A framework for en masse network security evaluation and network flow analysis for the Internet of Things era.

Nikolaos Tsirigotakis

Master of Science

Computer Science

School of Informatics

University of Edinburgh

2016

Abstract

The transition from an Internet of inter-connected computers operated by people, to an era of inter-connected autonomous devices is inevitable. This new era of Internet of things (IoT) is characterised by rapid expansion while the main aspect of IoT is the use of several standards, protocols and technologies making the security evaluation on a per device scenario time consuming. The number of devices introduced is expected to reach billions in the future and the current literature is well informed about the insecure design of many devices.

There is no platform or framework that enables the mass evaluation of devices, thus making the mass evaluation of the numerous devices impractical. The goal is not just the identification of what can be considered secure, and how can this be tested on a large scale, but also the implementation of the framework.

This paper introduces a new framework that enables the automation of security checks and vulnerability scanning while providing network flow behaviour analysis capability in real-time, in a scalable and expandable manner based on open source technologies, enabling new ways of interaction with network data and security assessment.

Acknowledgements

I would like to thank first of all my parents for their investment in my education. My supervisor Kami Vaniea for the original idea and her invaluable help during the life-cycle of this project. Last but not least, all the people that helped me with this project with their suggestions, input and patience.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(Nikolaos Tsirigotakis)

Table of Contents

Figures

Tables

Chapter 1
Introduction

The first time the words “Internet of Things” (IoT) were mentioned was in 1999 during a presentation from Kevin Ashton, as he used the term to describe the use of Radio-Frequency Identification (RFID) technology in Procter and Gamble products. [1], [2] Since then, the rapid advancement of technology resulted in a trend of building networked appliances, which surpassed the original usage of the term. At the time of writing, this range includes from radio tags to internet-enabled sensor platforms and embedded computers [3], and the list of devices with network connectivity capabilities keeps on expanding. A growing number of start-ups and well established companies are now pushing their products towards the IoT age. Cisco estimates that 50 billion IoT devices will be connected online by 2020 [4].

The current state of security in the IoT field and these billions of devices is uncertain mostly due to the large number of devices and fragmented market that follows no guidelines. Lacking adequate power many devices do not implement security measures such as encryption and their custom implementations sometimes do not consider textbook attacks such as buffer overflows. Following the rules of evolution, security matures as the operating system and software matures as well. While the security by design is becoming common place in mainstream operating systems and applications, the custom implementations of IoT devices are neither mature and in most cases not designed with security in mind as well.

Some of the current research focuses mostly on the design challenges. A large number of devices mostly rely on low power cheap microcontrollers. Initial results on per case experiments conducted to validate the hypothesis upon which this project is based on a small subset of the devices that need to be tested has revealed a disregard for basic security concepts such as the use of Transport Layer Security (TLS) and authentication. While this fact has been mentioned in the past[5], and despite that researchers have proposed alternative security concepts adapted to the nature of the IoT[4]–[6], their proposals never reached the manufactures design tables.

1.1 Motivation

While most of the literature is researching new security and authentication mechanisms and frameworks for the new era[7] [8] [9][10][11][4], others highlight that IoT security problems have known solutions which are not implemented undoing the security gains of the last 25 years[12]. Problems that were solved decades ago are still present, in an internet scan of SSH and HTTPS enabled devices that expose their services over the internet found that 150 server keys were shared by over three million devices while eighty (80) keys were used by almost one million devices[13]. During a mass security evaluation contacted by HP in 2015, the results were painting the picture of the current state of security and privacy in the IoT domain. According to the HP study 90 percent of the devices tested were collecting at least one piece of personal identifiable information, over 80 percent of the devices did not implement some authentication mechanism, while 70 percent were using unencrypted network services[14].

The security of the devices that flood the market is doubtful and the sheer volume of them makes their security evaluation almost impossible on a per device manner. While companies like HP and Tenable offer security suites that are capable of mass security evaluation and network analysis, and their results as noted previously show the extend of the problem, no such framework exists for the broader crowd of security and network professionals.

1.2 Aims

The main aim of this paper is helping the analysis and visualisation of network flows and packets as well as the enablement of easy and mass security evaluation is the main goal of this project. To achieve this goal, the following questions need to be answered:

1. What are the problems with the current approach?
2. How to implement generic mass security assessment?
3. How to enable easy access to network packet flow data in real time?
4. How secure are the devices currently in the market?

It is crucial to make it easier for both experts and not alike to analyse the behaviour and security of IoT devices. In total four are the aims that need to be achieved:

1. Create a detection procedure and a set of rules that detects possible attacks and privacy leaks.
2. Create a live data capturing procedure that enables behaviour analysis and security evaluation in real time.
3. Create an easy to use API which exploits the previously mentioned mechanisms, is the logical next step in order to provide a toolset that will make an impact in the future of the field. More specifically it aims to both provide an API that allows live packet manipulation and session simulation in order to analyse and map the behaviour of the devices, as well as the execution of simplified mass security auditing.
4. A review of IoT device security on a small subset of devices using the framework.

1.3 Overview

This research project has eight chapters in total, in chapter two the background highlighting the need for the framework is explored. The third chapter includes the design requirements that were set for the framework and the focus group that was used to validate the functionality and design that is proposed. Chapter four defines the concept of security and privacy, and includes an overview of the logic used to implement the framework. Chapter five has an overview of the software needed, the architecture and implementation of the framework. Chapters six and seven are the evaluation part of the project, where both the framework is evaluated by experts and devices get evaluated by the framework. The conclusion and the potential future expansion and improvement are discussed in the last chapter .

Chapter 2
Background

The IoT is a relatively new sector that bloomed during the last years. The research around devices was mostly dominated from the introduction of new security and authentication models. The following chapter will present an overview of the current state of the IoT devices security, explore the lack of a standardised framework, define the threat model and lastly introduce generic attacks and ways to verify the security of a device.

2.1 IoT devices security

The range of possible attacks is only limited by the power the devices hold and the kind of available exploits. Unlike traditional computer systems where the attacker could exploit the machine but not the environment, IoT devices have a range of usages and usually are embedded projects, which allow the manipulation of their environment.[15] That makes devices that were not exploitable by design to be exploited. An example of such an attack that breaches the digital isolation and enters the physical world would be an attack to a “smart” Wi-Fi controlled power socket, if an attacker manages to manipulate the socket they can control the machine attached to it, this fact creates new serious security and safety endangering network attacks that can now reach to the physical world.

The attacks of the past have a large range from infiltrating a smart TV and record the owners from its camera[16] to kettles that can inform the hacker about the WIFI password[17], [18]. During 2015, 1.4 million vehicles were recalled because hackers could control them remotely, and even turn off the engine and control the steering wheel[19]. The number of attacks is vast and on a number of domains.

The shodan search engine[20] is a prime example of the vast insecurity and wrong default configuration present in devices. It was developed and introduced in 2009 and

is described as a computer search engine, however in reality shodan is very different than a typical search engine, and it is closer to a mass security evaluation framework. It works by scanning for active hosts, and when one is discovered it does a port-scan and service identification, the information extracted is then indexed for searching[21]. The search engine exposes services that the owners of the devices thought as secure or not available and thousands of devices with no or default passwords can be found. Using the webcam search feature someone can find unsecured cameras from various locations including back rooms of banks and baby cribs[22].

The attacks are tailored in per device scenario in most of the cases, but repetition of past mistakes during the design and quality assurance of the devices have as a result insecure devices to reach the market. Devices such the ones that can be found on shodan, endanger the security and privacy of their users.

2.2 The need for a framework

The framework researched, is a multi-layered system that incorporates various technologies. Relative work as the framework aimed in this paper is limited but a research exists that cover parts of its functionality in a non-usable form for this project, but highlights the interest around these areas and the problems faced by researchers. Network data flow collection and visualisation monitors are being used for years to represent information through graphical means. There is a number of implementations in the literature, using various techniques to acquire and display the data, while the need for security as a service has been discussed before[23].

Mass security evaluations and network behaviour analysis have been conducted in the past, but the researchers had to rely on either proprietary data dumping systems[24] or tools like wireshark and tcpdump, which deprive their implementations of the live aspect of evaluation. [25],[26]. Others implemented security frameworks using the tools publically available to collect the data needed, but didn't provide a way to access the information other than the included application[27]. Other examples don not scale or don not allow per packet analysis[28],[29]. Some solutions present in the literature explore various possibilities to enable the network flow visualisation and information

through open standards, but require the use of special hardware such as a special network card[30].

While the idea of mass security evaluation is not new, the lack of a scalable and expandable mass security evaluation framework had as a result the need for researchers to define new frameworks and techniques in order to obtain quantitative results

The lack of network behaviour and security analysis is apparent, many tools already exist, such as Wireshark and tcpdump, but these tools target network professionals and are not capable of sharing the data in a way that the information can be consumed by another service. Proprietary solutions which dominate the area[31] are not agile, since they need special hardware and are not expandable.

The IoT is based on existing technologies and the manufacturers need to use several standards to comply with the myriad of different usage scenarios. It is very difficult to go through all these standards and technologies in order to find which to target. While a number of standards have been introduced, some researchers are concerned by the lack of standards and frameworks that cover all the aspects of security in the IoT naming it as an actual issue in conduction of research[32]. In this project it is chosen to focus on initiatives related to "generic" standards of the IoT. An open expandable framework could potentially fuel more research in the field and allow easier evaluation of devices.

2.3 Threat model

The targeted devices are networked mass consumer electronics that find their way in most consumer homes. Due to the individuality of the devices a generic framework is needed, the only thing all the targeted devices share is the networking capability, hence the framework targets to verify whatever a device is secure against common networking attack scenarios in a local area network, in order to simulate an average end-user network attack scenario Consumer electronics such as smart clocks, light bulbs and other network controlled devices are targeted.

A typical home Local Area Network (LAN) attack scenario where the attacker can tamper the characteristics of a link between two devices[33] is considered. The attacker controls a single device in the network and is able to ping the devices around her. It is assumed that the attacker has penetrated the network and is capable of man-in-middle attacks (MITM)[34]. The attacker is able to route traffic through her device, practically stripping the devices from the router's firewall and NAT protection. An attack of this nature is possible when the attacker either controls the access point itself or is capable of attacks, such as ARP poisoning which allow her to impersonate the target device[35]. The primary target of the attacker is considered the Transport Layer Security (TLS) employed by most devices to defend against man-in-the-middle attacks[36] as well as the HSTS mechanism employed by compatible browsers and web services to deter the attacks against the TLS encryption.[37]

2.4 Attacks

The main target of the attacks presented, is to verify the implementation of the network stack and available services of the devices by using attacks that target a wide range of systems. These attacks were chosen because they exploit implementation mistakes present at diverse sets of devices and platforms based on common programming errors or misconfiguration. This allows a more generic approach that with the complement of a vulnerability scanner allows to validate against most MITM attack scenarios.

2.4.1 LAND

The Local Area Network Denial (LAND) attack is executed by sending special spoofed TCP SYN packets that have both the sender and receiver fields set as the target's IP. This causes the device to potentially initiate a session with itself, thus overloading and crashing it. While it is a Denial of Service (DoS) attack it is distinct from traditional DoS attacks because the attacker does not need to possess the capability to exhaust the targets open connections or bandwidth but rather relies on the incomplete implementation of the network stack. [38], [39]

2.4.2 IP fragmentation

The datagram fragmentation mechanism is used to divide datagrams larger than the network's Maximum Transfer Unit (MTU). These smaller datagrams fit the frame size of the network and upon arrival to the destination they get reassembled. [40]

Six distinct attacks exploit this mechanism. The main idea of the attack is to create special custom datagram fragments with over-sized payloads that when the targets tries to reassemble, they overlap. This can lead to buffer overflows, DoS or crashing the device completely if there is no security or exception catching mechanism in place. Targeted protocols vary, but the publicly available source code is capable of attacking the TCP, UDP, SMB and ICMP protocols. Exploits based on IP fragmentation were last reported in a major operating system in 2009 in Microsoft Windows Vista [41](CVE-2009-3103), while a range of old Linux Kernel, BSD and Mac OS X versions are known to be vulnerable.

2.4.3 TLS/SSL implementation verification

The number of bits and type of hash used in the key have a crucial role in the security of the device, the usage of unsecure hashes such as RC4 undermine the overall security of the design. Advances in brute forcing weak ciphers make implementations such as RC4 and DES insecure and hence their use should be avoided. Any version of TLS predating TLSv1.0 should be considered insecure as well[42], [43]. The use of known insecure parameters in the design will have an impact on the privacy rating of the device. Good practice will be checked instead of bad one in order to avoid unknown bad configurations.

2.4.4 CRIME & BREACH

CRIME (Compression Ratio Info-leak Made Easy) and BREACH (Browser Reconnaissance and Exfiltration via Adaptive Compression of Hypertext) are used to target devices that offer a web interface either locally or remotely, while most mainstream browsers are immune to the attack, custom implementations (e.g. custom web-view style smart-phone apps, or a Smart-TV browser) can be exploitable. Both

attacks exploit the compression mechanisms in TLS and HTTP with the difference being which protocol they target. In their current state both attacks are capable of bypassing the encryption and steal session cookie data. While executing the actual attacks is complicated, it is easier to check for a vulnerable server[44], [45], [46], [47].

2.4.5 DROWN

The DROWN attack is a cross-protocol attack on servers supporting the obsolete, insecure, SSLv2. It targets TLS protocols that would be otherwise secure if SSLv2 support was not present. It achieves this by allowing the attacker to break a passively collected RSA key exchange for any TLS server which shares the keys between TLS and SSLv2. Since the attack does not require any bug to be present and relies on SSLv2 flaws, any server using this version is vulnerable. The researchers estimated that there were 11.5 million HTTPS servers vulnerable to the attack. There are numerous ways to execute the attack with the most common being through an OpenSSL vulnerability[48] (CVE-2015-3197), which allowed the supposedly removed SSLv2 to be selected by clients although it was not offered by the server. In a MITM attack scenario, the attacker can impersonate the server and send a ServerHello message that selects a cipher suite with RSA as the key-exchange method, then decrypts the premaster secret with DROWN. The main difficulty reported by the researchers was completing the decryption and producing a valid ServerFinished message before the client's connection times out. [49]

2.4.6 SSLstrip

In 2009 a hacker released a tool called SSLstrip and showcased it in Black Hat DC the same year[50]. The idea was simple, when a user types an incomplete URL in the form www.example.com browser requests an http connection at port 80 by default (http://www. example.com) the script intercepts the traffic and rewrites all future links to http so that the user never gets an https session. To combat this attack, the HTTP Strict Transport Security (HSTS) was created[37]. HSTS introduced a header which is sent over HTTPS which informs the compatible clients to only connect to this domain using HTTPS for a period of time which can be up to a year. This effectively killed

the attack since, if a user managed to connect once to a HSTS enabled site, then all her future requests will be using the encrypted connection. Since then various forks of the tool exist that try to overcome the HSTS protection. The attack is considered despite its age because advances have been made to combat HSTS and these will be presented later in the paper.

Chapter 3
Design requirements

3.1 Proposed system

The proposed system consists of a custom built router which includes all the needed software and is driven by an API that enables other services and products, that consume it, to be created. The target base is so diverse that a generic and multi-layered approach is chosen. The core needs of the project are three:

- Collecting packets.
- Configuration-free security analysis.
- An API that would allow the automation and data retrieval of the data gathered from previous two.

An overview of the proposed requirements follows for each of the three categories. It should be noted that the security analysis has no system requirements since it is considered both part of the API and the router.

3.1.1 Packet collection

Packet collection is the core of the functionality. A router with the built-in capability to dump network traffic would allow easy integration with existing systems hence it is the targeted design.

System Requirements

- WIFI access.
- External hard drive to save data.
- Open source operating system.

Functional Requirements

- Packet collection to a PCAP file.
- Packet collection to a database.
- Almost real-time capability.

3.1.2 Configuration free security analysis

The automation of the security evaluation is the second part of the framework and with the packet collection they form the core of the system.

Functional Requirements

- Automated TLS/SSL analysis.
- Automated vulnerability testing.
- Automated port scanning.
- Automated service detection.

3.1.3 API

The API is used to automate the previous components and to allow access to the data saved. The target functionality is to provide enough information, drive a web-based graphical user interface and has the following requirements:

System Requirements

- Portable and cross platform.
- Self-contained.

Functional Requirements

- The overview of current and past connected devices.
- The overview of current and past network connections.
- The simulation of the past network sessions.
- The vulnerability and encryption scanning of devices.
- The production of a simplified security score.
- Any device connected to the router has access to the API.

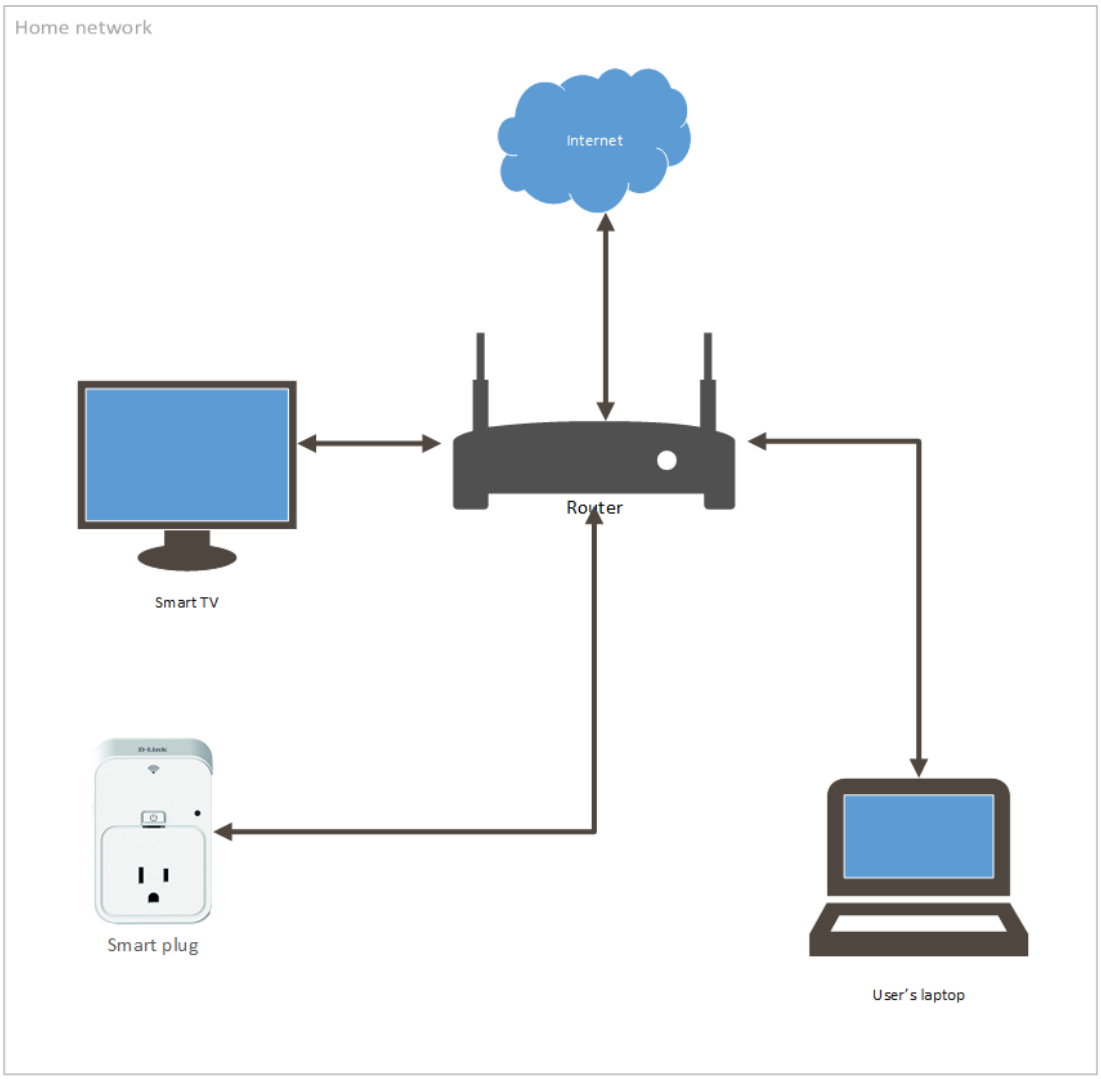


Figure 1. Graph of the proposed system.

