

TALLINN UNIVERSITY OF TECHNOLOGY
School of Information Technologies

Kevin Patric Schmidt 186358IVCM

EXTRACTION OF FORENSIC ARTIFACTS FROM HOME ROUTERS

Master's thesis

Supervisor: Dr. Hayretdin Bahsi

Tallinn 2020

Author's declaration of originality

I hereby certify that I am the sole author of this thesis. All the used materials, references to the literature and the work of others have been referred to. This thesis has not been presented for examination anywhere else.

Author: Kevin Patric Schmidt

19.05.2020

Abstract

Evidence retrieval is an important part of digital forensics. As the number of connected devices keeps growing, the digital traces left from them are important and needed to be extracted for further analysis. Those devices are mostly connected to a router to achieve high-speed internet connection with unlimited data usage. Therefore, we decided to analyse the gateway (router) between the devices and the internet due to its importance in the digital era.

We based our approach of data extraction on an existing forensic methodology. This approach consists of two different steps. The first one is going to be a manual extraction of forensic artifacts using the available interfaces of the router. The second step is going to be to a logical extraction. We consider hardware-based extraction to be another solution to retrieve artifacts; therefore an analysis of it needs to be done.

This methodology allowed us to retrieve forensic valuable data from the routers and therefore our research could be used as a step-by-step guide on how to gather artifacts. This paper can also be extended by using other extraction techniques that are presented but not used in our own work, because there are some limitations. An example of such a limitation is missing equipment in the lab environment for work at the hardware level.

This thesis is written in English and is 89 pages long, including 6 chapters, 53 figures and 8 tables.

List of abbreviations and terms

IoT	Internet of things
NTP	Network Time Protocol
HTTP	Hypertext Transfer Protocol
JTAG	Joint Test Action Group
ENISA	European Union Agency for Cybersecurity
GPS	Global Positioning System
SIM	Subscriber Identity Module
CuFA	Curated digital Forensic Artifact
MAC	Media Access Control
SSID	Service set identifier
IT	Information Technology
UART	Universal Asynchronous Receiver and Transmitter
DHCP	Dynamic Host Configuration Protocol
OS	Operating System
NMAP	Network Mapper
HTTPS	Hypertext Transfer Protocol Secure
API	Application Programming Interface
URL	Uniform Resource Locator
ISP	Internet Service Provider
GMT	Greenwich Mean Time
IMEI	International Mobile Equipment Identity
IMSI	International Mobile Subscriber Identity
SMB	Samba
XML	Extensible Markup Language
SSH	Secure Shell
UpnP	Universal Plug and Play
SPI	Serial Peripheral Interface

Table of Contents

1	Introduction.....	11
1.1	Motivation.....	12
1.2	Research objective.....	12
1.3	Scope.....	13
1.3.1	Contribution.....	13
1.3.2	Limitations.....	13
1.4	Thesis outline.....	14
1.5	Acknowledgements.....	14
2	Background.....	15
2.1	Theoretical and technical background.....	15
2.1.1	Triage.....	15
2.1.2	Volatile memory.....	15
2.1.3	Non-volatile memory.....	16
2.1.4	Valuable forensic artifacts.....	16
2.1.5	Methodology.....	17
	Manual Extraction.....	17
	Logical Extraction.....	17
	Hardware Extraction.....	18
2.2	Prior and related work.....	18
2.2.1	Routers related papers.....	19
2.2.2	Mobile devices related papers.....	20
2.2.3	Forensic framework.....	21
2.2.4	Summary.....	22
3	Methods and materials.....	23
3.1	Data of evidentiary value.....	23
3.2	Lab environment.....	24
4	Results.....	26
4.1	Huawei B315.....	26

4.1.1 Router specifications.....	26
4.1.2 Manual forensic artifacts extraction.....	28
Web User Interface (UI).....	29
Index page.....	31
Statistics page.....	32
Settings page.....	32
SMS page.....	35
Findings using the manual extraction methods.....	36
4.1.3 API.....	37
Findings using the script automating the manual process.....	39
4.1.4 Hardware-based artifacts extraction.....	40
UART.....	40
4.1.5 Forensic artifacts extraction.....	43
4.2 TP-Link TL-MR6400.....	46
4.2.1 Router specifications.....	46
4.2.2 Manual forensic artifacts extractions.....	47
Web User Interface.....	48
Index.....	48
Network topology.....	49
Wireless Settings.....	50
System logs.....	50
NTP settings.....	51
SMS page.....	51
Backup file.....	52
Findings using the manual extraction methods.....	53
4.2.3 Logical forensic artifacts extraction.....	54
SSH.....	54
Telnet.....	55
Universal Plug and Play (UpnP).....	56
4.2.4 Hardware-based artifacts extraction.....	57
4.3 Linksys EA6350.....	61
4.3.1 Router specifications.....	61
4.3.2 Manual forensic artifacts extractions.....	62

Findings using the manual extraction methods.....	63
4.3.3 Logical forensic artifacts extractions.....	65
SMB.....	65
Firmware analysis.....	66
4.3.4 Hardware-based data extraction.....	67
5 Discussion.....	69
5.1 Router investigation process.....	70
6 Conclusion.....	75
References.....	77
Appendix 1 – Login Bruteforcer for Huawei B315.....	81
Appendix 2 – Huawei B315 information gatherer.....	82

List of Figures

Figure 1. Methods used in our router investigations.....	17
Figure 2. Network topology of our lab environment.....	25
Figure 3. Huawei Router B315 - frontside.....	26
Figure 4. Huawei Router B315 - backside.....	26
Figure 5. Result of NMAP scan on Huawei B315.....	28
Figure 6. Wirshark dump of the Huawei B315.....	30
Figure 7. API answer from:/api/monitoring/status/.....	30
Figure 8. Homepage of the Huawei B315's web interface.....	31
Figure 9. Current connected devices to Huawei B315.....	32
Figure 10. NTP configuration of the Huawei B315.....	33
Figure 11. Logs contained in the Huawei B315.....	34
Figure 12. Device Information about the Huawei B315.....	35
Figure 13. SMS page of the Huawei B315's web interface.....	36
Figure 14. Result of /api/deivce/information request from Huawei B315.....	37
Figure 15. Device Information result from python script.....	38
Figure 16. Script results with credentials – Huawei B315.....	38
Figure 17. Huawei B315 UART interface finding from Ben Zante [28].....	41
Figure 18. USB to TTL Serial cable.....	41
Figure 19. USB to TTL Serial cable - Wires.....	41
Figure 20. USB to TTL Serial cable connected to the motherboard.....	42
Figure 21. udevadm monitor - result when plugging the USB to host.....	42
Figure 22. CP2102 USB to UART Bride Controller.....	43
Figure 23. Boot result from UART interface – Huawei B315.....	44
Figure 24. Linux boot - Huawei B315.....	45
Figure 25. LED states changes – Huawei B315.....	45
Figure 26: TP-Link TL-MR6400 - frontside.....	46
Figure 27. TP-Link TL-MR6400 - backside.....	46
Figure 28. NMAP scan on TP-Link TL-MR6400.....	47

Figure 29: Index page of the TP-Link TL-MR6400.....	48
Figure 30. Login function – TP-Link TL-MR6400.....	48
Figure 31. Currently connected devices to the Wi-Fi - TP-Link TL-MR6400.....	49
Figure 32. DNS and ISP artifacts - TP-Link TL-MR6400.....	49
Figure 33. DHCP system logs - TP-Link TL-MR6400.....	50
Figure 34. NTP settings - TP-Link TL-MR6400.....	51
Figure 35. SMS settings page - TP-Link TL-MR6400.....	52
Figure 36. Device information from backup file.....	53
Figure 37. Network password from backup file.....	53
Figure 38. ISP information from backup file.....	53
Figure 39. Help command result on telnet - TP-Link TL-MR6400.....	55
Figure 40. SSID and password retrieval using telnet - TP-Link TL-MR6400.....	56
Figure 41. Linux version gathered using UpnP - TP-Link TL-MR6400.....	57
Figure 42. TP-Link TL-MR6400 v4 motherboard.....	57
Figure 43. Possible debugging interface - TP-Link TL-MR6400.....	58
Figure 44. Lab environment performing voltage identification.....	59
Figure 45. Linksys EA6350 - frontside.....	61
Figure 46. Linksys EA6350 - backside.....	61
Figure 47. NMAP scan result for Linksys EA6350.....	63
Figure 48. Connected devices on the EA6350.....	64
Figure 49. SMB enumeration - Linksys EA6350.....	65
Figure 50. OpenWrt shell access - Linksys EA6350.....	67
Figure 51. UART pins from EA6350 router - Retrieved from [38].....	68
Figure 52. Linksys EA6350 connected to forensic workstation.....	68
Figure 53. Router investigation process at scene.....	71

List of Tables

Table 1. Forensic artifacts.....	24
Table 2. Huawei B315 Specifications.....	27
Table 3. Manual extraction results of Huawei B315.....	36
Table 4. Automated extraction results of Huawei B315.....	39
Table 5. TP-Link TL-MR6400 Specifications.....	46
Table 6. Manual extraction results of TP-Link TL-MR6400.....	54
Table 7. Pins voltage result - TP-Link TL-MR6400.....	59
Table 8. Manual extraction results of Linksys EA6350 using port 80 and 443.....	64

1 Introduction

According to Gartner, there were 14.2 billion devices connected to the Internet in 2019 [1]. That number keeps increasing every year as we people are more than ever connected. Predictions from Gartner give us an estimation of 25 billion Internet of things (IoT) devices that will be able to communicate over the network in 2021 [1]. This growth phenomenon leaves its traces on the Internet and more particularly on the routers to which these devices connect in order to get a better internet speed or an unlimited data usage. Being able to recover these kinds of traces from a router and use them to recreate a timeline of events that occurred – is what a digital forensics investigator is looking for in his cases.

Digital forensics can be defined as stated in [2, p.25], “the science of identifying, preserving, recovering, analysing and presenting facts about digital evidence found on computers or digital storage media”.

As it was defined above, digital forensic investigation starts from identifying valuable data (also called artifacts) from different devices that are available to the investigators on the crime scene [2, p.25]. Once the identification of those artifacts is done, the investigators are going to preserve the data in order not to alter the integrity of it. The preservation is done by copying/extracting the data to another device. Data integrity during that phase is probably one of the most important points that need to be respected during an investigation. A loss of integrity will transform the once valuable evidence to a non-usable artifact for the case [2, p.25]. Depending on the device the forensic analyst is working on, he will need to recover files that have been deleted intentionally by the user, data from encrypted files or information from the device even if it is damaged or corrupted [2, p.25]. Once the artifacts have been retrieved, the investigator has to analyse them to identify possible malicious actors or threats that could help advance the investigation. That analysis will be then reported in the case and presented to the court [2, p.25].

1.1 Motivation

Digital forensics is still an area that needs to be researched. There exists no real guideline due to the multitude of different devices as we can see in the chapter concerning the prior and related work in that area. We are not going to create such a guideline due to the important number of different branded devices using proprietary software or hardware. Instead, we are interested to investigate different gateways used by most of the connected devices nowadays which are the routers.

A point showing us the importance of such research is the statement of Cisco that says “globally, there will be 1.6 mobile-connected devices per capita by 2023, up from 1.2 per capita in 2018” [3]. The type of devices is large in IoT ranging from phones, watches, TVs, sensors, etc., thus giving new challenges during a digital investigation. All those devices have one point in common being the fact that they can connect to a router (Wi-Fi hotspot) to transmit Voice over Internet Protocol (VoIP), video streams, messages and other types of data to other devices over that gateway.

Cisco also released numbers showing us that in 2023 there will be 628.5 million public Wi-Fi hotspots (including home spots) and a total amount of 616.7 million Wi-Fi home spots [3] making the forensic analysis of routers an interesting research area. Routers being the gateway between the IoT devices and their internet accessibility. This means that valuable data could be stored on that gateway and the extraction of that data could be used as an important evidence for a case.

1.2 Research objective

The main objective will be to perform a forensic analysis on routers to see what valuable data could be retrieved from them. These data could then be used as an evidence or help with further investigation. Data of interest for us in our research can be the current connected users, system logs, time settings, etc. These data will be presented more detailed in the following chapters.

1.3 Scope

We decided that for this work to have more impact and data, we were going to compare three different routers. Two of those routers are using the 4G technology associated with a Subscriber Identity Module (SIM) card to have access to the Internet. That SIM card could also be forensically analysed but we determined that the scope of our paper will focus only on the router itself. We are going with the Huawei B315 and TP-LINK TL-MR6400 as 4G routers. We choose to go with 4G routers due the research gap in the forensic area concerning those devices as it can be seen in the chapter concerning the related works. The third router being the Linksys EA6350 needs a connection using the Ethernet cable connected to a modem to be able to deliver the Wi-Fi capabilities. We decided to add a non 4G router for comparison reasons.

1.3.1 Contribution

During our investigation on what has already be done in the domain of forensics and more specifically routers (that can be found in Chapter 2.2 the “Prior and related work”), we noticed that some papers are not disclosing on how the forensic artifacts were retrieved but are only presenting their findings.

We are contributing to the forensic field with our research by disclosing our results and failures during the investigation of our home routers. The process of valuable data extraction we’ve been through in our routers can be used for other kind of models or brands. Meaning that our paper can be used as a basis for a forensic investigation on routers. It will allow for faster triage when approaching a crime scene.

1.3.2 Limitations

We came across some technical issues when working with the hardware of the routers while trying to extract data from it. Due to our lab environment, we were not able to work entirely at that level. We were still able to use some hardware techniques in our case but as we said in the beginning due to the missing infrastructure in our lab not all of techniques are covered in our paper. Anyways, those techniques not being covered are still presented when encountered during our research.

1.4 Thesis outline

The layout of our research will be as follow :

- Chapter 1 : An introduction representing the scope of our paper, the motivation on why we decided to research in this area, the contribution and the limitations of this work.
- Chapter 2 : A background giving a better understanding of the forensic terms that can occur in the paper, the methodology used in our research and a presentation of the related works in the domain.
- Chapter 3 : Presentation of the forensics artifacts that we are looking for during our research and the topology of our lab environment.
- Chapter 4 : Presentation of the forensics artifacts and describing the processes of extraction that were used during the investigations of the routers. Presentation of the results.
- Chapter 5 : A discussion on what has been achieved with our work and a possible investigation process proposal.
- Chapter 6 : A conclusion to end our research that includes the future work that has still to be done in the router forensics area.

1.5 Acknowledgements

I would like to thank Dr. Hayretdin Bahsi for following me throughout my research and for the exchange of ideas and possible improvements that had to be done for the delivery of this thesis.

2 Background

This chapter contains all the information needed to get a better understanding of our paper and an explanation of the processes used for the extraction of forensic artifacts on the home routers. We are also providing a presentation of the existing works in the area of forensics giving us an idea on what has already be done in this field.

2.1 Theoretical and technical background

We are going to explain some technical terms from the digital forensic field that we come across during our paper. As most of the terms are not commonly used, it will allow people without a great knowledge of the field to understand most of our research. We are also presenting the methodology that is going to be used during our router investigations as some frequently used terms of it is occurring in the text.

2.1.1 Triage

The goal of the digital forensics triage is to give the investigators that are on the crime scene with vital intelligence [4]. This will allow them to directly start with the investigation as they will be able to know which devices are on site, can be gathered, etc. More information about a Korean triage model that can be used may be found in [5].

2.1.2 Volatile memory

The volatile memory is the area of the device allowing to store information at a high-speed. The data can be read/written from that memory only with aid of an electric current, as a loss of power would delete the content of that memory area [6]. An example of a functionality using that area would be a router logging information in the system.

2.1.3 Non-volatile memory

Non-volatile memory can be seen as the area of the memory that is permanently written to at manufacturing of the device [7, p.1-2]. Meaning that even after a reboot (power-off and on) of the router that memory area is not going to be altered. That memory does not need power to be able to retain any kind of information that is stored on it. Nowadays, in consumer routers, the Electrically Erasable Programmable ROMs (EEPROM) is in use allowing the modification byte by byte of that memory array [7, p.2]. This used to allow updates of the firmware as one of the examples on why that type of memory is useful.

2.1.4 Valuable forensic artifacts

The term “artifact” currently does not have any formal definition within the digital forensics domain that is why V. S. Harichandran et al. proposed the use of a new term: “Curated (digital) Forensic Artifact (CuFA)” [8]. Quoting from their report, the following stipulations for the linguistic-conceptual definition of a CuFA [8, p.131] :

- ♦Must be curated via a procedure which uses forensic techniques, such as the one proposed in the Results section [8, p.129-130].
- ♦Must have a location in a useful format (when applicable).
- ♦Must have evidentiary value in a legal proceeding.
- ♦Must be created by an external force/artificially.
- ♦Must have antecedent temporal relation/importance.
- ♦Must be exceptional (based on accident, rarity, or personal interest)

We decided not to use the term CuFA in our work but it was important to show the existence of it as the term artifact does not have a formal definition that could be used. For the purpose of this paper and the usage of “forensic artifacts”, we decided to reuse one of the proposed stipulations from the list above. An artifact in our case will be defined as data that needs to have evidentiary value in a legal proceeding making it evidence.

2.1.5 Methodology

We need to follow an order while using data extraction methods. This means, that we are going to have an order of action to perform during the extraction of artifacts in the routers. We decided to base our methods with the one [9, p.17] proposed by R. Ayers, S. Brothers and W. Jansen. Even if their methodology is aimed at mobile devices, we can apply a modified version of it that can be found below, on the routers that we are investigating.

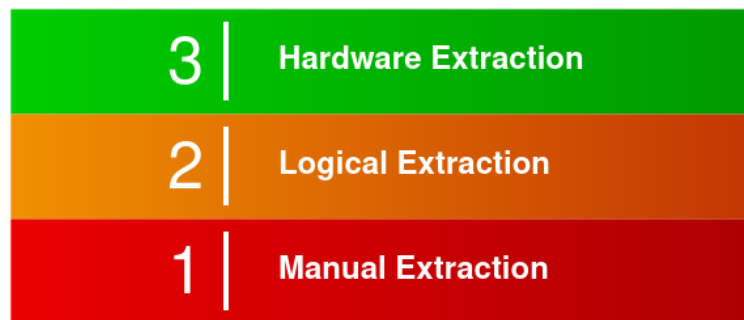


Figure 1. Methods used in our router investigations

Manual Extraction

The first method that we are going to use during our research is a manual approach. This involves all the information that can be gathered by an investigator manually. In our case, it will be by accessing the router's web interface and navigating through it by clicking on the different buttons. Some limitations of this technique have been described by R. Ayers et al. where such an extraction method can be time consuming depending on the amount of data to analyse [9, p.17]. However, such an approach could also alter the data by generating HTTP requests due to the navigation on the web interface. Those requests are probably logged and could alter the logging system of the router being analysed. A manual extraction can allow an efficient digital forensic triage as most of the devices are going to be connected to the router thus allowing to retrieve all the information needed to start with the investigation.

