Università degli Studi di Padova

Dipartimento di Ingegneria dell'Informazione

Corso di Laurea Magistrale in
Ingegneria Informatica

# Managing Security of Computer Network Applications using Encryption Techniques

*Advisor:*
Prof. Nicola Laurenti
*Co-Advisor:*
Prof. Dr-Ing Alexandru Soceanu

*Candidate:*
Marco Martini
1189321

Anno Accademico 2021/2022

# Contents

# Chapter 1

# Abstract


## 1.1 English

This paper covers the usage of SSL and TLS encryption techniques to improve the security of the Computer Network applications including their weaknesses. In order to do that an HTTPS web server will be implemented and will be accessed through a virtual network. The virtual network will be protected through a proprietary Next Generation Firewall (NGFW) from Palo Alto Networks, the paper will explore its Malware Detection and SSL Decryption capabilities showing their advantages and/or weaknesses. In order to verify the Firewall's effectiveness a Man In The Middle (MITM) attack will be deployed inside the virtual network. This paper will end by stating the results obtained by analyzing the NGFW tools and their behaviour against the network attacks.

## 1.2 Italian

Questo documento copre l'utilizzo di tecniche di cifratura SSL e TLS per aumentare la sicurezza di applicazioni di rete rimediando alle loro vulnerabilità. Per farlo verrà creata una rete virtuale che accederà ad un server web HTTPS. La rete virtuale sarà protetta dal Firewall di nuova generazione (NGFW) proprietario di Palo Alto Networks, esplorando le funzionalità di Malware Detection e SSL Decryption, elencandone i vantaggi e/o svantaggi. Per dimostrare l'efficacia del Firewall verrà creato un attacco Man In The Middle (MITM). Si dimostrano infine i risultati dell'esperimento dati dall'analisi del comportamento degli strumenti del Firewall contro gli attacchi di rete.

# Chapter 2

# Introduction

## 2.1 Motivation for the Work

During the past 30 years the way we use computers has fundamentally changed, we now have devices capable of connecting to the Internet in our pockets, and that has lead to an ever increasing interest for companies to focus on the Web.

Nowadays the 55.9% of Alexa's list of most popular sites in the world provide a Secure SSL/TLS implementation [1].

While Encryption provides Confidentiality and Integrity [2] for the end user, it also provide attackers and malware software a way to inject their payload to vulnerable clients without being able to be detected.

This paper will be focused on the Malware protection capabilities that NGFW provide, even in encrypted connections. It's capable of that through an SSL Forward Proxy.

Despite the added security achieved by having a mediator between the untrusted zone (Internet) and the client, a Man In The Middle (MITM) attack could be used to compromise the network if forged well enough, this work will prove whether or not NGFW are effective against this type of attacks.

## 2.2 Objective of the Work

The Objective of this work will be showing how to implement a Decryption tunnel and Malware Detection in Palo Alto FW and demonstrating it's effectiveness when the network has been compromised through a MITM attack.

## 2.3 Summary of the Work

The Work will be as following:

- Setting up the Virtual Network
- Setting up Palo Alto Firewall
- Setting up Malware Detection
- Creating the SSL/TLS Certificates
- Setting up Decryption
- Testing Malware Detection
- Setting up the MITM attack
- Testing Malware Detection again
- Setup a way to block the attack

# Chapter 3

# Description of the Components

The following sections will briefly describe the components used in this experiment.

## 3.1 Next Generation Firewalls and Palo Alto

### 3.1.1 Next Generation Firewalls

Next Generation Firewalls are the evolution of traditional firewalls and are bound to replace them entirely in the corporate space.

Traditional Firewalls can only filter traffic based on state (flow of data instead of single network packets), port, protocol or through hand crafted filters.

Even if a Traditional Firewall is aware of the state of the connection, the data it can extrapolate is very low, for example it knows:

- When was the flow started
- When the flow is being used
- When the flow is being closed

A Next Generation Firewall does everything a Traditional Firewall can and more by using AI enhanced algorithms and by using the Cloud as to always be up to date with new threats and malware.

In order for a Firewall to be classified as “New Generation” it must provide(source: “https://www.cisco.com/c/en/us/products/security/firewalls/what-is-a-next-generation-firewall.html# choose-an-ngfw-firewall”):

- Standard firewall capabilities like stateful inspection
- Integrated intrusion prevention
- Application awareness and control to see and block risky apps
- Threat intelligence sources
- Upgrade paths to include future information feeds
- Techniques to address evolving security threats

### 3.1.2 Palo Alto Firewalls

Palo Alto Networks is an American multinational cybersecurity company based in Santa Clara, California.

Other than the mandatory NGFW features, Palo Alto's Firewall solutions provide many more tools, some of which are [3]:

- Application-based policy enforcement (App-ID?)
- User identification (User-ID?).
- Threat prevention.
- URL filtering.
- Traffic visibility.
- Networking versatility and speed.
- GlobalProtect. (VPN)
- Fail-safe operation.
- Malware analysis and reporting.
- VM-Series firewall.
- Management and Panorama.

This paper will cover the Threat analysis feature of this platform enhanced by the decryption of SSL/TLS packets.

## 3.2 SSL/TLS Decryption

SSL and TLS protocols are the most used protocols to provide secure communication over the internet.

They are present between the Application Layer and the Transport Layer in the TCP/IP stack and enable to identify and authenticate two parties by keeping confidentiality and data integrity.

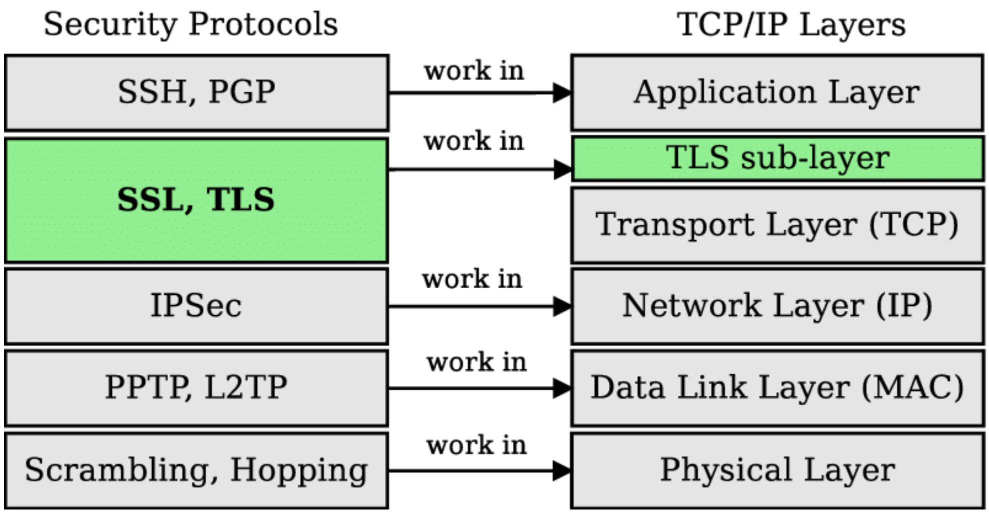



**Figure 3.1:** The SSL Layer in the TCP/IP Stack

In order for the two parties to communicate, an SSL/TLS Handshake must be performed first.

