

# Designing Network Security Tools for Home Users

by

John Faria, Yonghua Wang, and Michael Lai

A Major Qualifying Project (MQP)

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Bachelor of Science

in

Computer Science

by

---

December 2021

APPROVED:

---

Professor Lorenzo De Carli, MQP Advisor

## **Acknowledgements**

This MQP research project is under supervision of professor Lorenzo De Carli. We would especially thank him for bringing the research topic and advising the proceeding of the project.

Table 1: Abbreviations

C2	Command-and-Control
CTI	Cyber Threat Intelligence
DoS	Denial of Service
FN	False Negative
FP	False Positive
GUI	Graphical User Interface
IDS	Intrusion Detection System
IoC	Indicators of Compromise
IoT	Internet of Things
IP	Internet Protocol
ISP	Internet Service Providers
LAN	Local Area Network
ML	Machine Learning
PCAP	Packet Capture

# Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
1.1	Contributions . . . . .	3
<b>2</b>	<b>Background</b>	<b>5</b>
2.1	Home Network . . . . .	5
2.2	Products in the Market for Home Security . . . . .	6
2.3	Security Concerns with IoT Devices . . . . .	7
2.4	Understand Measure for Intrusion Detection . . . . .	8
2.5	Interviews to Understand Normal Home Network User's Concern With Home Security . . . . .	10
<b>3</b>	<b>Methodology</b>	<b>13</b>
3.1	Simulating Traffic . . . . .	13
3.2	Gathering Additional Network Traffic from IoT Devices . . . . .	16
3.3	Kitsune and Slips . . . . .	17
3.4	SNORT . . . . .	21
3.5	IDS Alert Generation With Fed Traffic . . . . .	24
3.6	Configuration of Slips . . . . .	26
3.7	Reverse Engineering to Understand Slips Deeper . . . . .	29
3.8	Understand Typical Process of Intrusion of Consequences . . . . .	33

3.9	Other Examples of Visualization Systems . . . . .	36
3.10	Principles of Designing Security-Awareness System . . . . .	38
3.11	Choice of Design Platform . . . . .	39
<b>4</b>	<b>Results and Limitations</b>	<b>41</b>
4.1	Interpreting Results . . . . .	41
4.2	Effectiveness of Warning and Awareness Enforcement . . . . .	43
4.3	Visualization on Dashboard . . . . .	46
4.3.1	Landing Page . . . . .	48
4.3.2	Statistics Page . . . . .	49
4.3.3	Devices Page . . . . .	50
4.3.4	Tips Page . . . . .	51
4.4	Use of IDS and Dashboard . . . . .	52
4.5	Limitations . . . . .	53
<b>5</b>	<b>Future Work</b>	<b>55</b>
5.1	Home Network Threat Intelligence . . . . .	55
5.2	Live Detections . . . . .	56
5.3	More Research For SLIPS and Similar AI Systems . . . . .	56
5.4	Improvements to Signature Detections . . . . .	57
5.5	Survey Non-Technical Users on Preferred Dashboard and Understanding . . . . .	58
<b>6</b>	<b>Conclusion</b>	<b>60</b>
<b>7</b>	<b>Appendix</b>	<b>63</b>

# List of Figures

2.1	Workflow of a network-based IDS [6]	8
3.1	High level view of our approaches	13
3.2	Diagram of virtualized network lab environment	14
3.3	Individual module's functionality of Slips [33]	19
3.4	Letter assignment, behavioral models	20
3.5	Slips' high-level work-flow	21
3.6	An example of two snort rules, one that's triggered by an outbound tcp request to a certain IP and one that's triggered by an inbound request	22
3.7	An example of Snort's alert	22
3.8	Live viewing of alert from Slips	24
3.9	Kalispo, the GUI in Slips to visualize outputs	25
3.10	Example of the alerts from Slips	25
3.11	Configuring Slips to training mode	27
3.12	Specifying samples' labels as normal	27
3.13	Real-time output of Slips	28
3.14	Human security cognition and augmentation workflow [37]	34
3.15	The landing page of the maritime intrusion monitoring system [38]	37
3.16	Incident UI widget	37

4.1	An example of an alert of Slips . . . . .	42
4.2	An example of evidence of Slips . . . . .	42
4.3	Visualizing post-processed JSON in browser . . . . .	42
4.4	Full E-AM model (HU stands for home users) . . . . .	45
4.5	Landing page of the dashboard . . . . .	48
4.6	Instructions and interpretations after clicking on the button . . . . .	49
4.7	Statistics page . . . . .	50
4.8	Devices page . . . . .	50
4.9	Tips page . . . . .	51
4.10	Stages of hacking . . . . .	52
7.1	Time window length available to adjust in Slips' configuration file . . . . .	65
7.2	Detecting direction . . . . .	65
7.3	Detection threshold . . . . .	65
7.4	Other adjustable threshold . . . . .	65

# List of Tables

1	Abbreviations . . . . .	ii
3.2	A brief summary of datasets we found . . . . .	17
3.3	Threat level and confidence associated with each type of detection . .	32



# Abstract

Cyberattacks are becoming more prevalent and threatening in our modern digital world but many home users, without large budgets for sophisticated security tools or skilled networks analysts, are left defenseless. Home networks are relatively small but still chaotic - with unrestricted use policies, various types of user endpoints (including vulnerable IoT devices), and new programs demanding more from home networks every day. The goal of our project is to research common attacks threatening home users and to prototype a system capable of helping users keep their home networks secure.

The main threats focused on in our research were phishing webpages, endpoint computers being compromised with malware, and IoT device take-over. Virtualized lab environments were created using Docker for the purposes of generating “malicious” network traffic in a controlled environment. The captured traffic included various stages of connecting to phishing webpages as well as different communications and attacks carried out by a command-and-control (C2) server.

We designed and prototyped an intrusion detection system (IDS) to find threats as they begin compromising a network and to efficiently communicate the detections to a non-technical home user so that they can secure their network. The IDS makes use of both malicious signature matching - using SNORT - as well as a machine learning (ML) for detecting attacks. The various detections are processed and displayed on a user-friendly dashboard. The IDS was designed to be host-centric, so the detections are related to potential compromise and mis-use of host devices. The dashboard also includes explanations of technical concepts and suggested actions for how users can secure their home network from detected threats.

# Chapter 1

## Introduction

Home users face cyber-attacks that range from phishing campaigns to IoT device compromise and even malware infections. Large organizations are also attacked and employ teams of information security professionals to monitor network threats and keep them safe - a solution that is not applicable to home users who are responsible for their own network security. Several home intrusion detection systems (IDSs) have been designed to provide home users with a network layer of defense [19, 20, 21]. However, these IDSs were adapted from those created for information security professionals, and yet home networks are administered by inexperienced users [22]. Home users who attempt to implement these IDSs into their network defenses are often confused by the generated warnings, get overwhelmed with alert-fatigue [23], or do not understand how the home IDS protects them [24, 27].

Currently, approaches have been proposed to bridge the gap between home network stakeholders and solutions to potential security issues [44, 45]. Those tools aimed to remove the burden from users to regulate home networks and take responsibility for it. Although those achievements have surely enhanced home network security from different aspects and through different ways, users are not always

consulted or considered during the decision making process, which leaves some limitations and cannot solve the problem completely. An interface between those algorithms and end users matters but lacks attention at this point. The first reason that the interface matters is the high false positive rate of an IDS [25]. Without a high enough true positive rate, leaving all decision making to the IDS would harm the performance and usability of the home network. This situation also concerns alert fatigue. Alert fatigue occurs when an individual is exposed to an excessive number of alerts which finally leads to desensitization [1]. Several factors contribute to alert fatigue, including a complex IT environment, lack of contextual information, and redundancy. Alert fatigue can result in people underestimating the emergence and significance of alerts or even totally ignore them. In the worst scenario, costs increase greatly to maintain operation of security while integrity, confidentiality, and availability is not guaranteed at all. Secondly, the automated process of intrusion detect and prevention, the black box way of tackling the problem, does not truly lower users' concern but even aggravates anxiety when breaches and anomaly occur and no context information or remediation are provided. Finally, beyond the previous two reasons, as stakeholders of the home network, users deserve the rights to have accessibility to security protection behind the scene and react to anomalies with their choice of consideration.

## 1.1 Contributions

Our project investigated technology which makes the output of a home IDS more accessible and understandable to its users. We deployed both signature-based IDS and anomaly detectors and produced outputs from existing IoT traffic for analysis. Based on contents included in output, we designed and implemented an al-

gorithm to reduce duplication of alert and aggregate evidence. We researched the specific threats to home users as well as how details on the incident and recommendations for remediation can be communicated to the user. We explored how the user will be given non-technical details on the events and what the steps they can take to resolve the incident. Finally, we engineered user-friendly dashboards that can provide information on the overall health of their network or help users better understand the threat-levels of specific alerts. The dashboards and presented alerts are intended to explain to the user why the incident is dangerous and potential consequences of it.

# Chapter 2

## Background

### 2.1 Home Network

A user's home network originally served to connect a user's computer to websites that they are interested in or important digital communications such as email. As technology has evolved and digital devices have become a more integral part of modern society, the usage of and strain on home networks has increased. Modern homes have several Internet-connected devices per user including smart phones, personal computers, and shared IoT devices. The users' personal devices are constantly connected to the Internet - getting updates about notifications or being used for various forms of web browsing. Devices are also commonly used for various forms of remote-work, content streaming, and online gaming. New Internet of Things (IoT) devices including Internet-connected cameras and smart home technologies are fully-automated, usually insecure, computers that significantly adds to the complexity. The Mirai Botnet, for example, is a malicious network of millions of compromised computers (bots) many of which are vulnerable or misconfigured IoT devices [26]. To be more specific, Mirai scans for and gains access and control of IoT devices that

host a stripped version of Linux system and do not change default username and password.

The drastically increased usage of home networks by so many new devices and programs has increased the potential attack surface against home networks. Complexities in web browsing, management of multiple different devices, and simply users accessing technology more has increased their likelihood of falling victim and increased the potential reward for attackers. Users who do not understand their technology are more likely to not notice a potential indicator of compromise or to not know how to secure their devices. It is important to empower home users so that they can have the tools needed to manage and protect their home network.

## **2.2 Products in the Market for Home Security**

There are few existing home network security products and many fail to efficiently communicate with a user and protect the complexities of a home network. Businesses have been dealing with multifaceted internal networks for decades and by utilizing highly technical workers, information security procedures, and developed security tools, businesses have made great improvements towards increased organizational cyber-security. Home networks can be significantly more chaotic than a businesses' network due to the lack of policies regarding what users can and should run on the network. Attempting to apply a professional business intrusion detection system (IDS) to a home network will result in many false positives due to benign unrestricted network usage. The large amount of false positives only increase user anxiety and waste time without adding to their home network's security. Additionally, home users are not as technical as a network security analyst and can neither configure the detection tools to their network nor manage all of the false positives

they are alerted to.

## 2.3 Security Concerns with IoT Devices

Although there are diverse IoT products which vary from size, functionality, and prices, there exist similarities at hardware and software level, which contribute to security challenges faced by IoT devices [2]. Typical hardware components of an IoT device include logic chips, memory, flash storage, network module, and serial debug interface. As for software, an IoT device has a bootloader, which initializes the hardware device and loads the firmware to the boot device, and firmware which includes operating systems, filesystem, and service programs.

IoT devices not only suffer from traditional cybersecurity issues, but also bring new attack surfaces due to their distinct structure. Vulnerabilities on the software layer are multifaceted. First, operating systems are tailored to satisfy devices with different capabilities and energy supply. Traditional buffer-overflow problems with computers also happen to IoT devices. Second, lightweight local storage solutions are always used for IoT devices, which lead to the consequence that many sensitive information are stored as plain text. Third, because of the lack of standards in the IoT industry, IoT manufacturer's products contain customized codes that are unprofessional and unsafe. There have been several cases of backdoors, hardcoded passwords, and several other issues [3]. Fourth, for ease of regulation and use, IoT devices are configured to have weak authentication. Some might even not require a password to login to access the device. Such devices can be easily found by IoT search engines such as Shodan [4]. Besides software, protocols used to communicate for IoT devices also reveal a certain level of risk. For devices that use HTTP services, they are potentially vulnerable to SQL injection, Cross-site Scripting (XSS), etc.

Furthermore, weak encryption algorithms are used to save energy [28] or there is no encryption at all for some cheap IoT devices. At worst cases, the password is transmitted in plain text, which can be easily intercepted and wiretapped.

## 2.4 Understand Measure for Intrusion Detection

An intrusion detection system (IDS) is a program that scans network traffic for known dangers as well as suspicious or malicious activities [5]. When the IDS identifies any security concerns or threats, records are documented and notifications are pushed to IT and security professionals for post analysis.

