMS

between ISP and Access point as shown in Fig. 4.6. This mode of deployment also requires minimum configuration changes on endpoints, as WiFi services won't be touched.

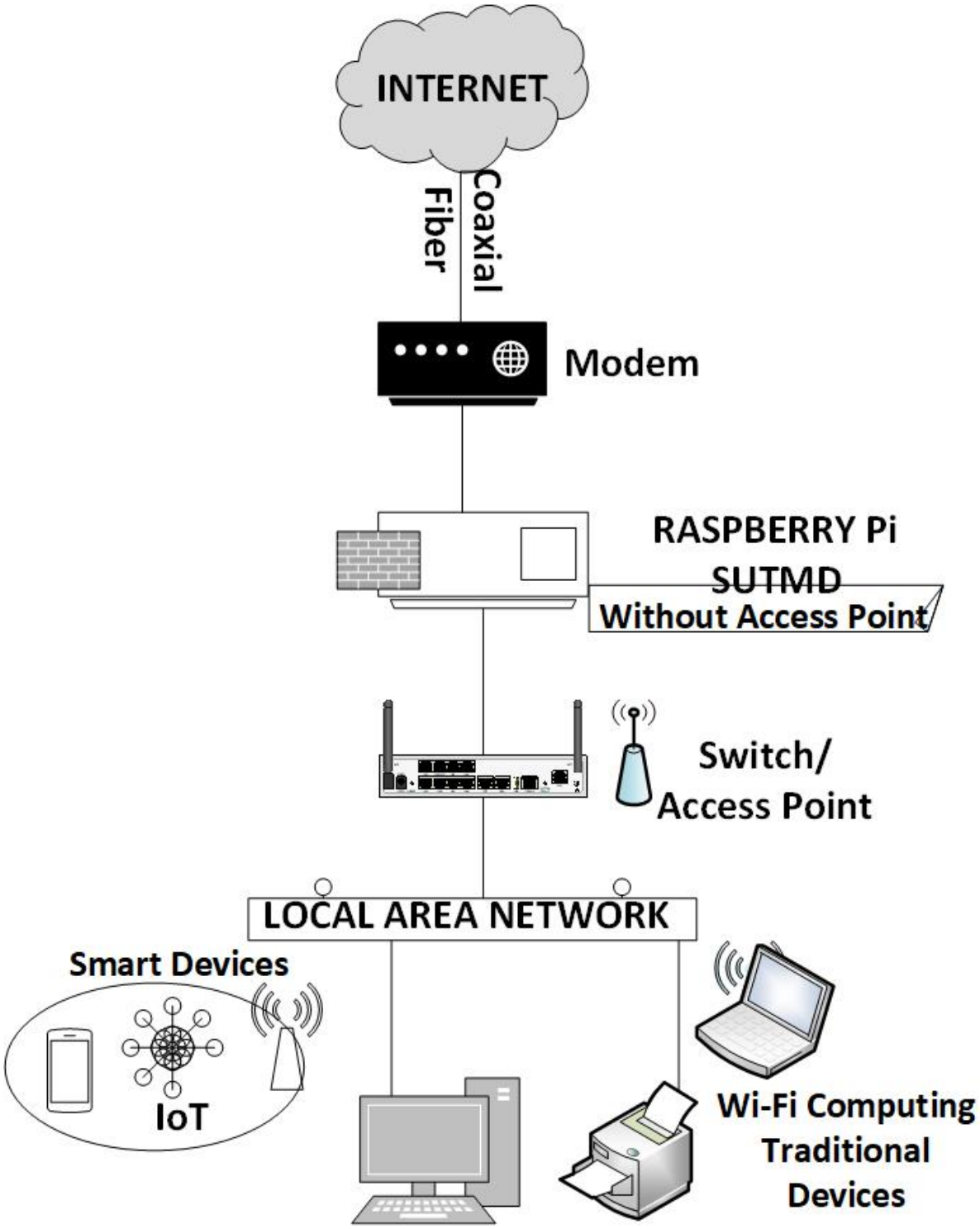


Figure 4.6: Inline implementation of SUTMS utilizing existing access point.

- Out of band deployment

Out-of-band implementation of SUTMS is also possible, however, the inspection capabilities are significantly reduced. The device won't be able to block any traffic, intrusions can be detected, and flow traffic can also be monitored. Out-of-band implementations are usually done either via an SPAN port Fig. 4.7 or require physical TAP installation.

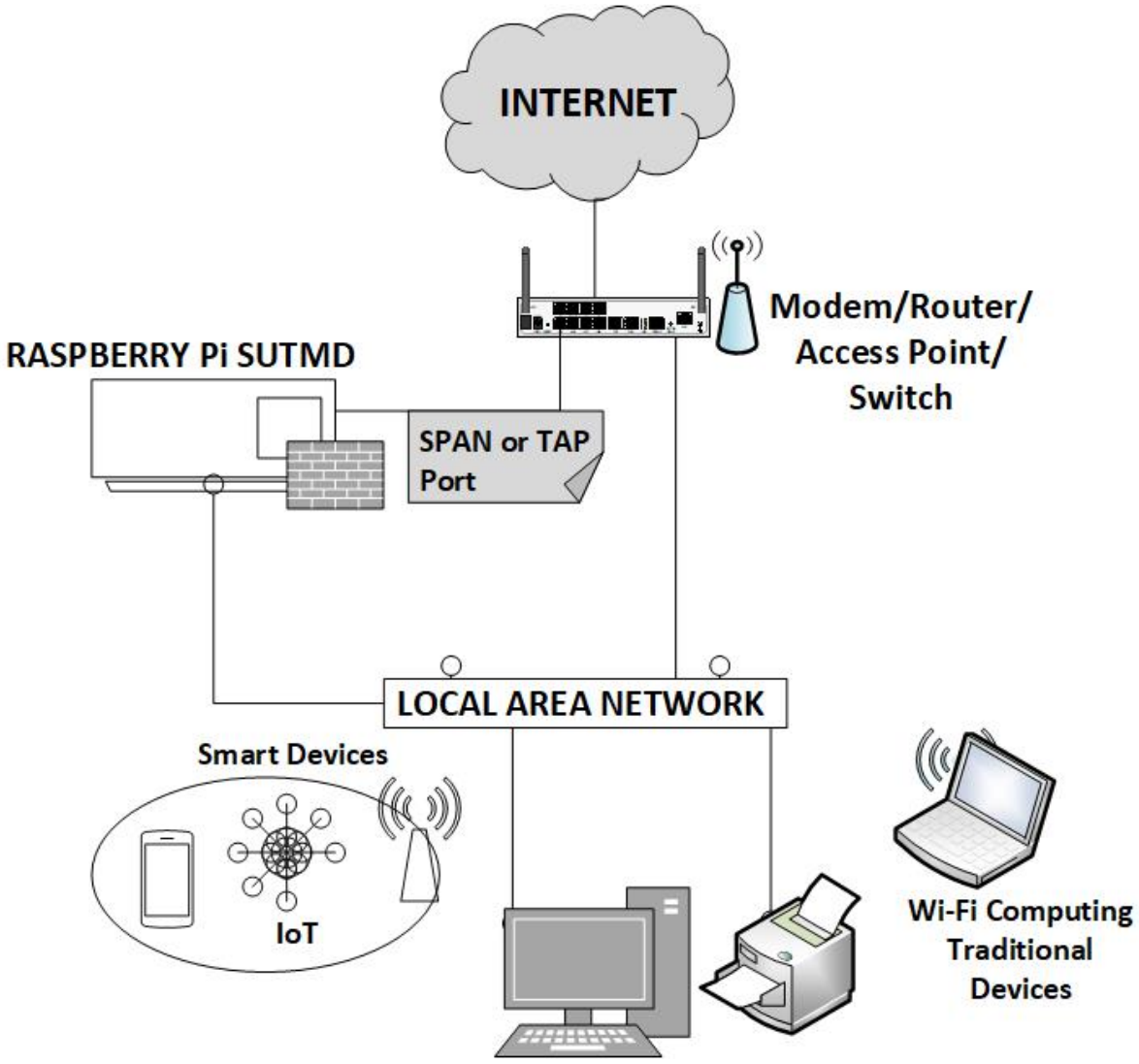


Figure 4.7: Out of band implementation of SUTMS.

4.3 Stateful Packet Inspection

Checkpoint invented stateful firewall inspection in the early 1990s [104]. Stateful firewalls are session-aware and keep track of the start and end of established connections (allowed traffic) in a session or state table. Filtering is conducted based on matching flows in a session table [105]. Stateful inspection greatly reduces processing for allowed traffic; flows for established connections are allowed by utilizing less resource-intense verification checks, and thus avoid the processing by standard firewall rules. Entries for allowed flows with standard protocol timeout settings are created in a session table, and the entries are deleted once the timeout expires. Any other traffic that is not allowed will be blocked, and no entries will be added to the session table. Typically, a session table consists of ¡source IP address (src_IP), destination IP address (dst_IP), protocol (port/service), source port (src_Port), session state and timeouts¿ [106]. Stateful firewalls dynamically allow return traffic by validating sequence numbers, timeouts, and state tables, and administrators do not have to worry about creating permanent access lists for return traffic. Session state and timeouts are the key components used to track and tear down connections gracefully.

Trabelsi, Zeidan, Shuaib, *et al.* [106] proposed architecture enhancements that can ultimately reduce processing times, and two new data structure fields were introduced: Session identifier & session verifier. The session identifier keeps track of ¡source IP address (src_IP), destination IP address (dst_IP), protocol (port/service), source port (src_Port)¿. The session verifier keep a trail of ¡session state and timeouts¿. The Trabelsi, Zeidan, Shuaib, *et al.* [106] architecture reduces the processing into single hash entries, compared to multiple hash processing engines, such as Netfilters [107]. Single hash processing requires less memory and computational power and reduces traffic filtering times.

Trabelsi and Zeidan [108] proposed an approach based on splay trees [109] for performance improvements and protection against denial of service (DoS) attacks. Fig. 4.8 illustrates the sample hash table used by Trabelsi and Zeidan [108] with the following

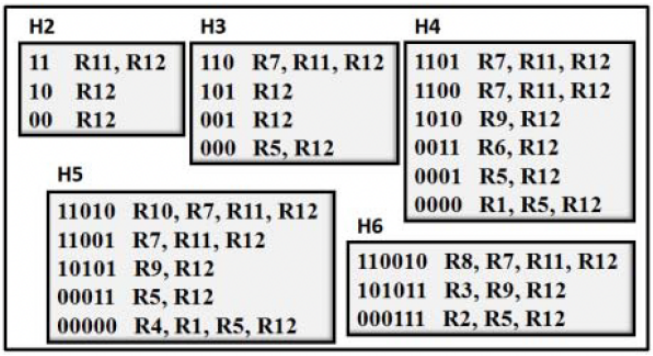


Figure 4.8: Collection of sample hash tables for improved stateful firewall by Trabelsi and Zeidan [108].

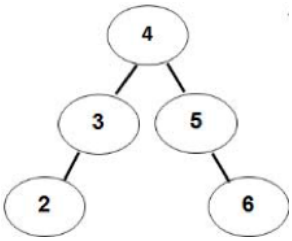


Figure 4.9: Splay tree developed from the hash tables in Fig. 4.8 [108].

characteristics:

- 11*, 10*, 00* are the binary representations of destination IP address prefixes
- R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, and R12 are firewall rules.
- R1–R11 ¡Allow Rules¿ and R12 ¡Default Deny Rule¿.
- H2, H3, H4, H5 and H6 are hash tables that group together prefixes by rule hits. DoS experimental testing [108] revealed that 90% of the packets matched some of the allow rules (Rules R1–R11), while 10% did not match any allow filtering rule from R1–R11, and therefore fell under the default deny rule, i.e., Rule R12.

For instance, destination IP address prefix 11 matches rules R11 and R12, which are listed in hash table H2. The splay tree generated from the collection of hash tables in Fig. 4.8 is depicted in Fig. 4.9 [108]. The splay tree in Fig. 4.9 is derived after applying a series of rotational operations, and counting the number of destination IP addresses prefixes

represented in Fig. 4.8. The default rule is the prime target for DoS attacks, and hackers can craft unnecessary traffic to overload the firewall. Rules, such as R12, gets the most hits as traffic increases; the splay tree in Fig. 4.9 moves the hash group with the maximum number of hits, i.e., hash group 4, to the top [109]. Consider the scenario of a continuous targeted web/Telnet scanning attack on the destination prefix 0011*. In this scenario, the scanning tool persistently targets port 80 (web) and 23 (Telnet). Among the various rules, Rule R6 belongs to hash group 4 and receives a majority of the hits. Following the general balancing principles [109], the splay tree then moves hash group 4 to the top. The hash group at the top will be processed first, then hash groups with fewer hits. In case of a DoS attack, Rule 12 (default) is processed first due to binary search using a splay tree structure that ultimately improves performance during peak usage. Since the UTM devices designed for home networks have generally limited processing power, the enhancements of Trabelsi, Zeidan, Shuaib, *et al.* [106] as well as Trabelsi and Zeidan [108] add significant value to the firewall inspection engine for UTM systems.

Xing, Wong, and Kumar [110] implemented a stateful firewall on a commercial ARM based architecture suitable for UTM systems. More specifically, they successfully integrated a stateful firewall engine on Keil Vision 4.7 with ARM MDK-Lite [110]. Home and IoT networks can take advantage of the integration effort in Xing, Wong, and Kumar [110] for the deployment of UTM systems on Arduino and Raspberry Pi platforms.

Recently, Melkov, Šaltis, and Paulikas [111] conducted performance evaluations that revealed that iptables [112] outperforms nftables [113]. Similarly, several studies have found that iptables is the more stable and efficient open-source Linux-based firewall for smaller implementations [114], [115]. Miano, Bertrone, Risso, *et al.* [116] have developed bpf-iptables, which achieves higher performance and throughput compared to traditional iptables by exploiting hooks for the extended Berkeley Packet Filter (eBPF) and eXpress DataPath (XDP) in the Linux kernel [117]–[122]. Firewalls with such performance enhancements and successful small-scale deployments are promising candidates for UTM-

based firewalls.

The latest firewall studies focus mostly on incorporating stateful inspection in switches [123], software-defined networks (SDN) [124][125], and the cloud [126]. Home networks are dominated by SOHO wireless internet routers; therefore, firewall studies on vendor-managed SDN and cloud solutions that are geared mainly towards corporate and enterprise networks likely will not have a significant impact on UTM firewall design.

4.3.1 SUTMS Implementation Of Stateful Inspection

SUTMS takes advantage of previous lightweight implementations of stateful inspection and deploy Iptables as a firewall and antibot agent. Iptables open source stateful firewall is deployed for IP address and TCP/UDP port filtration. Any traffic sourced from the internet is blocked, and outbound traffic is permitted according to need to know basis. Fig.4.10 depicts the flow of traffic initiated from home networks destined for the internet. The flow of traffic through iptables is processed as follows:

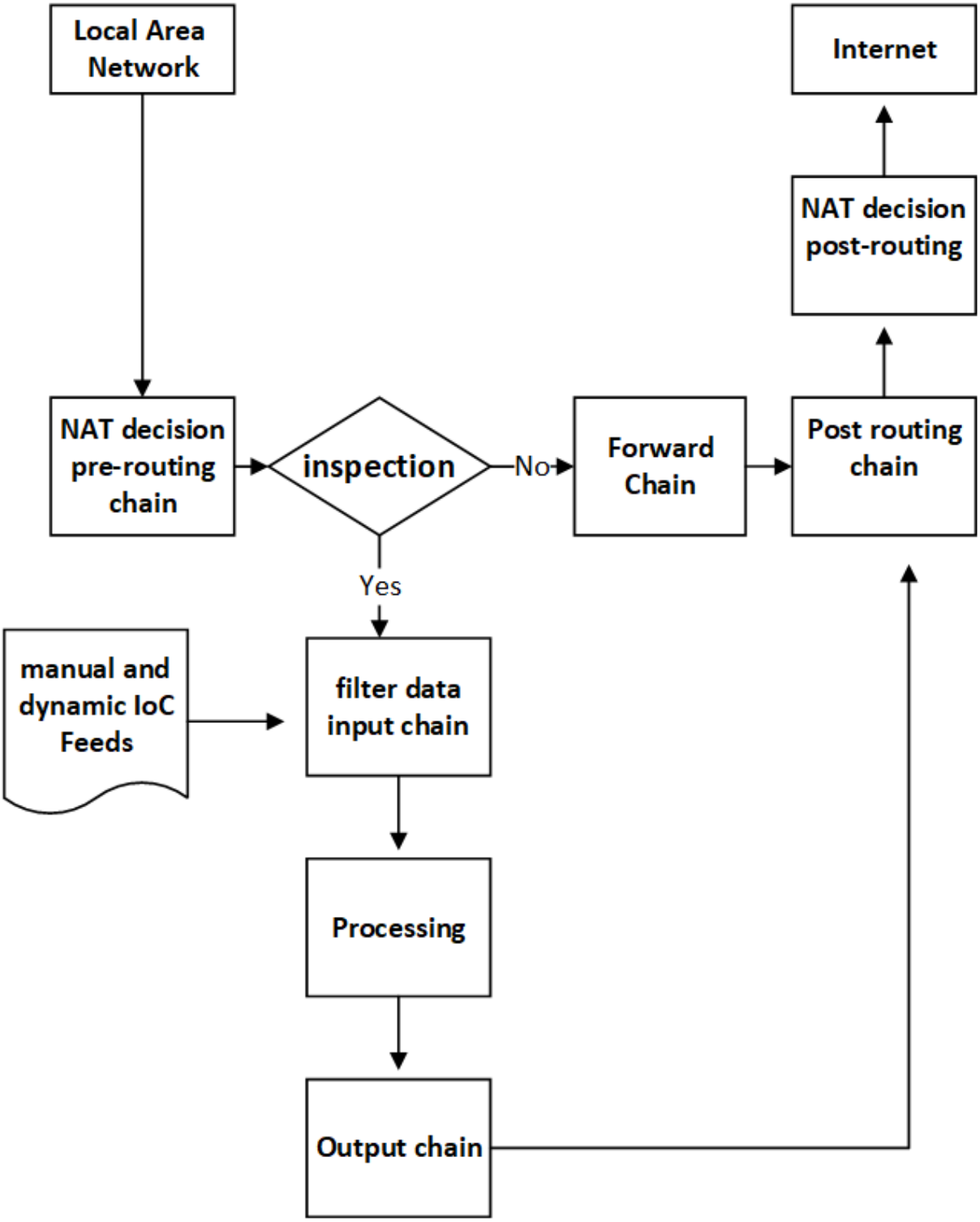


Figure 4.10: Flow Of Traffic via Firewall Engine.

i) The pre-routing chain is responsible for NAT decisions before the firewall rules get applied. NAT can be dynamic or one-to-one static. Address translation of multiple IP to

a single source IP requires Port address translation PAT.

ii) If there is no firewall inspection required, the traffic is forwarded directly to the outgoing interface. In case there is no route to the destination, the traffic will be dropped. In order to forward traffic, the iptables must have IP-forwarding enabled.

iii) In case of firewall inspection, the traffic is directed to the input chain and processed according to the applied rules. SUTMS iptables module has consisted of two sources of rules, one set of rules gets added manually by an administrator according to required functions. The other set of rules gets updated dynamically by incorporating IoC feeds from open-source threat intel platforms, i.e., Anomali via STIX/TAXII. Ingestion of dynamic feeds into iptables input chain provides protection against bots.

iv) Allowed packets goes through packet processing tasks and send to the output chain. Only packets destined or sourced from SUTMS are inspected by the output chain, all other packets are forwarded to the post-routing chain.

v) The post-routing chain is responsible for NAT packets after rules processing, whereas the pre-routing chain performs NAT before rules get applied.

Manual SUTMS Rules

The required rules attached to the input chain are stated below 4.1 and categorized as:-
#DNS Rule The rule is required for internal hosts to communicate to DNS servers, the destinations must be changed according to the local ISP.
#Allow-Web Traffic These rules allows HTTP and HTTPS traffic to the internet from RFC 1918 address space.
#SUTMS Management- Rule The rules allows management access to the SUTMS via

SSH and HTTPS over port 10000.

#NTOP Access Port The rules allows access to NTOP web portal for analyzing application and traffic flows captured by the SUTMS framework.

#Stealth Rule This rule works as a catch-all rule and drop any traffic that doesn't have a matching rule.

Listing 4.1: iptables rules

```
1  # DNS Rule
2  -A INPUT -p udp -m udp -s
       192.168.0.0/16,10.0.0.0/8,172.16.0.0/12 -d 8.8.8.8 --dport
        53 -j ACCEPT
3  # Allow-Web Traffic
4  -A INPUT -p tcp -m tcp -m multiport -s
       192.168.0.0/16,10.0.0.0/8,172.16.0.0/12 -j ACCEPT --dports
        80,443
5  # SUTMS Management -Rule
6  -A INPUT -p tcp -m tcp -m multiport -s
       192.168.0.0/16,10.0.0.0/8,172.16.0.0/12 -j ACCEPT --dports
        22,10000
7  # NTOP Access Port
8  -A INPUT -p tcp -m tcp -s
       192.168.0.0/16,10.0.0.0/8,172.16.0.0/12 --dport 3000 -j
       ACCEPT
9  # Stealth Rule
10 -A INPUT -j DROP
```

Rule Verification & Logging

The firewall rules can be verified by executing the command as shown below or accessing the WEBUI interface of the SUTMS framework i.e. Webmin. Webmin is an open-source web-based interface to manage Linux machines remotely via browser [127]. Webmin is highly customizable and various modules can be developed according to user requirements. The web interface gives easy access to home users to add/delete and modify firewall rules Fig .4.11

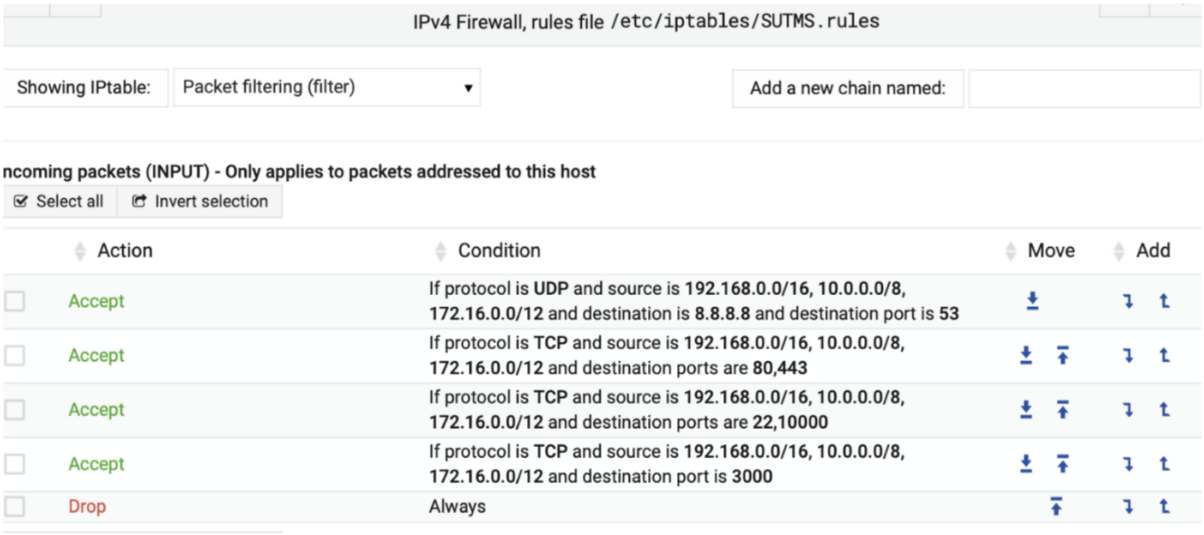


Figure 4.11: Webmin SUTMS Interface

Listing 4.2: iptables rules

```
1 root@SUTMS:/# iptables -L
2
3 # Warning: iptables-legacy tables present, use iptables-
    legacy to see them
4 Chain INPUT (policy ACCEPT)
5 target     prot opt source               destination
6 ACCEPT     udp  --  192.168.0.0/16       8.8.8.8
                      udp dpt:domain
```

```
7  ACCEPT     udp  --  10.0.0.0/8            8.8.8.8
                    udp dpt:domain ACCEPT     udp  --
   172.16.0.0/12          8.8.8.8              udp dpt:
   domain
8  ACCEPT     tcp  --  192.168.0.0/16        anywhere
                    tcp multiport dports http,https
9  ACCEPT     tcp  --  10.0.0.0/8            anywhere
                    tcp multiport dports http,https
10 ACCEPT     tcp  --  172.16.0.0/12         anywhere
                    tcp multiport dports http,https
11 ACCEPT     tcp  --  192.168.0.0/16        anywhere
                    tcp multiport dports ssh,webmin
12 ACCEPT     tcp  --  10.0.0.0/8            anywhere
                    tcp multiport dports ssh,webmin
13 ACCEPT     tcp  --  172.16.0.0/12         anywhere
                    tcp multiport dports ssh,webmin
14 ACCEPT     tcp  --  192.168.0.0/16        anywhere
                    tcp dpt:3000
15 ACCEPT     tcp  --  10.0.0.0/8            anywhere
                    tcp dpt:3000
16 ACCEPT     tcp  --  172.16.0.0/12         anywhere
                    tcp dpt:3000
17 DROP          all  --  anywhere           anywhere
```

The rules 4.2 by default are not persistent, in order to make the rules applied at reboot "iptables-persistent" package has been installed [128]. The file "/usr/share/netfilter-persistent/plugins.d/15-ip4tables" must be pointed to the rules file /etc/iptables/SUTMS.rules. The packets can be logged by creating a new chain and forwarding the messages to that

chain. Log messages that include a "denied" prefix in the message and have a severity log level of 4 can capture sufficient details.. Their log levels are debug, info, notice, warning, err, crit, alert, emerg [129], the command "-A FWLOGS -j LOG –log-prefix "denied: " –log-level 4" processes all packets to chain "FWLOGS" and denied messages are logged. The "FWLOGS" chain can be created either via CLI or Webmin, log messages are stored in the /var/log directory however it can be modified by editing /etc/rsyslog.conf. It is recommended to forward the logs to a Syslog server, as storing firewall logs locally can fill up the disk space on the Raspberry Pi SD card.

Ingesting STIX/TAXII Feeds

SUTMS takes advantage of Structured Threat Information Expression and Trusted Automated Exchange of Indicator Information [130]. STIX is a machine-readable language of threat intelligence information, and TAXII is a way to share that information [stix-taxii]. STIX-TAXII feeds allows ingesting IOCs automatically. The feeds can allow dynamic blocking of bad actors by using the inspection capabilities of iptables. The feeds can be collected using an API or manually via initiating an HTTP get request to the STIX-TAXII server. There many open-source STIX-TAXII threat intel feeds are available, some of the notable ones are MITRE [131], ANOMALI [132] and US Cybersecurity & Infrastructure Security Agency [133]. SUTMS ingests feeds via an API call provided by ANAMOLI [7] as shown below: -

curl -kv -o 'ioc_updates' -H 'Content-Type: application/json' 'https://192.168.200.160:8080/api/v1/intelligence' -d'{"token":"364d56025538c3dc193676cde8dd8ae9", "query":"confidence¿50 AND severity=very-high AND date_last¿-1d", "type":"csv", "size":100 }'

In order to secure the HTTPS transaction token can be retrieved by using the curl com-

mand and the output is shown below, IP = 192.168.200.160 is the IP address of the server and it requires a username/password for retrieving the data. SSL handshake is completed, followed by key exchange and certificate validation. The token acquired i.e. 364d56025538c3dc193676cde8dd8ae9 is used in the original API call.

root@SUTMS:/home/asif# curl -kv -H 'Content-Type: //application/json' 'https://192.168.200.160:8080/api/v1/login' -d '"username":"admin", "password":"xxxxxxxx"'*

Listing 4.3: Output of STIX/TAXII feed Ingestion

```
1  Trying 192.168.200.160:8080...
2  * Connected to 192.168.200.160 (192.168.200.160) port 8080
     (#0)
3  * ALPN, offering h2
4  * ALPN, offering http/1.1
5  * TLSv1.0 (OUT), TLS header, Certificate Status (22):
6  * TLSv1.3 (OUT), TLS handshake, Client hello (1):
7  * TLSv1.2 (IN), TLS header, Certificate Status (22):
8  * TLSv1.3 (IN), TLS handshake, Server hello (2):
9  * TLSv1.2 (IN), TLS header, Certificate Status (22):
10 * TLSv1.2 (IN), TLS handshake, Certificate (11):
11 * TLSv1.2 (IN), TLS header, Certificate Status (22):
12 * TLSv1.2 (IN), TLS handshake, Server key exchange (12):
13 * TLSv1.2 (IN), TLS header, Certificate Status (22):
14 * TLSv1.2 (IN), TLS handshake, Server finished (14):
15 * TLSv1.2 (OUT), TLS header, Certificate Status (22):
16 * TLSv1.2 (OUT), TLS handshake, Client key exchange (16):
17 * TLSv1.2 (OUT), TLS header, Finished (20):
18 * TLSv1.2 (OUT), TLS change cipher, Change cipher spec (1):
```

```
19 * TLSv1.2 (OUT), TLS header, Certificate Status (22):
20 * TLSv1.2 (OUT), TLS handshake, Finished (20):
21 * TLSv1.2 (IN), TLS header, Finished (20):
22 * TLSv1.2 (IN), TLS header, Certificate Status (22):
23 * TLSv1.2 (IN), TLS handshake, Finished (20):
24 * SSL connection using TLSv1.2 / ECDHE-RSA-AES256-GCM-SHA384
25 * ALPN, server did not agree to a protocol
26 * Server certificate:
27 *  subject: C=US; ST=CA; L=Redwood City; O=Anomali Inc; OU=
     Engineering; CN=STAXX; emailAddress=info@anomali.com
28 *  start date: Nov 18 21:08:37 2016 GMT
29 *  expire date: Nov 15 21:08:37 2031 GMT
30 *  issuer: C=US; ST=CA; L=Redwood City; O=Anomali Inc; OU=
     Engineering; CN=STAXX; emailAddress=info@anomali.com
31 *  SSL certificate verify result: EE certificate key too weak
      (66), continuing anyway.
32 * TLSv1.2 (OUT), TLS header, Supplemental data (23):
33 > POST /api/v1/login HTTP/1.1
34 > Host: 192.168.200.160:8080
35 > User-Agent: curl/7.81.0
36 > Accept: */*
37 > Content-Type: application/json
38 > Content-Length: 44
39 * TLSv1.2 (IN), TLS header, Supplemental data (23):
40 * Mark bundle as not supporting multiuse
41 < HTTP/1.1 200 OK
42 < Content-Length: 48
```

```
43 < Vary: Accept-Encoding
44 < Server: CherryPy/unknown
45 < Date: Wed, 22 Jun 2022 21:18:34 GMT
46 < X-Frame-Options: DENY
47 < Content-Type: application/json
48 < Set-Cookie: session_id=48
     d03f2cdf7835d070c005fffdce3b763eb90f8d; expires=Wed, 22
     Jun 2022 22:18:34 GMT; httponly; Path=/; secure
49 * TLSv1.2 (IN), TLS header, Supplemental data (23):
50 * Connection #0 to host 192.168.200.160 left intact
51 {"token_id": "364d56025538c3dc193676cde8dd8ae9"}
52 root@SUTMS:/home/asif#
```

Feed Formatting & Automation

The downloaded feed is stored in a local directory and can be accessed by using Linux command like more or cat as indicated below: -

root@SUTMS:/home/asif/Downloads# ls

ioc_updates

root@SUTMS:/home/asif/Downloads# more ioc_updates indicator,classification,confidence,itype,type,s

_site_netloc,feed_name,detail,date_last,actor,campaign,id,tlp

194.104.136.155,private,75,mal_ip,ip,very-high,limo.anomali.com:TAXII feeds:Emerging Threats C&C Server,limo.anomali.com,Emerging Threats C&C Server,"mal_ip:-194.104.136.155,malicious-activity",2022-06-22 04:02:48 PM,,,indicator–8dc28cca-34bf-4904-951e-eda05551551a,TLP:AMBER

154.56.0.108,private,92,mal_ip,ip,very-high,limo.anomali.com: TAXII feeds:Emerging Threats C&C Server,limo.anomali.com, Emerging Threats C&C Server,"mal_ip:-154.56.0.108,malicious-activity", 2022-06-22 04:02:40 PM,,,indicator–c73e43c4-

9f2a-4282-97fe-a4da3847a832, TLP:AMBER

The "ioc_updates" fetched two malicious IPs that met the criteria i.e., 194.104.136.155 & 154.56.0.108. Once the SUTMS has IoCs it requires proper formatting and automation, so that the rules can be dynamically created by iptables (anti-bot functionality). IoCs can be converted into a format that can easily be embedded into iptables by a simple while loop written below:

Listing 4.4: Program for IoC to iptables rule conversion

```
1  #!/bin/sh
2  i=0
3  while [ $i -le 1 ];
4  do
5     curl -kv -o 'ioc_updates' -H 'Content-Type: application/
          json' 'https://192.168.200.160:8080/api/v1/intelligence'
           -d '{"token":"364d56025538c3dc193676cde8dd8ae9", "query
          ":"confidence>5
6  0 AND severity=very-high AND date_last>-1d", "type":"csv", "
       size":100}'
7     perl -lne '/\b[0-9.]{7,15}\b/ && print $&' ioc_updates >
          blacklisted_IP
8  "$i";
9  i=$(($i+1));
10 done
```

The program above performs three main functions:

i) Download the IoCs.

ii) Extract IPs from the entire content.

iii) Remove any duplicate IPs.

4.4 Intrusion Detection Engine

Our study focuses on lightweight implementations of IDS engines that are suitable for UTM systems. Snort, developed by Martin Roesch [134], and Suricata [135] developed by the Open Information Security Foundation (OISF) are two widely considered signature-based open-source IDS engines. Bro, developed by Vern Paxson in 1995 and later re-branded as Zeek, has the qualities of both signature-based and anomaly-based systems [136].

Lightweight IDS

Jin, Chung, and Xu [137] developed a lightweight IDS algorithm for Controlled Area Network (CAN) bus networks. The drop attack, replay attack, and tempering attack modes were considered, and five signature fields were selected, namely ID, time interval, correlation, context changing amplitude, and value range. Fig. 4.12 illustrates how these attack modes are addressed with the selected signature fields in the lightweight IDS design [137]. The evaluation conducted by [137] demonstrated that IDS effectively detects a substantial number of malicious high-priority messages. This is achieved by utilizing a pre-determined ID set for normal traffic. An increased volume of malicious packets, which constitute a deviation from the normal ID set, is identified by the IDS as an anomaly and is promptly flagged as a potential threat. Time intervals are crucial to trigger correlation and context-changing amplitude. Context-changing amplitude is a specially designed signature used to detect variations in the content context of the data. The signature is crafted to identify alterations in the contextual aspects of the data. Anomalies can be detected by observing the changes in waveform patterns and amplitude. Correlation and context-changing signatures gets triggered for such behavior-based anomalies. The IDS

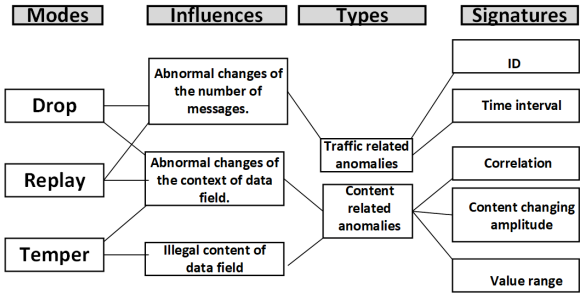


Figure 4.12: Illustration of lightweight IDS relationships [137]: Three different types (modes) of attacks are influenced by two distinct anomaly types and tested against specific signature fields (signatures).

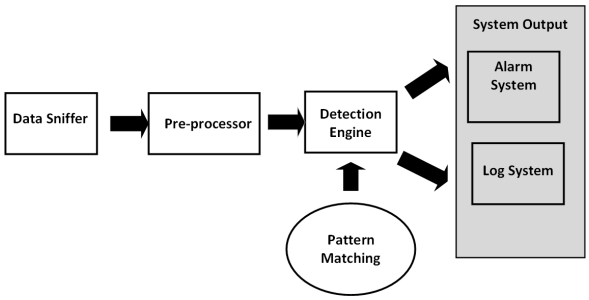


Figure 4.13: Architecture of Snort, version 2.8 [139], [140].

developed and evaluated by Jin, Chung, and Xu [137] allows the creation of signatures based on anomalies. However, as the number of signatures increases, the computational processing requirements escalate substantially, which can overwhelm the limited computational resources of UTM systems.

Snort

Snort depends on the Libpcap (an open-source C-language library for capturing network packets) and AF_PACKET (a socket in Linux that allows an application to receive and send raw packets) packet capturing libraries to capture packets from the network interface [138]. Given the importance of the Snort architecture for IDS for UTM systems, we briefly review the Snort architecture, which is illustrated in Fig. 4.13 [139], [140]. i) Data Sniffer: Raw data is captured, and packets are sent to preprocessors. Libpcap, AF_PACKET [138], and PCAP/LINPAC [141] are used for packet capturing.