Logical Extraction

We need connectivity between the router and the forensics workstation for logical extraction. That connection can be either wired or wireless [9, p.17]. The issues that can

be created using one of those connections (protocols) need to be known before continuing as it could alter the data contained in the router. Once that connection is created, it can be used to retrieve files, data, etc. by sending commands to it or with the help of forensic tools. This is done by using services (software), that are integrated into the router in our case. Those services could be an example of a router allowing the sharing of resources over the local network. Limitations that can be met during that stage are: missing logins for a certain service (not being able to use it), password protected files, etc.

Hardware Extraction

Hardware-based extraction in our case is the fact of being able to retrieve raw information that is stored in the memory [9, p.17-18]. That extraction technique will allow the investigator to have an overview of the contained file-system in the router. For that we need to locate a certain serial interface. That interface can be found on the motherboard of the router by analysing the board. The serial interfaces are standards used by manufacturers to verify and test the design of their product after manufacture. A universal asynchronous receiver and transmitter (UART) or Joint Test Action Group (JTAG) [9, p.17-18] serial interface is what we need to establish a connection between the router and the investigator's workstation. Upon successful connection and no unexpected events occurred, we will be able to access the file-system of the router. Some limitation for that method is that to create connectivity between the two devices, we will need to shut down the router to work on its hardware, thus leading to the loss of volatile memory data. Documentation about the motherboard is not often disclosed and research/testing needs to be done to retrieve the serial interface and be able to connect to it.

2.2 Prior and related work

Some researches in the domain of routers forensics already exist and we are going to present some of them in this chapter to give us an overview on what has been made and what needs to be done. We are also presenting other types of devices' researches (not only routers) – they sometimes use interesting extraction techniques for the artifacts,

that could be replicated on routers depending on the different software and hardware in use. We also decided to introduce a digital forensic framework as it is important to have a guideline during the case investigation even if it is out of our own research's scope.

2.2.1 Routers related papers

P. Szewczyk presented in [10] and [11] forensic analysis of ADSL routers. Those papers are more than 10 years old but still give us some guidelines on how to perform an acquisition of valuable data in routers. First of all, Dr. Szewczyk presents us on what data we should focus on when retrieving those artifacts from a router. As he stated in the research, we should look for a firmware version in case of an altered system that would be in use [11]. The Network Time Protocol (NTP) settings need to be known to be able to create a timeline of events by comparing the logs (if able to retrieve them) to the current time in use by the router [11]. Those are some examples of artifacts that could be of interest in router forensics. P. Szewczyk also presents on how he achieved to retrieve the valuable data. We can split those retrievals of information techniques into two different groups. The first group would be to use the software available on the router and the second group is the access to the hardware directly by connecting to it on the debugging interface using a serial cable. Software in that research is accessed by using Telnet and the Hypertext Transfer Protocol (HTTP) and for the hardware part he is connecting the motherboard to the computer using the JTAG on the board to a serial port on the workstation. This study is similar to ours but due to the rapidly evolving technology, we are updating it. This research being more than 10 years old, not everything can be followed as he stated it.

G. Horsman, B. Findlay, and T. James presented in their work [12] the forensic examination of five different internet service provider (ISP) supplied home routers from the United Kingdom. They are looking for very specific forensic artifacts like the currently connected devices, the devices that have been connected previously, system configuration and a date/time indication when someone was near the network [12]. Their investigation was performed on a router' factory settings meaning that the passwords were all by default and easily accessible for a manual extraction of valuable data, as it was their only method of data seizure. They also created a standard operating procedure for the investigators on the crime scene which can be interesting for real life

scenarios. The main difference with our research is that they are only focusing on the manual data extraction and analysis.

A research [13] from B. P. Turnbull and J. Slay where they present us different techniques of forensic analysis oriented to the network forensics concerning the 802.11 wireless standards. Even if the paper dates from 2008, some points are still valid. The points concerning the limitations of the embedded wireless device (hardware) are related to our own research. Data extraction difficulties being one of those points as there exists no universal interface on the motherboard to create connectivity between the router and forensic workstation. Forensic analysis has many dependencies. The software versions, the brand of the device, etc. – are still important points that need to be considered during the retrieval of evidence from the router [13, p.4]. Even more today as the technology is evolving rapidly. They also present different ways to approach a crime scene and locate not only possible networks and routers using live analysis tools, but also the limitations concerning the capturing of the network traffic [13, p.3]. The difference between our study and theirs is the fact that they are again only focusing on the manual data extraction and analysis.

An interesting e-book [14] maintained by the European Union Agency for Cybersecurity (ENISA) focusing on the network forensics part of an investigation can be of use in our research because the Wi-Fi router will act as the gateway between all the devices. Therefore, being able to analyse the traffic is an essential work.

2.2.2 Mobile devices related papers

P. Feng et al. have researched on a logical acquisition method based on system-level data migration services provided by the Android mobile device manufacturers [15]. They propose to perform a system-level data migration from an unrooted phone (target device) to an intermediary device (rooted android device) that will then be used for the logical acquisition [15]. A technique using an intermediary device can be of interest depending on the case and the concerned devices.

Another interesting method of forensic artifacts gathering [16] was proposed by A. Levinson et al. where they investigate third-party installed application on an Apple mobile. Third-party applications can hold a lot of information depending on the

authorization they got. As an example, if the app has access to the Global Positioning System (GPS) function of the mobile device it's possible to retrieve the GPS coordinates through the forensic analysis of that application. Bypassing some access restriction due to the proprietary of the mobile device by investigating 3rd party related application is a technique that can be used in our own investigation if we come across a similar situation.

D. Kim and S. Lee in their research [17] propose an interesting forensic analysis for android devices. In their paper, they present the extraction and classification of valuable files (data) on an app-by-app basis in the device [17]. They, therefore, developed an Android Data Taxonomy allowing an investigator to analyse the behaviour of the user based on the application's characteristics and internal structure information [17]. Creating such a taxonomy in a simple router would be difficult as most of the time it only serves one purpose. The purpose being the creation of the internet access but as the routers keep evolving into IoT devices enabling new functionalities. There will be a possibility to create a sort of taxonomy for the data contained in an IoT router.

There are, of course, other works and researches related to forensics but we are not going to expand them in the previous chapters. Some other papers are going to be referenced during our own analysis.

2.2.3 Forensic framework

We just wanted to add a proposed framework [18] from M. Kohn et al. on the different steps that are important during an investigation. In that case, they are presenting computer forensics but it can be extended to other type of devices. They propose to split the investigation into three different processes [18]. The first one being the preparation step where they plan and organize the investigation by examining the standards used in the organization, notifying the correct authorities, read documentation of previous incidents, etc. [18]. The second stage is the investigation itself. The digital forensic analysts are searching for and identifying evidences on the device and extract them [18] to not alter the original content to be able to analyse it without violating the integrity of the evidence. The last and third step is the presentation part where the investigators will present their findings and prove their analysis (evidence is relevant to the case).

2.2.4 Summary

We were able to retrieve much information due to the previous works that have been made in the digital forensic area for routers and IoT devices. We are going to try to apply the different methods of data acquisition and analysis in our research. We already know that some of the methods, such as the ones for the mobile phones, will differ or not be feasible in the case of the routers we are testing, but they were worth mentioning as they are related to the domain of IoT.

3 Methods and materials

As we already stated in the introduction, this research is going to focus on the forensic analysis of the gateway between IoT devices (also PCs, etc.) and the Internet. In the following chapters, we are providing an explanation of the valuable data we are looking for during our work and how we designed our lab environment for the different experiments.

3.1 Data of evidentiary value

As stated in the previous chapter concerning the valuable forensic artifacts, the goal of a digital forensic investigation is to find data that are of interest for the current investigation. In this section, we are going to go through researches that have been made in the past and create a table in which we will put the data that has been found. That table can then be used as a reference for our research to see if the extracted data from the router could be used as an evidence.

We went through different papers: [10], [11], and [19], that perform router forensics and present different valuable data that we should look for during an investigation. Then we were able to generate the following table due to the information gathered from different researches as the basis of data with evidentiary value that can be used in a case.

Table 1. Forensic artifacts

Forensic artifacts	Why we look for that
Firmware type/version	Looking for any alteration of the system
NTP settings	Performing investigation using the correct time zone
IP addresses of authenticated users	Able to associate an action to a certain user
Media Access Control (MAC) address of connected devices	MAC address is unique to a device and therefore – a significant artifact for further investigations
User names generated from the connected devices	Names can often be linked to a certain person
Configuration (files)	Service set identifier (SSID), SIM-card information, MAC address, etc. could allow the investigator to have more options to dig into
Logs	Recreating a timeline of events that occurred, could contain user information that could be linked together

As we can see on the Table above, a router holds information that can be valuable for a case. Depending on the sort of case, the digital forensic analyst is working on the artifacts value will change. After a malware infection on an Information Technology (IT) infrastructure the artifacts that the investigator will be looking for will differ from those of a murder crime scene. Hence, the importance of a certain artifact will vary depending on the case the analysts are working on.

3.2 Lab environment

As we already stated, the goal of this research is to extract valuable forensic artifacts from a router for an investigation. We needed therefore to setup a lab environment to simulate real user's interaction with their different devices to generate real-life looking data and see if some valuable information could be extracted from it. Therefore we are going to refer to the following network topology for the different routers' analyses.

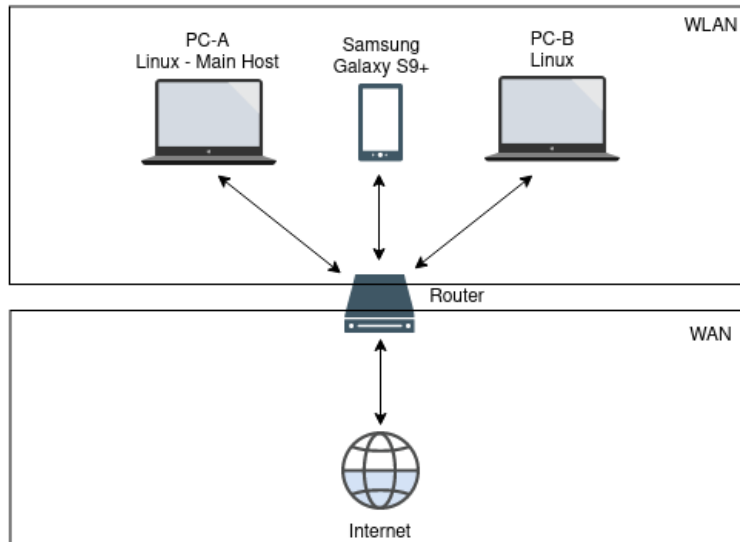


Figure 2. Network topology of our lab environment

The environment, that we decided to create to do our research, is based on two computers, one smartphone and a Wi-Fi router for access to the internet. An important point in our analysis is the fact, that all the routers that are being examined, use factory settings, as most of the ordinary users would not make any modification on those settings, not even changing the administrator's password (as it was found by Broadband Genie in their survey [20]).

We decided to generate traffic between those different devices to see if valuable data can be gathered. When we refer to traffic in our research, it represents data that could be generated from a real-life user. Examples of such data streams would be sending/receiving emails, sending of messages with the help of the smartphone, watching video online, etc.

We are not going to assign IP addresses to the machines directly on our topology as the routers are using the Dynamic Host Configuration Protocol (DHCP) to distribute the IP addresses randomly (first connected is the first served) between all the machines that connect to them. During the following scenarios, we are directly going to associate the IP address we are analysing with the machine names from the topology.

4 Results

In this chapter, we are going to present the forensic analysis of our different routers that were presented in the scope section. Firstly, we are showing the router's specifications. Secondly, we are going to describe the forensic artifacts that were found using the manual, logical and hardware approach. Finally, at the end of each approach, we are presenting the results in a table that summarises, what as exactly be found.

4.1 Huawei B315

4.1.1 Router specifications

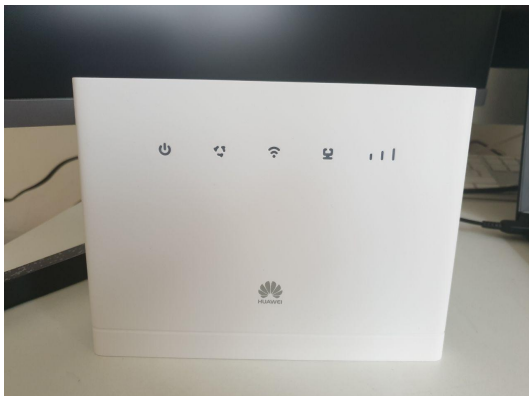


Figure 3. Huawei Router B315 - frontside



Figure 4. Huawei Router B315 - backside

Table 2. Huawei B315 Specifications

General Information	
Manufacturer	Huawei
Manufacturer no.	B315S-22
Router Properties	
Router type	LTE / 4G router
Mobile Internet Properties	
Sim card type	Default SIM
Mobile phone standard	3G - UMTS, 4G – LTE, GSM/2G
Max. data transfer rate	150 Mbit/s
Operating system compatibility	Mac, Linux, Android, iOS, Windows
Max. number of clients	32x
Wi-Fi standard	
Wi-Fi standard	Wi-Fi 2 / 802.11a, Wi-Fi 1 / 802.11b, Wi-Fi 3 / 802.11g, Wi-Fi 4 / 802.11n
2.4 GHz transmission rate	300 Mbit/s
Wi-Fi encryption	WPA-PSK, WPA2-PSK
Network Interface	
RJ45 LAN	4x
RJ11 Phone	1x
External antenna port	Yes

The specifications¹ found are not of a forensically importance. Information like the size of the memory (volatile and non-volatile), the operating system (OS) in use, software versions, etc., would be a good start for our investigation. Unfortunately, that kind of information is not disclosed publicly. We will need to use a black box testing [21]

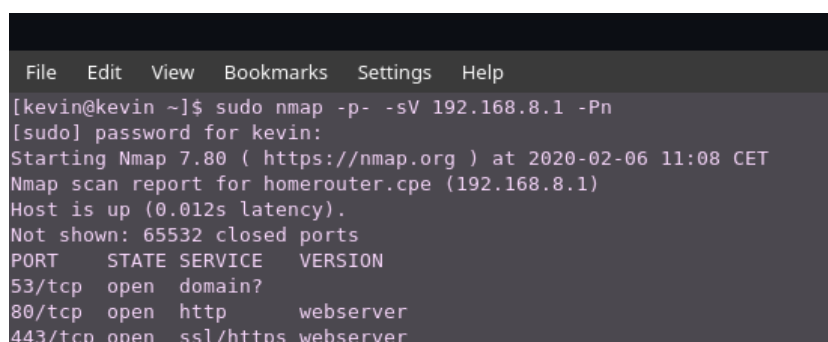
1 Specifications retrieved from : <https://www.digitec.ch/en/s1/product/huawei-b315-routers-5661550>

approach to understand the internals of the router to then be able to perform a complete digital forensic investigation.

In order to perform the black box testing on the B315, we will follow as we already stated in the previous chapter with a certain methodology. First, we are going to perform a manual extraction of valuable data. The second step will be an analysis of the motherboard to find a serial (debugging) port to create a connectivity between the router and our forensic workstation as there is no services (in the router) allowing the logical approach. All those steps and results are going to be presented in the following chapters.

4.1.2 Manual forensic artifacts extraction

We are going in a first phase to extract forensic artifacts from the router manually. This step is performed by navigating and clicking on all the available interfaces of the Huawei router. Those interfaces are network service software installed on the router allowing an ordinary user to perform modifications through a user interface (changing settings, upgrading firmwares, etc.). A way to find the running services on a router is to use the Network Mapper (NMAP) tool. It is important to mention, that an nmap scan of a host generates a lot of traffic and packets that could be logged by the router, thus leading to a possible loss of evidences. As our research is not a real life case and we are looking for the possible artifacts that can be obtained on the routers, we are going to ignore that fact. If it were to be a real investigation, probably, the best solution would be simply login to the network and then access the web interfaces by providing the routers IP on the web browser for further analysis.



```
File Edit View Bookmarks Settings Help
[kevin@kevin ~]$ sudo nmap -p- -sV 192.168.8.1 -Pn
[sudo] password for kevin:
Starting Nmap 7.80 ( https://nmap.org ) at 2020-02-06 11:08 CET
Nmap scan report for homerouter.cpe (192.168.8.1)
Host is up (0.012s latency).
Not shown: 65532 closed ports
PORT      STATE SERVICE  VERSION
53/tcp    open  domain?
80/tcp    open  http     webservers
443/tcp   open  ssl/https webservers
```

Figure 5. Result of NMAP scan on Huawei B315

As we can see on the nmap command result, the main software, contained in the router allowing a forensic analyst to perform a manual extraction of artifacts, is represented by the web interfaces located at port 80 and 443. The services running on those ports are the same, the only difference is that port 443 uses HyperText Transfer Protocol Secure (HTTPS) meaning that the traffic is encrypted. The following manual artifacts extraction is based on port 80 as the services are identical to the one, located at 443.

Web User Interface (UI)

The web interface is used to manage the router as we explained previously. Examples of the router management would be the modification of the administrator password, SSID, IP addresses or the viewing of the logs contained in the router. The logs available from the web interface are just informative. As an example, an admin connection to the router from the web interface is logged. We will go into the details of those logs in the next sections.

In a forensic way, altering the memory (volatile in this case) is something that should be avoided and can lead to the suppression of prior data that could have been used as evidence in the court. Unfortunately, we do not have any information on how big that memory is or on how to access it directly. Huawei does not disclose that sort of information publicly. We decided to call their customer service directly to see if there was a way for us to create a backup/dump of the configuration of the router for further data analysis but their answer was negative. We decided to continue with our investigation using the web interface even if the volatile memory could be altered from the HTTP requests that are made while we are navigating in the page.

As we can see in Figure 6, the router is generating traffic when we are browsing the interface (also when we are not doing anything – explanation can be found below). This Wireshark dump was taken from our host PC-A (192.168.8.100) while connected to the router's (192.168.8.1) network and browsing the available web interface.