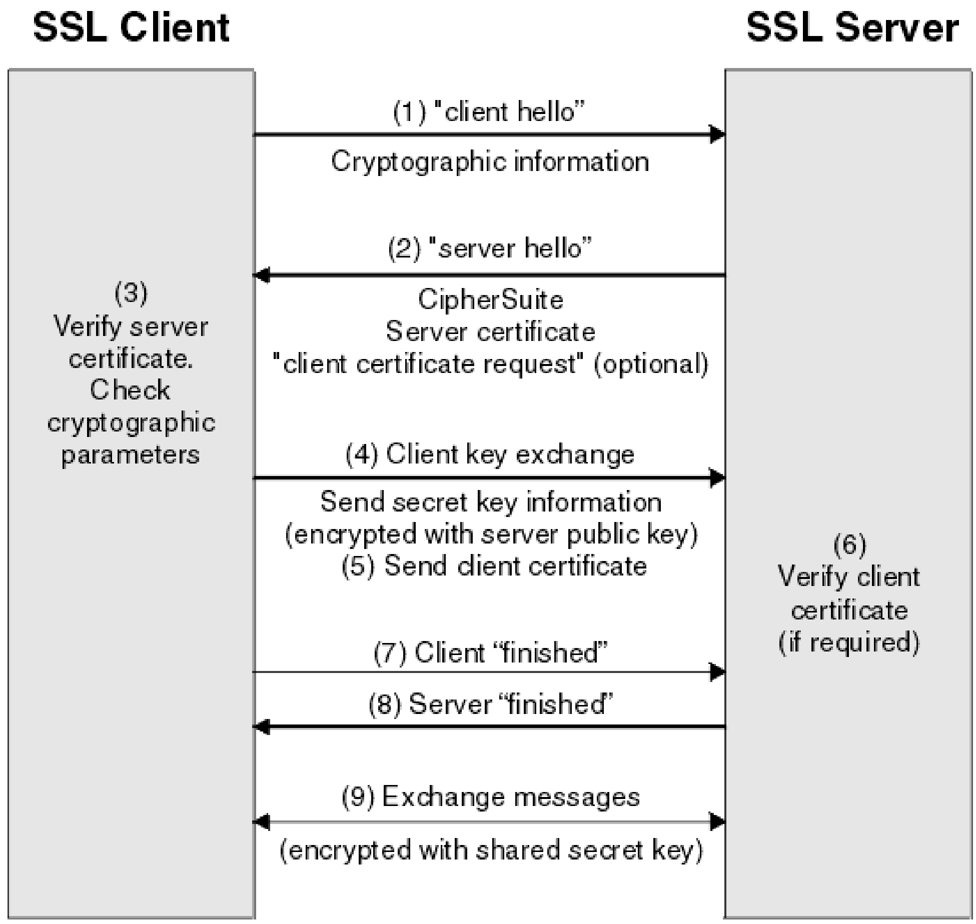



**Figure 3.2:** Overview of the SSL or TLS handshake
Source: https://www.ibm.com/docs/en/ibm-mq/7.5?topic=ssl-overview-tls-handshake

In short the first two packets are needed to establish the role of client/server between the two parties and establish a supported Cipher and Compression method along with the server sending the digital certificate.

After that the client verifies the server's certificate, sends a secret key used to encrypt the following data which is encrypted itself with the server's public key and optionally sends its own certificate in case of a symmetrical encryption method.

Finally both the client and server send a "finished" message encrypted with the secret key indicating that the handshake is complete.

The SSL Decryption covered in this paper refers as a technique where instead of having 2 parties, we have 3:

The server establishes a handshake with the firewall acting as a client and the firewall at the same time establishes an handshake to the real client by acting as the server.

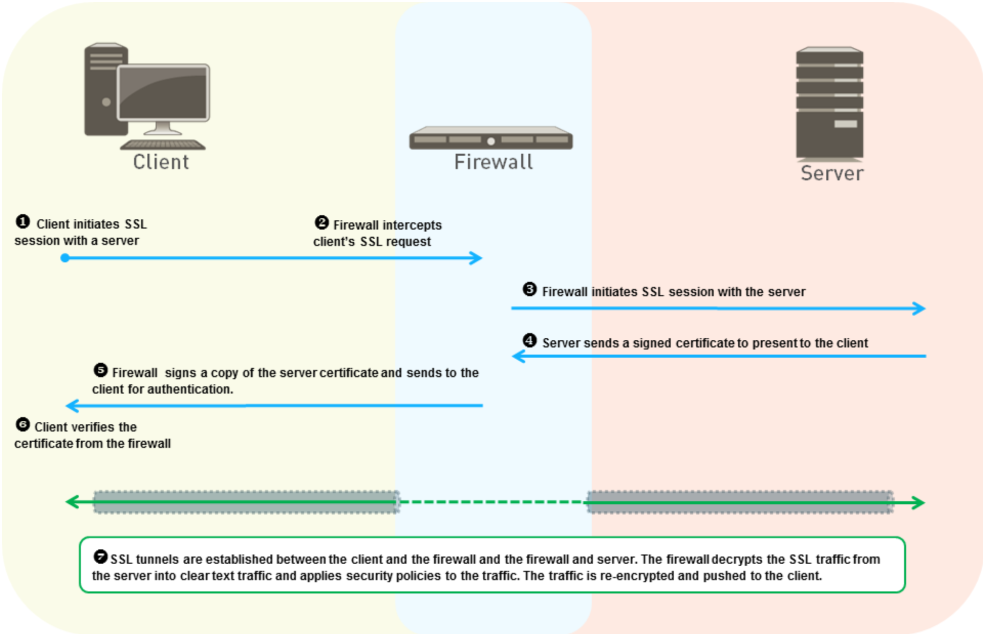



**Figure 3.3:** SSL Forward Proxy Diagram

Source: https://www.ibm.com/docs/en/ibm-mq/7.5?topic=ssl-overview-tls-handshake

## 3.3 Malware Detection in Firewalls

The first line of protection in an organization against malicious attackers is in most cases the Firewall.