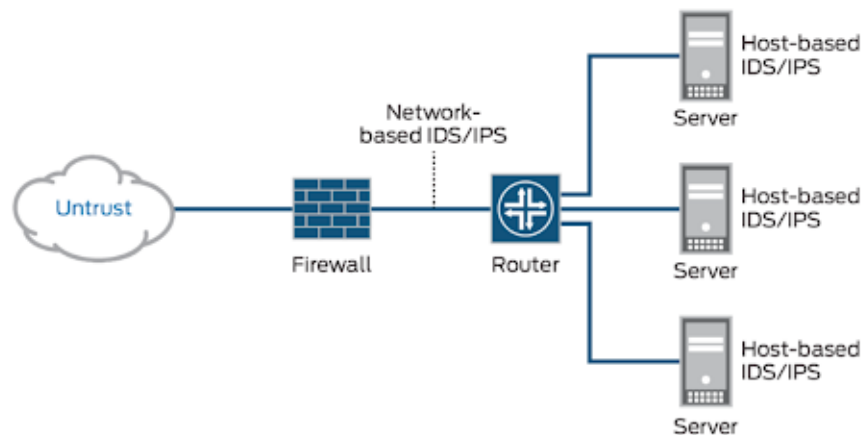


Figure 2.1: Workflow of a network-based IDS [6]

Traditionally IDS works in this way: it searches for signatures of known attack types or detects behavior that deviates from a set of rules. This type of IDS include Snort, Bro, and Suricata, etc. They require a database of predefined rules to identify potential malicious intrusions. This type of IDS works well if technical experts can define the right amount of rules to identify known attacks but this becomes less useful with zero-day attacks. In a word, signature-based IDS is good at identifying



existing and known attacks but underperforms at detecting attacks whose signature the database never documents.

The other way is anomaly detection. This category of IDS always employs and trains statistical models or neural networks to “understand” what an malicious activity might look like. After appropriate feature extraction and feeding the model with a large amount of traffic, the model is able to distinguish benign and malicious ones. Though promising, such a method is not fully reliable due to false positive rate and training cost [7]. As aforementioned heterogeneity of IoT devices, there are always new ways of attack toward IoT devices which may contain new features which are ignored by the model. Another concern is the training cost of the model. The cost involves two aspects: 1. Data gathering, 2. Computational power for training. The first has to do with privacy concerns. It is inevitable that normal home network traffic can contain sensitive information such as browsing history records. As a result, researchers might have to set up their own testing testbed and simulate traffic themselves [7]. The second depends on the layer of the neural network [8]. This is impossible if we try to deploy the model at a network gateway or router which has a very limited amount of storage and data processing capability.

Intrusion detection systems will alert a network analyst to activity that is non-standard and appears to be indicative of harm. Some potential detections could include large amounts of data being sent to an unfamiliar server, an endpoint device conducting internal network scans, or a user accessing an IP address which has been known to host malicious content [41, 42]. All of these detections could be evidence indicating malicious activity, or they could indicate benign network traffic indicative of a user at home. For example, a new process sending data to an unfamiliar server could indicate a malicious connection to a C2 server or a new benign service being implemented by a home user. It takes someone who can understand the

alerts to determine if a detection is legitimate. IDS technologies are improving to become more accurate - largely through combinations of signature detections and machine learning - but network monitoring systems are not capable of reliably differentiating between what is and what is not safe for a home network. It will be important for a home user to understand the alerts communicated from their security technology as much as possible and that technology itself should minimize false positives while maximizing communications with the home user. Home users need to be empowered with the knowledge of what has been detected and which of their devices are involved. Details such as these can be the difference between a user believing their data is being stolen and not knowing how to respond, versus a user recognizing that the large amounts of data being sent to an unfamiliar server is just from them playing a game online. Advice on how a user should respond should also be provided - including how they can get more information on if a compromise has occurred and what steps they can take to secure their network.

## **2.5 Interviews to Understand Normal Home Network User's Concern With Home Security**

The goal of our project is to make network security accessible to home users. Hence, it is essential to involve stakeholders during the design process in order to thoroughly understand the crux of the problem, i.e., what contributes to obscurity of home security issues and how to circumvent it. We conducted a semi-structured interview to gather relevant viewpoints from three other WPI students toward IoT network security and their expectations regarding the visualization dashboard. The reason we took semi-structured interviews is that the interviewees would be more familiar with the background and necessity of our project after answering our ques-

tions and could further provide suggestions from the angle of a normal home network user. It is also worth mentioning that the three students are majoring in RBE (robotics engineering), CS (computer science), and IMGD (interactive media and game development), respectively. Though our interviewees all have some technical background, none of them has significant knowledge regarding cyber security. The following questions were asked first during the interview:

1. Do you have any IoT devices?
2. Do you consider convenience over security?
3. Do you know any potential security problems related to IoT?

Surprisingly, none of our interviewees responded that they owned any IoT devices (excluding devices like Apple watches). With that, we could then expect that neither of them perceives any security issues pertaining to IoT. What impressed us is the turnaround of attitude for the trade-offs between convenience and security. Originally, every interviewee inclined that convenience is more important because one of the reasons that IoT is popular is that IoT makes life easier. After we introduced some scenarios of breaches however, such as your smart curtain opening and closing by itself and a smart bulb turning itself on and off suddenly, interviewees no longer believed that convenience matters the most. We concluded that users would pay much more attention to security issues if they perceived the consequences of an attack.

Similar research has been done to bridge IoT consumers with IoT security and privacy issues. The research team makes an effort improving standardized labels on IoT products to inform consumers about privacy [9]. The issue is that current labels focus mainly on privacy and do not take security into account enough. After in-depth and semi-structured interviews, they discovered that:

1. Customers have incorrect knowledge about privacy and security.
2. Customers seldom considered security issues before purchasing.

The first negatively impacted customer's ability to measure the risks of using IoT devices and make the right judgment and decision. Pre-purchasing considerations include curiosity, health/fitness, price/convenience, reliability, and privacy, ranking from highest to lowest. Nevertheless, the second is no longer held after purchasing. It has been found that customers have no idea where to get informed about privacy and security knowledge. This coincided with our conclusion that it does matter to provide sufficient and user-friendly security information.

# Chapter 3

## Methodology

In this chapter, we discuss our entire process to approach the explainability of security tools step by step. In the end, the dashboard incorporated all the effort we made and presented a way of visualizing the monitoring system and helping home network users grasp their network situation.

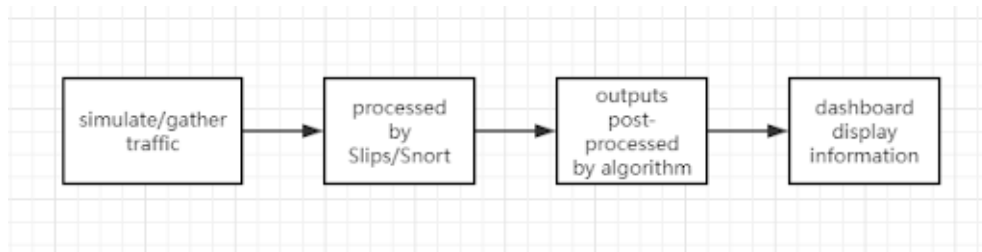


Figure 3.1: High level view of our approaches

### 3.1 Simulating Traffic

In order to test the effectiveness of the intrusion detection system, the team needed experimental “malicious” network traffic that was fully understood and could be modified as necessary. The two forms of malicious traffic that was generated are command-and-control (C2) communications as well as a phishing site. The

malicious test servers and the victim machine were all created in Docker [29] and a packet capture was run on the local host system. Using the experimental C2 server and phishing site, a series of tests were run to capture network traffic that would be generated from such malicious communications. The packet captures were later studied and used to create and test the intrusion detection system.



Figure 3.2: Diagram of virtualized network lab environment

The Caldera automated adversary emulation platform by MITRE [11] was used to create a C2 server for testing purposes. Caldera is a well supported C2 framework and the adversary actions that are automated by Caldera are already mapped to the ATT&CK Matrix - a professional layout describing parts of a cyber attack [12]. The Caldera framework also includes a Python payload, named Ragdoll Agent, which is written to communicate with the C2. Caldera automates the simulation

of various adversaries as they attack a network through a series of commands from the C2 server. The attacks mimicked in the team's experiments include the Ragdoll payload being downloaded and installed, aggressive internal network scanning from the infected computer, the Ragdoll Agent continuously querying the server for new commands, and data being exfiltrated from the infected computer. All of these malicious actions were run in the Docker test environment and individual packet captures were generated for each stage.

To emulate the threat of a user visiting malicious webpages, a fake phishing lab environment was set up to capture different parts of network traffic relating to dangerous web browsing. A test phishing web server was created to mimic a Facebook login page [43]. The web server was hosted on a local Docker container running Apache. Artifacts on the web page included media outside of the original webpage - which generated additional HTTP requests. The test computer's local DNS cache was also configured to include a record for fakebook.phish which resolved to the local docker container. Finally, after uploading the credentials to the web server via an HTTP Post request, there is a redirection to the real Facebook website. To generate different aspects of web browsing traffic in packet captures, the website was accessed via basic web clients like cURL, fully-functional web browsers, and simple DNS lookups.

To avoid poisoning their results with the team's own biases on malicious network traffic, the team also gathered network traffic generated from communications with real attacker infrastructure. One such network packet capture was from a real Facebook phishing page which inspired fakebook.phish. Several other example packet captures of command-and-control network traffic came from Brad Duncan and his blog on analyzing C2 communications [13].

To help test and train the team's IDS prototype, packet captures of benign net-

work traffic were also collected. By testing the prototype against the benign traffic, the team could work towards minimizing false positive alerts on home networks. Experimental benign network traffic was generated by running a packet capture on a virtual machine as team members browsed the web, played computer games, and transferred files.

## 3.2 Gathering Additional Network Traffic from IoT Devices

As stated earlier, in addition to our simulated lab network captures, our team also gathered additional network traffic from outside sources. Some of these were sourced from similar lab environments, such as network traffic used in a research paper regarding Kitsune (an IDS utilizing machine learning) [7]. For other sources of network traffic, our team searched online and not only did we discover phishing and C2 data from Malware-Traffic-Analysis, but we also found IoT network traffic datasets such as IoT-23 [30] from Avast and the Bot-IoT dataset from UNSW Canberra [31]. Both of these datasets primarily include network traffic relating to botnet malware as well as some benign network traffic. These scenarios that the datasets provide are from realistic, simulated environments, where the researchers would build a network using real hardware and would attack that network with malware. For example, the IoT-23 dataset uses an Amazon Echo and Philips smart LED lamp for their devices. The scenarios provided from these captures helped to increase the sample size of network traffic our team would work with, as well as provide some real scenarios for our IDS to analyze.



Name	Size	Attack Vector	Details (if any)
Aposemat IoT-23	21 GB (over 100 after uncompressed)	20 malware captures executed in IoT devices, and 3 captures for benign IoT devices traffic	A dataset contains malicious and benign IoT network traffic that has been tagged.
Kitsune Surveillance Network Intrusion Datasets	Around 10 GB	ARP MitM, SSDP Flood, OS Scan, Active Wiretap, SYN Flooding, Fuzzing, Video Injection, Renegotiation	
BoT-IoT Dataset	69.3 GB	DDoS, DoS, OS and Service Scan, Keylogging and Data exfiltration attacks	Built in UNSW Canberra's Cyber Range Lab by constructing a realistic network environment

Table 3.2: A brief summary of datasets we found

### 3.3 Kitsune and Slips

As the scope of our project does not include designing and building new tools or algorithms to detect intrusion, we want to utilize existing tools to process traffic in order to generate results for further analysis. In addition to using a traditional signature-based IDS, we also want to include an AI-based IDS to have a comparison.

The first IDS we considered using is Kitsune [7]. This is an IDS that embraces unsupervised learning and neural networks. What makes Kitsune different from other machine-learning based IDS is that Kitsune tries to address the problem of unsupervised learning and online processing. Typically, data used to train the model must be stored locally for future usage. This is not feasible in home situations because of the increase in overhead and cost to store everyday traffic produced by

the home network. After that, the second problem pertains to data cleaning and labeling. In an industry setting, this is achieved by IT specialists who know the network well, but in a home setting, this proves to be more challenging as home users would need to familiarize themselves with the technology and terminology. Also, it is increasingly expensive to identify and label attacks in real life as types of attack increase and evolve, becoming more sophisticated. The last problem is with computation power required by ANN, which naturally contradicts the resources we can expect at a gateway or home. To solve these problems, Kitsune gets rid of an expert's involvement to label traffic malicious or benign by leveraging unsupervised learning techniques. It also has an ensemble of small neural networks which substitute the complex ANN, making the IDS be able to process packets online and drop the packet immediately after training or executing.

Kitsune is a powerful IDS, however, the output of Kitsune was not so useful for our project. Given a packet, Kitsune will predict how similar this packet is to what Kitsune has seen based on the concept learnt by the model after training. For example, if the model is trained to understand the pattern of normal traffic, then given a malicious packet its reconstruction RMSE (root mean square error) will be relatively high compared to a normal packet. The crux then is that this RMSE is not useful for explainability of intrusion, which gives no other useful context information. Therefore, we finally decided not to use Kitsune for detection.

Slips is another IDS we found [32]. Slips is a behavior-based intrusion detection/prevention system that leverages both machine learning and signature to identify malicious behaviors in the network traffic. It can read live traffic live and also network captures, such as pcap files, Suricata, Zeek/Bro, and Argus flows. After running Slips against some sample pcap files, it outputs evidence and highlights alerts to which analysts should pay more attention.