No.	Time	Source	Destination	Protocol	Length	Info
52	0.633762524	192.168.8.100	192.168.8.1	TCP	66	57562 → 80 [ACK] Seq=986 Ack=1977 Win=501 Len=0 TSv
53	0.650700348	192.168.8.100	192.168.8.1	HTTP	559	GET /api/dialup/mobile-dataswitch HTTP/1.1
54	0.664151375	192.168.8.1	192.168.8.100	TCP	416	80 → 57562 [PSH, ACK] Seq=1977 Ack=1479 Win=8712 Len=
55	0.664692597	192.168.8.1	192.168.8.100	HTTP/...	151	HTTP/1.1 200 OK
56	0.664794976	192.168.8.100	192.168.8.1	TCP	66	57562 → 80 [ACK] Seq=1479 Ack=2412 Win=501 Len=0 TS
57	0.693585781	192.168.8.100	192.168.8.1	HTTP	564	GET /api/monitoring/traffic-statistics HTTP/1.1
58	0.693679366	192.168.8.100	192.168.8.1	HTTP	551	GET /api/net/current-plmn HTTP/1.1
59	0.693798734	192.168.8.100	192.168.8.1	TCP	74	57604 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACI
60	0.694967147	192.168.8.1	192.168.8.100	TCP	74	80 → 57604 [SYN, ACK] Seq=0 Ack=1 Win=14480 Len=0 M
61	0.694986681	192.168.8.100	192.168.8.1	TCP	66	57604 → 80 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=
62	0.695083859	192.168.8.100	192.168.8.1	HTTP	564	GET /api/monitoring/traffic-statistics HTTP/1.1

Figure 6. Wirshark dump of the Huawei B315

We observed that the router (192.168.8.1) is making us (192.168.8.100) do the same requests every five seconds approximately. Those requests that we made are done to be able to refresh some data on the router as example the up-time of our own connection to the internet. Those are some of the requests as examples that are done in the background automatically, without us performing any actions:

- *GET /api/monitoring/status*
- *GET /api/monitoring/check-notifications*
- *GET /api/monitoring/traffic-statistics*
- ...

As we can see, the URL of the request contains the abbreviation application programming interface (API) which means that in the back-end there is a server (the router in that case) that allows us to perform HTTP requests to retrieve information. We decided to navigate to one of those URL (Uniform Resource Locator) using a web browser. The result can be found below:

```
901 2 19 3 0 0 10.200.102.243 10.200.102.244 1 32 32 2 1 1 101 5 0 0 cpe 0 1 0
```

Figure 7. API answer from:/api/monitoring/status/

At first glance, that answer does not seem to give us much information but we are going to go more into details of the API calls answers in the next chapter.

Index page

Going back to the web interface, what data can be retrieved from it by a manual extraction method. When we first navigate to the page of the router (IP address) at 192.168.8.1 in our case, we are prompted with the following page.

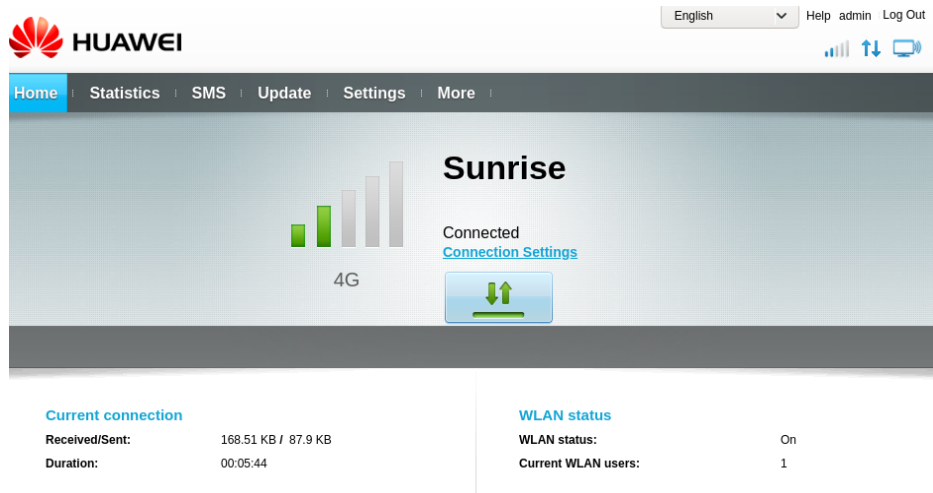


Figure 8. Homepage of the Huawei B315's web interface

The first page allows us to retrieve the following information: The ISP, data usage, up-time duration of the router and the number of connected users. That page can be accessed without any credentials. An interesting forensic artifact would be the ISP that is displayed. We are then able to investigate further by contacting them directly and see if they can provide us access to the router or even logs. Depending of their established policy ISPs can store logs that would be valuable depending the case [22, p.7].

To navigate to the other pages like “Statistics”, “SMS”, “Update”, “Settings” and “More”, we need to provide the username/password of the admin user. By default, the credentials as we already stated are admin/admin and are known to us. We know the credentials because they can be easily researched online as most of the manufacturers are putting the login information on their website. If those credentials are not known and can not be retrieved, some data can still be extracted despite that lack and will be presented in the next chapter concerning the API. Once logged in, we are able to navigate without restrictions. We are just going to present the options (pages) that might hold interesting forensic artifacts.

We need to add that after 3 wrong inputs of admin/password combination while trying to login we are getting prompt back with a “Maxium retry attempts reached. Please try again later”. In our tests, the lock lasts for approximately 10 minutes. Making brute force attacks difficult if the default passwords has been changed.

Statistics page

The statistics page contains the data of the currently connected devices. Some of the forensic artifacts of interest in that page would be the host names, MAC addresses and the duration of their connection with the network. We noticed that when we disconnected our two other devices from the network, we are removed from that list and no traces could be found on our previous connection as it can be seen in the following Figure.

Connected WLAN Clients

ID	IP Address	Host Name	MAC Address	Duration	Operation
Clients: 1					
1	192.168.8.102	kevin	44:85:00:A0:FD:DE	00:01:55	Block

Figure 9. Current connected devices to Huawei B315

Settings page

The settings page is used to modify the existing configuration of the B315 router. Among the available modifications for the user, there are: modification of the IP addresses pool, SSID, etc. Keeping in mind that modifying those data will alter the forensic investigation and therefore should only be used to view the data that is contained in it.

Among the forensic artifacts that can be gathered in that page, is the SSID as we said. But this is most often already known or need to be identified in the beginning of the digital forensic investigation against routers.

Another important valuable data for our investigation is the NTP settings. This one is a crucial artifact to gather to be able to recreate a correct timeline of events. If the

timezone is set to GMT+2 (Greenwich Mean Time) and we are currently working in another timezone, the conversion needs to be known to determine at what time a certain action occurred. In the Figure below, it can be seen that the current time zone in use is GMT-12:00.

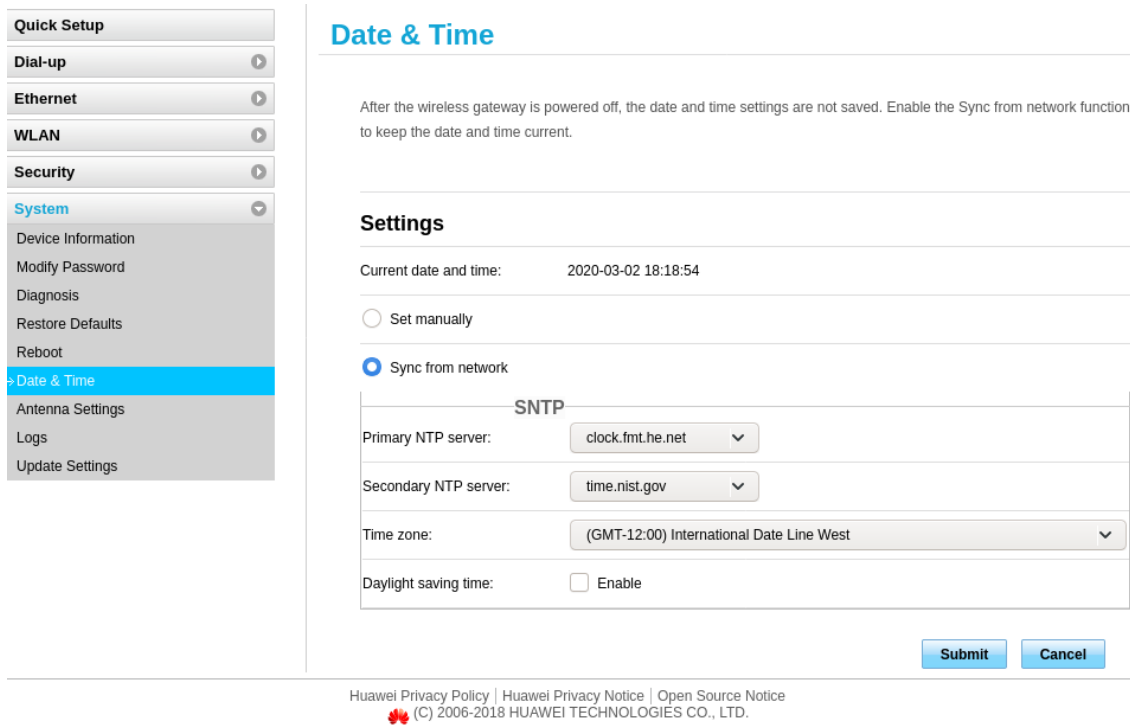


Figure 10. NTP configuration of the Huawei B315

As we stated in the beginning of the B315 investigation, there exist some logs that are stored in the non-volatile memory and thus still available after a reboot. These logs contain informative, warning, error or critical information. Modification of the behaviour on what is going to be logged is not possible. The logging system is enabled by default and its' disabling is not possible. An example would be the admin that connects or some settings that were changed (password change is one of those settings being logged by the router and seen as critical). As it can be seen below, we are able to notice all the successful and failed connection to the administration page.

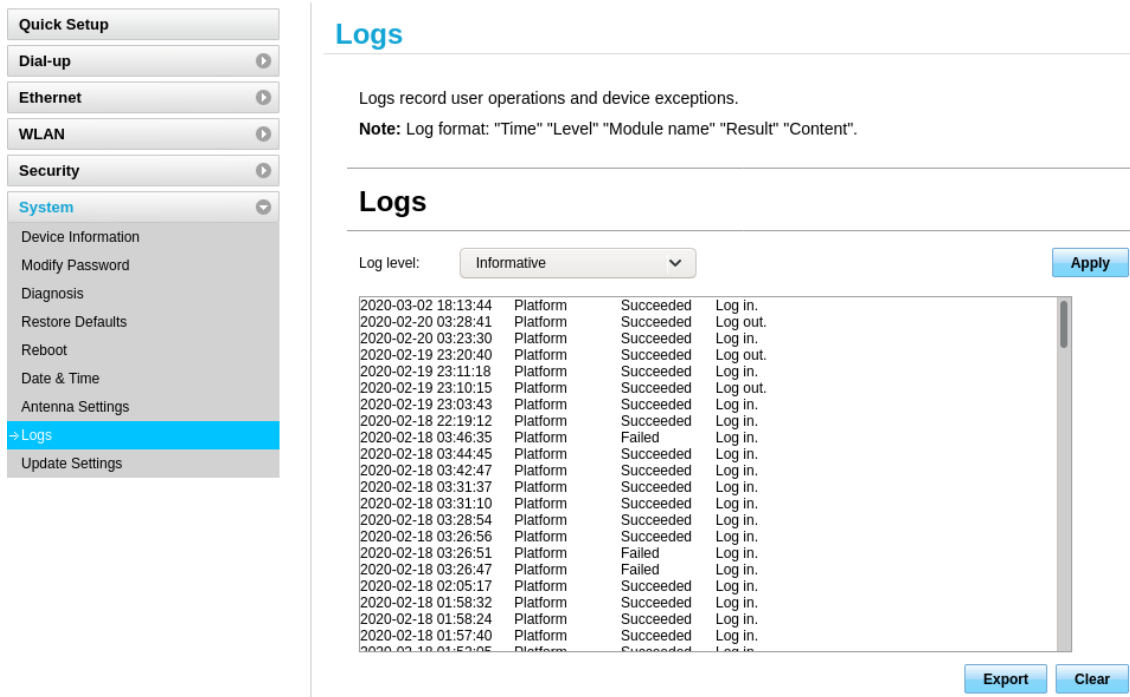


Figure 11. Logs contained in the Huawei B315

We decided to test the capabilities of those logs and we noticed that sometimes a login is not logged when the session is lost on the web page and a reconnection is done. When reconnecting, the username and password have to be entered again. We can not explain that behaviour exactly: why sometimes we can bypass that logging of connection but we decided to store that information in our research anyways. By deduction, the session may have expired but when reconnecting, the same one is used again thus not stored in the logs.

We created a python script that can be found in [Appendix 1] to see how many “lines” of logs are stored before getting deleted. The script allows us to create a desired number of logins against the Huawei router that are then going to be logged in the system. After some tests, we noticed that the limit of logged actions is 500 before getting deleted from the non-volatile memory. That deletion of logs is done chronologically meaning that the oldest entry is going to be deleted first. We noticed that there is no limit in the duration of storage at least during our research. We were able to see logs that were two months old.

The last forensic artifacts, that can be found in the “settings” page, represent us the device information. The International Mobile Equipment Identity (IMEI), International Mobile Subscriber Identity (IMSI) and MAC address of the router are the data that are valuable in a forensic analysis. It is interesting to keep a trace of the firmware version that could be used to identify any altered system in use. The firmware’s version could also be significant to check if there is any known security vulnerability. Security vulnerabilities can be used to bypass restriction and access the file system where more artifacts could be found.

Device Information



<input type="button" value="Refresh"/>	
Device name:	B315s-22
IMEI:	867962039713949
IMSI :	228024559505002
My number:	Unknown
Hardware version:	WL3B310FM01
Software version:	21.333.01.00.00
Web UI version:	17.100.09.00.03
LAN MAC address:	88:F8:72:98:8D:0E
WAN IP Address:	10.49.79.128

Figure 12. Device Information about the Huawei B315

SMS page

4G routers give us the possibility to send and receive SMS like a phone. In our case, we just had a SIM card allowing us internet access without the SMS feature enabled. Because of the usefulness of that feature, it is necessary for us to present it even if no messages are present in our box. As it can be seen in the Figure below, there is no current SMS.

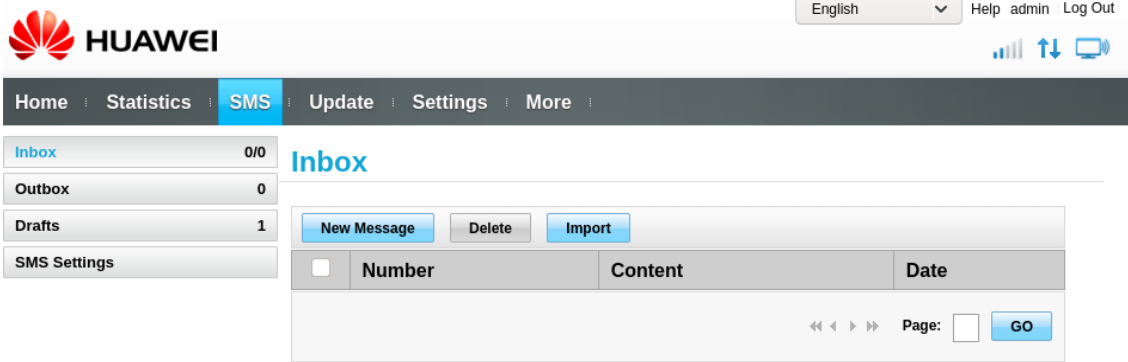


Figure 13. SMS page of the Huawei B315's web interface

We notice that there is an inbox where it is possible to read messages that have been received and sent. An ID representing the number of the message, the content and a timestamp representing the date can be seen. As we said, we went for a default installation of the Huawei Box with a SIM card that only allows Internet access and not SMS feature enabled that is why we are not digging deeper in that section.

Findings using the manual extraction methods

Table 3. Manual extraction results of Huawei B315

Artifacts	Credentials required
ISP	<input checked="" type="checkbox"/>
Connected users – IP, host names, MAC	<input checked="" type="checkbox"/>
SSID	<input checked="" type="checkbox"/>
NTP settings	<input checked="" type="checkbox"/>
System logs	<input checked="" type="checkbox"/>
Device information – IMEI, IMSI, firmware	<input checked="" type="checkbox"/>
SMS	<input checked="" type="checkbox"/>

As we can see in the Table above, the web interface is interesting for forensic data gathering. Credentials are not needed to gather the ISP running the service and the up

time of the router. For the rest, the username and password are needed to be known, and this could be a problem if the retrieving of those logins cannot be managed.

In the next chapter we will present, how to automate the collection of information from the web interface using the API provided by Huawei and the differences that have been noticed.

4.1.3 API

An API allows us to make HTTP requests to a server (in our case GET requests) that will (depending on the specific request) give us information that is stored in the server. The call that is made to the server (Huawei router) has been presented in the previous chapter about the web user interface but will be explained in detail in here.

Huawei implemented such an API for the routers they provide. As we saw on the Wireshark analysis in Figure 6, the server (router) gives us a response to our request that contains information that could be used during a digital forensic investigation. We decided to perform a get request to *routerIP/api/device/information* to explain the result given by the server.

GET /api/device/information

The response of our request can be seen below.

B315s-22 EFY7	86796	2280245	89410225694900050026 WL3B310FM01
21.333.01.00.00 17.100.09.00.03 88:F8:72:		LTE cpe LTE 10.51.205.199	

Figure 14. Result of */api/deivce/information* request from Huawei B315

We can by deduction understand some of the information that is given to us but this is not the case for all the possible API requests. Therefore, we decided to create a python tool based on a wrapper for the API that can be found in the [Appendix 2].

The tool allows us to gather and translate the data given by the server to a more readable language. As we can see below the request made just before would be translated to:

```
[+] Device Information [+]
Device Name : B315s-22
Software Version : 21.333.01.00.00
Hardware Version : W
Mac Address : 8
Serial Number : E
```

Figure 15. Device Information result from python script

We decided to remove data when running our tool that would not be useful during a forensic analysis. The tool can be seen as automation to the manual extraction method. The script allows us by simply providing the credentials to the administration panel of the router to retrieve all the information that was presented in the Web UI part.

```
(Huawei-MasterThesis) [kevin@kevin src]$ python huawei.py -i 192.168.8.1 -u admin -p password123
#####
# Program created to gather information about your Huawei router #
# for a forensic analysis. #
# Use it with caution, every call to the API can alter the data #
# #
# Default username/password is admin/admin. #
#####

[+] Internet Service Provider(ISP) [+]
Provider : Sunrise

[+] Device Information [+]
Device Name : B315s-22
Software Version : 21.333.01.00.00
Hardware Version : W
Mac Address : 8
Serial Number : E

[+] Sim Card Information [+]
Sim Card Identity(Iccid): 8
Mobile Identity(Iimei): 8
Mobile subscriber(Imsi): 2

[+] Current Language of router [+]
Language: en-us

[+] List of connected devices [+]
[+] Device number 1 [+]
Hostname: kevin
IP address: 192.168.8.100
Mac address: 44

[+] NTP Information (Time) [+]
Date: 18 February 2020
Time: 03:44:45
Timezone: GMT-12:00 - International Date Line West

[+] Syslog Information [+]
2020-02-18 03:44:45 Informative Platform Succeeded Log in
2020-02-18 03:42:47 Informative Platform Succeeded Log in
2020-02-18 03:31:37 Informative Platform Succeeded Log in
2020-02-18 03:31:10 Informative Platform Succeeded Log in
2020-02-18 03:28:54 Informative Platform Succeeded Log in
2020-02-18 03:26:56 Informative Platform Succeeded Log in
```

Figure 16. Script results with credentials – Huawei B315

As we already stated in the web part, some information cannot be accessed without credentials. If we do not provide credentials, we are able to retrieve information about the ISP, the language of the router, NTP information and the web logs. To this

information, we can add the router information, SIM card information and the currently connected devices if the username and password are provided. A table of the differences between the automated and manual extraction method can be found in the next chapter.