3.2 Focus Group & survey

External input was deemed necessary during the initial phase of the design process, in order to take into consideration more than one perspectives before the finalization of the design. In order to define the specifications needed, a front-end mock-up was designed which would define the minimal functionality of the framework.

3.2.1 Setup and Materials

A new design was conceptualised, requirements were set and a survey based on the Likert-type scale response anchors by Vagias Wade [51] was created. There were 17

questions which included 5 demographic, 7 yes or no questions and 3 likert-type response questions while 3 questions were comment fields, the full document is available in appendix A.

All participants were given a handout with three mock-up screens with the aimed functionality presented. They were asked to read and answer each page without looking the next one, after each page was filled a discussion started, all the discussion was recorded and then transcribed and anonymised.

The participants were introduced to a simplified version of the concept, where a special router exists, that captures all the network traffic and allows them to see security related information about any device connected to it. The router is also capable to do basic penetration testing, vulnerability scanning and display the output to a web interface hosted on the router. Then they were asked to express their opinion about the information displayed on each of the mock-up screens. The results of the comments, discussion and survey were used to evaluate and enhance the design.

3.2.2 Focus Group objectives

The main objective was to determine suitability of the security scoring mechanism as well as to get informed feedback from experts in order to verify the minimum functionality expected from the framework, in order to provide the functionality needed to create a graphical interface that conveys information relevant to each user group.

3.2.3 The Participants

This survey was used in a focus group of Information Technology students which can be described as a group of experts. The expert group consisted of nine students with ages ranging from 21 to 28, and one lecturer, five males and five females. Every person on average was using at least two different operating systems on average, which aids the diverse insight needed by people used to different interfaces when designing interfaces. The following table presents how familiar participants believe they are with the technologies used by this system sorted by their familiarity.

Level of familiarity				
Participants	Networking	PCS	Vulnerabilities	TLS
P10	Moderately	Extremely	Moderately	Somewhat
P1	Moderately	Moderately	Moderately	Somewhat
P8	Somewhat	Moderately	Somewhat	Extremely
P3	Moderately	Moderately	Somewhat	Somewhat
P5	Moderately	Somewhat	Somewhat	Somewhat
P6	Somewhat	Somewhat	Somewhat	Somewhat
P4	Somewhat	Somewhat	Somewhat	Slightly
P7	Somewhat	Not at all	Somewhat	Somewhat
P2	Slightly	Not at all	Not at all	Not at all
P9	Not at all	Not at all	Not at all	Not at all

Table 1. Focus Group participants sorted by familiarity.

3.2.4 Results

The following part will present the input provided by the expert group based on the mock-up design they evaluated. Detailed results of the questions can be found in appendix D.

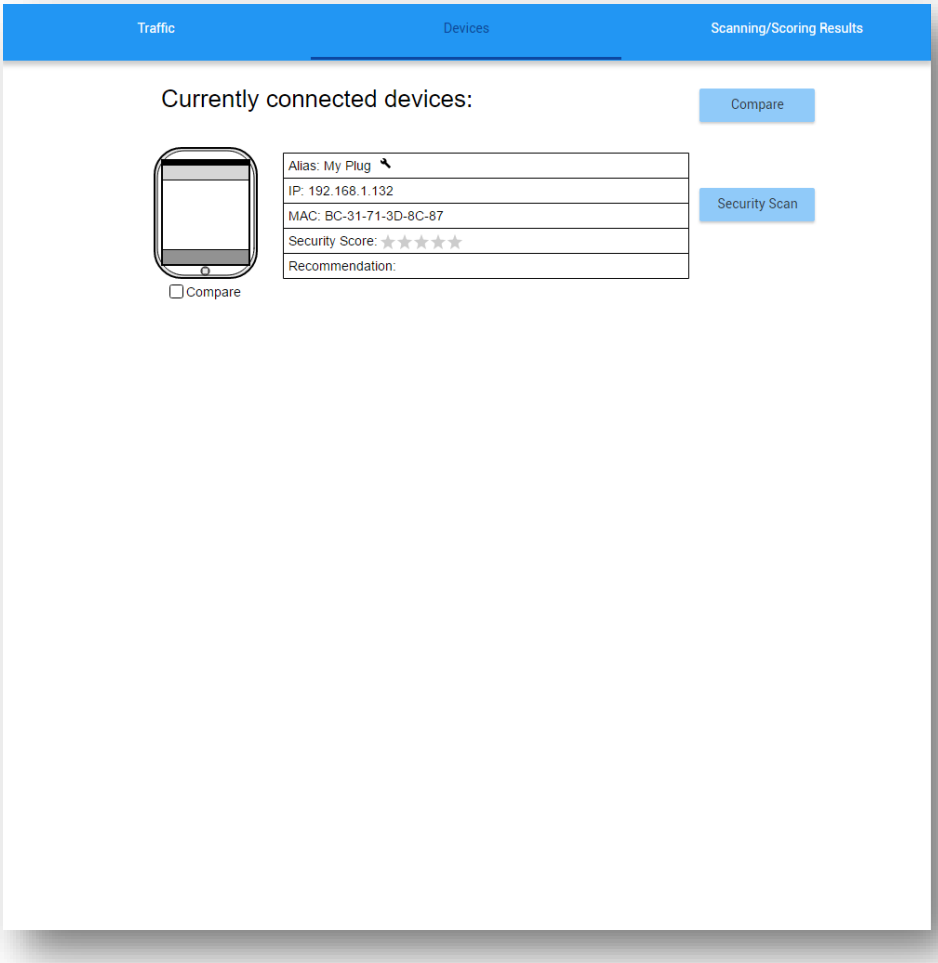


Figure 2. First page of the mock-up design.

The participants were asked two questions, if the information presented is adequate for the use of this device and what changes they would make to the design.

Feedback:

While eight out of ten participants (8/10) agreed that the information was adequate both during discussion and on their written comments some concerns for the design were highlighted. Most agreed that two buttons linking to the traffic and scoring results pages is needed to be where the security button stands and that the compare functionality should be hidden if just one device is connected to the network.

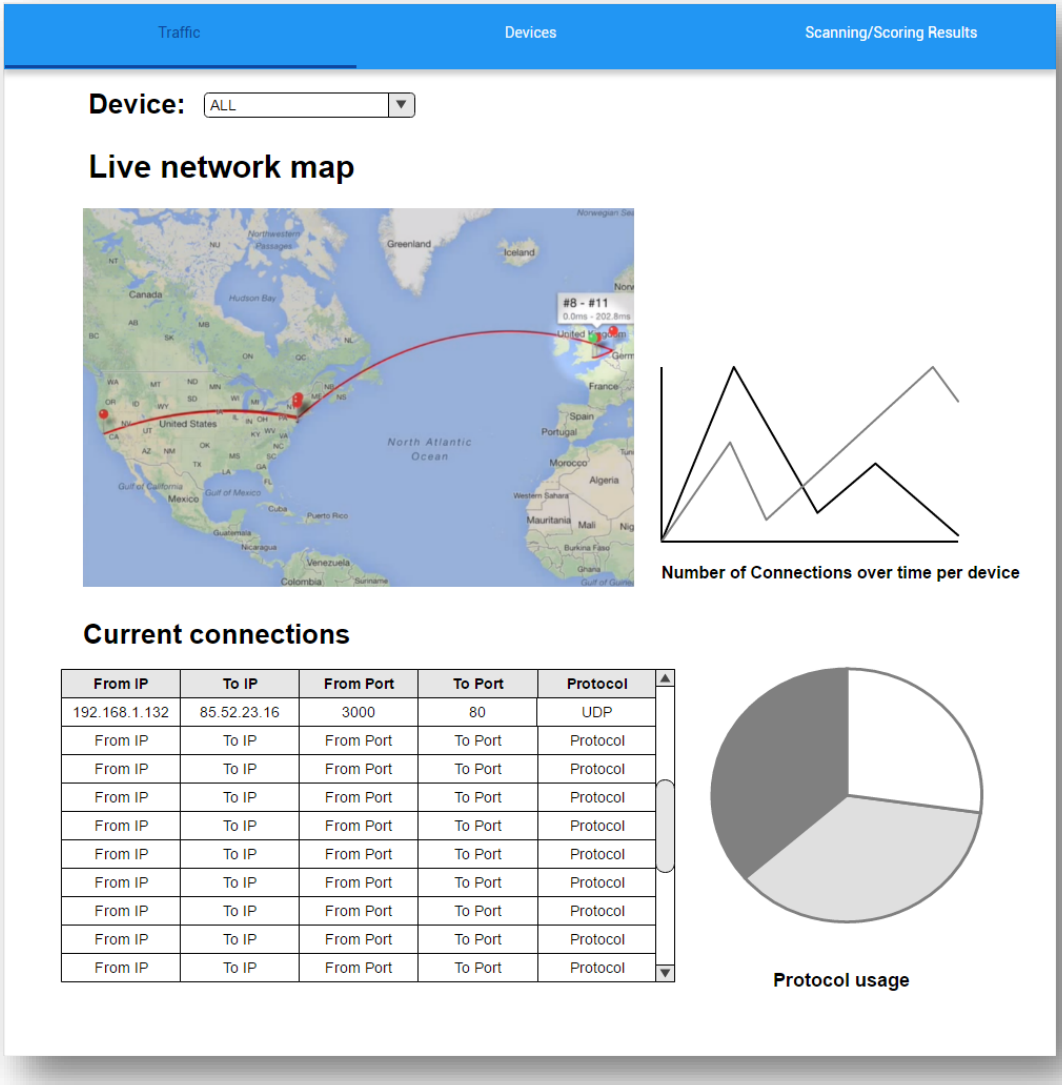


Figure 3. Second page of the mock-up design.

The participants were asked to identify what affects their opinion on the device security from the information displayed on this page and during the discussion what is important for them and missing and their overall opinion on the design of the page.

Feedback:

With the exemption of protocol usage percentage, the participants mostly agreed that the information presented on this page are relevant and affected their opinion. During the discussion the most prominent feature requested was the inclusion of bandwidth usage statistics per device.

Map: Influence				
Participants	Live Map	Connections	Protocols	Line Chart
P1	Agree	Strongly Agree	Agree	Agree
P2	Strongly Agree	Agree	Neither	Neither
P3	Agree	Agree	Strongly Agree	Strongly Agree
P4	Agree	Agree	Neither	Agree
P5	N/A	N/A	N/A	N/A
P6	N/A	N/A	N/A	N/A
P7	Agree	Agree	Agree	Agree
P8	Agree	Agree	Disagree	Neither
P9	Strongly Agree	Agree	Strongly disagree	Agree
P10	Strongly Agree	Strongly Agree	Strongly disagree	Disagree

Table 2. Map influence table.

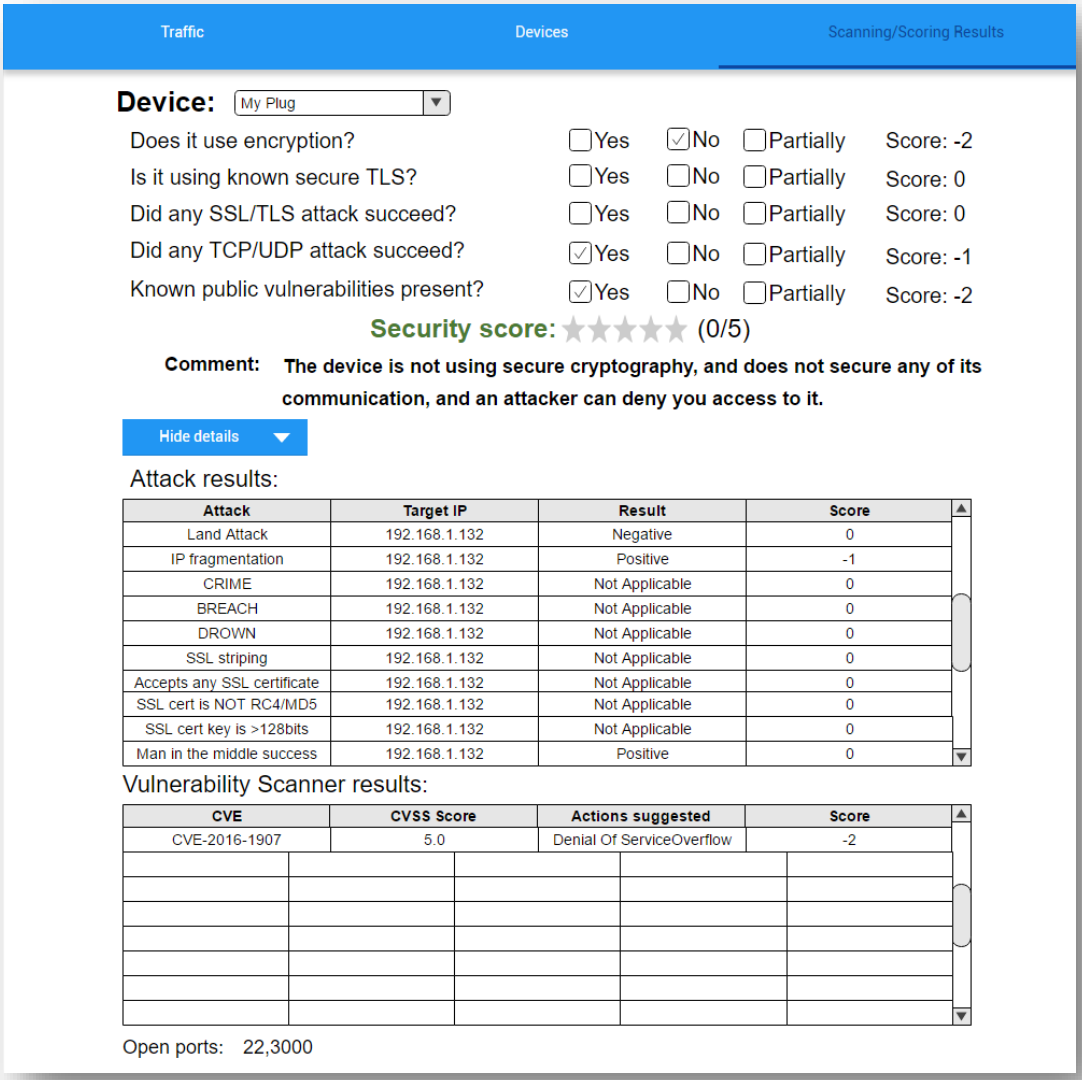


Figure 4. Third page of the mock-up design.

The group was asked about the importance of the criteria presented as metrics for the security of the device, if they would consider a device that passed these tests secure and if they believe that the framework would potentially harm their device.

Feedback:

Seven participants replied that they would consider the device secure if it passed all the steps, one wouldn't while two people didn't answer, voicing concerns during the discussion that the feedback should be on a higher level to be easily understandable, and that many people will not understand the output in the way it is presented. Some

of the participants were not aware of some of the attacks and requested a sort description to be provided or a link to a description of the attack. Criticality and ease of exploitation were raised as security metrics that the group would like to be implemented. Three experts believed that the framework can cause potential harm to their devices.

Security score system metrics: Level of Importance						
Participants	Encryption	Safe TLS	Land Attack	IP fragmentation	CRIME	BREACH
P1	Strongly Agree	Strongly Agree	Neither	Agree	Strongly Agree	Strongly Agree
P2	Strongly Agree	Don't know	Don't know	Don't know	Don't know	Don't know
P3	Strongly Agree	Strongly Agree	Agree	Don't know	Agree	Agree
P4	Agree	Strongly Agree	Don't know	Don't know	Don't know	Don't know
P5	Strongly Agree	Strongly Agree	Don't know	Don't know	Don't know	Don't know
P6	Agree	Agree	Don't know	Agree	Don't know	Don't know
P7	Agree	Agree	Agree	Agree	Neither	Neither
P8	Strongly Agree	Agree	Don't know	Don't know	Don't know	Don't know
P9	Don't know	Don't know	Don't know	Don't know	Don't know	Don't know
P10	Strongly Agree	Strongly Agree	Don't know	Agree	Don't know	Don't know

Table 3. Security metrics importance, part 1.

Security score system metrics: Level of Importance						
Participants	DROWN	SSL stripping	Any cert	Not RC4/MD5	>128bits	Scanner
P1	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Agree
P2	Don't know	Don't know	Don't know	Don't know	Don't know	Strongly Agree
P3	Agree	Agree	Agree	Don't know	Strongly Agree	Agree
P4	Don't know	Agree	Agree	Don't know	Don't know	Strongly Agree
P5	Don't know	Don't know	Strongly Agree	Agree	Agree	Strongly Agree
P6	Don't know	N/A	Neither	Neither	Agree	Agree
P7	Neither	Agree	Agree	Agree	Agree	Agree
P8	Don't know	Neither	Strongly Agree	Agree	Strongly Agree	Strongly Agree
P9	Don't know	Don't know	Don't know	Don't know	Don't know	Don't know
P10	Don't know	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Agree	Strongly Agree

Table 4. Security metrics importance, part 2.

Chapter 4
Methodology & Design

A new design is introduced that enables easier and widespread security and privacy evaluation by both computer scientists and not alike. The input from the design requirements stage was used and the same functionality goals were set. A new methodology was needed to determine the basic factors that define a device as secure and private and a way to convey the information to the user.

A privacy score is considered in order to give the end user a clear answer on whether her data can be accessed in any way by the potential attackers, based on the use of encryption by the device and whether it is implemented correctly thus avoiding all the encryption targeting attacks.

A security score is introduced as well in order to provide a higher level representation of security as it was requested by the majority of the experts which attended the focus group.

The following chapter sums the theoretical background upon which the experimental implementation is based.

4.1 Defining Security and Privacy

Defining privacy and security is vague because people usually have different definition based on the context. In this case since the proposed system operates in the network level, hence privacy is directly linked to the confidentiality as defined by CVSS.[52]

Integrity and availability of the data are also two categories which will impact the score, integrity impacts both security and privacy while availability can impact only the security score. Some of the categories will have set CVSS score depending on the type of attack/design validity, while the rest is given a score dynamically depending on the results. CVSS can be used to explain what is considered secure:

4.1.1 Confidentiality

The confidential data should be encrypted. Thus the information is not accessible to the potential attacker. Defining which information should be confidential and which should not, is impossible, and can be achieved only in a per case scenario and not in the en masse scenario this paper explores. All information will be considered confidential, so any device that does not encrypt its traffic or an attacker is capable of removing the encryption, will be treated with a high confidentiality impact score.