Since a Firewall provides a gateway to the outside world it makes sense that a malware protection strategy will also be installed there.

Traditional Firewalls used to only be able to inspect a flow of non-encrypted data, as such, the only way to detect malware was to compare the hash of the downloaded data from the client to a local database which is highly exploitable (by for example changing a few bytes in the payload).

Through NGFWs Malware signatures can be constantly updated through the Cloud and instead of comparing hashes, the threat prevention in this new technology can analyse the payload itself, even if compressed or comes from an encrypted source such as HTTPS.

## 3.4 HTTPS Server with Let's Encrypt

The HTTPS protocol is a secure version of the HTTP, to make it secure, SSL/TLS certificates must be installed into the server that deploys it.

Let's Encrypt is a non-profit Certification Authority that provides TLS certificates for free, although valid for only 90 days.

The official implementation is 'certbot', a tool that automates the generation and renewal of the certificates.

It also provide an automatic certificate installation for 'nginx' and 'Apache', the most popular and Open Source web server software.

## 3.5 The Network Attack

Since SSL Decryption is just a Man in The Middle implementation, the web client must trust the firewall before the website, so if not careful an user can be a victim of another MITM implementation.

### 3.5.1 ARP Spoofing

In order to deploy a successful MITM attack the user must connect to the Attacker machine first.

ARP Spoofing, or ARP poison, consists in a technique where the attacker sends multiple spoofed ARP messages.

Since ARP, Address Resolution Protocol, is used to associate a network device MAC address with its IP-address, spoofing an ARP message means that the attacker will forcefully associate the MAC address of the LAN Gateway to the machine of the attacker himself.

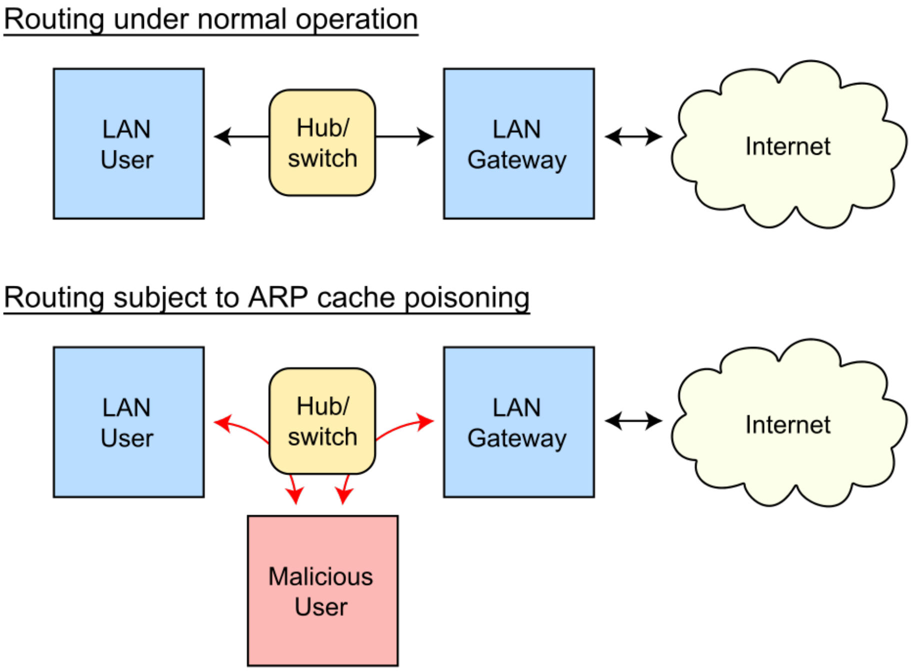



**Figure 3.4:** A successful ARP spoofing (poisoning) attack allows an attacker to alter routing on a network, effectively allowing for a man-in-the-middle attack

Source: https://en.wikipedia.org/wiki/ARP_spoofing

### 3.5.2 HTTPS Proxy

An HTTPS proxy is a server application that acts as an intermediary between a client and a SSL encrypted website.

If used together with a spoofer, in our case an ARP Spoofer, every HTTPS traffic in the network will be redirected to the attacker allowing them to modify the resource at will.

It works exactly like an HTTPS server so it also needs its own SSL/TLS certificates, when used in a Network Attack they're usually forged to seem legitimate.

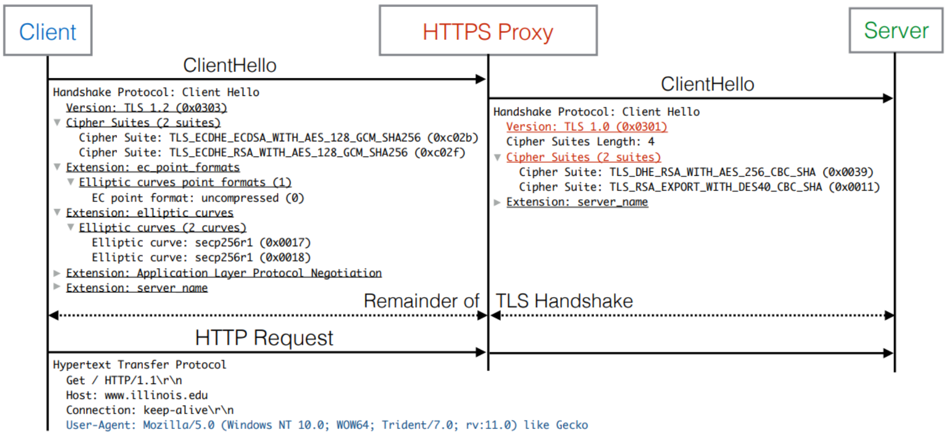



**Figure 3.5:** HTTPS Proxy Interception [4]

# Chapter 4

# The Experiment

## 4.1 Methodology

In order to verify the firewall effectiveness a virtual laboratory will be setup.

The virtual laboratory is deployed through Virtual Machines, since the Palo Alto Firewall is very resource intensive the hypervisor of choice has been KVM(source), with libvirt/qemu as the userspace component.