Findings using the script automating the manual process

The tool is interesting to use to automate the process of information gathering instead of going for a manual approach to collect the data of value. The difference when we use the python script is that it allows us to retrieve the SIM card information that could then be used for further investigation. Forensic analysis of the SIM card is a method that can be used but is out of scope in our own research as we are looking for the data in the router itself. But the analysis of the SIM card should not be forgotten as it can contain evidentiary data or can help with the progress of the investigation. Interesting papers where researchers share their findings in that area can be found in [23], [24] and [25].

Some restrictions where we needed credentials in the web interface were not necessary when using the API instead. It's a sort of authentication bypass. Below you will find the different artifacts that are gathered using our tool.

Table 4. Automated extraction results of Huawei B315

Artifacts	Credentials required
ISP	<input type="checkbox"/>
Connected users – IP, host names, MAC	<input checked="" type="checkbox"/>
Sim Card information – Sim card Identity (ICCID), IMEI, IMSI	<input checked="" type="checkbox"/>
NTP settings	<input type="checkbox"/>
System logs	<input type="checkbox"/>
Device information – Firmware, MAC, Serial Number	<input checked="" type="checkbox"/>
Language in use on the router	<input type="checkbox"/>

4.1.4 Hardware-based artifacts extraction

We were able to retrieve some forensic artifacts using manual extraction method. As we want to retrieve as much evidentiary data as possible, we are going to perform a hardware-based data extraction. Due to the lack of installed services on the Huawei B315, we had to skip the logical extraction method. Before being able to achieve the hardware-based extraction, we need connectivity between the router and our forensic workstation that is why we are going to analyse the motherboard of the router and then create the connectivity.

UART

We are not going into the details of how that interface works, we are just going to present the important points needed to create the connectivity between the router and our forensic workstation. The UART is (as we explained in the beginning of our research) used to test the design of the product after manufacture. That interface is still available after purchase, meaning that everyone can access it and most of the time root access is granted to the connected user. An explanation on how the interface is designed technically has been provided by U. Nanda and S. Pattnaik in their research paper [26]. As more detailed hardware-based extraction is out of scope in our research, we are not going into the details of that research but it was worth mentioning it. We decided to go with a less technical explanation allowing us to create that connectivity between the devices.

A book from A. Guzman and A. Gupta [27] allows us to understand, what we need to find on the board to create the connection. In [27, Ch.1, p.22], they present the pins that need to be found on the board. Those pins are TX (transmit data), RX (receive data) and GND (ground). In order to exchange data with the board, we need to connect the pins to the serial interface connector. This is done by soldering the two device's pins together. This process is going to be presented below.

After many online researches, we came across a PDF file [28] containing slides from the CHCon 2018 conference concerning "Router Hacking". One of the slides contained

information about the motherboard of the exact same model as the one we are trying to analyse (Huawei B315).

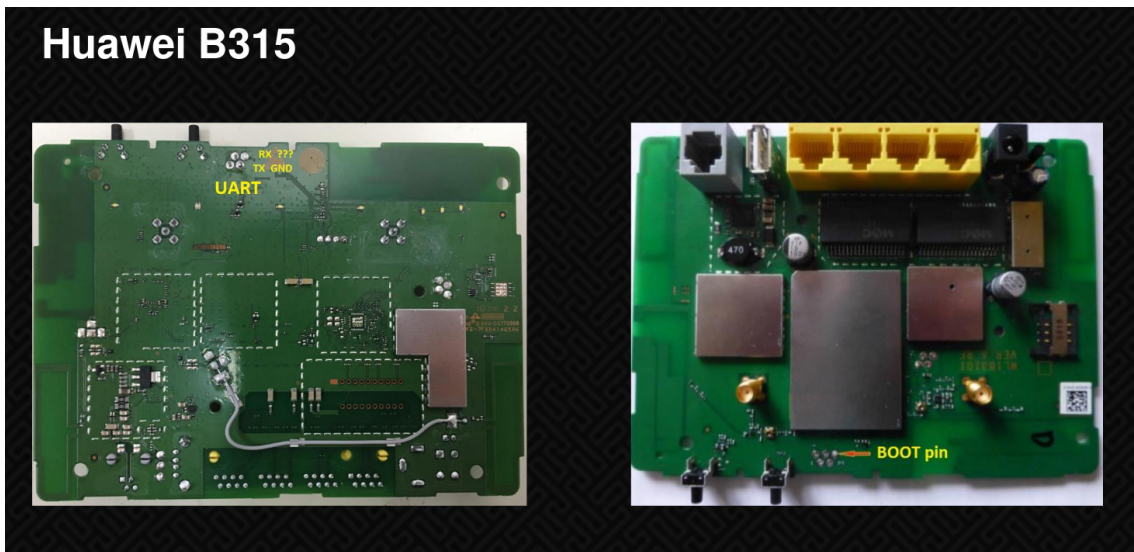


Figure 17. Huawei B315 UART interface finding from Ben Zante [28]

We can see on the left of Figure 17, the location of the pins from UART interface needed to create the connectivity. The location of those three pins (RX, TX and GND) is the most important point to perform our hardware-based extraction. As we told earlier, we need a second device acting as connector to the interface. This is where we decided to go with a USB to TTL Serial cable (as it can be seen in Figure 18) to be able to connect it to the computer (forensic workstation).



Figure 18. USB to TTL Serial cable

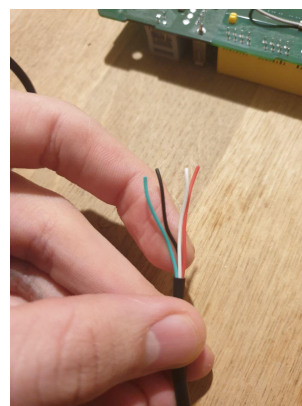


Figure 19. USB to TTL Serial cable - Wires

The white wire from the USB to TTL Serial cable will be soldered to the RX pin, the green one to the TX and the black one to the GND. RX and TX from the routers pins are inverted with the RX and TX from the USB cable wires for the communication to work. The red one is used for the VCC but it is strictly discouraged to connect to that one due to the possible disruption of the hardware. The final result should look similar to Figure 20.

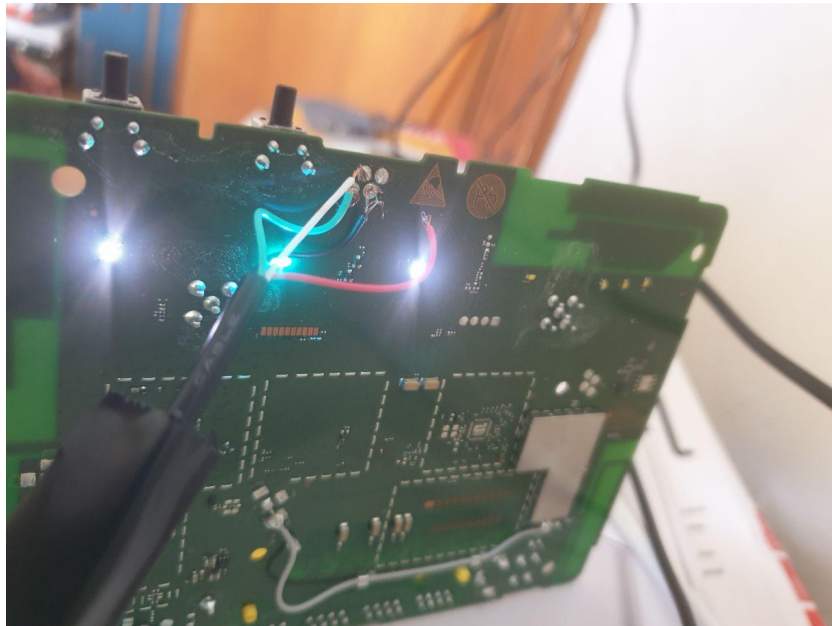


Figure 20. USB to TTL Serial cable connected to the motherboard

Once everything is setup, we can plug the USB to the PC and if the drivers are installed (should be installed by default) the motherboard of the router will be recognized by the host.

```
[kevin@kevin ~]$ udevadm monitor
monitor will print the received events for:
UDEV - the event which udev sends out after rule processing
KERNEL - the kernel uevent

KERNEL[76332.325990] add      /devices/pci0000:00/0000:00:14.0/usb1/1-4 (usb)
KERNEL[76332.327550] add      /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0 (usb)
KERNEL[76332.328899] add      /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0/ttyUSB3 (usb-serial)
KERNEL[76332.329244] add      /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0/ttyUSB3/tty/ttyUSB3 (tty)
KERNEL[76332.329265] bind    /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0/ttyUSB3 (usb-serial)
KERNEL[76332.329285] bind    /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0 (usb)
KERNEL[76332.329310] bind    /devices/pci0000:00/0000:00:14.0/usb1/1-4 (usb)
UDEV [76332.885915] add      /devices/pci0000:00/0000:00:14.0/usb1/1-4 (usb)
UDEV [76332.886960] add      /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0 (usb)
UDEV [76332.887472] add      /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0/ttyUSB3 (usb-serial)
UDEV [76332.889125] add      /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0/ttyUSB3/tty/ttyUSB3 (tty)
UDEV [76332.891568] bind    /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0/ttyUSB3 (usb-serial)
UDEV [76332.892318] bind    /devices/pci0000:00/0000:00:14.0/usb1/1-4/1-4:1.0 (usb)
UDEV [76332.896215] bind    /devices/pci0000:00/0000:00:14.0/usb1/1-4 (usb)
```

Figure 21. udevadm monitor - result when plugging the USB to host

Udevadm monitor is an integrated linux tool allowing us to listen to kernel uevents and events. One of those events being a device being attached to the host. In Figure 21 above, we can see that when we plugged the USB cable into our working environment, it was recognized as /ttyUSB3 (usb-serial) in the file system. It means that connectivity has been established.

```
[24853.830551] usb 1-4: new full-speed USB device number 23 using xhci_hcd
[24853.973216] usb 1-4: New USB device found, idVendor=10c4, idProduct=ea60, bcdDevice= 1.00
[24853.973223] usb 1-4: New USB device strings: Mfr=1, Product=2, SerialNumber=3
[24853.973227] usb 1-4: Product: CP2102 USB to UART Bridge Controller
[24853.973231] usb 1-4: Manufacturer: Silicon Labs
[24853.973234] usb 1-4: SerialNumber: 0001
[kevin@kevin ~]$
```

Figure 22. CP2102 USB to UART Bride Controller

We can see that the device is recognized and can be used for a direct access by using the “/dev/ttyUSB3” path in the Linux file system with a tool like Minicom for serial communication. Before being able to communicate with the router, the baud rate needs to be known (for Minicom or any other tool) but a simple google search allowed us to find that number. The default baud rate for Huawei B315 is 115200 bits/s. We can now proceed with the hardware-based extraction method.

4.1.5 Forensic artifacts extraction

We decided to see what would happen, if we boot the router with the Minicom connected to “/dev/ttyUSB3”. Some researches like [29] and [30] show us, how they were able to use the debugging interface to get into a shell and perform basic commands like changing directory, listing files or showing the content of a file. Being able to spawn a shell on the router would be interesting in a forensic point of view to access the volatile memory or other restricted area that we could not access before.

```

onchip
NF_id boot!
NF ID 0x98AC9026 0x76160800
NF pagesz 0x00001000B,pagenm 0x00000040,oobsz 0x00000000B,ecc 0x00000000B,addrnm 0x00000005,chipsz 0x00000200MB
Nand save 0x8A800132
UnSec_boot^Wp123
page_sz 0x00001000
page_nm 0x0
cp from 0xA0044
^Z
[0000009ms]
[0000009ms]
[0000009ms]*****
[0000009ms]FASTBOOT simple console, enter 'help' for commands help.
[0000009ms]*****
[000000Ams]balong_version_get_hw_version doesn't judge udp!
[000000Ams][bsp save hw version]:get hw version ok, hw_main=0x00002008!
[000000Bms]Hisilicon NANDC_V6.00 initialize...^M
[000000Bms]nandc_get_paramer_from_onchiprom
[000000Bms]NAND_ARGS_FULL_AVAIL
[000000Bms]nandc_native_get_spec pagesize:0x00001000, blocksize:0x00040000, chipsize:0x20000000, sparesize:0x00000090, ecctypes:0x00000006
[000000Cms]pagesize 4096 ecctype 6 addrnm 5 blocksize 0x00040000 sparesize 144 chipsize 0x20000000
[000000Dms]fastboot: nv dload cap is 0x00400000.
[000000Fms]fastboot: dload nv invlv_blk_num:3, total_blk_num:16!
[000000Fms]fastboot: dload nv skip total bad blk:0!
[000000Fms]warning: end page size not aligned :addr_logic:0x00af2000, blockleft:0x000007be
[0000019ms]nv boot init ok!
[0000019ms][tsensor]: tsensor init ok!
[0000019ms]board_init ok
[000001Bms]USB FastBoot: V0.9
[000001Bms]Machine ID: 3339 v0
[000001Bms]Serial Number: UNKNOWN
[000001Bms]
[000001Bms]Heap:0x5fd31fe0 -- 0x5fd34640, 9824
[000001Bms]Please distribute uart with command L/V/M...
[000001Cms] heap:0x5fd31fe0 -- 0x5fd34640, 9824
[000001Cms]^M
[000001Cms] [ ON OFF ] Start up by Warm Reset!,reboot_cmd=0x52454348.^M

```

Figure 23. Boot result from UART interface – Huawei B315

The result of the communication with the UART interface while booting the Huawei B315 can be seen in Figure 23. That booting process presents us with an interesting line. “FASTBOOT simple console, enter ‘help’ for commands help.” Unfortunately, we were not able to stop or input commands during the booting of the router. After a discussion with Ben Zante [28], he told us that he was able to obtain a root shell by exploiting a security vulnerability on the Samba (SMB) service that has been patched since. That service is disabled by default; therefore we did not look at it during the logical extraction method as the goal of this study is to work on factory settings router.

We decided to stop with the hardware-based extraction as finding security vulnerability would be out of scope of our own research and that process could be time consuming. Such an approach would alter the data contained in the volatile memory that we are trying to access but should be considered if everything else fails.

We can add that no hardware specification could be found using that UART interface unfortunately. The only point that would be worth mentioning is the fact that the router is running a Linux OS as we can see on Figure 24.

```
[00001Fms]boot linux from flash
[00001Fms]^M
[000020ms][ ON OFF ] Load kernel form kernel!
[000020ms]warning: end page size not aligned :addr_logic:0x01201000,          blockleft:0x00000080
[000021ms]warning: end page size not aligned :addr_logic:0x01613000,          blockleft:0x00000080
[000046ms]warning: end page size not aligned :addr_logic:0x01631000,          blockleft:0x00000080
[000048ms]
[000048ms]kernel @ 55e10000 (4266024 bytes)
[000048ms]ramdisk @ 56e08000 (122262 bytes)
[000048ms]
[000048ms]
[000048ms]hdr->cmdline = 'root=/dev/ram0 rw console=ttyAMA0,115200 console=uw tty0,115200 rdinit=/init loglevel=5 mem=0x9200000'
[000049ms][fastboot]: power_down_charge_flag 0.
[000049ms][fastboot]: soft_version_flag 0.
[00004Ams][fastboot]: abnormal_reset_flag 0.
[00004Ams][fastboot]: get_pw_on_mode 1.
[00004Ams]cmdline = 'root=/dev/ram0 rw console=ttyAMA0,115200 console=uw tty0,115200 rdinit=/init loglevel=5 mem=0x9200000 boardid=0x00000000,0x0000000,0x00000000 pd_charge=0 androidboot.swtype=normal reset_type=normal power_mode=normal '
[00004Bms]
[00004Bms]Booting Linux
```

Figure 24. Linux boot - Huawei B315

We noticed that the interface is only used to monitor the hardware events. An example would be the led lights of the router that are turning on and off as it can be seen below.

```
[led][led_light_state]LED_STATE_ON brightness running!
[led][led_light_state]LED_STATE_ON brightness running!
[led][led_dev_ctrl]led_dev_ctrl the lamp ID 10 color is cyan
[led][led_light_state]LED_STATE_ON brightness running!
[led][led_light_state]LED_STATE_ON brightness running!
[led][led_dev_ctrl]led_dev_ctrl the lamp ID 6 color is white
[led][led_light_state]LED_STATE_ON brightness running!
[led][led_dev_ctrl]led_dev_ctrl the lamp ID 7 color is white
[led][led_light_state]LED_STATE_ON brightness running!
[led][led_dev_ctrl]led_dev_ctrl the lamp ID 8 color is white
[led][led_light_state]LED_STATE_ON brightness running!
```

Figure 25. LED states changes – Huawei B315

An hardware-based extraction of artifacts is not possible on the Huawei B315 with our current lab environment.

4.2 TP-Link TL-MR6400

4.2.1 Router specifications



Figure 26: TP-Link TL-MR6400 - frontside



Figure 27. TP-Link TL-MR6400 - backside

Table 5. TP-Link TL-MR6400 Specifications

General Information	
Manufacturer	TP-LINK
Manufacturer no.	TL-MR6400
Router Properties	
Router type	LTE / 4G router
Mobile Internet Properties	
Sim card type	micro-SIM
Mobile phone standard	3G - UMTS, 4G – LTE, GSM/2G
Max. data transfer rate	150 Mbit/s
Operating system compatibility	Mac, Linux, Android, iOS, Windows
Wi-Fi standard	
Wi-Fi standard	Wi-Fi 1 / 802.11b, Wi-Fi 3 / 802.11g, Wi-Fi 4 / 802.11n
2.4 GHz transmission rate	300 Mbits/s

Wi-Fi encryption	WPA-PSK, WPA2-PSK, WEP
Network Interface	
RJ45 LAN	3x
RJ11 Phone	1x
External antenna port	Yes

As for the Huawei B315, those specifications¹ are not of a forensically interest. We are directly going to present our findings in the following chapters.