Slips has different modules which are respectively designed and programmed to detect one type of malicious traffic. A brief summary of each module's capability can be found in the table below.

module	description
https	training&test of RandomForest to detect malicious https flows
port scan detector	detects Horizontal and Vertical port scans
threat Intelligence	checks if each IP is in a list of malicious IPs
timeline	creates a timeline of what happened in the network based on all the flows and type of data available
rnn-cc-detection	detects command and control channels using recurrent neural network and the stratosphere behavioral letters
VirusTotal	module to lookup IP address on VirusTotal
flowalerts	module to find malicious behaviour in each flow. Current measures are: long duration of the connection, successful ssh
IP_Info	module to find Geolocation, ASN, RDNS info about IPs and MAC vendors
RiskIQ	Module to get different information from RiskIQ
ARPScanDetector	module to check for ARP scans in ARP traffic
ExportingAlerts	module to export alerts to slack, STIX or suricata format
http_analyzer	module to analyze HTTP traffic
blocking	module to block malicious IPs connecting to the device
flowmldetection	module to detect malicious flows using ML pretrained models
leak_detector	module to detect leaks of data in the traffic using YARA rules

Figure 3.3: Individual module's functionality of Slips [33]

The two modules that make Slips an anomaly detector are rnn-cc-detection and flowmldetection, which can be found in the table above. The researchers who developed Slips have trained a recurrent neural network model with their own data based on behavioral letters, in which the details of these letters are listed below. This comes from the source code of Slips.

	Size Small			Size Medium			Size Large		
	Dur. Short	Dur. Med.	Dur. Long	Dur. Short	Dur. Med.	Dur. Long	Dur. Short	Dur. Med.	Dur. Long
<b>Strong Periodicity</b>	a	b	c	d	e	f	g	h	i
<b>Weak Periodicity</b>	A	B	C	D	E	F	G	H	I
<b>Weak Non-Periodicity</b>	r	s	t	u	v	w	x	y	z
<b>Strong Non-Periodicity</b>	R	S	T	U	V	W	X	Y	Z
<b>No Data</b>	1	2	3	4	5	6	7	8	9

**Symbols for time difference:**

**Between 0 and 5 seconds:** .  
**Between 5 and 60 seconds:** ,  
**Between 60 secs and 5 mins:** +  
**Between 5 mins and 1 hour:** \*  
**Timeout of 1 hour** 0

Figure 3.4: Letter assignment, behavioral models

We have looked at every detail of documentation available but there is no more relevant information introducing the model such as exact datasets used to train the model. The flowmldetection module uses by default the SGDCClassifier with a linear SVM (support vector machine), according to the official guide suggests [34]. The advantage of using a SVM is that the model can be trained synchronously and extended with new data, which allows us to train Slips later to reduce false positives from benign traffic.

Zeek [21], an open source network security monitoring tool, is placed inside Slips to generate flow. Slips further classifies traffic by the packet's IP address, and assigns each IP address a unique profile ID. Traffic belonging to the same profile is further grouped by time window size, i.e., the period during which traffic happened. After processing, flows are fed into each module to find any suspicious activity. Detections are done for each time interval. In other words, Slips focus on the general behavior of the traffic rather than individual packets. If any potential malicious behavior is found, contextual information about that activity, such as IP and start time, is documented in JSON and log files in an output folder.

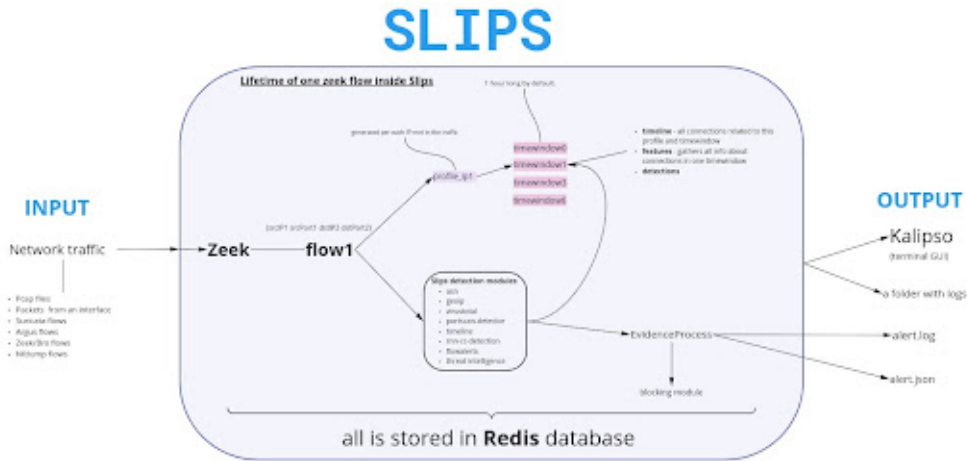


Figure 3.5: Slips' high-level work-flow

For detailed commands to save outputs inside the container, they can be found in the appendix.

### 3.4 SNORT

For our signature-based IDS, we chose Snort [19], a rule-based IDS/IPS which allows us to use a set of defined rules and match them towards our malicious packets to display the alerts we want. Snort can be used to check live traffic or analyze packet captures.

Our setup of Snort 2.9 was run on the Windows operating system, with a modified config file in order to work on the operating system (while an executable file was developed for Windows, Snort was developed mainly for Linux, and as such, the config file assumed a Unix file system). In order to troubleshoot any potential issues that would occur, an instance of Snort 3.0 on a Linux system was used. This was primarily just in case the configuration for Windows didn't work, in which we could easily switch to the Linux version to figure out if it was an OS issue or a configuration issue. The difference in version numbers was only due to the fact that Snort

3.0 wasn't available on Windows. This difference didn't have a significant change on how Snort operated, but there were slight changes in the rule syntax. Due to that, we had to create parallel rules depending on the version and make comments in the rule files letting us know which rules to use.

After we completed the Snort setup, we were able to start writing custom rules to generate alerts. These alerts would correspond to malicious packets found in our lab captures. The first step was to manually look for these malicious packets and figure out any identifiable attributes that could indicate malicious activity. This could include the protocol type, IP address, port number, or the content of the packet itself. Once we identified these attributes, we created rules that mapped those attributes. This way, every time a rule would come across a packet that matches those attributes, it would produce an alert with a message of our choice.

```
alert tcp any any -> 192.168.1.14 8888 (msg:"80, C2, Outbound request to known C2 Server"; sid:20000001;)
alert tcp 192.168.1.14 8888 -> any any (msg:"80, C2, Inbound request from known C2 Server"; sid:20000002;)
```

Figure 3.6: An example of two snort rules, one that's triggered by an outbound tcp request to a certain IP and one that's triggered by an inbound request

Other than the message we can customize for when the alert is triggered, the alert contains some additional information we can display to the user. This includes the attributes used in the SNORT rule (the IP, port number, protocol type), as well as the time and date the packet was sent.

```
[**] [1:10000002:0] 50, Phishing, DNS response for known phishing site [**]
[Priority: 0]
10/06-03:49:00.855370 127.0.0.53:53 -> 127.0.0.1:33588
UDP TTL:64 TOS:0x0 ID:53584 IpLen:20 DgmLen:76 DF
Len: 48
```

Figure 3.7: An example of Snort's alert

The custom message in the Snort alert contained three parts. The last two parts details the type of attack that was detected, for example, one specific Snort rule

would have been designed to capture a phishing attack, based on the DNS response for our fakebook.phish site. The first part is a score we assign to the severity of the attack the packet is capturing. This score is from a scale of 0-100, where:

- (0-19): Probably nothing, best to check anyway
- (20-39): Possibly something, please check
- (40-59): Attack detected, nothing serious yet, determine action immediately
- (60-79): Attack in progress, pursue action now
- (80-100): Critical, pursue action now to mitigate or stop attack

The idea of the score was to closely match the threat level that Slips was outputting. Our team ranked the potential danger and relative certainty of each Snort rule based on our background research on that specific attack vector.

In order to run Snort on a packet capture, we would run the Snort executable in the command line with this command:

```
./snort -r <packet_capture_location> -c <snort_config_location> -k none
```

The first two parameters are self explanatory; `packet_capture_location` refers to the packet capture location and `snort_config_location` is the Snort config file. This config file had to be given in order to specify which rule files you wanted to use as well as where we wanted Snort to place its output logs. `-k none` is an interesting option, and we discovered the use of this option thanks to our own lab captures. We discovered that some of our lab captures had bad checksum data, and while we could verify the packet integrity through Wireshark, Snort couldn't distinguish this on its own, and refused to analyze the packet. In order to bypass this, the `-k` option specifies the checksum mode, so `none` will cause Snort to turn off its checksum verification system.

## 3.5 IDS Alert Generation With Fed Traffic

Slips depends on several dependencies and libraries to function properly, including Python, Redis, NodeJS, and Zeek and each has its own version requirement. Considering Slips' error-prone and time consuming installation process, we decide to run Slips inside a docker container. To download the image, we run the following command:

```
Dockek pull stratosphereips/slips:latest
```

After installation, we executed the container with:

```
docker run -it --rm --net=host -v /dataset:/StratosphereLinuxIPS/dataset  
stratosphereips/slips:latest
```

The `-rm` flag will make sure the container is deleted after exit or it could consume lots of disk space. The `-v` flag instructs docker to combine the container with a volume so that the container can read the datasets from the local file system.

To start inspecting traffic, run

```
./slips.py -c slips.conf -f dataset/<File to inspect>
```

First Slips will update a list of remote threat intelligence files. After this update, evidence and alerts will be available to view immediately after they are generated.

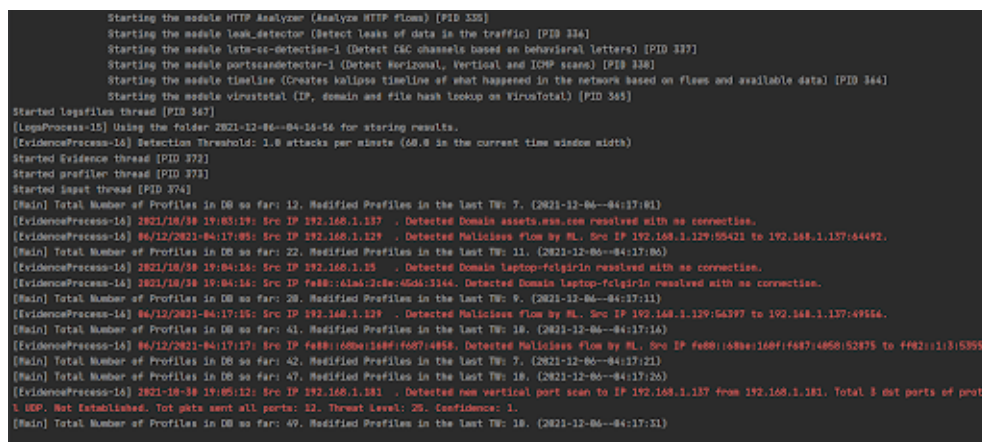
A terminal window showing the output of the Slips application. The output includes several status messages for various modules like HTTP Analyzer, leak\_detector, lsm-sc-detector, portscan, timeline, and virustotal. It also shows log file processing, evidence thread, profiler thread, and input thread. The main output consists of several lines of alerts, such as: '[Main] Total Number of Profiles in DB so far: 12. Modified Profiles in the last TW: 7. [2021-12-04--04:17:01]', '[EvidenceProcess-16] 2021/10/30 19:03:19: Src IP 192.168.1.137 . Detected Domain assets.wsn.com resolved with no connection.', and '[EvidenceProcess-16] 06/12/2021-04:17:03: Src IP 192.168.1.129 . Detected Malicious Flow by ML. Src IP 192.168.1.129:55421 to 192.168.1.137:64492.' The terminal also shows the total number of profiles in the database and modified profiles in the last time window.

Figure 3.8: Live viewing of alert from Slips



Then we can execute `./kalipso.sh`, the GUI tool associated with Slips which supports inspecting the output in real time. Profiles that have triggered alerts will be highlighted as red.

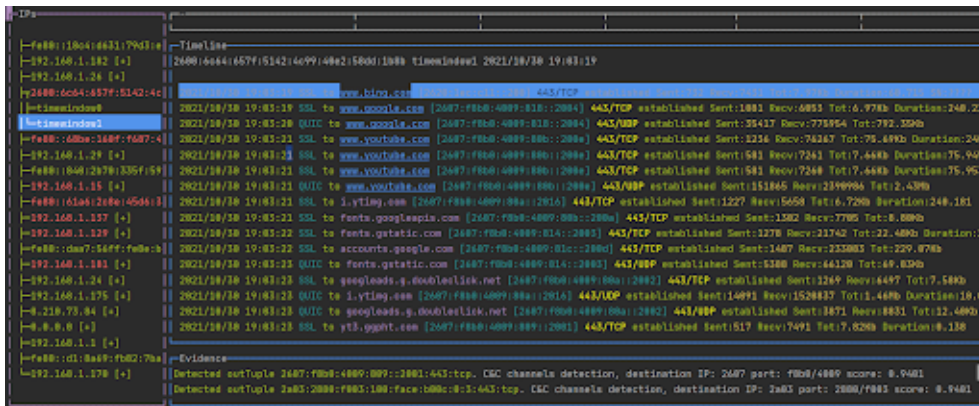


Figure 3.9: Kalipso, the GUI in Slips to visualize outputs

Finally, all records and files will be saved to the output folder.

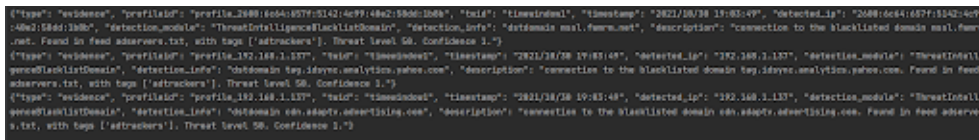


Figure 3.10: Example of the alerts from Slips

The results are promising and useful because we feed Slips with traffic generated from the Caldera framework and we know the traffic is malicious itself. It turns out Slips is able to recognize attacks with all built-in and programmed features before diving into deeper analysis. However, the results are obviously not friendly to home network users at all, even IT experts. Post data processing must be done to make the logs more approachable.