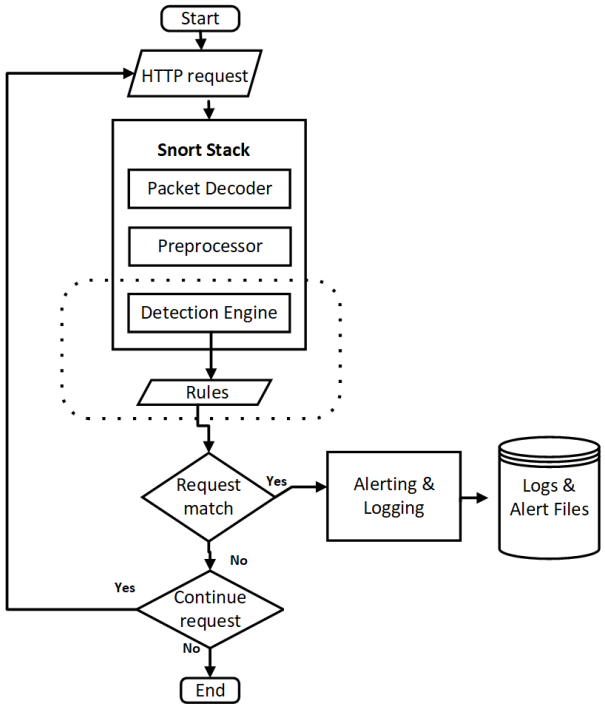


Figure 4.14: Snort rule matching process flow [142].

ii) Preprocessor: It is critical for Snort to identify the packets that it has to analyze. Snort uses preprocessors/plugins to classify traffic. *Http_inspect* is a commonly used preprocessor to identify web traffic, and can be customized to web server type, IP address, or port numbers [139]:

For instance, *"preprocessor http_inspect_server: server 10.1.2.3 profile apache"* utilizes the "preprocessor http_inspect_server" to conduct a thorough inspection of the HTTP protocol, tailored for the "Apache" server at IP address 10.1.2.3. This rule enables Snort to perform deep packet inspection on web applications hosted on Apache, thereby triggering alarms for potential attacks. *sfPortscan* is another useful plugin to decode protocol behavior and port scanning.

iii) Detection Engine: The detection engine is responsible for identifying, alerting, and blocking (IPS) intrusions. Once Snort can classify a packet and its behavior, the packet is compared against a set of rules in a database. Fig. 4.14 [142] illustrates the rule matching process, Zalbina and Stiawan [142] use Snort-specific rules as criteria for *Http_inspect* in

Table 4.1: Performance comparison of various IDS versions for a 10 Gbps TCP Flow [138].

	Snort 2.8 10 Gbps AF Pkt.	Suricata 2.1 10 Gbps AF Pkt.	Snort 3.0 10 Gbps Libpcap	Snort 3.0 10 Gbps AF Pkt.	Suricata 4.1 10 Gbps Libpcap	Suricata 4.1 10 Gbps AF Pkt.
CPU usage	57%	11.5%	46%	40%	30%	10%
Memory usage	2%	6%	0.10%	0.20%	0.20%	0.10%
Drop rate	13.8%	5.9%	0%	0%	0%	0%

their study.

We further illustrate Snort with the following sample rule which has the following noteworthy components [139]:

alert tcp $EXTERNAL_NET any –> $HOME_NET 7597 (msg:"MALWARE-BACKDOOR QAZ Worm Client Login access"; flow:to_server,established; content:"qazwsx.hsq"; metadata:ruleset community; classtype:misc-activity; sid:108; rev:12;)

alert = action is set to detect

protocol = tcp

destination port = 7597

direction = External (internet) to Home Net (private) networks.

State = Established connections.

Malicious content = qazwsx.hsq

sid = unique identifier

The rule gets triggered if a connection attempt is made from the internet to internal networks over port 7597 with a payload of "qazwsx.hsq". The same rule can be set to block by changing "alert" to "drop." *flow:to_server, established* has added performance benefits as only established connections will be scanned. Alerts can be sent to log servers via Syslog, and the rules database can be set for automatic updates.

Table 4.1 [138] illustrates performance characteristics of various IDS versions on high-

speed 10 Gbps networks. Suricata 4.1 AF Pkt exhibits lower CPU usage (of 10%) than all other IDS versions. Also, Suricata 4.1 AF Pkt has the same low memory usage (of 0.1%) as Snort 3.0 Libpcap, with 0% packet drop. These performance results from [138], were further corroborated by a study of Kumar, Chandak, and Dewanjee [143] that found that Suricata has generally better performance and higher throughput than Snort.

Cruz, Goyzueta, and Cahuana [144] implemented Snort [139] on a Raspberry Pi 3 (1.2 GHz CPU, 1 GB of memory), demonstrating the use of Snort in home network UTM deployments. However, the evaluation study in [144] was conducted with limited attacks and does not detail the number of Snort signatures.

Suricata

Suricata was created to overcome some of the drawbacks of Snort, e.g., Snort is single-threaded, and Suricata is multi-threaded [145]. Suricata takes advantage of multiple CPUs and can distribute processing across multiple threads. Suricata has four thread modules, see Fig. 4.15.

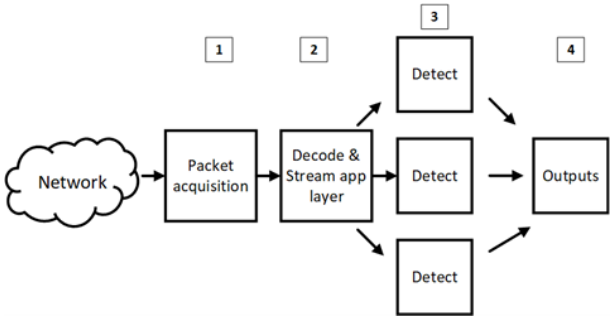
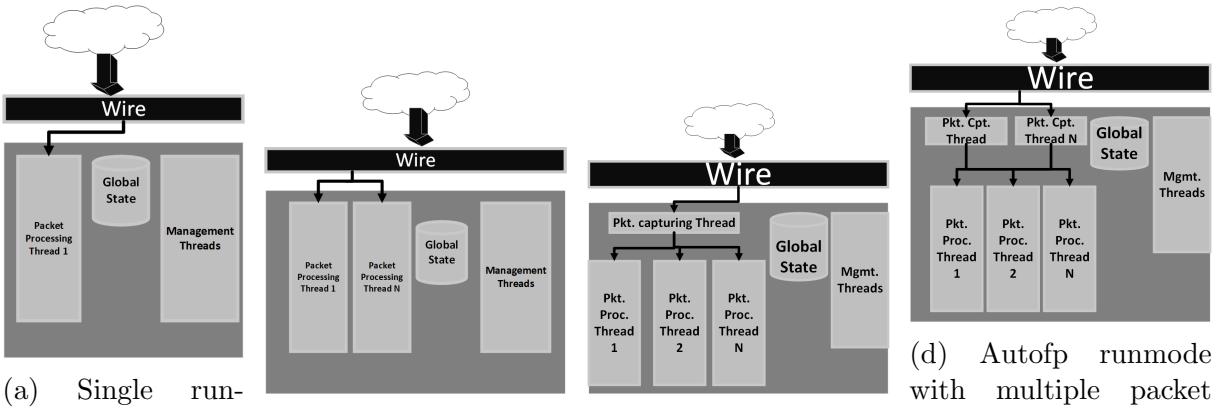


Figure 4.15: Suricata 6.0.0 architecture [146].

The packet acquisition thread captures packets using `Libpcap`, similar to Snort [138]. Packets are decoded and classified, and the flow state is established by the decode and stream application layer. Suricata can parse application layer (Layer 7) protocols, such as HTTP, using a stateful parser and can remove malicious content [147]. The detection thread runs multiple instances of threads concurrently; rules are scanned, and alerts for



(a) Single runmode. (b) Worker runmode. (c) Autofp runmode. (d) Autofp runmode with multiple packet capture threads.

Figure 4.16: Illustration of different Suricata runmodes.

any matches are sent via the outputs thread.

Suricata Runmodes Suricata core components are threads, modules, and queues. Threads are processes. Suricata can run multiple threads (processes) to enhance performance. Modules are responsible for functionality. The modules can be used for processing, decoding, and ingesting packets. Packets are transferred from one thread to another via a queue. The runmodes are responsible for arranging threads, modules, and queues. Suricata features the following runmodes:

i) Single: Traffic that enters the network interface card is be processed by a single thread, see Fig.4.16a; multi-threading is not used.

ii) Worker: In this mode, the network interface driver load balances packets between multiple Suricata packet processing threads, see Fig. 4.16b. An advantage of the Suricata worker runmode is that load balancing is offloaded to the network interface.

iii) Autofp: The Suricata autofp mode is used to process packet capture (pcap) files or NFQueue [148] IPS modes. In NFQueue, iptables firewall modules are configured in routed mode to forward traffic to IPS for inspection. NFQ requires the Netfilter package and the following iptables command to launch the packet inspection of Suricata: *"sudo iptables -I FORWARD -j NFQUEUE"* [148]. Autofp can be deployed with a single packet capturing thread (Fig. 4.16c) or multiple packet capturing threads (Fig. 4.16d). Packets

are decoded by the packet capturing threads and then rendered to the packet processing threads.

Suricata Signature Action Order Suricata has four types of actions to perform on a packet, namely Pass, Drop, Reject, and Alert. Suricata has to be running in IPS inline mode for the Drop and Reject actions to take effect.

i) Pass: As the name implies, upon a signature match, the packet is allowed to pass and scanning will be stopped for that packet.

ii) Drop: Packets matching any signature are silently dropped, and a log message will be generated. The receiver will not be notified about the packet's status; scans, such as reconnaissance scans [149], may have limited information about the transaction. As a best practice, "Drop" actions are preferred over "Reject" due to their stealthy nature.

iii) Reject: Reject performs a similar function as "Drop", except for notifying the client with a reject message in the form of a Reset packet. An attacker can craft a packet with a "reset" bit set to analyze the response from the target and employ it for fingerprinting and scanning. Packet crafting tools, such as scapy, can be used to craft such custom packets. This way, an attacker can trick the IPS device by crafting abnormal packets, such as sending a "reset" packet without an initial conversation. Reset packets can also facilitate DDoS attacks and may exhaust resources on the responding IPS.

iv) Alert: In case of a signature match, packets are allowed, and an alert is be generated for an analyst to investigate further. The Alert action is a typical configuration for the IDS mode. IPS can be deployed in span or TAP mode to scan traffic and generate alerts based on rule matches. In SPAN mode, the network device is configured to send the traffic from the uplink ports to any available destination port on that switch. An administrator can plug in the Suricata machine network interface to the destination port for packet analysis. The TAP mode functions similar to the SPAN mode, with the exception of a

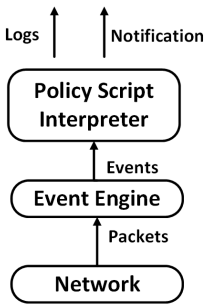


Figure 4.17: Illustration of Zeek 4.0 architecture [136].

network TAP being connected inline to forward the traffic to a Suricata machine.

Zeek

Formerly known as Bro, Zeek is a hybrid IDS that has adopted functions of both signature and anomaly (behavior)-based systems. Zeek can generate events from the captured packets. Zeek has a policy script interpreter engine with powerful features that are highly customizable. Fig. 4.17 illustrates the core components of the Zeek architecture.

Event Engine: Traffic received from the network is processed by the "event engine," and raw data is converted into events. Events can be used to track transaction logs. For example, clients going to a website can generate *HTTP::http_methods:* (http_methods could be GET, POST, DELETE, etc.). DNS events, such as *dns_mapping_lost_name: event*, can be configured to identify name resolution problems. Failed DNS queries that were successfully answered in the past can trigger such events. Zeek can also process stored pcap files, and log files native to Zeek are generated from raw data. Log files depend on the used protocol. For example, an HTTP transaction can create *conn.log*, *dns.log*, and *http.log*. The analyst may elect to only analyze *dns.log* if the focus of the investigation is to identify malicious domains.

Policy Script Interpreter: Zeek has a custom scripting language that executes a set of instructions. Scripts can be utilized to take action on the activities detected by the event engine. The Zeek sample script below creates notifications for ssh brute force attempts. The script starts by loading protocol and framework definitions, whereby the "Notice"

framework allows Zeek to flag any pattern outside of normal behavior. Three bad password attempts "*password_guesses_limit= 3*" within a period of five minutes "*guessing_timeout = 5 mins*" will be marked as an SSH brute force attempt.

Table 4.2: Comparison of Snort, Suricata & Zeek (bro).

Features	**Snort**	**Suricata**	**Zeek (bro)**
Detection Method	Signature based	Signature based	Signature & anomaly based
Blocking	IDS / & IPS	IDS / & IPS	IDS Only
Complexity	Easy to implement / configure	Easy to implement / configure	Complicated to set up
OS Support	Linux, Windows	Linux, Windows	Linux
Multi-Threaded	No	Yes	No
Signature Updates	Can be configured for automatic updates	Can be configured for automatic updates	Signatures are not the preferred detection method
Technical Features	Thousands of signatures with wide variety of protocol support	Thousands of signatures with wide variety of protocol support	Features depend on Zeek scripting language
CPU Usage	Medium CPU usage in high-traffic networks [150]	High CPU usage in high-traffic networks [150]	Low CPU usage in high-traffic networks[150]
Integration	Can be easily integrated with security inform. & event managmt. (SIEM), threat intellig. platf., and firewalls	Can be easily integrated with SIEM, threat intellig. platf. & firewalls; turnkey solution available, i.e., SELKS	Integration with other tools is complicated
Customer Base	Cisco owns Snort and has large customer base	Smaller footprint	Smaller customer base than Snort and Suricata
User Acceptance	High – Due to proven track record and Cisco acquisition	Medium – Good for small environm. due to integr. /w Kibana & Elasticsearch	Low – Due to Complexity and Lack of IPS functionality

Listing 4.5: Zeek Script For SSH Brute Force

```
@load base/protocols/ssh
@load base/frameworks/sumstats
@load base/frameworks/notice
module SSH;
export {
        redef enum Notice::Type += {
                Password_Guessing
            } ;
const password_guesses_limit: double = 3 &redef;
const guessing_timeout = 5 mins &redef;
      }
```

Zeek can also process signatures. However, signature based detection is not one of the preferred detection methods [136]. The following example signature triggers a "signature_match" event when an anonymous FTP attempt is made:

Listing 4.6: Zeek Signature Template

```
signature zeek-sig {
    ip-proto == tcp
    dst-port == 21
    payload /.*anonymous/
    event "FTP Anonymous access!"
}
```

The Zeek "signature_match" event has the following format:

event signature_match(state: signature_state, msg: string, data: string).

"*state*" includes information about the FTP connection, "*msg*" contains "FTP Anonymous access!" and "*data*" is the payload "/.*anonymous/". Zeek makes use of regular expressions to filter out the content of interest from the payload.

4.4.1 SUTMS IDS Implementation

Suricata version 6.0.4 is selected as an intrusion detection service, Suricata's multi-threaded architecture and integration with other engines like Logstash, and Kibana makes it a better choice over Snort. While Snort operates on a single thread, Suricata leverages a multi-threaded approach for enhanced performance [145]. By harnessing the power of multiple CPUs, Suricata efficiently distributes processing tasks across multiple thread, which makes it a ideal IDS candidate for UTM. The Suricata rule written below has the following key components [146].

alert http $HOME_NETany− > $EXTERNAL_NET 80 (msg:"TROJAN"; flow:established,to_server; flowbits:isset; content:"trojan" ; pcre:"/trojan .[0-9]3,/i"; classtype:trojan-activity; sid:200; rev:2;)*

alert = action is set to detect

protocol = http

destination port = 80

direction = Home Net (private) networks to External (internet).

State = Established connections (flowbits:isset).

pcre = regex search set to trojan .*[0-9]3,/i

malicious content = trojan

sid = unique identifier

Suricata rule format is very similar to Snort, except for the ability to specify application layer protocols like HTTP, DNS, etc. Raspberry Pi 4 has a single-core architecture therefore it won't be able to utilize the benefits of multi-threading processing, however future releases of Rasberry Pi can be multi-core. Suricata rules are downloaded automatically via running the command "sudo Suricata-update", key output details are shown below.

Listing 4.7: Suricata Installation Log Showing 27716 enabled rules

```
1 asif@SUTMS:~$ sudo suricata-update
2 10/8/2022 -- 19:11:34 - <Info> -- Found Suricata version
    6.0.4 at /usr/bin/suricata.
3 10/8/2022 -- 19:11:45 - <Info> -- Backing up current rules.
4 10/8/2022 -- 19:11:52 - <Info> -- Writing rules to /var/lib/
    suricata/rules/suricata.rules: total: 35266; enabled:
    27716;added: 1642; removed 84; modified: 1492
```

The key takeaway from the above output is that signature download process automatically detects the version and writes the rules to the "var/lib/suricata/rules/suricata.rules" file. Another notable fact is by default 27716 rules gets enabled out of 35266 total number of rules. Scanning 27716 rules requires intense processing power which could significantly degrade the performance of UTM devices. In order to avoid any negative impact to performance, SUTMS takes advantage of NTOP probe to only enable the signatures for the protocols that are used in an environment. In order to verify Suricata is working as expected, test signature "suricata_test_rule.rules" is created and traffic is generated from a remote system simulating as an attacker. Logs were observed in "fast.log" file for matching signature description *"This is SUTMS test signature"* as shown below:-

root@SUTMS:/var/log/suricata# suricata -S suricata_test_rule.rules -i wlan0
root@SUTMS:/var/log/suricata# more fast.log
08/10/2022-21:53:01.707410 [**] [1:2008124:0] This is SUTMS test signature [**] [Classification: (null)] [Priority: 3] TCP 192.168.200.155:56314 -¿ 192.168.200.156:22

Signature Refinement

Data derived from ntop prob(discussed in later section) revealed that only certain protocols are used in home networks, therefore it makes sense to disable the signatures for

unused protocols. This significantly improves the performance of the SUTMS IDS inspection engine. To automate the disablement of unwanted signatures Suricata config file i.e., "disable.conf" is created in /etc/suricata directory. Regex is used to filter out unused protocols, rules with "pcre" and older irrelevant rules are also disabled, "pcre" is a very resource intense process and can overload the UTM device. It reduces the number of Suricata rules from, 27716 to 5565.

Listing 4.8: Suricata rule generation based on relevant protocols

```
1 root@SUTMS:/var/lib/suricata/rules# suricata-update
2 14/8/2022 -- 17:30:42 - <Info> -- Loading /etc/suricata/
    disable.conf.
3 14/8/2022 -- 17:30:42 - <Info> -- Loading /etc/suricata/
    enable.conf.
4 14/8/2022 -- 17:31:03 - <Info> -- Writing rules to /var/lib/
    suricata/rules/suricata.rules: total: 35267; enabled:
    5565; added: 0; removed 0; modified: 33
```

4.5 Anomaly Detection Engine

Anomaly-based Intrusions Detection Systems (AIDS) have a fundamentally different method of identifying intrusions than signature-based IDS, which mainly rely on rules (signatures). Signature-based IDS are effective against the known attacks, but are ineffective against any zero-day attacks. AIDS is designed to protect networks against advanced persistent threats (APTs) [151] that do not have any known signatures. AIDS establishes a baseline and issues alarms for anomalies. Traffic or protocol patterns that deviate from the baseline are marked as an anomaly; whereby, AIDS tends to have many false positives [152]. We conduct a comprehensive up-to-date survey of the anomaly detection mechanism that are suitable for UTM systems. AIDS approaches can be categorized into approaches based on

statistics [153], knowledge [154], machine learning [155], association rule data mining, and flow data. For brevity, we abbreviate "AIDS" to "IDS" in the remainder of this section.

Statistics-based IDS

Statistics-based IDS monitors traffic to learn patterns and identify low-probability events; statistical operations, such as mean, mode, median, ratio, and standard deviation, are applied to detect abnormal behavior. Khraisat, Gondal, Vamplew, *et al.* [156] notes that two types of statistical IDS models are commonly used, namely univariate statistical IDS models and multivariate statistical IDS models. Univariate statistical IDS models are based on a single variable to classify the normal traffic, while multivariate statistical IDS models rely on two or more variables to build relationships. The multivariate method can involve complex statistical operations when correlations of multiple variables are considered. Multivariate statistical IDS models tend to reduce false alarms [157], [158], but have difficulty in managing high-dimensional data [156]. Table 4.3 lists sample use cases that can predict data hoarding and data exfiltration based on statistics. Statistical algorithms can be applied to correlated use cases to improve detection capabilities. Table 4.4 summarizes the existing IDS statistical models that are suitable for UTM systems.

Table 4.3: Use cases for statistical analysis.

Use Cases – Alerts via Syslog/email
Source communicating to a new domain never seen in last 30 days
Source is sending/receiving more traffic than usual
More outbound/inbound encrypted traffic than usual
New protocols/applications are discovered
Machines from the internal network are communicating to Botnet Command & Control
Single source communicating with multiple destinations in a given time
New protocols/applications are discovered

In the following, we briefly discuss three representative statistics-based IDS approaches from Table 4.4; the other approaches in Table 4.4 are based on similar statistics-based

principles following the methodologies and providing the features summarized in Table 4.4.

Generally, IDS algorithms and methods are derived from statistical operations, such as variance and entropy, as well as univariate and multivariate statistics. Xue and Hu [159] developed a method to detect worm propagation. The developed method uses the First Contact Connection (FCC) as one of the key variables to determine abnormalities. FCC is defined as the first time that a connection request is made to an IP address (that has never been used before). FCC is employed as a detection index. FCC request packet size and failed probability are the two metrics for statistical processing. The two parameters are compared with prescribed thresholds to decide on the marking as a worm or as normal traffic. The algorithm is based on FCC failure attempts and packet size to determine abnormalities. A key difficulty for the FCC based approach is to distinguish between normal and malicious connection requests and packet sizes.

Table 4.4: Comparison of statistical-based IDS models suitable for UTM systems.

Proposed by	**Year**	**Algorithm/Methods**	**Key Features**
Ye, Emran, Chen, *et al.* [157]	2002	Multivariate Approach	Based on Hotelling's T-Squared method; limited to host-based intrusion detection
Kumar, Kumar, and Srinivasan [160]	2007	Six sigma technique	Utilizes raw network data to determine thresholds and uncertain traffic patterns
Tao, Lin, and Liu [161]	2010	IDSV Algorithm	Detects intrusions on different applications with statistics variance method
Xue and Hu [159]	2015	Worm detection algorithm	Prevents worm propagation and detects worm in the first connection; based on First Contact Connection (FCC) statistical indicators
Rastegari, Lam, and Hingston [162]	2015	Rule Based Algorithm	Entropy, volume combined with rule base techniques are utilized.
Ghanshala, Mishra, Joshi, *et al.* [163]	2018	Statistical learning with feature selection	Achieves accuracy of 98.9%, false positive rate of 1.6% without extensive monitoring of memory writes
Siffer, Fouque, Termier, *et al.* [164]	2020	SPOT Algorithm	Detects anomalies using extreme value theory
Carreón, Gilbreath, and Lysecky [165]	2020	Cumulative Distr. Fcts. (CDFs)	Incorporates system timing with CDF to detect anomalies in embedded system; few false positives and high accuracy
Das, Hamdan, Shukla, *et al.* [166]	2023	Network port statistics	Proposed an intrusion dataset UNR-IDD based on network statistics
Kuo, Tseng, and Chou [167]	2023	Sequent. Prob. Ratio Test SPRT)	Wormhole attack detection in metaverse environment
Sasikala and Vasuhi [168]	2023	Logistic regression	Predicts anomalies with 90% accuracy

Profiling every connection attempt is a resource-intensive process for IDS packet-capturing modules. Siffer, Fouque, Termier, *et al.* [164] take advantage of the simplicity of the univariate SPOT algorithm [169] and design an anomaly detection IDS (Netspot) based on Extreme Value Theory (EVT). The key variable in SPOT is the probability g of an abnormal pattern, which is computed as $g = P(Y > u_g)$, where Y denotes the sample data and u_g is the threshold. The performance evaluations in Siffer, Fouque, Termier, *et al.* [164] indicate that Netspot processes packets faster than Suricata and Kitsune [170]. A major limitation of Netspot is that the underlying SPOT algorithm cannot process multiple variables.

However, the applications in today's world have complex queries with many relationships. Therefore, univariate methods are being replaced by multivariate methods to accommodate complex application transactions. Ye, Emran, Chen, *et al.* [157] incorporated the statistical Hotelling T^2 method, is a multivariate approach, to flag abnormalities. With X_i denoting a vector of observations of multiple relevant system or process metrics at a given time instant i, $\overline{X}$ denoting the vector of the corresponding sample means, and S^{-1} the inverse of the sample variance, Hotelling T^2 is evaluated as [157], [171]:

$$T^2 = (X_i - \overline{X})' S^{-1} (X_i - \overline{X}), . \tag{4.1}$$

whereby the apostrophe $'$ denotes the vector transpose. Large T^2 values indicate substantial deviations and, thus, possible anomalies. The specific architecture for the anomaly detection based on the Hotelling T^2 statistic in Ye, Emran, Chen, *et al.* [157] applies only to host-based intrusion detection systems; therefore, the specific architecture of Ye, Emran, Chen, *et al.* [157] is not applicable for UTM systems. However, the general methodology of detecting anomalies via the Hotelling T^2 statistic can be employed in future UTM systems and we therefore include the Hotelling T^2 statistic in this UTM survey.

Knowledge-based IDS

Knowledge-based IDS creates standard traffic profiles to record legitimate behavior, and profiles are created based on human knowledge in the form of rules [156]. The process can be tedious as it requires user intervention, but has few false positives. Knowledge-based intrusion systems are ideal for static networks, where applications/systems are strictly controlled, and every process is recognized. Military, regulatory, and air-gapped networks are prime candidates for knowledge-driven IDS as data is tightly controlled and persistent in such networks. Implementing knowledge-based IDS in dynamic enterprise computing environments is difficult because of constant changes. Khraisat, Gondal, Vamplew, *et al.* [156] noted Finite State Machine (FSM) models as computational model that can be utilized in knowledge-based IDS to identify deviations (attacks) from normal patterns. In a general conformance testing context, EI-Fakih, Yevtushenko, and Bochmann [172] described FSM as an administered graph that consists of prescribed fields and associated functions that are triggered in case of an event. For general cybersecurity analysis, Qi, Zhong, Jiang, *et al.* [173] proposed a model to analyze advanced attacks, and generated a cybersecurity knowledge graph using FSM. FSM state changes were observed at each attack and upon successful execution of conditions. The model of Qi, Zhong, Jiang, *et al.* [173] lacked accuracy as the algorithm did not have any assessment mechanism. Against this backdrop of computational and knowledge modeling, several knowledge-based IDS models that are suitable for UTM systems have been proposed, as summarized in Table 4.5. We discuss representatives of these approaches in the following; the other approaches in Table 4.5 are variations of the discussed approaches with the specific characteristics given in Table 4.5.

Attacks can be detected by ingesting threat feeds. Open-source intelligence (OSINT) collects data that is freely and publicly available. The data can be useful for adversaries or ethical hackers to profile a target without actively engaging the target. Security experts can utilize OSINT to strengthen company defensive and perimeter protection. Vacas,

Medeiros, and Neves [174] proposed a knowledge-based detection mechanism based on OSINT. The IDS system detected botnet C&C communication, phishing, and remote attacks by ingesting 49 OSINT feeds. The IDS system of Vacas, Medeiros, and Neves [174] was built on three main components:

i) *Information Gathering:* The OSINT collector ingests feeds from a diverse source of OSINT feeds. Feeds can be categorized, e.g., as phishing, malware, or C&C. Multiple collectors can also be installed to ingest feeds based on various categories. The information can come in different formats, e.g., HTML, CSV, PDF, and TXT. The *collector parser* harmonizes various events into an indicator of compromise (IoC). The collector is also responsible for deduplication, and multiple sources can have similar events. Therefore, it is critical to clear any duplicate records.

ii) *Knowledge Generation:* In this phase, events are generated, and the IoC is blacklisted. The administrator can write scripts to automatically tag the IoC with relevant information, such as geolocation, domain registrar, Autonomous System Number (ASN), and malware information. Tags can be useful for security operation center (SOC) investigations and reporting purposes. IDS rules similar to Snort/Suricata are created to block the IoC.

iii) *Incident Detection:* This is the validation phase that ensures that no duplicate IDS rules are created by conducting optimization and performance checks. The checks are performed at periodic intervals to ensure IDS integrity.

Table 4.5: Comparison of knowledge-based IDS models.

Study, Year	Algorithm / Method	Key Features
Bouzar-Benlabiod, Meziani, Chebieb, *et al.* [175], 2016	Expert knowledge	Merges knowledge from industry experts to fine-tune IDS alerts
Vacas, Medeiros, and Neves [174], 2018	OSINT	Detection based on open-source threat intel feeds
Olimpio, Silva, Camargos, *et al.* [176], 2021	Stream mining alg.	Classifies data w/o capturing all packet headers
Li, Wang, Liu, *et al.* [177], 2021	FDEn: Feature Deriv. (FD) & Ensemble models (En_{pk})	Large number of features derived; improvements up to 62%
Arikan and Acar [178], 2021	Data mining techn.	Creates cyber threat intellig. based on stored or live traffic

Machine Learning-Based IDS

Machine learning (ML) is a technique of applying a set of rules, transfer functions, and complex algorithms to a large dataset to analyze and predict behaviors [179]. Sophisticated attacks can be predicted by integrating ML into IDS; thereby, unknown patterns can be marked as an anomaly. Our survey focuses on ML-based IDS that are suitable for UTM systems from the last twelve years, see Table 4.6. The main components of ML-based intrusion systems are as follows.

i) *Feature Selection Method:* Feature selection refers to selecting the features available within the dataset. Feature selection is mainly used to reduce training time, to simplify data for modeling, and to remove noise [180]. Wrapper-based, embedded-based, filter-based, and hybrid are the commonly used types of feature selection methods [181]. The wrapper-based method uses a supervised learning algorithm (SLA) for validation and generates feature subsets by using searching techniques [182]. The embedded-based method partially uses SLA for feature selection and has less computational cost than the wrapper-

based method [183]. The filter-based method does not rely on SLA, and selection depends on various statistical tests and their correlation [181]. Therefore, the filter-based method is suitable for high-dimension data due to less overhead. Hybrid is the combination of the wrapper and filter-based feature selection [184].

ii) *Feature Extraction Method:* Feature extraction processes raw data in such a way that key features are discovered, and the features will later be used by algorithms to detect behavior patterns. There are two types of feature extraction methods, namely linear and non-linear. Both types are deployed to reduce the dimensionality of the data in the extraction method. Linear methods are suitable for initial tasks, such as pre-processors; while non-linear methods accommodate data that require complex processing [180]. Zhang and Chen [185] observed that higher feature extraction performance can be achieved by using standard deviation and variance (linear functions).

iii) *Classifier:* A classifier is an algorithm that categorizes data into labels or buckets. Exemplary classification labels are DDoS, high data usage, and high domain conversation. Classifiers are first trained with labeled data. Then, after adequate learning, the classifier can create labels based on the learned behavior patterns. Support vector machine (SVM), K-nearest neighbor (KNN), and decision tree are commonly used classifiers [186].

iv) *Datasets:* Datasets have been curated to benchmark intrusions detection systems. Notable datasets are KDD99 [187] and NSL-KDD (improved version of KDD99) [188].

The IDS ML comparison in Table 4.6 reveals that SVM is a frequently used classifier. Table 4.6 also indicates that an accuracy rate of $> 90\%$ for both the KDD99 and NSL-KDD datasets can be achieved with SVM based IDS models. Additional studies found that linear SVM is generally faster than non-linear SVM in processing training data [189], which makes linear SVM a promising choice for UTM systems.

Generally, ML-based IDS implementations can be supervised, unsupervised, semi-supervised, and self-supervised.

Study	**Year**	**Dataset**	**Feature Selection/Extraction method**	**Classifier**	**Accuracy**
Zhang, Zhang, and Sun [190]	2009	KDD99	N/A	N/A	greater than 90%
Al-Janabi and Saeed [191]	2011	KDD99	N/A	Artif. Neural Netw. (ANN)	Varies
Damopoulos, Menesidou, Kambourakis, *et al.* [192]	2012	iPhone user data log	N/A	Random	99.8%
Alomari and Othman [193]	2012	KDD99	Bees algorithm (BA)	SVM	82%∼96%
Al Mehedi Hasan, Nasser, and Pal [194]	2013	KDD99	N/A	SVM	∼99%
Kasliwal, Bhatia, Saini, *et al.* [195]	2014	KDD99	G-LDA	N/A	88.5%
Zhang, Xu, and Gong [196]	2015	KDD99	One-class SVM	SVM	99.0%

Aburomman & Reaz [197]	2016	KDD99	Linear Discriminant Analysis(LDA) and Principle Component Analysis (PCA)	PCA-LDA-SVM	92.2%
Jaiswal, Manjunatha, Madhu, *et al.* [198]	2016	NSL-KDD, ISCX	N/A	Random Tree, Naïve Bayes	77% ∼99.8%
Thaseen and Kumar [199]	2017	NSL-KDD	Chi-Square	Multiclass SVM	98%
Chang, Li, and Yang [200]	2017	KDD 99	Random forest	SVM	88.2 ∼93%
Mighan and Kahani [201]	2018	UNB ISCX 2012	Hybrid - Self Proposed	DL-SVM	>90%
Shenfield, Day, and Ayesh [202]	2018	Custom Dataset	N/A	Artif. Neural Netw. (ANN)	98%
Belouch, El Hadaj, and Idhammad [203]	2018	UNSW-NB15	N/A	Random	97.5%

Mohammadi, Mirvaziri, Ghazizadeh-Ahsaee, *et al.* [204]	2019	KDD99	Feature grouping based on lin. corr. coeff. (FGLCC) and cuttlefish alg. (CFA)	Decision tree	95.0%
Maniriho, Niyigaba, Bizimana, *et al.* [205]	2020	IoTID20	HFS-Engine	Random	∼99%
Kabir and Luo [206]	2020	KDD & ISCX	N/A	K-Means, SOM, DAGMM, ALAD	99 ∼99.8
Chen, Yin, Cai, *et al.* [207]	2021	Honeybrid honeyport system	L-KPCA	SVM	93.6%.
Amaran and Mohan [208]	2021	KDDCup 99	Deep belief networks (DBN) & Whale optimization algorithm.	Optimal SVM	94.1%
Tarek, Mazumder, Sharmin, *et al.* [209]	2022	NSL-KDD	Robust Mahalanobis Distance (RMD)	N/A	99.1%

Chen, Xu, Wang, *et al.* [210]	2022	NSL-KDD	Convolutional Neural Network (CNN)	SVM	94.5%
Fatani, Dahou, Al-Qaness, *et al.* [211]	2022	KDD99	Convolutional Neural Network (CNN)	KNN	99.9%
Cholakoska, Gjoreski, Rakovic, *et al.* [212]	2023	IoTID20	Federated Learning (FL)	N/A	∼84%
Elnakib, Shaaban, Mahmoud, *et al.* [213]	2023	CICIDS2017	Deep learning multi-class classification	Customized	95%
Hidayat, Ali, and Arshad [214]	2023	TON_IoT	Pearson correlation coefficient	Hybrid	99 ∼99.2%
Mohy-eddine, Guezzaz, Benkirane, *et al.* [215]	2023	Bot-IoT	Principal component analysis (PCA), Univariate stat. test, genetic alg. (GA)	KNN	99.99%
Rani, Gill, Gulia, *et al.* [216]	2023	DS2OS	Exclusive Feature Bundling (EFB)	Logistic regr., Random Forest, Gradient boosting	>99%

Thakkar & Lohiya [217], [218]	2023	NSL-KDD, UNSW_NB-15, CIC-IDS-2017, BoT-IoT	Statistical importance variables	Deep neural network (DNN)	75.3∼99.0%
Wang, Yang, and Weng [219]	2023	ToN_IoT	Feature Tokenizer	NA	95.8∼98.0%

Table 4.6: Comparison of ML-based IDS models.

Supervised Learning ML algorithms can be applied to datasets, such as KDD99 and NSL-KDD. The data can be collected via sniffers or NetFlow in a production environment. Supervised learning techniques either involve regression or classification, relying on previous data to predict anomalies. Anomalies are categorized as outliers, and some prerequisites have to be met to identify imbalanced data. First, the data baselining has to be determined to calculate normal traffic patterns. Secondly, before applying supervised learning, data augmentation procedures should be utilized to reduce false positives. Augmentation facilitates consideration of all possibilities. For example, a system can be accessed both by an IP address and a fully qualified domain name. Therefore, it is essential to include both values in the classification. Augmentation increases the size of the training data. A commonly used augmentation classifier is KNN, which can group the data based on commonalities.

Supervised learning can be applied to the augmented dataset. A classification method is commonly used instead of regression to detect anomalies. Since anomalies are imbalanced traffic behaviors, regression techniques, such as linear regression or polynomial regression, are not beneficial in identifying outliers. Many ISS models utilize Support Vec-

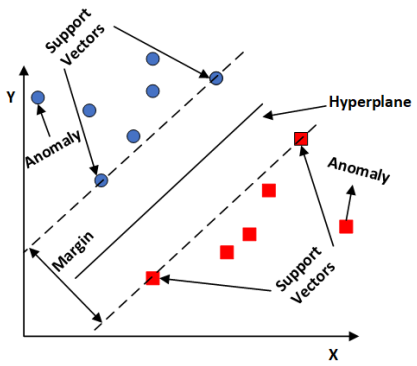


Figure 4.18: Illustration of support vector machine (SVM) hyperplane in supervised learning: The points furthest from the hyperplane are considered anomalies.

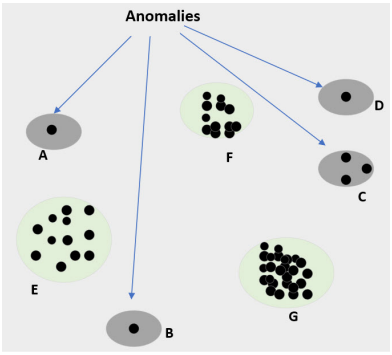


Figure 4.19: Illustration of clustering in unsupervised learning: Individual outliers or clusters of outliers, which are shaded in grey, are identified as anomalies.

tor Machines (SVMs), which use hyperplanes for segmentation, as illustrated in Fig. 4.18. The points closest to the hyperplane are called support vectors; whereas, the points furthest from the hyperplane are considered anomalies. The IDS model of Amaran and Mohan [208] tunes data with the whale optimization algorithm [220] before applying SVM, and the resulting system was able to achieve 94.1% accuracy. Another approach for detecting intrusions is to use separate algorithms for feature selection and for classification. Thereby, an SVM for feature classification can be combined with random forest for selection to improve precision [200]. IDS detection and performance capabilities can further be improved when the SVM is combined with other data-tuning algorithms.