4.2.2 Manual forensic artifacts extractions

Before starting any investigation, we need to know the different services running on the router to see the possible manual methods that could be used.

```

TPLinkTL-MR6400 : bash — Konsole
File Edit View Bookmarks Settings Help
[kevin@kevin TPLinkTL-MR6400]$ nmap -p- 192.168.1.1
Starting Nmap 7.80 ( https://nmap.org ) at 2020-03-23 11:32 CET
Nmap scan report for 192.168.1.1
Host is up (0.033s latency).
Not shown: 65529 closed ports
PORT      STATE SERVICE
22/tcp    open  ssh
23/tcp    open  telnet
53/tcp    open  domain
80/tcp    open  http
1900/tcp  open  upnp
7547/tcp  open  cwmpp

Nmap done: 1 IP address (1 host up) scanned in 9.62 seconds
[kevin@kevin TPLinkTL-MR6400]$

```

Figure 28. NMAP scan on TP-Link TL-MR6400

As we can see on the nmap scan result, there is already more applications running than on the Huawei B315. We are going to go through each of those services and see, what possible forensic artifacts could be gathered in the following chapters.

1 Specifications retrieved from : <https://www.digitec.ch/en/s1/product/tp-link-tl-mr6400-routers-6078813>

Web User Interface

As for the previous router, there is a web interface that can be accessed using its IP on a web browser for the manual data acquisition. We are going to present some of the pages and discuss the interesting artifacts that could be gathered using that method.

Index

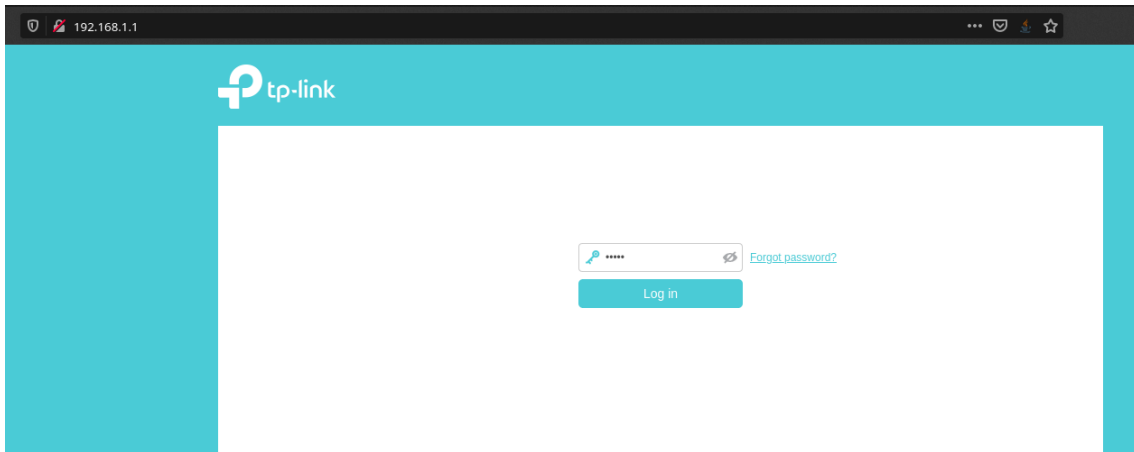


Figure 29: Index page of the TP-Link TL-MR6400

We can see in Figure 29 that the difference between the Huawei and the TP-Link is that this one stores the username for us and only asks for a password. The username being hardcoded during the request of authentication as it can be seen in the code below. That code has been retrieved while analysing the source code of the web page.

```
1 var loginSubmit = function(o) {
2     if (l != isLocked) {
3         var t, i = $("#pc-login-password"),
4             r = $("#pc-login-user"),
5             l = $("#pc-login-password").tpInput("val");
6         !0 === o && (i = $("#ph-login-password"), r = $("#ph-login-user"), l = i.val());
7         var s = r.val();
8         INCLUDE_LOGIN_USERNAME && "" == s ? inputError(r, $.tpLang.login.tipsText, o) :
9         " != l ? (t = rsaEncrypt($.Base64Encoding(l), n, e), doLogin(INCLUDE_LOGIN_USERNAME ? rsaEncrypt(
10            s, n, e) : rsaEncrypt("admin", n, e), t, l, o, o ? $("#ph-login-btn") : $("#pc-login-btn"))) : in
            putError(i, $.tpLang.login.tipsText, o)
11     }
12 }
```

Figure 30. Login function – TP-Link TL-MR6400

Unfortunately, if the password is not known (changed from default), it cannot be guessed or brute-forced easily due to a 10 minute lockout after 5 wrong tries. As we are working on routers with factory settings, the access will not be a barrier for us (default

password is admin). Once logged in, we are able to search for forensically valuable data on the different pages that are presented below.

Network topology

We are able to retrieve the following artifacts on the network topology page: the host names of the current connected devices, as well as their associated IP address and MAC address. This can be seen in the following Figure.

Wireless Clients

ID	Name	IP Address	MAC Address
1	Galaxy-S9	192.168.1.101	52-F9-A3-3F-25-D9
2	kevin	192.168.1.102	18-5E-0F-82-2D-28

Figure 31. Currently connected devices to the Wi-Fi - TP-Link TL-MR6400

We are able to see the devices connected with an Ethernet cable as TP-Link makes a distinction between those different connection types. It can be seen on Figure 31, that we are presenting the “Wireless Clients”. On the same page, we can gather the DNS servers that are in use, but also the ISP providing the internet connection as presented in Figure 32.

Internet

WAN Interface Name: LTE

Internet Status: Connected

IP Address: 10.52.59.66

DNS Server: 10.200.102.243 10.200.102.244

3G/4G

SIM Card Status: SIM card prepared.

Signal Strength: 25%

ISP: Sunrise

Network Type: 4G LTE

Figure 32. DNS and ISP artifacts - TP-Link TL-MR6400

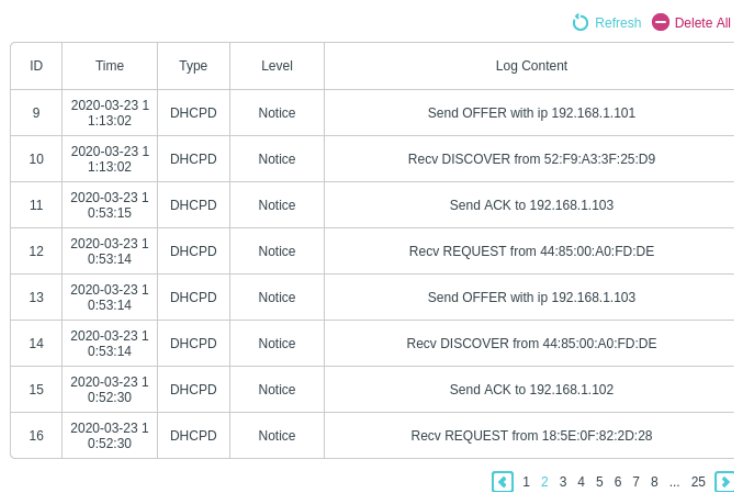
Wireless Settings

In this page, we can retrieve the SSID of the network but also the password that is in use. If the default password is in use it can be found on a sticker at the back of the router.

System logs

On the TL-MR6400, we can find system logs generated by the router and stored in its volatile memory. It is stored in the volatile memory because a power off or reboot will delete the stored logs. We can state this because we tried it in our own lab environment. The investigators need to be sure that everything has been gathered on the scene before plugging off the power supply of the router. Those logs are categorized by type and level. The most important logs in a forensic point of view are the ones concerning the DHCP service allowing the distribution of IP address of the connecting devices.

This type of logs allows us to see the different devices that tried to connect and that are already connected to the router. An interesting research [31] from Tobias Fiebig presents us with the analysis of similar DHCP artifacts using a physical (hardware) extraction method with help of the JTAG interface. A technique that we are going to present in the chapter concerning the hardware-based acquisition. Back at the manual extraction method, as we said, we are able to view those logs as shown in the Figure below making it an important artifact.



ID	Time	Type	Level	Log Content
9	2020-03-23 11:13:02	DHCPD	Notice	Send OFFER with ip 192.168.1.101
10	2020-03-23 11:13:02	DHCPD	Notice	Recv DISCOVER from 52:F9:A3:3F:25:D9
11	2020-03-23 10:53:15	DHCPD	Notice	Send ACK to 192.168.1.103
12	2020-03-23 10:53:14	DHCPD	Notice	Recv REQUEST from 44:85:00:A0:FD:DE
13	2020-03-23 10:53:14	DHCPD	Notice	Send OFFER with ip 192.168.1.103
14	2020-03-23 10:53:14	DHCPD	Notice	Recv DISCOVER from 44:85:00:A0:FD:DE
15	2020-03-23 10:52:30	DHCPD	Notice	Send ACK to 192.168.1.102
16	2020-03-23 10:52:30	DHCPD	Notice	Recv REQUEST from 18:5E:0F:82:2D:28

Figure 33. DHCP system logs - TP-Link TL-MR6400

In these logs, we can see all our three devices of the lab going through the DHCP process to get an IP address and Internet connection from the router.

NTP settings

As we already explained, to be able to recreate a correct timeline of occurred events on the router, the NTP settings need to be known. These settings will allow us to calculate the difference of time between the current time zone and the one used in the router if there is any.

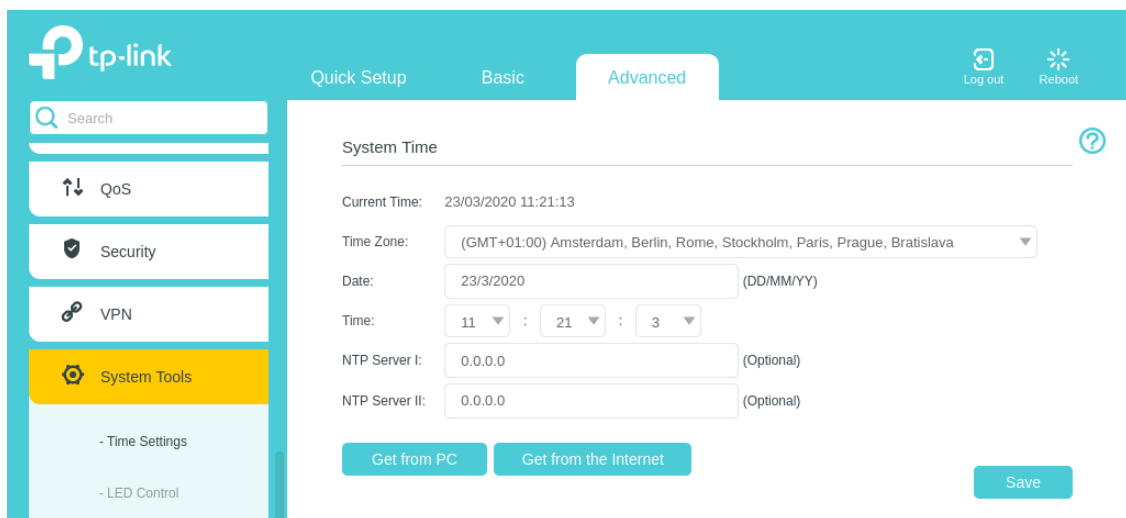


Figure 34. NTP settings - TP-Link TL-MR6400

SMS page

As for most 4G routers, the possibility to send and receive SMS is also a feature, which allows us to retrieve the data of interest. We already stated in the beginning that our lab environment does not permit us to experiment this sort of feature but it is important to mention it. Figure 35 presents us with the possible artifacts that could be gathered using that feature.

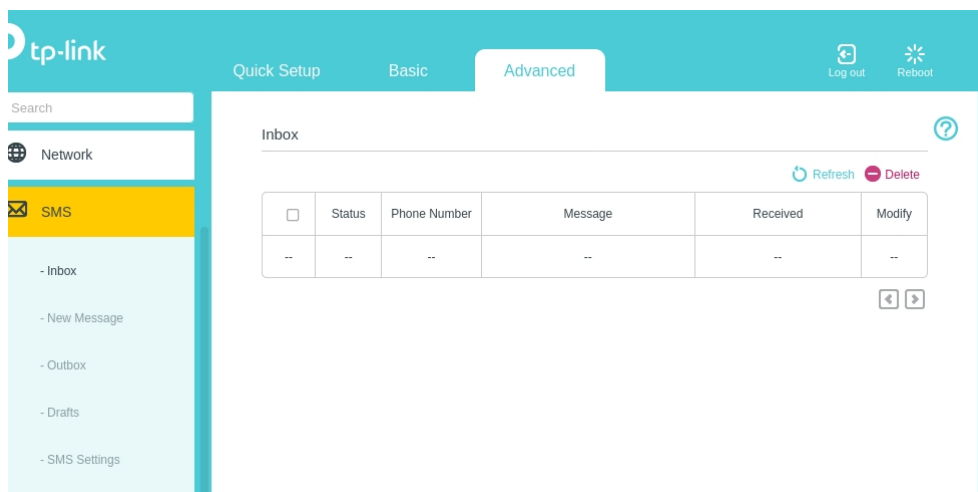


Figure 35. SMS settings page - TP-Link TL-MR6400

Backup file

An interesting feature of the TP-Link TL-MR6400 is the fact that we are able to create a backup file containing all the configuration settings. This file called “conf.bin” by default can be stored on a PC or an external device, and represents itself a file that should be looked for during an investigation as we will present below. The file can be generated by anyone having access to the web interface as there is an option that allows us to create a backup file.

That file, when created, is encrypted and cannot be read by simply opening it. Luckily, there exists a tool called RouterPassView¹ on Windows allowing us to decrypt that configuration file and allowing us to bypass the need to authenticate to the web interface (useful if password is not known and backup exists). The tool is able to decrypt the file with ease due to the weak encryption used. Once decoded, the file shows us the configuration used on the router in an Extensible Markup Language (XML) format with some artifacts that can be used during an investigation. Following Figures represent us some of the findings that were gathered from the decrypted file.

1 Tool can be found at : https://www.nirsoft.net/utills/router_password_recovery.html

```

RouterPassView - C:\Users\odin\Desktop\conf.bin
File Edit View Options Help
<?xml version="1.0"?>
<DslCpeConfig>
  <InternetGatewayDevice>
    <DeviceSummary val="InternetGatewayDevice:1.1[(Baseline:1, EthernetLAN:1)" />
    <LANDeviceNumberOfEntries val=1 />
    <DeviceInfo>
      <ManufacturerOUI val=CC32E5 />
      <SerialNumber val=CC32E5C6B33A />
      <HardwareVersion val="TL-MR6400 v4 00000002" />
      <SoftwareVersion val="1.10.0 0.9.1 v0001.0 Build 190521 Rel.38696n" />
      <UpTime val=279 />
      <X_TP_IsFD val=3 />
    </DeviceInfo>
  </InternetGatewayDevice>
</DslCpeConfig>

```

Figure 36. Device information from backup file

Device information like the Serial number (MAC address), firmware are available in plain text after decrypting the file with the RouterPassView tool.

```

  <Enable val=1 />
  <DevicePassword val=55915557 />
  <ConfigurationState val=Configured />
</WPS>

```

Figure 37. Network password from backup file

```

<IspName val=Sunrise />

```

Figure 38. ISP information from backup file

Being able to locate that file, if it exists, can be an interesting advance for the investigation.

Findings using the manual extraction methods

Below we can find a Table giving us an overview of all the forensic artifacts that were gathered using the manual approach.

Table 6. Manual extraction results of TP-Link TL-MR6400

Artifacts	Credentials required
Device information – Firmware, MAC, IP	<input checked="" type="checkbox"/>
Connected devices – Host names, IP and MAC addresses	<input checked="" type="checkbox"/>
DNS Server in use	<input checked="" type="checkbox"/>
ISP	<input checked="" type="checkbox"/>
Wireless settings – SSID, password	<input checked="" type="checkbox"/>
System logs (volatile memory)	<input checked="" type="checkbox"/>
NTP settings	<input checked="" type="checkbox"/>
SMS	<input checked="" type="checkbox"/>
Backup file (if exists)	<input type="checkbox"/>

As it can be seen in the table, all the artifacts, except the backup file, need access to the administrator account to be gathered. The backup file does not need any credentials but simply needs to be found on any storage device. As it is not mandatory to create backups, it is possible that the file does not exist but looking for it is worth the time due to the amount of information contained in it.

4.2.3 Logical forensic artifacts extraction

The methodology used during our investigation, presented a logical extraction method where sending commands should return data to us (files). On the Huawei B315, there were no services running allowing us to perform a logical approach. But on this router, other network services were open as it can be seen on the nmap scan result in Figure 28. Those services require authentication and once that process is done, we are able to send commands to retrieve evidentiary data.

SSH

The Secure Shell (SSH) port 22 is open as it was stated by the result from our nmap scan. After some tries to connect to it from our workstation, we saw that we were not allowed to connect to it. A google search [32] allowed us to see that this port is only

used for the TP-Link Tether smartphone application. That application is used to manage the settings of the router using a phone. Bypassing that restriction was not possible and the scope of our research was the analysis of our router that is why we decided to stop our investigation on this service.

Telnet

Connecting to that port, we are prompted back with a password request. This password is the same as the one from the administrator account. Trying to guess or brute-force the telnet service's password will result to the same 10 minutes lockout as for the web service. The administrator account for this service is the same as the one from the web. This password should be tried first. If the password has been changed, cracked using brute force attack or guessed, these are, as we stated, unfortunately not solutions to get into the router. As the scope of our research is factory settings routers, we are able to get in.

Once connected, we are able to send commands to the router and get responses back from it. The first command that can be used to see what is available in this application is "help".

```
TP-Link#help
normal mode commands:
  clear      ---   clear screen
  exit       ---   leave to the previous mode
  help       ---   help info
  history    ---   show history commands
  logout     ---   logout cli model
config mode commands:
  config     ---   enter config mode
  igmp      ---   igmp config
  wlctl     ---   wireless config
  iptv      ---   iptv config
  lan       ---   lan config
  dev       ---   device control
  usb       ---   usb config
TP-Link#
```

Figure 39. Help command result on telnet - TP-Link TL-MR6400

The result of the "help" command shows us other available commands. We can see that they will allow us to modify and view the configuration of the router. An example where we retrieve the password of the network can be seen below.

```
TP-Link#wlctl show 5g
[ util_judgeProductName ] 661: Device is MR6400V4

[ util_judgeProductName ] 661: Device is MR6400V4

Status=Up
SSID=TP-Link_B33A
bandWidth=Auto
autoChannelEnable=1
channel=2
QSS=Enable
SecMode=PSK WPA2-AES
Key=55915557
cmd:SUCC
TP-Link#
```

Figure 40. SSID and password retrieval using telnet - TP-Link TL-MR6400

During our own research concerning the telnet service, we came across a paper [33] from K. V. Erkelens and E. Wit. In their research they are investigating the forensic analysis of the Netgear WNDR3700. One of their extraction methods concerning the volatile memory is to access a hidden root shell in the telnet daemon. This shell is only available by sending a specially crafted packet to the telnet service before logging into it [34]. Once this is done, they get full control of the router and are then able to copy the memory to their forensic workstation using the dd (copy file) and nc (netcat) tools of the router [34].