4.1.2 Integrity

Integrity assures that the data received are exactly as sent by an authorized entity. A MITM attack, where the traffic is decrypted or unencrypted, can lead in most cases in loss of integrity however it does not necessarily mean that all cases will lead to such an event. If authentication is implemented properly the attacker should not be able to tamper the data, this is another a case where a compromise must be done in the design. It is possible to replay captured streams and compare the responses to that of the original legitimate requests. If a device replies means that there is authentication system or the implemented system is broken, hence the integrity will be considered compromised. Such a check though is out of the scope of the framework at this stage and should be done manually.

4.1.3 Availability

The availability of a device that is connected in a home LAN usually does not concern owners since it is rarely targeted by an attack. This is the case because the devices expose only specific services to the internet, while the router firewall and NAT protect them from external attacks. In a local attack scenario an attacker may be able to cause a denial of service attack from within the network. Attacks that exhaust the resources of the target device by brute-force are not considered.

4.1.1 Attack Vector

The attack vector for all the attacks will be set to Network, since this project is only able to exploit network level attacks en masse.

4.1.2 Attack Complexity

Attack complexity is analogous to the attack and will be set based on the score that the vulnerabilities that are used have in reported CVEs.

4.1.3 Privileges Required

All the attacks need no special privileges since they all are network level attacks, that target generic implementation mistakes and do not target some specific software or product.

4.1.4 User Interaction

As with the previous category, only network level attacks are considered and user interaction is not required for any of them.

4.1.5 Scope

The change of scope is difficult to define just by the results of scanning and hence is not considered. In the vast majority of cases, the scope will remain unchanged since most of the devices are consumer electronics, but in the case of remote sensors and other distributed implementations the change of scope would be useful. Future work could involve this security check as well.

4.2 Attacks and Vulnerability scanning

During the first stage the device needs to execute a preliminary scan both by a port and a vulnerability scanner. The second stage consists of checking the typical characteristics of the encryption mechanism, if any is in place. The security of the mechanism is checked for well-known TLS downgrade attacks from the literature.

4.3 Privacy and security score system

While the implemented attacks are deterministic of the exploitability of the target and the CVSS score and the attack results can be understood by experts, a precise and simple set of rules is needed to define the score on a higher level. As noted previously, the base metric group of Common Vulnerability Scoring System (CVSS) version 3.0 is used as a base methodology to calculate the score, since a traditional vulnerability scanner will be employed as well this will help in the easier aggregation of the final results.

Grouping the scores and producing a score based on categories that end users can understand is crucial. The security score should be numerical and include the number of vulnerabilities and a text representation of the criticality of the attack. That would allow the score to be transformed into a 5-star rating which is universally understood and would also allow for a traffic light style implementation were the security of the device can be represented with amber, yellow, green colours, thus bypassing any boundaries imposed by the lack of security expertise by the users.

Hence a grouping will occur to represent a five-star final score on two categories network security and privacy.

4.4 Privacy and security API

A router implementing the basic functionality is the first part that is needed, and the minimum functionality should include the capability to save data flows and do basic security checks and attacks standalone. Since the main target of the project is mostly the networking, information design and security community, instead of building directly the front end based on the services available on the router, an API will be introduced that automates the functionality present in a way that the implementation can prove useful in the future in further research or security and data flow visualization projects.

All the technologies to be used, need to be free open source software (FOSS) hence the abstraction and creation of the layered design will prove useful to future expanding and modification of the system. The finalised API should provide the aims set at the design requirements phase, namely it should be able to provide:

- The overview of current and past connected devices.
- The overview of current and past network connections.
- The simulation of the past network sessions.
- The vulnerability and encryption scanning of devices.
- The production of a simplified security score.

A Service Oriented Architecture (SOA) approach was deemed to be the best fitting approach for this project. SOA is an architectural approach in which systems are built as decentralised autonomous services. The integration is part of the design before any functional consideration is made while the produced framework is composed of services running on different devices and platforms[53]. While this architecture sounds overcomplicated it allows for the autonomy and modularity needed by the target system. Through the use of abstraction layers, a future researcher or user of the system will be able to only use the part of the functionality she needs based on the needs. Someone can use the scripts and commands integrated in to the router without the need of the API, or may use the API without the need for an OpenVAS instance if they do not need the vulnerability scanning capabilities. This architecture approach also allows for the implementation of interchangeable modules, such as different web interfaces consuming the same API concurrently[53]. The final architecture and details of the design were researched during the implementation phase hence more information can be found in the next chapter.

Chapter 5
Experimental Implementation

The first step towards any practical implementation is to find the software which would be necessary, based on the conceptual design. A thorough research was conducted in order to identify the open source software and libraries needed to achieve the goal. The research lead to various tools that could potentially be ported to the framework, although only a small subset was possible to be implemented given the projects time limit. Some of the state of the art tools that are powering the implementation are analysed in order to produce an overview of the current state, what they provide to the project and the future possibilities for expansion.

5.1 Equipment

5.1.1 Packet capture router

A Linksys WRT1900ACS was used to implement the base of the system. It was chosen because it supports the open source OpenWRT Linux-based operating system out of the box[54]. A custom OpenWRT image was created that incorporates Python and all the dependencies needed to implement the scripts in the next steps. Traditional packet capture software was included as well as a MySQL database to store all the session data and results. A fork of OpenWRT was created since the changes were beyond customisation and included modifying and adding make files and software patches.

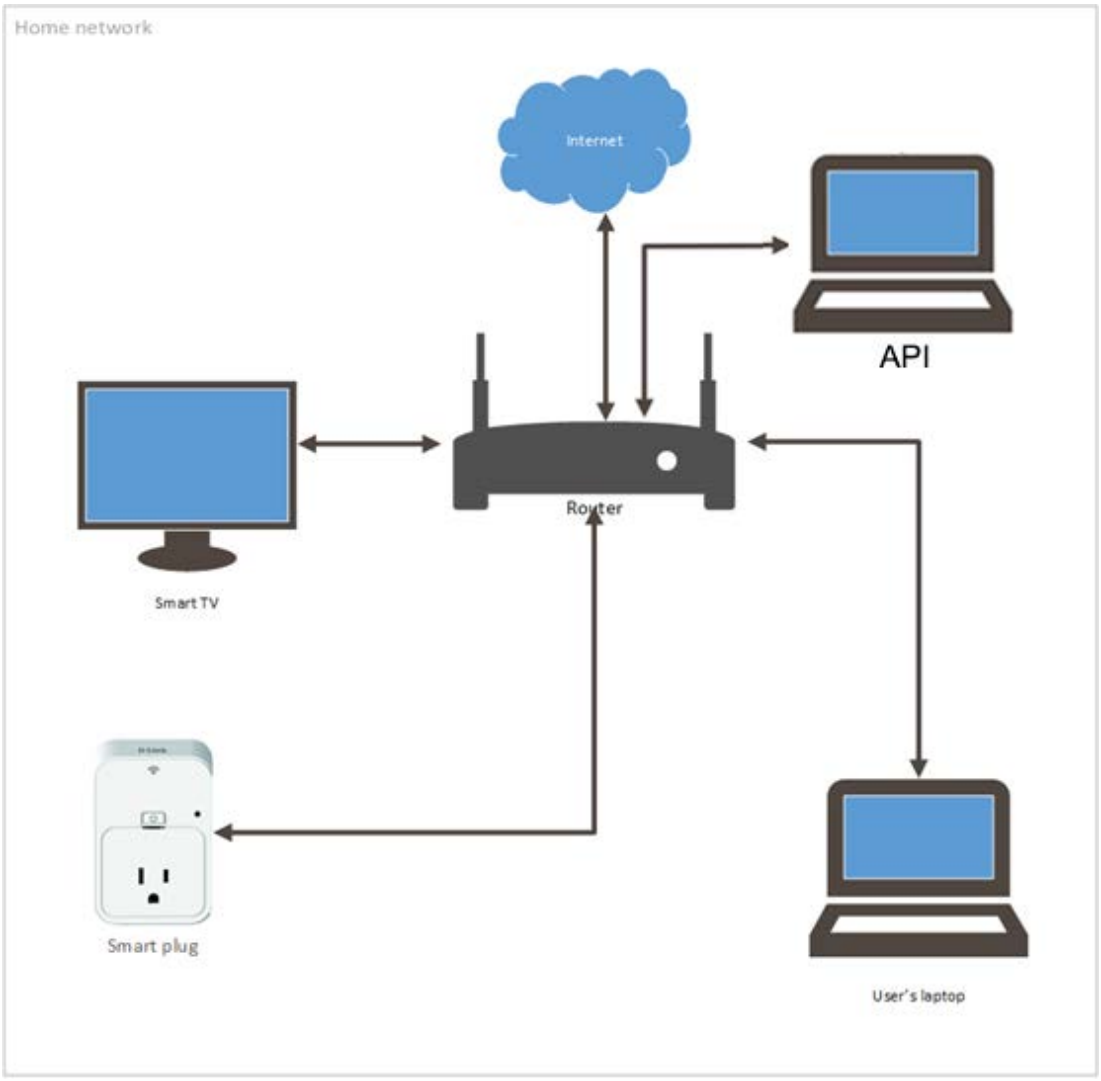


Figure 5. Platform overview.

5.1.2 API and front-end

Originally the design was aiming the framework to be available for the Raspberry PI minicomputer, a custom Linux operating system was created with the minimal number of packages needed in order to maximise performance. The memory usage of the API and the need for outsourcing of OpenVAS reporting because of the lack of power, led to the decision to host all the software including a Kali Linux virtual machine with a preconfigured OpenVAS and SSH connection on laptop attached to one of the routers Ethernet interfaces. Any modern computer can be used to host the API since there are only two dependencies, Java and virtualbox.

5.2 Software

This section highlights the software that enables the creation of the framework, although non-exhaustive, the list includes the core software used for the implementation of both the API and the router.

5.2.1 OpenVAS

OpenVAS is a fork of the last free version of the Nessus vulnerability scanner and it is considered one of the industry standards[55], [56]. It has proved during benchmarks that it can perform in par with closed source solutions[57]. It also offers an open-source communications protocol, making ideal candidate for the project. Implementing the automation proved troublesome. While OpenVAS offers a state of the art protocol which supports an XML like interface, all the available open source libraries capable of automating communication with the system for both Java and Python were outdated and incompatible with the version 6.0 of the protocol.[58] The XML report format was used to create a custom object factory and responses thus practically creating a new Java library that covers part of its functionality.

5.2.2 Libpcap

Libpcap is a platform-independent library for low-level network monitoring and packet capture[59], it powers industry standard tools like Wireshark[60] and tcpdump[61]. It is in the core of API, since all other packet manipulating scripts were using various wrappers. Libpcap is available in OpenWRT and the integration was flawless. By using libpcap and its wrappers it is possible to create code that is small agile, system-independent hence portable.

5.2.3 Nmap

Nmap is a well-known open-source network scanner that exists since 1997[62] and is common to the security community as it offers both a Python library [63]as well as XML output[64] which makes the automation of its functions trivial. It can be used for

port scanning and service detection as well as for host detection[65] in case the DHCP server of the router is not running.

5.2.4 Scapy

The library is a libpcap python wrapper which allows for the creation, sniffing, dumping and manipulating network packets. It supports protocols in the form of layers which allows easy packet manipulation as well as the introduction of new protocols[66]. A community driven project named Scapy-SSL/TLS brought the TLS layer to the Scapy library[67], thus enabling packet manipulation of that protocol as well. The library sits at the core of the implementation being responsible through scripts for data dumping to the database and through the TLS protocol layers for analysis of the TLS characteristics as well. The possibilities of the library are endless; any network level attack can be executed or simulated, making it a very powerful library to manipulate traffic at a low level.

5.2.5 OpenSSL

The OpenSSL library is part of the core functionality in the project as well. It is used to get the TLS public key and certificate as well as to analyse certificate fields such as the common authority name and the length and type of the key. The library is used also by the SSLstrip attack explained later. Other than the usages that already has, OpenSSl allows further expansion of the TLS/SSL analysis capabilities and is considered an integral part of the project.

5.2.6 GeoIP

The built-in geoip functionality is provided by the Max Mind Java library[68] using a local database file, that maximizes efficiency and portability. Automating completely the service would considerably impact the router under heavy loads, thus an API call was implemented that allows for the retrieval of geolocation information from a given IP address.

5.2.7 SSLstrip2 and dns2proxy

A developer named Leonardo Nve presented at black hat Asia 2014 an attack against HSTS which enables to SSLstrip to function again, a new modified version was needed and the introduction of a new software called dns2proxy was introduced[69].

A combination of SSLstrip2[70] and a dns2proxy[71] can bypass HSTS using a simple technique. When the user enters a website the attacker highjacks the HTTP session and redirects it to a non-existing subdomain of the target site. There's no record for this domain cached so a new DNS query is conducted by the user's machine. The attacker highjacks the DNS query and returns the IP address of the original site, this way even if HSTS and compatible browsers are used there is no rule set for the non-existing subdomain, hence the protection is not enforced.

As with all kinds of this attack using the secure link directly nullifies the attack. The problem is using links that don't start with "https://" this is more apparent when using bing.com since many links in the results are in the form of www.example.com instead of https://www.example.com that allows the script to work transparently when the user clicks a result.

5.3 Router implementation

The OpenWRT Chaos Calmer version 15.05.1 Linux-based operating system was forked and used as the base firmware of the router. The targets were four:

- Create an operating system which includes all the tools needed from the previous section which were not previously ported to the operating system, thus needed to be cross-compiled to the architecture since OpenWRT lacks native compilation.
- Create the needed environment for the development of the scripts that automate the functionality.
- Create the scripts that automate the router functions
- Create a deploy procedure which would allow the easy setup of multiple systems.

5.3.1 Operating system and needed tools

The development team of OpenWRT has streamlined the development process allowing easy modifications to the software. The base software is the minimal possible that would not deprive the router of its normal routing functions.

To allow for more space both for software and the dumped data, an external hard drive is used to which the router file system is expanded thus proving ample space.

Libpcap was already available as well as the nmap suite, and full Python support. MySQL was slightly modified to allow more recent database engines to be used and a specific version of OpenSSL needed by SSL strip was ported to the device. The vast majority of the Python libraries needed to be ported to allow their C components to be compiled and thus run natively on the device.

5.3.2 Development environment

The external hard drive as noted previously holds an expanded file system, but also a data partition. All the data saved to the database as well as all the scripts reside on this partition. The data partition is available through a network share to the connected devices and allows for easy data transfers and script development. The database is accessible over the network and locally. The Python libraries present on the router and needed for its function are numerous, an exhaustive list with the libraries available and their versions as reported by Python pip:

- dnspython==1.14.0
- futures==3.0.5
- IPy==0.83
- MySQL-python==1.2.5
- ndg-httpsclient==0.4.2
- pcapy==0.10.9
- pyasn1==0.1.9
- pycrypto==2.6.1
- pyOpenSSL==0.13.1

- requests==2.6.0
- scapy==2.3.2
- service-identity==16.0.0
- tinyec==0.3.1
- Twisted==13.1.0
- zope.interface==4.1.3

This set of libraries allow the implementation of virtually any network packet operation and power all of the routers functions and attacks.

5.3.3 Development of Python scripts

In total five python scripts were created and an open source one was modified, these scripts can be used as standalone command line tools as well. An overview of their functionality follows.

The datadump script is based on scapy and is responsible for the packet sniffing and dumping to the database as well as to pcap files, although it uses tcpdump for its pcap functionality due to the speed improvement.

The livehosts scripts parses the output of the ARP and DHCP tables to update the database about the currently connected devices.

The nmapInterface script is alternative implementation of the previous functionality which although slower it allows the router to have the same level of information when its DHCP server is not running.

The nmapdb script is based upon the synonymous open source script[72] which allows the parsing of nmap XML reports and export the results to SQLite, the script was modified to save to the local MySQL database instead.

The security_scanner script uses both OpenSSL and Scapy to acquire information about the SSL server hosted at a target device port and save them to the database. It is based partly upon example code from the Scapy SSL_TLS project [73]. The information it generates include the public key, its length, supported ciphers and the

certificate itself as well as if the server is vulnerable to BREACH, DROWN, FREAK or LOGJAM attacks.

The script_controller is a simple script that is used to shut down the rest of the scripts, and uses unix sockets to pass a kill message to the scripts, which when received by the scripts, they shutdown safely.

5.3.4 Implement network level attacks

The Network level attacks were implemented in Python using Scapy. Public exploit code was examined and attacks were implemented based on these implementations. In cases like CRIME, in which the attack is rather complex to implement scripts were created that test the system in the way a traditional vulnerability scanner does by checking if the vulnerable components exist, while LAND attack and IP fragmentation were implemented on a proof of concept level they were not included in the final design, as no usable information could be obtained from the attacks. SSL stripping was the only active attack implemented into an API call, the reason being that its ability to strip the traffic of encryption on the fly can be proved useful in a framework where network data is the target. Nevertheless, the router proved its capability as a development environment and attack deployment mechanism.

5.3.5 Deployment

The deployment of the custom image and settings is done in three stages. During the first stage the user has to flash a custom firmware to the router, like they would for any other firmware. The second step is connecting the pre- partitioned hard drive to the router and pressing the “WPS” button that the device has on its back. The button is assigned a soft reset function which expands the main file system of the device to the external hard drive and reboots the device. Last step is to connect to the router through SSH and give the command initsql, which is a custom command that setups the database. The deployment needs less than five minutes. The router setup has been documented and.

5.4 Implementation of the API and device

The API was created on top of free open source software to guarantee its multiplatform and expansion capabilities. During the research from which the final architecture decisions were made, different types of APIs and setups were considered. Through evaluation of the project's targets which is to offer the most platform independent and easy to use solution in order to guarantee larger adaptation and expandability, it was decided only two main areas to be researched.

The two options were a web service or web socket based API. At the initial phases a web socket based API was considered due to the fact that this approach is usually faster and less resource hungry[74], [75], hence would be a logical choice for the retrieval of large amounts of data. Web sockets though are harder to adapt to and require more specialised knowledge in order to interact with, hence the slower but easier to use web-service design was chosen. The final design uses a web-service in representational state transfer (REST) architectural style. In order for an application to be considered RESTful the following architectural contains should be followed[76], [77]:

- **Client-Server**

Separation of concerns is the principle behind the client-server constraints. The separation of user interface and data concerns improves the scalability and portability and cross platform compatibility while allowing each component to evolve independently, since it decouples the consumers from the producers of data.

- **Stateless**

Communication must be stateless in nature, requests from clients to servers must contain all of the information necessary to execute the request, and should not use any data available on the server to keep the state of interaction, state if needed should be kept client-side, only. This constraint has as a result the following advantages, it induces visibility, reliability, and scalability, while the main disadvantage is the decrease in performance by the increase of repetitive data in a per-interaction overhead fashion.

- **Cache**

Cache constraints mean that a response to a request must be implicitly or explicitly noted as cacheable or non-cacheable. Since the results vary in our implementation and most of them are real time or inside specific time constrains, all requests are considered non-cacheable to guarantee that no outdated data are delivered to the clients, this approach has the disadvantage of poorer performance.

- **Uniform interface**

By the term uniform interface, it is meant the use of similar interactions in a standardised format rather on per case scenario. This creates an abstraction layer that decouples the implementation from the service, increasing visibility since the users does not use different interfaces for different functions. The main drawback is the increase of overhead.