Unsupervised Learning Unsupervised learning can retrieve information from datasets without the need for class labels. Unlike supervised learning, unsupervised learning does not require training data, and input points are treated as random values. The learning process allows the grouping of data to find extreme values. One such technique is clustering analysis; data are clustered together based on observed common learning patterns. Fig. 4.19 illustrates clustering of data patterns. Clusters with greater density, i.e., clusters E, F, and G, are treated as normal traffic. Anomalies can be identified by clustering together outliers, i.e., clusters A, B, C, and D. K-Means is a commonly used clustering method [221]; whereby, n data objects are grouped into k clusters. A density-based outlier detection model can also be used to categorize clusters based on cell density [222]. Zhang, Zhang, and Sun [190] proposed a hybrid intrusion detection system with characteristics of both signature and anomaly-based systems, a modified version of the K-means algorithm has improved accuracy. Kabir and Luo [206] demonstrated that K-means has a higher detection rate than other unsupervised learning algorithms, such as Self Organizing Maps (SOMs) [223] and deep autoencoding Gaussian mixture models (DAGMMs) [224].

Semi-Supervised Semi-supervised learning is designed to bridge the gap between supervised learning and unsupervised learning. Supervised learning can be time-consuming due to manually labeling training data, while unsupervised leaning is limited to clustering methods. Accuracy is also a concern for discontiguous datasets. Semi-supervised learning can be applied to a dataset that is a mixture of labeled and unlabeled data points. Thereby, the labeled data is utilized to create classifiers, which are used to predict labels for the unlabeled data. The resulting output is called *pseudo-labels*. The *pseudo-labels* are added according to a confidence level, and the two datasets (i.e., initially labeled data and *pseudo-labels*) are trained on again to improve the classifiers.

The IDS model of Jaiswal, Manjunatha, Madhu, *et al.* [198] is built upon semi-supervised learning. In particular, an intermediary model is established from the labeled

data, and the model is then integrated into the data mining tool Pentaho and the WeKa plugin [225]. New classifiers (normal or anomaly) ware predicted for unlabeled data after being processed by Pentaho and WekaScoring. Thereby, IDS efficiency and detection capabilities can be improved by classifying previously unknown data.

Self-Supervised Learning Labeling data can be tedious and may require a specialized skill set. For example, medical or scientific data cannot be easily labeled without human intervention. To overcome this labelling challenge, so-called self-supervised learning forms models without processing data labels. Supervised learning requires data labeling, and unsupervised learning uses algorithms to create labels for unlabeled data; whereas, self-supervised learning relies on supervisory signals from a dataset to predict significant features about the dataset In many analysis contexts, there are considerable amounts of unlabeled data from which self-supervised learning can learn, and, instead of labeling, self-supervised learning can simply forecast the hidden values. One implementation of self-supervised learning is Natural Language Processing (NLP). Email and document processing programs can apply NLP to automatically populate missing text from sentences. Analogously, IDS can utilize self-supervised learning for feature extraction and optimization, whereby features that tend to contribute little towards classification can be removed [226]. Threat feeds and rules can be classified based on confidence and severity. Administrators can only be alerted for intrusions with critical severity and high confidence levels.

Data Mining & Flow Data Based

Data mining is another technique to detect anomalies when dealing with large data volumes. Data mining can be applied to live or stored data to extract threat vectors and to return IoCs in a standard format. Bouzar-Benlabiod, Meziani, Chebieb, *et al.* [175] used data mining to identify standard classifiers and convert them into the Structured Threat

Information Expression (STIX) format. The classification of threats into STIX has the additional benefit of compatibility. Researchers have formulated STIX as the standard format for threat intelligence feeds [227]. Data mining has benefits over machine learning algorithms in terms of complexity, performance, and learning times. Subaira and Anitha [228] found that IDS based on Support Vector Machines (SVMs) and neural networks require higher computational power, more memory, and longer baselining (learning) time than IDS based on data mining. Data mining effectively extracts information from an existing dataset and requires human intervention; whereas, machine learning can predict future attack patterns. Another approach to reduce false positives and to improve accuracy is to ingest the knowledge of industry experts. More specifically, the syntactic approach can merge knowledge from a diverse set of experts and find common sub-bases. Thereby, duplicates can be eliminated and harmonized for effective IDS alert filtration [175]. Data mining techniques can also reduce the overhead of IDS processing and the number of log messages (i.e., tune the logging), and thus improve the UTM system performance by reducing the computational burden due to the IDS processing and logging. Foundational mining techniques, such as Apriori [229], can be deployed to achieve these UTM efficiency enhancements.

Specifically, association rule mining techniques, such as the Apriori algorithm [229], can be integrated into UTM systems for correlation and feature extraction. Generally, association rule mining techniques analyze correlations between datasets in large databases. In particular, the Apriori algorithm [230] is a data mining algorithm based on level-wise and breadth-first search methods [229], [231], [232]. The Apriori algorithm automatically generates association rules by determining the commonalities and frequencies of sets of items in a database. The integration of the Apriori algorithm into a UTM intrusion detection engine improves the inspection of correlated data so that anomalies can be better detected. Also, intrusions rules can be filtered with the Apriori algorithm. In addition, the detection capabilities can be further improved with neuro-fuzzy [233], neural networks

[234], and semi-supervised models [235], which should be further explored for UTM systems in future research.

An alternative method for detecting anomalies involves recognizing variations in traffic patterns or flows and flagging any deviations as anomalies. A common method of gathering flow data [236] is using the Netflow protocol [237]. Netflow is a protocol used by network devices to collect traffic statistics, application details, and routing information. Routers, switches, firewalls, and cloud devices can be configured to forward flow information to collectors for analysis. As a result, Netflow gives a wealth of information. However, this solution requires enabling the Netflow protocol on network devices. To avoid the dependency of Netflow capabilities on third-party devices, the Open-source application NTOP [238] can perform similar functions without the need to configure network devices.

4.5.1 SUTMS Implementation Of Anomaly Detection Engine

Netflow functions as a protocol employed by enterprise network devices to accumulate telemetry data. Nonetheless, implementing the Netflow protocol on network devices is an essential step in this solution. To avoid the dependency of Netflow capabilities on third-party devices, the Open-source application NTOP can perform similar functions without the need to configure network devices. SUTMS takes advantage of the open source probing service "NTOP", it has a dual purpose. First, it is used to identify any abnormal traffic patterns, and second, the identified protocols are utilized to enable matching Suricata signatures. NTOP [238] is a web-based open-source tool for monitoring and measuring network traffic. The key characteristics of NTOP include:

i) Compatible with Linux, Unix, and Win32 platforms.

ii) Efficient Kernel with easy to implement on low resource devices without impacting CPU and memory.

iii) Can be managed remotely using a web interface.

iv) Gives a visual representation of data in tables, charts, etc.

v) Identifies applications, protocols, and network flows.

The features listed above make the NTOP an excellent flow detection service similar to the Netflow protocol without relying on network devices. UTMs can utilize NTOP flow detection capabilities by integrating the application data into their inspection engine. Integration of NTOP takes UTM inspection capabilities to the next level, including advanced persistence threat detection. Attacks that are stealthy and happen over time are difficult to detect. APT relies on zero-day exploits (publicly unknown vulnerabilities) and uses sophisticated means of payload delivery [239]. APTs bypass IDS and firewall technologies as those exploits do not have signatures. One of the approaches to detect APTs is to analyze normal behavior and look for abnormalities (anomalies). *Anomalies* are activities that can potentially be suspicious [240].

Application Detection Engine

NTOP can be configured to sniff flows on the internal and external interfaces, *eth0* is programmed for capturing internal traffic and *wlan0* give flows for internet-bound traffic. The application discovered in our test environment is listed in Fig. 4.20. NTOP and SSH represent management traffic to SUTMS therefore it can be ignored, HTTP constitutes about 36.1% of overall traffic. Traffic identified in Fig. 4.20 allows us to only enable IDS signatures for the HTTP protocol, this greatly reduces the overhead on the overall UTM resources.

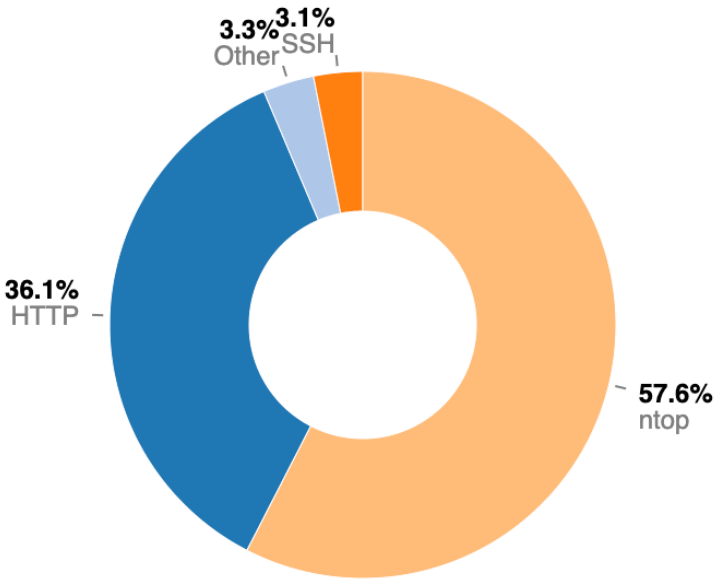


Figure 4.20: Home applications discovered via NTOP.

Alerts on anomalies

NTOP application detection capabilities can not only be used for tuning intrusion signatures, it can also be crucial in identifying anomalies. Table 4.7[241] enlists default security alerts that can be triggered based on any anomalies observed. Alert key 23,24, and 25 are able to inspect encrypted traffic without decrypting it. TLS certificate mismatch, legacy versions, and insecure cipher can be an indication of vulnerable applications and possible anomalies. Alert 6 and 64 deals with payload abnormalities of DNS packet, hackers tend to use DNS for data exfiltration and command and control C&C communication. Custom alerts can also be created based on the flows observed by SUTMS, Fig. 4.21 gives the visual representation of flows, alerts can be generated for any new flows not seen in the last 30 days.

Table 4.7: NTOP Built-in Security alerts [241]

Alert Key	Alert Key String	Alert Name
23	alert_tls_certificate_mismatch	TLS Certificate Mismatch
24	alert_tls_old_protocol_version	Obsolete TLS Version
25	alert_tls_unsafe_ciphers	Unsafe TLS Ciphers
6	alert_dns_data_exfiltration	DNS Data Exfiltration
64	ndpi_dns_large_packet	DNS Packet Larger Than 512 bytes
58	alert_lateral_movement	Lateral Movement
62	ndpi_clear_text_credentials	Clear-Text Credentials

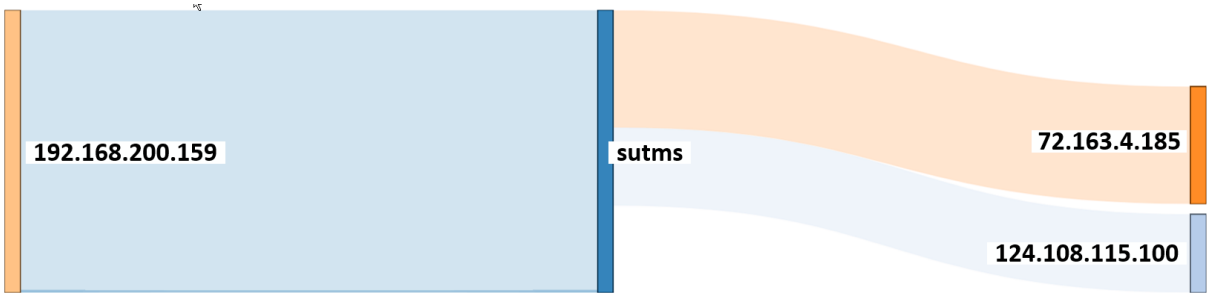


Figure 4.21: Traffic pattern from SUTMS.

4.6 Logging Engine

Logs are crucial for troubleshooting, security analysis, and forensics investigation. Logging requires additional storage and processing power; therefore, SUTMS is designed to process logs in real time only. The product fully supports Syslog and logstash; IDS/IPS, Firewall, and NTOP alerts/logs can be configured to send to external sources like SIEM and Elasticsearch. Fig. 4.39 illustrates the flow of information and log generation, and logging is an optional component of SUTMS. It is recommended to integrate logging with data mining algorithms like Apriori [230] to further reduce false positives.

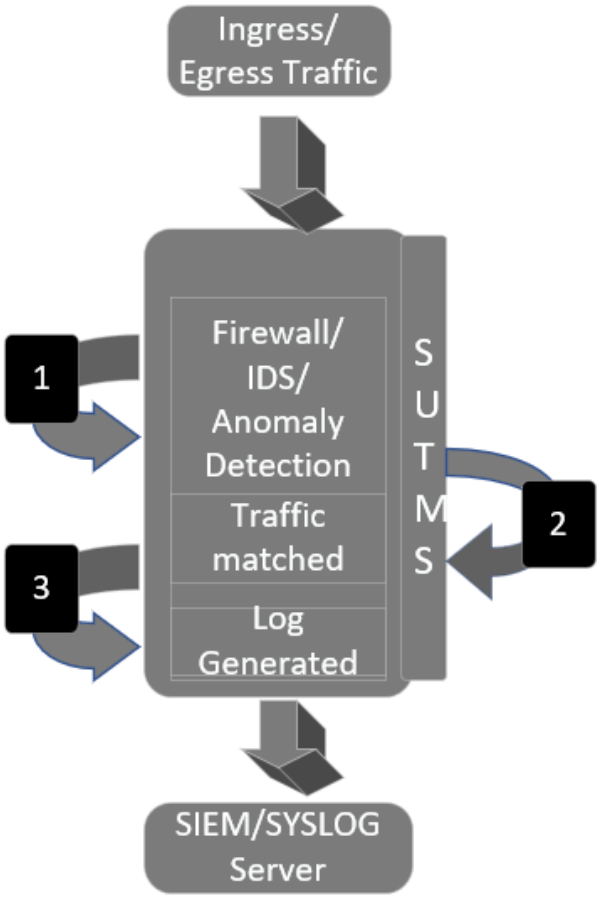


Figure 4.22: SUTMS log analysis engine.

4.7 SUTMS Testing and Evaluation

In this section, SUTMS accuracy and efficiency is evaluated by using standard datasets. Impact on performance is tested with and without tuning. The tests can ensure detection and inspection capabilities under high traffic volume. The tests are performed by considering the real-world home and SOHO networks, and a special-purpose network is provisioned to simulate the actual production environment.

4.7.1 Test Network Overview

The network of various computing devices, including smartphones, IoT, laptops, and printers, are deployed for SUTMS evaluation. Test network topology is illustrated in Fig. 4.23. SUTMS is installed in in-line mode and acts as a central inspection point before the traffic exits to the internet router. SUTMS has access to the internet for signature updates, cloud integration, STIX/TAXII feeds, and remote security management. Any traffic destined for the internet, regardless of device type, is inspected. The network provides both WiFi and ethernet access. SUTMS can be configured via SSH version 2 or directly connecting to a console (keyboard, mouse, and monitor). Test machine or SUTMS itself can be used to run the dataset, and the output of the results are exported to elastic search [242]. Kibana [243] for analyses and graphical representation.

HTTP and HTTPS ports are allowed for outbound web browsing traffic, and DNS traffic (UDP/53) destined for trusted DNS servers is also allowed. Application management ports (TCP/3000 *NTOP, TCP/10000 *Webmin, TCP/22 *SSH) are allowed from the internal network i.e., 192.168.200.0/24. Traffic required for testing SUTMS engines are also allowed. Traffic that doesn't match the rule base is blocked and logged for analysis

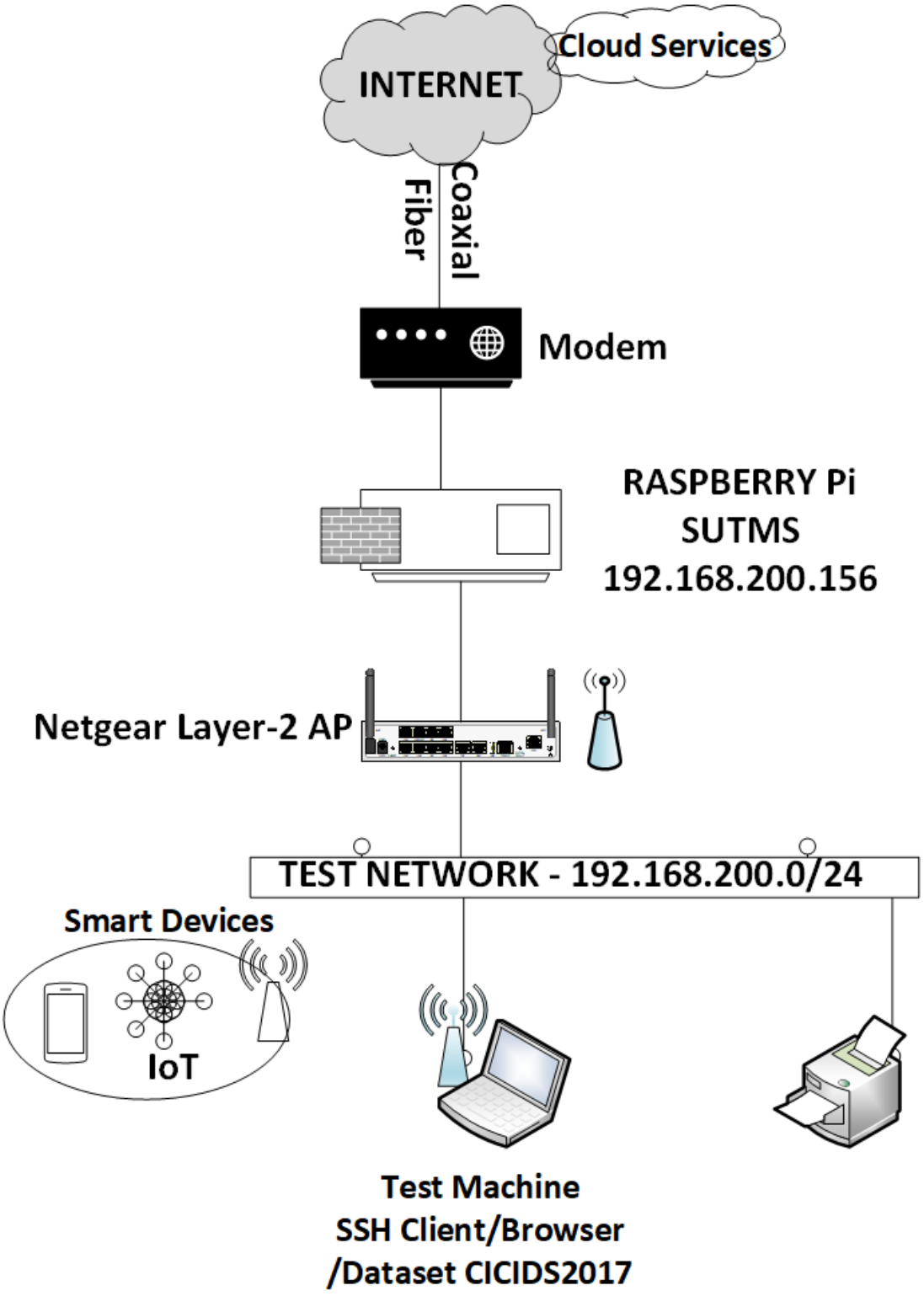


Figure 4.23: SUTMS Evaluation Test Network.

4.7.2 Dataset

Specialized datasets are required for evaluating UTM solutions. In our evaluation criteria, we selected CICIDS2017 dataset [244]. The dataset is used for evaluating IDS accuracy and performance. The 8.2 GB dataset in pcap format includes common and relatively newer attacks, and it was last updated on 2019-09-10. Tcpreplay [245] was used to run the capture on a test machine, and traffic was intercepted by SUTMS for inspection. The reasoning behind selecting CICIDS2017 dataset are : -

i) It is built up unique profiling mechanism, i.e., B-Profile system [246], it allows simulating of human behavioral traffic patterns and benign attacks.
ii) Generates data of 25 users and commonly used protocols, i.e., HTTP, HTTPS, SSH, FTP, and email [244]. The number of users and protocols closely match our home network UTM solution design.
iv) Includes common home network attacks, i.e., DDoS, Brute Force protocols, HTTP/HTTPS exploitation, C&C communication, and DDoS.
iii) Data in pcap format is easier to manage and simulate actual user traffic.
iv) The dataset is designed for IDS evaluation. However, the size and quality of the dataset are significant enough for stress testing and inspection capabilities of SUTMS IDS with other modules enabled, i.e., firewall and flow detection.

4.7.3 Intrusion Engine Evaluation

SUTMS IDS engine is built upon Suricata, and it is evaluated against CICIDS2017. The tests have two phases; in phase I, signature detection and system resources are evaluated with default configurations. Results from the NTOP engine is integrated into IDS, and default rule set is modified according to protocol detection in phase II. SUTMS is eval-

uated against the modified set of configurations, and core services (firewall and NTOP) are enabled as part of UTM evaluation criteria. Test results are recorded for individual component testing. However, all the UTM services should be running to ensure device integrity. It won't be beneficial to test IDS without the firewall and flow detection engine running because, in a UTM device, all the components are interdependent regarding resource allocation. Cockpit [247] and Netdata [248] are utilized for monitoring CPU, ram, system load, and disk I/O.

Threat Detection - Phase I

SUMTS is able to detect approximately 404,776 events Fig 4.24 within minutes of dataset execution, it is also able to detect anomalies apart from signatures. IDS events have consisted of flows, protocols, and signatures. Fig. 4.25 represents the events detected during our tests. DNS and flow are the two dominant categories of events, constituting 38.79% & 36.14%, respectively. Security-related events are categorized as "alerts", Fig. 4.26 lists the number and type of signatures detected. IDS is successfully able to detect matching signatures and hits, "SURICATA TCPv4 invalid checksum" and "SURICATA STREAM CLOSEWAIT FIN out of the window" has the highest number of hits recorded. Invalid checksum and out-of-window packets can be an indication of a DDoS attack. DNS alerts are useful in investigating data exfiltration and C&C communication.

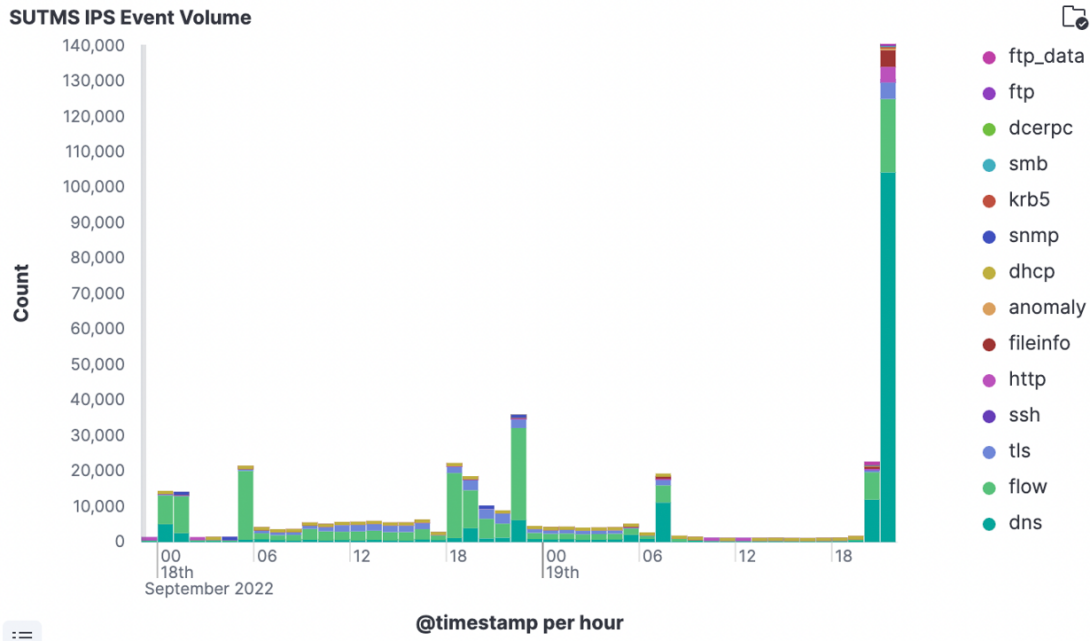


Figure 4.24: SUTMS Events - Phase I.

A higher number of events and alerts can waste systems resources and generate false positives. In phase 1, we tested system resources without ingesting data from the flow detection engine, i.e., NTOP.

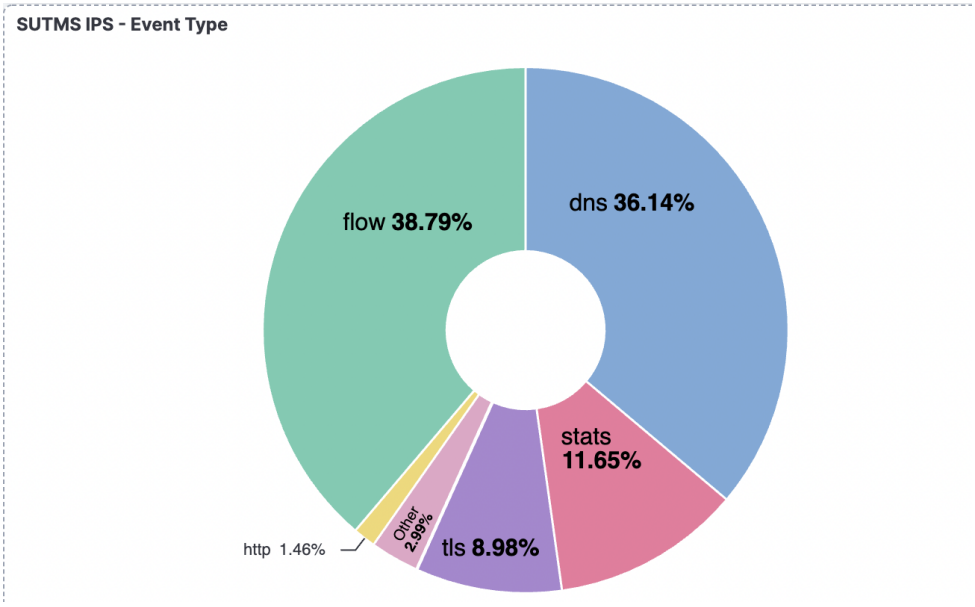


Figure 4.25: SUTMS Event Type Detected - Phase I.

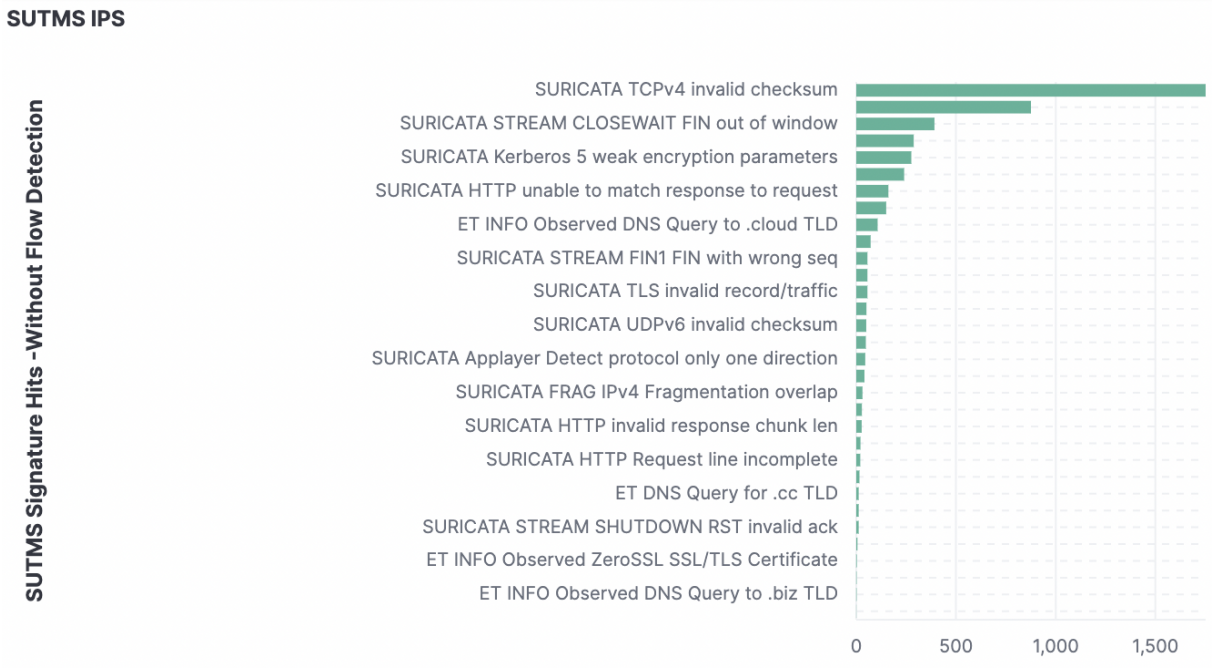


Figure 4.26: SUTMS Security Events - Phase I.

System Resources - Phase I

Performance testing is based on CPU, system load, memory, and disk inputs/outputs, while the dataset is running. During the IDS test window, all the critical services were enabled for inspecting and analyzing traffic to measure actual usage. CPU jumped to 25% from 3%-4% as dataset inspection started. Fig. 4.27 shows the CPU usage.

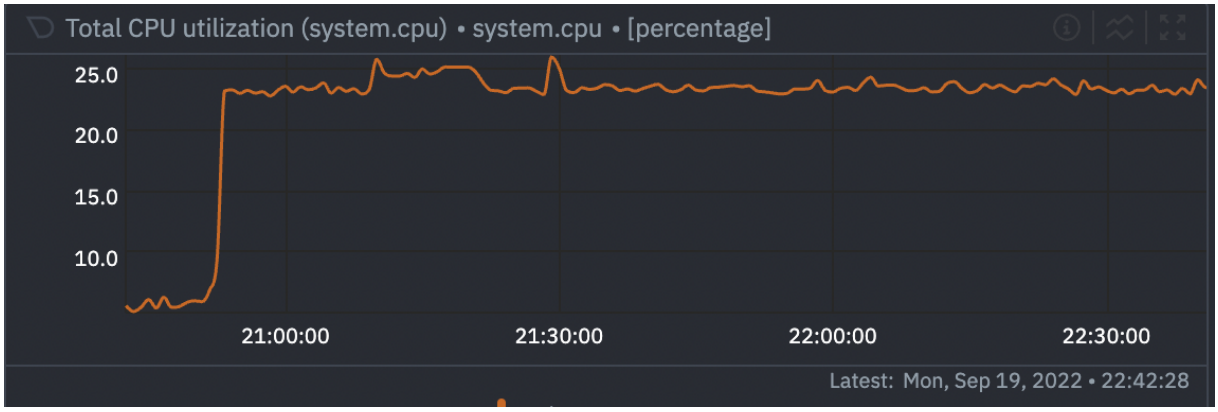


Figure 4.27: SUTMS CPU Usage - Phase I.

We noticed an initial memory spike to 2.34 GB and then stabilized below 2 GB

(Fig .4.28). System load defines the total number of processes a CPU runs at a given time, and it was observed that load stayed constant around 1.50 -2.52 and then dropped to below 1.50 (Fig. 4.29). Disk reads and write can be problematic, especially when dealing with systems like Raspberry Pi with non-volatile flash memory (SD-Card), disk i/o depicted in Fig. 4.30 reveals only fewer spikes, overall it stayed well below 0.1 MiB/S.

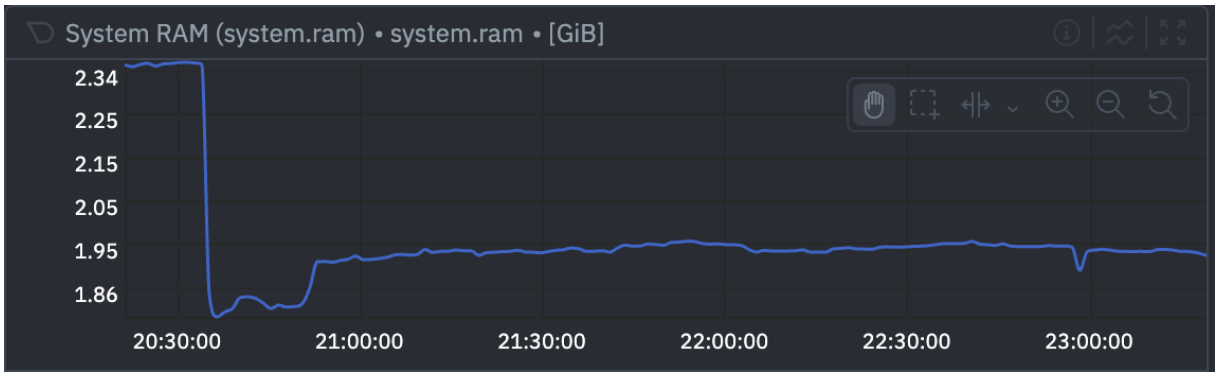


Figure 4.28: SUTMS Memory Usage - Phase I.

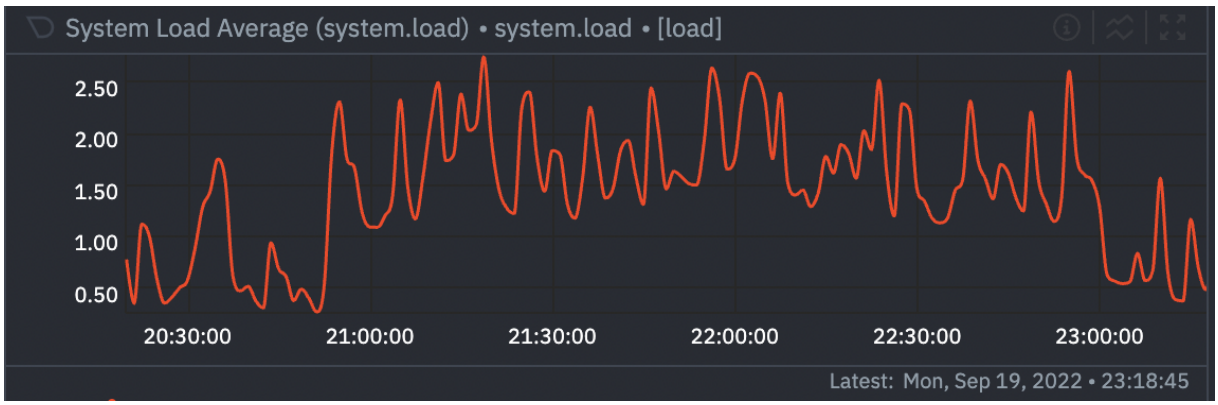


Figure 4.29: SUTMS System Load - Phase I.

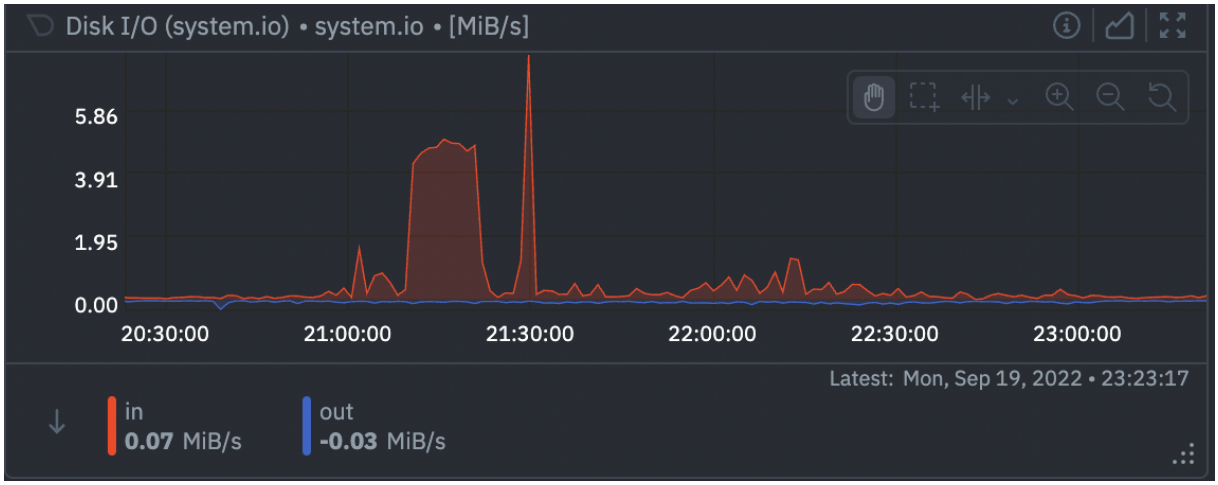


Figure 4.30: SUTMS Disk Input/Output - Phase I.

IDS Phase-I testing concluded that SUTMS was able to achieve 99.99% of accuracy in terms of event detection without causing a significant performance impact on system resources. SUTMS performed an IDS inspection with Max.CPU = 25% , Max.disk inputs = 5.9 MiB/S, Max.Load=2.52 and Max.RAM = 2.34 GB, overall Raspberry Pi performed within the system's specification in Phase-1 (without flow detection). In Phase II of IDS testing, SUTMS is fine-tuned to improve performance and efficiency and reduce false positives.

Threat Detection - Phase II

In Phase II of the evaluation, IPS signatures were tuned according to the traffic observed by the NTOP engine. Flow events were also eliminated as flows were collected using NTOP. The modification resulted in almost 50% reduction of events to 229,341 as depicted in Fig. 4.31. The notable events are DNS 59.63%, TLS 14.52%, and HTTP 2.44% as represented in Fig. 4.32. Among the top signatures detected in Fig. 4.33 corresponds to HTTP, TLS, and UDP (possible DNS) protocols, which coincide with the NTOP flow detection engine.