## 3.6 Configuration of Slips

Before generating more traffic with datasets, it is important to understand how Slips behaves with normal traffic. We want to compare the results of normal traffic with that of malicious ones to see any issue of false positives or the results can be misleading. To prevent leakage of private information and ensure the traffic is without malicious ones, we decided to set up a virtual environment. The tools we used include Virtual Box, a Windows 10 operating system (using an ISO), and Wireshark. From within the Windows virtual machine, we would use Wireshark to begin recording network traffic and then try to mimic normal behavior a home network user would produce. This involved opening the browser to visit popular websites including Facebook, Youtube, Twitter, etc, without inputting any credentials. We also downloaded small files and played some online games. This primarily involved downloading randomly selecting zipped Github repositories and because of the limitations of playing an online game in a VM, we opted to use a simple online browser game, agar.io. Considering the time frame of network activity might make a difference to predictions of the machine learning model, i.e., network traffic happens at late night when people sleep might be more suspicious than one that happens during the day, we repeated the recording two more times, for a total of three. These recordings occurred respectively at 4pm, 4:30am, and 10:30pm. Besides, we also researched online to explore any normal IoT traffic available to use. Unfortunately, there are far less publicly available normal captures than malicious ones. Even files of benign traffic from the IoT-23 [30], obtained by capturing the network traffic of real IoT devices, are corrupted after we uncompressed them.

After obtaining the control group, which is the normal traffic that we recorded in the isolated environment, we ran Slips against it. As per expectation, there is

much evidence that indicates potentially malicious activity. Those evidence are from modules including: DNSWithoutConnection, PortScan, C&C channels detection, flow detection, and ConnectionWithoutDNS. The last one is reasonable as the laptop maintains its database which saves and maps domain names to IP addresses then querying the DNS server is no longer needed. Evidence from the C&C channels detection module is generated by the built in neural network model, which is not controllable and retrainable for us, so there is not much we can do about it until later to adjust the threat level associated with such evidence. Regarding alerts from the machine learning model, we are able to retrain it by reconfiguring Slips to make it more adaptable to our production environment. Generally speaking, it is reasonable to see false positive results from IDS. Nevertheless, we must find a way to lower their appearing frequency so that we are not telling home users there is malicious activity while there is actually not. In the worst case, this could lead to users distrust the tool and ignore any alert or instruction provided by the tool.

To train the machine learning model, we first edited the configuration file.

```
#####
# [6] Specific configuration for the module flowdetection
[flowdetection]
# The mode 'train' should be used to tell the flowdetection module that the flows received are all for training.
# A label should be provided in the [Parameters] section
#mode = train

# The mode 'test' should be used after training the models, to test in unknown data.
# You should have trained at least once with 'Normal' data and once with 'Malicious' data in order for the test to work.
mode = test
```

Figure 3.11: Configuring Slips to training mode

Also, we made sure the label is normal.

```
# Set the label for all the flows that are being read. For now only normal and malware directly. No option for setting labels with a filter
label = normal
#label = malicious
#label = unknown
```

Figure 3.12: Specifying samples' labels as normal

After adjusting configuration, we ran Slips as usual. We fed in the first two

files that contained the normal traffic and then reconfigured Slips to test mode. The third file is used as a test to see how much the model is improved after training. Due to the scope and time limitation of our project, we are not able to systematically and quantitatively measure the improvements. However, we did see a reduction in the number of evidence from the ML detection module after training although evidence from other modules such as PortScan and Command and Control are still present, which is expected to happen. The JSON file of those will be included as part of submission and deliverables.

```
[Main] Total Number of Profiles in DB so far: 32. Modified Profiles in the Last TH: 7. (2021-12-06--04:17:01)
[EvidenceProcess-16] 2021/10/30 19:05:19: Src IP 192.168.1.137 . Detected Domain assets.msn.com resolved with no connection.
[EvidenceProcess-16] 06/12/2021-04:17:05: Src IP 192.168.1.129 . Detected Malicious flow by ML. Src IP 192.168.1.129:55421 to 192.168.1.137:64492.
[Main] Total Number of Profiles in DB so far: 22. Modified Profiles in the Last TH: 11. (2021-12-06--04:17:04)
[EvidenceProcess-16] 2021/10/30 19:04:16: Src IP 192.168.1.15 . Detected Domain laptop-felgin16 resolved with no connection.
[EvidenceProcess-16] 2021/10/30 19:04:16: Src IP fe80::61a6:208e:d506:3144. Detected Domain laptop-felgin16 resolved with no connection.
[Main] Total Number of Profiles in DB so far: 28. Modified Profiles in the Last TH: 9. (2021-12-06--04:17:11)
[EvidenceProcess-16] 06/12/2021-04:17:15: Src IP 192.168.1.129 . Detected Malicious flow by ML. Src IP 192.168.1.129:56397 to 192.168.1.137:49566.
[Main] Total Number of Profiles in DB so far: 41. Modified Profiles in the Last TH: 18. (2021-12-06--04:17:16)
[EvidenceProcess-16] 06/12/2021-04:17:17: Src IP fe80::60ba:169f:f607:4868. Detected Malicious flow by ML. Src IP fe80::60ba:169f:f607:4868:52875 to fe80::113:5365.
[Main] Total Number of Profiles in DB so far: 42. Modified Profiles in the Last TH: 7. (2021-12-06--04:17:21)
[Main] Total Number of Profiles in DB so far: 47. Modified Profiles in the Last TH: 18. (2021-12-06--04:17:26)
[EvidenceProcess-16] 2021-10-30 19:05:12: Src IP 192.168.1.181 . Detected scan vertical port scan to IP 192.168.1.137 from 192.168.1.181. Total 3 dat parts of protoce
1.000. Not Established. dat gets sent all ports: 32. Threat Level: 25. Confidence: 1.
[Main] Total Number of Profiles in DB so far: 49. Modified Profiles in the Last TH: 19. (2021-12-06--04:17:31)
[Main] Total Number of Profiles in DB so far: 49. Modified Profiles in the Last TH: 7. (2021-12-06--04:17:36)
[Main] Total Number of Profiles in DB so far: 49. Modified Profiles in the Last TH: 6. (2021-12-06--04:17:41)
[Input] We read everything. No more input. Stopping input process. Sent 2562 Lines
[EvidenceProcess-16] 2021/10/30 19:05:25: Src IP 2000:6064:057f:5142:4e99:48a2:596d:1000. Detected connection to the blacklisted domain googleads.g.doubleclick.net. F
ound in feed observers.txt, with tags ['adtrackers']. Threat Level 58. Confidence 1..
```

Figure 3.13: Real-time output of Slips

Further, we also found other features in the configuration file that are worth exploration. The first one is `time_window_width`. We mentioned previously that Slips assigns each IP address a profile ID and traffic that belongs to the same profile which is further grouped by time window size. The default time window length is an hour.

The second is should Slips inspect only traffic that goes out from home or those that either go out or come in. For our project, we are interested in all traffic so we configure the analysis direction to the all option.

The last ones are several thresholds we can adjust based on need which include minimum confirmed attacks per minute needed to generate an alert, connection duration to decide if a connection is long.

Above all, we realized that reverse engineering is required to understand how Slips works exactly and how those thresholds are used. Only in this way, we are able to better explain and convey home network security issues to home users in plain language, with help of analogy and simile if necessary. This might pave the way and inspire us for designing and implementing a visualization solution.

### **3.7 Reverse Engineering to Understand Slips Deeper**

After inspecting the details of configuration files, we start from the program's entry point to understand Slips deeper. When Slips starts running, the parent process spawns several child processes, each of which represent a module that performs a type of scanning strategy.

Redis, an in-memory data structure store, is used to store all information pertaining to the traffic. Redis supports multiple processes to access cached data at the same time. The PUB/SUB messaging system allows those modules to collaborate by enabling processes to post data to channels and subscribe to channels.

It is also worth mentioning the composition of a record before diving into details of each module. There are two types of records, the first is evidence, and the second is alerts. At the beginning, we misinterpreted the difference between the two as that evidence is derived from anomaly-based modules which include module that detects command and control channels using recurrent neural network and the stratosphere behavioral letters, and flowmldetection, which detects malicious flows using ML pre-trained models, and alert is from signature-based module, which is potentially more accurate than the model's predictions because the alert is triggered when certain rules are violated. After inspecting details of the evidence process module, it turns out that our assumption is incorrect. In fact, every module produces evidence. The

evidence process module will receive those evidence published by other modules because of subscription. After receiving publication from channels, it parses the evidence and saves it to the database. The evidence has several fields including type (is evidence or alert), profile ID, twid (time window ID), time stamp, detection IP, detection\_module (the module that produces the evidence), detection\_info (the two IP addresses involved in communication), and description that summarizes everything. After that, the program will decide if a certain condition has been met to trigger an alert. It queries the database with both profile ID and time window as a filter to find all evidence that falls under these two criteria. The threat level of a evidence is calculated as the following:

$$\text{threat\_level} = \text{threat\_level} * \text{confidence}$$

Different detection methods have different confidence levels which scale between 0 and 1, and different types of attack have different threat levels, which range from 0 to 80. Recollect that in the configuration file we are able to assign the detection threshold and time window length. This is used in the following formulae:

$$\text{scaled\_threshold} = \text{detection\_threshold} * \text{time\_window\_size} / 60$$

The reason it is divided by 60 is that the accumulation of threat is minute based and the units of time window size is in seconds. Threat level of evidence with the same time window and profile ID will be added together. If the accumulated threat level is higher than the scaled threshold and the IP address is not included in the white list, then an alert is triggered which contains the type, profile ID, time window, and accumulated threat level. Understanding this is integral to our further decisions making to interpret and visualize results.

Threat Level	Module	Confidence	Detail
--------------	--------	------------	--------

80	Threat Intelligence	1	Check if the srcIP or dstIP are in a malicious list of IPs
60	ARP Scan Detector	0.8	ip x sends arp requests to 3 or more different ips within 30 seconds
60	Flow Alert Port Scan	1	
60	Flow Alert Data_exfiltration	0.6	Systems that are transferring large amount of data in 20 mins span
50	Threat Intelligence	0.8	If the domain is in the blacklist
50	ARP Scan Detector	0.8	Sending ARP packet to a destination address outside of local network
30	Flow Alert (self signed certificates)	0.5	
30	Flow Alert (connection_without_dns_resolution)	1	A connection without DNS resolution
30	Flow Alert (dns_resolution_without_connection)	0.3	No connection established after DNS resolution
30	Machine Learning Model Detection	0.6	Prediction results as malware
30	RNN C2 Detection	0.6	

20	Flow Alert (Reconnection Attempts)	0.5	Multiple reconnection attempts to a destination IP
20	Flow Alert (Connection to multiple ports)	0.5	
20	Flow Alert (Port 0 scanning)	0.8	
10	Flow Alert (Long connection)	0.5	Give a malicious label if duration is larger than threshold in configuration file
10	Flow Alert (Unknown port)	1	Checks destination ports that are not in our modules/timeline/services.csv file (contains typical visiting service)
0.01	Flow Alert (SSH successful)	0.5	

Table 3.3: Threat level and confidence associated with each type of detection

A detailed description of each detection method and its associated threat level can be found in the above table. This greatly improves our understanding toward the tool. We can even adjust the threshold depending on our needs, however, we are not focusing on identifying the most accurate and appropriate threshold, so we proceed with the existing ones.



## 3.8 Understand Typical Process of Intrusion of Consequences

It is crucial to understand the process of attacks and associated consequences to effectively persuade home users on why security matters [35]. Cyber situational awareness (CSA) [36] is the understanding of the IT environment, the cybersecurity threats and vulnerabilities confronting that environment, as well as the process of anticipating the potential consequences. It is pivotal for an effective cyber security analysis and incident response, which is what we want to enforce in home situations. CSA is roughly classified into three major categories: Recognition (being aware of situation), comprehension (applying appropriate knowledge to interpret data with specific context), and projection (education and assessment to neutralize future attacks and mitigate their threats). The diagram [37] below shows one way that different stages of CSA evolve and interact.