- **Layered System**

A layered system is composed by hierarchical layers which constrains the components to only be able to interact with their immediate layer hiding the rest of the layers. The layered design allows easier expansion and promotes module independence. Every layer is only aware of the previous and next layer and each layer exposes a set of functions that can be consumed by clients. The layered style is also useful when large projects are concerned since it allows the distribution of layers and the use of intermediate load balancers hence improving performance and manageability.

The Spring boot framework was used as it allows the creation of web-service RESTful APIs, while it provides self-contained projects with no external dependencies other than Java. Modular by design, the database and existing code can be expanded considerably, while features can be added and removed, without harming other features. The backend consists of an SSH library that is used to send commands to custom python scripts residing on the router, the scripts output directly to the database hosted on the router and the API reads the results directly from the database, upon request.

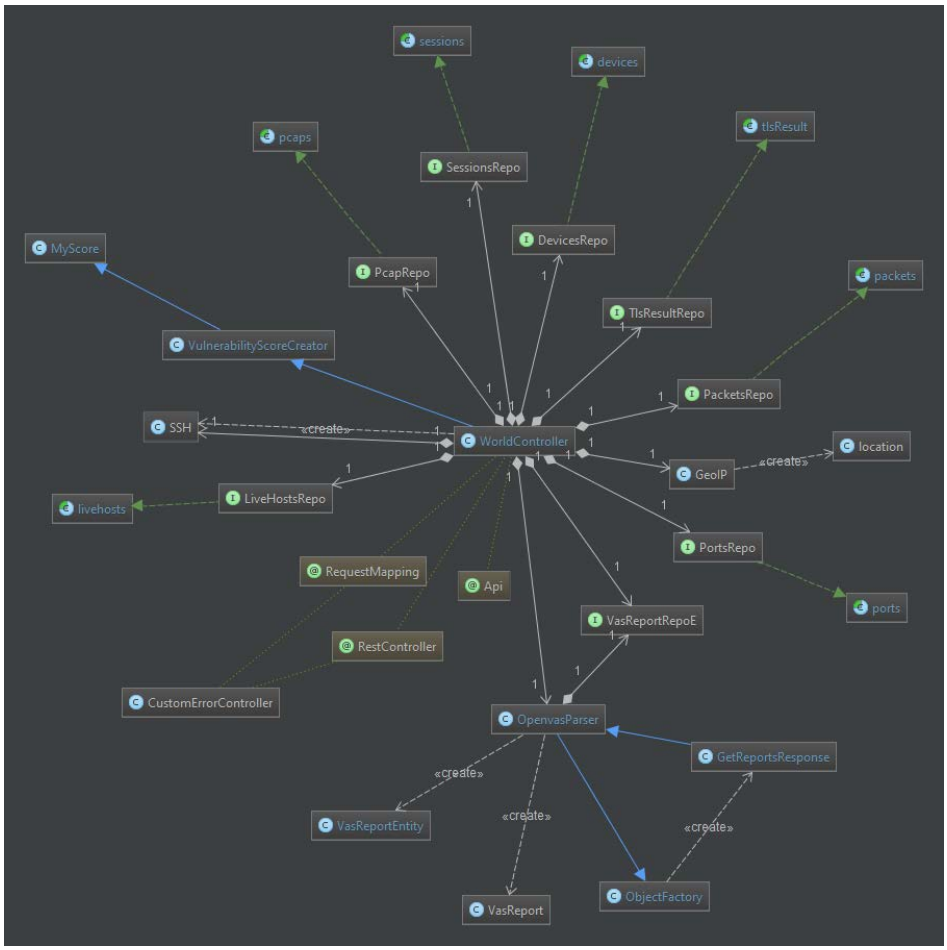


Figure 6. A class diagram with each module being a repository of entities mapped to the database.

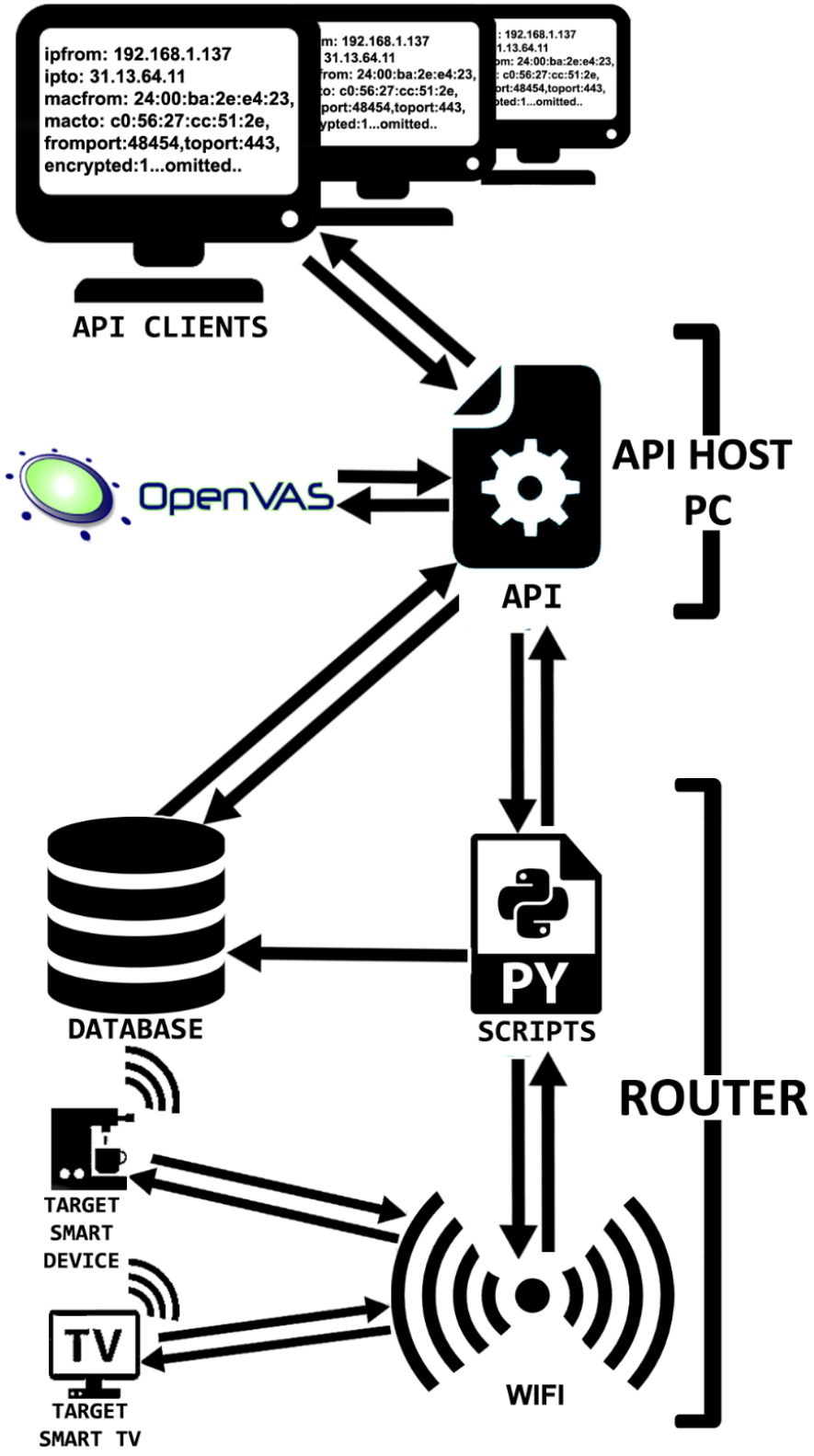


Figure 7. System diagram and overview of the platform.

5.5 Implementation of the front end

While a front-end was considered in the beginning of this project, by the end it was dropped in favour of more API functionality which would allow more feature rich front-ends to be built. An example implementation was created by a university intern, purely in python which showcases the possibilities of the router itself, since it does not use the API that was developed to automate the framework.

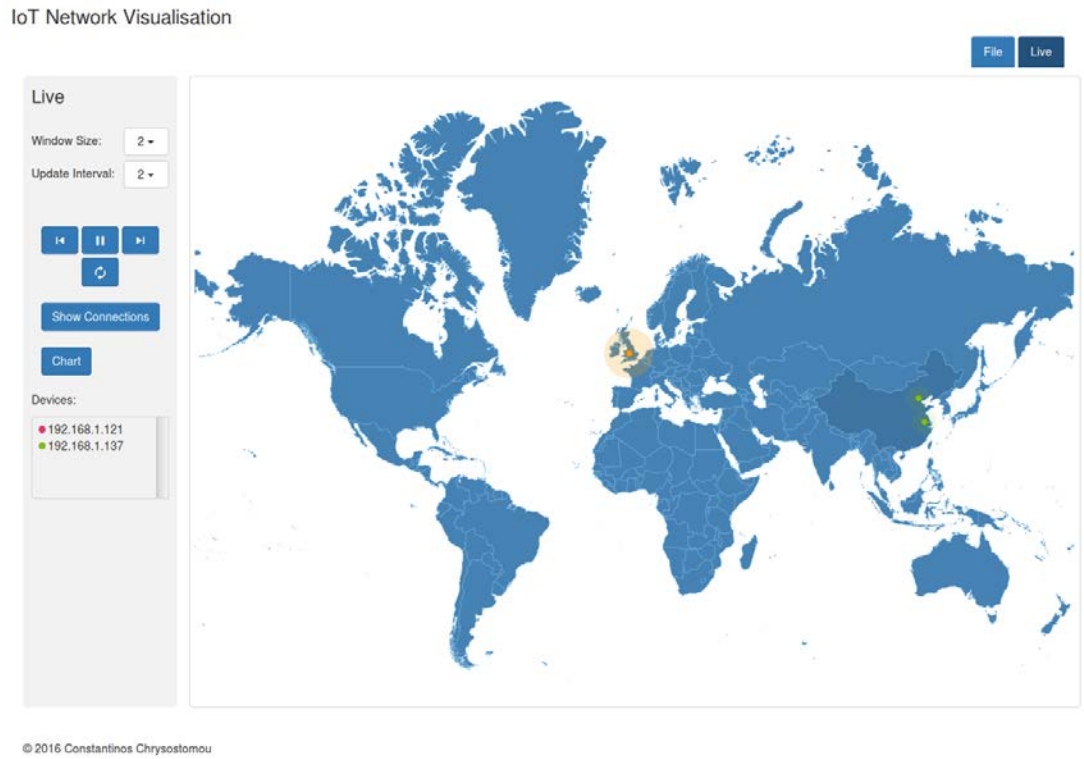


Figure 8. The web interface mapping traffic during a test visit to baidu.com.

The front-end showcased is capable of displaying live sessions, connected devices, currently active connections and plot the connections on a world map, as well as replaying previously saved sessions. This highlights the convenience of the layered design, as the router can be used autonomously or in conjunction with the API.

5.6 Outputs

The main goal of this research was to introduce a standard methodology and toolset for IoT device network behavioural analysis. Since it is a framework the outputs are more than just the core API functionality. The main outputs are:

- A definition of privacy and security for large scale evaluation.
- A man-in-the-middle rogue AP built upon Linksys WRT1900ACS router with the added capabilities of capturing all network traffic while attacking target machines.
- The custom OpenWRT operating system which includes Python libraries and that were not originally available to the device and which allow the implementation of various network attacks.
- An API to control the device, automate the attacks and evaluate the captured data.

The framework introduces 68 new API calls in multiple functionality domains. All the calls are executed using HTTP GET, making the integration of the API trivial.

The attached storage of the router can store the traffic in both the integrated database and in the popular PCAP format, with the results being available over the network. It is possible to set an alias per device and use this instead of IDs or other identification which simplifies the interaction. The database makes the recording and replaying of sessions and basic analysis such as behaviour analysis, encryption and basic security analysis possible in a scalable and automated fashion. On average the framework can dump to and fetch from the database an average of 2.658 packets, per second when only the IP layer header is dumped according to the benchmarks conducted.

A small subset of some of the implemented API calls and their outputs are shown as an overview of the two main categories of the functionality and interaction with the API in the following part, an overview of all the commands available can be reviewed in appendix B.

Packets

Packet capturing is half the functionality of the framework, to get a packet commands in the following form are used:

/api/packets/getall/firstpacket

This call will return the first packet ever recorded in any session by the router. Most of the packet related commands will return multiple objects like the following filtered by time ranges and/or devices.

Result:


```
[{"id":"1a10d070-5e8a-11e6-8995-
c05627cc512e","ipfrom":"192.168.1.137","ipto":"31.13.64.11","macfrom
":"24:00:ba:2e:e4:23","macto":"c0:56:27:cc:51:2e","fromport":48454,"
toport":443,"length":83,"protocol":6,"time":"2016-08-10
00:36:31.720854","bytes":"wFYnzFEuJAC6LuQjCABFAABT1Ac=","encrypted":
1,"unix_time":1470785792,"session_id":"041545d9-5e8a-11e6-9362-
c05627cc512e"}]
```


The id is a unique id for this packet, the IP, MAC and ports from and to, the protocol number an IANA defined protocol number[78] the timestamp in text and Unix format as well as the binary payload of the packet encoded in base64 is provided. The session id allows to group the packets to sessions which allows easy simulation later.

Security

There are various security capabilities available, for this demo the sequence of commands needed to generate a TLS and vulnerability scanner report is considered, since these are the perquisites in order to generate the final simplified score. The result of the vulnerability and TLS scanner is omitted because of their large size.

The user can easily execute ports scans, custom TLS scans as well as full OpenVAS reports, by following this procedure. The alias used in the examples is the custom name that the user can give to its device. The first step is to execute the scans; this can be achieved with the following calls:

/api/security/scan/vulnerability/byalias/{alias}

This call initiates an OpenVAS vulnerability scan, with no further interaction needed and saves the report results to the database when the scan is finished. Then to execute a TLS scan the following command is used:

/api/security/scan/tls/alias/{alias}/{port}

This call initiates the TLS scan, as with the vulnerability scan, a multithreaded script will execute the scan and save the results to the database when it finishes. The port parameter is the port that hosts the TLS server and the script will target. In this example

the device is named lametric and the TLS server is hosted at port 443, the final commands would be:

/api/security/scan/vulnerability/byalias/lametric

/api/security/scan/tls/alias/lametric/443

The scans take several minutes depending the device and when completed the results are saved into the database as noted previously. The next step aggregates the results from the TLS scans and the OpenVAS report. The attacks checked by the TLS scanner are set a CVSS score based on existing CVEs that exploit this kind of attack. More specifically CRIME attack has a score of 2.6 [79], while the DROWN attack has a score of 4.3 [80].

The score of the highest CVSS reported for the selected device is set at its final score along with the number of scanned objects that had a CVSS score, thus giving an easy to understand overview and a score that can be transformed to five-star rating:

/api/security/score/byalias/lametric

Result:


```
[{"id":"be7aea67-4fbc-4441-bbee-
e54a734b486e","mac":"58:63:56:2d:b5:28","score":5.5,"criticality":"M
edium","numberofvuln":4,"encrypted":null,"unix_time":1470804480}]
```


The criticality is calculated with the same values CVSS calculates its severity rating[81] and gives an easy overview to users with no network or security experience, thus enabling the implementation of easy to use front ends as well as the deep security analysis of devices.

Chapter 6
Expert evaluation

A number of experts were contacted in order to evaluate the design of the framework. Expert 1 was working on the project since the first stages, hence never used the API or was provided with the documentation that the rest of the experts were. His opinion is merely about the functionality of the router itself and not for the framework as a whole. The rest of the experts were informed about the design and were provided with a demo showcasing three potential usage scenarios and projects, and their solutions, the full document can be reviewed in appendix C. The main reason for the lack of a live demo was due to university rules, which require special permissions to attach the framework on the university network.

The experts were also provided with the full API documentation which is not included in this document due its large size since the documentation of the API is over 70 pages long. An overview of the available commands can be reviewed in appendix B.

Meetings were setup with experts 1,2 and 4 while all the experts were required to send their opinion in an email as well. The experts were invited to ask questions and comment on the design and functionality of the framework. More specific questions were asked about the potential use of the framework the familiarity of interaction and if it is appropriate for its aims. Each expert’s opinion is summarised and presented separately.

6.1 Expert 1

An informatics student intern was tasked to generate a web interface using the router as his workbench. The final implementation does not use the API but rather the scripts, tools and the database available. The web interface is a pure Python implementation that displays live or recorded network sessions on a map (figure 9). His project's aim was to investigate data transferred over Internet of Things objects, such as smart TVs or smart kettles. In total the first expert worked over a month with the device.

He used the router in his project which was visualising network traffic across a network of IoT objects. His project aim was to use a geolocation API to find the location of any public IP address given and display this on a map. Both the data dumping capabilities of the device were used. The user could either capture a file and run his application to go through the packet capture, visualising transfers on the map sequentially, or could run it while live packet capture was taking place, displaying any traffic in real-time.

The router was used in the live aspect of the web-interface through its MySQL feature. He used SQL queries in Python code to access the most recent packets captured and also to retrieve a group of packets that occurred during some specified timeframe. The router was also used to capture PCAP files and store them separately depending on the type of traffic expected.

Targeting a specific device was described as effortless since a separate PCAP file could be produced for each, filtering out any irrelevant traffic, allowing for more accurate inspections. He described the setup as “quick and easy” after following the given instructions. The packet capturing was described as working flawlessly and he did not have any serious issues with the router. The command line interface of the Python scripts was described as “comprehensive and to the point “and very similar to tools that he had already used in UNIX type of systems. MySQL proved very quick and efficient for carrying out any queries. It was effortless to login to the database through Python code and carry out queries. The first expert had no previous experience with synchronous reading and writing of data, but found the implement procedure straight-forward.

A negative aspect highlighted was the lack of filters for the data-dumping script, which resulted the developer dumping his own SSH traffic when the targeted device was the one from which he was accessing the router’s command line interface. There was no command to filter out the SSH traffic and the expert had to modify the python code on the router. He described the procedure as easy, since the code was very readable and well structured.

Hardware-wise the router sometimes did not boot correctly which reset it to default configuration settings, with which he could not access it, but a power-cycle always fixed the problem.

6.2 Expert 2

The second expert is a design informatics PhD student in Design Informatics who has worked with the master's students on multiple occasions.

He could see this framework being used for individual projects of master's students and ones being housed together which may not have any specific ties, but the data retrieved could be used for comparison. Any IoT setup the master's students will build has potential use for the framework, which includes the Design Informatics, Design with Data, and Histories Technology & Future core curriculum.

In terms of creating visuals it is believed to be a simple method to achieve similar and better results that what the master students used last year and even novices who never used JSON before could probably familiarise themselves with the API in a short period of time.

In terms of analysis his opinion is that the framework will prove useful for his work in IoT and other possible interactive designs. This can be a tool for applying trace ethnography, network ethnography, and digital semiotics. All three would be able to be geared towards network behaviour and security visualisation. The framework also provides a way to teach said methods in the classroom with design interventions currently in place or being built. As a teaching tool it can be useful since the dumped data open the possibility for making quick adjustments to design or service interventions.

Two main down sides were noted. The first was the need to setup everything on the custom separate router, although it was an expected requirement. The second was that while the demo tutorials and documentation are easy to follow they assume a reasonable knowledge base, which some students will be devoid. A more basic introduction is suggested like a workshop with basic terminology, calls and their functions as a way to increase the percentages of potential users in a master class.