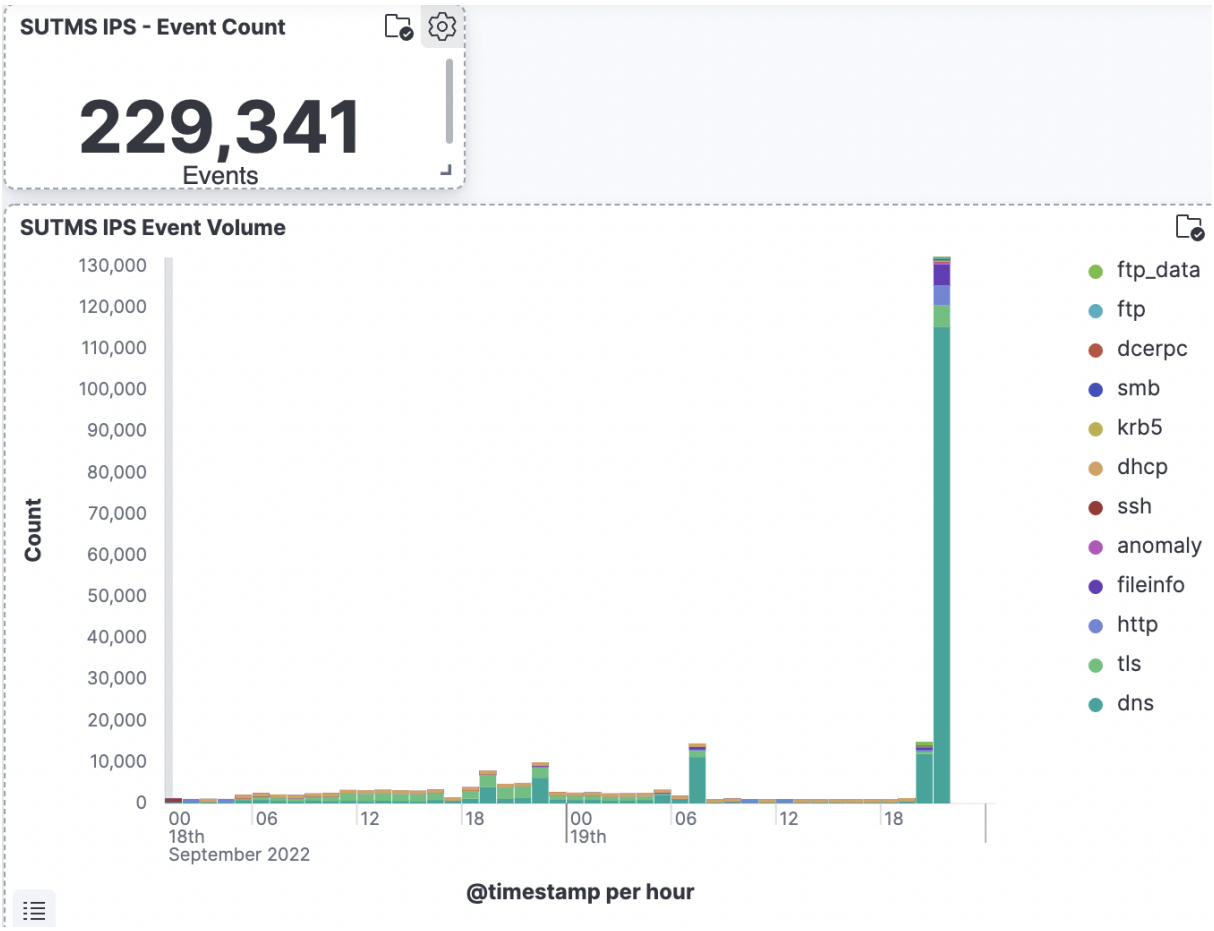


Figure 4.31: SUTMS Events - Phase II.

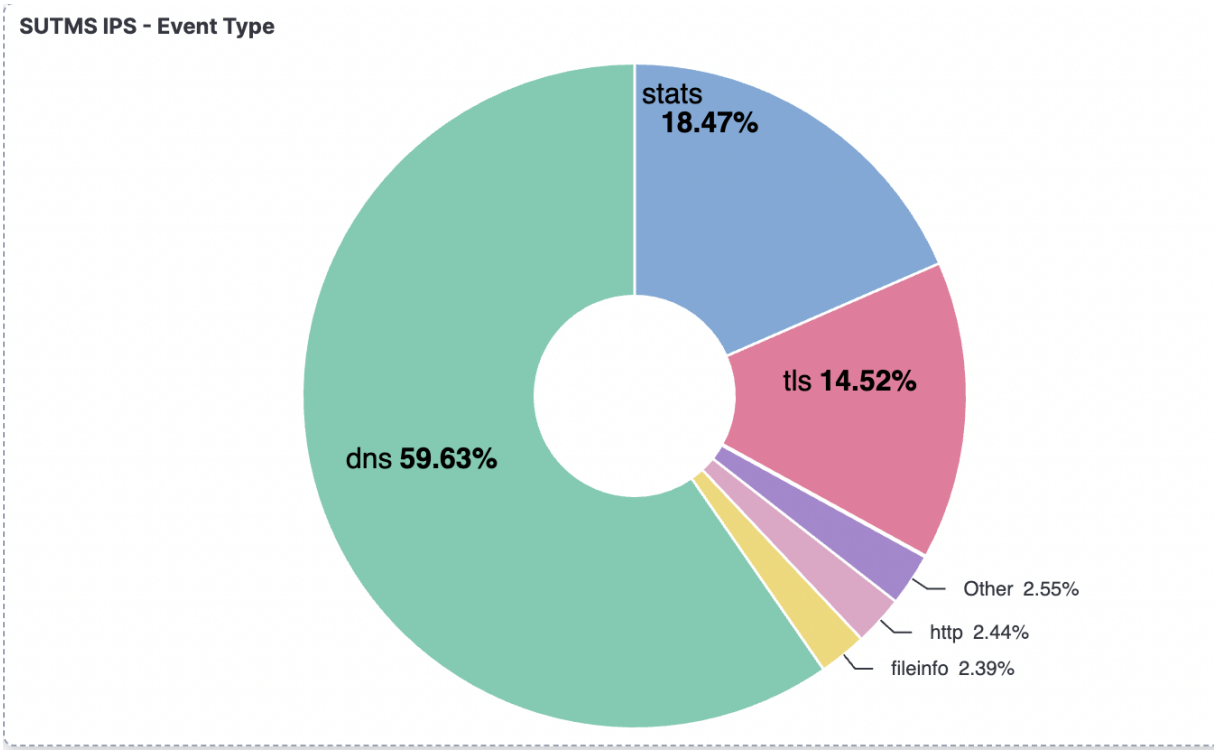


Figure 4.32: SUTMS Event Type Detected - Phase II.

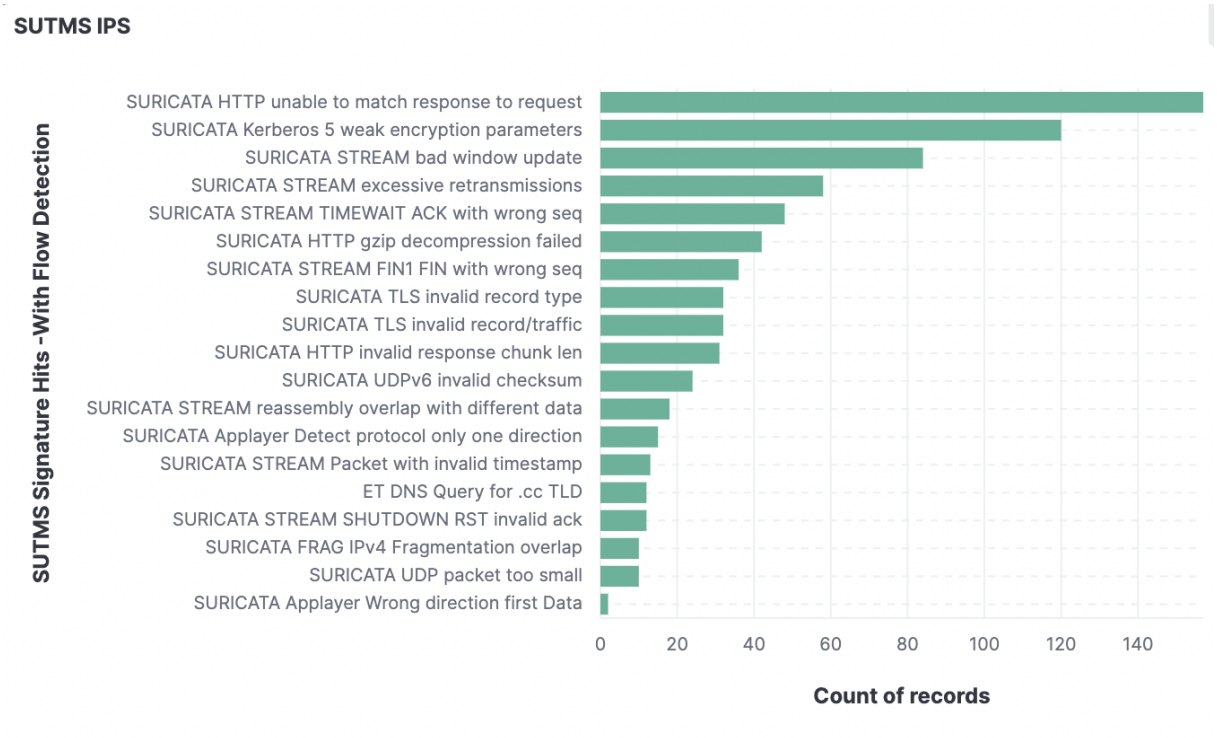


Figure 4.33: SUTMS Security Events - Phase II.

Integration of NTOP data into the IPS engine has enhanced detection capabilities by targeting common protocols and reducing false positives. Inspecting encrypted traffic is a challenge. Detection of Kerberos and TLS-based signatures like "SURICATA Kerberos 5 weak encryption parameters" and "SURICATA TLS invalid record type" respectively, is an indication of early detection of weak ciphers and TLS compliance failures before the encryption process starts. It could also highlight any outliers in the form of anomalies and standard deviation, newer ciphers, weak authentication, and possible C&C communication.

System Resources - Phase II

In Phase II of IPS testing, all the conditions were kept the same as in Phase I except for configuration optimization utilizing the NTOP flow detection engine. The system achieved performance improvements across the board, with the Maximum CPU = 22.50%, Maximum Memory= 1.01 GB, and Maximum Load= 1.50. CPU, Memory, Load, and Disk Input/Output usage during the test window are represented in Fig. 4.34, Fig. 4.35, Fig. 4.36 and Fig. 4.37 respectively. An initial spike and then constant usage of resources was observed, which is a similar trend noticed in Phase I. SUTMS Core firewall and flow detection services were also enabled along with IPS inspection.

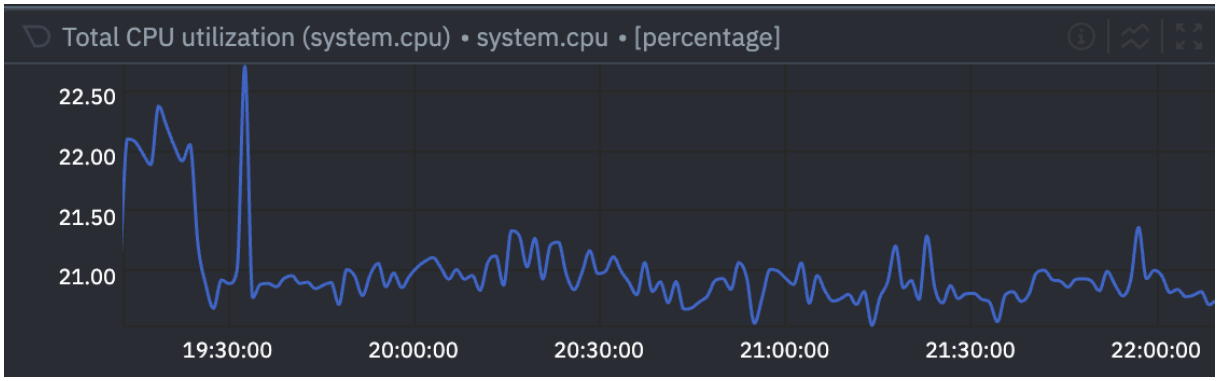


Figure 4.34: SUTMS CPU Usage - Phase II.

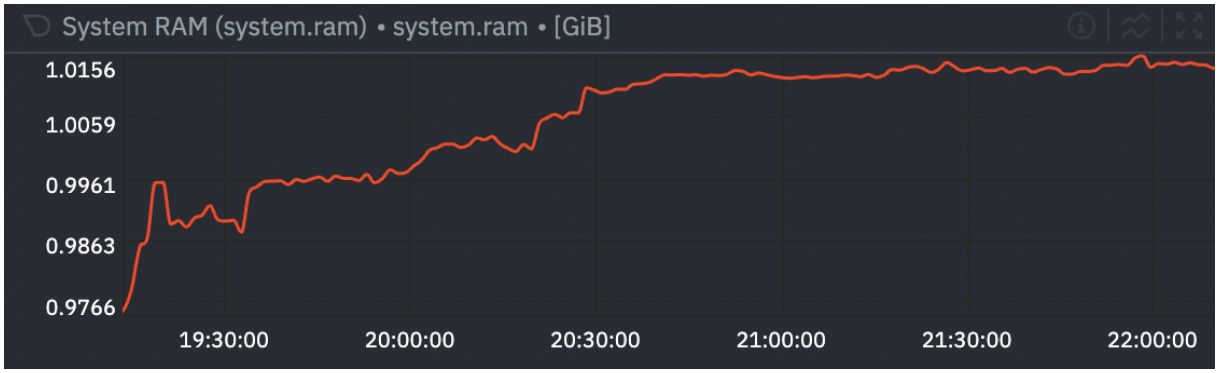


Figure 4.35: SUTMS Memory Usage - Phase II.

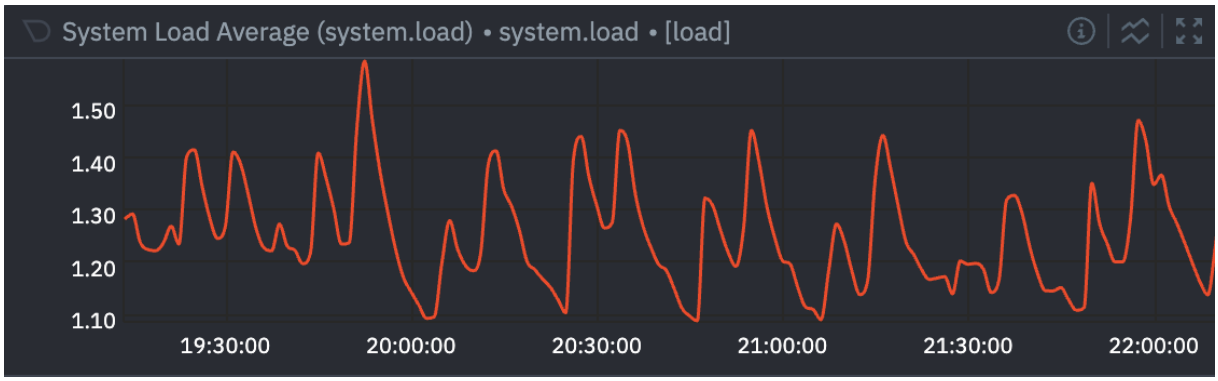


Figure 4.36: SUTMS System Load Usage - Phase II.

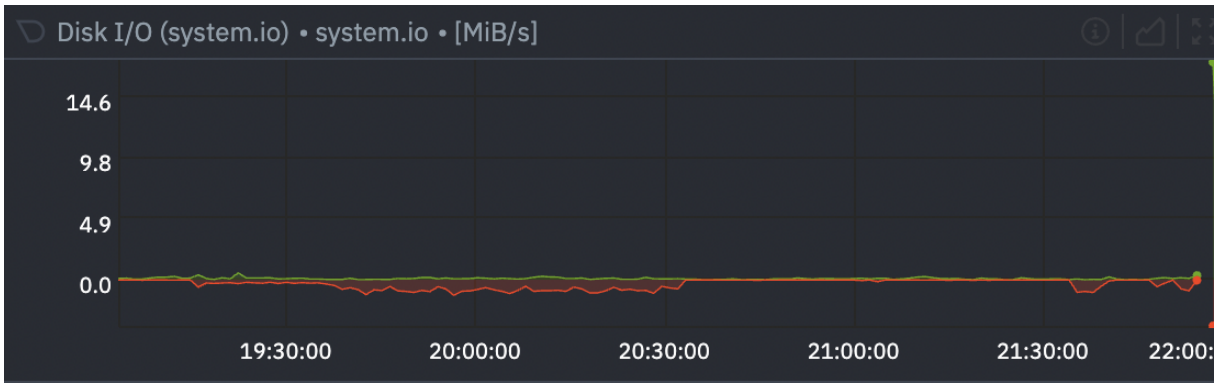


Figure 4.37: SUTMS Disk Input/Output - Phase II..

Comparison of Phase I and Phase II Test Results

Systems resources used by SUTMS IPS during Phase I and II are shown in Table. 4.8. Signature refinement (Phase II) can achieve improvement across the board. Memory and

System Load emerged as the biggest resource savers, i.e., 56.83% and 40% respectively Fig. 4.38 without compromising system accuracy. CPU efficiency can also be improved further by disabling low-priority signatures. Another significant impact of signature refinement is the reduction of security events; fewer events can highlight alerts of critical nature. In both phases, it is determined that it is ideal for sending the logs to third-party log collection engines like Syslog or SIEM. Raspberry Pi 4 with 32 GB is able to save logs for weeks, but eventually, it can run out of space. An automated script can also be created to remove logs older than a week or two, depending on the predefined data retention policy.

Table 4.8: Summary of Peak System resources utilized in Phase I & II

	CPU %	**Mem (GB)**	**Load**	**Accuracy**
IPS Phase I	25	2.34	2.5	99.99%
IPS Phase II	22.5	1.01	1.5	99.99%

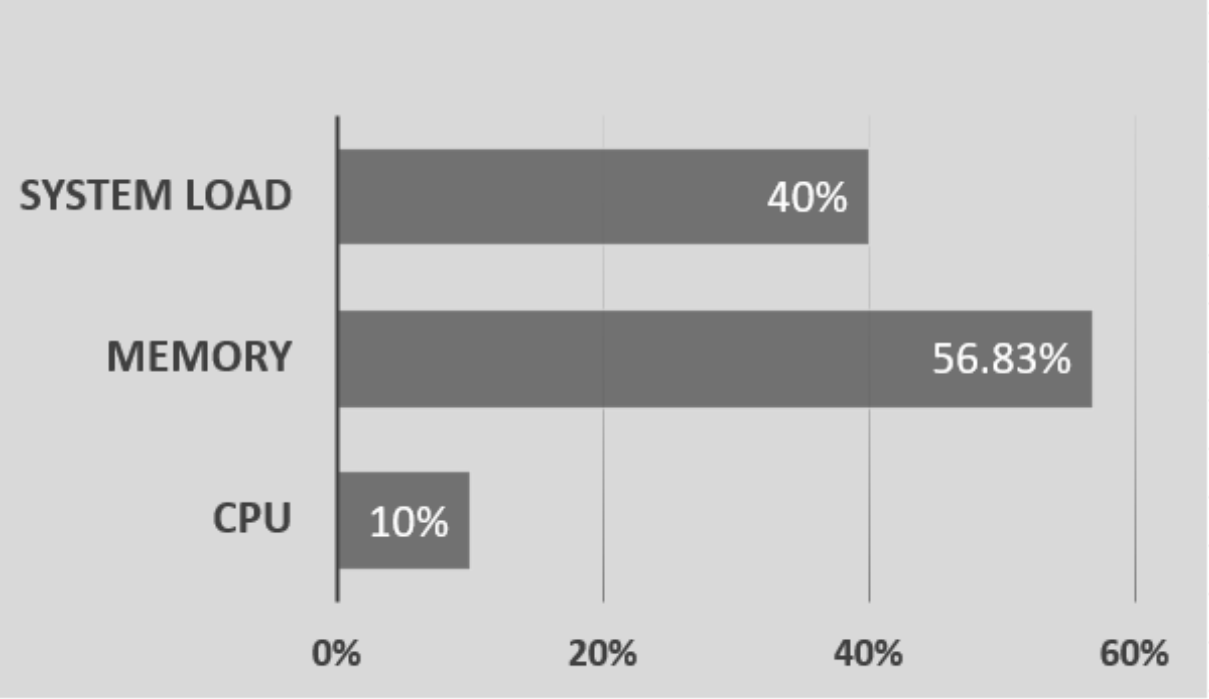


Figure 4.38: SUTMS IPS Performance Improvement after Phase II NTOP Integration (Signature Refinement)

4.7.4 Firewall and NTOP Engine Evaluation

SUTMS is equipped with an open-source iptables firewall inspection engine. Iptables are configured with two types of rule sets. One set of rules is manually configured based on observed traffic patterns, and another is automatically created according to the IoC feeds using STIX/TAXII. The rules are generated using the conversion scripts and programs specified in Section II. This way, the STIX/TAXII feeds are automated, optimized, and customized. In the end, there is a default block rule to make sure any traffic that is not matched will be blocked. The evaluation criteria used for analyzing firewall and NTOP performance are as follows:-

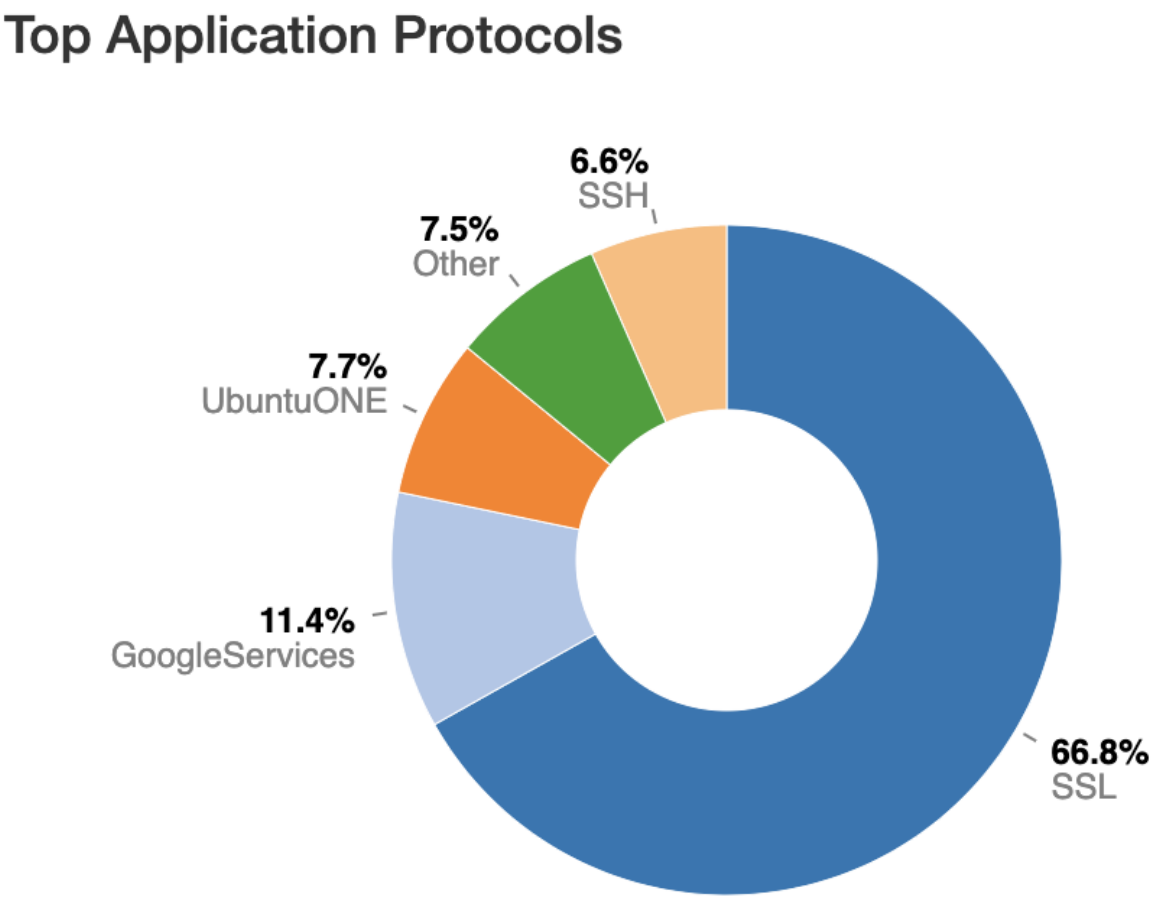


Figure 4.39: Traffic Detected by SUTMS NTOP Engine.

i) Allowed rules for traffic detected by NTOP (Fig. 4.39) were created.

ii) Default block any rule is created to block any unknown traffic.

iii) Firewall hits & performance were recorded by generating traffic.

iv) During the test, the IDS service was disabled.

One major difference between IPS and Firewall/NTOP performance tests is that during the IPS evaluation, all the core services (firewall/NTOP) were also running to measure the overall system resource usage, however in Firewall/NTOP testing, IDS engine was disabled to calculate the impact of core services on the overall UTM device. It can be inferred from the test results that the average CPU Fig. 4.41 and memory Fig. 4.42 are significantly lower than the one utilized by the IDS engine. The average rule hits observed were around 5000, with the maximum hits of 45,000 in Fig. 4.40. It is also noted that the system load was slightly lower than IDS Phase-II testing with some abrupt spikes, Fig. 4.44.

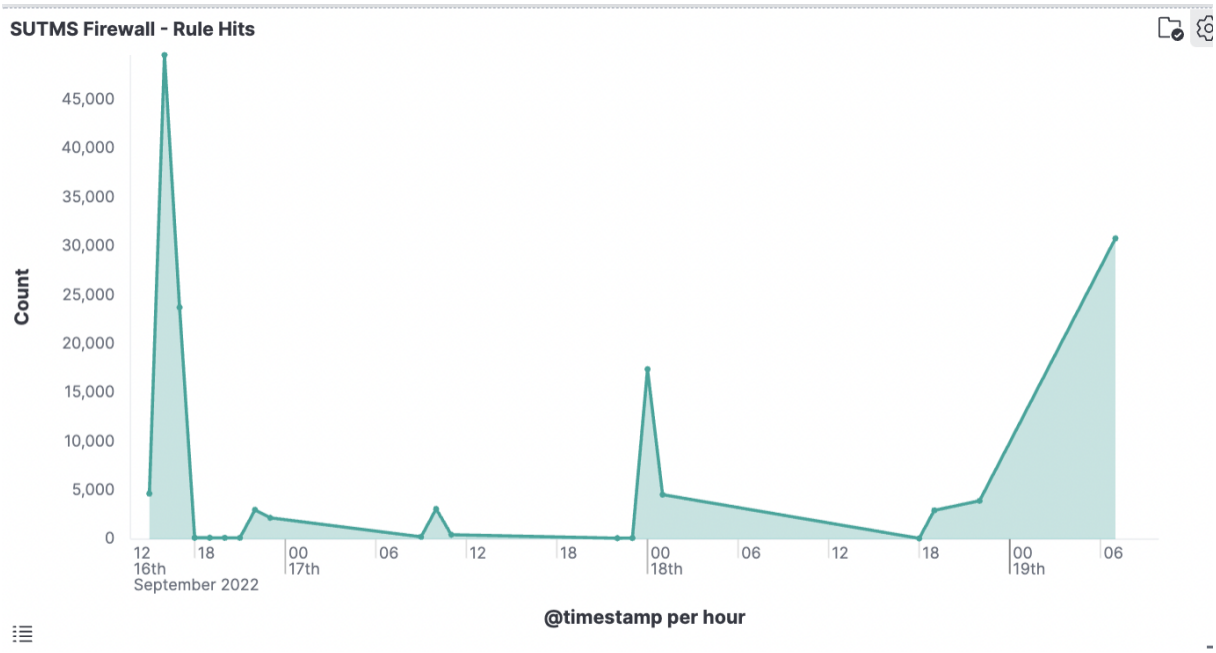


Figure 4.40: SUTMS Firewall Rule Hits

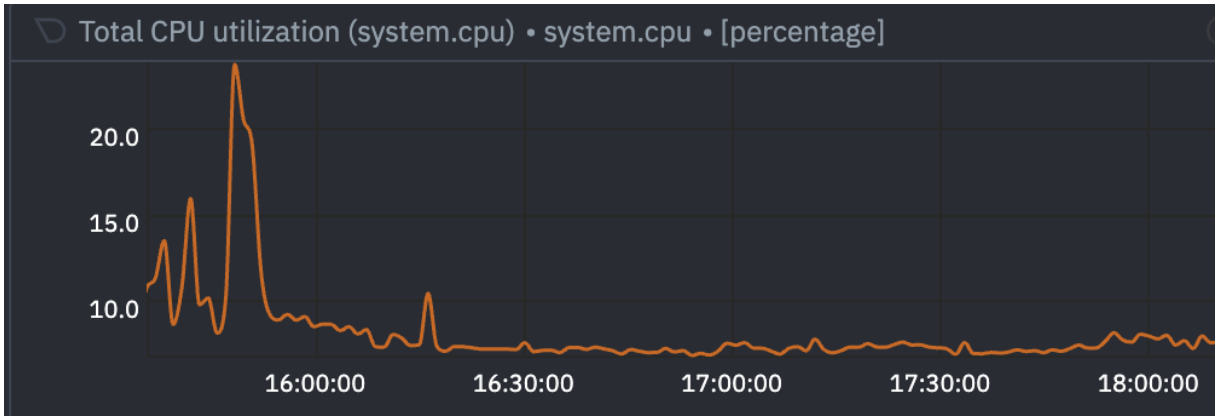


Figure 4.41: SUTMS CPU Usage - Firewall & NTOP Engine.

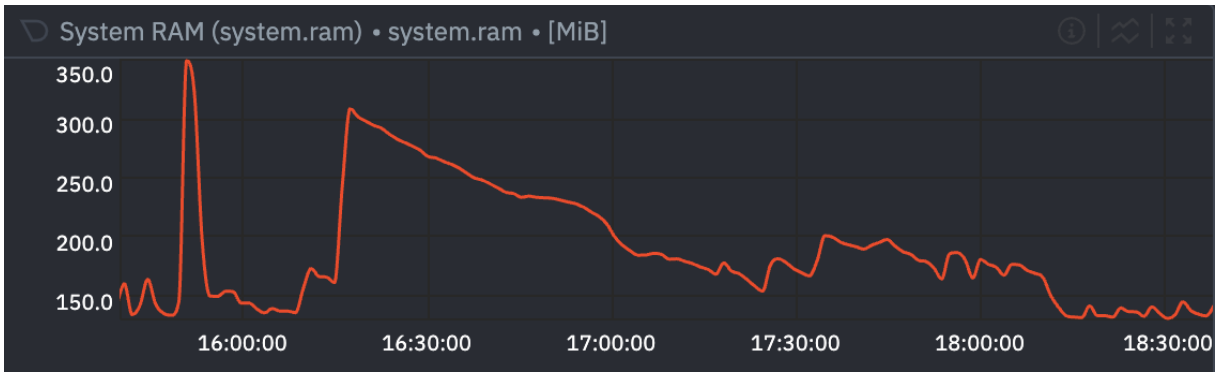


Figure 4.42: SUTMS Memory Usage - Firewall & NTOP Engine.

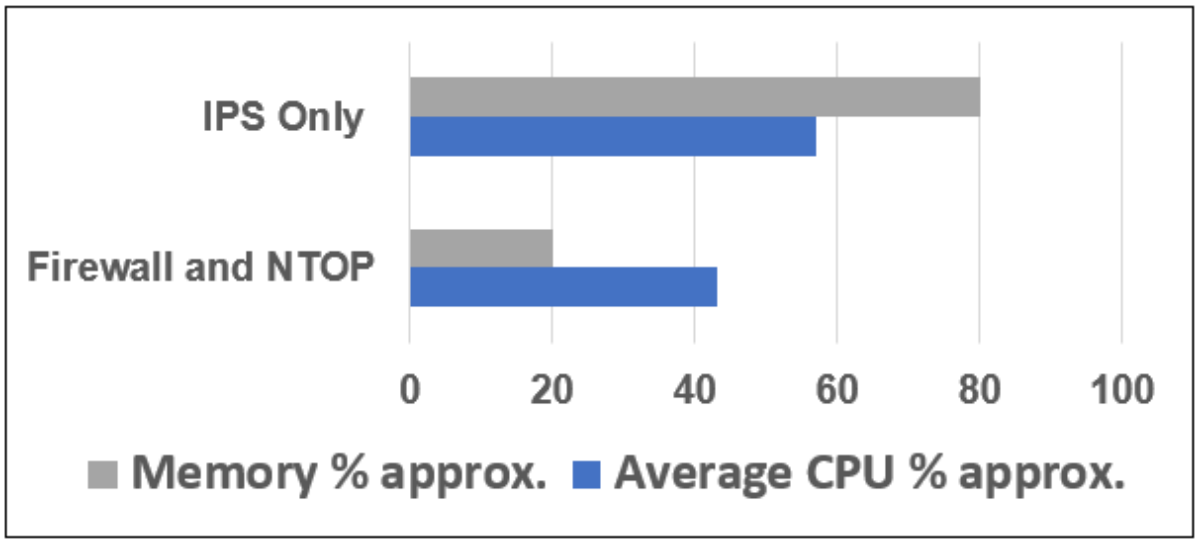


Figure 4.43: CPU & Memory Comparison by UTM Service.

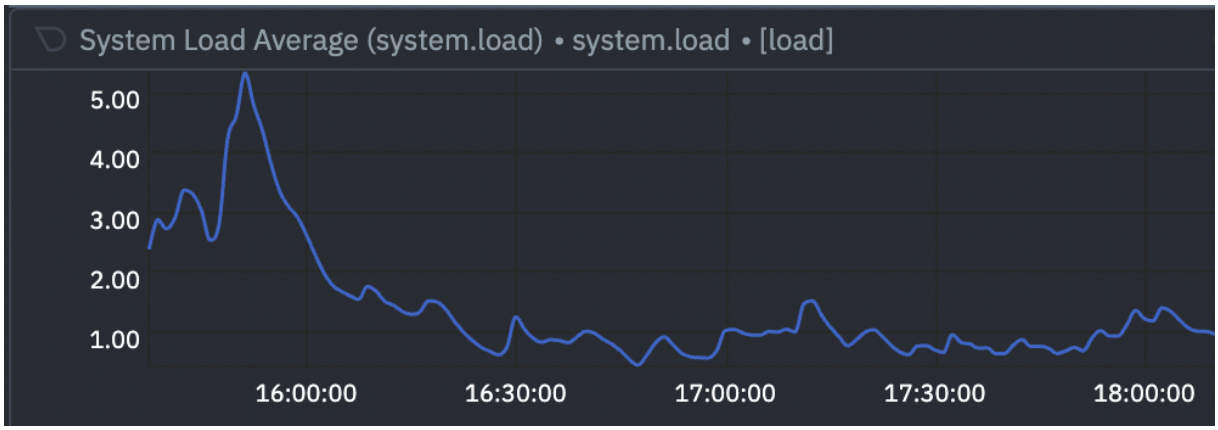


Figure 4.44: SUTMS System Load - Firewall & NTOP Engine.

Table 4.9: Comparison of IPS Phase II, Firewall-NTOP & IPS Only Resource Usage

	Average CPU % approx.	**Mem (GB) approx.**	**Load**
All UTM Services running (IPS Phase II)	21	1	1.2
Firewall and NTOP	9	0.2	1
IPS Only	12	0.8	0.2

Table 4.9 provides a comparison of system resources used by each service along with overall SUTMS utilization. It can be concluded that IPS consumes 80% and 57% of the memory and CPU, respectively, of the entire SUTMS usage (Fig 4.43). It has also been noted that there is a requirement for a centralized asset management system for home devices. Such a system would prove advantageous for tasks such as traffic profiling, firewall rule automation, and signature optimization. Chapter. 5 introduces a lightweight implementation of an asset inventory system. .

Chapter 5

Home Network Device Management Model

5.1 Introduction

Complexity of home networks have increased significantly due to the advent of smartphones, tablets, Internet of Things IoT and remote work. Technology has introduced numerous features into our homes, a person sitting thousands of miles away can remotely monitor his property anytime, air conditioning units can efficiently turn on based on user geolocation. Technological home enhancements have also contributed towards security and management concerns. Higher number of smart devices means more vulnerable systems, managing various makes and model of home appliances adds another layer of complexity. There is a need of a centralized management system for smaller devices that will monitor, manage with integrated security assessment. In this chapter we propose Secure Centralized Management System SCMS, the artifact provides centralized management platform for smart home devices. SCMS will be able to detect and add devices dynamically into the inventory, integrated scanning engine is also be able to flag vulnerable and non-compliant devices. System statistics, load, CPU, memory can be collected in real time and alarms can be generated for any abnormal behavior. SCMS is capable of supporting wide variety of computing devices and alerting on any possible rogue devices in network. The proposed solution not only addresses the management problems associated with home devices, it also increases visibility by providing the security risk associated

with each device.

Artificial intelligence, machine learning and improvements in voice recognition technologies have introduced a new line of computing devices, Amazon Alexa voice service has revolutionized the connected home appliances by developing a collective response with a single command [249]. Multiple devices can collaborate with each other to improve automation and operate without user intervention. Home sensors can be used as smoke detectors, security cameras, fire alarms and other critical tasks that are directly related to safety and well-being of human life. Security breaches and malfunctioned devices can have a significant impact, it is therefore necessary to manage and monitor them. It is unfortunate that network home appliances are designed by manufacturers without considering security, 70% of internet devices are vulnerable to threats [250]. Indirect threats have also raised privacy concerns, adversaries can gather a person's daily routine by analyzing the power consumption of smart homes [251].

There is a need to develop a management platform with security intelligence and dedicated towards home devices. The proposed SCMS is a multicomponent management system, the solution provides ease of management with a customizable web front end. SCMS incorporates a scanning engine into its management platform. By seamlessly integrating these two components, it not only dynamically identifies the diverse range of devices connected to the home network but also categorizes them based on their associated risk levels. This approach effectively resolves the challenges of device identification and risk assessment through a centralized platform. SCMS not only empowers home users with awareness of the devices on their network but also enables them to easily associate each device with its respective risk level, all through a unified interface.