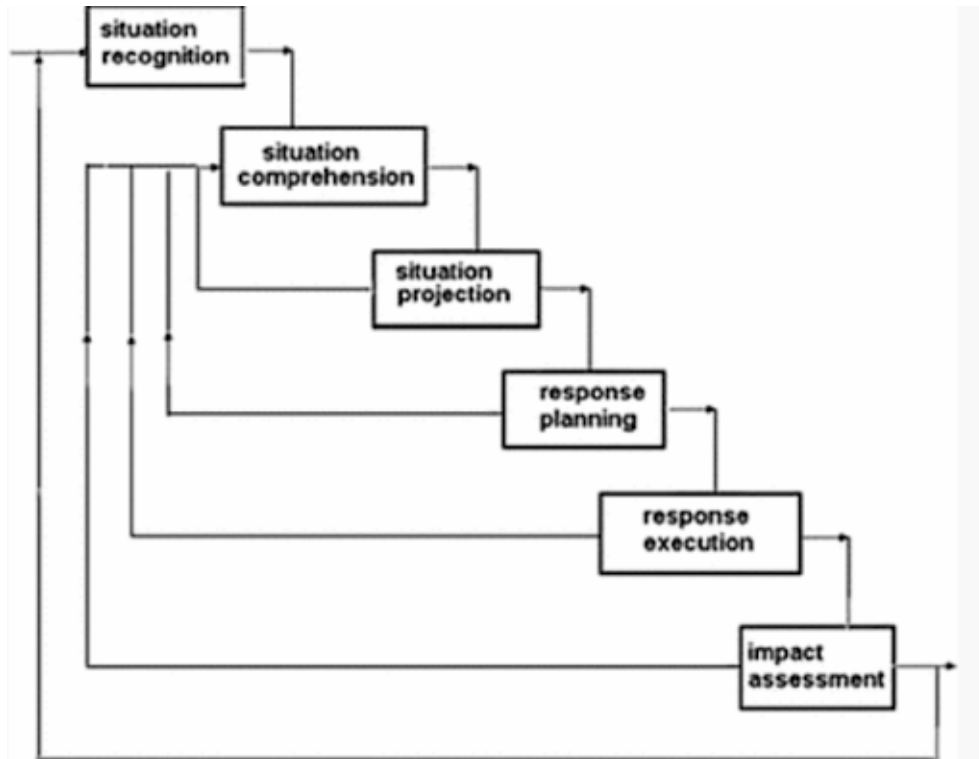


Figure 3.14: Human security cognition and augmentation workflow [37]

The above figure depicts the process a home user would go through if it encounters an intrusion. The user should be able to recognize and comprehend the situation first. Then, instructions should be available for home users to execute with and respond to intrusion. Based on knowledge gained from the incident, the home user is able to better handle similar situations in the future. This is what we want to achieve via the design of our dashboard.

The amount of time required to reinforce CSA isn't trivial. Some common questions can be summarized to be answers for CSA:

- Is there an intrusion incident?
- How does the situation develop?
- What is the consequence of the attack?

- How to measure severity of attack?
- What does an intruder gain from attack?
- Knowledge gained after an attack?

We want to include answers to those questions in our dashboard to help home users fully grasp the situation and even take efficacious actions to stop cyber crimes.

We also need to find a commonality between cyber attacks though there are heterogeneous types of attacks emerging. This plays a role making cyber intrusion accessible for home users who have no domain knowledge of computer networks. Typically, a successful attack might consist of the following steps [35]:

1. Reconnaissance
2. Scanning
3. Gaining access
4. Maintaining access
5. Covering tracks

At the first stage, the attacker gathers the target's information. This can be achieved in many ways such as social engineering and physical reconnaissance. At the next stage, the attacker will try scanning the target's network to discover devices connected to the local area network, which services are running, and any vulnerability that could be exploited. With the information in hand, the attacker tries to penetrate the firewall, enter the network, and gain access by sending phishing links and emails, spoofing packets, etc. Fourth, the attacker stays within the affected network in order to acquire the information they seek or to leave behind

something (perhaps turning the device into part of a botnet). Finally, the attacker erases enough traces they left behind in order to remain undetected. Those are to be placed in the dashboard for home users' reference because detection modules of Slips can also affiliate with one of them. By doing so, the detection module and intrusion steps mutually refer to each other and enhance intelligibility.

### **3.9 Other Examples of Visualization Systems**

We are also interested in other existing visualization systems which might inspire us what contents and layouts can look like. One we found is a platform that monitors operational maritime cyber security [38]. As more and more IoT devices engage in the infrastructure of harbors and are used to automate optional equipment, cyber intrusions also increase while awareness is low. Many devices' systems are outdated and many of them use default passwords. Intrusions can be left undetected for a long time or even being detected, the process involving multi-parties to resolve the intrusion is lengthy and verbose. As a result, it is pressing to have an operational platform that backs up analyzing heterogeneous security incidents by visualizing information exchange and sharing in real time.

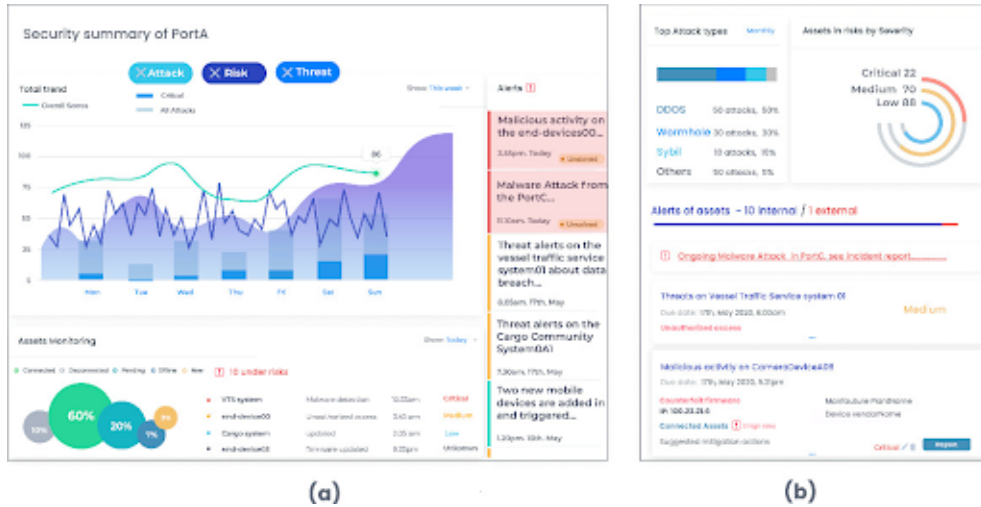


Figure 3.15: The landing page of the maritime intrusion monitoring system [38]

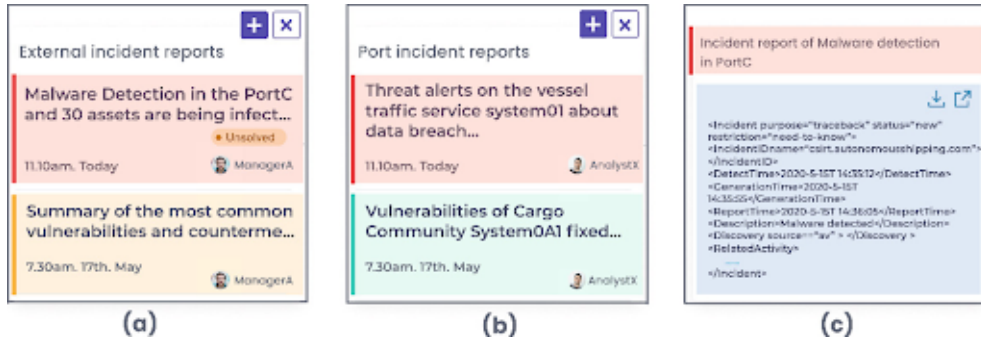


Figure 3.16: Incident UI widget

In order to support multiple role collaboration, different user interfaces are designed accordingly. In our case, the home user will take care of its own home network. However, other aspects are worth learning from. First, the landing page informs the regulator with an overview of the security situation by employing a bar chart and curve. Colors are actively used to visualize and differentiate levels of severity and types of attacks. Statistical data are summarized such as total number of incidents from a type of attack. For individual alerts, its elements include title, date, cause of alert, and threat level. Though this interface is primarily designed for industry use, we actually discovered that many of the design philosophies can apply to home

situations as well.

### 3.10 Principles of Designing Security-Awareness System

From the perspective of information retrieval and processing, humans should be involved during decision making with computers to improve the throughput [10]. Computers excelled in handling large-scale computation tasks, but underperformed at identifying patterns. Given information fragments, they face challenges piecing them together and further making inference. On the contrary, humans can naturally recognize regularity and pattern with intuition. Instead of requiring massive amounts of data for training, we can easily recollect and reapply knowledge only after we see something a few times. Therefore, we can naturally draw the conclusion that to maximize efficiency, the combination and collaboration of human cleverness and machine's power is necessary.

Visualization is the technique that visualizes abstract concepts by incorporating shapes, colors, and any other elements to assist human's perception and understanding [14]. In cyber security, a visualization system can serve as an entry point for humans to understand the status of a network, devices connected in the network, ect [15]. However, [15] also mentions that there are incorrect ways of designing an interface, such as a complex navigation system and dense information presentation, which can lead to aversion to the interface. According to [16], typical challenges include:

1. Volume, variety, and velocity of data
2. Multiple data source

3. Unlinked data source
4. Data quality
5. Pattern of network
6. Threat escalation progression
7. Balancing risk and reward.

Our project primarily focuses on problems of 1, 6, and 7. We have designed an algorithm to aggregate outputs and IDS. An alert is only given if it exceeds a certain threat level threshold. We also tried to give users instructions to recover from an intrusion.

In [17], several principles and thumb-of-rules are summarized for display of information. The paramount goal is to engender efficient and effective perception, comprehension and decision making. Perceptual principle stresses legibility. Multiple factors and parameters should be taken into account to avoid absolute judgement based on a single value. Deviations from expectations should be highlighted. Differences ought to be accentuated. The mental model requires us to cater to the user's mental state. The attention model indicates that events or incidents from the same cause or share the same level of severity should have similar visual looking. Finally, the memory principle pertains to availability of information. Users should be able to access and grasp information and knowledge easily. It is also helpful if some predictive aiding is provided.

### **3.11 Choice of Design Platform**

The decision which platform and language to implement a visualization system is under thorough consideration. Several factors are scrutinized. The first is mobility

of the platform. In the open discussion with interviewees, all of them express an inclination to receive alerts from mobile devices such as phones. Second, the ease of implementation. The entire process of designing a user interface demands a long period of effort. Stages include gathering background information to understand the needs of stakeholders through interview, prototype and layout and color scheme, and finally implementation with programming languages. We expect the final stage to not consume a lot of time. Third, an ample number of UI toolkits and visualization libraries available to use. All together leads to the choice of the nodeJS with express framework, which is the server side, and JavaScript, CSS, and HTML for the front end. The product of the interface will be shown in the result section.



# Chapter 4

## Results and Limitations

### 4.1 Interpreting Results

Many IDSs face the challenge of alert fatigue and non-trivial false-positive rate, this happens to Slips as well. For instance, a pcap file with size around 120 MB, which is the malware dataset from Kitsune project [7], can result in over 3000 records generated by Slips. It is imperative to design and implement an algorithm to process and summarize those records for better intelligibility. The reverse engineering of Slips paves the way to do so. Aforementioned difference between alert and evidence is the key to distinguishing trivial evidence and non-trivial ones. Recollect that each evidence has a threat level associated with it. An alert is generated only if the aggregate threat level of evidence from the same profile and time window exceeds a certain threshold predefined by configuration files. Therefore, it is reasonable to first filter out evidence that does not trigger any alert. After that, it is also important to reformat the contents of an alert. An alert of Slips consists of five fields which are type, profile ID, time window ID, and the accumulated threat level. The alert by default is in this format. Although some of those are informative such as the threat



time, we can safely ignore this inconsistency and continue working on visuazalition. The ending time now is simply the start time plus the time window length. The start time is the one that happened first between the two in the record.

Network traffic from a host can easily mimic part of an attack without actually being malicious. For example, an endpoint conducting a basic internal network scan could be indicative of an attacker conducting reconnaissance - or it could just be an IoT device trying to connect. A home user should not be overly concerned with just one detection because IDS detections can be triggered for both malicious and benign traffic - of course given time and interest it is always beneficial to investigate such detections however this is not required for most home network security. A single detection is not necessarily a cause for concern, however when different detections appear in the same several hours and are all coming from a single endpoint, then that is much more indicative of a real attack. For example, the internal network scanning discussed above is not necessarily a cause for concern, however when the same endpoint begins periodic encrypted communications with a new endpoint, then the likely false-positive now appears to be part of a botnet.

The algorithm used by the team for communicating threats to the user does not treat all directions equally and attempts to only highlight endpoints that truly appear to be under attack.

## **4.2 Effectiveness of Warning and Awareness Enforcement**

It is important for the dashboard to be approachable for non-technical users and to empower them to secure their network. The log files from IDS tools SNORT and Slips, which had been configured to detect threats in a home network, were

passed-into a program for the dashboard to be visualized.

The first challenge we face is the effectiveness of warning. Akhawe conducted a large-scale field study of user decisions after seeing browser's security warnings [39]. They utilize the browser's telemetry framework to collect pseudonymous data. Through comparing click-through rate of phishing and SSL warning messages with different designs and appearance, they concluded that user experience can affect user's behavior. In other words, properly and carefully implemented procedures and interface can enhance effectiveness of warning. There are several lessons we learn. First, subtle detail makes a difference. For example, the click through rate of warning messages showing untrusted SSL certificates and that showing expired SSL certificates is different. Second, clicking through is a single cognitive decision. This reveals that extra steps to bypass the warning does not actually play the role we expect them to. This is because viewers' decisions to proceed is a binary choice. If he/she wants to ignore the warning, the extra steps only delay but never suppress the intention. Hence, it is critical to make the warning effective enough at the first glimpse. Third, the effectiveness of warnings collapses if they appear too frequently. This referred to alert fatigue which has been discussed before. Last, users typically do not click on the button to learn more so succinct and informative warning should be given at the first time. All of the above will be applied in dashboard design.