After plenty of researches we did not come across such an hidden access to the specific model we are currently investigating. We decided to abandon the idea of accessing the volatile memory using the telnet service.

Universal Plug and Play (UpnP)

We were able to retrieve the Linux version of the router using the Python-based application called Miranda¹ that allows us to interact with the UpnP protocol. As we can see in Figure 41, the router is running a Linux/2.6.36 version.

1 Tool can be found at : <https://tools.kali.org/information-gathering/miranda>


```
upnp> msearch
Entering discovery mode for 'upnp:rootdevice', Ctl+C to stop...
*****
SSDP notification message from 192.168.1.1:1900
XML file is located at http://192.168.1.1:1900/gatedesc.xml
Device is running Linux/2.6.36, UPnP/1.0, Portable SDK for UPnP devices/1.6.19
*****
```

Figure 41. Linux version gathered using UpnP - TP-Link TL-MR6400

The XML file that is showing up in the results of our tool is not worth presenting as it does not contain any valuable data for a digital forensic investigation.

4.2.4 Hardware-based artifacts extraction

As for the Huawei B315, we wanted to have access to the file system of the router by getting access to a privileged user like root. For that, we had to search for a debugging interface on the motherboard allowing us internal access to the system. After some researches, we were not able to find any documentation about the board that we had in our hands.

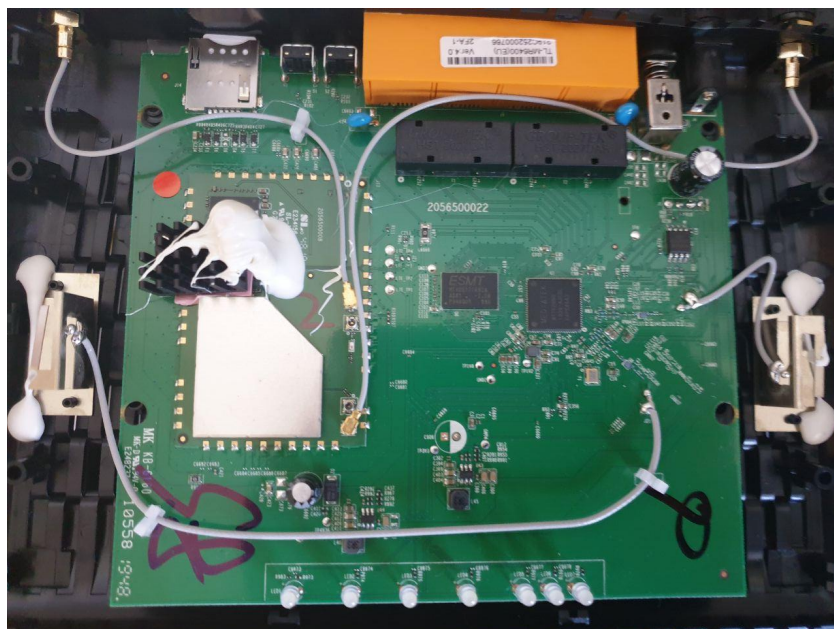


Figure 42. TP-Link TL-MR6400 v4 motherboard

Because of our incapacity to find anything online regarding the debugging interface, we had to opt for a more advanced approach. The first thing we had to do was to locate a number of 4 to 6 pins on the board that could possibly be used for that purpose.

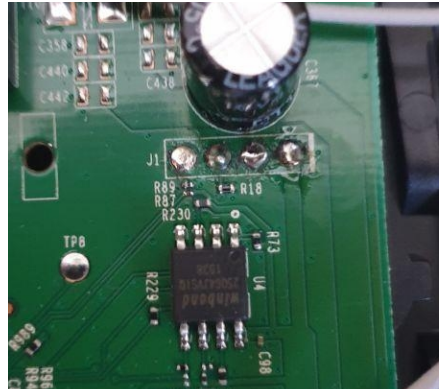


Figure 43. Possible debugging interface - TP-Link TL-MR6400

As it can be seen in the Figure above, we were able to locate 4 pins grouped together that are probably connected to the chip under them. An online research¹ of the chip name's redirected us to the properties of the chip and the configuration of each pin. In this configuration we found that they were talking about GND (Ground), VCC (Power supply) and DI (Data Input) and DO (Data Output). These terms being similar to the one from the UART interface (RX, TX, GND and VCC); we were confident that it was a UART serial access again.

We based then our next steps on an online article [35] written by I. Sindermann where he explains on how to abuse the UART interface. As we located some interesting pins, we had now to check the voltage of each pin. This is done by using a multimeter in continuity mode. The lab now looks similar to Figure 44.

1 Properties of the chip retrieved at : <https://www.winbond.com/resource-files/w25q64jv%20revj%2003272018%20plus.pdf> – Chapter 3.1 to 3.3



Figure 44. Lab environment performing voltage identification

Checking the voltage was done by connecting the black probe of the multimeter to a known ground surface. We decided to put it on the metallic part of the motherboard on the right as it can be recognized on the lab environment. Then we simply used the red probe to retrieve the voltage of each pin. We got the following results:

Table 7. Pins voltage result - TP-Link TL-MR6400

Pin 1	3.00v
Pin 2	0.00v
Pin 3	0.00v
Pin 4	3.00v

We numbered the pins from left to right based on Figure 43 to make it easier for us. The results seemed strange for a UART interface based on the research [35] we saw earlier. Anyway, we decided to try and solder our USB to TTL serial cable to them for testing purpose. After some soldering/unsoldering we were not able to make it recognized as device in the forensic workstation. Of course, we went blindly and did not read carefully the specifications of the chip. On the footnote it is written that “DI and DO are used for Standard and Dual SPI instructions” meaning that the protocol in use for communication is the Serial Peripheral Interface (SPI) and not UART.

A demonstration [36] from A. Useche where he shows how the SPI protocol can be used to dump the firmware of the router could be of interest in our case. Unfortunately, our lab environment did not allow us to perform such tests and therefore we had to stop with the hardware-based acquisition method. As a side note, the firmware can hold valuable data for a digital investigation and therefore should not be overlooked during a case.

4.3 Linksys EA6350

4.3.1 Router specifications



Figure 45. Linksys EA6350 - frontside



Figure 46. Linksys EA6350 - backside

General Information	
Manufacturer	Linksys
Manufacturer no.	EA6350-EJ
Router Properties	
Router type	WLAN router
Number of ports	5
Max. port speed	1000 Mbit/s
WLAN frequency band	Dual band
Guest access	Yes
WPS	Yes
Firewall	Yes

Wi-Fi standard	
Wi-Fi standard	Wi-Fi 2 / 802.11a, Wi-Fi 1 / 802.11b, Wi-Fi 3 / 802.11g, Wi-Fi 4 / 802.11n, Wi-Fi 5 / 802.11ac
5 GHz transfer rate	866 Mbits/s
2.4 GHz transmission rate	300 Mbits/s

The specifications¹ show us that there is nothing about the SIM card properties unlike the other two analysed routers. As we already stated, the main difference between the Linksys EA6350 and the two routers that we analysed before is the fact that this one does not use the 4G technology. Instead, we need to connect this router to a modem with an Ethernet cable that has internet access to be able to provide the Wi-Fi capabilities in our lab environment. We wanted to have a non 4G router to be able to compare the results with the others using that specific technology.

4.3.2 Manual forensic artifacts extractions

In our methodology, we always start by identifying the different services that are available to us for further analysis by running a full nmap scan on the router. This time we added the “-A” option to nmap to retrieve the applications and versions in use on the open ports. As we can see in the following Figure, the output of the scan is bigger than usual.

1 Specifications retrieved from : <https://www.digitec.ch/en/s1/product/linksys-ea6350-routers-5343047>

```

[kevin@kevin ~]$ nmap -A -p- 10.210.1.1
Starting Nmap 7.80 ( https://nmap.org ) at 2020-04-09 15:24 CEST
Nmap scan report for Linksys06028.home (10.210.1.1)
Host is up (0.038s latency).
Not shown: 65525 closed ports
PORT      STATE      SERVICE      VERSION
53/tcp    open      domain       dnsmasq 2.78
| dns-nsid:
|_  bind.version: dnsmasq-2.78
80/tcp    open      http         lighttpd 1.4.39
|_ http-server-header: lighttpd/1.4.39
|_ http-title: Linksys Smart Wi-Fi
139/tcp   open      netbios-ssn Samba smbd 3.X - 4.X (workgroup: WORKGROUP)
443/tcp   open      ssl/http     lighttpd 1.4.39
|_ http-server-header: lighttpd/1.4.39
|_ http-title: Linksys Smart Wi-Fi
|_ ssl-cert: Subject: commonName=linksysmartwifi.com/organizationName=Belkin International, Inc./stateOrProvinceName=California/countryName=US
| Subject Alternative Name: DNS:linksysmartwifi.com, DNS:www.linksysmartwifi.com, DNS:myrouter.local, DNS:EA6350.home.linksys.com
| Not valid before: 2018-10-12T18:24:25
| Not valid after:  2028-10-09T18:24:25
|_ ssl-date: TLS randomness does not represent time
445/tcp   open      netbios-ssn Samba smbd 3.0.37-(Optimized by Tuxera Inc, 3015.10.21) (workgroup: WORKGROUP)
10000/tcp open      http         lighttpd 1.4.39
|_ http-server-header: lighttpd/1.4.39
|_ http-title: 403 - Forbidden
10080/tcp open      http         lighttpd 1.4.39
|_ http-server-header: lighttpd/1.4.39
|_ http-title: Linksys Smart Wi-Fi
49152/tcp open      upnp        Portable SDK for UPnP devices 1.6.19 (Linux 3.14.43; UPnP 1.0)
49153/tcp open      upnp        Cisco-Linksys E4200 WAP upnpd (UPnP 1.0)
51000/tcp filtered unknown
Service Info: OS: Linux; CPE: cpe:/o:linux:linux_kernel:3.14.43, cpe:/h:cisco:e4200

Host script results:
|_ clock-skew: mean: 1s, deviation: 2s, median: 0s
|_ nbstat: NetBIOS name: LINKSYS06028, NetBIOS user: <unknown>, NetBIOS MAC: <unknown> (unknown)
|_ smb-os-discovery:
|   OS: Unix (Samba 3.0.37-(Optimized by Tuxera Inc, 3015.10.21))
|   NetBIOS computer name:
|   Workgroup: WORKGROUP\x00
|_ System time: 2020-04-09T13:26:37+00:00
|_ smb-security-mode:
|   account used: guest
|   authentication_level: share (dangerous)
|   challenge_response: supported
|_ message signing: disabled (dangerous, but default)
|_ smb2-time: Protocol negotiation failed (SMB2)

```

Figure 47. NMAP scan result for Linksys EA6350

We can see that there are more ports and applications available to us, than at the last 2 routers. As our research focuses on the retrieval of forensic valuable data in routers, we are only going to present those in the following chapters.

Findings using the manual extraction methods

For the two other routers, we presented the web interfaces with screenshots containing the evidential data. As manual extraction is not a problem, except if the password is not known, we decided for the Linksys just to present a table with the results and some detailed explanation about the differences we were able to identify.

Below we can find a table presenting an overview of all the forensic artifacts that were gathered using the manual approach.

Table 8. Manual extraction results of Linksys EA6350 using port 80 and 443

Artifacts	Credentials required
Web User Interfaces – port 80 and 443	
Device information – Firmware, MAC, IP	<input checked="" type="checkbox"/>
Devices that connected to the network – Host names, IP and MAC addresses	<input checked="" type="checkbox"/>
Wireless settings – SSID, password	<input checked="" type="checkbox"/>
NTP settings	<input checked="" type="checkbox"/>

As we can see in the table, the retrieving of forensic artifacts on the web user interface is not as interesting, as for the Huawei and TP-Link due to the little information available. For example, there are no logs available when gathering information in the Linksys interface opposed to the other routers. We are not able to retrieve any information without the credentials. That is why we are going to focus more on the logical extraction method in the next chapter.

Some interesting differences that are noticeable are the fact that the devices, which are connecting to the router are “logged”. This means that even if a device disconnects a trace of it can still be gathered.



Figure 48. Connected devices on the EA6350

As it can be seen on the Figure above, we are still able to gather the MAC address of the Galaxy-S9 (our IoT device in the lab environment) after it disconnects from the

network. This artifact being stored in the non-volatile memory can be restored even if the router got disconnected from the power supply.

4.3.3 Logical forensic artifacts extractions

SMB

As we saw in the result of our nmap scan, to detect the current installed application on the router, we have the Samba ports that are enabled by default as opposed to the Huawei B315.

There exists tools like smbclient¹, which allow us enumeration of that service and the retrieval of possible forensic artifacts without having to remove the device storage that is connected to the USB port. By default, there are no security settings, so everyone connected to the network has read/write access to the samba shares without providing any username/password.

```
[kevin@kevin ~]$ smbclient -L 10.210.1.1
Unable to initialize messaging context
smbclient: Can't load /etc/samba/smb.conf - run testparm to debug it
Enter WORKGROUP\kevin's password:

      Sharename      Type      Comment
      -----      -
      Kevin          Disk
      IPC$           IPC       IPC Service (Samba 3.0.37-(Optimized by Tuxera Inc, 3015.10.21))
Reconnecting with SMB1 for workgroup listing.

      Server          Comment
      -----
      LINKSYS06028    Samba 3.0.28a

      Workgroup       Master
      -----
      WORKGROUP      LINKSYS06028
[kevin@kevin ~]$ smbclient \\\10.210.1.1\Kevin
Unable to initialize messaging context
smbclient: Can't load /etc/samba/smb.conf - run testparm to debug it
Enter WORKGROUP\kevin's password:
Try "help" to get a list of possible commands.
smb: \> ls
.
..
$RECYCLE.BIN
.Trash-1000
Autres
[REDACTED]
VM_VIRTUALBOX
488384032 blocks of size 1024. 7403668 blocks available
smb: \>
```

Figure 49. SMB enumeration - Linksys EA6350

1 Tool manual can be found at : <https://www.samba.org/samba/docs/current/man-html/smbclient.1.html>

Firmware analysis

Linksys as stated on their webpage [37], allow users with technical knowledge to customise their router. This is done by flashing a custom firmware that was made by the OpenWrt [38] community, in our case – to add new functionalities to the router.

The OpenWrt website [38] provides us with the needed custom firmware allowing enabling services that were disabled or not accessible by default such as SSH. We did not use this website for the other routers as the modification of the firmware would possibly erase the contained data and possibly brick the device not allowing us to restore those valuable artifacts. Another point why we did not choose this option was due to the mismatching versions of the hardware in use to the previous routers.

As we are doing a research on the different extraction methods, we decided to flash a custom firmware to have access to a file system and see if some data would still be available after flashing. That would be interesting and a possible technique during a digital investigation. We need to add that on the website [38] it can be seen that :

The EA6350v3 is a dual firmware device. ie. There are 2 partitions and Linksys firmware is copied to both partitions at the factory. When you install/update the Linksys firmware, or install/update OpenWrt, the new firmware is always written to the other partition. Upon restarting the EA6350v3, it will subsequently try to boot from the newly installed firmware image from other partition.

As there exists 2 partitions of the same Linksys firmware, we thought that maybe if we replaced one of those partitions with the custom one, we could retrieve valuable data from the Linksys firmware. Following the steps from the OpenWrt website concerning the EA6350 v3 router, we were able to get a root shell access using the custom firmware as it can be seen below.

be seen below, indicates us that there is a UART serial interface that we could possibly access for further analysis.



Figure 51. UART pins from EA6350 router - Retrieved from [38]

As we already did for the Huawei B315, we have to connect to that interface using the same USB to TTL cable as for our last experiment. Once we soldered the wires to the UART pins of the board, we are able to access the router in console. But once again, we are only able to follow the booting process (can also be found in [38]) without the ability to input commands. We decided to stop here with the investigation of our 3rd router.

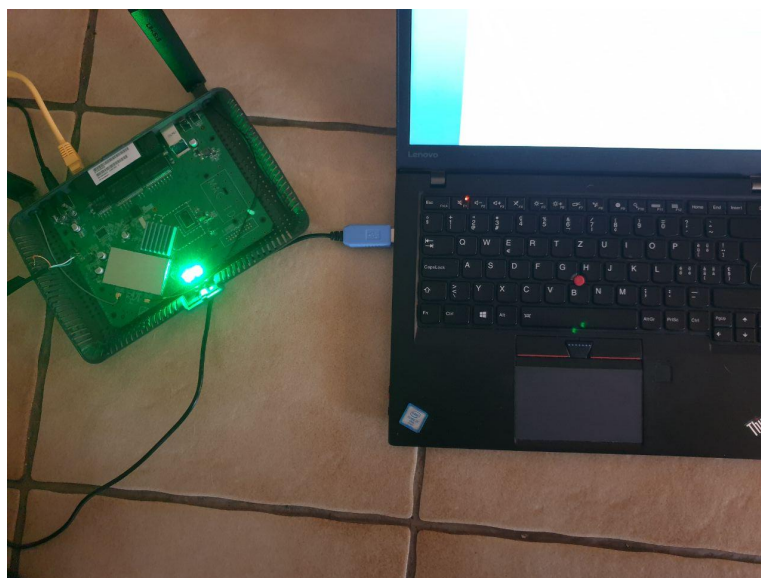


Figure 52. Linksys EA6350 connected to forensic workstation

5 Discussion

During the investigation of our different routers, we went as deep as we possibly could to retrieve some forensic artifacts that could be used during a real-life case. In the methodology used, we presented the possible valuable data contained in a router by first performing a manual approach. That part of the methodology is easily performed as it was presented to us by Broadband Genie in a survey [20] that a little more than 80% of the ordinary Wi-Fi users do not change their passwords when they get a new router. Meaning that our lab environment only contains factory settings routers, facilitating the manual approach makes sense when applied to most of the real-world scenarios.

During that approach, we discovered that the 4G routers in our test cases held more forensic artifacts than the router connected to the modem. For the Huawei B315, we were even able to automate the manual extraction method by creating a python script using the API of Huawei. That script allowed us to gather information about the SIM card that was not accessible on the web interface. Such information could be used to perform a forensic analysis of the SIM card [23]. The tool even allowed us to bypass some web authentication restriction by retrieving the logs of the router that needed credentials, but not when using the API. This makes our tool an interesting automated forensic tool for Huawei routers as the API is also used in other models.