Instead of direct access to the Internet the VM clients will connect to the host machine, the host will run NGINX(source) configured with Let's Encrypt(source) certificates as the HTTPS server, it will host a simple web page with a link that points to malware.

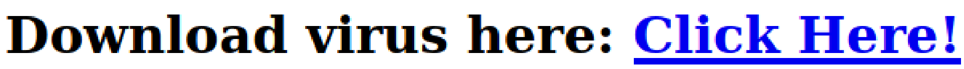



**Figure 4.1:** The web page the client will connect to

```
 <html>
    <body>
        <h1>Download virus here:
            <a href="./eicar.com">
                Click Here!
            </a>
        </h1>
    </body>
</html>
```

The Source Code of the web page

The Malware in question is a test file created by “eicar.org”, the European Institute for Computer Antivirus Research, which is purposely made to test the response of antivirus programs [5], in this case Palo Alto’s Firewall.

The 2 Firewall clients on the other hand will be running Kali Linux, an operating system designed for penetration testing, since it comes preinstalled with many tools.

The 2 clients have different purposes, one will be used as a standard client and the other as a malicious intruder which will deploy the network attack.

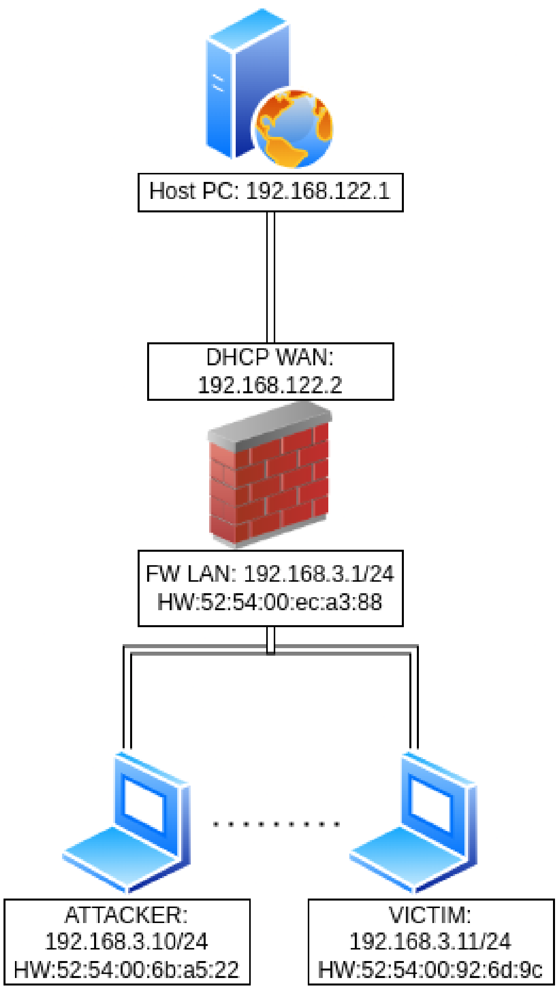
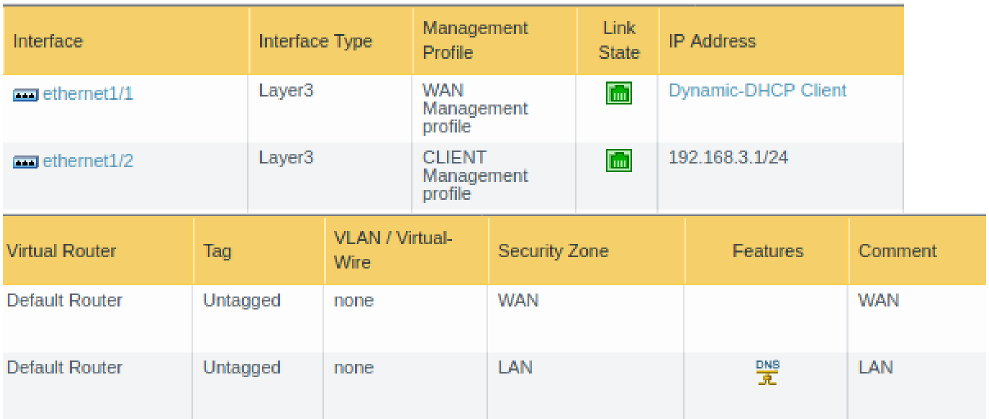



**Figure 4.2:** The Network Plan

## 4.2 Setting up the Firewall

The first thing to do would be setting up the Firewall.

Since it's a simple network the firewall was setup with only 2 network interfaces, a WAN connected interface (in this case the host) and a LAN connected interface, where the clients are connected.



| Interface | Interface Type | Management Profile | Link State | IP Address |
|---|---|---|---|---|
| ethernet1/1 | Layer3 | WAN Management profile | | Dynamic-DHCP Client |
| ethernet1/2 | Layer3 | CLIENT Management profile | | 192.168.3.1/24 |

| Virtual Router | Tag | VLAN / Virtual-Wire | Security Zone | Features | Comment |
|---|---|---|---|---|---|
| Default Router | Untagged | none | WAN | | WAN |
| Default Router | Untagged | none | LAN | DNS | LAN |

**Figure 4.3:** The Network Interfaces' Configuration in Palo Alto FW

The two interfaces must be configured to be part of a Virtual Router, so that the packets can be forwarded to each other.