5.2 Related Work

There are numerous technology specific management solutions developed, artifacts proposed utilizing IoT's, Blockchain, RFID's and sensor systems are some of the common research areas. Lightweight implementation of asset management system that has security as a component is often neglected as security is overall treated as part of a broader enterprise solution like SIEM. Yuan and Xin [252] presented RFID based inventory system, asset gets added into database after being validated using radio signals. The research has commercial significance, as RFID readers are commonly deployed in stores and warehouses to process barcodes. In home networks, inventory is not fully known and use of RFID devices is inefficient and costly. Smart devices can also be categorized according to applications, Kivimäki, Sinkkonen, Marttonen, *et al.* [253] contributes in the field of IoT management by developing asset groups based on applications, it also takes advantage of "Flexible Asset Management Model (FAM)" [254] and "Life-Cycle Model for Maintenance Service Management " [255]. FAM is a decision-making model, it has a business significance as it correlates return on investment and operational cost. Life-Cycle Model considers device maintenance as part of decision-making process, maintenance, and management are significant in determining IoT viability as it impacts associated cost. Assets can be grouped together by applying FAM and Life-Cycle Model, however such models have commercial relevance. Aboubakar, Kellil, and Roux [256] conducts a comprehensive survey on IoT network management, the study highlights on IoT management protocols and next generation frameworks associated with cloud and software defined networks(SDN). Typical network management platforms relies on agent, manager, and a device itself [256]. Device CPU, wireless network range and throughput are the key factors in designing a network management platform, home sensors counts on wireless signals and supported encryption protocols. Encryption protocols like Wired Equivalent Privacy WEP that leverages static keys for encryption and authentication [257] should be avoided at all cost.

A decentralized network management system is another avenue to explore, instead of all the sensors reporting to central authority, clusters of network assets that have common parameters can be self-managed. Karle [258] proposes a model that is self-reliant and independent of centralized system, each radio node collaborates concurrently to manage a radio network without the need of network management system NMS. The ideal of decentralization is elegant and works well when the inventory is already known and number of nodes are fewer. Service based management solutions are beneficial for faster deployment of services, policies, and configurations. Mishra, Dhakwal, Pathan, *et al.* [259] introduces CSPMS Centralized Squid Proxy Management System that automatically registers Squid proxy to a centralized system, apart from enrollment other features like backup, restoration, deployment etc. are also provided. In a home network environment, services and vendor list are widely different, therefore any application or service based solution will have limited applicability. Artificial intelligence can be used to learn data patterns and automate device management based on the dataset, neural networks can be used to develop pattern matching by collecting and training data, observing patterns can be significant for optimization and network management [260]. IoT acquisition, storage, and lifecycle can be managed and validated using blockchain, "IEEE Standard for Framework of Blockchain-based Internet of Things (IoT) Data Management" [261] introduces a framework for IoT management using blockchain, it also classifies elements that can be used as blockchain building blocks. Researchers have proposed management solutions that can be categorized as centralized, decentralized, AI-based, cloud-based, service/protocol and application specific. The ideal solution for home network device management can utilize previous research conducted and develop a robust multidimensional platform.

5.3 Secure Centralized Management System - SCMS Architecture

SCMS relies on traditional network management protocols like Simple Network Management Protocol SNMP [262], Syslog [263] and non-traditional agents like Zabbix agent[264]. Zabbix is deployed as a core component for management and monitoring, it is completely opensource solution available and free of cost under GPL license [265]. Zabbix is highly scalable and customizable, unlimited number of nodes can be administered via webui and source code can also be modified as needed. Home network devices including IoT's, sensors and any gadget that supports SNMP is supported. SCMS is distinct from conventional monitoring systems as it delivers security along with management. SCMS has following feature enhancements: -

- Integration of thousands of nodes via Zabbix platform [265].
- Compact on premises solution suitable for Home networks.
- Monitoring of on premises home devices from anywhere via Microsoft Azure fusion.
- Dynamic security rating of each appliance ensures customer confidence.
- Improve home visibility by automatic and manual detection of sensors.
- Customize alerting and system resource monitoring.

Fig. 5.1 represents the core components of SCMS architecture, Zabbix, NMAP [266] and Microsoft Azure [267] works in co-ordination to provide an end user a web interface that is accessible from anywhere and anytime. Home user's main concerns are, that the device inventory should be current and clarity of high risk devices, the system is designed by keeping in mind home user requirements. Fig. 5.2 illustrates the tasks involved in

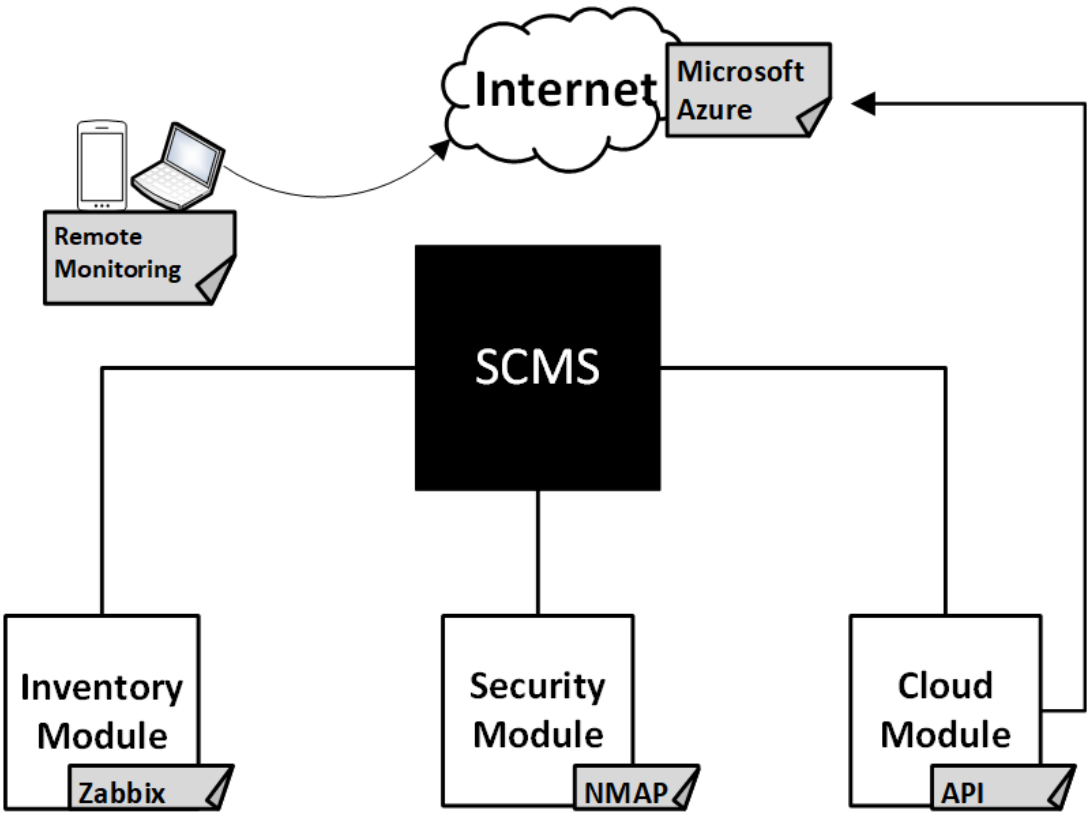


Figure 5.1: SCMS Architecture

SCMS execution, the steps are as follows :-

i) Low resource SCAN is performed every hour, if the device is detected it will be added to the inventory. System resources, usage, alerting and monitoring can also be started upon addition.

ii) In case the device is not detected, manual addition is required with user intervention.

iii) Both the auto and manual assets are sent to the security engine, quick NMAP operating system OS detection scan is performed to discover associated OS.

iv) Security rating system (High, Medium and Low) is added to the scan output, it is easy to decipher for non-technical users.

v) The final output is sent to a file (report), it can be available in text, XML or another format.

vi) The report is uploaded to the Microsoft Azure cloud via Application programming interface API.

vii) Users can access the report via browser or Azure mobile application.

The tasks are automated and run at regular intervals, various scripts are developed and scheduled. Data is shared to the cloud via secure encrypted Transport Layer Security TLS protocol. Scanning, scripting and data transfer is carefully designed by contemplating limited resource and low bandwidth connectivity.

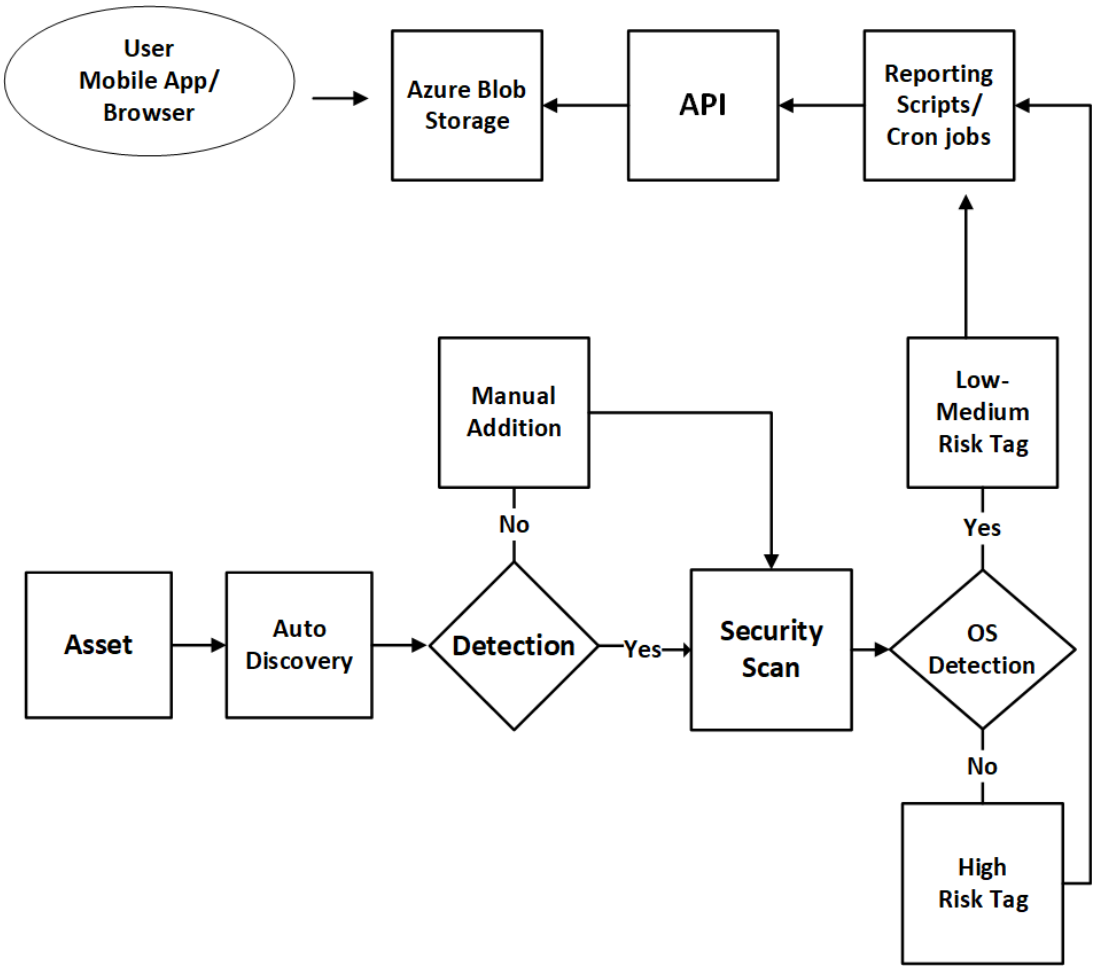


Figure 5.2: SCMS Process Flow

5.3.1 Inventory Module

Asset inventory is one of the critical functionality of SCMS architecture, Zabbix is configured to automatically discover devices in home network subnet 192.168.0.0/16 by utilizing the "discovery" rule shown in Fig. 5.3. System tries to find devices by using SNMP, ICMP and Zabbix agent, SNMP can be configured with a default or custom community strings. The scan is performed every 2 hours to make sure appliances are identified as soon as they come online. Zabbix provides an intuitive interface to monitor and manage a wide variety of systems, devices that are not detected by "discovery" rule will have to be added manually.

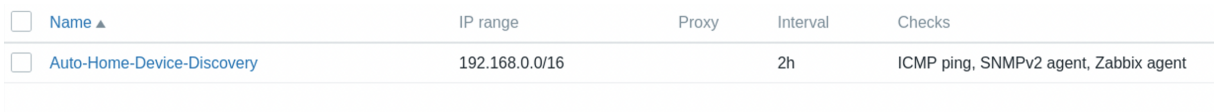


Figure 5.3: Zabbix Discovery Rule. [264]

5.3.2 Security Module

The Security Module is responsible for categorizing the devices by High, Medium and Low risk. NMAP scan is performed on the devices discovered by the inventory module, the goal is to fingerprint the OS and tag the corresponding output with a score of 1-10 (1 being the highest risk). Once the scan is performed, a series of scripts are run to filter out the output to human-readable format and assign the proper security labels to each device. The scripts are automated by using a cronjob and final results are saved locally and uploaded to the cloud. The purpose of security module is to highlight the high risk devices, so that it can be addressed. Appliances with "Undetected OS" in an enterprise networks can be an indication that devices are secure or scans are blocked by the firewall, in home networks the situation is different as firewalls are not usually deployed within the local area network and host devices also lack intrusion prevention systems. It is alarming

to identify systems with "Undetected OS" in home networks, therefore SCMS classifies them as "high-risk" devices.

5.3.3 Cloud Integration

The data can be accessed anytime from anywhere either via mobile application or browser with the help of Microsoft Azure API. The output scoring file generated as part of security module processing is uploaded to an Azure Blob Storage via azcopy [268]. Access to a Blob storage can be restricted via API key and selected public IP addresses. TLS is used as encryption in transit and security score of devices can be access in real time. There is also an option to alert home user about insecure devices via email, however that will require integration of simple mail transfer protocol SMTP [269].

5.4 SCMS Evaluation and Validation

5.4.1 Test Network

The network was designed to evaluate SCMS functionality and performance, network topology is demonstrated in Fig. 5.4. The environment was build to simulate the real world scenario, Windows 10, Linux, MacBook, and IoT devices were deployed. SCMS server was installed on a virtual instance of Ubuntu 20.04 ARM architecture with 3.20 GHz Processor, 6 GB of Ram and 20Gb of Disk space. Zabbix 5.0.31 [264] and NMAP [266] were set up and accessible over a network. Connectivity is provided both via WiFi and Ethernet, dynamic host configuration protocol DHCP is utilized for automatic assignment of IP addresses. Wireless Access Point acts as an internet gateway, firewall and DHCP server, network scope of 192.168.0.0/16 is configured on the access point. Access to internet is filtered via firewall and any inbound access from internet is blocked.

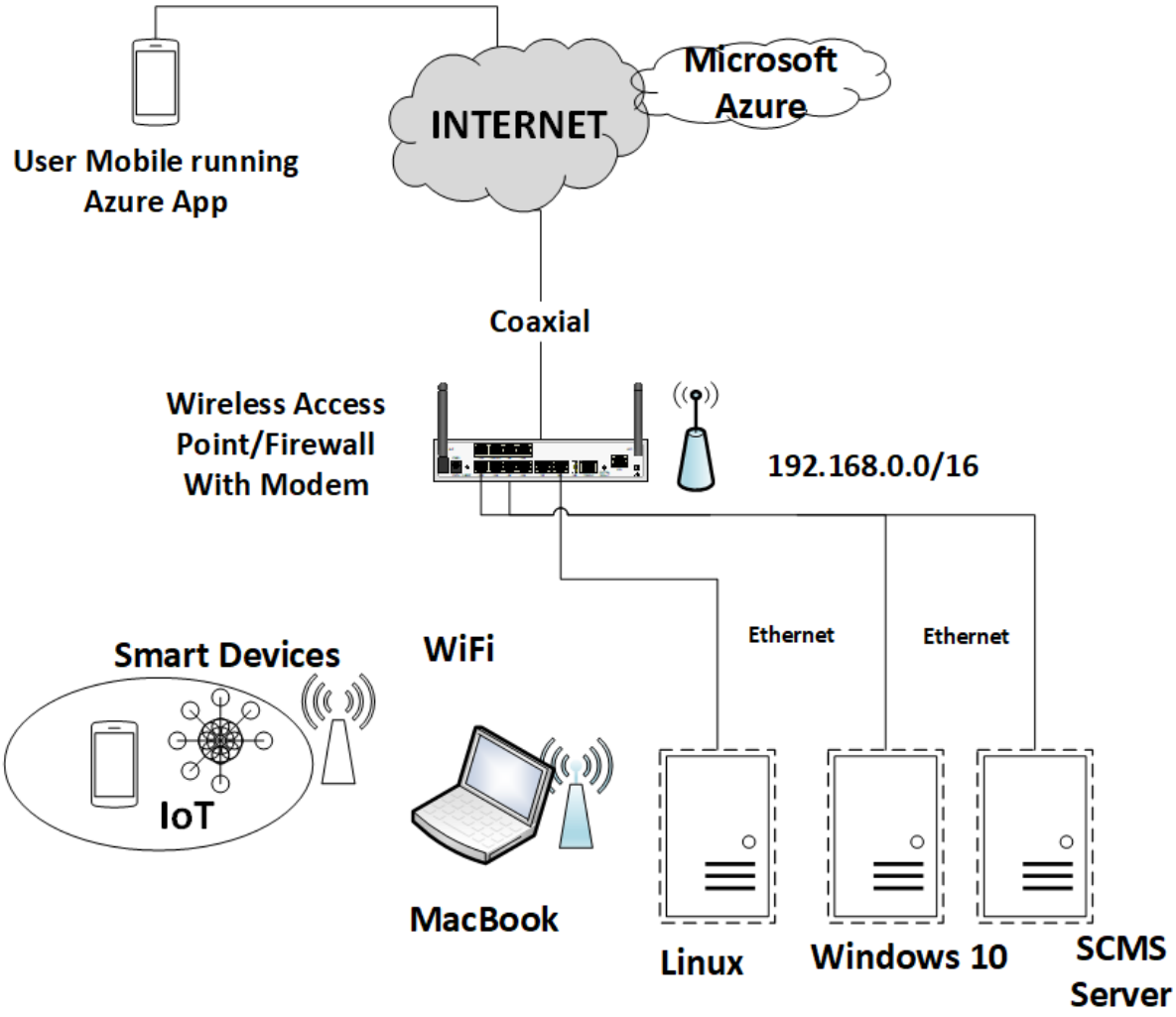


Figure 5.4: Test Network Topology.

5.4.2 System Detection Phase

Zabbix's auto-discovery process ran to discover devices in our test network. In order to validate typical home device inventory, the Zabbix agent was deployed on Raspberry Pi (simulated IoT) , MAC, and SNMP service on the Windows home firewall. It is observed that devices can be identified automatically as long as the SNMP strings and Zabbix agent are pointing to the correct server. System resources disk, CPU, interface status, traffic, etc. can also be monitored. IoT resource monitoring can assist in identifying anomalies and trigger alarms in cases above-normal thresholds were observed.

5.4.3 Security Scoring Phase

SCMS is the asset management system that is able to assign a security score to a device after discovering it. Table.5.1 summarizes risk scores, the scoring system ranges from 1 to 10, with 1 indicating the highest and 10 the lowest risk level. Scores are determined through NMAP scans, where any undetected operating systems are automatically classified as 'high risk' and assigned a score of 1. Linux-based devices are categorized as 'medium' with a default score of 5. However, as the number of open ports fluctuates, the score adjusts accordingly. For instance, a standard Ubuntu machine with typical open ports falls into the 'medium' category, but one with an unusually high number of open ports may receive a score lower than 5. The primary criteria for risk assessment include open port identification and operating system discovery. Nevertheless, this assessment can be further enhanced through the integration of advanced scanning tools capable of detecting patch levels.

Table 5.1: SCMS Risk Score

Score	**Risk Label**
1-4	High
5	Medium
6-10	Low

The administrator is able to prioritize patches and updates for high-risk devices. SCMS makes use of the NMAP scripting engine to automate the security tagging. It is a multistep process consisting of the integration of NMAP script, custom scripts by device type, and the verification of results and merging of files, as discussed in the text below.

Integration of NMAP Script

NMAP script is run via SCMS and resulting output is filtered for IP addresses and operating system OS related tags. XML script "scms_scan.xml" is run to detect the IP address and find open services along with the OS as shown below.

```
Script:/usr/bin/nmap --privilege -oN /home/scms/scms_scan.xml -O
HOST.CONN
./usr/bin/nmap --privilege -oN /home/scms/scms_scan.xml -O
192.168.200.157
Starting Nmap 7.80 ( https://nmap.org ) at 2023-05-22 19:19 CDT
Nmap scan report for 192.168.200.157
Host is up (0.0079s latency)
Not shown:  994 closed ports
PORT STATE SERVICE
22/tcp open ssh
143/tcp open imap
1720/tcp open h323q931
3000/tcp open ppp
9090/tcp open zeus-admin
10000/tcp open snet-sensor-mgmt
```

No exact OS matches for host (if we know what OS is running on it, see https://nmap.org/submit/)

Once the script is successfully run, the OS can either be matched or not, SCMS marks any undetected OS as "High" risk.

Custom scripts by device type

Custom scripts are created according to the OS, score is automatically added based on the output of scanning engine. Table.5.2 enlists shell scripts created to identify various OS types and assign the respective score. Once the OS is detected via security scan, scores are assigned and exported to files that are labelled as low_risk, medium_risk and high_risk.

Table 5.2: Shell Scripts For Security Scoring

microsoft.sh: `grep -E -i '192.168.*.*| Microsoft' scms_scan.xml microsoft_scan ; sed '/^Nmap/ s/$/ *** Score=5, Risk=Medium *** /' microsoft_scan > medium_risk`
linux.sh: `grep -E -i '192.168.*.*| linux' scms_scan.xml >linux_scan ; sed '/^Nmap/ s/$/ *** Score=5, Risk=Medium *** /' linus_scan> medium_risk`
windows.sh: `grep -E -i '192.168.*.*| windows' scms_scan.xml >microsoft_scan ; sed '/^Nmap/ s/$/ *** Score=5, Risk=Medium *** /' microsoft_scan> medium_risk`
router.sh: `grep -E -i '192.168.*.*| cisco' scms_scan.xml >cisco_scan ; sed '/^Nmap/ s/$/ *** Score=7, Risk=Low *** /' cisco_scan > low_risk`
router_netgear.sh: `grep -E -i '192.168.*.*| netgear' scms_scan.xml >netgear_scan ; sed '/^ Nmap/ s/$/ *** Score=7, Risk=Low *** /' cisco_scan > low_risk`
Apple.sh: `grep -E -i '192.168.*.*| Apple' scms_scan.xml >apple_scan ; sed '/^Nmap/ s/$/ *** Score=5, Risk=Medium *** /' apple_scan > medium_risk`
Freebsd.sh : `grep -E -i '192.168.*.*| FreeBSD' scms_scan.xml >linux_scan ; sed '/^Nmap/ s/$/ *** Score=5, Risk=Medium *** /' linux_scan > medium_risk`
no_os.sh: `grep -E -i '192.168.*.*| No OS matches' scms_scan.xml >undetected_OS_scan ; sed '/^Nmap/ s/$/ *** Score=1, Risk=High (NO OS DETECTECD) *** /' undetected_OS_scan > high_risk`

Verifying of results and merging of file

Once the execution of shell scripts are completed, the files along with their corresponding scores are created.

There are multiple files created, therefore it is important to merge them into one file. Table.5.3 shows the machines identified with their risk score, the three files (low, medium,

Table 5.3: Files by Risk

Medium Risk root@scms:/home/scms# more medium_risk # Nmap 7.80 scan initiated Mon May 22 19:19:52 2023 as: /usr/bin/nmap –privilege -oN /home/scms/myscan1.xml -O 192.168.200.156 Nmap scan report for 192.168.200.156 *** Score=5, Risk=Medium ***
High Risk root@scms:/home/scms# more high_risk # Nmap 7.80 scan initiated Mon May 22 19:55:02 2023 as: /usr/bin/nmap –privilege -oN /home/scms/scms_scan.xml -O 192.168.200.157 Nmap scan report for 192.168.200.157 *** Score=1, Risk=High(NO OS DETECTED) ***
Low Risk root@scms:/home/scms# more low_risk # Nmap 7.80 scan initiated Mon May 22 20:01:10 2023 as: /usr/bin/nmap –privilege -oN /home/scms/scms_scan.xml -O 192.168.200.160 Nmap scan report for 192.168.200.160 *** Score=10, Risk=Low ***

and high) are merged together into one file i.e., scms_scan_result and the output is shown in Table.5.4. The file can be accessible within a local area network, however in order to ensure the file is accessible from anywhere as devices get added, the cloud module is integrated into scms.

Table 5.4: Merging output scoring file

root@scms:/home/scms# cat low_risk medium_risk high_risk ¿scms_scan_result root@scms:/home/scms# more scms_scan_result # Nmap 7.80 scan initiated Mon May 22 20:01:10 2023 as: /usr/bin/nmap –privilege -oN /home/scms/scms_scan.xml -O 192.168.200.160 Nmap scan report for 192.168.200.160 *** Score=10, Risk=Low *** # Nmap 7.80 scan initiated Mon May 22 19:19:52 2023 as: /usr/bin/nmap –privilege -oN /home/scms/scms_scan.xml -O 192.168.200.156 Nmap scan report for 192.168.200.156 *** Score=5, Risk=Medium *** # Nmap 7.80 scan initiated Mon May 22 19:55:02 2023 as: /usr/bin/nmap –privilege -oN /home/scms/scms_scan.xml -O 192.168.200.157 Nmap scan report for 192.168.200.157 *** Score=1, Risk=High(NO OS DETECTED) ***

5.4.4 Cloud Access Phase

Microsoft Azure blob storage is used to upload the security score file and accessible from anywhere securely. There are two main steps for setting this up:

Step-1 : Microsoft Azure Blob storage has to be configured according to the template defined in Appendix.B.

Step 2 - Access to blob storage is only allowed from legitimate IP address ("10.0.0.0/16", "73.246.185.21/32) as specified in the template. The file is uploaded using a script shown in Fig.5.5. There is a token generated from Azure blob storage and used to upload a file, Fig.5.6 shows the successful transfer of a file to the online blob storage via script.

```
root@scms:/home/scms/azcopy_linux_arm64_10.18.1# more scms_azure_transfer.sh
./azcopy cp "/home/scms/scms_scan_result" "https://onlineaccess.blob.core.windows.net/scms?sp=ra
cwdli&st=2023-05-24T23:47:22Z&se=2023-05-25T07:47:22Z&spr=https&sv=2022-11-02&sr=c&sig=HHFOc4Tso
VpnMzb4Zwn5DEkmI1sKoHv7cZo5fW2XhOQ%3D"
root@scms:/home/scms/azcopy_linux_arm64_10.18.1#
```

Figure 5.5: Script for cloud upload.

```
root@ubuntu-linux-20-04-desktop:/home/scms/azcopy_linux_arm64_10.18.1# ./azcopy cp "/home/scms/scms_scan_result" "https://onlineaccess.blob.core.windo
ws.net/scms?sp=racwdli&st=2023-05-24T23:47:22Z&se=2023-05-25T07:47:22Z&spr=https&sv=2022-11-02&sr=c&sig=HHFOc4TsoVpnMzb4Zwn5DEkmI1sKoHv7cZo5fW2XhOQ%3D
"
INFO: Scanning...
INFO: Any empty folders will not be processed, because source and/or destination doesn't have full folder support

Job 0bd4f12e-3f08-a444-48ed-87e052b11eb8 has started
Log file is located at: /root/.azcopy/0bd4f12e-3f08-a444-48ed-87e052b11eb8.log

100.0 %, 1 Done, 0 Failed, 0 Pending, 0 Skipped, 1 Total, 2-sec Throughput (Mb/s): 0.0024


Job 0bd4f12e-3f08-a444-48ed-87e052b11eb8 summary
Elapsed Time (Minutes): 0.0334
Number of File Transfers: 1
Number of Folder Property Transfers: 0
Number of Symlink Transfers: 0
Total Number of Transfers: 1
Number of File Transfers Completed: 1
Number of Folder Transfers Completed: 0
Number of File Transfers Failed: 0
Number of Folder Transfers Failed: 0
Number of File Transfers Skipped: 0
Number of Folder Transfers Skipped: 0
TotalBytesTransferred: 600
Final Job Status: Completed
```

Figure 5.6: Successful transfer of risk score file .

Step 3 - In the last step, upload can be automated using a cron job, file can be uploaded at regular intervals. Fig. 5.7 exhibits the cron job created for regular uploads. Once the file is uploaded to the blob storage, it can be accessible from anywhere via browser.

```
root@scms:/etc/cron.d# more scms_cron_job
0 0 * * * /home/scms/azcopy_linux_arm64_10.18.1/scms_azure_transfer.sh
root@scms:/etc/cron.d#
```

Figure 5.7: Cron job for file transfer automation.

5.4.5 Results

The results are analyzed based on two criteria, i.e., device detection automatically or manually and resource utilization by SCMS.

Device Detection

The accuracy of the SCMS relies on the successful detection of diverse devices. To achieve this, dynamic scans are conducted on devices within the 192.168.0.0/16 network. The protocols used for detecting live machines include ICMP and SNMP. Figure.5.8 presents the list of discovered devices, which can alternatively be identified by utilizing the Zabbix agent. Additionally, Figure.5.9 illustrates the protocols used for these discoveries. System statistics like uptime, memory, cpu usage etc. can also be collected from the device. Fig. 5.10 exhibits the CPU spikes, and Fig. 5.11 gives us inbound/outbound traffic patterns of a Raspberry Pi (simulated IoT). SCMS was successfully able to detect MacBook, Windows, IoT (Raspberry Pi) and home router.

Discovered device ▾	Monitored host	Uptime/Downtime
Local network (9 devices)		
192.168.100.1		92 days, 21:21:44
192.168.48.1		103 days, 01:16:08
192.168.47.1		103 days, 01:49:33
192.168.46.1		103 days, 02:19:40
192.168.1.1 (my.firewall)		03:31:09
192.168.0.254		104 days, 21:04:22
192.168.0.6		04:00:34
192.168.0.2		04:01:03
192.168.0.1		104 days, 21:34:00

Figure 5.8: Test network inventory [264].

Name ▲	Interface	Availability	Tags	Problems	Status	Latest data	Problems	Graphs	Screens	V
Firewall-Router-AP	192.168.253.1: 161	ZBX SNMP JMX IPMI			Enabled	Latest data	Problems	Graphs	Screens	V
iot	192.168.200.156: 10050	ZBX SNMP JMX IPMI			Enabled	Latest data	Problems	Graphs 14	Screens 2	V
macbook	192.168.253.12: 10050	ZBX SNMP JMX IPMI		1	Enabled	Latest data	Problems 1	Graphs 6	Screens 1	V
scms	127.0.0.1: 10050	ZBX SNMP JMX IPMI		1 1	Enabled	Latest data	Problems 2	Graphs 23	Screens 4	V
windows-sensor	192.168.253.13: 161	ZBX SNMP JMX IPMI			Enabled	Latest data	Problems	Graphs 8	Screens 2	V

Figure 5.9: Test network inventory by agent type [264].

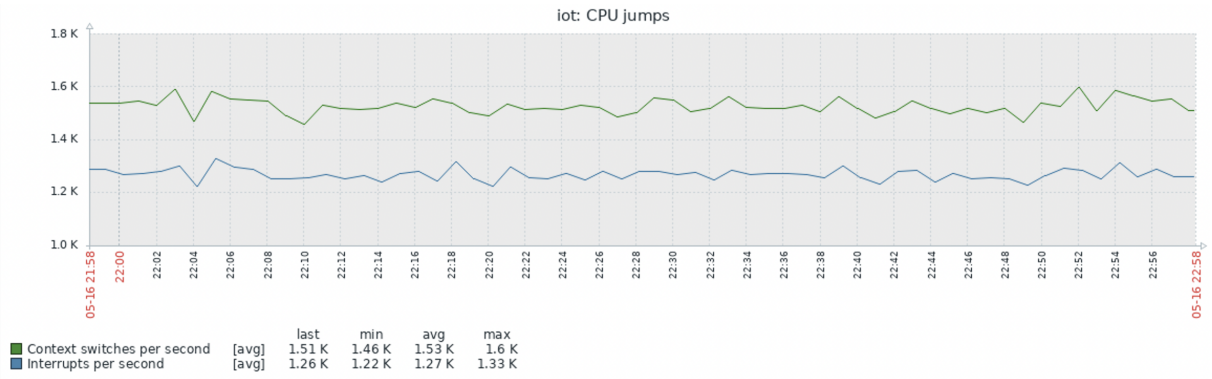


Figure 5.10: IoT CPU spike graph [264].

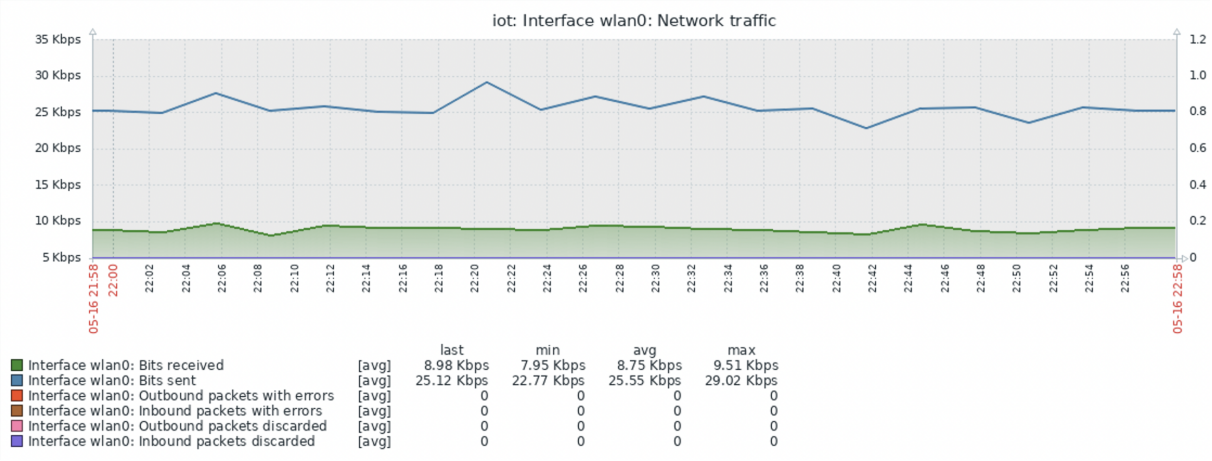


Figure 5.11: IoT interface status [264].

System Resource Utilization

The CPU usage of the SCMS system is depicted in Fig. 5.12. While there were occasional CPU spikes reaching 75% and 45%, the overall CPU utilization remained below 30%. It is worth noting that these spikes did not appear to be correlated with any specific SCMS task. The memory usage, as illustrated in Fig. 5.13, consistently hovered around 1.07 GB of RAM. Furthermore, as represented in Fig. 5.14, the maximum system load reached 1.80, A load of 1.80 indicates that the CPU was 80% overloaded, or it can be interpreted as having one CPU with 1.80 running processes, with 0.80 processes waiting for their turn.

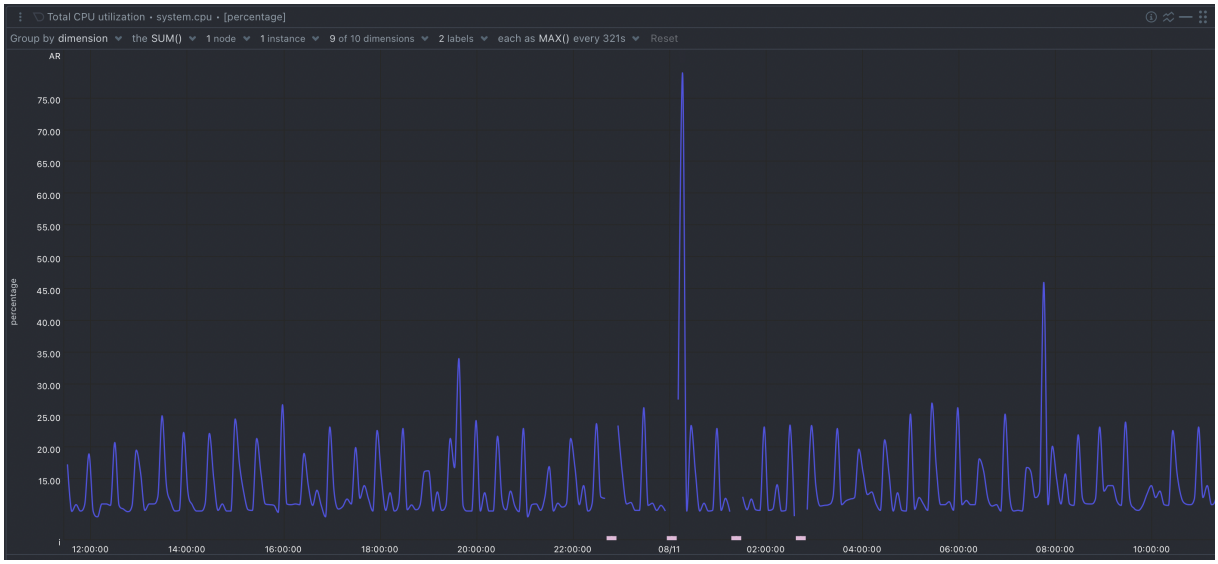


Figure 5.12: SCMS CPU usage.

Figure 5.13: SCMS memory usage.