6.3 Expert 3

The third expert is a lecturer at the University of Edinburgh.

He described the framework as interesting and he could see a couple applications for monitoring network activity. Neither actually critically revolves around the security aspect, although knowing what is going on and what devices are really attaching to is important and relevant. The first application is for monitoring change in activity and separating human invoked activity from automatic activity, identifying usage patterns can be interesting with social, behavioural, environmental, adaptive significance. Obviously deviant behaviour may indicate a security problem. The second would involve contrasting LAN activity, with external communication.

With the growth in Internet of Things and smart environments, it might be useful to know how much communication devices really need or have with the outside world rather than just the immediate local environment. An example usage would be to enable users to know what their phone is doing even when they are not directly using it. The potential mapping visualisation is considered an interesting project as well. Security-wise, the framework offers a fast and easy way of checking that you are communicating with where you think you should be, an example usage would the verification of the location which the server we would expect to be, in the sense that the Bank of America server is unlikely to be hosted in Russia.

Creating awareness of what connections are being made and frequency or usage rates would be useful for developers. It might even help encourage greater efficiency in bandwidth use. For security, being aware of incoming and outgoing data packets is considered obviously important, along with geolocalisation and the time of establishing communication. It might enable easier detection of potential problems (e.g. which of your 20 devices is the most vulnerable or likely to compromise your entire network).

The expert noted that the implemented alias mechanism does not replace the unique MAC address, but exists only for human readability purposes, but he can see the benefits of its implementation. Similarly, in terms of HCI, attaching devices in some

way analogous to attaching Bluetooth devices would make the process easier and what people are more likely to be used to.

6.4 Expert 4

Is an independent artist, musician and machine-builder working across the fields of live performance and small scale robotics, with an interest in how the philosophy of the open source movement improves access to advanced digital fabrication techniques, and the effects of open versus closed approaches to information sharing.

He describes himself as a relative newcomer to programming and network analysis, his project's aim is to find ways to display information relating to network security in a clear and tangible manner, making a physical object display that is changing state in response to real-time network data.

He used the framework to control a traffic light system of LEDs via an Arduino Yun board, using green-amber-red to show low-medium-heavy network traffic over a rolling 5-second time window, thus implementing the first example in the demos document provided.

The framework made it possible to capture and visualise data quickly and easily even for a novice programmer. He described the API documentation to be exceptionally clear and well organised, as well as suggestive of many ways that the provided functions could be used to drive display objects of more or less complexity. He was able to build a small python application that worked well with the API with minimal problems. While it is a simplistic implementation of the system given the wide range of commands available at the API, as a proof of concept it considered still informative, since it displays network activity at even the slightest use of the monitored device. His implementation was informative as it made him aware that the phone transmitted network data simply on being picked up and having the screen unlocked, activities that the users might not think of as being visible on a network.

When asked about the potential usages and the importance of the framework he replied: "The firmware and API have a great potential in uncovering the normally hidden network activity of a device in everyday use and displaying it simply enough for a non-technical user to understand, raising awareness of the pervasiveness of the

network. There will be many ways for other artists to gain a greater understanding of network activity in IoT devices, enabled by this system. Given further project time, I would be particularly interested in building a display object that responded to the suite of security tests available."

The main problem described is the same as expert 1. There were setup issues with the router's firmware requiring the installation process to be run many times. The likely source of the problem was insufficient power available at the router's USB port for the external hard drive. Since the firmware requires the use of external storage to operate, failure at this point made the installation unreliable and it required reflashing the firmware several times. The problem was fixed temporarily when the external hdd was replaced by a 64GB flash disk and the system behaviour improved immediately. Subsequently, the external hdd was connected via a powered USB hub and this also appeared to improve the system stability.

6.5 Overview

The experts provided mostly positive feedback for the design and believe that the framework has a number of usages, providing easy and streamlined access to the data, potential usages described were in terms with the original aim usages of this project, visualisation of the traffic dominated the input of the experts highlighting the reason data packet dumping is the core of the framework. The problems noted by the experts that actually used the framework or part of it was about the stability of the system, the problem has been identified as being the lack of power both by expert 1 and 4 and the researcher as well.

Chapter 7
Devices evaluation

A number of IoT devices were chosen at random and were analysed using the framework to highlight what is possible in the current state, as well as the lack of security for some of the devices. All the tested devices were found to either be exploitable, use weak encryption or no encryption at all. The evaluation consisted of three steps: port scanning, vulnerability reporting and manual analysis of the dumped packets both in PCAP and JSON to verify the correctness of the implementation. The results were checked against manual execution of the tools used and comparing the end results to verify the correctness of the system.

7.1 Smarter WIFI Kettle

This device has been targeted in the past by many hackers mostly due its insecure design and the widely news covered attack against it which allowed attackers to obtain the WIFI password that the kettle was connected to[17], [18]. The device has two ports listening 23 and 2000, both of them accept unecrypted messages, all data packets can be observed including the “HELLOKETTLE” and “HELLOAPP” messages exchanged with the smart phone application at their connection initialisation phase.

No authentication mechanism is implemented for the commands destined to the telnet service available at port 23 while authentication is required at port 2000, the literature suggests that the default password for the interface available at port 2000, at least when the device is configured using the official application on an android device is “000000” [18],[82].

The device communication is so insecure that third parties have created libraries, allowing the device to be controlled without the manufacturer’s application [83],[84],[85]. While a newer version has been introduced by the manufacturer and the

security of the old device is almost non-existent the device that was tested is still sold from major online retailers[86].

7.2 La Metric Time

La Metric Time is a “smart clock” with a pixel style display, it features Bluetooth connectivity as well and can be used as a wireless speaker. The device is controlled via a smart-phone application and each device is assigned to an online account. It should be noted that version 1.0.21 of the device firmware was tested which is not the latest version.

After reviewing the device results, it was noted that weak ciphers and service identification is possible, an average of 5.5 CVSS score was reported by the vulnerability scanner due to vulnerabilities present in the device. According to the vulnerability scan the device has one vulnerability CVE-2016-3116 which is newer than its latest firmware release version, and uses weak keys for both TLS and SSH thus confirming the results of the custom TLS scan that was conducted as well.

The device proved to be insecure implementation upon further research, while it utilises encryption in its network communication the following problems were found. The device uses default ports for its services and has services unneeded for its operation working at all times. A vulnerable version dropbear SSH server runs at port 22 and a TLS server listens at port 443, both services are compatible with weak ciphers such as RC4 while the vulnerable SSLv3 version is enabled by default.

While this device was the only one tested that tried to obfuscate its traffic, the results suggest that unmasking the traffic and gaining access to the device itself should be possible. It should also be added that the device does not auto-update or require the user to update in order to continue using the smartphone application, thus invalidating the update requirement that has as a result an even more insecure device.

7.3 Smart plug

A smart plug under the brand name of ORVIBO, model number S20 was tested. The plug offered the basic functionality expected from such a device. It could be remotely turned on and off and could setup timers. The plug was the least secure of the tested

devices since it did not use any kind of security or authentication mechanism, other than the initial pairing process. All its communication in conducted by sending UDP packets to port 10000. All the data packets were in plain text and the packets could be captured and replayed, during security audits to the device it was possible to resend a spoofed previously captured packet and control the device. The complete lack of authentication makes the device unsafe to use since anyone with access to network can send it commands. Another unexpected behaviour of the device is that it sends the command received to servers located in London, while it is assumed that this behaviour is in order the manufacturer to monitor the device usage, the fact that the device receives commands in plain text, without any kind of authentication and forwards these commands to a remote server can be seen as a privacy breach. The findings for this device can be verified by previous reverse engineering of the device[87], [88]and its protocol[89], [90].

7.4 Philips hue bridge

Philips has introduced a way to communicate with their smart led lights through a bridge thus creating a layer of abstraction. The bridge is in reality a small computer hosting a server. A RESTful API is exposed from the bridge which the various applications controlling the lights use. The communication from the bridge to lights is through the zigbee protocol.

While the vulnerability scans show no apparent problems in the implementation, the complete lack of TLS is apparent, no call to the API is encrypted thus all commands can be captured using the framework and analysed. Authentication is implemented and the abstraction layer created by the bridge serves its purpose. The design only misses' encryption to provide adequate security, assuming the zigbee communication between the bridge and the lamps is not compromised.

It should be noted that due to rules and constrains the device was not used using the official Philips application since the official application needed to update the device. Two alternatives applications found on Google Playstore were used instead[91], [92],

both used the same API calls when observed in network level to achieve their functionality which was based only on the API available.

7.5 Dragon Touch Y88X

The Dragon Touch tablet is an Android based machine targeting young kids. The device is advertised as providing a "Kid-safe cyber environment"[93]. Upon testing the device, it proved to behave mostly like any other Android device, that means that the tablet has no ports listening and no reported network level vulnerabilities. On the other hand, the device like all Android devices with Google play services installed, sent periodic messages to Google, showing that the privacy of the device was not customised and the child's privacy was never considered. Furthermore, network packets were sent to third party servers that were not in any known Google range, hence it is assumed that the device further breaches the privacy of the user on top of the default Android behaviour, by sending customised data to its makers, all the communications observed were encrypted.

7.6 SSL stripping

SSL stripping as noted earlier is an old attack that was the reason for the introduction of the HSTS, the newer updated version has two unique features which were explained earlier to combat HSTS and hence being able to strip SSL again.

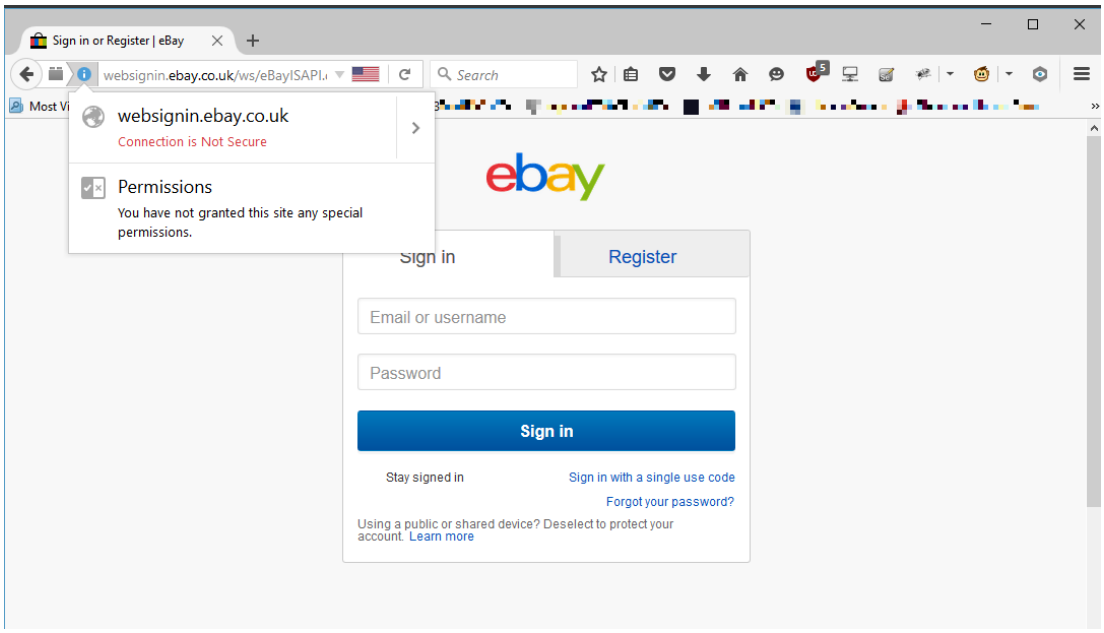


Figure 9 Ebay unsecured after SSL stripping

During testing the router imposed a LAN wide SSL stripping attack and a Windows 10 machine with a fresh install of the latest Firefox was used to test its effectiveness. The attack was successful against: google.com, live.com, tsb.co.uk, halifax.co.uk, hsbc.co.uk and ebay.com while it failed to redirect facebook.com, paypal.com and twitter.com. Manually opening the high-jacked generated domains of the three invulnerable sites had as a result the compromise of the encryption but since someone would have to send the malformed URL to the user and it was not a transparent redirection is not considered successful.

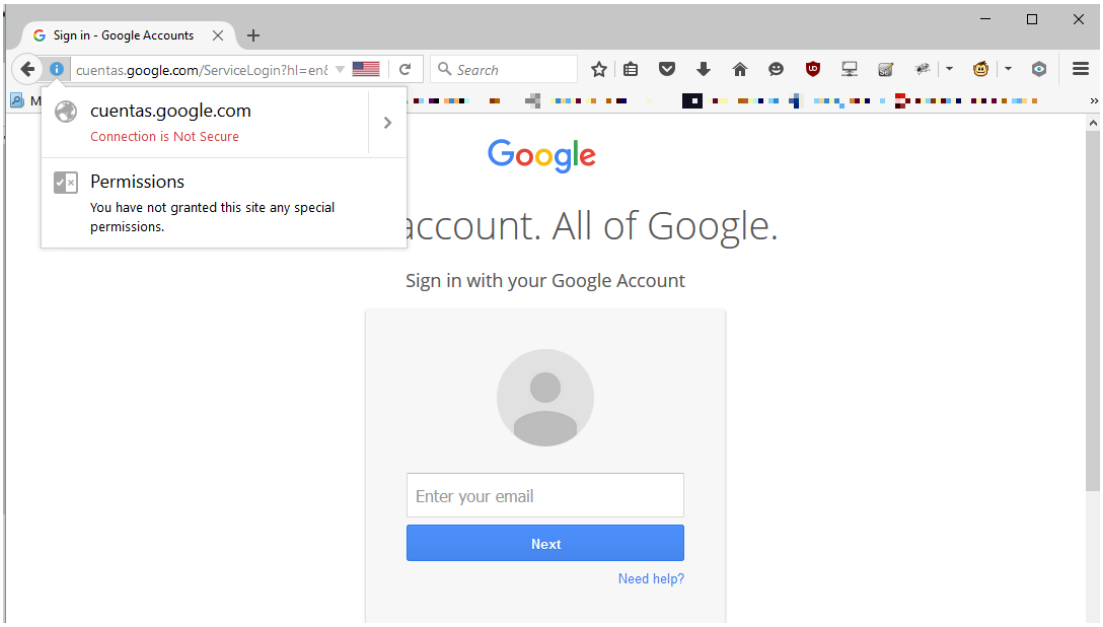


Figure 10 Google unsecured after SSL stripping

In figure 10 one can note the domain “cuentas.google.com” which is a non-existing subdomain that is mapped by the spoofed DNS to the real subdomain “accounts.google.com”. The framework was capable of dumping clear text packets with the credentials from both Google and Live services using the implemented attack technique while using publically available code.

According to the results, SSL stripping is a real danger, since the attack works if implemented correctly in a MITM scenario, the number of websites that are vulnerable is unknown, but the implementation proves that stealing data from HSTS enabled websites is possible. It should be noted that using the “HTTPS Everywhere” plugin by Electronic Frontier Foundation (EFF)[94] or setting the DNS server to a public one proved to be actively deterring the attack and hence should be considered as the easiest way to stop it.

Chapter 8
Conclusion and Future Work

In this paper it was discussed how the lack of a framework that allows easy access across devices to live network flow data and mass security assessment delays the research in the IoT field allowing insecure devices to flood the market. The need for the project thus was established. Theoretical and practical approaches were created in order to provide a generic network level security evaluation.

The feedback of the focus group during the design stages was deterministic for the rest of the design and implementation process while the expert opinion verified the usability and potential of the framework.

Both mass security assessment and live network packet data was discussed researched and implemented on top of open source technologies, inviting future improvement. As part of the original design the functionality present aims to the potential visualisation of the provided data and the automation of data capturing. The functionality implemented exceeded the original expectations and the feedback received shows potential usage of the framework in many fields including the visualisation and education.

The original questions asked were answered and all the set aims were achieved with the exception of the web-interface.

The evaluated devices proved the insecurity prevalent in IoT while showcasing the capabilities available to security professionals by the framework. For the cases of non security or network experts, access to data is possible in a simplified manner while providing a high level representation of security in the form of a simple score.

This paper verifies that the free open source framework introduced can simplify the interaction experts and not alike have with networked devices. A plethora of calls were introduced that can be significantly expanded further providing the necessary toolset to implement simple and complex projects thus allowing for more network

transparency, simplifying the procedure needed by researchers to acquire network data.

8.1 Future work

8.1.1 Full OpenVAS support

The XML interface of the OMP protocol can be easily used in order to create an OpenVAS library that implements the whole set of the software's functionality. The current implementation although it works for the needs of this project, it supports only a small subset of the functionality available, namely only targets and tasks can be created and reports can be parsed. The response classes needed can be generated automatically leaving only the logic to be implemented.

8.1.2 Full nmap support

Nmap offers a python library that allows full interaction and its results can be saved in XML format thus making a full autonomous nmap a possibility, as with the OpenVAS integration, it is possible to create a full interface with nmap.

8.1.3 Full libcap wrapping api

Scapy proved to be an excellent library upon which the project was based and enabled almost all of the packet dumping functionality and TLS scanning. Scapy itself though being a Python wrapper library for libpcap introduces a lot of overhead when compared to native libpcap implementations like tcpdump. During my performance tests while both tcpdump and Scapy were dumping a Youtube high definition streaming session to a PCAP file, Scapy used an average of 60% of the CPU while tcpdump was consuming less than 10% for the same exact function. Creating a Java library that implements some of the core libpcap functionality using the low level native implementation of libpcap available on the router, would considerably speed up the execution and multiply the amount of data the device can dump.

8.1.4 Integration with existing security frameworks.

The Wi-Fi pineapple introduced the concept of a mass marketed rogue AP, their implementation is based on OpenWRT and is used by numerous security researchers worldwide. Using a custom firmware like this as a base with the added capabilities introduced from this framework would allow further expansion. Pineapple Wifi stopped publishing their source since 2014 and removed all their public repositories[95] turning into a corporation, hence their implementation is not any longer easy to acquire or adapt, there are alternative projects such as fruitywifi trying to replicate the functionality on other platforms and may be a viable option for porting to the OpenWRT platform[96],[97].

8.1.5 Privacy score implementation

Although the privacy score system was planned, was never practically implemented, the base for its implementation exists since the framework is aware both if the packets are encrypted and the type of encryption they use, the privacy score in conjunction with the simplified vulnerability score would reach the original goal set by this project.

8.1.6 Web interface

While planned the web interface was present during the design stage, it was dropped in later stages and never materialised. An example web interface was introduced which while sufficient to highlight the possibilities present in the router it does not implement any API functionality. The functionality present allows for the implementation of various interfaces that be displaying information or control the framework itself allowing for more ways of both visualisation and interaction.

8.1.7 Hard drive problems

Finally, there were four systems produced and tested during the practical implementation, while some worked without issues, other implementations faced serious problems with the hard drive disconnecting. It was noted by people who used the system that the hard drive sometimes timeouts and since the state is held in it, the

router appears to have reseted to default settings, while this can be fixed with a power-cycle of the router, it is apparent that the device cannot provide enough power to drive the HDD, to solve this there are a number of options including the use of flash drives, the use of a powered hub or even to move the database to a different machine, preferably the machine that hosts the API. Knowing these problems, and solutions, the power management of the device can be revised to abolish the defect.