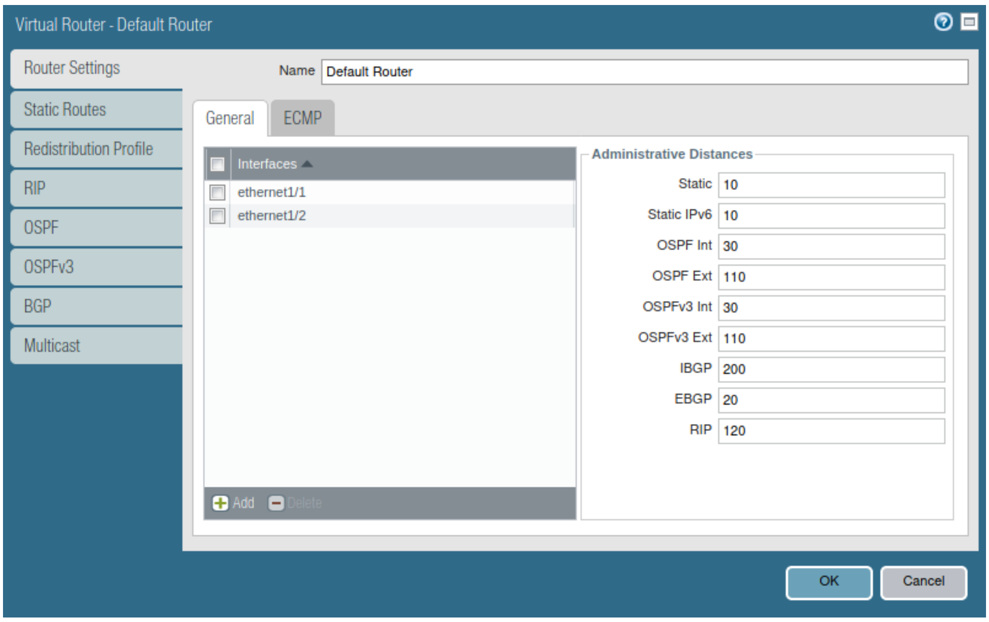



**Figure 4.4:** The Virtual Router Configuration in Palo Alto FW

We need to create some policies in order for the inside network to reach the WAN area.

Since the hosts outside of the internal network have no way to know where the source address is coming from, the next step is configuring NAT Masquerading, It's a technique in which IP addressed are mapped from one realm to another, in this case from the internal network to the external one and vice-versa(source:"https://www.rfc-editor.org/rfc/rfc2663.txt").

| | | | | Source | | | | Destination | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Name | Tags | Type | Zone | Address | User | HIP Profile | Zone | Address | Application | Service | Action | Profile | Options | Hit Count |
| 1 | Pass LAN to WAN a... | none | interzone | LAN | any | any | any | WAN | any | any | application-d... | Allow | none | | 21094 |
| 2 | intrazone-defaul | none | intrazone | any | any | any | any | (intrazone) | any | any | any | Allow | none | none | 2376 |
| 3 | interzone-defaul | none | interzone | any | any | any | any | any | any | any | any | Deny | none | none | 1620 |

**Figure 4.5:** The Firewall Policies in Palo Alto FW

| | | | Original Packet | | | | | | Translated Packet | |
|---|---|---|---|---|---|---|---|---|---|---|
| | Name | Tags | Source Zone | Destination Zone | Destination Interface | Source Address | Destination Address | Service | Source Translation | Destination Translation |
| 1 | Source NAT Masque... | none | LAN | WAN | any | any | any | any | dynamic-ip-and-port ethernet1/1 | none |

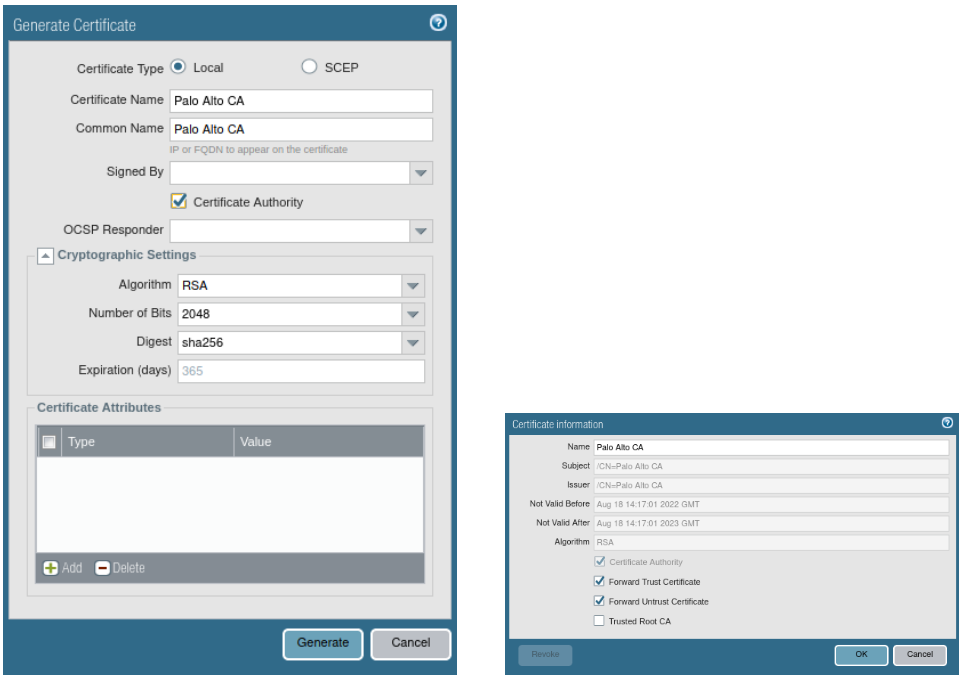
**Figure 4.6:** NAT Masquerading in Palo Alto FW

## 4.3 Setting up Decryption

After the firewall has been configured, much like NAT works, the firewall stands in the middle between outbound and inbound connections.

The firewall connects to the server as the client would, representing it, and uses its own certificates to encrypt the connection between itself and the client making it so that the client believes to communicate directly with the server in a transparent way.

In order to do that we must generate our self signed certificate, and enable the option to Forward Trusted and/or Untrusted Certificates.




(a) The Certificate Generation Menu in Palo Alto FW (b) The Certificate Settings Menu in Palo Alto FW

**Figure 4.7:** SSL/TLS Certificates configuration in PanOS

After the Certificate Generation we need to have a working Decryption Profile, Palo Alto Firewall provides by default a working one but one could create a customised one if needed.