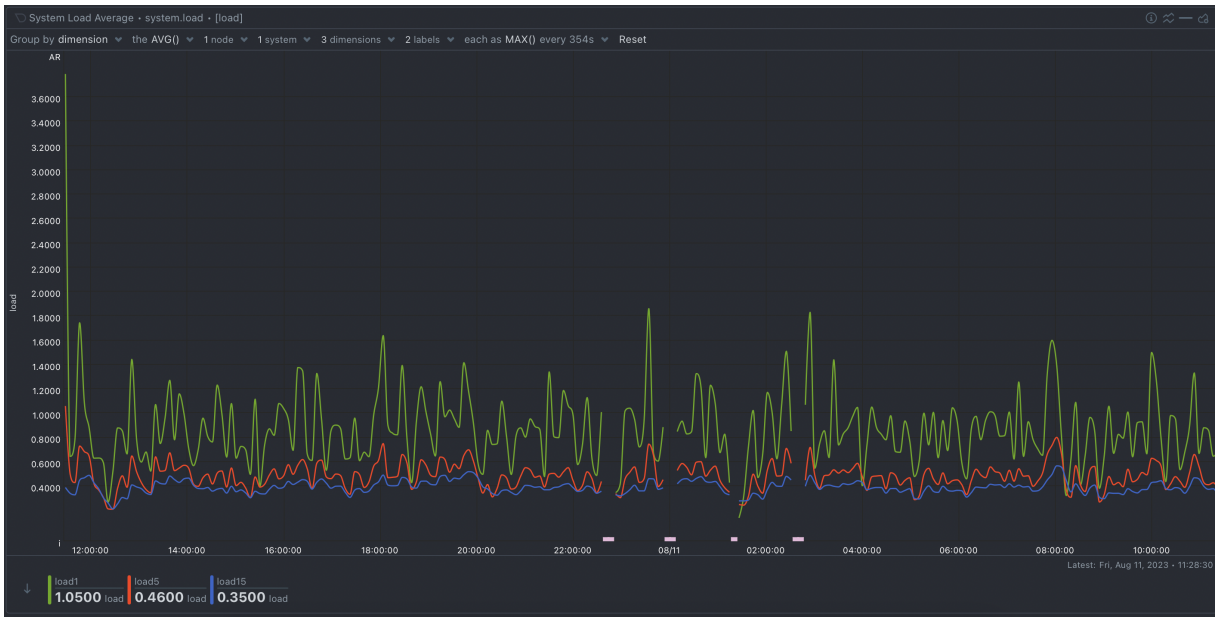


Figure 5.14: SCMS load usage.

5.5 SCMS research Challenges

The proposed management platform is a centralized asset management system accessible remotely, with the capability to conduct security assessments of devices. While the system can automatically detect common home devices with well-known operating systems, it may face challenges when detecting devices with customized operating systems. Given the extensive list of IoT vendors, automated sensor detection may prove challenging. The security assessment relies on the NMAP scanning engine, and the scores generated by SCMS provide a basic indication of the risk level associated with the devices. To obtain more precise results, more comprehensive vulnerability assessment tools will be necessary. To enable remote management of SCMS and automated alerts, internet access and a Microsoft Azure subscription is required. This research opens up opportunities for future researchers to explore centralized management solutions, whether they are on-premises, cloud-based, or integrated into existing SOHO routers. Furthermore, there is a growing demand for dynamic asset feeds categorized by device type to enhance the detection process.

Chapter 6

Contributions and Future Outlook

6.1 Contributions

The research makes a significant contribution to the advancement of a unified threat management framework tailored for small office and home networks. The proposed solution integrates advanced security features into a device with constrained computational resources. The framework leverages flow data to deliver dual benefits: optimizing IPS signatures and detecting anomalies effectively. The implementation of automated IoC feed ingestion and the dynamic generation of firewall access control lists within SUTMS has led to the development of an effective antibot solution. By integrating Suricata signatures with the ntop flow detection module and combining iptables with STIX/TAXII feeds, we achieved a substantial enhancement in SUTMS's inspection capabilities, leading to notable performance improvements. Furthermore, the inclusion of application data from ntop not only improved IDS performance but also resulted in a notable reduction in false positives. We conducted benchmarking on critical UTM components while monitoring system resource utilization. Specifically, we concurrently tested the Firewall, IDS, Antibot, and Anomaly detection engines to assess their accuracy and effectiveness. SCMS enhances SUTMS by offering complementary benefits in the form of centralized management for home devices.

Another significant contribution of this study is its identification of security challenges specific to home networks. This information serves as a valuable resource for future researchers who can leverage the SUTMS framework to address these limitations and

challenges effectively.

6.2 Future Outlook

Home networks are typically more heterogeneous and complex than corporate networks when considering the coexistence of multiple technologies and user technical skills. Corporate networks have trained dedicated staff and allocated budgets with vendor-support contracts. In contrast, home networks are set up and maintained by users with limited skill sets. In this section, we identify the main open challenges that are faced by UTM systems in their entirety, and need to be addressed in future research. That is, we focus on the challenges that the overall UTM system faces [and not the challenges that individual UTM components (firewall, IDS, antibot) face]. We categorize these open challenges into architecture-related challenges (implementation, scalability, and management) and product-specific challenges (security). The main factors that give rise to novel open UTM system challenges are technological advancements (e.g., IoT and novel protocols) and increasing attack sophistication. Addressing future prospects involves taking into account the research challenges and limitations encountered.

6.2.1 UTM Implementation Issues

In the open-source realm, no single solution can provide firewall inspection along with the next generation of defense, i.e., intrusion prevention, anomaly detection, and antibot capabilities. However, some of the available open-source tools that can provide firewall inspection and intrusion detection services are iptables and Snort, respectively [270]. There are some caveats with both the COTS and open-source SOHO solutions. COTS security solutions are costly and require some advanced technical expertise. Most advanced services require regular updates on a subscription basis, such as intrusion detection signature updates. Vendors charge based on the subscription duration of 1 to 3 years. The

cost and complexity increase as more services are added to the firewall module. Running multiple services on a firewall can reduce inspection rates and cause performance problems. COTS SOHO solutions are ideal for the company's managed small offices, as they have the technical expertise to deploy and manage advance security features running on a UTM system.

However, the COTS solution can be overly complex and costly for home networks that are managed by the users themselves. On the other hand, there are also numerous problems associated with available open-source solutions. Many individual modules are available and can help build the next generation of inspection. However, they are not integrated and normally run as a standalone service. For example, an iptables module is available for stateful inspection, but the iptables module is not integrated with other protections, such as antibot protection. Hence, a UTM solution based on iptables lacks the dynamic creation of access control lists. Out of the box, open-source solutions can also be very resource-intensive for devices with limited resources.

6.2.2 Scalability

Home networks utilize services beyond voice, video, and data. IoT has become an integral part of various required home applications. Sensor networks are one of the contributing factors to design complications. Varied requirements for power, data rate, and IP connectivity give rise to scalability issues. It is possible to quantify the number of laptops, desktops, and mobile computing devices in a home network. However, each smart unit can have more than one sensor with many interfaces. The coexistence of multiple topologies, including ZigBee 802.15.4, WiFi 802.11, and Bluetooth LE 802.15.1, introduces management challenges. Homeowners may end up managing multiple local area networks. Supporting a multi-vendor environment with limited technical expertise is challenging for many homeowners.

Devices with finite power and processing power offload resource-intense processes to

network gateways (such as internet routers), i.e., routers are not only responsible for routing. Network Address Translation (NAT) is required for internet connectivity [271]. NAT tables grow with the number of devices. The router must translate every private address into a public address to route the traffic to the internet. NAT has implications regarding reduced visibility, Quality of Service (QoS) issues, and specific peer-to-peer applications that do not work with NAT [271]. ZigBee uses a 16-bit addressing scheme instead of the 32-bit IP addressing scheme. Hence, ZigBee gateways have to implement NAT for any IP-based communication.

Various UTM services require the transmission of large datasets, such as IDS and application signature updates, over the access network that connects a home network to the internet. Traditional data transmission is based on the principle of replicating messages bit-by-bit at a destination, i.e., the so-called technical-level communication via message transmission defined by Shannon [272] and Weaver [273]. Some of the data that needs to be downloaded to or uploaded from home networks for UTM services can in principle be modelled and predicted by digital twins. Generally, digital twins [274], [275] provide a digital model (e.g., simulation) of real processes, e.g., real robots operating in a physical environment [276], or of a real network operating in a particular setting [277]. Recently, a few studies have begun to explore digital twin-assisted network security mechanisms [278], [279], including digital twin-assisted IDS [70], [280], [281] and anomaly detection [282]. Threat intelligence obtained from diverse cloud UTM sources becomes highly valuable when shared with the on-premises UTM twin. By promptly implementing newer attack mitigation measures, home networks can enhance their security significantly. Similarly, changes detected in home network applications, traffic flows, firewall logs, and intrusion alerts by home UTM systems allow the cloud digital twin to adopt and develop advanced security strategies in response. For such data, it is not always necessary to download or upload the messages bit-by-bit with conventional message transmission; instead, the synchronization status between a digital twin in the cloud or UTM system with the actual

process that is modelled by the digital twin can be verified with the goal-oriented communications paradigm of identification via channels [283]–[288]. Thereby, identification via channels achieves an exponential efficiency gain compared to conventional message transmission, i.e., identification requires the transmission of only approximately the logarithmically scaled number of bits that are required for conventional message transmission. If the identification verifies that the digital twin is still synchronous, then there is no message transmission needed. Only, if the identification finds that the digital twin is out of synch, then message transmission is needed to restore the synchronization.

Future research should explore the adoption of the goal-oriented communication paradigm of identification via channels for reducing the traffic amounts that need to be downloaded to and uploaded from UTM systems in home networks. For instance, future research should explore how digital twins operating in the cloud (or home network) can model that actual processes in the UTM system (or on the internet at large) to enhance the UTM security services. In the digital twin context, future research should examine how the paradigm of identification via channels can reduce the access network traffic that is required to keep the digital twin in the cloud (or UTM system) synchronized with the respective modelled real-life processes. A related future research direction is to exploit the security features afforded by identification via channels [289], [290] while accounting for the security vulnerabilities that arise from operating digital twins [291].

Task-specific hardware modules can achieve significant accelerations of narrowly defined computing tasks, while reducing the energy consumption [292]–[295]. Future research should explore the use of hardware modules in UTM systems, whereby the tradeoffs between increased computing power and reduced energy consumption need to be carefully weighed against the increased cost and potentially reduced configuration flexibility of UTM systems with task-specific hardware modules.

As quantum computing matures in the future, cloud-based quantum malware detection can become a useful tool to perform the processing of network security related computing

tasks, e.g., for network intrusion detection [277], [296], [297]. Future research should explore how home network UTM systems can take advantage of these emerging quantum computing capabilities in the cloud, e.g., how can UTM systems effectively outsource compute-intensive network security tasks to a quantum-computing cloud.

6.2.3 Connectivity Challenges

Home IoT devices rely on wireless technologies to connect to control systems or the cloud. Connectivity is the main component of IoT operations. Various networking protocols and standards are utilized, e.g., WiFi, Bluetooth, LTE, ZigBee, and Z-Wave [28]. IoT devices are manufactured by a variety of vendors, and relying on different protocols introduces numerous challenges. IoT success depends on compatibility and interaction with other home devices. It is challenging to deploy a device that does not integrate with the existing WiFi infrastructure. Interoperability allows devices to communicate, share information, and interact with each other. Therefore, it is critical for an IoT to work in coordination with other home devices. The key incompatibility challenges are the lack of open APIs, proprietary services and applications, as well as complexity. In some cases, vendors have intentionally left a gap to monopolize their products [298].

Challenges associated with IoT introduce complexity in UTM design and implementation. A UTM system has to support sensor connectivity standards, such as ZigBee, Z-Wave LTE, and Bluetooth, and in most cases, it requires different hardware components or radios. Software drivers will also need to be written or updated to incorporate a wide variety of interfaces. Home-connected devices can be in proximity or far apart, which raises a question about the placement of the UTM gateway. Deployments where sensors are dispersed, may require an intermediary relay that supports the respective signals and media types before the traffic will get redirected to a UTM system for advanced inspection. A higher number of sensors in a network can add additional load on the UTM system processor, and increase the UTM system interface usage.

Another emerging connectivity-related challenge arises due to so-called digital nomads that operate their "home offices" in a mobile setting [299]–[302]. The digital nomads may be stationary during a given home office work session, but could potentially also be mobile, e.g., on a lengthy train ride, or in the passenger seat of a recreational vehicle (RV) traveling on a highway (with printers and additional computing hardware in the back of the RV). Future UTM systems should cater to such "nomadic home offices", by providing comprehensive security services in a mobile setting. In particular, the network infrastructure changes as the home office moves from one location to another. Therefore, the settings for core components of home networks, such as the settings for router, switch, WiFi, and ISP, need to be adjusted. However, the UTM configurations are typically static, i.e., routes, IP addresses, and access lists are hard coded. Hence, changes in network topology requires re-programming of static UTM configurations. Such changes can be avoided, if the UTM is able to dynamically learn the network topology and to automatically tune network and security settings without user intervention.

Additional challenges arise with digital nomads that are mobile during a home office work session. During a mobile work session, the UTM should be able to alter its configurations on the fly and without dropping and reconnecting a session. A network of UTM devices that are synchronized according to a user profile instead of static variables can be a useful design for stable and protected mobile work sessions. Shifting of access control lists from static parameters, such as IP addresses and port numbers, to more dynamic content-based fields can also be crucial for supporting mobile home office work sessions.

Also, any UTM-related services that require or interact with edge cloud computing resources need to operate in a mobile edge computing (MEC) context [303]–[306] during a mobile work session. Importantly, the current state information of the various UTM functions needs to be continuously updated. Signature based inspection, such as IDS and application-aware firewalls, as well as threat intelligence feeds require regular updates via a cloud and edge cloud infrastructure [307]–[309]. Future research needs to adapt

approaches for MEC state migration, e.g., [310]–[314] to MEC-supported UTM services, and to conduct comprehensive evaluations of the resulting mechanisms.

An interesting future research direction is to integrate the UTM for mobile nomadic home offices with the threat assessment and defense systems for the intelligent vehicle [315] in which the nomadic office is housed. Future research should explore synergies between intelligent vehicle security and UTM functionalities for nomadic offices operated in intelligent vehicles.

6.2.4 Protocol Challenges

IoT devices have limited processing and energy resources. Therefore, IoT networks are built upon low-energy communication protocols, such as Bluetooth LE, ZigBee, or Z-Wave. An IoT device that wants to communicate to the internet or smartphone applications requires an IP based stack, or the home routers have to support the IoT protocol interface. ZigBee uses a 2.4 GHz ISM band, and Bluetooth LE uses a frequency of 2.4 to 2.485 GHz [28]. Separate receivers are required for signal processing. Therefore, it is easier to integrate the two protocols with the existing WiFi network without the need for new hardware components. The protocols are developed according to market significance, without considering the broader IoT domain. For example, ZigBee is mostly designed for temperature, lighting, and security systems; whereas, Z-Wave is geared towards remote control applications, such as home theater, pool, and automatic meter reading controls [18]. A home network UTM system has to support such protocols for connectivity and should also be able to inspect traffic and prevent any associated malicious payload. A firewall engine should be equipped with an IoT protocol detection function to build access control lists effectively. Similarly, the intrusion prevention module must have the relevant signatures to block any IoT-related attacks.

6.2.5 Management Issues

IoT devices are designed as self-managed systems that do not require human intervention. An IoT device should have the intelligence to monitor its own health and to only send alerts to a control unit (for that particular type of IoT device) in case of any problems. The self-management functionality is intriguing for home automation, as it does not require much supervision. However, the self-management focused design introduces scalability challenges for the next generation of smart homes. Collaboration among different types of home IoT devices is critical for machine learning and automated responses. For instance, temperature sensors may need to coordinate with lighting and security controls to determine that it is OK to turn on/off heat/air-conditioning. In addition, managing, monitoring, updating, and keeping track of all the IoT devices in home networks will be challenging when there is no centralized management control. In Chapter. 5 of this research, we introduce a centralized management platform enhanced with security intelligence to address the management challenges associated with home networks.

A crucial requirement is the development of a centralized database of IoT products categorized by their functionality, associated security score, vendor assessment, and adherence to best security practices. Additionally, empowering home users to anonymously upload device data, report bugs, and share performance statistics via an API is essential for enhancing overall IoT security and data transparency. To enhance affordability and simplify management, vendors should consider offering IoT products as part of bundled packages. This approach would facilitate easier management, upgrades, and deployment of security patches to devices through a centralized portal. Implementing a dedicated internet solely for IoT, i.e., a dedicated IoT internet, could be a potential avenue for addressing this open challenge. By provisioning management servers with stringent security controls, IoT registration could be allowed only after thorough compliance and security

Table 6.1: Percentages of IoT devices that use vulnerable services, where HV = Highly Vulnerable, V = Vulnerable, S = Slightly Vulnerable; the Server Message Block (SMB) protocol operates over two ports.

Port	Service	Devices	Vulnerab. Level
1900	UPnP	46.2%	HV
80	HTTP	45.7%	V
5353	mDNS	39.2%	HV
8080	HTTP Alt	26.9%	V
443	HTTPS	21.2%	SV
139	SMB	10.6%	V
445	SMB	8.7%	V

checks. Home users could access their personalized IoT portal hosted on a management server through the IoT internet and only from an IoT device. This approach would likely limit intrusions and enable secure centralized management. However, implementing such a dedicated Iot internet would demand financial investment and commitment from governmental and corporate entities. Also, extensive research, development, evaluation, and standardization efforts would be needed.

6.2.6 Security Challenges

Computing devices are primarily based on Microsoft Windows or Apple Mac operating systems; whereby, both Microsoft and Apple have well-established patch management programs to mitigate software bugs and security vulnerabilities. IoT vendors with various applications and hardware components typically do not have well-established patch management programs, leading to a large proportion of devices that lack the latest patches. Due to their smaller footprint, they are also used for various functions. As the number of vendor and device functions increases, so does the number of services that expose the system to newer vulnerabilities. Table 6.1 reveals some of the most common protocols used by IoT devices [10]. It is alarming to see the relatively high percentages of vulnerable protocols, such as UPnP and HTTP.

Universal Plug and Play (UPnP) services are related to device discovery. HTTP is

mostly deployed for device administration. UPnP, in particular, is widely accepted in the IoT domain due to its ability to discover and interact with devices in a multi-vendor enterprise environment without any configurations [18]. HTTP is widely used in the industry. Therefore, HTTP security issues are addressed quickly by patches or replacing them with HTTPS. A more significant concern in the IoT domain is UPnP. Attackers can eavesdrop on sensor communication and can retrieve a wide variety of personnel information. Unauthorized access to cameras can provide criminals with detailed insights in their victims' personal lives. Cameras are considered the most dominating privacy-violating IoT [316]. Exploitation can assist organized criminal groups in using technology for crimes, such as theft, burglary, kidnapping, and blackmail.

The significance of IoT security came to the surface in 2016, when IoT Bots targeted the internet DNS infrastructure and brought down parts of the Internet via a DDoS attack [317]. There is a need for a UTM module specifically designed to counter IoT vulnerabilities, security flaws, and non-compliant protocols. Zero-trust profiling needs to be established for UPnP devices that fall under the high-risk category, and strict UTM controls should be applied for profiled traffic. Data exfiltration and C&C communication over encrypted channels are also significant concerns. UTM services cannot inspect encrypted traffic without decrypting and utilizing additional computational power.

An ongoing security challenge is the validation of STIX-TAXII feeds. While the existing approaches that are suitable for UTM systems, e.g., existing blockchain approaches to validate the STIX-TAXII feeds, the management, validation, and auditing of STIX-TAXII threat feeds continue to pose significant problems. Future STIX-TAXII research needs to develop and evaluate efficient approaches for STIX-TAXII threat feed management, validation, and auditing that are suitable for the limited computational resources in home network UTM systems. Possible avenues for developing such future STIX-TAXII approaches could build on the principle of collaboration [318] or on the principle of proactive defensive actions [319]. Also, rigorous mathematical foundations may provide novel

avenues for efficient STIX-TAXII management [320].

An overarching future research direction for enhancing the security levels provided by a UTM system is to exploit synergies between the different UTM system components, i.e., firewall, IDS, and antibot synergistically cooperate to support and enhance each other. The synergy among UTM system components can come in the form of serial or parallel processing. Serial processing starts with firewall inspection, followed by IDS and antibot inspection. In contrast, parallel processing starts examining traffic by all the inspection engines concurrently. Parallel processing can achieve performance benefits (reduced inspection latency and higher throughput) since an individual UTM system component (inspection engine) does not have to wait for the completion of the processing of other components. In contrast, in serial processing, the IDS and antibot inspection engines may have to wait for the decision of the firewall inspection; once the firewall module allows the traffic, then IDS and antibot inspection are activated and run in parallel. The latency reduction of parallel processing comes at the cost of increased rule management complexity and loss of functionality. The Single Pass architecture [321] introduced by Palo Alto Networks is an enterprise implementation of UTM parallel processing. The Single Pass architecture requires the administrator to attach security profiles (IDS, antibot) to every firewall rule, which can get difficult to manage as the number of rules increases; also, there is no option to configure separate policies for firewall, IPS, and antibot inspection. Hence, there is an urgent need for developing hybrid UTM solutions that strive to achieve the performance benefits of parallel processing, while retaining the low complexity of the management/administrative control of serial processing.

A related direction for exploiting synergies is to foster cooperation between the IDS inspection and the applications that run in a home network. Specifically, future research should explore the integration of application signatures with the IDS engine in UTM systems, so that the IDS policy can automatically be updated and tuned according to the evolving home network systems, services, and applications. This updating is generally nec-

essary since changes of systems, services, and application require changes of the signatures. There have been attempts by the enterprise security vendors to develop an application-integrated IDS solution. For instance, Cisco introduced "Firepower recommendations" [322] to tune IDS profiles according to applications and network infrastructure. However, protocol categorization is a challenge for the Cisco approach as the application engine is not very robust. Hence, many protocols are classified into the "undefined" category. Also, the Cisco solution is geared towards enterprise customers. Palo Alto Networks introduced a cloud-based inspection engine [323] for real-time malware detection. However, the Palo Alto Networks solution is limited to specific protocols and certain attack types.

6.2.7 SSL Inspection

According to the F5 Networks malware report, 80% of internet traffic is encrypted, and approximately 46% of malware was hidden using encryption techniques [324]. It is evident that inspecting encrypted traffic is inevitable. Analyzing encrypted payload requires extensive processing power. Home UTM devices will need to be upgraded to more robust hardware, which could increase the appliance's cost and size. SUTMS NTOP engine can perform basic SSL inspection without decryption; tasks like certificate verification, compliance checks, abnormal SSL payload detection, etc., can be analyzed. Advanced malware variants obfuscate themselves within the payload using encryption techniques and therefore require full packet decryption. Newer avenues of SOHO UTM devices will need to be explored to optimize SSL processing and simplify the encryption/decryption mechanism.

6.2.8 STIX/TAXII Threat Feeds Validation

Threat intelligence feeds are critical for blocking IoCs that come in the form of IPs, domains, and hashes. SUTMS relies on open source feeds like Open Threat Exchange (OTX) [325], Anomali [132] etc. Open-source platforms lack verification, real-time updates, and

targeted adversaries. It will require human intervention to review IoCs geared toward home networks and client machines before applying access control lists. Commercial threat intelligence vendors like Crowdstrike [326], and ThreatConnect [327] extract data from various intelligence sources and validate and recommend tailored profiles according to customer needs. Another limitation is blocking hashes due to resource constraints. The research opens doors for developing open-source Threat intelligence platforms equivalent to commercial feeds and technologies to inspect hashes without overloading UTM devices.

6.2.9 Signature Optimization

Our results show that IPS is the single most resource-intensive process. Optimizing signatures is an integral part of SUTMS. Signatures of applications discovered via NTOP were enabled to reduce any unnecessary processing. However, services utilized by home networks are dynamic and can change quite frequently. There will also be services left uncategorized due to the lack of pattern detection. Application detection engines require constant updates, and there is a need for open-source application detection engines that process, validate, and categorize newer applications as they emerge. Integration of such engines with UTM's will improve device efficiency and zero-day attacks.

6.2.10 SUTMS Usability & Economic Sustainability

One challenge lies in the technical proficiency of home users to set up and oversee a UTM solution. Successful implementation may demand basic networking skills; however, automation can transform it into a user-friendly plug-and-play system. Customization might pose difficulties, as configurations are streamlined to reduce user intervention. An intuitive web front end with fewer checks could enhance the solution's appeal to home users.

Another challenging aspect is ensuring the commercial viability of SUTMS. Despite its minimal cost compared to commercially available solutions, convincing home users to

invest in a security appliance remains a difficult task. The solution's marketability and acceptance within the user community hinge on factors such as awareness and integration with elements like WiFi access points and ISP. WiFi access points are already prevalent in home networks, and integrating UTM with a WiFi access point can serve as a replacement for such standalone access points. Another approach is to delegate UTM services to the ISP to reduce costs and management efforts; however, this may raise privacy concerns.

6.3 Conclusion

In this research, the proposed SUTMS solution was able to achieve 99.99% of accuracy with significant improvements in CPU and memory. Evaluation of individual components exposed IPS as the single most resource-intense process. The firewall engine was also upgraded from traditional access controls list to dynamic STIX/TAXII feeds. It introduced the capability of proactive blocking of bad actors. Integration of the NTOP engine served a dual purpose, flow detection allowed us to identify anomalies, and application awareness assisted in fine-tuning IPS signatures, ultimately improving CPU and memory usage. Running multiple inspection engines efficiently on a single device with limited resources is the biggest challenge for SOHO UTM appliances. SUTMS could address the performance issues without compromising accuracy, and traditional and modern prevention techniques were combined by innovative means to develop a next-generation of home network protection solutions.

In the realm of smart home management, the proposed Smart Home Control and Management System (SCMS) stands out as a streamlined implementation of a device management solution. It utilizes a multifaceted approach by seamlessly integrating security and remote management modules. A significant feature of SCMS is its ability to dynamically include Small Office/Home Office (SOHO) devices based on their risk scores, which plays a crucial role in creating a comprehensive inventory of smart homes and assessing their security

posture. The outcomes achieved align perfectly with the system's expected functionality and performance. The CPU, memory, and system load remain well within manageable limits, ensuring efficient operation even on devices with limited computational power.

The research work conducted opens new avenues for the mitigation, performance, and optimization of threat management solutions. Inspecting encrypted traffic is a challenge. Encryption is a resource-intensive process. Adversaries utilize encryption as a tool for data hiding, malware obfuscation, and C&C communication. Most internet traffic is encrypted, and there is a dire need to address advanced packet inspection challenges associated with encrypted traffic. A unified threat management solution geared towards IoT is another avenue to explore.

References

[1] "Barracuda." (2020), [Online]. Available: https://blog.barracuda.com/2020/03/26/threat-spotlight-coronavirus-related-phishing/..

[2] L. VanHulle, "Cyberattacks are a remote possibility: Dealerships can fight work-from-home vulnerabilities," English, *Automotive News*, vol. 94, no. 6933, p. 14, 2020, Copyright - Copyright 2020 Crain Communications Inc. All Rights Reserved; Last updated - 2022-10-21. [Online]. Available: http://www.ezproxy.dsu.edu:2048/login?url=https://www.ezproxy.dsu.edu:2074/trade-journals/cyberattacks-are-remote-possibility/docview/2402551127/se-2.

[3] Y. Rekhter, B. Moskowitz, D. Karrenberg, G. J. de Groot, and E. Lear, "Address Allocation for Private Internets, RFC 1918," IETF, Tech. Rep., 1996.

[4] A. H. Lashkari, M. M. S. Danesh, and B. Samadi, "A survey on wireless security protocols (WEP, WPA and WPA2/802.11i)," in *Proc. 2nd IEEE Int. Conf. on Computer Science and Inform. Techn.*, 2009, pp. 48–52.

[5] S. Zhong, "Wi-Fi protected access (WPA)," in *Encyclopedia of Wireless Networks*, Springer, Cham, 2020, pp. 1461–1463.

[6] H. B. Duc, S. Pocarovsky, M. Orgon, and M. Koppl, "Penetration testing of WiFi networks secured by WEP and WPA/WPA2 protocols," in *Informatics and Cybernetics in Intelligent Systems: Proc. of 10th Computer Science On-line Conf.*, Springer, Cham, 2021, pp. 571–585.

[7] *IEEE Standard for Information Technology - Telecommunications and Information Exchange Between Systems - Local and Metropolitan Area Networks - Specific Requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, 2007. DOI: 10.1109/IEEESTD.2007.373646.

[8] Y. Liu, Z. Jin, and Y. Wang, "Survey on security scheme and attacking methods of WPA/WPA2," in *Proc. 6th Int. Conf. on Wireless Commun. Netw. and Mobile Comp. (WiCOM)*, 2010, pp. 1–4. DOI: 10.1109/WICOM.2010.5601275.

[9] J. Mbale and K. Mufeti, "Phase teaching model for subnetting IPv4," *Int. J. of Internet Techn. & Sec. Trans.*, vol. 3, no. 1, pp. 1–15, 2011.

[10] D. Kumar, K. Shen, B. Case, *et al.*, "All things considered: An analysis of IoT devices on home networks," in *Proc. 28th USENIX Security Symp.*, 2019, pp. 1169–1185.

[11] J. Bugeja, A. Jacobsson, and P. Davidsson, "On privacy and security challenges in smart connected homes," in *Proc. IEEE Europ. Intelligence and Security Informatics Conf. (EISIC)*, 2016, pp. 172–175.

[12] A. Stasinopoulos, C. Ntantogian, and C. Xenakis, "The weakest link on the network: Exploiting ADSL routers to perform cyber-attacks," in *Proc. IEEE Int. Symp. on Signal Processing and Information Technology*, 2013, pp. 135–139. DOI: 10.1109/ISSPIT.2013.6781868.

[13] A. K. Nag, A. Roy, and D. Dasgupta, "An adaptive approach towards the selection of multi-factor authentication," in *Proc. IEEE Symposium Series on Computational Intelligence*, 2015, pp. 463–472.

[14] N. DeLeon, *Many wireless routers lack basic security protections, consumer reports' testing finds*, [Online].Available:https://www.consumerreports.org/wireless-routers/wireless-routers-lack-basic-security-protections/, [Accessed: 2-Jan-2022], 2021.

[15] E. Montalbano, *Most popular home routers have 'critical' flaws*, [Online].Available:https://threatpost.com/report-most-popular-home-routers-have-critical-flaws/157346/, [Accessed: 5-Feb-2022], 2021.

[16] D. Goodin, *Bizarre attack infects Linksys routers with self-replicating malware*, [Online].Available:https://arstechnica.com/information-technology/2014/02/bizarre-attack-infects-linksys-routers-with-self-replicating-malware/, [Accessed: 5-Feb-2022], 2021.

[17] Z. Cutlip, "SQL injection to MIPS overflows: Rooting SOHO routers," in *Proc. DEF CON 20*, [Accessed: 7-Feb-2022], 2012.

[18] G. Kayas, M. Hossain, J. Payton, and S. M. R. Islam, "An overview of UPnP-based IoT security: Threats, vulnerabilities, and prospective solutions," in *Proc. 11th IEEE Annual Information Technology, Electronics and Mobile Commun. Conf. (IEMCON)*, 2020, pp. 0452–0460. DOI: 10.1109/IEMCON51383.2020.9284885.

[19] D. M. Junior, L. Melo, H. Lu, M. d'Amorim, and A. Prakash, "Beware of the app! On the vulnerability surface of smart devices through their companion apps," *arXiv preprint arXiv:1901.10062*, 2019.

[20] G. Kayas, M. Hossain, J. Payton, and S. R. Islam, "SUPnP: Secure access and service registration for UPnP-enabled internet of things," *IEEE Internet of Things J.*, vol. 8, no. 14, pp. 11 561–11 580, 2021.

[21] T. Bernard, *MiniUPnP project homepage*, [Online].Available:http://miniupnp.free.fr/, [Accessed: 15-Mar-2022], 2017.

[22] B. S. Distribution, *UPnP Devices 1.2. 1 (libupnp)*, [Online].Available:http://upnp.sourceforge.net, [Accessed: 15-Mar-2022], 2000.

[23] E. Lopez-Aguilera, E. Garcia-Villegas, and J. Casademont, "Evaluation of IEEE 802.11 coexistence in WLAN deployments," English, *Wireless Networks*, vol. 25, no. 1, pp. 87–104, Jan. 2019, Last updated - 2019-01-24. [Online]. Available: http://www.ezproxy.dsu.edu:2048/login?url=https://www.proquest.com/scholarly-journals/evaluation-ieee-802-11-coexistence-wlan/docview/213418506/se-2?accountid=27073.

[24] J. Robert, S. Rauh, H. Lieske, and A. Heuberger, "IEEE 802.15 low power wide area network (LPWAN) PHY interference model," in *Proc. IEEE Int. Conf. on Communications (ICC)*, 2018, pp. 1–6. DOI: 10.1109/ICC.2018.8422801.

[25] "Ieee standard for low-rate wireless networks amendment 3: Advanced encryption standard (aes)-256 encryption and security extensions," *IEEE Std 802.15.4y-2021 (Amendment to IEEE Std 802.15.4-2020 as amended by IEEE Std 802.15.4z-2020 and IEEE Std 802.15.4w-2020)*, pp. 1–23, 2021. DOI: 10.1109/IEEESTD.2021.9444766.

[26] D. Law, D. Dove, J. D'Ambrosia, M. Hajduczenia, M. Laubach, and S. Carlson, "Evolution of ethernet standards in the ieee 802.3 working group," *IEEE Communications Magazine*, vol. 51, no. 8, pp. 88–96, 2013. DOI: 10.1109/MCOM.2013.6576344.

[27] T. Zahariadis, K. Pramataris, and N. Zervos, *A Comparison of Competing Broadband In-Home Technologies*. IEE Electronics and Communications Engineering Journal (ECEJ), Aug. 2002.

[28] S. S. I. Samuel, "A review of connectivity challenges in IoT-smart home," in *Proc. 3rd MEC Int. Conf. on Big Data and Smart City (ICBDSC)*, 2016, pp. 1–4. DOI: 10.1109/ICBDSC.2016.7460395.

[29] S. Al-Sarawi, M. Anbar, K. Alieyan, and M. Alzubaidi, "Internet of things (iot) communication protocols: Review," in *2017 8th International Conference on Information Technology (ICIT)*, 2017, pp. 685–690. DOI: 10.1109/ICITECH.2017.8079928.

[30] F. Samie, L. Bauer, and J. Henkel, "IoT technologies for embedded computing," in *Proceedings of the Eleventh IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, Pittsburgh Pennsylvania: ACM, Oct. 2016.

[31] T. Salman, "Internet of things protocols and standards," *M. Of END, Affairs*, 2015.

[32] O. Karahan and K. Berat, "Raspberry pi firewall and intrusion detection system," *Journal of Intelligent Systems: Theory and Applications*, vol. 3, no. 2, pp. 21–24, 2020.

[33] J. E. C. De la Cruz, C. A. R. Goyzueta, and C. D. Cahuana, "Intrusion detection and prevention system for production supervision in small businesses based on raspberry pi and snort," in *2020 IEEE XXVII International Conference on Electronics, Electrical Engineering and Computing (INTERCON)*, 2020, pp. 1–4. DOI: 10.1109/INTERCON50315.2020.9220240.

[34] V. Visoottiviseth, G. Chutaporn, S. Kungvanruttana, and J. Paisarnduangjan, "Piti: Protecting internet of things via intrusion detection system on raspberry pi," in *2020 International Conference on Information and Communication Technology Convergence (ICTC)*, 2020, pp. 75–80. DOI: 10.1109/ICTC49870.2020.9289311.

[35] S. S. Tirumala, N. Nepal, and S. K. Ray, "Raspberry pi-based intelligent cyber defense systems for smes: An exploratory study," in *International Summit Smart City 360°*, Springer, 2022, pp. 3–14.

[36] M. A. F. M. Fauzi and L. M. Abdullah, "Malicious/phishing url detection system in a network with raspberry pi (netbits)," *International Journal on Perceptive and Cognitive Computing*, vol. 8, no. 2, pp. 30–36, 2022.

[37] G. A. J. Saskara, I. M. E. Listarta, G. S. Santyadiputra, and P. Bayu, "Performance of kismet wireless intrusion detection system on raspberry pi," in *IConVET 2021: Proceedings of the 4th International Conference on Vocational Education and Technology, IConVET 2021, 27 November 2021, Singaraja, Bali, Indonesia*, European Alliance for Innovation, 2022, p. 110.

[38] H. Siddharthan and D. Thangavel, "A novel framework approach for intrusion detection based on improved critical feature selection in internet of things networks," *Concurrency and Computation: Practice and Experience*, vol. 35, no. 1, e7445, 2023.

[39] M. M. Alani, "Botstop: Packet-based efficient and explainable iot botnet detection using machine learning," *Computer Communications*, vol. 193, pp. 53–62, 2022.

[40] I. Butun, P. Österberg, and H. Song, "Security of the internet of things: Vulnerabilities, attacks, and countermeasures," *IEEE Commun. Surv. & Tut.*, vol. 22, no. 1, pp. 616–644, 2020. DOI: 10.1109/COMST.2019.2953364.

[41] F. D. Keersmaeker, Y. Cao, G. K. Ndonda, and R. Sadre, "A survey of public iot datasets for network security research," *IEEE Commun. Surv. & Tut., in print*, pp. 1–1, 2023. DOI: 10.1109/COMST.2023.3288942.

[42] S. Kim, K.-J. Park, and C. Lu, "A survey on network security for cyber–physical systems: From threats to resilient design," *IEEE Commun. Surv. & Tut.*, vol. 24, no. 3, pp. 1534–1573, 2022. DOI: 10.1109/COMST.2022.3187531.

[43] I. Makhdoom, M. Abolhasan, J. Lipman, R. P. Liu, and W. Ni, "Anatomy of threats to the Internet of Things," *IEEE Commun. Surv. & Tut.*, vol. 21, no. 2, pp. 1636–1675, 2019. DOI: 10.1109/COMST.2018.2874978.