Bibliography

[1] Hachem Sara, Teixeira Thiago, and Issarny Valérie, “Ontologies for the Internet of Things.,” ACM, Lisbon, Portugal, 2011.

[2] A. Wood, “The internet of things is revolutionising our lives, but standards are a must,” May 2015.

[3] Edoardo Pignotti and Peter Edwards, “Trusted Tiny Things: Making the Internet of Things More Transparent to Users.,” ACM, Zurich, Switzerland.

[4] Arun Kanuparthi, Ramesh Karri, and Sateesh Addepali, “Hardware and Embedded Security in the Context of Internet of Things.,” CyCAR, Berlin, 2013.

[5] Teng Xu, James B. Wendt, and Miodrag Potkonjak, “Security of IoT Sytems: Design Challnges and Opportunities,” University of California, Los Angeles, 2014.

[6] Z.-K. Zhang, Michael Cheng Yi Cho, and Shiuhpyng Shieh, “Emerging Security Threats and Countermeasures in IoT,” *ACM*.

[7] I. Alqassem, “Privacy and Security Requirements Framework for the Internet of Things (IoT),” in *Companion Proceedings of the 36th International Conference on Software Engineering*, New York, NY, USA, 2014, pp. 739–741.

[8] S. Poslad, M. Hamdi, and H. Abie, “Adaptive Security and Privacy Management for the Internet of Things (ASPI 2013),” in *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication*, New York, NY, USA, 2013, pp. 373–378.

[9] S. Horrow and A. Sardana, “Identity Management Framework for Cloud Based Internet of Things,” in *Proceedings of the First International Conference on Security of Internet of Things*, New York, NY, USA, 2012, pp. 200–203.

[10] S. S. M. Chow, “Functional Credentials for Internet of Things,” in *Proceedings of the 2Nd ACM International Workshop on IoT Privacy, Trust, and Security*, New York, NY, USA, 2016, pp. 1–1.

[11] J. young Kim, “Secure and Efficient Management Architecture for the Internet of Things,” in *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*, New York, NY, USA, 2015, pp. 499–500.

[12] G. Condra, “A Plea for Incremental Work in IoT Security,” in *Proceedings of the 5th International Workshop on Trustworthy Embedded Devices*, New York, NY, USA, 2015, pp. 39–39.

[13] S. E. C. Consult, “SEC Consult: House of Keys: Industry-Wide HTTPS Certificate and SSH Key Reuse Endangers Millions of Devices Worldwide.” .

[14] HP, “Internet of things research study.”

[15] C. Paar, “Constructive and Destructive Aspects of Embedded Security in the Internet of Things,” in *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, New York, NY, USA, 2013, pp. 1495–1496.

[16] R. W. for Metro.co.uk, “Smart TV hackers are filming people having sex on their sofas,” *Metro*, 23-May-2016. .

[17] Patching, Research, Security, Vulnerabilities, Malware, P. cruelty: B. flay L. ransomware for the third time, says wares dead after 2018 Reverser laments crypto game protection, and L. coder released from clink after mega-millions bank raids, “Connected kettles boil over, spill Wi-Fi passwords over London.” [Online]. Available: http://www.theregister.co.uk/2015/10/19/bods_brew_ikettle_20_hack_plot_vulnerable_london_pots/. [Accessed: 13-Aug-2016].

[18] “Why the iKettle Hack Should Worry You (Even If You Don’t Own One),” *MakeUseOf*. [Online]. Available: http://www.makeuseof.com/tag/ikettle-hack-worry-even-dont-one/. [Accessed: 13-Aug-2016].

[19] A. Greenberg, “The Jeep Hackers Are Back to Prove Car Hacking Can Get Much Worse,” *WIRED*, 01-Aug-2016. [Online]. Available: https://www.wired.com/2016/08/jeep-hackers-return-high-speed-steering-acceleration-hacks/. [Accessed: 13-Aug-2016].

[20] “Shodan.” [Online]. Available: https://www.shodan.io/. [Accessed: 13-Aug-2016].

[21] Michael Schearer, “DEFCON-18-Schearer-SHODAN.pdf,” *Defcon 18*. [Online]. Available: https://www.defcon.org/images/defcon-18/dc-18-presentations/Schearer/DEFCON-18-Schearer-SHODAN.pdf. [Accessed: 17-Aug-2016].

[22] A. Staff, “‘Internet of Things’ security is hilariously broken and getting worse,” *Ars Technica*, 23-Jan-2016. [Online]. Available: http://arstechnica.com/security/2016/01/how-to-search-the-internet-of-things-for-photos-of-sleeping-babies/. [Accessed: 17-Aug-2016].

[23] Z. Liu, R. H. Campbell, and M. D. Mickunas, “Security as services in active networks,” in *Seventh International Symposium on Computers and Communications, 2002. Proceedings. ISCC 2002*, 2002, pp. 883–890.

[24] S. Pennefather and B. Irwin, “An exploration of geolocation and traffic visualisation using network flows,” in *2014 Information Security for South Africa*, 2014, pp. 1–6.

[25] R. Fontugne, T. Hirotsu, and K. Fukuda, “A visualization tool for exploring multi-scale network traffic anomalies,” in *International Symposium on Performance Evaluation of Computer Telecommunication Systems, 2009. SPECTS 2009*, 2009, vol. 41, pp. 274–281.

[26] N. Promrit and A. Mingkhwan, “Traffic Flow Classification and Visualization for Network Forensic Analysis,” in *2015 IEEE 29th International Conference on Advanced Information Networking and Applications*, 2015, pp. 358–364.

[27] S. K. Pandey, V. K. Yadav, S. Kumar, S. Verma, and P. Dansena, “Implementation of a new framework for automated network security checking and alert system,” in *2014 Eleventh International Conference on Wireless and Optical Communications Networks (WOCN)*, 2014, pp. 1–7.

[28] L. Harrison and A. Lu, “The future of security visualization: Lessons from network visualization,” *IEEE Netw.*, vol. 26, no. 6, pp. 6–11, Nov. 2012.

[29] J. Ortiz-Ubarri, H. Ortiz-Zuazaga, A. Maldonado, E. Santos, and J. Grullón, “Toa: A Web Based Network Flow Data Monitoring System at Scale,” in *2015 IEEE International Congress on Big Data*, 2015, pp. 438–443.

[30] Y. E. Kwasi and R. Rojas-Cessa, “High-resolution hardware-based packet capture with higher-layer pass-through on NetFPGA card,” in *2014 23rd Wireless and Optical Communication Conference (WOCC)*, 2014, pp. 1–6.

[31] H. Shiravi, A. Shiravi, and A. A. Ghorbani, “A Survey of Visualization Systems for Network Security,” *IEEE Trans. Vis. Comput. Graph.*, vol. 18, no. 8, pp. 1313–1329, Aug. 2012.

[32] S. Elbouanani, M. A. E. Kiram, and O. Achbarou, “Introduction to the Internet of Things security: Standardization and research challenges,” in *2015 11th International Conference on Information Assurance and Security (IAS)*, 2015, pp. 32–37.

[33] J. T. Chiang, J. J. Haas, Y. C. Hu, P. R. Kumar, and J. Choi, “Fundamental Limits on Secure Clock Synchronization and Man-In-The-Middle Detection in Fixed Wireless Networks,” in *IEEE INFOCOM 2009*, 2009, pp. 1962–1970.

[34] R. K. Guha, Z. Furqan, and S. Muhammad, “Discovering Man-in-the-Middle Attacks in Authentication Protocols,” in *MILCOM 2007 - IEEE Military Communications Conference*, 2007, pp. 1–7.

[35] H. A. Mangut, A. Al-Nemrat, C. Benzaïd, and A. R. H. Tawil, “ARP Cache Poisoning Mitigation and Forensics Investigation,” in *2015 IEEE Trustcom/BigDataSE/ISPA*, 2015, vol. 1, pp. 1392–1397.

[36] Y. Joshi, D. Das, and S. Saha, “Mitigating man in the middle attack over secure sockets layer,” in *2009 IEEE International Conference on Internet Multimedia Services Architecture and Applications (IMSAA)*, 2009, pp. 1–5.

[37] C. Jackson, A. Barth, and J. Hodges, “HTTP Strict Transport Security (HSTS).” [Online]. Available: https://tools.ietf.org/html/rfc6797. [Accessed: 09-Aug-2016].

[38] “The LAND attack (IP DOS).” [Online]. Available: http://insecure.org/sploits/land.ip.DOS.html. [Accessed: 11-Aug-2016].

[39] Juniper Networks, “Understanding Land Attacks.” [Online]. Available: https://www.juniper.net/documentation/en_US/junos12.1x47/topics/concept/denial-of-service-network-land-attack-understanding.html. [Accessed: 11-Aug-2016].

[40] Information Sciences Institute, “RFC: 791,” Marina del Rey, California, 1981.

[41] “CVE-2009-3103.” [Online]. Available: https://www.cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2009-3103. [Accessed: 11-Aug-2016].

[42] M. S. M. 105 and 052 Points 2 2 2 Recent Achievements Blog Party Starter Blog Conversation Starter New Blog Rater View Profile, “Security Advisory 2868725: Recommendation to disable RC4,” *Security Research & Defense*. [Online]. Available: https://blogs.technet.microsoft.com/srd/2013/11/12/security-advisory-2868725-recommendation-to-disable-rc4/. [Accessed: 11-Aug-2016].

[43] OWASP, “Transport Layer Protection Cheat Sheet.” [Online]. Available: https://www.owasp.org/index.php/Transport_Layer_Protection_Cheat_Sheet. [Accessed: 11-Aug-2016].

[44] Thai Duong and Juliano Rizzo, “BREACH ATTACK.” [Online]. Available: http://breachattack.com/. [Accessed: 11-Aug-2016].

[45] “nealharris/BREACH,” *GitHub*. [Online]. Available: https://github.com/nealharris/BREACH. [Accessed: 11-Aug-2016].

[46] “CVE-2012-4929.” [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=cve-2012-4929. [Accessed: 11-Aug-2016].

[47] “CRIME Presentation,” *Google Docs*. [Online]. Available: https://docs.google.com/presentation/d/11eBmGiHbYcHR9gL5nDyZChu_-lCa2GizeuOfaLU2HOU/edit?usp=embed_facebook. [Accessed: 11-Aug-2016].

[48] “CVE-2015-3197.” [Online]. Available: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-3197. [Accessed: 11-Aug-2016].

[49] N. Aviram, S. Schinzel, J. Somorovsky, N. Heninger, M. Dankel, J. Steube, L. Valenta, D. Adrian, J. A. Halderman, V. Dukhovni, and others, “DROWN: Breaking TLS using SSLv2.”

[50] Moxie Marlinspike, “Moxie Marlinspike >> Software >> sslstrip.” [Online]. Available: https://moxie.org/software/sslstrip/. [Accessed: 14-Aug-2016].

[51] Vagias, Wade M, “Likert-type scale response anchors,” Clemson International Institute for Tourism & Research Development, Department of Parks, Recreation and Tourism Management, Clemson University, 2006.

[52] “CVSS v2 Complete Documentation.” [Online]. Available: https://www.first.org/cvss/v2/guide. [Accessed: 09-Aug-2016].

[53] Microsoft, “Chapter 1: Service Oriented Architecture (SOA).” [Online]. Available: https://msdn.microsoft.com/en-us/library/bb833022.aspx. [Accessed: 09-Aug-2016].

[54] “Linksys WRT1X00AC/S Series [OpenWrt Wiki].” [Online]. Available: https://wiki.openwrt.org/toh/linksys/wrt1x00ac_series. [Accessed: 11-Aug-2016].

[55] “OpenVAS - NVT Development.” [Online]. Available: http://www.openvas.org/nvt-dev.html. [Accessed: 18-Aug-2016].

[56] “Vulnerability scanners – SecTools Top Network Security Tools.” [Online]. Available: http://sectools.org/tag/vuln-scanners/. [Accessed: 18-Aug-2016].

[57] “Nessus, OpenVAS and Nexpose VS Metasploitable,” *HackerTarget.com*, 22-Aug-2012. [Online]. Available: https://hackertarget.com/nessus-openvas-nexpose-vs-metasploitable/. [Accessed: 18-Aug-2016].

[58] OpenVAS, “OMP: OpenVAS Management Protocol.” [Online]. Available: http://www.openvas.org/omp-6-0.html. [Accessed: 07-Aug-2016].

[59] “the-tcpdump-group/libpcap,” *GitHub*. [Online]. Available: https://github.com/the-tcpdump-group/libpcap. [Accessed: 15-Aug-2016].

[60] “libpcap - The Wireshark Wiki.” [Online]. Available: https://wiki.wireshark.org/libpcap. [Accessed: 15-Aug-2016].

[61] tcpdump, “Tcpdump/Libpcap public repository.” [Online]. Available: http://www.tcpdump.org/. [Accessed: 15-Aug-2016].

[62] “The History and Future of Nmap.” [Online]. Available: https://nmap.org/book/history-future.html#history. [Accessed: 15-Aug-2016].

[63] “python-nmap : nmap from python.” [Online]. Available: http://xael.org/pages/python-nmap-en.html. [Accessed: 15-Aug-2016].

[64] “XML Output (-oX).” [Online]. Available: https://nmap.org/book/output-formats-xml-output.html. [Accessed: 15-Aug-2016].

[65] “Chapter 15. Nmap Reference Guide.” [Online]. Available: https://nmap.org/book/man.html. [Accessed: 15-Aug-2016].

[66] Scapy, “Scapy.” [Online]. Available: http://www.secdev.org/projects/scapy/. [Accessed: 07-Aug-2016].

[67] “tintinweb/scapy-ssl_tls,” *GitHub*. [Online]. Available: https://github.com/tintinweb/scapy-ssl_tls. [Accessed: 15-Aug-2016].

[68] “maxmind/geoip-api-java,” *GitHub*. [Online]. Available: https://github.com/maxmind/geoip-api-java. [Accessed: 15-Aug-2016].

[69] Leonardo Nve, “Asia-14-Nve-Offensive-Exploiting-DNS-Servers-Changes.pdf,” 2014. [Online]. Available: https://www.blackhat.com/docs/asia-14/materials/Nve/Asia-14-Nve-Offensive-Exploiting-DNS-Servers-Changes.pdf. [Accessed: 17-Aug-2016].

[70] “byt3bl33d3r/MITMf,” *GitHub*. [Online]. Available: https://github.com/byt3bl33d3r/MITMf/tree/master/core/sslstrip. [Accessed: 17-Aug-2016].

[71] “LeonardoNve/dns2proxy,” *GitHub*. [Online]. Available: https://github.com/LeonardoNve/dns2proxy. [Accessed: 17-Aug-2016].

[72] “argp/nmapdb,” *GitHub*. [Online]. Available: https://github.com/argp/nmapdb. [Accessed: 15-Aug-2016].

[73] “tintinweb/scapy-ssl_tls/examples,” *GitHub*. [Online]. Available: https://github.com/tintinweb/scapy-ssl_tls/tree/master/examples. [Accessed: 15-Aug-2016].

[74] S. P. Ahuja and R. Quintao, “Performance evaluation of Java RMI: a distributed object architecture for Internet based applications,” in *8th International Symposium on Modeling, Analysis and Simulation of Computer and Telecommunication Systems, 2000. Proceedings*, 2000, pp. 565–569.

[75] D. Jagannadham, V. Ramachandran, and H. N. H. Kumar, “Java2 distributed application development (Socket, RMI, Servlet, CORBA) approaches, XML-RPC and web services functional analysis and performance comparison,” in *International Symposium on Communications and Information Technologies, 2007. ISCIT ’07*, 2007, pp. 1337–1342.

[76] Fielding, Roy Thomas, “Fielding Dissertation: CHAPTER 5: Representational State Transfer (REST),” 2000. [Online]. Available: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm. [Accessed: 11-Aug-2016].

[77] R. T. Fielding and R. N. Taylor, “Principled Design of the Modern Web Architecture,” in *Proceedings of the 22Nd International Conference on Software Engineering*, New York, NY, USA, 2000, pp. 407–416.

[78] IANA, “Protocol Numbers.” [Online]. Available: https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml. [Accessed: 14-Aug-2016].

[79] “CVE-2012-4930.” [Online]. Available: https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2012-4930. [Accessed: 09-Aug-2016].

[80] “CVE-2016-0800.” [Online]. Available: https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2016-0800. [Accessed: 09-Aug-2016].

[81] “CVSS v3.0 Specification Document,” *Qualitative Severity Rating Scale*. [Online]. Available: https://www.first.org/cvss/specification-document#i5. [Accessed: 10-Aug-2016].

[82] “Internet of Things – do you really need a kettle that can boil your security dry?,” *Naked Security*, 20-Oct-2015. .

[83] “iamamoose/moosekettle,” *GitHub*. [Online]. Available: https://github.com/iamamoose/moosekettle. [Accessed: 18-Aug-2016].

[84] “loftdigital/PhiKettle,” *GitHub*. [Online]. Available: https://github.com/loftdigital/PhiKettle. [Accessed: 18-Aug-2016].

[85] “lloydwatkin/ikettle.js,” *GitHub*. [Online]. Available: https://github.com/lloydwatkin/ikettle.js. [Accessed: 18-Aug-2016].

[86] “iKettle Wi-Fi Electric Kettle 1.0, 1.8L, 2400W - Stainless Steel.” [Online]. Available: https://www.amazon.co.uk/iKettle-Wi-Fi-Electric-Kettle-2400W/dp/B00BHXAWX4/ref=sr_1_2?ie=UTF8&qid=1471522373&sr=8-2.

[87] Andrius Štikonas, “Reverse engineering Orvibo S20 socket « Andrius Štikonas,” 2015. [Online]. Available: https://stikonas.eu/wordpress/2015/02/24/reverse-engineering-orvibo-s20-socket/. [Accessed: 11-Aug-2016].

[88] “Glen Pitt-Pladdy :: Blog - Orvibo S20 (Wifi Power Socket) Utility.” [Online]. Available: https://www.pitt-pladdy.com/blog/_20160121-103754_0000_Orvibo_S20_Wifi_Power_Socket_Utility/. [Accessed: 15-Aug-2016].

[89] “Grayda/ninja-allone,” *GitHub*. [Online]. Available: https://github.com/Grayda/ninja-allone. [Accessed: 15-Aug-2016].

[90] “Orvibo Wifi Socket - Pastebin.com,” *Pastebin*. [Online]. Available: http://pastebin.com/0w8N7AJD. [Accessed: 15-Aug-2016].

[91] Rene Wahl, “all 4 hue.” [Online]. Available: https://play.google.com/store/apps/details?id=de.renewahl.all4hue. [Accessed: 18-Aug-2016].

[92] Urbandroid Team, “hueManic.” [Online]. Available: https://play.google.com/store/apps/details?id=com.urbandroid.hue. [Accessed: 18-Aug-2016].

[93] “Dragon Touch y88x plus Pre installed with Bonus Disney Games App and Audio Book kids Tablet.” [Online]. Available: http://www.tabletexpress.com/dragon-touch-y88x-plus-kids-tablet.html. [Accessed: 15-Aug-2016].

[94] “HTTPS Everywhere,” *Electronic Frontier Foundation*. [Online]. Available: https://www.eff.org/https-everywhere%20. [Accessed: 14-Aug-2016].

[95] “Hak5,” *GitHub*. [Online]. Available: https://github.com/hak5. [Accessed: 19-Aug-2016].