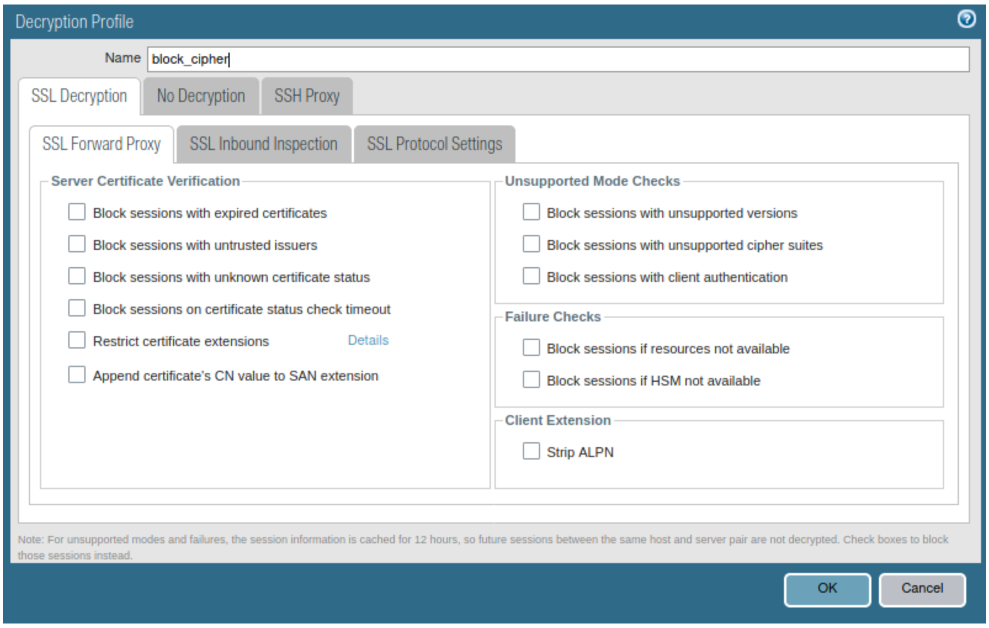



**Figure 4.8:** A few of the many options configurable for Decryption

Finally we can create a Decryption Policy, as with every other Firewall Policy, the source and Destination traffic must be selected, in this case any type of traffic, and then the decryption policy option, there are 3 types of Decryption available in Palo Alto FW: `SSL Forward Proxy`, `SSL Inbound Inspection`, `SSH Proxy`

In this case an SSL Forward Proxy will be used as it's a general approach which works for every SSL/TLS based server without any need to import the private key from external servers.

| | | | Source | | | Destination | |
|---|---|---|---|---|---|---|---|
| | Name | Tags | Zone | Address | User | Zone | Address |
| 1 | Decryption Policy | none | any | any | any | any | any |

| | | Decrypt Options | | |
|---|---|---|---|---|
| URL Category | Service | Action | Type | Decryption Profile |
| any | any | decrypt | ssl-forward-proxy | default |

**Figure 4.9:** Brief overview of the Decryption Policy used

It's also possible to define some exceptions where the website included in it won't ever be decrypted, in case of trusted websites or when the website policy doesn't allow this form of redirection, for example when HSTS (HTTP Strict Transport Security) is enabled.

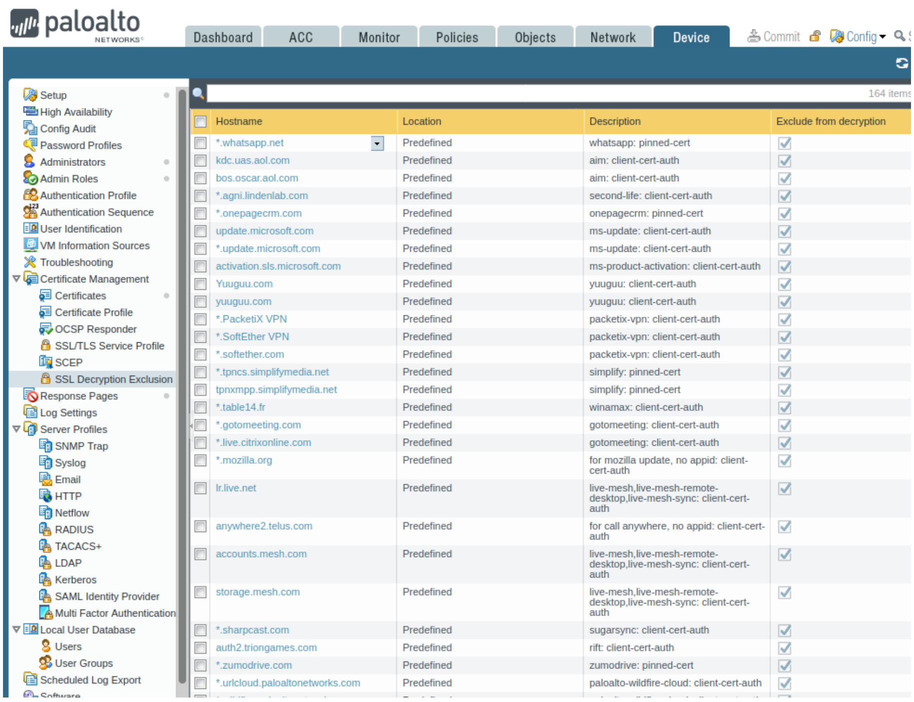



**Figure 4.10:** A list of Decryption Exceptions

To verify if SSL Decryption is working correctly, after connecting to an HTTPS enabled website, this Captive Portal web page should show up before being able to connect.

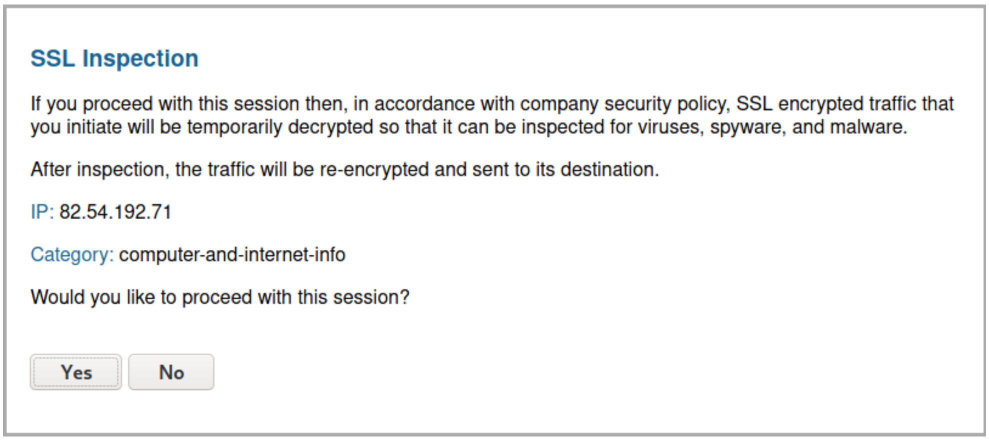



**Figure 4.11:** The SSL Inspection Captive portal

It is also possible to look at the certificate used to decrypt the web page and verify that its Certificate Authority is the same as the one that was generated through the Firewall


**Figure 4.12:** The Palo Alto Generated Certificate on a foreign web page

## 4.4 Setting up Malware Protection

TODO

## 4.5 Testing the Setup

## 4.6 Setting up the Network Attack

In order to setup the Network Attack the Open Source penetration testing tool ‘bettercap‘ will be used.