The second challenge is how to incorporate users into the decision making process for their own home network security. Kritzinger proposed the Electronic Awareness Model (E-AM) [40]. E-AM emphasizes that users should acquaint themselves with risks associated with surfing the internet. E-AM addresses the issues that home users find difficulty absorbing cyber security knowledge and understanding home security situations by providing up-to-date information based on the user's home network environment. Information includes devices connected to the LAN, relevant

security issues, why those issues matter, and how to tackle them. We also should keep in mind that our audiences are home users without any domain knowledge of cyber security. Therefore, the information in the dashboard should satisfy those requirements:

1. Accessible
2. Integrated
3. User-friendly
4. Comprehensive
5. Relevant
6. Up-to-date

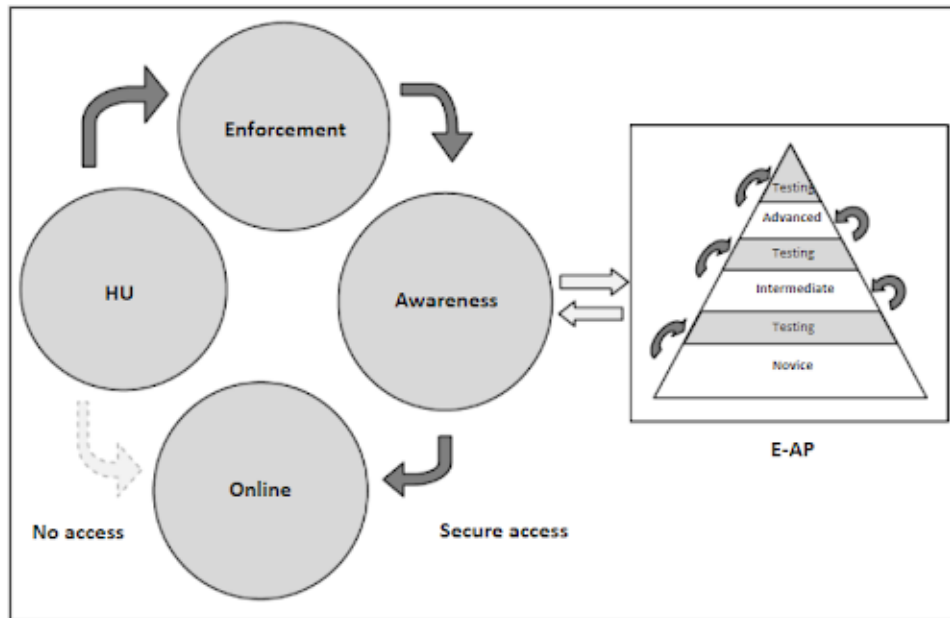


Figure 4.4: Full E-AM model (HU stands for home users)

It can be told from the graph that the home user does not have access to the internet if he/she does not go through the entire training process. This of course fully

prepares the home users to surf the internet more safely, similar to having a driving license before driving on the road. However, this is not feasible considering social and legal aspects. Hence, at the current stage, we should focus on making cyber security knowledge accessible so that home users are willing to actively participate in the movement to raise awareness of risks.

### 4.3 Visualization on Dashboard

To understand how normal home network users envision a GUI of a cyber-security monitoring and alert system, we also have another semi-structured interview with our previous interviewee. Questions asked can be found in the appendix. Interviewees are three students majoring in RBE (robotics engineering), CS (computer science), and IMGD (interactive media and game development), respectively. Though our interviewees all have some technical background, none of them has domain knowledge of cyber security. There are several relevant findings. First, all three participants replied they prefer to receive alert notifications from their phone without hesitation. And after clicking on the notification UI widget, they want to jump into the app to see more details of the alert. Second, for contents of an alert notification, threat level of the intrusion, time of intrusion, and name of the device that triggers the alert should be included. For customizable features, three participants responded differently. First participant mentioned he wants to be able to decide the sensitivity level of a device, which should be counted to calculate the threat level. The second participant suggested that he wanted to be able to set the notification frequency himself, and a default should not exceed per device per day. The third person expressed that he should be empowered to have control over all his devices, i.e., he can block traffic of any device simply by clicking a button. Besides,

we found that it is not that people do not pay attention to security but they do not realize how significant the consequences could be if an intruder launches the attack successfully. When we asked how to enforce awareness of security issues they instead asked us what security issues exist. We concluded it is necessary to include tips, stories, and news about IoT cyber-intrusion real cases within the dashboard to enhance user's awareness. Finally, there is one point worth mentioning at the final open discussion section. One interviewee summarized that there are two types of attack regarding privacy. The first is privacy-sensitive one. For instance, a camera installed in a TV room is no doubt very sensitive. The users are very willing to see alerts triggered by such devices and review warning messages. The other is privacy insensitive such as a smart TV. This type of device might be turned into part of a botnet but it does not intrude on privacy or affect normal use too much. Hence, it comes to another conclusion that the dashboard should also teach users why they should care about both types of alert and take responsibility for keeping their devices from attack by adopting security measures.

Features of post processing log files from IDS Slips, which had been processed by our algorithm to remove duplication and alert fatigue, were also considered for the design process. After combining existing data to use and knowledge gained from the interview, we are finally able to realize the GUI. The GUI has a navigation bar at the top and it has four pages, which are general information, statistical data, devices connected at LAN, and tips, respectively. Flexbox is used so that the size of the dashboard can adapt to the size of mobile devices' screen size.

### 4.3.1 Landing Page

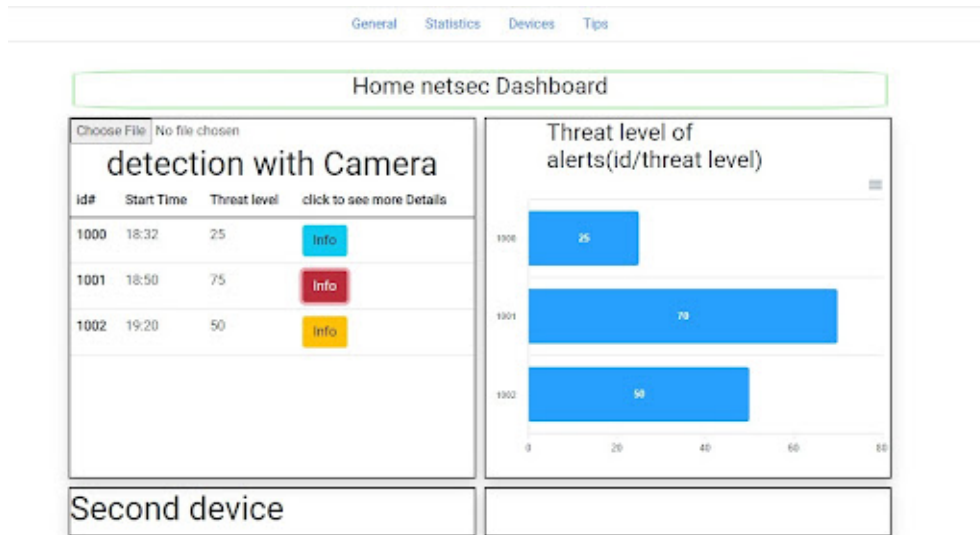


Figure 4.5: Landing page of the dashboard

For the landing page, the design idea is to give the user a summary of detections of his devices. The table will enumerate alerts pertaining to one particular device and list statistical data of alerts. The threat levels are also visually emphasized by the bar chart to help users quickly identify alerts with the highest severity. Colors are also employed to distinguish the seriousness of an intrusion. We also expect detections of other devices to be displayed like Camera as the picture shows.

After clicking on the button, we want the user to browse interpretations generated by algorithms and also the questions. As IDS cannot guarantee a network activity is malicious itself so we do not want to warn the user directly. Instead, a conversation-driven style is doable to involve both machine and user in decision making. We kindly ask the home user to think about anomalies that happened recently, given IDS generated an alert. Combining assistance from the IDS and instruction, the home users can make the final decision based on its judgment.



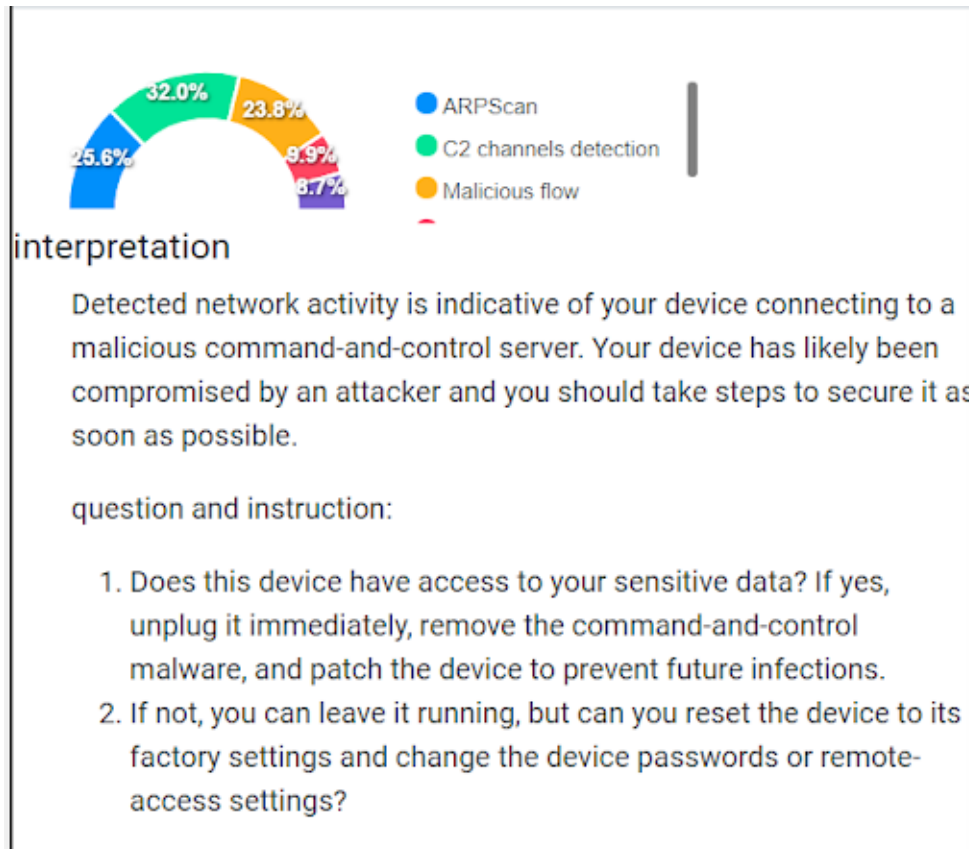


Figure 4.6: Instructions and interpretations after clicking on the button

Additionally, the user can see the composition of an alert. This could provide more background information about why the IDS believes the situation is worth investigating.

### 4.3.2 Statistics Page

The second page is to help users grasp the security situation from a high level. The users can see which type of attack appears the most by looking at modules that trigger alerts the most. They can also understand which devices suffer from attack the most by looking at the pie chart.

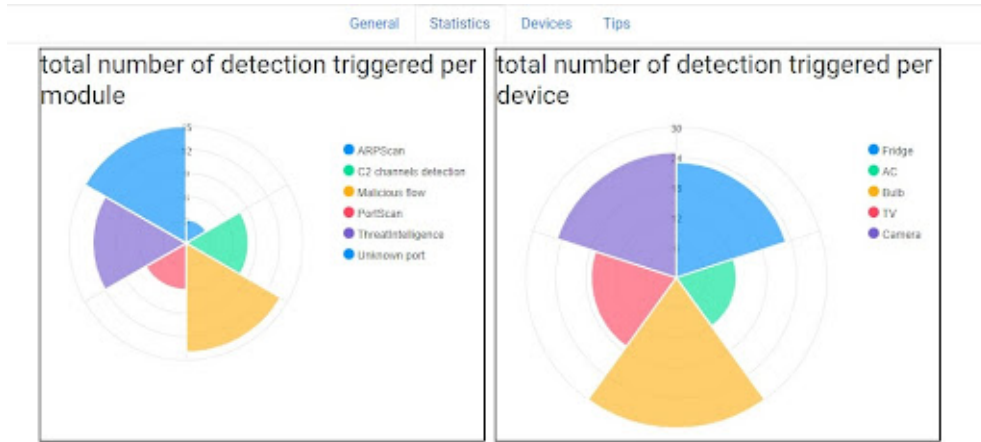


Figure 4.7: Statistics page

### 4.3.3 Devices Page

On the devices page, the user can see all devices connected to its homework. We assume users should also be able to customize privacy sensitivity and security sensitivity. Alert with the highest severity score that is associated with particular devices is included to remind the user for later reference. By clicking on the checkbox, traffic of that device should be dropped immediately. This is doable as suggested by Grimm, Bordeleau, and Wirkala [18]. [18] discovered that most IoT devices can only establish one TCP connection at a time. Therefore, there is no need to distinguish connections but simply drop traffic associated with that device.

The figure shows the 'Devices' tab with a table titled 'Devices connected in home network'. The table lists three devices: AC, Camera, and TV, with their respective privacy and security sensitivities, recent alert details, and a checkbox to block the device.

Name of device	privacy sensitivity	security sensitivity	Recent severest alert	Block this device
AC	low	medium	id: 9343, date: 12/22 18:32, severity: 40	<input type="checkbox"/>
Camera	high	low	id: 9443, date: 12/22 14:32, severity: 60	<input type="checkbox"/>
TV	low	low	id: 9443, date: 12/22 19:32, severity: 30	<input type="checkbox"/>

Figure 4.8: Devices page

### 4.3.4 Tips Page

Finally, the tips page is trying to provide more accessible background information for users to be able to diagonalize their own network health given alerts reported by IDS. The top left box walks the user through stages of hacking in plain language.