Another interesting test case that we performed during our manual approach was to see if the logs that were contained in the routers (Huawei and TP-Link) would be deleted after powering off the devices. We noticed that the system logs of the Huawei router were not deleted but the one from the TP-Link were not retrievable once the device was shut down. This is an important point to present, as this means that the investigators will need to be on the crime scene to perform the manual extraction and retrieve as much information as possible before powering off the devices.

On the other hand, we have had a lot of trouble for hardware-based extraction due to our IT background being oriented to software engineering and not to hardware. But we were

still able to create connectivity between the motherboards and our forensic workstation when there was a UART interface. That connection allowed us to retrieve the OS from the Huawei router but nothing useful on the Linksys router. Probably being in the hardware field would help to use those gathered information to investigate further. An interesting finding while analysing the 2nd router (TP-Link) was a chip that contains the firmware, as communication in use is the SPI protocol. As our lab environment was not equipped with tools allowing us to perform dumping of firmware we had to abandon that idea. Even once dumped, a strong reverse engineering work needs to be done to retrieve information from the dump and that would most likely fail in our case due to the lack of knowledge in this field.

5.1 Router investigation process

During our investigation, as we already stated, we were able to gather some forensic artifacts by performing an analysis of the router in our lab environment. The entire experimental process of artifacts extraction that we performed during our research cannot be used in real-life scenarios due to the lack of forensic equipment and time at the scene. For achieving a complete forensic analysis of the router, the investigators will need to seize the router on the scene to bring it to their lab environment.

Often, the decision of seizure varies depending on the type of investigation we are working on. As there exist dependencies for the seizure, it is preferable if possible that the router is left behind while all the needed data has been collected directly on the scene. We decided to create a flowchart that can be found in Figure 53 below where we present a base for investigators on how to approach the crime scene and when seizing of the router needs to be done to allow further investigation.

A paper [19, p.44-45] where they propose a possible database of forensic artifacts that can be seen as a point of reference will also be used to compare it to the stages of our flowchart. In their research [19, p.44-45], they are providing a database scheme with a rating system for each forensic artifacts. The higher the rating, the greater the collection of forensic data. That database is interesting and would allow investigators to quickly have a reference on what to focus on while performing the analysis. We decided to

create a process on how to gather those artifacts step by step when entering the crime scene instead of showing the importance as it highly varies depending on the case.

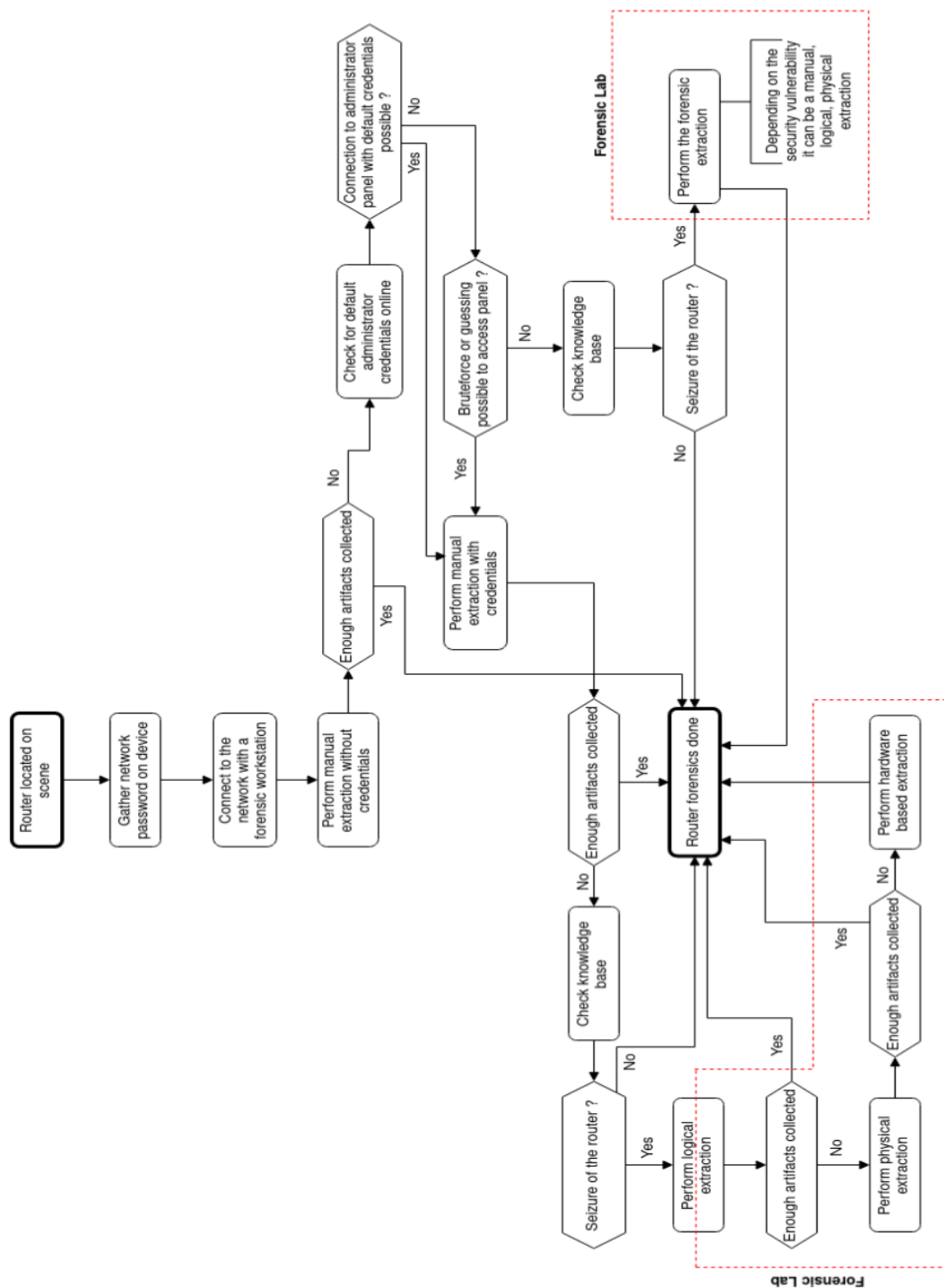


Figure 53. Router investigation process at scene

As we stated before, performing all the steps on the scene is not possible. Seizing the router can also alter some data that is why the manual approach needs to be performed directly at the scene before having to decide to shut down the router for a possible seizure. First of all, we have to locate the router on the scene. Once the location is done, we have to check the device itself as most of the time it has a sticker on it. The sticker contains information such as the SSID and the network password.

In our research on the Huawei router, we were able to bypass the need for credentials for some artifacts using their API. We had access to the system logs and other data without the need to authenticate. This means that it is also interesting to search what is used in the back-end of the router to be able to gather data manually without the need of any administrator password. As this step is not general and depends on certain specific brands we decided not add it in the process. But due to the data that can be collected without any credentials, depending on the type of the case, we decided that it can be enough to simply connect to the network and search for the artifacts manually.

The next step, if not enough artifacts were collected without the administrator panel, is to find those credentials for access to that panel as it is the one containing most of the forensic artifacts. For that, a simple online research is enough, as most of the companies are disclosing the default credentials on their website. It can also be found on dedicated forums if it is not available on the manufacturers website. Once the credentials are known, we have to connect to the router's management panel to start the manual extraction of forensic artifacts. The connection is done by simply connecting to the network and entering the IP address of the router in a browser. It may happen that the passwords have been changed in that case a guessing or bruteforce attack can be performed depending on the protection mechanism in place. As we saw on our routers, there was a lockout of a certain amount of time before being able to reconnect. That would limit the chances to get data by performing a manual approach. That is why a seizure of the router would be needed in that case due to the time constraints on the scene not allowing to perform such tests.

Before the seizure of the device, we added a label "Check knowledge base". This label is more of a proposal for a future work that would be of interest for the investigators where a database with different brands of routers would be in there to see if a seizure

would be the right choice. That database would contain information such as known security vulnerabilities for a certain brand/version allowing more in depth forensic investigation. It is also possible to add the routers that have been analysed in the past to know if a seizure would be of any help in the current case. For now, some research has to be done before seizing the router instead of querying an existing database.

As we stated above, there is a possibility that a router from a certain brand, with a specific firmware version, could have security vulnerabilities allowing access to the internals of the routers such as the file system. Due to the large variety of security issues, depending on the sort of issue there will be a manual, logical or physical forensic extraction on the router. As it really depends on the vulnerability, the method of extraction is not predictable therefore we are going with a “Perform the forensic extraction” label in our process.

As we performed it on our routers, we first searched manually for forensic artifacts. Depending on the case a manual extraction may not be enough. That is why we decided to continue our investigation by performing a logical extraction. As we saw during our investigation, there exist some software like a file-sharing system (SMB) in the router that would allow to gather more data (files) and that could be performed directly on the scene without a seizure. Depending on the software installed on the router, a seizure would be needed that is why we decided to not include entirely the label into the “Forensic Lab” requirement as some investigations can still be done manually at the scene. Once all of the logical extraction has been performed, we can continue if needed with a physical extraction where the main difference will be that instead of having access to some files (logical structure) we will get the entire memory dump bit-by-bit. This can be performed by having access to a shell on the router and then use a tool like dd (on Linux) to copy the entire system.

As the last step, where the seizure of the router needs to be performed due to the time and equipment constraints, we went for an hardware-based extraction by locating a debugging interface (UART in our case) or dumping the firmware. This is the last step if no forensic evidence could be gathered before due to possible damage that can happen to the hardware.

This is the kind of process that with our knowledge due to our research would be used during a real life scenario regardless of the brand of the router. We did not mention that the investigators need to document every action they perform allowing them to differentiate their actions with the one being analysed, as it is more of common sense in this field. It is also important to not disconnect the router from the power supply before the manual extraction has been done as loss of data could happen.

It is also hard to have a strict guideline as every case is different due to the type of crime or the different router brands for example. Research has to be done on the scene nevertheless but we are confident that our process can be used for any kind of router during a crime investigation.

6 Conclusion

Digital forensics is a wide area. That area is composed by many different types and brands of devices that is why we decided to contribute to the field by analysing the routers. For an effective and quick triage on the crime scene, the router is probably one of the central points, on which every other device connects to making it an important part of every digital investigation.

Due to the importance of a router on the scene, a methodology should be used when starting an investigation. With our research, we were able to prove that the methodology [9, p.17] from R. Ayers, S. Brothers and W. Jansen concerning mobile devices could be forwarded into the forensics of routers. A rapid manual extraction will allow the investigators to have crucial information concerning the devices available on the crime scene for further analysis. After a proper and quick manual approach on scene, digging deeper into the router using a logical or hardware-based extraction could allow the analysts to retrieve more artifacts that were not available using the first method. As we already stated, the analysts need a solid background in software engineering and hardware to be able to perform logical and hardware-based extractions.

In our research it could be seen that using a manual approach we were able to retrieve information like network topology (connected devices, IP addresses, MAC, etc.), SIM card information (for the 4G routers), that would allow any analysts to continue their investigations when hitting a dead end with the router itself. Making the manual extraction of artifacts a crucial step when entering a crime scene.

As a conclusion, our research will allow forensic analysts to have a solid understanding of the importance in a manual approach. We are also showing how a logical and hardware-based approach could be performed. Meaning that our work can be used for other types and brands of routers and allowing the analysts to retrieve artifacts, making our a paper a basis for router forensics.

The research showed us different items that can be improved in the future for this work to have a bigger impact. First, there exists an incredible amount of different routers. Extending the number of routers in our paper could be an interesting future work. Second, we were not able to use some techniques related to hardware because of our lab environment. Research in this field would need qualified people who can work in software but also much deeper in the hardware of the concerned router.

References

- [1] "Gartner Identifies Top 10 Strategic IoT Technologies and Trends," *Gartner*, 07-Nov-2018. [Online]. Available: <https://www.gartner.com/en/newsroom/press-releases/2018-11-07-gartner-identifies-top-10-strategic-iot-technologies-and-trends>. [Accessed: 10-Mar-2020].
- [2] P. K. Reddy, *Big data analytics: 5th International Conference, BDA 2017, Hyderabad, India, December 12-15, 2017: proceedings*. Cham, Switzerland: Springer, 2017.
- [3] "Cisco Annual Internet Report - Cisco Annual Internet Report Highlights Tool," *Cisco*, 18-Feb-2020. [Online]. Available: <https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report/air-highlights.html>. [Accessed: 08-Mar-2020].
- [4] D. McClelland and F. Marturana, "A Digital Forensics Triage methodology based on feature manipulation techniques," *2014 IEEE International Conference on Communications Workshops (ICC)*, Sydney, NSW, 2014, pp. 676-681.
- [5] I. Hong, H. Yu, S. Lee, and K. Lee, "A new triage model conforming to the needs of selective search and seizure of electronic evidence," *Digital Investigation*, 14-Feb-2013. [Online]. Available: <https://www.sciencedirect.com/science/article/abs/pii/S1742287613000042>.
- [6] "Volatile Memory," *Volatile Memory Definition*, 18-Oct-2019. [Online]. Available: https://techterms.com/definition/volatile_memory. [Accessed: 20-Dec-2019].
- [7] P. Cappelletti, C. Golla, P. Olivo, and E. Zanoni, *Flash memories*. Boston, MA: Kluwer Acad. Publ., 1999.
- [8] V. S. Harichandran, D. Walnycky, I. Baggili, and F. Breitingner, "CuFA: A more formal definition for digital forensic artifacts," *Digital Investigation*, 07-Aug-2016. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1742287616300366>. [Accessed: 22-Jan-2020].
- [9] R. Ayers, S. Brothers, and W. Jansen, "Guidelines on Mobile Device Forensics - NIST," May-2014. [Online]. Available: <https://doi.org/10.6028/NIST.SP.800-101r1>. [Accessed: 24-Mar-2020].
- [10] P. Szewczyk, "ADSL Router Forensics Part 1: An introduction to a new source of electronic evidence," *Research Online*, 03-Dec-2007. [Online]. Available: <https://doi.org/10.4225/75/57ad5b8e7ff31>. [Accessed: 12-Jan-2020].
- [11] P. Szewczyk, "ADSL Router Forensics Part 2: Acquiring Evidence," *Research Online*, 03-Dec-2009. [Online]. Available: <https://doi.org/10.4225/75/57b28d6240cd4>. [Accessed: 12-Jan-2020].

- [12] G. Horsman, B. Findlay, and T. James, "Developing a 'router examination at scene' standard operating procedure for crime scene investigators in the United Kingdom," *Digital Investigation*, 30-Jan-2019. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1742287618302676>. [Accessed: 19-Feb-2020].
- [13] B. P. Turnbull and J. Slay, "Wi-Fi Network Signals as a Source of Digital Evidence: Wireless Network Forensics," 23-May-2008. [Online]. Available: <https://doi.org/10.1109/ARES.2008.135>. [Accessed: 17-Jan-2020].
- [14] "Introduction to network forensics," *European Union Agency for Cybersecurity(ENISA)*, Aug-2019. [Online]. Available: <https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-material/documents/introduction-to-network-forensics-ex3-toolset.pdf>. [Accessed: 04-Jan-2020].
- [15] P. Feng, Q. Li, P. Zhang, and Z. Chen, "Logical acquisition method based on data migration for Android mobile devices," *Digital Investigation*, 31-May-2018. [Online]. Available: <https://doi.org/10.1016/j.diin.2018.05.003>. [Accessed: 16-Feb-2020].
- [16] A. Levinson, B. Stackpole, and D. Johnson, "Third Party Application Forensics on Apple Mobile Devices," 22-Feb-2011. [Online]. Available: <https://doi.org/10.1109/HICSS.2011.440>. [Accessed: 13-Feb-2020].
- [17] D. Kim and S. Lee, "Study of identifying and managing the potential evidence for effective Android forensics," *Forensic Science International: Digital Investigation*, 29-Jan-2020. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S1742287619301367>. [Accessed: 12-Feb-2020].
- [18] M. Kohn, M. S. Olivier, and J. H. P. Eloff, "Framework for a Digital Forensic Investigation," Jan-2006. [Online]. Available: https://www.researchgate.net/publication/220803284_Framework_for_a_Digital_Forensic_Investigation. [Accessed: 17-Jan-2020].
- [19] D. Blackman and P. Szewczyk, "The challenges of seizing and searching the contents of Wi-Fi devices for the modern investigator," *Research Online*, 2015. [Online]. Available: <https://doi.org/10.4225/75/57b3f385fb887>. [Accessed: 18-Dec-2019].
- [20] M. Powell, "Wi-Fi router security knowledge gap putting devices and private data at risk in UK homes," *Million of UK home broadband routers are insecure | Broadband Genie blog*, 12-Apr-2018. [Online]. Available: <https://www.broadbandgenie.co.uk/blog/20180409-wifi-router-security-survey>. [Accessed: 02-Apr-2020].
- [21] M. Ehmer and F. Khan, "A Comparative Study of White Box, Black Box and Grey Box Testing Techniques," *International Journal of Advanced Computer Science and Applications*, vol. 3, no. 6, 2012.
- [22] *Investigations involving the Internet and computer networks*. Washington, D.C.: U.S. Dept. of Justice, Office of Justice Programs, 2007.