[44] N. Moustafa, N. Koroniotis, M. Keshk, A. Y. Zomaya, and Z. Tari, "Explainable intrusion detection for cyber defences in the Internet of Things: Opportunities and solutions," *IEEE Commun. Surv. & Tut., in print*, pp. 1–1, 2023. DOI: 10.1109/COMST.2023.3280465.

[45] N. Neshenko, E. Bou-Harb, J. Crichigno, G. Kaddoum, and N. Ghani, "Demystifying IoT security: An exhaustive survey on IoT vulnerabilities and a first empirical look on internet-scale IoT exploitations," *IEEE Commun. Surv. & Tut.*, vol. 21, no. 3, pp. 2702–2733, 2019. DOI: 10.1109/COMST.2019.2910750.

[46] A. K. Sikder, G. Petracca, H. Aksu, T. Jaeger, and A. S. Uluagac, "A survey on sensor-based threats and attacks to smart devices and applications," *IEEE COMST*, vol. 23, no. 2, pp. 1125–1159, 2021. DOI: 10.1109/COMST.2021.3064507.

[47] I. Stellios, P. Kotzanikolaou, M. Psarakis, C. Alcaraz, and J. Lopez, "A survey of IoT-enabled cyberattacks: Assessing attack paths to critical infrastructures and services," *IEEE Commun. Surv. & Tut.*, vol. 20, no. 4, pp. 3453–3495, 2018. DOI: 10.1109/COMST.2018.2855563.

[48] M. Chen, J. Wan, S. Gonzalez, X. Liao, and V. C. Leung, "A survey of recent developments in home M2M networks," *IEEE Commun. Surv. & Tut.*, vol. 16, no. 1, pp. 98–114, 2014. DOI: 10.1109/SURV.2013.110113.00249.

[49] N. Komninos, E. Philippou, and A. Pitsillides, "Survey in smart grid and smart home security: Issues, challenges and countermeasures," *IEEE Commun. Surv. & Tut.*, vol. 16, no. 4, pp. 1933–1954, 2014. DOI: 10.1109/COMST.2014.2320093.

[50] Z. Wang, D. Liu, Y. Sun, *et al.*, "A survey on IoT-enabled home automation systems: Attacks and defenses," *IEEE COMST*, vol. 24, no. 4, pp. 2292–2328, 2022. DOI: 10.1109/COMST.2022.3201557.

[51] E. Bout, V. Loscri, and A. Gallais, "How machine learning changes the nature of cyberattacks on IoT networks: A survey," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 248–279, 2022.

[52] B. Abdul, Z. Maham, X. Liu, A. R. Javed, J. Zunera, and K. Kashif, "A comprehensive survey of AI-enabled phishing attacks detection techniques," *Telecommun. Sys.*, vol. 76, no. 1, pp. 139–154, 2021.

[53] S. Zaman, K. Alhazmi, M. A. Aseeri, *et al.*, "Security threats and artificial intelligence based countermeasures for internet of things networks: A comprehensive survey," *IEEE Access*, vol. 9, pp. 94 668–94 690, 2021.

[54] M. Alicea and I. Alsmadi, "Misconfiguration in firewalls and network access controls: Literature review," *Future Internet*, vol. 13, no. 11, Art. no. 283, 2021.

[55] A. Voronkov, L. H. Iwaya, L. A. Martucci, and S. Lindskog, "Systematic literature review on usability of firewall configuration," *ACM Computing Surveys (CSUR)*, vol. 50, no. 6, pp. 1–35, 2017.

[56] R. W. Anwar, T. Abdullah, and F. Pastore, "Firewall best practices for securing smart healthcare environment: A review," *Applied Sciences*, vol. 11, no. 19, Art. no. 9183, 2021.

[57] S. Applebaum, T. Gaber, and A. Ahmed, "Signature-based and machine-learning-based web application firewalls: A short survey," *Procedia Computer Science*, vol. 189, pp. 359–367, 2021.

[58] H. Albasheer, M. Md Siraj, A. Mubarakali, *et al.*, "Cyber-attack prediction based on network intrusion detection systems for alert correlation techniques: A survey," *Sensors*, vol. 22, no. 4, Art. no. 1494, 2022.

[59] A. Thakkar and R. Lohiya, "A survey on intrusion detection system: Feature selection, model, performance measures, application perspective, challenges, and future research directions," *Artificial Intelligence Review*, vol. 55, no. 1, pp. 453–563, 2022.

[60] L. N. Tidjon, M. Frappier, and A. Mammar, "Intrusion detection systems: A cross-domain overview," *IEEE Commun. Surv. & Tut.*, vol. 21, no. 4, pp. 3639–3681, 2019. DOI: 10.1109/COMST.2019.2922584.

[61] D. J. Weller-Fahy, B. J. Borghetti, and A. A. Sodemann, "A survey of distance and similarity measures used within network intrusion anomaly detection," *IEEE COMST*, vol. 17, no. 1, pp. 70–91, 2015. DOI: 10.1109/COMST.2014.2336610.

[62] J. J. Jinisha and S. Jerine, "Survey on various attacks and intrusion detection mechanisms in wireless sensor networks," *Turkish J. Computer and Mathematics Education*, vol. 12, no. 11, pp. 3694–3704, 2021.

[63] W. Li, W. Meng, and L. F. Kwok, "Surveying trust-based collaborative intrusion detection: State-of-the-art, challenges and future directions," *IEEE Commun. Surveys & Tutorials*, vol. 24, no. 1, pp. 280–305, 2021.

[64] J. Franco, A. Aris, B. Canberk, and A. S. Uluagac, "A survey of honeypots and honeynets for internet of things, industrial internet of things, and cyber-physical systems," *IEEE Commun. Surv. & Tut.*, vol. 23, no. 4, pp. 2351–2383, 2021. DOI: 10.1109/COMST.2021.3106669.

[65] G. Abdelmoumin, J. Whitaker, D. B. Rawat, and A. Rahman, "A survey on data-driven learning for intelligent network intrusion detection systems," *Electronics*, vol. 11, no. 2, Art. no. 213, 2022.

[66] A. Alotaibi and M. A. Rassam, "Adversarial machine learning attacks against intrusion detection systems: A survey on strategies and defense," *Future Internet*, vol. 15, no. 2, 62:1–62:34, 2023.

[67] S. Arisdakessian, O. A. Wahab, A. Mourad, H. Otrok, and M. Guizani, "A survey on IoT intrusion detection: Federated learning, game theory, social psychology, and explainable AI as future directions," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 4059–4092, 2023. DOI: 10.1109/JIOT.2022.3203249.

[68] A. Belenguer, J. Navaridas, and J. A. Pascual, "A review of federated learning in intrusion detection systems for IoT," *arXiv preprint arXiv:2204.12443*, 2022.

[69] N. Chaabouni, M. Mosbah, A. Zemmari, C. Sauvignac, and P. Faruki, "Network intrusion detection for IoT security based on learning techniques," *IEEE COMST*, vol. 21, no. 3, pp. 2671–2701, 2019. DOI: 10.1109/COMST.2019.2896380.

[70] K. He, D. D. Kim, and M. R. Asghar, "Adversarial machine learning for network intrusion detection systems: A comprehensive survey," *IEEE Commun. Surveys & Tutorials*, vol. 25, no. 1, pp. 538–566, 2023. DOI: 10.1109/COMST.2022.3233793.

[71] H. Kheddar, Y. Himeur, and A. I. Awad, "Deep transfer learning applications in intrusion detection systems: A comprehensive review," *arXiv preprint arXiv:2304.10550*, 2023.

[72] G Logeswari, S Bose, and T Anitha, "An intrusion detection system for SDN using machine learning," *Intelligent Automation & Soft Computing*, vol. 35, no. 1, pp. 867–880, 2023.

[73] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE COMST*, vol. 21, no. 1, pp. 686–728, 2019. DOI: 10.1109/COMST.2018.2847722.

[74] S. Santhosh Kumar, M Selvi, and A Kannan, "A comprehensive survey on machine learning-based intrusion detection systems for secure communication in internet of things," *Computational Intelligence and Neuroscience*, vol. 2023, Article ID 8981988, 2023.

[75] A. Thakkar and R. Lohiya, "A review on challenges and future research directions for machine learning-based intrusion detection system," *Archives of Computational Methods in Engineering*, pp. 1–25, 2023.

[76] Z. Yang, X. Liu, T. Li, *et al.*, "A systematic literature review of methods and datasets for anomaly-based network intrusion detection," *Computers & Security*, p. 102 675, 2022.

[77] L. Yi, M. Yin, and M. Darbandi, "A deep and systematic review of the intrusion detection systems in the fog environment," *Trans. on Emerging Telecommun. Techn.*, vol. 34, no. 1, Art. no. e4632, 2023.

[78] A. K. Tyagi and G Aghila, "A wide scale survey on botnet," *Int. J. of Computer Applications*, vol. 34, no. 9, pp. 10–23, 2011.

[79] I. Ullah, N. Khan, and H. A. Aboalsamh, "Survey on botnet: Its architecture, detection, prevention and mitigation," in *Proc. 10th IEEE Int. Conf. on Netw., Sensing and Contr. (ICNSC)*, 2013, pp. 660–665.

[80] M. J. Elhalabi, S. Manickam, L. B. Melhim, M. Anbar, and H. Alhalabi, "A review of peer-to-peer botnet detection techniques," *Journal of Computer Science*, vol. 10, no. 1, pp. 169–177, 2014.

[81] M. Latah, "Detection of malicious social bots: A survey and a refined taxonomy," *Expert Systems with Applications*, vol. 151, Art. no. 113383, 2020.

[82] J. Gharat, A. Vidhate, and A. Barve, "Performance analysis of Unified Threat Management (UTM)," in *Inventive Communication and Computational Technologies*, Springer, Singapore, 2020, pp. 121–133.

[83] Y. Qi, B. Yang, B. Xu, and J. Li, "Towards system-level optimization for high performance unified threat management," in *Proc. Int. Conf. on Networking and Services (ICNS'07)*, 2007, pp. 1–6.

[84] J. Liang and Y. Kim, "Evolution of firewalls: Toward securer network using next generation firewall," in *Proc. IEEE 12th Ann. Computing and Commun. Workshop and Conf. (CCWC)*, 2022, pp. 752–759.

[85] N. Miloslavskaya, "A brief evolution of network protection tools and methods," *Procedia Computer Science*, vol. 190, pp. 590–596, 2021.

[86] A. Kapoor, A. Gupta, R. Gupta, S. Tanwar, G. Sharma, and I. E. Davidson, "Ransomware detection, avoidance, and mitigation scheme: A review and future directions," *Sustainability*, vol. 14, no. 1, 8:1–8:24, 2021.

[87] D. Christopoulos, *Survey of unified threat management appliances, Master Thesis, Kingston Univ., London, UK, School of Computing and Information Systems*, 2011.

[88] H. Al-Aqrabi, R. Hill, P. Lane, and H. Aagela, "Securing manufacturing intelligence for the industrial internet of things," in *Proc. Fourth Int. Congress on Information and Commun. Techn.*, 2020, pp. 267–282.

[89] A. Sharma, V Chintala, and S. Kumar, "Feasibility analysis of photovoltaic (PV) grid tied system for Indian military station considering economic & grid cyber-security aspects," in *Recent Advances in Mechanical Infrastructure: Proceedings of ICRAM 2019*, Springer, Singapore, 2020, pp. 153–162.

[90] K. Latesh Kumar and H. Leena, "NGEN firewall security augmentation using Brooks-Iyengar and random forest classfier method: By predicting cyber threats from: Darkweb/deepweb data," *International Journal of Next-Generation Computing*, vol. 11, no. 1, pp. 1–19, 2020.

[91] S. Jose, D Malathi, B. Reddy, and D. Jayaseeli, "A survey on anomaly based host intrusion detection system," in *Journal of Physics: Conference Series*, IOP Publishing, vol. 1000, 2018, Paper no. 012049.

[92] D. L. Nazareth, J. Choi, and T. L. Ngo-Ye, "The security-as-a-service market for small and medium enterprises," *Journal of Computer Information Systems*, vol. 62, no. 5, pp. 954–964, 2022.

[93] O. Yurekten and M. Demirci, "SDN-based cyber defense: A survey," *Future Generation Computer Systems*, vol. 115, pp. 126–149, 2021.

[94] T. Akiyama, A. Haruta, K. Tamai, T. Okada, H. Yamaoka, and H. Masuda, "Building a secure network during the COVID-19," in *Proc. ACM SIGUCCS Annual Conference*, 2022, pp. 16–20.

[95] R. J. Wieringa, *Design science methodology for information systems and software engineering*. Springer, 2014.

[96] A. Hevner and S. Chatterjee, "Design science research in information systems," in *Integrated Series in Information Systems*, Boston, MA: Springer US, 2010, pp. 9–22.

[97] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee, "A design science research methodology for information systems research," *Journal of management information systems*, vol. 24, no. 3, pp. 45–77, 2007.

[98] https://oregon.comcast.com/2020/08/14/comcast-internet-essentials-enhancements-meet-critical-needs-of-local-families/, Accessed: 2023-2-9.

[99] L. Lucas, "Alibaba sales rise on chinese ecommerce boom," English, *FT.com*, 2017, Name - Securities & Exchange Commission; Copyright - Copyright The Financial Times Limited Aug 17, 2017; Last updated - 2022-10-23; SubjectsTermNotLitGenreText - Beijing China; United States–US; China. [Online]. Available: http://www.ezproxy.dsu.edu:2048/login?url=https://www.proquest.com/trade-journals/alibaba-sales-rise-on-chinese-ecommerce-boom/docview/1939383332/se-2.

[100] Research and Markets, *Cloud computing industry to grow from* 371.4*billionin*2020*to832.1 billion by 2025, at a cagr of 17.5%*, 2020. [Online]. Available: https://www.globenewswire.com/news-release/2020/08/21/2081841/0/en/Cloud-Computing-Industry-to-Grow-from-371-4-Billion-in-2020-to-832-1-Billion-by-2025-at-a-CAGR-of-17-5.html.

[101] "Will the rise of remote work become a permanent pandemic legacy?" English, *The Public Record*, vol. 48, no. 21, pp. 1–1,3, 2021, Copyright - Copyright Desert Publication, Inc. and Sharon Apfelbaum Mar 16, 2021; Last updated - 2022-10-18. [Online]. Available: http://www.ezproxy.dsu.edu:2048/login?url=https://www.proquest.com/magazines/will-rise-remote-work-become-permanent-pandemic/docview/2512305099/se-2.

[102] F. Dahlqvist, M. Patel, A. Rajko, and J. Shulman, *Growing opportunities in the internet of things*, 2022. [Online]. Available: https://www.mckinsey.com/industries/private-equity-and-principal-investors/our-insights/growing-opportunities-in-the-internet-of-things#.

[103] "Raspberry pi 4 model specifications." (2021), [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/.

[104] I. Dubrawsky, "Firewall evolution-deep packet inspection," *Security Focus*, vol. 29, pp. 1–5, 2003, available from https://www.academia.edu/6937224/Firewall_evolution_deep_packet

[105] W. Goralski, "Firewalls," in *The Illustrated Network*, Morgan Kaufmann, Cambridge, MA, 2017, pp. 799–812.

[106] Z. Trabelsi, S. Zeidan, K. Shuaib, and K. Salah, "Improved session table architecture for denial of stateful firewall attacks," *IEEE Access*, vol. 6, pp. 35 528–35 543, 2018.

[107] P Ayuso, "Netfilter's connection tracking system," *LOGIN: The USENIX Magazine*, vol. 31, pp. 34–39, 2006.

[108] Z. Trabelsi and S. Zeidan, "Multilevel early packet filtering technique based on traffic statistics and splay trees for firewall performance improvement," in *Proc. IEEE ICC*, 2012, pp. 1074–1078.

[109] D. D. Sleator and R. E. Tarjan, "Self-adjusting binary search trees," *Journal of the ACM (JACM)*, vol. 32, no. 3, pp. 652–686, 1985.

[110] Y. Xing, F. L. Wong, and A. Kumar, "Lightweight bare-metal stateful firewall," in *Proc. IEEE 20th Pacific Rim Int. Symp. on Dependable Computing*, 2014, pp. 53–58.

[111] D. Melkov, A. Šaltis, and Š. Paulikas, "Performance testing of Linux firewalls," in *Proc. IEEE Open Conf. of Electrical, Electronic and Information Sciences (eStream)*, 2020, pp. 1–4.

[112] G. N. Purdy, *Linux iptables Pocket Reference: Firewalls, NAT & Accounting*. O'Reilly Media, Inc.Sebastopol, CA, 2004.

[113] S. Suehring, *Linux firewalls: Enhancing security with nftables and beyond*. Addison-Wesley Professional, Boston, MA, 2015.

[114] C. Diekmann, L. Hupel, J. Michaelis, M. Haslbeck, and G. Carle, "Verified iptables firewall analysis and verification," *Journal of Automated Reasoning*, vol. 61, pp. 191–242, 2018.

[115] P. Likhar and R. Shankar Yadav, "Impacts of replace venerable iptables and embrace nftables in a new futuristic Linux firewall framework," in *Proc. 5th Int. Conf. on Computing Methodologies and Communication (ICCMC)*, 2021, pp. 1735–1742. DOI: 10.1109/ICCMC51019.2021.9418298.

[116] S. Miano, M. Bertrone, F. Risso, M. V. Bernal, Y. Lu, and J. Pi, "Securing Linux with a faster and scalable iptables," *ACM SIGCOMM Computer Commun. Rev.*, vol. 49, no. 3, pp. 2–17, 2019.

[117] A. Deepak, R. Huang, and P. Mehra, "eBPF/XDP based firewall and packet filtering," in *Linux Plumbers Conference*, 2018, pp. 1–5.

[118] J. Nam, S. Lee, P. Porras, V. Yegneswaran, and S. Shin, "Secure inter-container communications using XDP/eBPF," *IEEE/ACM Trans. on Networking*, vol. 31, no. 2, pp. 934–947, 2022.

[119] D. Scholz, D. Raumer, P. Emmerich, A. Kurtz, K. Lesiak, and G. Carle, "Performance implications of packet filtering with Linux eBPF," in *Proc. IEEE 30th Int. Teletraffic Congr. (ITC 30)*, vol. 1, 2018, pp. 209–217.

[120] M. A. Vieira, M. S. Castanho, R. D. Pacífico, E. R. Santos, E. P. C. Júnior, and L. F. Vieira, "Fast packet processing with eBPF and XDP: Concepts, code, challenges, and applications," *ACM Computing Surveys (CSUR)*, vol. 53, no. 1, 16:1–16:36, 2020.

[121] Z. Xiang, F. Gabriel, E. Urbano, G. T. Nguyen, M. Reisslein, and F. H. Fitzek, "Reducing latency in virtual machines: Enabling tactile internet for human-machine co-working," *IEEE JSAC*, vol. 37, no. 5, pp. 1098–1116, 2019.

[122] Z. Xiang, M. Höweler, D. You, M. Reisslein, and F. H. Fitzek, "X-MAN: A non-intrusive power manager for energy-adaptive cloud-native network functions," *IEEE TNSM*, vol. 19, no. 2, pp. 1017–1035, 2021.

[123] L. Teng, C.-H. Hung, and C. H.-P. Wen, "P4SF: A high-performance stateful firewall on commodity P4-programmable switch," in *Proc. IEEE/IFIP Network Operations and Management Symp*, 2022, pp. 1–5.

[124] B. P. Kavin, S. Srividhya, and W.-C. Lai, "Performance evaluation of stateful firewall-enabled SDN with flow-based scheduling for distributed controllers," *Electr.*, vol. 11, no. 19, Art. no. 3000, 2022.

[125] S. Prabakaran, R. Ramar, I. Hussain, *et al.*, "Predicting attack pattern via machine learning by exploiting stateful firewall as virtual network function in an SDN network," *Sensors*, vol. 22, no. 3, Art. no. 709, 2022.

[126] Y Ariyanto, B Harijanto, V. Firdaus, and S. Arief, "Performance analysis of Proxmox VE firewall for network security in cloud computing server implementation," in *IOP Conference Series: Materials Science and Engineering*, vol. 732, 2020, p. 012 081.

[127] "Webmin management tool." (2022), [Online]. Available: `https://www.webmin.com/`.

[128] "Ubuntu iptables persistent package." (2022), [Online]. Available: `https://packages.ubuntu.com/bionic/admin/iptables-persistent`.

[129] "Linux firewall." (2005), [Online]. Available: https://www.linuxtopia.org/Linux_Firewall_iptables/x4983.html..

[130] M Apoorva, R. Eswarawaka, and P Reddy, "A latest comprehensive study on structured threat information expression (stix) and trusted automated exchange of indicator information (taxii)," pp. 477–482, 2017.

[131] "Mitre org." (2012), [Online]. Available: https://www.mitre.org/sites/default/files/publications/stix.pdf.

[132] "Anomali threat feeds." (2012), [Online]. Available: https://www.anomali.com/resources/what-are-stix-taxii.

[133] "Cisa shared feeds." (2015), [Online]. Available: https://www.cisa.gov/ais..

[134] P. Gray, "https://www.computerweekly.com/news/2240019669/Meet-Martin-Roesch-Creator-of-Snort", "Accessed: 2022-7-4".

[135] *Our story*, en, https://suricata.io/our-story/., Accessed: 2022-7-4, Sep. 2012.

[136] *About Zeek — Book of Zeek*, en, https://docs.zeek.org/en/master/about.html, Accessed: 2022-7-17.

[137] S. Jin, J.-G. Chung, and Y. Xu, "Signature-based intrusion detection system (IDS) for in-vehicle can bus network," in *Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS)*, 2021, pp. 1–5.

[138] Q. Hu, S.-Y. Yu, and M. R. Asghar, "Analysing performance issues of open-source intrusion detection systems in high-speed networks," *J. Inform. Sec. and Appl.*, vol. 51, Art. no. 102426, 2020.

[139] https://www.snort.org/., Accessed: 2022-7-4.

[140] R. Chi, "Intrusion detection system based on Snort," in *Proc. 9th Int. Symp. on Linear Drives for Industry Applications, Volume 3*, Springer Berlin Heidelberg, 2014, pp. 657–664.

[141] V. Bontupalli, C. Yakopcic, R. Hasan, and T. M. Taha, "Efficient memristor-based architecture for intrusion detection and high-speed packet classification," *ACM Journal on Emerging Technologies in Computing Systems (JETC)*, vol. 14, no. 4, pp. 1–27, 2018.

[142] M. R. Zalbina and D. Stiawan, *HTTP Attack Detection System Based on HTTP Inspect Preprocessor and Rule Options*, Available on academia.edu, last accessed April 10, 2023.

[143] A. Kumar, S. Chandak, and R. Dewanjee, "Recent advances in intrusion detection systems: An analytical evaluation and comparative study," *Int. Journal of Computer Applications*, vol. 86, no. 4, pp. 32–37, 2014.

[144] J. E. C. De la Cruz, C. A. R. Goyzueta, and C. D. Cahuana, "Intrusion detection and prevention system for production supervision in small businesses based on Raspberry Pi and Snort," in *Proc. IEEE Int. Conf. on Electron., Electr. Eng. & Comp. (INTERCON)*, 2020, pp. 1–4.

[145] W. Park and S. Ahn, "Performance comparison and detection analysis in Snort and Suricata environment," *Wireless Personal Commun.*, vol. 94, no. 2, pp. 241–252, 2017.

[146] *10.1. Suricata.Yaml — Suricata 6.0.4 Documentation*, en, `https://suricata.readthedocs.io/en/suricata-6.0.4/configuration/suricata-yaml.html?highlight=thread`, Accessed: 2022-7-4.

[147] C. Hoover, *Comparative study of Snort 3 and Suricata intrusion detection systems, computer science and computer engineering undergraduate honors theses, university of arkansas, fayetteville*, 2022.

[148] `https://docs.mirantis.com/mcp/q4-18/mcp-security-best-practices/use-cases/idps-vnf/ips-mode/nfq.html`, Accessed: 2022-7-17.

[149] T. Zitta, M. Neruda, L. Vojtech, *et al.*, "Penetration testing of intrusion detection and prevention system in low-performance embedded iot device," in *Proc. 18th Int. Conf. on Mechatronics – Mechatronika (ME)*, 2018, pp. 1–5.

[150] K. Thongkanchorn, S. Ngamsuriyaroj, and V. Visoottiviseth, "Evaluation studies of three intrusion detection systems under various attacks and rule sets," in *Proc. IEEE Int. Conf. of IEEE Region 10 (TENCON)*, 2013, pp. 1–4.

[151] A. Alshamrani, S. Myneni, A. Chowdhary, and D. Huang, "A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities," *IEEE Commun. Surv. & Tut.*, vol. 21, no. 2, pp. 1851–1877, 2019. DOI: `10.1109/COMST.2019.2891891`.

[152] C. Callegari, S. Giordano, and M. Pagano, "Anomaly detection: An overview of selected methods," in *Proc. Int. Multi-Conference on Eng., Computer and Inform. Sciences (SIBIRCON)*, 2017, pp. 52–57. DOI: `10.1109/SIBIRCON.2017.8109836`.

[153] I. Dutt, S. Borah, and I. K. Maitra, "Immune system based intrusion detection system (IS-IDS): A proposed model," *IEEE Access*, vol. 8, pp. 34 929–34 941, 2020.

[154] R. Samrin and D Vasumathi, "Review on anomaly based network intrusion detection system," in *Proc. IEEE Int. Conf. on Electrical, Electronics, Communication, Computer, and Optimization Techniques (ICEECCOT)*, 2017, pp. 141–147.

[155] Kunal and M. Dua, "Machine learning approach to IDS: A comprehensive review," in *Proc. IEEE 3rd Int. Conf. on Electronics, Commun. and Aerospace Techn. (ICECA)*, 2019, pp. 117–121.

[156] A. Khraisat, I. Gondal, P. Vamplew, and J. Kamruzzaman, "Survey of intrusion detection systems: Techniques, datasets and challenges," *Cybersecurity*, vol. 2, no. 1, pp. 1–22, Dec. 2019.

[157] N. Ye, S. Emran, Q. Chen, and S. Vilbert, "Multivariate statistical analysis of audit trails for host-based intrusion detection," *IEEE Transactions on Computers*, vol. 51, no. 7, pp. 810–820, 2002. DOI: 10.1109/TC.2002.1017701.

[158] L. K. Hotta, E. C. Lucas, and H. P. Palaro, "Estimation of VaR using copula and extreme value theory," *Multinational Finance Journal*, vol. 12, no. 3/4, pp. 205–218, 2008.

[159] L. Xue and Z. Hu, "Research of worm intrusion detection algorithm based on statistical classification technology," in *Proc. 8th Int. Symp. on Comput. Intellig. and Design (ISCID)*, vol. 1, 2015, pp. 413–416. DOI: 10.1109/ISCID.2015.215.

[160] S. A. P. Kumar, A. Kumar, and S Srinivasan, "Statistical based intrusion detection framework using six sigma technique," *IJCSNS Int. J. Computer Science and Netw. Sec.*, vol. 7, no. 10, pp. 333–342, 2007.

[161] J. Tao, H. Lin, and C. Liu, "IDSV: Intrusion detection algorithm based on statistics variance method in user transmission behavior," in *Proc. Int. Conf. on Comput. and Inform. Sciences*, 2010, pp. 1182–1185. DOI: 10.1109/ICCIS.2010.292.

[162] S. Rastegari, C.-P. Lam, and P. Hingston, "A statistical rule learning approach to network intrusion detection," in *Proc 5th Int. Conf. on IT Convergence and Security (ICITCS)*, 2015, pp. 1–5. DOI: 10.1109/ICITCS.2015.7292933.

[163] K. K. Ghanshala, P. Mishra, R. C. Joshi, and S. Sharma, "BNID: A behavior-based network intrusion detection at network-layer in cloud environment," in *Proc. First Int. Conf. on Secure Cyber Computing and Commun. (ICSCCC)*, 2018, pp. 100–105. DOI: 10.1109/ICSCCC.2018.8703265.

[164] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouet, "Netspot: A simple intrusion detection system with statistical learning," in *Proc. IEEE 19th Int. Conf. on Trust, Security and Privacy in Computing and Communications (TrustCom)*, 2020, pp. 911–918. DOI: 10.1109/TrustCom50675.2020.00122.

[165] N. A. Carreón, A. Gilbreath, and R. Lysecky, "Statistical time-based intrusion detection in embedded systems," in *Proc. Design, Automation & Test in Europe Conf. & Exhibition (DATE)*, 2020, pp. 562–567. DOI: 10.23919/DATE48585.2020.9116369.

[166] T. Das, O. A. Hamdan, R. M. Shukla, S. Sengupta, and E. Arslan, "UNR-IDD: Intrusion detection dataset using network port statistics," in *Proc. IEEE 20th Consumer Communications & Networking Conf. (CCNC)*, 2023, pp. 497–500.

[167] S.-Y. Kuo, F.-H. Tseng, and Y.-H. Chou, "Metaverse intrusion detection of wormhole attacks based on a novel statistical mechanism," *Future Generation Computer Systems*, vol. 143, pp. 179–190, 2023.

[168] K Sasikala and S Vasuhi, "Anomaly based intrusion detection on IOT devices using logistic regression," in *Proc. Int. Conf. on Netw. and Commun. (ICNWC)*, 2023, pp. 1–5.

[169] A. Siffer, P.-A. Fouque, A. Termier, and C. Largouet, "Anomaly detection in streams with extreme value theory," in *Proc. ACM SIGKDD Int. Conf. on Knowl. Disc, and Data Mining*, 2017, pp. 1067–1075.

[170] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," *arXiv preprint arXiv:1802.09089*, 2018.

[171] H Hotelling, "Multivariate quality control," in *Techniques of Statistical Analysis*, C Eisenhart, M. W. Hastay, and W. A. Wallis, Eds., New York: McGraw-Hill, 1947.

[172] K EI-Fakih, N Yevtushenko, and G. Bochmann, "FSM-based incremental conformance testing methods," *IEEE Trans. Softw. Eng.*, vol. 30, no. 7, pp. 425–436, Jul. 2004.

[173] Y. Qi, J. Zhong, R. Jiang, *et al.*, "FSM-based cyber security status analysis method," in *Proc. IEEE Fourth Int. Conf. on Data Science in Cyberspace (DSC)*, 2019, pp. 510–515. DOI: 10.1109/DSC.2019.00083.

[174] I. Vacas, I. Medeiros, and N. Neves, "Detecting network threats using OSINT knowledge-based IDS," in *Proc. 14th European Dependable Computing Conf. (EDCC)*, 2018, pp. 128–135. DOI: 10.1109/EDCC.2018.00031.

[175] L. Bouzar-Benlabiod, L. Meziani, A. Chebieb, N.-E. Rim, and Z. Mellal, "Experts' knowledge merging to reduce IDS alerts number," in *Proc. Int. Conf. on Collab. Techn. and Sys. (CTS)*, 2016, pp. 418–423. DOI: 10.1109/CTS.2016.0080.

[176] G. Olimpio, P. F. C. Silva, L. Camargos, R. S. Miani, and E. R. de Faria, "Intrusion detection over network packets using data stream classification algorithms," in *Proc. IEEE 33rd Int. Conf. on Tools with Artificial Intelligence (ICTAI)*, 2021, pp. 985–990. DOI: 10.1109/ICTAI52525.2021.00157.

[177] B. Li, Y. Wang, M. Liu, *et al.*, "FDEN: Mining effective information of features in detecting network anomalies," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Proc. (ICASSP)*, 2021, pp. 8553–8557. DOI: 10.1109/ICASSP39728.2021.9415099.

[178] S. M. Arikan and S. Acar, "A data mining based system for automating creation of cyber threat intelligence," in *Proc. 9th Int. Symp. on Digital Forensics and Security (ISDFS)*, 2021, pp. 1–7. DOI: 10.1109/ISDFS52919.2021.9486335.

[179] S. Dua and X. Du, *Data Mining and Machine Learning in Cybersecurity*. CRC Press, Boca Raton, FL, 2016.

[180] U. Aparna and S. Paul, "Feature selection and extraction in data mining," in *Proc. Online Int. Conf. on Green Eng. and Techn. (IC-GET)*, 2016, pp. 1–3. DOI: 10.1109/GET.2016.7916845.

[181] D. A. A. Gnana, S. A. A. Balamurugan, and E. J. Leavline, "Literature review on feature selection methods for high-dimensional data," *Int. J. Computer Applications*, vol. 136, no. 1, pp. 9–17, 2016.

[182] M. Dash and H. Liu, "Feature selection for classification," *Intelligent Data Analysis*, vol. 1, no. 1, pp. 131–156, 1997, ISSN: 1088-467X. DOI: https://doi.org/10.1016/S1088-467X(97)00008-5.

[183] J. Li, H. Zhang, J. Zhao, X. Guo, W. Rihan, and G. Deng, "Embedded feature selection and machine learning methods for flash flood susceptibility-mapping in the mainstream Songhua River Basin, China," *Remote Sensing*, vol. 14, no. 21, p. 5523, 2022.

[184] S. Tabakhi, P. Moradi, and F. Akhlaghian Tab, "An unsupervised feature selection algorithm based on ant colony optimization," *Engineering Applications of Artificial Intelligence*, vol. 32, 112–123, 2014. DOI: 10.1016/j.engappai.2014.03.007.

[185] T. Zhang and W. Chen, "LMD based features for the automatic seizure detection of EEG signals using SVM," *IEEE Trans. on Neural Systems and Rehabilitation Engineering*, vol. 25, no. 8, pp. 1100–1108, 2016.

[186] H. Liu and B. Lang, "Machine learning and deep learning methods for intrusion detection systems: A survey," *Applied Sciences*, vol. 9, no. 20, Art. no. 4396, 2019.

[187] *KDD99 Dataset*, `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html`, Accessed: 2022-7-6.

[188] *NSL-KDD Dataset*, en, `https://www.unb.ca/cic/datasets/nsl.html`, Accessed: 2022-7-6.

[189] V. K. Chauhan, K. Dahiya, and A. Sharma, "Problem formulations and solvers in linear SVM: A review," *Artificial Intelligence Review*, vol. 52, no. 2, pp. 803–855, 2019.

[190] C. Zhang, G. Zhang, and S. Sun, "A mixed unsupervised clustering-based intrusion detection model," in *Proc. Third Int. Conf. on Genetic and Evolutionary Comp.*, 2009, pp. 426–428. DOI: 10.1109/WGEC.2009.72.

[191] S. T. F. Al-Janabi and H. A. Saeed, "A neural network based anomaly intrusion detection system," in *Proc. IEEE Developments in E-systems Engineering*, 2011, pp. 221–226.

[192] D. Damopoulos, S. A. Menesidou, G. Kambourakis, M. Papadaki, N. Clarke, and S. Gritzalis, "Evaluation of anomaly-based IDS for mobile devices using machine learning classifiers," *Security and Communication Networks*, vol. 5, no. 1, pp. 3–14, 2012.

[193] O. Alomari and Z. A. Othman, "Bees algorithm for feature selection in network anomaly detection," *Journal of Applied Sciences Research*, vol. 8, no. 3, pp. 1748–1756, 2012.

[194] M. Al Mehedi Hasan, M. Nasser, and B. Pal, "On the KDD'99 dataset: Support vector machine based intrusion detection system (IDS) with different kernels," *Int. J. Electron. Commun. Comput. Eng*, vol. 4, no. 4, pp. 1164–1170, 2013.

[195] B. Kasliwal, S. Bhatia, S. Saini, I. S. Thaseen, and C. A. Kumar, "A hybrid anomaly detection model using G-LDA," in *Prc. IEEE Int. Advance Computing Conf. (IACC)*, 2014, pp. 288–293.

[196] M. Zhang, B. Xu, and J. Gong, "An anomaly detection model based on one-class SVM to detect network intrusions," in *Proc. 11th IEEE Int. Conf. on Mobile Ad-hoc & Sensor Netw. (MSN)*, 2015, pp. 102–107.

[197] A. A. Aburomman and M. Bin Ibne Reaz, "Ensemble of binary SVM classifiers based on PCA and LDA feature extraction for intrusion detection," in *Proc. IEEE*

Adv. Inform. Management, Commun., Electronic and Autom. Control Conf. (IMCEC), 2016, pp. 636–640. DOI: 10.1109/IMCEC.2016.7867287.

[198] A. Jaiswal, A. S. Manjunatha, B. Madhu, and M. P. Chidananda, "Predicting unlabeled traffic for intrusion detection using semi-supervised machine learning," in *Proc. Int. Conf. on Electric., Electron., Commun., Computer and Optim. Techn. (ICEECCOT)*, 2016, pp. 218–222. DOI: 10.1109/ICEECCOT.2016.7955218.

[199] I. S. Thaseen and C. A. Kumar, "Intrusion detection model using fusion of chi-square feature selection and multi class SVM," *J. King Saud Univ.-Computer and Inform. Sci.*, vol. 29, no. 4, pp. 462–472, 2017.

[200] Y. Chang, W. Li, and Z. Yang, "Network intrusion detection based on random forest and support vector machine," in *Proc IEEE Int. Conf. on Computational Science and Eng. (CSE) and IEEE Int. Conf. on Embedded and Ubiquitous Comp. (EUC)*, vol. 1, 2017, pp. 635–638. DOI: 10.1109/CSE-EUC.2017.118.

[201] S. N. Mighan and M. Kahani, "Deep learning based latent feature extraction for intrusion detection," in *Proc. IEEE Iranian Conference on Electrical Engineering (ICEE)*, 2018, pp. 1511–1516.

[202] A. Shenfield, D. Day, and A. Ayesh, "Intelligent intrusion detection systems using artificial neural networks," *ICT Express*, vol. 4, no. 2, pp. 95–99, 2018.

[203] M. Belouch, S. El Hadaj, and M. Idhammad, "Performance evaluation of intrusion detection based on machine learning using Apache Spark," *Procedia Computer Science*, vol. 127, pp. 1–6, 2018.

[204] S. Mohammadi, H. Mirvaziri, M. Ghazizadeh-Ahsaee, and H. Karimipour, "Cyber intrusion detection by combined feature selection algorithm," *J. Inform. Security and Appl.*, vol. 44, pp. 80–88, 2019.