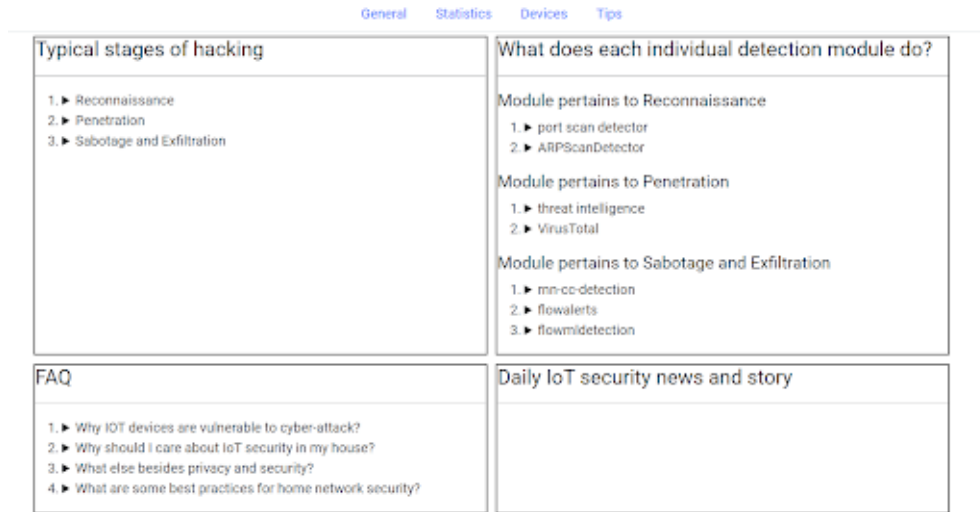


Figure 4.9: Tips page

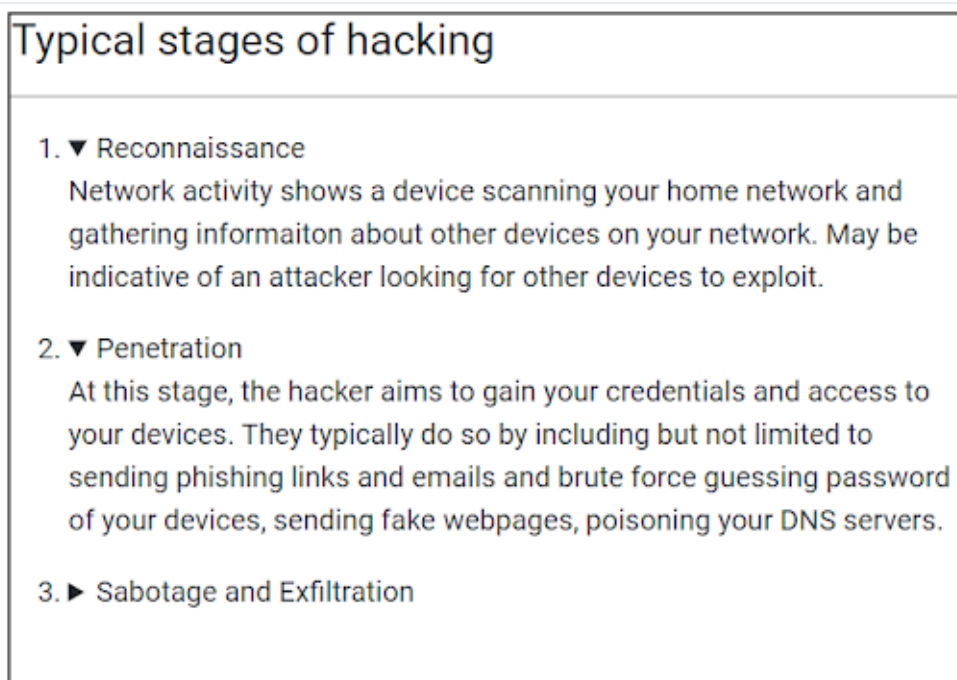


Figure 4.10: Stages of hacking

After clicking on the arrow, the bullet point expands and displays detailed information. The top right one helps the user understand how IDS work. Knowledge gained from reverse engineering plays a role here. The bottom left one applies the awareness-enforcement model. It makes users realize the significance of protecting their own network even not as network experts. In the end, as indicated by the interview results, users do care about security if they have a clear sense of the consequences of cyber-intrusion. Based on the assumption, we include this place to spread news and stories of IoT abuse and intrusion.

## 4.4 Use of IDS and Dashboard

By providing a packet capture file (PCAP) to the created prototype, the user can have a dashboard generated for them which shows detected threats and guides

the user through how to secure their network. The dashboard displays threats to and from specific devices and the network as well as metadata information about the threat such as time. Information is provided to the user giving them the option to learn more about a specific detection and steps that they can take to secure the device or verify that the detection is a false positive.

The dashboard also depicts the types of threats affecting the overall network and which devices on the network are the greatest threat. Displays such as these help a user understand which devices on their network they should pay attention to or potentially understand what detections are common and can be safely ignored. If ARP scanning is a common detection on the user's network, then they can learn that those detections are benign and have time to focus on more dangerous attacks.

## 4.5 Limitations

The prototyped IDS and dashboard do not yet allow for users to tweak the setting for their network. For example, the user cannot set the display to ignore ARP scans or increase the reported threat for particularly sensitive devices. The prototype also does not process threats live as they occur. Instead, a network packet capture needs to be run and then the generated PCAP file needs to be submitted. The signature detections generated by Snort only serve to warn the user of known pre-programmed threats. For example, the Snort rules used in the prototype can only detect malware C2 communications from the Ragdoll Payload (malware used in testing) and can only alert the user to known phishing sites. The inclusion of Slips serves to supplement this weakness by detecting general network misbehavior and learning a user's network.

The primary concern of using Slips is performance. Though we are not running

benchmark tests professionally and specifically, Slips can run over hours to process files whose size is over hundreds of MB. This is understandable because Slips compare current detection against historical detection to identify suspicious activity. However, this might not be scalable deploying at home for detections. Second, we cannot verify the exact false positive rate of each individual alert. This is due to many reasons such as we are not aware of details of how datasets we use are collected and under what circumstances. Otherwise, we have not decided the update frequency of the dashboard. Currently, we assume the dashboard should be updated daily or the amount of alerts and information could be overwhelming. More field work should be done to understand the amount of traffic a normal home user can produce on average.

Last, usability testing is integral to refine and adjust the interface based on stakeholders' feedback. Though following the best practice and realizing the GUI after incorporating stakeholders' thought, current dashboard are not tested against other home users who are not involved in the project. Therefore, it is imperative to run usability studies to understand if the dashboard can achieve the goal that we expect it to, such as awareness enforcement. The results are invaluable for making the next generation of the dashboard.

# Chapter 5

## Future Work

### 5.1 Home Network Threat Intelligence

The conducted research was done based on the idea that threats to home users are primarily phishing, Remote Access Trojans, and IoT device takeover. The decision to focus on these threats came from research. Still, a more relevant home IDS could be created with a better understanding of the adversary and increased research on related CTI (Cyber Threat Intelligence). Perhaps personal user anti-virus developers or home ISPs (Internet Service Providers) have conducted research into the threats facing home networks and could provide relevant CTI.

With access to home network CTI, the prototyped IDS could be improved to better target certain threats. For example if all phishing sites that target home users do not implement HTTPS, then a content signature could be written in SNORT to look for password fields over HTTP or other suspicious login features. Another example is that lateral movement inside a compromised home network may not be a common tactic used by adversaries - making detections for internal network scanning rather pointless. Overall, in order to build solid defenses, one first needs

to understand in exact detail what the existing attacks are.

## 5.2 Live Detections

As it stands, the prototyped IDS can only process data after the network traffic has been captured and uploaded in a PCAP file. It is of course unreasonable to ask a non-technical home user to run a full packet capture then export the file to some program in order to understand threats which may have been detected. A live version of the IDS should be developed so home users can simply check the current security of their network. Logging every single packet which crosses the network would rapidly overflow the storage of any reasonable system. In order to manage the data from a live IDS, captured packets should be stored temporarily in a large temporary cache until the cache is full and new packets can take their place. It may be possible for the IDS to still log summaries of captured packets before they are removed from the cache for the purposes of detecting stealthy drawn-out attacks. Such a method is still not perfect and a large flood of packets could be used in DoS (Denial of Service) attack against the IDS. The flood of packets could potentially fill the entire cache over-writing any other packets and allowing for a threat actor to hide their true attack.

## 5.3 More Research For SLIPS and Similar AI Systems

The first question to be answered is the performance of IDS such as Slips. Aforementioned time consumption record has proven that it takes a non-trivial amount of time to fully process the pcap file. The situation could change if live



capturing and real-time processing is enabled. To make the entire system deployable in real-life situations, more experiments are expected to understand the processing speed of the IDS, which leads to another question of where to store or cache the network traffic and how much storage is going to be used.

The second research interest is the exact FP (true positive) and FN (false negative) rate of the IDS. This might require the research team to simulate all traffic so they understand every aspect and detail of the PCAP. Our project focused on existing IoT traffic simulated by other institutes. The result could make a difference with both post-processing algorithm and interface.

## 5.4 Improvements to Signature Detections

Threat intelligence companies are constantly researching current attacks and infrastructure used by cybercriminals. The compiled research and attacker signatures can be used to, among other things, create signatures for network defenses. Domains, IP addresses, and file hashes are commonly referred to as IOCs (Indicators of Compromise) in the threat intelligence community. There are many services which offer network IOCs for the purposes of blocklists or detections, there are even several free blocklists such as those provided by Abuse.CH or Honeypot.FYI. Many of these domains and IP addresses could be easily added to the SNORT rules used by the prototyped IDS for the purposes of alerting the user to a potential compromise.

There are other improvements to our SNORT rules that we originally envisioned but did not have time to implement. While our rules did detect malicious lab packets, these rules only focused on elements such as IP and port numbers. In the C2 SNORT rules, these rules also included the ability to match specific potentially malicious bytes in the content of the packet. While these rules are enough to generate

basic alerts, we would have liked to have done more research into the depth of these alerts, looking into other functionalities SNORT rules are capable of in terms of detecting malicious traffic. This would have also been in tandem with thorough research in the structure of different types of malicious attacks on home networks, specifically looking into how each attack works and based on that, determine what types of SNORT rules would be best suited to detecting those attacks. This would have culminated with our implementation of these SNORT rules into the dashboard, using any information and methods from what would have been learned in order to best communicate alerts to the user.

## **5.5 Survey Non-Technical Users on Preferred Dashboard and Understanding**

During our background research, one of our team members conducted a small study looking into what non-technical users would like to see. This study primarily involved them asking their roommates various questions, and details of the question could be found in the appendix. While this study was useful for creating our prototype dashboard, due to the small sample size (3 people), it was not a comprehensive representation of the general public of non-technical users. Relevant field work that could be done is to create some sort of survey meant for a larger sample size, perhaps around 100 people or greater. This method provides two advantages. We would have been able to dedicate time into creating specific survey questions which would have been based on our research. These questions would have been designed in such a way so that the person taking the survey understands what the question is asking and that they'll provide a non-biased response. The other benefit would be from the larger sample size, in which we would need to figure out who we want

to distribute the survey to and how we want to distribute the survey. With these methods, we could create a successful survey regarding what non-technical users think of the dashboard, however, these methods take extensive time, something we didn't have when we came closer to the end of our project.

# Chapter 6

## Conclusion

The current threat landscape involves home users being targeted by cybercriminals attempting to exploit user weaknesses for monetary gain. Home networks are becoming increasingly complex with new IoT devices and home users relying on their computers more for both entertainment and work. The home network complexity adds to user technical confusion and increases the attack surface against home networks. Attackers are attempting to install malware on user personal computers, take-over IoT devices for botnets, and scam users digitally - among other attacks.

Home networks do not have adequate defenses to stay secure in the current threat landscape with such advanced network threats. Many home users are simply confused about how to keep their network secure. Some attempt to protect themselves by running anti-virus software on their computer and some Internet Service Providers attempt to block malicious Internet connections. Neither of these defenses adequately protect home users from the varying threats of modern attacks. An Intrusion Detection System, however, reverses an attacker's advantage by catching them as they attempt to carry out their malicious actions.

In conclusion, the team approached the problem of designing security tool for

home network by identifying appropriate and representative IDS, launching typical attack to simulate intrusion scenario, gathering existing IoT devices' network traffic log files, designing algorithm based on outputs of particular IDS to remove duplicate and summarise useful statistical data, and finally incorporating above effort to realize a prototype, envisioning an extensible and reusable design of home network monitoring system.

Though not covering all types of IDS or other defense strategies, our choice of IDS includes two most widely adopted and deployed IDS, i.e., signature-based and anomaly based tools. Therefore, our work outlines one of solutions to handle challenges of interpreting output of those types of IDS. Our algorithm successfully compresses thousands lines of evidence and alerts by IDS to only several informative records. Lastly, our dashboard provides a way of visualizing processed data, displaying relevant information that concerns home users. We recognized the need to involve the users in the whole security related decision process and further empower the users to do so by sharing accessible knowledge of cyber-security.