It’s a “powerful, easily extensible and portable framework written in Go which aims to offer to security researchers, red teamers and reverse engineers an easy to use, all-in-one solution with all the features they might possibly need for performing reconnaissance and attacking WiFi networks, Bluetooth Low Energy devices, wireless HID devices and Ethernet networks.” [6]

It has both a GUI interface and a CLI interface, for simplicity the CLI interface will be used.

After launching it we can look at the various features through the `help` command:

```
     any.proxy > not running
      api.rest > not running
     arp.spoof > not running
     ble.recon > not running
            c2 > not running
       caplets > not running
    dhcp6.spoof > not running
     dns.spoof > not running
 events.stream > running
           gps > not running
           hid > not running
    http.proxy > not running
   http.server > not running
   https.proxy > not running
  https.server > not running
   mac.changer > not running
   mdns.server > not running
  mysql.server > not running
     ndp.spoof > not running
     net.probe > not running
     net.recon > not running
     net.sniff > not running
  packet.proxy > not running
      syn.scan > not running
     tcp.proxy > not running
        ticker > not running
            ui > not running
        update > not running
          wifi > not running
           wol > not running
```

**Figure 4.13:** Bettercap help page

Every tool is called a caplet, we’ll be using the “https.proxy” and “arp.spoof” caplets.

### 4.6.1 Setting up ARP Spoofing

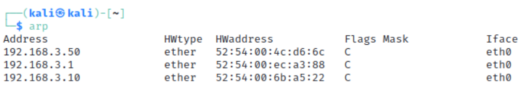
Once bettercap has been launched, to start ARP spoofing we just need to enter `arp.spoof on` in the console.

By default the tool will send a spoofed ARP message with the attacker's IP address associated to the LAN Gateway every second to the entire subnet where the attacker is connected.

The configurable options are [?]:

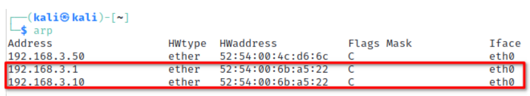
- `arp.spoof.fullduplex` : If true, both the targets and the gateway will be attacked, otherwise only the target (if the router has ARP spoofing protections in place this will make the attack fail). (default=false)
- `arp.spoof.internal` : If true, local connections among computers of the network will be spoofed, otherwise only connections going to and coming from the external network. (default=false)
- `arp.spoof.skip\_restore` : If set to true, targets arp cache won't be restored when spoofing is stopped. (default=false)
- `arp.spoof.targets` : Comma separated list of IP addresses, MAC addresses or aliases to spoof, also supports nmap style IP ranges. (default=¡entire subnet¿)
- `arp.spoof.whitelist` : Comma separated list of IP addresses, MAC addresses or aliases to skip while spoofing. (default=)

The results in the victim are the following:



```
┌──(kali㉿kali)-[~]
└─$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.3.50             ether   52:54:00:4c:d6:6c   C                     eth0
192.168.3.1              ether   52:54:00:ec:a3:88   C                     eth0
192.168.3.10             ether   52:54:00:6b:a5:22   C                     eth0
```

**(a)** ARP Table before Spoofing

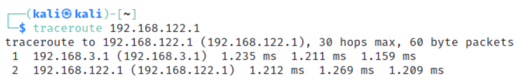


```
┌──(kali㉿kali)-[~]
└─$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.3.50             ether   52:54:00:4c:d6:6c   C                     eth0
192.168.3.1              ether   52:54:00:6b:a5:22   C                     eth0
192.168.3.10             ether   52:54:00:6b:a5:22   C                     eth0
```

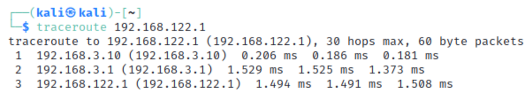
**(b)** ARP Table after Spoofing

**Figure 4.14:** The result of ARP Spoofing/Poisoning

Since the Gateway now points to the Attacker machine, the traceroute output is also changed:




**(a)** Traceroute before ARP Spoof




**(b)** Traceroute after ARP Spoof

### 4.6.2 Setting up the HTTPS Proxy

As with the Arp Spoofer, launching the HTTPS Proxy through bettercap is similar.

The available options are:

- `https.port` : HTTPS port to redirect when the proxy is activated. (default=443)
- `https.proxy.address` : Address to bind the HTTPS proxy to. (default=¡interface address¿)
- `https.proxy.blacklist` : Comma separated list of hostnames to skip while proxying (wildcard expressions can be used). (default=)
- `https.proxy.certificate` : HTTPS proxy certification authority TLS certificate file. (default= /.bettercap-ca.cert.pem)
- `https.proxy.certificate.bits` : Number of bits of the RSA private key of the generated HTTPS certificate. (default=4096)
- `https.proxy.certificate.commonname` : Common Name field of the generated HTTPS certificate. (default=Go Daddy Secure Certificate Authority - G2)
- `https.proxy.certificate.country` : Country field of the generated HTTPS certificate. (default=US)
- `https.proxy.certificate.locality` : Locality field of the generated HTTPS certificate. (default=Scottsdale)
- `https.proxy.certificate.organization` : Organization field of the generated HTTPS certificate. (default=GoDaddy.com, Inc.)

- `https.proxy.certificate.organizationalunit` : Organizational Unit field of the generated HTTPS certificate. (default=https://certs.godaddy.com/repository/)
- `https.proxy.injectjs` : URL, path or javascript code to inject into every HTML page. (default=)
- `https.proxy.key` : HTTPS proxy certification authority TLS key file. (default= /.bettercap-ca.key.pem)
- `https.proxy.port` : Port to bind the HTTPS proxy to. (default=8083)
- `https.proxy.redirect` : Enable or disable port redirection with iptables. (default=true)
- `https.proxy.script` : Path of a proxy JS script. (default=)
- `https.proxy.sslstrip` : Enable or disable SSL stripping. (default=false)
- `https.proxy.whitelist` : Comma separated list of hostnames to proxy if the blacklist is used (wildcard expressions can be used). (default=)

As mentioned earlier in the paper, the certificate can be forged meticulously to look like a big corporation's one.