[96] “xtr4nge/FruityWifi,” *GitHub*. [Online]. Available: https://github.com/xtr4nge/FruityWifi. [Accessed: 19-Aug-2016].

[97] “FruityWifi.” [Online]. Available: http://www.fruitywifi.com/index_eng.html. [Accessed: 19-Aug-2016].

Appendix

10.1 Appendix A Focus Group

Demographic questions

Q1 What is your age?

Q2 What is your gender?

( ) Male
( ) Female

Q3 What devices in your home are connected to the internet normally?

Q4 What type of OS do you use in your everyday Desktop/Laptop machine? (Check all that apply)

( ) Windows
( ) Linux
( ) Mac OS X
( ) BSD or other *nix like.

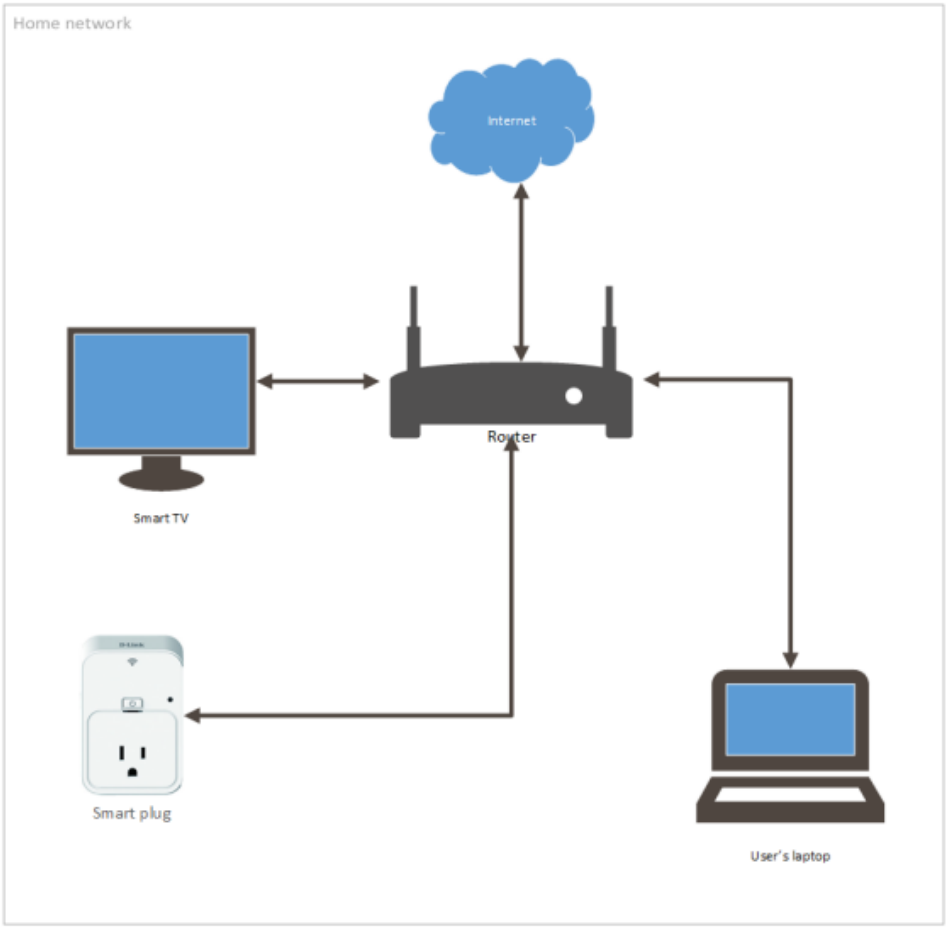
Q5 What kind of browser do you use? (Check all that apply)

( ) Chrome
( ) Firefox
( ) Safari
( ) Opera
( ) Internet Explorer

Q6 How familiar are you with...

	Not at all familiar	Slightly familiar	Moderately familiar	Very familiar	Extremely familiar
Networking	○	○	○	○	○
Packet Capture Software	○	○	○	○	○
Vulnerability Reporting	○	○	○	○	○
Transport Layer Security (TLS/SSL)	○	○	○	○	○

1

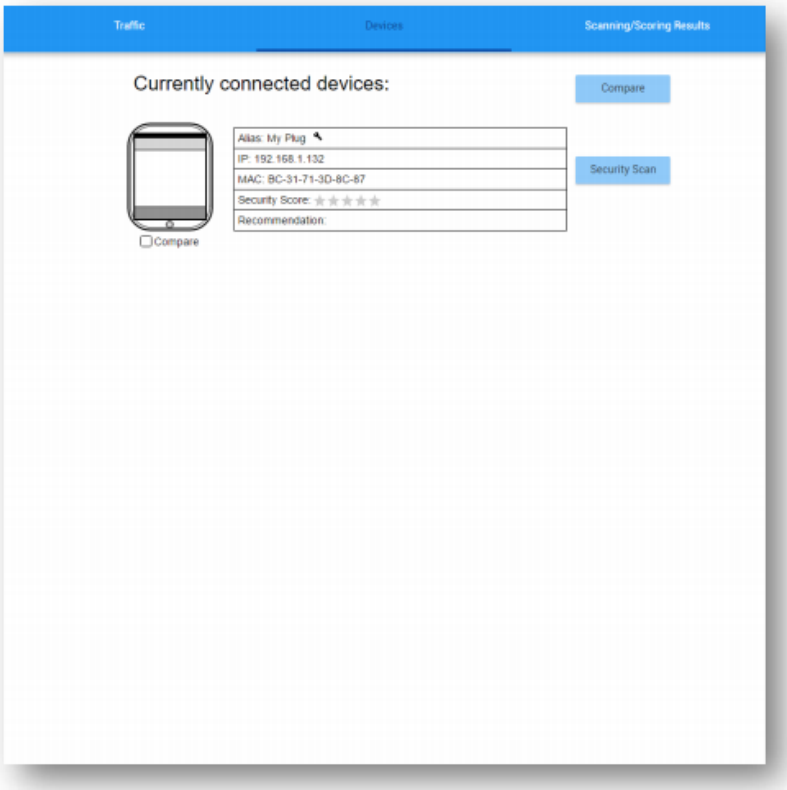


Imagine that you could purchase a router that captures all the network traffic and lets you see security related information about any device connected to it. The router is even able to do basic penetration testing (attacking the objects in your home to see if they are safe to use) and vulnerability scanning (scan your devices for known security flaws).

Q7 How might you use such a system?

Scenario 1

You bought this system and want to evaluate the security of your newly bought "smart plug". Following the instructions provided with the system, you connect the smart-plug device wirelessly and your laptop through one of the LAN ports. Then load the web-interface provided by the router. The first page that loads is the following. Since you only have the smart-plug connected to the router only one device is shown. You pressed the wrench icon and set an alias "My Plug" for this specific device which will be remembered by the router itself and you selected the "Security Scan" to initiate the check.



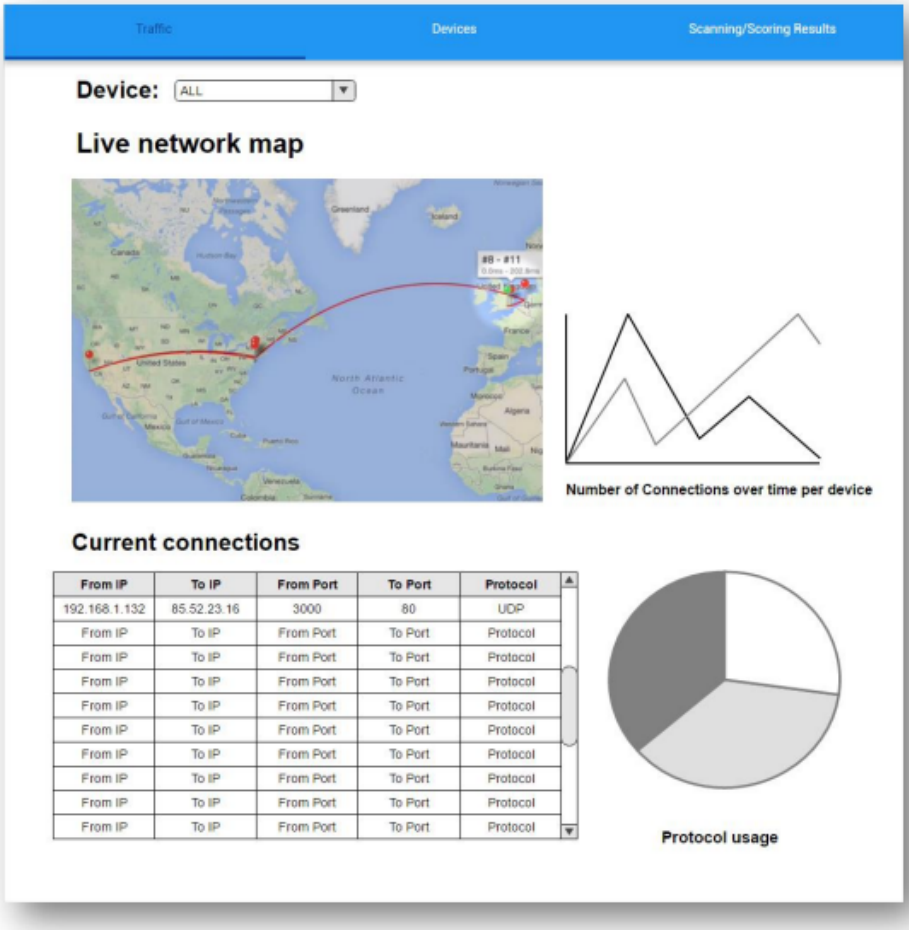
Q8 Is the information about the device and its status adequate?

( ) YES

( ) NO

Q9 What would you change or add?

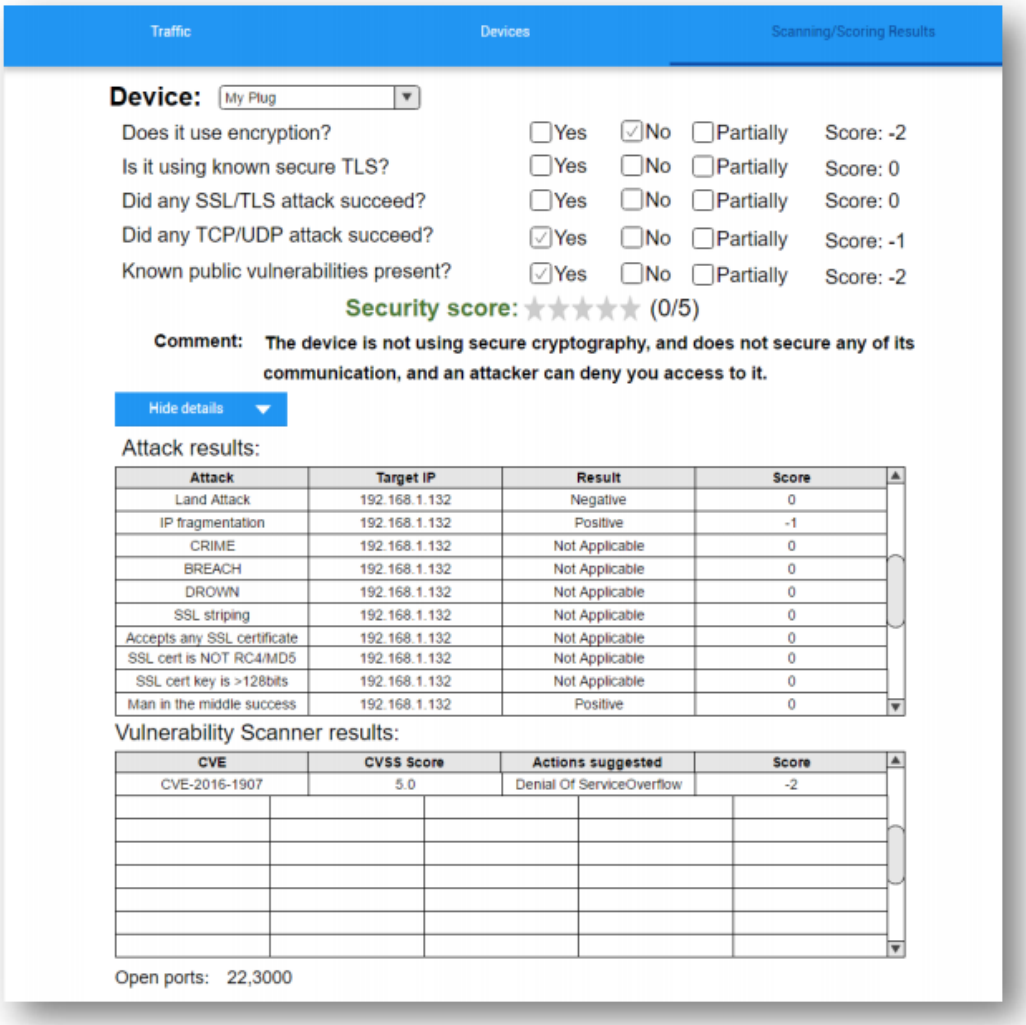
Then you move to the traffic tab which displays a live map where the external connections initiated by the device are displayed. A list of the last active connections exists on the bottom of the map. A pie chart displays the amount of traffic dedicated to each protocol used by the device, the line chart displays the overall trend of bandwidth consumption on all the devices currently connected to the router.



Q10 Which of the following affects your opinion on the device security?

	Strongly Disagree	Disagree	Neither Agree or Disagree	Agree	Strongly Agree
Live network map	○	○	○	○	○
Current connections	○	○	○	○	○
Protocol Pie Chart	○	○	○	○	○
Number of Connections Line Chart	○	○	○	○	○

Finally, you selected the Scanning scoring results tab which includes all the information used to calculate the end score and give the recommendation to the user. You clicked the "Show details" button and the overview came forth.



Q11 Would you consider the score results, a helpful parameter in the evaluation of your devices' security?

( ) YES ( ) NO

Q12 Does the comment section provide you with the appropriate information for your device?
( ) YES ( ) NO

Q13 Would you also desire negative score values regarding the evaluation?

( ) YES ( ) NO

5

Security Score System

Which of the following metrics do you believe is important to determine the device security?

	Strongly Disagree	Disagree	Neither Agree or Disagree	Agree	Strongly Agree	I don't know
Use of encryption	○	○	○	○	○	○
Use of safe TLS versions	○	○	○	○	○	○
Land Attack	○	○	○	○	○	○
IP fragmentation	○	○	○	○	○	○
CRIME	○	○	○	○	○	○
BREACH	○	○	○	○	○	○
DROWN	○	○	○	○	○	○
SSL stripping	○	○	○	○	○	○
Accepts any SSL certificate	○	○	○	○	○	○
SSL cert is NOT RC4/MD5	○	○	○	○	○	○
SSL cert has a key >128bits	○	○	○	○	○	○
Vulnerability Scanner Results	○	○	○	○	○	○

Q14 Would a device that passed these checks qualify as secure to you?

( ) YES
( ) NO

Q15 What metric you believe is important in your opinion but is missing?

Q16 A brief explanation of each attack would be desirable?

( ) YES
( ) NO

Q17 In your opinion could the system possibly harm your devices?

( ) YES
( ) NO

10.2 Appendix B List of API commands

PACKETS: Dump data packets to PCAP files by IP

packets/dump/pcap/ip/{IP}/{interface}

PACKETS: Dump data packets to PCAP by IP

packets/dump/pcap/alias/{alias}/{interface}

PACKETS: Dump data packets to SQL by IP

packets/dump/sql/ip/{IP}/{interface}

PACKETS: Dump data packets to SQL by subnet

packets/dump/sql/subnet/{IP}/subnet/{subnet}/{interface}

PACKETS: Dump data packets to SQL by ALIAS

packets/dump/sql/alias/{alias}/{interface}

PACKETS: Dump Headers Only to SQL by IP

packets/dump/sql/ip/ho/{IP}/{interface}

PACKETS: Dump Headers Only to SQL by subnet produces

packets/dump/sql/subnet/ho/{IP}/subnet/{subnet}/{interface}

PACKETS: Dump Headers Only to SQL by ALIAS

packets/dump/sql/alias/ho/{alias}/{interface}

PACKETS: Stop any data dumping session active

packets/dump/stop

PACKETS: Get if the Dumping session is active

packets/dump/status

PACKETS: Get all the packets

packets/getall/everything

PACKETS: Get the first packet

packets/getall/firstpacket

PACKETS: Get the last packet

packets/getall/lastpacket

PACKETS: Get all the packets since

packets/getall/since/{time_start}

PACKETS: Get packets between two times

packets/getall/from/{time_start}/to/{time_end}

PACKETS: Get all the packets from

packets/getbysessionid/{session_uuid}/all

PACKETS: Get the first packet from selected session

packets/getbysessionid/{session_uuid}/firstpacket

PACKETS: Get the last packet from selected session

packets/getbysessionid/{session_uuid}/lastpacket

PACKETS: Get the first packet from selected session

packets/getbysessionid/{session_uuid}/from/{time_start}/to/{time_end}

PACKETS: Get all the packets from a selected session since

packets/getbysessionid/{session_uuid}/since/{time_start}

PACKETS: Get the packets over a time period for a specific alias

packets/getbyalias/{alias}/from/{time_start}/to/{time_end}

PACKETS: Get the packets for an ALIAS since a time

packets/getbyalias/{alias}/since/{time}

PACKETS: Get packet by its UUID

packets/getbyid/{uuid}/packet

PCAP: Get a list with all the PCAP files captured on the router

pcap/getall

PCAP: Get a PCAP record by its UUID

pcap/getbyid/{uuid}/all

PCAP: Get the location of the PCAP file for the selected UUID

pcap/getbyid/{uuid}/location

PCAP: Get the session id for the selected UUID

pcap/getbyid/{uuid}/sessionid

PCAP: Get the IP of the PCAP file for the selected UUID

pcap/getbyid/{uuid}/ip

PCAP: Get pcap record by its session id

pcap/getbysessionid/{session_uuid}/all

PCAP: Get PCAP location by its session id

pcap/getbysessionid/{session_uuid}/location

PCAP: Get PCAP location by its session id

pcap/getbysessionid/{session_uuid}/id

PCAP: Get PCAP target IP by its session id

pcap/getbysessionid/{session_uuid}/ip

LIVEHOSTS: Start "live host" detection

devices/livehosts/start

LIVEHOSTS: Get "live hosts"