Although not all of our ideas for this project made it to the final deliverable, there are some next steps we could recommend to future teams working on this or similar projects. While our research into possible attack vectors on a home network is useful for our project, it's not guaranteed to be an accurate look into the threats home users would face, in which we could have utilized relevant Cyber Threat Intelligence (CTI). We also could have implemented a way for our prototype to utilize live detections, so that a live system being compromised would be flagged by our IDS, which in turn the alerts would be pushed to the dashboard. With our IDS, we've noticed there are possible improvements to our implementations of Slips and SNORT. Finally, our implementation of the dashboard and the language of our alerts could benefit from a medium to large scale survey with non-technical users

on what they would find useful in our prototype.

# Chapter 7

## Appendix

In this section, we include questions we asked during the interview, screenshots to foster understanding, and relevant commands to use Slips.

Questions to understand users' thought on what a GUI of dashboard should look like and what functionality it should have:

1. What platform do you expect the dashboard to base on? (desktop application, mobile app, web pages).
2. In what ways do you want to receive alert messages? (messages, phone call, email, app notification, etc)
3. What should a notification include?
4. Any features you want to be able to customize yourself?
5. What should the home pages highlight?
6. What is the most effective way to convince you that security matters?
7. How frequent do you expect to receive notifications of alerts without being fatigued?

8. Any other questions?

Commands to enter the container from a new terminal: Open a new terminal, run **docker ps** to find the id of the container and **docker exec -it ;container\_id; bash** to enter the container.

Commands to copy results from container to local file system: **Docker ps** to locate ID of the container **docker cp e48c5981c7ce:/StratosphereLinuxIPS/output C:\MQP\StratosphereLinuxIPS\results\output\_calderaC2\** to copy files to local file systems Display limited lines of files **sed -n 1,10p <file name>**



```

# [3.2] The width of the time window used
# 1 minute
time_window_width = 60
# 5 min
time_window_width = 300
# 1 hour
time_window_width = 3600
# 1 day
time_window_width = 86400
# One time window only. Is like if not time windows were used. Beware that the names of the files for the TB have a year in the name that is 100 years back.
time_window_width = 'only_one_tb'

```

Figure 7.1: Time window length available to adjust in Slips' configuration file

```

# [3.5] Analyse only what goes OUT of the home_net? or also what is coming IN the home_net?
# Options: out, all
# In the _out_ configuration we only pay attention to what each IP in the home net _produces_. We look at the traffic _originating_ from the home net only. The behavior of each IP. If
# its attacked from the outside we don't process that
analysis_direction = out

```

Figure 7.2: Detecting direction

```

#####
# [4] Configuration for the detections
[detection]
# This threshold means: minimum confirmed attacks per minute needed to generate an alert
#evidence_detection_threshold = 0.1
#evidence_detection_threshold = 0.25
evidence_detection_threshold = 1
#evidence_detection_threshold = 2
#evidence_detection_threshold = 5

```

Figure 7.3: Detection threshold

```

#####
# [9] configuration for long connections detection module
[flowalerts]

# we need a thrshold to determine a long connection. in slips by default is.
long_connection_threshold = 1500

ssh_succesful_detection_threshold = 4290

# threshold in MBs
data_exfiltration_threshold = 700

```

Figure 7.4: Other adjustable threshold

# Bibliography

- [1] Alert fatigue: Imperva. Alert Fatigue. (2020, December 9). Retrieved December 10, 2021, from <https://www.imperva.com/learn/data-security/alert-fatigue/>.
- [2] Yu, Miao, et al. “A Survey of Security Vulnerability Analysis, Discovery, Detection, and Mitigation on IoT Devices.” *Future Internet*, vol. 12, no. 2, MDPI AG, 2020, p. 27–, <https://doi.org/10.3390/fi12020027>.
- [3] Al-Alami, H.; Ali, H.; Hussein, A.B. Vulnerability scanning of IoT devices in Jordan using Shodan. In *Proceedings of the 2nd International Conference on the Applications of Information Technology in Developing Renewable Energy Processes & Systems (IT-DREPS)*, Amman, Jordan, 6–7 December 2017.
- [4] Shodan. (2009). Retrieved 10 November 2021, from <https://www.shodan.io/>
- [5] What is an Intrusion Detection System (IDS)? — Fortinet. Retrieved 10 December 2021, from <https://www.fortinet.com/resources/cyberglossary/intrusion-detection-system>
- [6] What is IDS and IPS? — Juniper Networks. Retrieved 3 November 2021, from <https://www.juniper.net/us/en/research-topics/what-is-ids-ips.html>

- [7] Yisroel Mirsky, Tomer Doitshman, Yuval Elovici, and Asaf Shabtai. 2018. Kit-sune: An Ensemble of Autoencoders for Online Network Intrusion Detection. In NDSS. Internet Society.
- [8] Rupesh K Srivastava, Klaus Greff, and Jurgen Schmidhuber. Training very deep networks. In Advances in neural information processing systems, pages 2377–2385, 2015.
- [9] Pardis Emami-Naeini, Henry Dixon, Yuvraj Agarwal, and Lorrie Faith Cranor. 2019. Exploring How Privacy and Security Factor into IoT Device Purchase Behavior. Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems. Association for Computing Machinery, New York, NY, USA, Paper 534, 1–12. DOI: <https://doi.org/10.1145/3290605.3300764>
- [10] Nuamah, J., & Seong. Y. (2017). Human Machine Interface in the Internet of Things (IoT). A paper presented at the 2017 IEEE System of Systems Engineering2017 Conference, Waikoloa, Hawaii, USA
- [11] Hunt, David. “Caldera Readme.” GitHub, MITRE AT&CK, 28 Nov. 2021, <https://github.com/mitre/caldera/blob/master/README.md>.
- [12] Strom, Blake. “The Philosophy of AT&CK.” The MITRE Corporation, AT&CK, 24 July 2018, <https://www.mitre.org/capabilities/cybersecurity/overview/cybersecurity-blog/the-philosophy-of-attck>.
- [13] Duncan, Brad. “Traffic Analysis Exercises.” Malware-Traffic-Analysis.net, 4 Nov. 2021, <https://www.malware-traffic-analysis.net/training-exercises.html>.

- [14] Robertson, G., Czerwinski, M., Fisher, D., & Lee, B. (2009). Selected human factors issues in information visualization. . *Reviews of human factors and ergonomics*,5(1), 41-81
- [15] Gutzwiller, R. S., Fugate, S., Sawyer, B. D., & Hancock, P. A. (2015, September). The Human factors of cyber network defense. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting (Vol. 59, No. 1, pp. 322-326)*. SAGE Publications.
- [16] Best, D. M., Endert, A., & Kidwell, D. (2014, November). 7 key challenges for visualization in cyber network defense. In *Proceedings of the Eleventh Workshop on Visualization for Cyber Security (pp. 33-40)*. ACM
- [17] Seong, Younho & Nuamah, Joseph & Yi, Sun. (2020). Guidelines for cybersecurity visualization design. 1-6. 10.1145/3410566.3410606.
- [18] Grimm, J., Bordeleau, D., & Wirkala, R. (2019). *Graceful Degradation in IoT Security*. : Worcester Polytechnic Institute.
- [19] Snort - Network intrusion and prevention system. Retrieved 14 September 2021, from <https://www.snort.org/>
- [20] Home - Suricata. (2021). Retrieved 14 September 2021, from <https://suricata.io/>
- [21] The Zeek Network Security Monitor. (2021). Retrieved 14 October 2021, from <https://zeek.org/>
- [22] Lorenzo De Carli, Indrakshi Ray, and Erin T. Solovey. 2021. Vision: Stewardship of Smart Devices Security for the Aging Population. In *EuroUSEC '21*:

European Symposium on Usable Security, October 11–12, 2021, Online. ACM, New York, NY, USA, 9 pages.

[23] Hassan, W.U., Guo, S., Li, D., Chen, Z., Jee, K., Li, Z., Bates, A.: NoDoze: combatting threat alert fatigue with automated provenance triage. In: Network and Distributed Systems Security (NDSS) Symposium 2019 (2019)

[24] XFi Advanced Security, what should I know?. (2020). Retrieved 14 August 2021, from [https://www.reddit.com/r/Comcast\\_Xfinity/comments/jkls7z/xfi\\_advanced\\_security\\_what\\_sh](https://www.reddit.com/r/Comcast_Xfinity/comments/jkls7z/xfi_advanced_security_what_sh)

[25] Axelsson, S. (2000). The base-rate fallacy and the difficulty of intrusion detection. *ACM Transactions On Information And System Security*, 3(3), 186-205. doi: 10.1145/357830.357849

[26] What is the Mirai Botnet?. Retrieved 18 September 2021, from <https://www.cloudflare.com/learning/ddos/glossary/mirai-botnet/>

[27] xFi Advanced Security is blocking my company's website. (2020). Retrieved 15 September 2021, from <https://forums.xfinity.com/conversations/xfinity-app/xfi-advanced-security-is-blocking-my-companys-website/602dae20c5375f08cddf28af>

[28] McCann, D.; Eder, K.; Oswald, E. Characterising and comparing the energy consumption of side channel attack countermeasures and lightweight cryptography on embedded device. In *Proceedings of the International Workshop on Secure Internet of Things (SIoT)*, Vienna, Austria, 21–25 September 2015.

[29] Empowering App Development for Developers — Docker. (2021). Retrieved 23 September 2021, from <https://www.docker.com/>

- [30] Sebastian Garcia, Agustin Parmisano, & Maria Jose Erquiaga. (2020). IoT-23: A labeled dataset with malicious and benign IoT network traffic (Version 1.0.0) [Data set]. Zenodo. <http://doi.org/10.5281/zenodo.4743746>
- [31] Booij, Tim M., Irina Chiscop, Erik Meeuwissen, Nour Moustafa, and Frank TH den Hartog. "ToN IoT-The role of heterogeneity and the need for standardization of features and attack types in IoT network intrusion datasets." *IEEE Internet of Things Journal* (2021).
- [32] Stratosphere IPS for Linux — Stratosphere IPS. Retrieved 29 September 2021, from <https://www.stratosphereips.org/stratosphere-ips-for-linux>
- [33] GitHub - stratosphereips/StratosphereLinuxIPS: Slips. A machine learning-based Intrusion Prevention System (IDS/IPS). Free Software. Stratosphere Laboratory. (2021). Retrieved 27 September 2021, from <https://github.com/stratosphereips/StratosphereLinuxIPS/tree/master>
- [34] Training - Slips 0.8.2 documentation. (n.d.). Retrieved November 15, 2021, from <https://stratospherelinuxips.readthedocs.io/en/develop/training.html?highlight=vector+machine>
- [35] Ait Maalem Lahcen, Rachid, et al. "Cybersecurity: A Survey of Vulnerability Analysis and Attack Graphs." *Mathematics and Computing*, vol. 253, Springer Singapore, 2018, pp. 97–111, [https://doi.org/10.1007/978-981-13-2095-8\\_9](https://doi.org/10.1007/978-981-13-2095-8_9).
- [36] What is Cyber Situational Awareness? — reciprocity. What Is Cyber Situational Awareness? (n.d.). Retrieved 2021, from <https://reciprocity.com/resources/what-is-cyber-situational-awareness/>
- [37] Pino, R.E.: *Cybersecurity Systems for Human Cognition Augmentation*. Springer, New York (2014)

- [38] Zhao, Hanning, and Bilhanan Silverajan. “A Dynamic Visualization Platform for Operational Maritime Cybersecurity.” *Cooperative Design, Visualization, and Engineering*, Springer International Publishing, 2020, pp. 202–08, [https://doi.org/10.1007/978-3-030-60816-3\\_23](https://doi.org/10.1007/978-3-030-60816-3_23).
- [39] Akhawe, D., & Felt, A. P. (2013, Augustus). Alice in Warningland: A Large-Scale Field Study of Browser Security Warning Effectiveness. 22nd USENIX Security Symposium (USENIX Security 13), 257–272. Opgehaal van <https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/akhawe>
- [40] Kritzinger, Elmarie & Solms, S.H.. (2010). Cyber security for home users: A new way of protection through awareness enforcement. *Computers & Security*. 29. 840-847. 10.1016/j.cose.2010.08.001.
- [41] Compromised Personal Network Indicators and Mitigations. (2020). National Security Agency. [https://media.defense.gov/2020/Sep/17/2002499615/-1/-1/0/COMPROMISED\\_PERSONAL\\_NETWORK\\_INDICATORS\\_AND\\_MITIGATIONS\\_20200914](https://media.defense.gov/2020/Sep/17/2002499615/-1/-1/0/COMPROMISED_PERSONAL_NETWORK_INDICATORS_AND_MITIGATIONS_20200914)
- [42] A look into the most noteworthy home network security threats of 2017. (2018, February 27). Trend Micro; Trend Micro Incorporated. <https://www.trendmicro.com/vinfo/us/security/research-and-analysis/threat-reports/roundup/a-look-into-the-most-noteworthy-home-network-security-threats-of-2017>
- [43] Facebook—Log in or sign up. (n.d.). Facebook. Retrieved December 16, 2021, from <https://www.facebook.com/>

- [44] Home Network Security. (n.d.). Trend Micro; Trend Micro Incorporated. Retrieved December 16, 2021, from [https://www.trendmicro.com/en\\_us/forHome/products/homenetworksecurity.html](https://www.trendmicro.com/en_us/forHome/products/homenetworksecurity.html)
- [45] Rubenking, Neil J. (2020, May 27). Trend Micro Home Network Security Review. PCMAG; Ziff Davis. <https://www.pcmag.com/reviews/trend-micro-home-network-security>