### Developing the Script

Through the `https.proxy.script` option, it is possible to develop a small javascript program that interacts with various proxy functions [7]:

```
 // called when the script is loaded
function onLoad() {

}

// called when the request is received by the proxy
// and before it is sent to the real server.
function onRequest(req, res) {

}

// called when the request is sent to the real server
// and a response is received
function onResponse(req, res) {

}

// called every time an unknown session command is typed,
// proxy modules can optionally handle custom commands this way:
function onCommand(cmd) {
    if( cmd == "test" ) {
        /*
         * Custom session command logic here.
         */

        // tell the session we handled this command
        return true
    }
}
```

In this case in order to bypass the Malware Detection in “PanOS”, a small script was created that replaces the url of the malware on the external server with the url of the same malware but located on the attacker’s machine.

```
function onResponse(req, res){
        var body = res.ReadBody();
        //Checks if there's a link to eicar.com
        if ( body.indexOf('<a href="./eicar.com">') != -1){
            res.Body = body.replace('<a href="./eicar.com">',
            '<a href="http://192.168.3.10/eicar2.com">');
        }
}
```

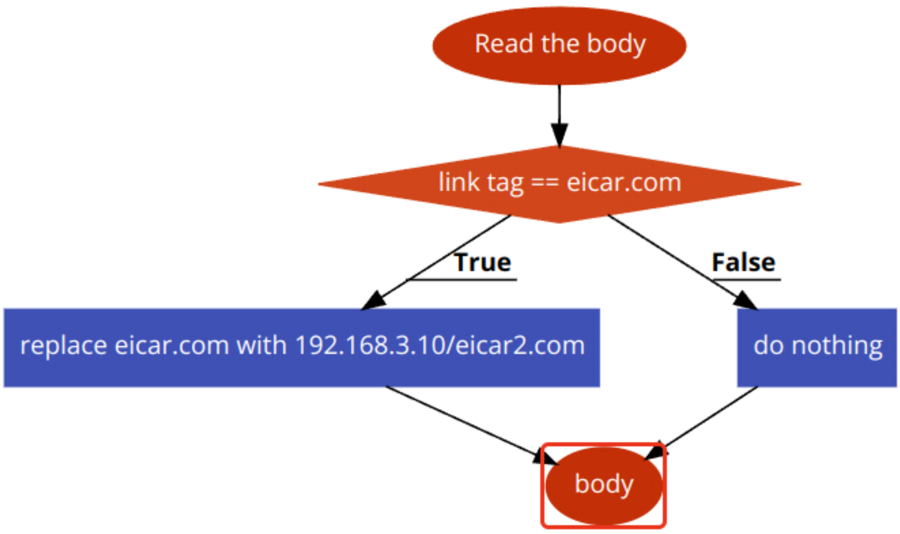



**Figure 4.16:** The Script’s Flowchart

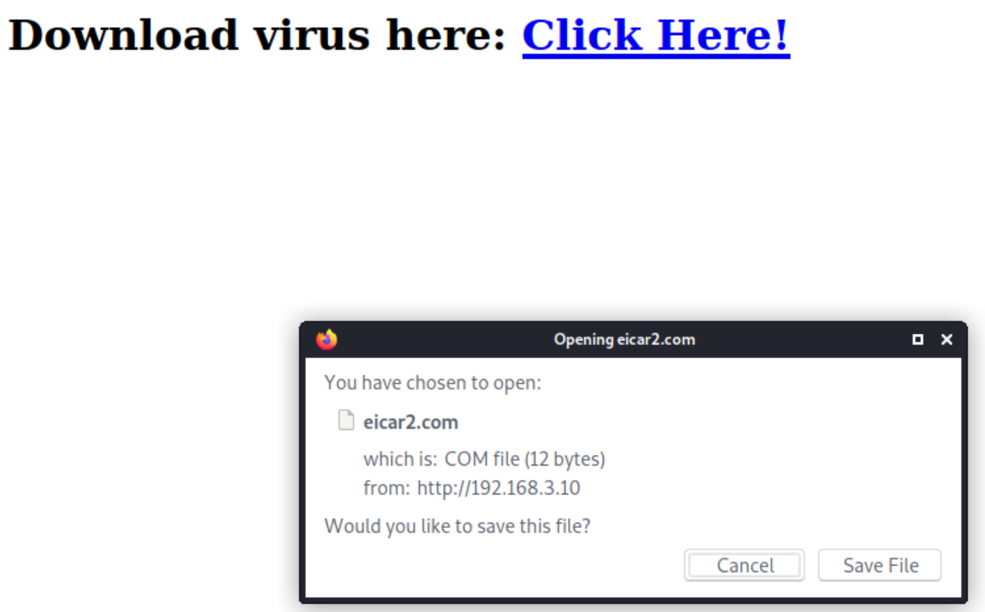
### Testing the Proxy

Once the script has been written, to load it into the https.proxy the command `set https.proxy.script /file/location/script.js‘` is used in the bettercap shell and the proxy itself is started through the `https.proxy on` command.

When loaded, once the client connects to the web server the SSL/TLS certificate will also have changed.


**Figure 4.17:** The compromised website along with its forged Certificate

And by Clicking the link the download will start as envisioned.




This way the victim is downloading a malware by completely bypassing the Malware Detection system in the firewall.

## 4.7 Mitigating the Attack

# Chapter 5

# Results and discussion

## 5.1 Conclusions

# References


[1] "Ssl pulse." https://www.ssllabs.com/ssl-pulse/, 2022. SSL Pulse is a continuous and global dashboard for monitoring the quality of SSL / TLS support over time across 150,000 SSL- and TLS-enabled websites, based on Alexa's list of the most popular sites in the world.

[2] "How ssl and tls provide identification, authentication, confidentiality, and integrity." https://www.ibm.com/docs/en/ibm-mq/7.5?topic=ssl-how-tls-provide-authentication-confidentiality-integrity, 2022.

[3] "Pan-os web interface reference: Features and benefits." Available Online https://docs.paloaltonetworks.com/pan-os/9-1/pan-os-web-interface-help/web-interface-basics/features-and-benefits, August 2022.

[4] Z. D. Z. M. D. S. R. B. N. S. E. B. M. B. J. A. Halderman and V. Paxson, "The security impact of https interception." Network and Distributed System Security Symposium (NDSS) Online available Online available https://zakird.com/papers/https_interception.pdf, February 2017.

[5] "Anti malware testfile." Available Online https://www.eicar.org/download-anti-malware-testfile/.

[6] "bettercap." https://github.com/bettercap/bettercap.

[7] "Https proxy modules, bettercap." Available Online https://www.bettercap.org/modules/ethernet/proxies/http.proxy/.

# List of Figures