[205] P. Maniriho, E. Niyigaba, Z. Bizimana, V. Twiringiyimana, L. J. Mahoro, and T. Ahmad, "Anomaly-based intrusion detection approach for IoT networks using machine learning," in *Proc. Int. Conf. on Comp. Eng., Netw., and Intellig. Multim. (CENIM)*, 2020, pp. 303–308.

[206] M. A. Kabir and X. Luo, "Unsupervised learning for network flow based anomaly detection in the era of deep learning," in *Proc. IEEE Sixth Int. Conf. on Big Data Computing Service and Appl. (BigDataService)*, 2020, pp. 165–168. DOI: 10.1109/BigDataService49289.2020.00032.

[207] J. Chen, S. Yin, S. Cai, L. Zhao, and S. Wang, "L-KPCA: An efficient feature extraction method for network intrusion detection," in *Proc. 17th Int. Conf. on Mob., Sensing & Netw. (MSN)*, 2021, pp. 683–684.

[208] S. Amaran and R. M. Mohan, "Intrusion detection system using optimal support vector machine for wireless sensor networks," in *Proc. Int. Conf. on Artif. Intellig. and Smart Sys. (ICAIS)*, 2021, pp. 1100–1104. DOI: 10.1109/ICAIS50930.2021.9395919.

[209] M. H. Tarek, M. M. H. U. Mazumder, S. Sharmin, M. S. Islam, M. Shoyaib, and M. M. Alam, "RHC: Cluster based feature reduction for network intrusion detections," in *Proc. IEEE 19th Annual Consumer Commun. & Netw. Conf. (CCNC)*, 2022, pp. 378–384.

[210] C. Chen, X. Xu, G. Wang, and L. Yang, "Network intrusion detection model based on neural network feature extraction and PSO-SVM," in *Proc. 7th Int. Conf. on Intelligent Computing and Signal Processing (ICSP)*, 2022, pp. 1462–1465.

[211] A. Fatani, A. Dahou, M. A. Al-Qaness, S. Lu, and M. A. Elaziz, "Advanced feature extraction and selection approach using deep learning and aquila optimizer for IoT intrusion detection system," *Sensors*, vol. 22, no. 1, Art. no. 140, 2021.

[212] A. Cholakoska, H. Gjoreski, V. Rakovic, *et al.*, "Federated learning for network intrusion detection in ambient assisted living environments," *IEEE Internet Computing, in print*, pp. 1–9, 2023. DOI: 10.1109/MIC.2023.3264700.

[213] O. Elnakib, E. Shaaban, M. Mahmoud, and K. Emara, "EIDM: Deep learning model for IoT intrusion detection systems," *The Journal of Supercomputing*, vol. 79, 13241–13261, 2023.

[214] I. Hidayat, M. Z. Ali, and A. Arshad, "Machine learning-based intrusion detection system: An experimental comparison," *J. Computational and Cognitive Engineering*, vol. 2, no. 2, pp. 88–97, 2023.

[215] M. Mohy-eddine, A. Guezzaz, S. Benkirane, and M. Azrour, "An efficient network intrusion detection model for iot security using K-NN classifier and feature selection," *Multimedia Tools and Applications*, vol. 82, 23615–23633, 2023.

[216] D. Rani, N. S. Gill, P. Gulia, F. Arena, and G. Pau, "Design of an intrusion detection model for IoT-enabled smart home," *IEEE Access*, vol. 11, pp. 52 509–52 526, 2023. DOI: 10.1109/ACCESS.2023.3276863.

[217] A. Thakkar and R. Lohiya, "Attack classification of imbalanced intrusion data for iot network using ensemble-learning-based deep neural network," *IEEE IoT J.*, vol. 10, no. 13, pp. 11 888–11 895, 2023. DOI: 10.1109/JIOT.2023.3244810.

[218] A. Thakkar and R. Lohiya, "Fusion of statistical importance for feature selection in deep neural network-based intrusion detection system," *Information Fusion*, vol. 90, pp. 353–363, 2023.

[219] M. Wang, N. Yang, and N. Weng, "Securing a smart home with a transformer-based IoT intrusion detection system," *Electronics*, vol. 12, no. 9, Art. no. 2100, 2023.

[220] S. Mirjalili and A. Lewis, "The whale optimization algorithm," *Advances in Engineering Software*, vol. 95, pp. 51–67, 2016.

[221] D. T. Pham, S. S. Dimov, and C. D. Nguyen, "Selection of K in K-means clustering," *Proc. of the Inst. of Mech. Eng., Part C: Journal of Mechanical Eng. Science*, vol. 219, no. 1, pp. 103–119, 2005.

[222] M. Kumar and R. Mathur, "Unsupervised outlier detection technique for intrusion detection in cloud computing," in *Proc. Int. Conf. for Convergence for Technology-2014*, 2014, pp. 1–4. DOI: 10.1109/I2CT.2014.7092027.

[223] S. Kaski, J. Kangas, and T. Kohonen, "Bibliography of self-organizing map (SOM) papers: 1981–1997," *Neural Computing Surveys*, vol. 1, no. 3&4, pp. 1–176, 1998.

[224] B. Zong, Q. Song, M. R. Min, *et al.*, "Deep autoencoding Gaussian mixture model for unsupervised anomaly detection," in *Proc. Sixth Int. Conf. on Learning Representations (ICLR)*, 2018, pp. 1–19.

[225] *Pentaho*, http://wiki.pentaho.com/display/DATAMINING/Using+the+Weka+Scoring+Plugin., Accessed: 2022-7-21.

[226] Z. Wang, Z. Li, J. Wang, and D. Li, "Network intrusion detection model based on improved BYOL self-supervised learning," *Security and Communication Networks*, vol. 2021, no. 9486949, pp. 1–23, 2021.

[227] V. Mavroeidis and S. Bromander, "Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence," in *Proc. IEEE European Intelligence and Security Informatics Conf. (EISIC)*, 2017, pp. 91–98.

[228] A. Subaira and P Anitha, "Efficient classification mechanism for network intrusion detection system based on data mining techniques: A survey," in *Proc. IEEE 8th Int. Conf. on Intelligent Systems and Control (ISCO)*, 2014, pp. 274–280.

[229] R. Agarwal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. of the 20th VLDB Conf.*, 1994, pp. 487–499.

[230] E. Saboori, S. Parsazad, and Y. Sanatkhani, "Automatic firewall rules generator for anomaly detection systems with Apriori algorithm," in *Proc. IEEE 3rd Int. Conf. on Adv. Computer Theory and Eng. (ICACTE)*, vol. 6, 2010, pp. V6–57–V6–60.

[231] M. Hahsler, K. Hornik, and T. Reutterer, "Implications of probabilistic data modeling for mining association rules," in *From Data and Information Analysis to Knowledge Engineering*, Springer, Berlin, Heidelberg, 2006, pp. 598–605.

[232] R. Agrawal, T. Imieliński, and A. Swami, "Mining association rules between sets of items in large databases," in *Proc. ACM SIGMOD Int. Conf. on Management of Data*, 1993, pp. 207–216.

[233] S. Chavan, K. Shah, N. Dave, S. Mukherjee, A. Abraham, and S. Sanyal, "Adaptive neuro-fuzzy intrusion detection systems," in *Proc. Int. Conf. on Information Technology: Coding and Computing (ITCC)*, vol. 1, 2004, pp. 70–74. DOI: 10.1109/ITCC.2004.1286428.

[234] H. Wang, Z. Cao, B. Hong, and J. L. García Guirao, "A network intrusion detection system based on convolutional neural network," *J. Intelligent & Fuzzy Systems*, vol. 38, no. 6, pp. 7623–7637, 2020, ISSN: 10641246.

[235] Y. Gao, Y. Liu, Y. Jin, J. Chen, and H. Wu, "A novel semi-supervised learning approach for network intrusion detection on cloud-based robotic system," *IEEE Access*, vol. 6, pp. 50 927–50 938, 2018. DOI: 10.1109/ACCESS.2018.2868171.

[236] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, "An overview of ip flow-based intrusion detection," *IEEE Commun. Surv. & Tut.*, vol. 12, no. 3, pp. 343–356, 2010. DOI: 10.1109/SURV.2010.032210.00054.

[237] W. He, G. Hu, and Y. Zhou, "Large-scale IP network behavior anomaly detection and identification using substructure-based approach and multivariate time series mining," *Telecommunication Systems*, vol. 50, no. 1, pp. 1–13, Apr. 2012.

[238] L. Deri and S. Suin, "Effective traffic measurement using ntop," *IEEE Communications Magazine*, vol. 38, no. 5, pp. 138–143, 2000. DOI: 10.1109/35.841838.

[239] C. Tankard, "Advanced persistent threats and how to monitor and deter them," *Network Security*, vol. 2011, no. 8, pp. 16–19, 2011.

[240] P. Chen, L. Desmet, and C. Huygens, "A study on advanced persistent threats," in *Proc. IFIP Int. Conf. on Communications and Multimedia Security*, Springer, Berlin, Heidelberg, 2014, pp. 63–72.

[241] "High performance network monitoring solutions based on open source and commodity hardware." (2022), [Online]. Available: https://www.ntop.org/.

[242] "We're the creators of the elastic (elk) stack - elasticsearch, kibana, beats, and logstash. securely andreliably search, analyze, and visualize your data in thecloud or on-prem." (2022), [Online]. Available: https://www.elastic.co/..

[243] (2021), [Online]. Available: https://aws.amazon.com/opensearch-service/the-elk-stack/kibana/..

[244] "Intrusion detection evaluation dataset." (2017), [Online]. Available: https://www.unb.ca/cic/datasets/ids-2017.html..

[245] "Tcpreplay tool for packet replay." (2023), [Online]. Available: https://tcpreplay.appneta.com.

[246] I. Sharafaldin, A. H. Lashkari, and A. A. Ghorbani, "Toward generating a new intrusion detection dataset and intrusion traffic characterization.," *ICISSp*, vol. 1, pp. 108–116, 2018.

[247] "Cockpit makes it easy to administer your linux servers via a web browser." (2022), [Online]. Available: https://cockpit-project.org/..

[248] "Slash your time to detect, troubleshoot and resolve infrastructure performance anomalies." (2022), [Online]. Available: https://www.netdata.cloud/..

[249] "Amazon alexa." Accessed:Jan 31, 2023. (2023), [Online]. Available: https://www.amazon.com/b?node=14730500011.

[250] I. Makhdoom, M. Abolhasan, J. Lipman, R. P. Liu, and W. Ni, "Anatomy of threats to the internet of things," *IEEE communications surveys & tutorials*, vol. 21, no. 2, pp. 1636–1675, 2018.

[251] E. McKenna, I. Richardson, and M. Thomson, "Smart meter data: Balancing consumer privacy concerns with legitimate applications," *Energy Policy*, vol. 41, pp. 807–814, 2012.

[252] R. Yuan and T. Xin, "Design and research of multi-dimensional asset intelligent management system based on RFID technology," in *Second International Conference on Green Communication, Network, and Internet of Things (CNIoT 2022)*, X. Yuan, Ed., International Society for Optics and Photonics, vol. 12586, SPIE, 2023, 125861A. DOI: 10.1117/12.2670655. [Online]. Available: https://doi.org/10.1117/12.2670655.

[253] H Kivimäki, T Sinkkonen, S Marttonen, and T Kärri, "Creating a life-cycle model for industrial maintenance networks," in *Proc. 3rd Int. Conf. on Maintenance Performance Measurement and Management*, 2013, pp. 178–191.

[254] S. Marttonen, S. Monto, and T. Kärri, "Profitable working capital management in industrial maintenance companies," *Journal of Quality in Maintenance Engineering*, vol. 19, no. 4, pp. 429–446, 2013.

[255] T. Sinkkonen, H. Kivimäki, S. Marttonen, D. Galar, R. Villarejo, and T. Kärri, "Using the life-cycle model with value thinking for managing an industrial maintenance network," *International Journal of Industrial and Systems Engineering*, vol. 23, no. 1, pp. 19–35, 2016.

[256] M. Aboubakar, M. Kellil, and P. Roux, "A review of iot network management: Current status and perspectives," *Journal of King Saud University-Computer and Information Sciences*, vol. 34, no. 7, pp. 4163–4176, 2022.

[257] K. Akomea-Agyin and M. Asante, "Analysis of security vulnerabilities in wired equivalent privacy (wep)," *International Research Journal of Engineering and Technology*, vol. 6, no. 1, pp. 529–536, 2019.

[258] L. Karle, "Nms-less management of radio networks for small and medium sized enterprise networks," in *2022 13th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2022, pp. 1–5. DOI: 10.1109/ICCCNT54827.2022.9984494.

[259] D. D. Mishra, V. Dhakwal, S. Pathan, and C. Murthy, "Design and development of centralized squid proxy management system," in *2020 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT)*, 2020, pp. 1–6. DOI: 10.1109/CONECCT50063.2020.9198539.

[260] S. Erfani and M. Malek, "Potential applications of neural networks in network management," in *Proceedings of 36th Midwest Symposium on Circuits and Systems*, 1993, 257–260 vol.1. DOI: 10.1109/MWSCAS.1993.343081.

[261] "Ieee standard for framework of blockchain-based internet of things (iot) data management," *IEEE Std 2144.1-2020*, pp. 1–20, 2021. DOI: 10.1109/IEEESTD.2021.9329260.

[262] J. D. Case, M. Fedor, M. L. Schoffstall, and J. Davin, "Simple network management protocol (snmp)," Tech. Rep., 1989.

[263] R. Gerhards, "The syslog protocol," Tech. Rep., 2009.

[264] "Zabbix - opensource monitoring solution." Accessed:May 30, 2023. (2023), [Online]. Available: https://www.zabbix.com/features.

[265] "Zabbix - opensource monitoring solution." Accessed:May 30, 2023. (2023), [Online]. Available: https://www.zabbix.com/true_open_source#:~:text=Zabbix%20is%20released%20under%20the,thousands%20of%20devices%20absolutely%20free..

[266] "Nmap - scanning tool." Accessed:May 31, 2023. (2023), [Online]. Available: https://nmap.org/..

[267] "Azure cloud." Accessed:May 31, 2023. (2023), [Online]. Available: https://azure.microsoft.com/en-us..

[268] "Azure copy." Accessed:June 6, 2023. (2023), [Online]. Available: https://learn.microsoft.com/en-us/azure/storage/common/storage-ref-azcopy-copy?toc=%2Fazure%2Fstorage%2Fblobs%2Ftoc.json..

[269] J. Postel, "Simple mail transfer protocol," Tech. Rep., 1982.

[270] G. Lawton, "Open source security: Opportunity or oxymoron?" *Computer*, vol. 35, no. 3, pp. 18–21, 2002. DOI: 10.1109/2.989921.

[271] M. Starsinic, "System architecture challenges in the home M2M network," in *Proc. IEEE Long Island Systems, Applications and Technology Conf.*, 2010, pp. 1–7. DOI: 10.1109/LISAT.2010.5478336.

[272] C. E. Shannon, "A mathematical theory of communication," *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948.

[273] W. Weaver, "Recent contributions to the mathematical theory of communication," *ETC: A Review of General Semantics*, vol. 10, no. 4, pp. 261–281, 1953.

[274] S. Mihai, M. Yaqoob, D. V. Hung, *et al.*, "Digital twins: A survey on enabling technologies, challenges, trends and future prospects," *IEEE Commun. Surv. & Tut.*, vol. 24, no. 4, pp. 2255–2291, 2022. DOI: 10.1109/COMST.2022.3208773.

[275] H. Xu, J. Wu, Q. Pan, X. Guan, and M. Guizani, "A survey on digital twin for industrial internet of things: Applications, technologies and tools," *IEEE COMST, in print*, pp. 1–1, 2023. DOI: 10.1109/COMST.2023.3297395.

[276] M. Elbasheer, F. Longo, G. Mirabelli, L. Nicoletti, A. Padovano, and V. Solina, "Shaping the role of the digital twins for human-robot dyad: Connotations, scenarios, and future perspectives," *IET Collaborative Intelligent Manufacturing*, vol. 5, no. 1, Art. no. e12066, 2023.

[277] L. U. Khan, Z. Han, W. Saad, E. Hossain, M. Guizani, and C. S. Hong, "Digital twin of wireless systems: Overview, taxonomy, challenges, and opportunities," *IEEE COMST*, vol. 24, no. 4, pp. 2230–2254, 2022. DOI: 10.1109/COMST.2022.3198273.

[278] M. Masi, G. P. Sellitto, H. Aranha, and T. Pavleska, "Securing critical infrastructures with a cybersecurity digital twin," *Software and Systems Modeling*, vol. 22, no. 2, pp. 689–707, 2023.

[279] A. Pokhrel, V. Katta, and R. Colomo-Palacios, "Digital twin for cybersecurity incident prediction: A multivocal literature review," in *Proc. IEEE/ACM 42nd Int. Conf. on Software Eng. Workshops*, 2020, pp. 671–678.

[280] N. S. Thalpage and T. A. D. Nisansala, "Exploring the opportunities of applying digital twins for intrusion detection in industrial control systems of production and manufacturing–a systematic review," in *Data Protection in a Post-Pandemic Society*, Springer, Cham, 2023, pp. 113–143.

[281] S. A. Varghese, A. D. Ghadim, A. Balador, Z. Alimadadi, and P. Papadimitratos, "Digital twin-based intrusion detection for industrial control systems," in *Proc. IEEE Int. Conf. on Pervasive Comp. and Commun. Workshops and other Affil. Events (PerCom Workshops)*, 2022, pp. 611–617.

[282] Q. Xu, S. Ali, and T. Yue, "Digital twin-based anomaly detection in cyber-physical systems," in *Proc. IEEE Conf. on Software Testing, Verif. and Valid. (ICST)*, 2021, pp. 205–216.

[283] R. Ahlswede and G. Dueck, "Identification via channels," *IEEE Transactions on Information Theory*, vol. 35, no. 1, pp. 15–29, 1989.

[284] R. Ahlswede, "Identification via channels," in *Identification and Other Probabilistic Models*, Springer, Cham, Switzerland, 2021, pp. 3–43.

[285] S. Derebeyoğlu, C. Deppe, and R. Ferrara, "Performance analysis of identification codes," *Entropy*, vol. 22, no. 10, p. 1067, 2020.

[286] R. Ferrara, L. Torres-Figueroa, H. Boche, *et al.*, "Implementation and experimental evaluation of Reed-Solomon identification," in *Proc. Eu. Wireless*, 2022, pp. 1–6.

[287] C. Von Lengerke, A. Hefele, J. A. Cabrera, M. Reisslein, and F. H. Fitzek, "Beyond the bound: A new performance perspective for identification via channels," *IEEE Journal on Selected Areas in Communications, in print*, 2023.

[288] C. V. Lengerke, A. Hefele, J. A. Cabrera, O. Kosut, M. Reisslein, and F. H. P. Fitzek, "Identification codes: A topical review with design guidelines for practical systems," *IEEE Access*, vol. 11, pp. 14 961–14 982, 2023. DOI: 10.1109/ACCESS.2023.3244071.

[289] H. Boche and C. Deppe, "Secure identification for wiretap channels; robustness, super-additivity and continuity," *IEEE Trans. on Inform. Forensics and Sec.*,

vol. 13, no. 7, pp. 1641–1655, Jul. 2018, ISSN: 1556-6021. DOI: 10.1109/TIFS.2018.2797004.

[290] H. Boche and C. Deppe, "Secure identification under passive eavesdroppers and active jamming attacks," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 2, pp. 472–485, Feb. 2019, ISSN: 1556-6021. DOI: 10.1109/TIFS.2018.2854729.

[291] C. Alcaraz and J. Lopez, "Digital twin: A comprehensive survey of security threats," *IEEE Commun. Surv. & Tut.*, vol. 24, no. 3, pp. 1475–1503, 2022.

[292] L. Linguaglossa, S. Lange, S. Pontarelli, *et al.*, "Survey of performance acceleration techniques for network function virtualization," *Proc. IEEE*, vol. 107, no. 4, pp. 746–764, 2019.

[293] G. S. Niemiec, L. M. Batista, A. E. Schaeffer-Filho, and G. L. Nazar, "A survey on FPGA support for the feasible execution of virtualized network functions," *IEEE COMST*, vol. 22, no. 1, pp. 504–525, 2019.

[294] P. Shantharama, A. S. Thyagaturu, and M. Reisslein, "Hardware-accelerated platforms and infrastructures for network functions: A survey of enabling technologies and research studies," *IEEE Access*, vol. 8, pp. 132 021–132 085, 2020.

[295] A. S. Thyagaturu, P. Shantharama, A. Nasrallah, and M. Reisslein, "Operating systems and hypervisors for network functions: A survey of enabling technologies and research studies," *IEEE Access*, vol. 10, pp. 79 825–79 873, 2022. DOI: 10.1109/ACCESS.2022.3194913.

[296] O. K. Nicesio, A. G. Leal, and V. L. Gava, "Quantum machine learning for network intrusion detection systems, a systematic literature review," in *Proc. IEEE 2nd Int. Conf. on AI in Cybersec. (ICAIC)*, 2023, pp. 1–6.

[297] M. Kalinin and V. Krundyshev, "Security intrusion detection using quantum machine learning techniques," *J. Computer Virology and Hacking Techniques*, vol. 19, no. 1, pp. 125–136, 2023.

[298] V. R. Konduru and M. R. Bharamagoudra, "Challenges and solutions of interoperability on IoT: How far have we come in resolving the IoT interoperability issues," in *Proc. Int. Conf. On Smart Technologies For Smart Nation (SmartTechCon)*, 2017, pp. 572–576. DOI: 10.1109/SmartTechCon.2017.8358436.

[299] J. Aroles, C. Bonneau, and S. Bhankaraully, "Conceptualising 'meta-work'in the context of continuous, global mobility: The case of digital nomadism," *Work, Employment and Society, in print*, 2023.

[300] J. Marx, S. Stieglitz, F. Brünker, and M. Mirbabaie, "Home (office) is where your heart is: Exploring the identity of the 'corporate nomad' knowledge worker archetype," *Business & Information Systems Engineering*, vol. 65, 293—308, 2023.

[301] M. Orel, "Wanderlust workforce: A journey into understanding digital nomadism," *World Leisure Journal*, vol. 65, no. 2, pp. 143–149, 2023.

[302] J. I. Sánchez-Vergara, M. Orel, and I. Capdevila, ""Home office is the here and now." Digital nomad visa systems and remote work-focused leisure policies," *World Leisure J.*, vol. 65, no. 2, pp. 236–255, 2023.

[303] L. A. Haibeh, M. C. Yagoub, and A. Jarray, "A survey on mobile edge computing infrastructure: Design, resource management, and optimization approaches," *IEEE Access*, vol. 10, pp. 27 591–27 610, 2022.

[304] B. Mao, J. Liu, Y. Wu, and N. Kato, "Security and privacy on 6g network edge: A survey," *IEEE Commun. Surv. & Tut.*, vol. 25, no. 2, pp. 1095–1127, 2023. DOI: 10.1109/COMST.2023.3244674.

[305] F. Shirin Abkenar, P. Ramezani, S. Iranmanesh, *et al.*, "A survey on mobility of edge computing networks in IoT: State-of-the-art, architectures, and challenges," *IEEE COMST*, vol. 24, no. 4, pp. 2329–2365, 2022. DOI: 10.1109/COMST.2022.3211462.

[306] R. Singh, R. Sukapuram, and S. Chakraborty, "A survey of mobility-aware multi-access edge computing: Challenges, use cases and future directions," *Ad Hoc Networks*, vol. 140, Art. no. 103044, 2023.

[307] A. J. Ferrer, J. M. Marquès, and J. Jorba, "Towards the decentralised cloud: Survey on approaches and challenges for mobile, ad hoc, and edge computing," *ACM CSUR*, vol. 51, no. 6, 111:1–111:36, 2019.

[308] D. O. Rodrigues, A. M. de Souza, T. Braun, G. Maia, A. A. Loureiro, and L. A. Villas, "Service provisioning in edge-cloud continuum: Emerging applications for mobile devices," *J. Internet Services and Appl.*, vol. 14, no. 1, pp. 47–83, 2023.

[309] D. Rosendo, A. Costan, P. Valduriez, and G. Antoniu, "Distributed intelligence on the edge-to-cloud continuum: A systematic literature review," *J. Parallel and Distr. Comp.*, vol. 166, pp. 71–94, 2022.

[310] T. V. Doan, G. T. Nguyen, M. Reisslein, and F. H. Fitzek, "FAST: Flexible and low-latency state transfer in mobile edge computing," *IEEE Access*, vol. 9, pp. 115 315–115 334, 2021.

[311] M. A. Hathibelagal, R. G. Garroppo, and G. Nencioni, "Experimental comparison of migration strategies for MEC-assisted 5G-V2X applications," *Computer Communications*, vol. 197, pp. 1–11, 2023.

[312] I. Labriji, F. Meneghello, D. Cecchinato, *et al.*, "Mobility aware and dynamic migration of MEC services for the internet of vehicles," *IEEE TNSM*, vol. 18, no. 1, pp. 570–584, 2021. DOI: 10.1109/TNSM.2021.3052808.

[313] J. Santa, J. Ortiz, P. J. Fernandez, *et al.*, "MIGRATE: Mobile device virtualisation through state transfer," *IEEE Access*, vol. 8, pp. 25 848–25 862, 2020.

[314] N. Toumi, M. Bagaa, and A. Ksentini, "Machine learning for service migration: A survey," *IEEE Commun, Surv, & Tut,*, pp. 1–1, 2023. DOI: 10.1109/COMST.2023.3273121.

[315] Y. Li, K. Li, Y. Zheng, B. Morys, S. Pan, and J. Wang, "Threat assessment techniques in intelligent vehicles: A comparative survey," *IEEE Intelligent Transp. Systems Mag.*, vol. 13, no. 4, pp. 71–91, 2020.

[316] C. Debes, A. Merentitis, S. Sukhanov, M. Niessen, N. Frangiadakis, and A. Bauer, "Monitoring activities of daily living in smart homes: Understanding human behavior," *IEEE Signal Processing Magazine*, vol. 33, no. 2, pp. 81–94, 2016.

[317] https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet., Accessed: 2023-2-2.

[318] M. Guarascio, N. Cassavia, F. S. Pisani, and G. Manco, "Boosting cyber-threat intelligence via collaborative intrusion detection," *Future Generation Computer Systems*, vol. 135, pp. 30–43, 2022.

[319] N. Sun, M. Ding, J. Jiang, *et al.*, "Cyber threat intelligence mining for proactive cybersecurity defense: A survey and new perspectives," *IEEE COMST, in print*, pp. 1–1, 2023. DOI: 10.1109/COMST.2023.3273282.

[320] I. Trenchev, W. Dimitrov, G. Dimitrov, T. Ostrovska, and M. Trencheva, "Mathematical approaches transform cybersecurity from protoscience to science," *Applied Sciences*, vol. 13, no. 11, Art. no. 6508, 2023.

[321] *Palo Alto Single Pass Architecture*, https://www.paloguard.com/SP3-Architecture.asp, [Accessed: 26-July-2023], 2023.

[322] *Cisco Firepower Recommendations*, https://www.cisco.com/c/en/us/td/docs/security/firepower/640/configuration/guide/fpmc-config-guide-v64/tuning_intrusion_policies_using_rules.html#concept_F962F5CF64B74DBFA942BC21C227A [Accessed: 26-July-2023], 2023.

[323] *Palo Alto Advance Threat Inspection*, https://docs.paloaltonetworks.com/pan-os/10-2/pan-os-new-features/content-inspection-features/inline-cloud-analysis#id7a59d6db-6b7f-4ab1-aebf-d6468225cc40, [Accessed: 26-July-2023], 2023.

[324] "Half the world's malware is now encrypted." (2021), [Online]. Available: https://www.f5.com/company/blog/half-the-world-s-malware-is-now-encrypted.

[325] "Otx is a free stix/taxii feed." (2019), [Online]. Available: https://cybersecurity.att.com/blogs/security-essentials/otx-is-now-a-free-stix-taxii-server.

[326] "Threat intelligence products." (2022), [Online]. Available: https://www.crowdstrike.com/products/threat-intelligence/.

[327] "Smarter security, maximum impact." (2022), [Online]. Available: https://threatconnect.com/.

Appendix A

List of main acronyms used

Acronym	Explanation	Acronym	Explanation
AAF	Application-Aware Firewalls:	MANET	Mobile Ad-Hoc Networks
ADSL	Asymmetric Digital Subscriber Line	NAT	Network Address Translation
AES	Advanced Encryption Standard	NFS	Neuro-Fuzzy System
AI	Artificial Intelligence	NLP	Natural Language Processing
AIDS	Anomaly-based Intrusion Detection Systems	OS	Operating System
ANN	Artificial Neural Networks	OSINT	Open Source Intelligence
API	Application Programming Interface	PCA	Principal Component Analysis
APT	Advanced Persistent Threat	QoS	Quality of Service
ARM	Advanced RISC Machine	RFC	Request for Comments
C&C	Command and Control	RMD	Robust Mahalanobis Distance
CAN	Controlled Area Network	RNN	Recurrent Neural Network
CFA	Cuttlefish Algorithm	SASE	Secure Access Service Edge

CNN	Convolutional Neural Network	SDK	Software Development Kit
COTS	Commercial Off the Shelf	SELKS	Suricata, Elasticsearch, Logstash, Kibana Scirius CE
CPU	Central Processing Unit	SFTP	Secure File Transfer Protocol
CVE	Common Vulnerabilities and Exposures	SLA	Supervised Learning Algorithm
DAGMM	Deep Autoencoding Gaussian Mixture Model	SMB	Server Message Block protocol
DDoS	Distributed Denial of Service	SOHO	Small Office and Home Office
DGA	Domain Generation Algorithms	SOM	Self Organizing Maps
DoS	Denial of Service	SQL	Structured Query Language
DT	Decision Tree	SSH	Secure Socket Shell
EA	Evolutionary Algorithms	STIX	Structured Threat Information Expression
EVT	Extreme Value Theory	SVM	Support Vector Machine
FGLCC	Feature Grouping based on Linear Correlation Coefficient	TAXII	Trusted Automated Exchange of Indicator Information
FL	Fuzzy Logic	TCP	Transmission Control Protocol
FSM	Finite State Machine	TKIP	Temporal Key Integrity Protocol
FTP	File Transfer Protocol	UDP	User Datagram Protocol
HR	Human Resource	UL	Unsupervised Learning
HTTP	Hypertext Transfer Protocol	UPnP	Universal Plug and Play

HTTPS	Hypertext Transfer Protocol Secure	URL	Uniform Resource Locator
HVAC	Heating, Ventilation, and Air Conditioning	UTM	Unified Threat Management
IDS	Intrusion Detection Systems	VPN	Virtual Private Network
IoC	Indicator of Compromise	WAF	Web Application Firewalls
IoT	Internet of Things	WCCP	Web Cache Communication Protocol
IP	Internet Protocol	WPA	WiFi Protected Access
IPS	Intrusion Prevention Systems	WSN	Wireless Sensor Networks
KNN	K-Nearest Neighbor	XSS	Cross-Site Scripting
LDA	Linear Discriminant Analysis	YARA	Yet Another Recursive/Ridiculous Acronym
MFA	Multi-Factor Authentication		
ML	Machine Learning		

Appendix B

Microsoft Azure copy Template

```
{
"$schema":␣"http://schema.management.azure.com/schemas
/2015-01-01/deploymentTemplate.json#",
"contentVersion":␣"1.0.0.0",
"parameters":␣{
"location":␣{
"type":␣"string"
},
"storageAccountName":␣{
"type":␣"string"
},
"accountType":␣{
"type":␣"string"
},
"kind":␣{
"type":␣"string"
},
"minimumTlsVersion":␣{
"type":␣"string"
},
"supportsHttpsTrafficOnly":␣{
```

```
"type": "bool"
},
"allowBlobPublicAccess": {
"type": "bool"
},
"allowSharedKeyAccess": {
"type": "bool"
},
"defaultOAuth": {
"type": "bool"
},
"accessTier": {
"type": "string"
},
"publicNetworkAccess": {
"type": "string"
},
"allowCrossTenantReplication": {
"type": "bool"
},
"networkAclsBypass": {
"type": "string"
},
"networkAclsDefaultAction": {
"type": "string"
},
"networkAclsVirtualNetworkRules": {
```

```
"type": "array"
},
"dnsEndpointType": {
"type": "string"
},
"keySource": {
"type": "string"
},
"encryptionEnabled": {
"type": "bool"
},
"keyTypeForTableAndQueueEncryption": {
"type": "string"
},
"infrastructureEncryptionEnabled": {
"type": "bool"
},
"isContainerRestoreEnabled": {
"type": "bool"
},
"isBlobSoftDeleteEnabled": {
"type": "bool"
},
"blobSoftDeleteRetentionDays": {
"type": "int"
},
"isContainerSoftDeleteEnabled": {
```

```
"type": "bool"
},
"containerSoftDeleteRetentionDays": {
"type": "int"
},
"changeFeed": {
"type": "bool"
},
"isVersioningEnabled": {
"type": "bool"
},
"isShareSoftDeleteEnabled": {
"type": "bool"
},
"shareSoftDeleteRetentionDays": {
"type": "int"
}
},
"variables": {},
"resources": [
{
"apiVersion": "2018-05-01",
"type": "Microsoft.Resources/deployments",
"name": "virtualNetworks_0.555364931222831",
"subscriptionId": "19d0790e-aeae-49e0-90da-e858be9855a5",
"resourceGroup": "scms",
"dependsOn": [],
```

```
"resources":␣[],
"properties":␣{
"mode":␣"Incremental",
"parameters":␣{},
"template":␣{
"$schema":␣"http://schema.management.azure.com/schemas/
2015-01-01/deploymentTemplate.json#",
"contentVersion":␣"1.0.0.0",
"parameters":␣{},
"variables":␣{},
"resources":␣[
{
"apiVersion":␣"2021-01-01",
"name":␣"home_IP",
"type":␣"Microsoft.Network/virtualNetworks",
"location":␣"eastus",
"properties":␣{
"subnets":␣[
{
"name":␣"default",
"id":␣"/subscriptions/19d0790e-
aeae-49e0-90da-e858be9855a5/resourceGroups
/scms/providers/Microsoft.Network/
virtualNetworks/home_IP/subnets/default",
"properties":␣{
"addressPrefix":␣"10.0.0.0/24",
"serviceEndpoints":␣[
```

```
{
"service": "Microsoft.Storage"
}]}}],
"addressSpace": {
"addressPrefixes": [
"10.0.0.0/16",
"73.246.185.21/32"
]
}
},
"tags": {}
}
],
"outputs": {}
}}},
{
"name": "[parameters('storageAccountName')]",
"type": "Microsoft.Storage/storageAccounts",
"apiVersion": "2022-05-01",
"location": "[parameters('location')]",
"properties": {
"minimumTlsVersion": "[parameters('minimumTlsVersion')]",
"supportsHttpsTrafficOnly": "[parameters('supportsHttpsTrafficOnly')]",
"allowBlobPublicAccess": "[parameters('allowBlobPublicAccess')]",
"allowSharedKeyAccess": "[parameters('allowSharedKeyAccess')]",
"defaultToOAuthAuthentication": "[parameters('defaultOAuth')]",
"accessTier": "[parameters('accessTier')]",
```

```
"publicNetworkAccess": "[parameters('publicNetworkAccess')]",
"allowCrossTenantReplication": "[parameters('allowCrossTenantReplication')]",
"networkAcls": {
"bypass": "[parameters('networkAclsBypass')]",
"defaultAction": "[parameters('networkAclsDefaultAction')]",
"ipRules": [],
"virtualNetworkRules": "[parameters('networkAclsVirtualNetworkRules')]"
},
"dnsEndpointType": "[parameters('dnsEndpointType')]",
"encryption": {
"keySource": "[parameters('keySource')]",
"services": {
"blob": {
"enabled": "[parameters('encryptionEnabled')]"
},
"file": {
"enabled": "[parameters('encryptionEnabled')]"
},
"table": {
"enabled": "[parameters('encryptionEnabled')]"
},
"queue": {
"enabled": "[parameters('encryptionEnabled')]"
}
},
"requireInfrastructureEncryption": "[parameters
('infrastructureEncryptionEnabled')]"
```

```
}
},
"dependsOn": [
"Microsoft.Resources/deployments/virtualNetworks_0.555364931222831"
],
"sku": {
"name": "[parameters('accountType')]"
},
"kind": "[parameters('kind')]",
"tags": {}
},
{
"name": "[concat(parameters('storageAccountName'), '/default')]",
"type": "Microsoft.Storage/storageAccounts/blobServices",
"apiVersion": "2022-05-01",
"properties": {
"restorePolicy": {
"enabled": "[parameters('isContainerRestoreEnabled')]"
},
"deleteRetentionPolicy": {
"enabled": "[parameters('isBlobSoftDeleteEnabled')]",
"days": "[parameters('blobSoftDeleteRetentionDays')]"
},
"containerDeleteRetentionPolicy": {
"enabled": "[parameters('isContainerSoftDeleteEnabled')]",
"days": "[parameters('containerSoftDeleteRetentionDays')]"
},
```

```
"changeFeed": {
"enabled": "[parameters('changeFeed')]"
},
"isVersioningEnabled": "[parameters('isVersioningEnabled')]"
},
"dependsOn": [
"[concat('Microsoft.Storage/storageAccounts
/', parameters('storageAccountName'))]"
]},
{
"name": "[concat(parameters('storageAccountName'), '/default')]",
"type": "Microsoft.Storage/storageAccounts/fileservices",
"apiVersion": "2022-05-01",
"properties": {
"shareDeleteRetentionPolicy": {
"enabled": "[parameters('isShareSoftDeleteEnabled')]",
"days": "[parameters('shareSoftDeleteRetentionDays')]"
}
},
"dependsOn": [
"[concat('Microsoft.Storage/storageAccounts/',
parameters('storageAccountName'))]",
"[concat(concat('Microsoft.Storage
/storageAccounts/',
parameters('storageAccountName')), '
/blobServices/default')]"
]}],
```

"outputs": {}
}