devices/getlivehosts

LIVEHOSTS: Stop "live host" detection

devices/livehosts/stop

DEVICES: Get all stored devices

devices/getall

DEVICES: Sets a custom ALIAS for a specified MAC

devices/getbymac/{mac}/set/alias/{alias}

DEVICES: GET the ALIAS for a specified MAC

devices/getbymac/{mac}/alias

DEVICES: Sets a NEW ALIAS for a specified ALIAS

devices/getbyalias/{alias}/set/alias/{newalias}

DEVICES: GET the MAC for a specified ALIAS

devices/getbyalias/{alias}/mac

DEVICES: GET the ID for a specified ALIAS

devices/getbyalias/{alias}/id

DEVICES: Sets a NEW ALIAS for a specified UUID

devices/getbyid/{uuid}/set/alias/{newalias}

DEVICES: GET the ALIAS for a specified UUID

devices/getbyid/{uuid}/alias

DEVICES: Get a device by its UUID

devices/getbyid/{uuid}

SESSIONS: Get all sessions

session/getall

SESSIONS: Get the interface for a specified UUID

session/getbyid/{uuid}/interface

SESSIONS: Get the mode for a specified UUID

session/getbyid/{uuid}/mode

SESSIONS: Get the target for a specified UUID

session/getbyid/{uuid}/target

SESSIONS: Get the timestamp for a specified UUID

session/getbyid/{uuid}/time

SECURITY: Start an OpenVas scan for IP

security/scan/vulnerability/byip/{IP}/{name}

SECURITY: Start an OpenVas scan for ALIAS

security/scan/vulnerability/byalias/{alias}

SECURITY: Get all OpenVas reports

security/getreports/all

SECURITY: Start a NMAP port scan for IP

security/scan/ports/IP/{IP}

SECURITY: Start a NMAP port scan for ALIAS

security/scan/ports/device/{alias}

SECURITY: Get all open ports

security/ports/getall

SECURITY: Get all TLS results

security/results/tls/getall

SECURITY: Get the TLS results for an IP/DOMAIN

security/results/tls/{IP/domain}

SECURITY: Scan the TLS of an IP/PORT

security/scan/tls/ip/{IP}/{port}

SECURITY: Scan the TLS of an ALIAS/PORT

security/scan/tls/alias/{alias}/{port

SECURITY: Stop any TLS scans running

security/scan/tls/stop

SECURITY: Get the status of TLS scanner

security/scan/tls/status

SECURITY: Get the status of sslstrip

security/attack/sslstrip/status

SECURITY: Start sslstrip

security/attack/sslstrip/start

SECURITY: Stop sslstrip

security/attack/sslstrip/stop

SECURITY: get score from alias

security/score/byalias/{alias}

SECURITY: get score from ALIAS

security/score/byalias/{alias}

SECURITY: get score from MAC

security/score/bymac/{mac}

GEOIP: get location from IP

/geoip/ip/{IP}

PROTOCOL: get protocol description by IANA number

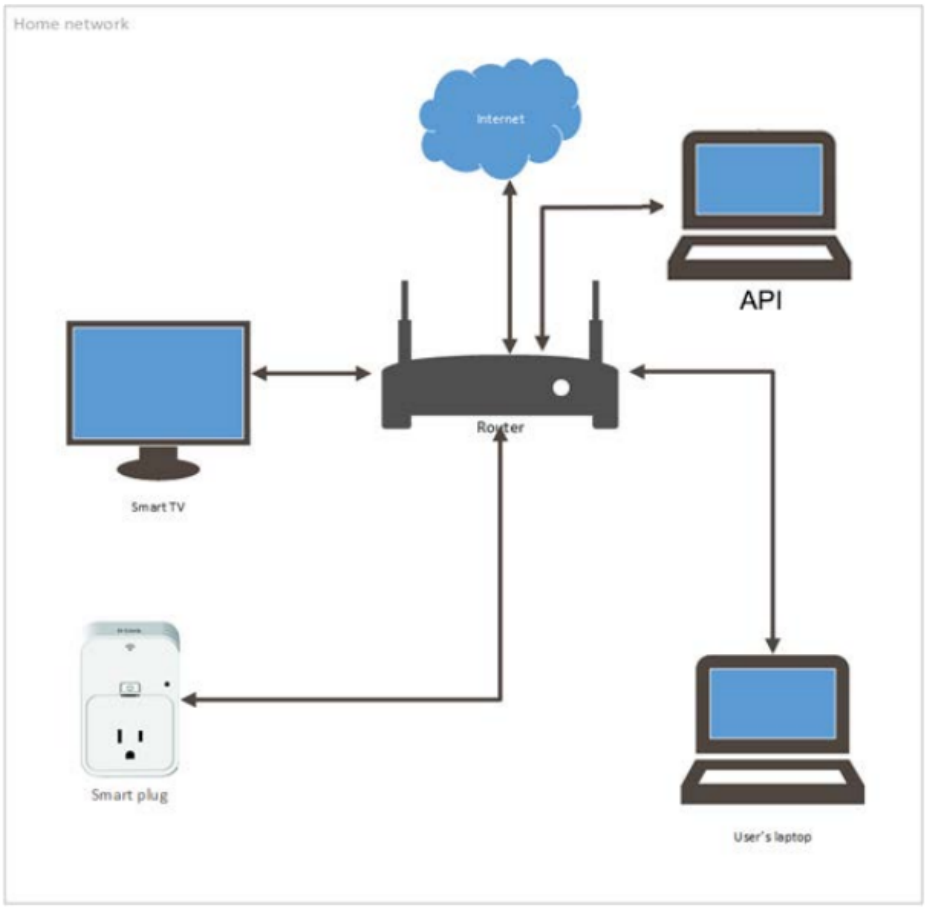
protocol/bynumber/{number}

10.3 Appendix C Experts Demo

Introducing the framework:

The framework consists of a custom router plus a computer which runs the needed software. All the calls are plain HTTP and results are in the JSON format.

In context with its original design purpose the API mostly focuses on network flow recording and simulation and most of its commands are around this subject. While it offers security functionality it is beyond this demo.



The API supports a range of functions, with its major functionality laying in two fields, network packet capture and retrieval, and security. It can be used in real-time to parse information and packets for any device connected to the router's WIFI interfaces and save them for future use.

Security functions include full vulnerability scanning, TLS/SSL certificate analysis and retrieval and TLS vulnerability testing

An alias can be used to represent a device thus simplifying the procedure since there's no need to track IP addresses or remember MAC addresses.

The API is designed in a modular fashion meaning that it works even with some of it features disabled or unavailable. While being easily expandable and scalable. The following examples consider live data scenarios and they build on top of each other to highlight the expandability.

First steps:

The user is assumed to be using one of the Ethernet ports of the router to connect to the framework, while the targeted devices in the wireless interface. Other configurations are possible such as connecting to one of the two wireless interfaces and have the target devices to the other, as well as chaining a second router to the Ethernet of the first thus providing API access to many devices, but using the Ethernet to access the API at all times is encouraged.

The first step in order to start the live functions of the router the API is to issue the livehosts/start command:

http://localhost:9090/api/devices/livehosts/start

Result:

{"status":"command sent"}

This command will start a daemon that searches periodicaly for devices connected to the router.

Now you need to connect the target device(s) to the wireless network provided by the router, then get a list of the currently active devices from the API client:

http://localhost:9090/api/devices/getlivehosts

Result:

[{"mac":"58:63:56:2d:b5:28","ip":"192.168.1.200","hostname":"*","netinterface":"wlan1"},{"mac":"34:4d:f7:7f:37:6a","ip":"192.168.1.144","hostname":"android-3f55a2b4aae0564e","netinterface":"wlan1"}]

There are two devices connected, both provide us with hard to remember names. The router keeps a list of all the devices connected to it while livehosts was running and can set custom names. Let's assign a name to each of the devices. In this case the hostname is the default alias hence:

http://localhost:9090/api/devices/getbyalias/*/set/alias/lametric

result:

[{"id":"cce6f111-5759-11e6-ab66-c05627cc512e","mac":"58:63:56:2d:b5:28","alias":"lametric"}]

Likewise:
http://localhost:9090/api/devices/getbyalias/android-3f55a2b4aae0564e/set/alias/myphone

Result:
[{"id":"6a4269f8-5aa4-11e6-aac1-c05627cc512e","mac":"34:4d:f7:7f:37:6a","alias":"myphone"}]

Now that our devices have aliases we can interact with them in this way as well.

The necessary steps to start the scenarios is complete.

Scenario 1

Imagine a simple project where you want to change the colour of an LED based on number of network packets all the devices send and receive per second.

First you need to start the data dumping, this can be either targeted to an interface as a whole by using the network plus a subnet mask or targeted at a device, we need to capture everything in this example so the command is:
http://localhost:9090/api/packets/dump/sql/subnet/ho/192.168.1.0/subnet/24/wlan1

Result:

{"status":"started data dumping"}

(note the "ho" in the URL it stands for headers-only and instructs the API to dump only the IP headers and not the whole packet, since in this examples no information from the actual payload is needed, this option can make the whole process a lot faster.)

Now that the data-dumping has started to the database it is time to get the time the last packet arrived:

http://localhost:9090/api/packets/getall/lastpacket

Result:

[{"id":"1382f46b-5aad-11e6-af65-c05627cc512e","ipfrom":"192.168.1.182","ipto":"239.255.255.250","macfrom":"44:8a:5b:9d:ed:ba","fromport":57535,"toport":1900,"length":202,"protocol":17,"time":"2016-08-05 02:36:48.458626","bytes":"WGNWLbUoRIpbne26CABFAADK5Gg=","session_id":"eccf6bfd-5aac-11e6-b510-c05627cc512e","unix_time":1470361008}]

Along with a text representation of time, unix_time is used which is time in unix timestamp format. It is an epoch timestamp counting all the seconds since 1970. Converting the time in this scenario is not needed though because we can get the last packet.

Now that we know that the last packet was received at 1470361008 we can get all the packets since then

http://localhost:9090/api/packets/getall/since/1470361008

Result:

[{"id":"22fb2f7a-5aad-11e6-abb6-c05627cc512e","ipfrom":"192.168.1.200","ipto":"129.250.35.250","macfrom":"58:63:56:2d:b5:28","fromport":123,"toport":123,"length":76,"protocol":17,"time":"2016-08-05 02:37:14.412376","bytes":"wFYnzFEuWGNWLbUoCABFwABMRgA=","session_id":"eccf6bfd-5aac-11e6-b510-c05627cc512e","unix_time":1470361034},{"id":"22fd8cf3-5aad-11e6-a6aa-c05627cc512e","ipfrom":"129.250.35.250","ipto":"192.168.1.200","macfrom":"c0:56:27:cc:51:2e","fromport":123,"toport":123,"length":76,"protocol":17,"time":"2016-08-05 02:37:14.427871","bytes":"WGNWLbUowFYnzFEuCABFAABMAAA=","session_id":"eccf6bfd-5aac-

11e6-b510-c05627cc512e","unix_time":1470361034},{"id":"31e3b8b0-5aad-11e6-9014-c05627cc512e","ipfrom":"192.168.1.200","ipto":"129.250.35.251","macfrom":"58:63:56:2d:b5:28","fromport":123,"toport":123,"length":76,"protocol":17,"time":"2016-08-05 02:37:39.424448","bytes":"wFYnzFEuWGNWLbUoCABFwABM5pI=","session_id":"eccf6bfd-5aac-11e6-b510-c05627cc512e","unix_time":1470361059},...rest of the results omitted

Following this logic adding one to the since function time in a loop can generate almost real time traffic data.

Pseudocode example:
i=1470361008

count = 0

while true:

```
packets[] = curl get http://localhost:9090/api/packets/getall/since/I

StaticJsonBuffer<10000> jsonBuffer;

JsonObject& root = jsonBuffer.parseObject(packets);

foreach root["id"]

        count++

if count>100

        LED.setcolor("red")

        .... omitted

sleep(1000)
I++
```

In this example we can get the total number of packets just by counting the id fields.

Comments:

Scenario 2

Imagine a more complex project based on the previous example. Let's assume you want to blink an LED every time a device in the network connects to facebook.

First you need to know the facebook IP, there are many ways to find it, one is to connect from a target pc while in dumping mode and get the ipfrom and ipto value when connecting to facebook, in my case it was 31.13.90.36.

Then since you know the IP all you need is to iterate through the results each second as in scenario 1 and blink the led when ipfrom or ipto = 31.13.90.36

Pseudocode example:

```
i=1470361008

while true:
        packets[] = curl get http://localhost:9090/api/packets/getall/since/i
        StaticJsonBuffer<10000> jsonBuffer;
        JsonObject& root = jsonBuffer.parseObject(packets);
        foreach root["ipfrom"]
                if root["ipfrom"]== 31.13.90.36
                        LED.setcolor("blue")
        sleep(1000)
        i++
```

comments:

Scenario 3

Third and more complex project, visualise the connections on a map highlighting the countries a device communicates to.

The steps are the same until the part where you get the results, this time the ALIAS we set earlier will be used to get the traffic only from the device we want, in this case the android phone that was renamed to "myphone"

http://localhost:9090/api/packets/getbyalias/myphone/since/1470361008

Result:

[{"id":"30495fd7-5aae-11e6-9a0b-c05627cc512e","ipfrom":"192.168.1.144","ipto":"74.125.133.188","macfrom":"34:4d:f7:7f:37:6a","fromport":35359,"toport":5228,"length":85,"protocol":6,"time":"2016-08-05 02:44:46.231917","bytes":"wFYnzFEuNE33fzdqCABFAABVEOk=","unix_time":1470361486,"session_id":"eccf6bfd-5aac-11e6-b510-c05627cc512e"},{"id":"304d535c-5aae-11e6-9108-c05627cc512e","ipfrom":"74.125.133.188","ipto":"192.168.1.144","macfrom":"c0:56:27:cc:51:2e","fromport":5228,"toport":35359,"length":52,"protocol":6,"time":"2016-08-05 02:44:46.257778","bytes":"NE33fzdqwFYnzFEuCABFAAA0584=","unix_time":1470361486,"session_id":"eccf6bfd-5aac-11e6-b510-c05627cc512e"},{"id":"304e5bd9-5aae-11e6-9d9c-c05627cc512e","ipfrom":"74.125.133.188","ipto":"192.168.1.144","macfrom":"c0:56:27:cc:51:2e","fromport":5228,"toport":35359,"length":85,"protocol":6,"time":"2016-08-05 02:44:46.264566","bytes":"NE33fzdqwFYnzFEuCABFAABV588=","unix_time":1470361486,"session_id":"eccf6bfd-5aac-11e6-b510-c05627cc512e"},{"id":"304f502e-5aae-11e6-af38-c05627cc512e","ipfrom":"192.168.1.144","ipto":"74.125.133.188","macfrom":"34:4d:f7:7f:37:6a","fromport":35359,"toport":5228,"length":52,"protocol":6,"time":"2016-08-05 02:44:46.270792","bytes":"wFYnzFEuNE33fzdqCABFAAA0EOo=","unix_time":1470361486,"session_id":"eccf6bfd-5aac-11e6-b510-c05627cc512e"},{"id":"305f0fc0-5aae-11e6-8279-c05627cc512e","ipfrom":"192.168.1.144","ipto":"54.230.11.108","macfrom":"34:4d:f7:7f:37:6a","fromport":44830,"toport":80,"length":60,"protocol":6,"time":"2016-08-05 02:44:46.374037","bytes":"wFYnzFEuNE33fzdqCABFAAA8EzU=","unix_time":1470361486,"session_id":"eccf6bfd-5aac-11e6-b510-c05627cc512e"}... rest of the results omitted

The device is connecting to two IP addresses, after getting the IP from the JSON object the GEOIP function can be called for these IPs.

http://localhost:9090/api/geoip/ip/74.125.133.188

Result:

{"countryisocode":"US","countryname":"United States","citynisocode":null,"cityname":"Mountain View","postalcode":"94043","latitude":"37.419200000000004","longitude":"-122.0574"}

http://localhost:9090/api/geoip/ip/54.230.11.108

Result:

{"countryisocode":"US","countryname":"United States","citynisocode":null,"cityname":"Seattle","postalcode":"98144","latitude":"47.5839","longitude":"-122.2995"}

This information can be used to map the connections to country or city level. It also possible to use the API to create a front end that visualises the network traffic and topology.

comments:

Finishing

Now that the examples are finished it is time stop the data-dumping:

http://localhost:9090/api/packets/dump/stop

Result:
{"status":"sent kill command"}

Make sure that data dumping is closed:

http://localhost:9090/api/packets/dump/status

result:
{"status":"false"}

As well as livehosts:

http://localhost:9090/api/devices/livehosts/stop

Result:

{"status":"sent kill command"}

All these information is available and saved under a session, each session lasts as long as the data dumping script is running, allowing to simulate and replay each session. The id of each session server as the session-id for all data dumps.

http://localhost:9090/api/session/getall

result:

[{"id":"eccf6bfd-5aac-11e6-b510-c05627cc512e","interf":"wlan1","target":"192.168.1.0/24","mode":"db","time":"2016-08-05 02:35:43.531592"}]

Comments:

10.4 Appendix D Focus Group results

	Participants	P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	
Age		28	26	23	21	24	23	27	27	27	0	25.1
Gender	Male	1	0	1	1	1	0	0	1	0	0	5
	Female	0	1	0	0	0	1	1	0	1	1	5
Type of OS	Windows	1	1	1	1	1	1	0	0	1	1	8
	Linux	1	0	1	0	0	0	0	1	0	1	4
	Mac OS X	0	0	0	0	1	0	1	1	0	0	3
	BSD	0	0	1	0	0	0	0	0	0	0	1
Browser	Chrome	1	1	1	1	1	1	1	1	1	0	9
	Firefox	1	0	1	0	1	0	0	1	0	1	5
	Safari	0	0	0	0	1	1	1	1	0	0	4
	Opera	0	0	0	0	0	0	0	0	0	0	0
	IE	0	0	0	0	0	0	0	0	0	0	0
Level of Familiarity	Networking	80	40	80	60	80	60	60	60	20	80	62
	PCS	80	20	80	60	60	60	20	80	20	100	58
	Vulnerabilities	80	20	60	60	60	60	60	60	20	80	56
	TLS	60	20	60	40	60	60	60	100	20	60	54
Adequate Information	Yes	1	1	1	1	0	1	1	0	1	1	8
	No	0	0	0	0	0	0	0	1	0	0	1
Map: Influence	Live Map	80	100	80	80	0	0	80	80	100	100	87.5
	Conections	100	80	40	80	0	0	80	80	80	100	80
	Protocols	80	60	100	60	0	0	80	40	20	20	57.5
	Line Chart	80	60	100	80	0	0	80	60	80	40	72.5
Helpful Score	Yes	1	0	1	1	0	0	1	0	1	0	5
	No	0	1	0	0	0	0	0	1	0	1	3
Helpful Comments	Yes	1	1	1	1	0	0	1	0	1	0	6
	No	0	0	0	0	0	0	0	0	0	1	1
Negative Score	Yes	0	1	0	1	0	0	0	1	1	0	4
	No	1	0	1	0	0	0	1	0	0	0	3

Level of Importance	Encryption	100	100	100	80	100	80	80	100	0	100	93.3
	Safe TLS	100	0	100	100	100	80	80	80	0	100	92.5
	Land Attack	60	0	80	0	0	0	80	0	0	0	73.3
	IP fragmentation	80	0	0	0	0	80	80	0	0	80	80
	CRIME	100	0	80	0	0	0	60	0	0	0	80
	BREACH	100	0	80	0	0	0	60	0	0	0	80
	DROWN	100	0	80	0	0	0	60	0	0	0	80
	SSL stripping	100	0	80	80	0	0	80	60	0	100	83.3
	Any cert	100	0	80	80	100	60	80	100	0	100	87.5
	Not RC4/MD5	100	0	0	0	80	60	80	80	0	100	83.3
	>128bits	100	0	100	0	80	80	80	100	0	100	91.4
	Scanner	100	100	80	100	100	80	80	100	0	100	93.3
Secure	Yes	1	1	1	1	0	0	1	0	1	1	7
	No	0	0	0	0	0	1	0	0	0	0	1
Attack Description	Yes	1	1	1	1	1	1	1	1	1	1	10
	No	0	0	0	0	0	0	0	0	0	0	0
Potential Harm	Yes	0	0	1	0	1	0	1	1	0	1	5
	No	1	0	0	1	0	0	0	0	1	0	3