- [23] Warlock, "SIM Card Forensics: An Introduction," *Infosec Resources*, 19-Nov-2013. [Online]. Available: <https://resources.infosecinstitute.com/sim-card-forensics-introduction/>. [Accessed: 07-Mar-2020].
- [24] N. Ibrahim, N. A. Naqbi, F. Iqbal, and O. AlFandi, "SIM Card Forensics: Digital Evidence," *Annual ADFSL Conference on Digital Forensics, Security and Law*. 3., 2016. [Online]. Available: <https://commons.erau.edu/cgi/viewcontent.cgi?article=1367&context=adfs>.
- [25] F. Casadei, A. Savoldi, and P. Gubian, "Forensics and SIM cards: An overview," *International Journal of Digital Evidence (IJDE)*, vol. 5, no. 1, Jan. 2006.
- [26] U. Nanda and S. Pattnaik, "Universal Asynchronous Receiver and Transmitter (UART)," in *2016 3rd International Conference on Advanced Computing and Communication Systems (ICACCS)*, 2016.
- [27] A. Guzman and A. Gupta, *IoT penetration testing cookbook: identify vulnerabilities and secure your smart devices*. Birmingham, UK: Packt Publishing, 2017.
- [28] B. Zante, "Router Hacking CHCon 2018," *insomnia Security*, 19-Nov-2018. [Online]. Available: <https://insomniasec.com/cdn-assets/RouterHacking-CHCon2018.pdf>. [Accessed: 09-Jan-2020].
- [29] S. Golam, "Reverse engineering UART to gain shell," *Medium*, 24-May-2019. [Online]. Available: <https://medium.com/@shubhamgolam10/reverse-engineering-uart-to-gain-shell-de9019ae427a>. [Accessed: 24-Feb-2020].
- [30] I. Sindermann, "Hardware Hacking 101 - Lesson 3: Abusing UART (U Are RooT)," *The Ethical Hacker Network*, 09-Apr-2019. [Online]. Available: <https://www.ethicalhacker.net/columns/sindermann/hardware-hacking-101-lesson-3-abusing-uart-u-are-root/>. [Accessed: 15-Feb-2020].
- [31] T. Fiebig, "Forensic DHCP Information Extraction from Home Routers," 06-Aug-2013. [Online]. Available: <https://www.delaat.net/rp/2012-2013/p94/report.pdf>. [Accessed: 19-Feb-2020].
- [32] "Why SSH TCP port 22 is tested as open?," *TP*, 01-Feb-2020. [Online]. Available: <https://www.tp-link.com/en/support/faq/2462/>. [Accessed: 23-Mar-2020].
- [33] K. V. Erkelens and E. Wit, "Forensic Analysis of Consumer Routers," 01-Jun-2014. [Online]. Available: https://www.os3.nl/_media/2013-2014/courses/ccf/homerouters-esan-kim.pdf. [Accessed: 10-Jan-2020].
- [34] "Unlocking the Netgear Telnet Console," *OpenWrt Project*, 25-Dec-2019. [Online]. Available: <https://openwrt.org/toh/netgear/telnet.console>. [Accessed: 10-Jan-2020].
- [35] I. Sindermann, "Hardware Hacking 101 - Lesson 3: Abusing UART (U Are RooT)," *The Ethical Hacker Network*, 09-Apr-2019. [Online]. Available: <https://www.ethicalhacker.net/columns/sindermann/hardware-hacking-101-lesson-3-abusing-uart-u-are-root/>. [Accessed: 25-Mar-2020].
- [36] A. Useche, "Intro to Hardware Hacking - Dumping your First Firmware," *nVisium*, 07-Aug-2019. [Online]. Available: <https://nvisium.com/blog/2019/08/07/extracting-firmware-from-iot-devices.html>. [Accessed: 01-Mar-2020].

- [37] “Linksys Official Support - OpenWRT firmware information and download links,” *Linksys*, 2020. [Online]. Available: <https://www.linksys.com/us/support-article?articleNum=140719>. [Accessed: 02-Apr-2020].
- [38] “OpenWrt Project : Linksys EA6350 v3,” *OpenWrt Project*, 31-Jan-2020. [Online]. Available: https://openwrt.org/toh/linksys/linksys_ea6350_v3#installation. [Accessed: 28-Mar-2020].
- [39] Z. Liu, Y. Chen, W. Yu and X. Fu, "Generic network forensic data acquisition from household and small business wireless routers," *2010 IEEE International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, Montreal, QC, 2010, pp. 1-6.
- [40] J. Porter, “Detecting malicious behavior in OpenWrt with QEMU tracing,” dissertation, 2019.

Appendix 1 – Login Bruteforcer for Huawei B315

Source code of bruteforce_login.py (python script) that allows to perform a number of specified connection against the router.

bruteforce_login.py

```
from huawei_lte_api.Client import Client
from huawei_lte_api.AuthorizedConnection import AuthorizedConnection
import sys
import datetime

def banner():
    motd = """
    BRUTEFORCE LOGIN
    """
    print(motd)

def main():
    banner()
    currentDT = datetime.datetime.now()
    number_logins = int(input("Number of login attempts: "))
    for x in range(0,number_logins):
        connection = AuthorizedConnection('http://admin:password123@192.168.8.1/')
        client = Client(connection)
        print("Login number " + str(x+1) + " at: " + currentDT.strftime("%I:%M:%S %p"))

if __name__ == "__main__":
    main()
```

Appendix 2 – Huawei B315 information gatherer

Python program to gather forensic artifacts based on the Huawei API. Currently stored in a repository on GitHub that can be found at (with an user guide) : <https://github.com/0x0D1n/Huawei-MasterThesis>

huawei.py

```
from huawei_api import *
from huawei_lte_api.Connection import Connection
from huawei_lte_api.Client import Client
from huawei_lte_api.AuthorizedConnection import AuthorizedConnection
import ipaddress
import sys

def banner():
    motd = """
#####
# Program created to gather information about your Huawei router #
# for a forensic analysis. #
# Use it with caution, every call to the API can alter the data #
# #
# Default username/password is admin/admin. #
#####
"""
    print(motd)

#https://stackoverflow.com/questions/11264005/using-a-regex-to-match-ip-addresses-in-python
def valid_ip(address):
    try:
        ipaddress.ip_address(address)
        return True
    except:
        return False

#Save the data gathered to a file
def writeToFile(filename, content):
    file = open(filename, 'w')
    file.write(content)
    file.close()

#Gather all the data using credentials
def getAllInfoLogin(client):
```

```

data = getISP(client) + "\n"
data += getDeviceInformation(client) + "\n"
data += getLanguage(client) + "\n"
data += getConnectedDevices(client) + "\n"
data += getTimezone(client) + "\n"
data += getLogs(client)
return data

#Gather all the data without credentials
def getAllInfoWithoutLogin(client):
    data = getISP(client) + "\n"
    data += getLanguage(client) + "\n"
    data += getTimezone(client) + "\n"
    data += getLogs(client)
    return data

def main():
    banner()
    ### Command line -helper for args
    import argparse
    parser = argparse.ArgumentParser(description="Huawei B315s-22 4G router information gathering tool")
    parser.add_argument("-i", "--ip", type=str, help="IP address of the router", metavar=("IP Router"))
    parser.add_argument("-n", "--nologin", action='store_true', help="Use this if you don't have any credentials")
    parser.add_argument("-u", "--username", type=str, default="admin", help="Username used to login to the router, default=admin")
    parser.add_argument("-p", "--password", type=str, default="admin", help="Password used to login to the router, default=admin")
    parser.add_argument("-w", "--write", type=str, help="Filename", metavar=("FILENAME"))
    args = parser.parse_args()

    #Check if atleast the ip address is given
    if args.ip == None:
        parser.print_usage()
    elif valid_ip(args.ip) == False:
        sys.exit("Enter a valid IP address")

    if args.nologin == True:
    try:
        connection = Connection('http://192.168.8.1/')
        client = Client(connection)
        data = getAllInfoWithoutLogin(client)
        if args.write:
            print("Data has been written into : " + args.write + "\n")
            writeToFile(args.write, data)
        else:
            print(data)
    except Exception as e:
        print(e)

    if args.username == True and args.password == True:
    try:

```

```

connection = AuthorizedConnection('http://'+args.username+':'+args.password+'@'+args.ip+'/')
client = Client(connection)
data = getAllInfoLogin(client)
    if args.write:
        print("Data has been written into : " + args.write + "\n")
        writeToFile(args.write, data)
    else:
        print(data)
except Exception as e:
    print(e)

if __name__ == "__main__":
    main()

```

huawei_api.py

```

from huawei_lte_api.Client import Client
import huawei_time_zones as zone
import json
import calendar

def beautifyjson(jsonobj):
    text = json.dumps(jsonobj, sort_keys=True, indent=4)
    return text

def getISP(client):
    """
    Retrieve the ISP operating the router
    """
    data = beautifyjson(client.net.current_plmn())
    new_dict = json.loads(data)
    isp = new_dict["FullName"]
    return "[+]\tInternet Service Provider(ISP)\t[+]\nProvider : " + isp + "\n"

def getDeviceInformation(client):
    """
    Retrieve all the devices information.
    Model, firmware version, IMEI, serial, ...
    """
    data = beautifyjson(client.device.information())
    new_dict = json.loads(data)
    formatted_info = "[+]\tDevice Information\t[+]\n"
    formatted_info += "Device Name :\t\t" + new_dict["DeviceName"] + "\n"
    formatted_info += "Software Version : \t" + new_dict["SoftwareVersion"] + "\n"
    formatted_info += "Hardware Version : \t" + new_dict["HardwareVersion"] + "\n"
    formatted_info += "Mac Address : \t\t" + new_dict["MacAddress1"] + "\n"
    formatted_info += "Serial Number :\t\t" + new_dict["SerialNumber"] + "\n"
    formatted_info += "\n[+] Sim Card Information [+)\n"

```

```

formatted_info += "Sim Card Identity(Iccid): \t" + new_dict["Iccid"] + "\n"
formatted_info += "Mobile Identity(Imei): \t\t" + new_dict["Imei"] + "\n"
formatted_info += "Mobile subscriber(Imsi): \t" + new_dict["Imsi"] + "\n"

return formatted_info

def getLanguage(client):
    """
    Retrieve the language of the router
    """
    data = jsonify(client.language.current_language())
    new_dict = json.loads(data)
    lang = new_dict["CurrentLanguage"]
    return "[+]\tCurrent Language of router\t[+]\nLanguage: " + lang + "\n"

def getConnectedDevices(client):
    """
    Retrieve the currently connected devices
    """
    data = jsonify(client.wlan.host_list())
    new_dict = json.loads(data)
    formatted_devices = "[+]\tList of connected devices\t[+]\n"
    if len(new_dict["Hosts"]["Host"]) == 0:
        formatted_devices += "No connected devices currently on the WiFi\n"
    else:
        for x in range(0, len(new_dict["Hosts"]["Host"])):
            formatted_devices += "[+]\tDevice number " + str(x+1) + "\t[+]\n"
            formatted_devices += "Hostname: \t" + new_dict["Hosts"]["Host"][x]["HostName"] + "\n"
            formatted_devices += "IP address: " + new_dict["Hosts"]["Host"][x]["IpAddress"] + "\n"
            formatted_devices += "Mac address: " + new_dict["Hosts"]["Host"][x]["MacAddress"] + "\n"
    return formatted_devices

def getTimezone(client):
    """
    Retrieve NTP information
    """
    data = jsonify(client.s_ntp.get_settings())
    new_dict = json.loads(data)
    current_time = new_dict["time"]
    year = current_time[0:4]
    month = calendar.month_name[int(current_time[4:6])] #current_time[4:6]
    day = current_time[6:8]
    hour = current_time[8:10]
    minutes = current_time[10:12]
    seconds = current_time[12:14]
    current_time = "[+]\tNTP Information (Time)\t[+]\n"
    current_time += "Date: " + day + " " + month + " " + year + "\nTime: " + hour + ":" + minutes + ":" + seconds
    time_zone = "Timezone: " + zone.getTimeZoneName(new_dict["timezone"])
    return current_time + "\n" + time_zone + "\n"

```

```

def getLogs(client):
    """
    Retrieve SysLogs - ONLY LOGS AVAILABLE WITHOUT TOUCHING THE HARDWARE SEEMS LIKE
    Log format: "Time","Level","Module name","Result","Content"
    Level : 0 -> Informative
    Level : 1 -> Warning
    Level : 2 -> Error
    Level : 3 -> Critical

    Module name : 0 -> Platform
    1,2??
    Module name : 3 -> Upgrade

    Result: 0 -> Succeeded
    Result: 1 -> Failed

    Content: 0xff1a0001 -> Log in (Succeed)
    Content: 0xff1a0002 -> Log in (Failed)
    Content: 0xff1a0003 -> Log out (Succeed)
    Other codes not disclosed ?!
    """
    def checkLevel(level):
        if level == '0':
            return "Informative"
        elif level == '1':
            return "Warning"
        elif level == '2':
            return "Error"
        else:
            return "Critical"

    def checkModule(level):
        if level == '0':
            return "Platform"
        elif level == '3':
            return "Upgrade"

    def checkResult(level):
        if level == '0':
            return "Succeeded"
        elif level == '1':
            return "Failed"

    def checkContent(level):
        if level == '0xff1a0001':
            return "Log in"
        elif level == '0xff1a0002':
            return "Log in"
        elif level == '0xff1a0003':
            return "Log out"
        else:

```

```

        return level + ". Code is not known, check on the web UI"

data = client.syslog.querylog()["content"]
lines = data.split(";")
new_lines = []
for x in lines:
    new_data = x.split(",")
    if " " in new_data:
        break
    new_data[1] = checkLevel(new_data[1])
    new_data[2] = checkModule(new_data[2])
    new_data[3] = checkResult(new_data[3])
    new_data[4] = checkContent(new_data[4])
    to_str = " ".join(new_data)
    new_lines.append(to_str)

formatted_logs = "[+]\tSyslog Information\t[+]\n"

for x in new_lines:
    formatted_logs += x+"\n"
return formatted_logs + "\n"

```

huawei_time_zones.py

```

new_dict = {
    "0": "GMT-12:00 - International Date Line West",
    "1": "-12:00 (GMT-12:00) International Date Line West",
    "2": "-11:00 (GMT-11:00) Midway Island, Samoa",
    "3": "-10:00 (GMT-10:00) Hawaii",
    "4": "-09:00 (GMT-09:00) Alaska",
    "5": "-08:00 (GMT-08:00) Pacific Time, Tijuana",
    "6": "-07:00 (GMT-07:00) Arizona, Mazatlan",
    "7": "-07:00 (GMT-07:00) Chihuahua, La Paz",
    "8": "-07:00 (GMT-07:00) Mountain Time",
    "9": "-06:00 (GMT-06:00) Central America",
    "10": "-06:00 (GMT-06:00) Central Time",
    "11": "-06:00 (GMT-06:00) Guadalajara, Mexico City, Monterrey",
    "12": "-06:00 (GMT-06:00) Saskatchewan",
    "13": "-05:00 (GMT-05:00) Bogota, Lima, Quito",
    "14": "-05:00 (GMT-05:00) Eastern Time",
    "15": "-05:00 (GMT-05:00) Indiana",
    "16": "-04:00 (GMT-04:00) Atlantic Time",
    "17": "-04:00 (GMT-04:00) Caracas, La Paz",
    "18": "-04:00 (GMT-04:00) Santiago",
    "19": "-03:30 (GMT-03:30) Newfoundland",
    "20": "-03:00 (GMT-03:00) Brasilia",
    "21": "-03:00 (GMT-03:00) Buenos Aires, Georgetown",
    "22": "-03:00 (GMT-03:00) Greenland",
    "23": "-02:00 (GMT-02:00) Mid-Atlantic",

```

"24" : "-01:00 (GMT-01:00) Azores",
"25" : "-01:00 (GMT-01:00) Cape Verde Islands",
"26" : "(GMT) Casablanca, Monrovia",
"27" : "(GMT) Greenwich Mean Time: Dublin, Edinburgh, Lisbon, London",
"28" : "+01:00 (GMT+01:00) Amsterdam, Berlin, Bern, Rome, Stockholm, Vienna",
"29" : "+01:00 (GMT+01:00) Belgrade, Bratislava, Budapest, Ljubljana, Prague",
"30" : "+01:00 (GMT+01:00) Brussels, Copenhagen, Madrid, Paris",
"31" : "+01:00 (GMT+01:00) Sarajevo, Skopje, Warsaw, Zagreb",
"32" : "+01:00 (GMT+01:00) West Central Africa",
"33" : "+02:00 (GMT+02:00) Athens, Istanbul, Minsk",
"34" : "+02:00 (GMT+02:00) Bucharest",
"35" : "+02:00 (GMT+02:00) Cairo",
"36" : "+02:00 (GMT+02:00) Harare, Pretoria",
"37" : "+02:00 (GMT+02:00) Helsinki, Kyiv, Riga, Sofia, Tallinn, Vilnius",
"38" : "+02:00 (GMT+02:00) Jerusalem",
"39" : "+03:00 (GMT+03:00) Baghdad",
"40" : "+03:00 (GMT+03:00) Kuwait, Riyadh",
"41" : "+03:00 (GMT+03:00) Moscow, St. Petersburg, Volgograd",
"42" : "+03:00 (GMT+03:00) Nairobi",
"43" : "+03:30 (GMT+03:30) Tehran",
"44" : "+04:00 (GMT+04:00) Abu Dhabi, Muscat",
"45" : "+04:00 (GMT+04:00) Baku, Tbilisi, Yerevan",
"46" : "+04:30 (GMT+04:30) Kabul",
"47" : "+05:00 (GMT+05:00) Ekaterinburg",
"48" : "+05:00 (GMT+05:00) Islamabad, Karachi, Tashkent",
"49" : "+05:30 (GMT+05:30) Chennai, Kolkata, Mumbai, New Delhi",
"50" : "+05:45 (GMT+05:45) Kathmandu",
"51" : "+06:00 (GMT+06:00) Almaty, Novosibirsk",
"52" : "+06:00 (GMT+06:00) Astana, Dhaka",
"53" : "+06:00 (GMT+06:00) Sri Jayawardenepura",
"54" : "+06:30 (GMT+06:30) Rangoon",
"55" : "+07:00 (GMT+07:00) Bangkok, Hanoi, Jakarta",
"56" : "+07:00 (GMT+07:00) Krasnoyarsk",
"57" : "+08:00 (GMT+08:00) Beijing, Chongqing, Hong Kong, Urumqi",
"58" : "+08:00 (GMT+08:00) Irkutsk, Ulaanbaatar",
"59" : "+08:00 (GMT+08:00) Kuala Lumpur, Singapore",
"60" : "+08:00 (GMT+08:00) Perth",
"61" : "+08:00 (GMT+08:00) Taipei",
"62" : "+09:00 (GMT+09:00) Osaka, Sapporo, Tokyo",
"63" : "+09:00 (GMT+09:00) Seoul",
"64" : "+09:00 (GMT+09:00) Yakutsk",
"65" : "+09:30 (GMT+09:30) Adelaide",
"66" : "+09:30 (GMT+09:30) Darwin",
"67" : "+10:00 (GMT+10:00) Brisbane",
"68" : "+10:00 (GMT+10:00) Canberra, Melbourne, Sydney",
"69" : "+10:00 (GMT+10:00) Guam, Port Moresby",
"70" : "+10:00 (GMT+10:00) Hobart",


```
"71" : "+10:00 (GMT+10:00) Vladivostok",  
"72" : "+11:00 (GMT+11:00) Magadan",  
"73" : "+11:00 (GMT+11:00) Solomon Islands, New Caledonia",  
"74" : "+12:00 (GMT+12:00) Auckland, Wellington",  
"75" : "+12:00 (GMT+12:00) Fiji, Kamchatka, Marshall Islands",  
"76" : "+13:00 (GMT+13:00) Nuku'alofa, Tonga",  
}  
  
def getTimeZoneName(idzone):  
    return new_dict.get(idzone)
```