Network Hardening Assignment - Group 09

Student Names and Numbers:

- Alessio Civica, Mat: 1916744 - civica.1916744@studenti.uniroma1.it

- Antonio Giorgino, Mat: 2126475 - giorgino.2126475@studenti.uniroma1.it

- Federico Detomaso, Mat: 1903906 - detomaso.1903906@studenti.uniroma1.it

Brainstorming

The search for security implementations to be provided to the network started from a number of our own considerations and security analyses performed by the host greenbone. The first improvements identified concerned the most basic aspects of ACME network security, so we started with the security configurations of the two firewalls in the network of which we identified the need for stronger passwords and a configuration of the firewalls that would allow their logs to be sent to the servers set up within the network. We then moved on to identifying attacks that can be executed through the use of the network, identifying the need to protect the entire network from attacks such as ICMP redirect or unreachable attacks that require special configuration of the two network firewalls. Another point of discussion was then inherent in the digital certificates we issued and self-signed for the configuration of internal services in our network and the protection of the host fantasticcoffe, which, unlike other hosts, which does not use the proxy for network access and especially does not use the HTTPS protocol for communications and requests from him. We therefore identified the need to replace the default certificates with those of our certification authorities in order to have a better security of the authentication certificate for encryption and authentication method; and the need to configure our own to support the protection of fantasticcoffe's services.

After that we've used the Greenbone scan to evaluate the security of our network, finding inside of it some vulnerabilities here listed

- One related to a deprecated version of OpenSSL
- Some Squid 0-Day Vulnerability that, unfortunately, which to this day have not yet been resolver by the software owners
- Unencrypted communication inside the network, the most important between firewall

By doing this, we've found the necessity to provide a daily scan of the networks that will be done during the lunch break in order to avoid an excessive flow of communications within the network and to provide a report that can be consulted at appropriate times by those who manage security.

Our goal has been to think about the logic of the network in such a way as to configure it to be frequently scanned for new vulnerabilities and to be able to withstand a wide range of cyber attacks that affect not only the software used but also the underlying infrastructure. To fulfill this need, the main idea, as is already understandable, was to configure the two firewalls in such a way that they would limit communications to only those necessary and expected, and we configured hosts and services to collaborate and work together with the security systems or, likewise, be the "security systems" themselves.

Protection plan definition

We decided to start by implementing the security configurations and then adapt to the new settings and communications, the host and firewall rules so that the new configurations could work properly. In particular, we addressed the following new configurations

1. Changing the main password for both Main and Internal Firewall
2. Enabling the NTP server inside the firewalls and enabled the host to use it
3. Forcing HTTPS communication between both servers
4. Enabling daily network scan by Greenbone
5. Enabling the Rsyslog to send his log for analysis to Graylog
6. Configuring the reverse proxy on the webserver to make fantasticcoffe available from the outside in a secure way
7. Forced to use OpenSSL version greater than 1.1
8. Disabled dangerous traffic made by spoofed IPs, ICMP request, ICMP reply and ICMP redirect to avoid network scanning and to avoid routing table manipulation
9. Replaced the default linux certificates with those generated by our certificate authorities (used to create the VPN)

We then moved on to reogranize the firewall and configurations to make the services collaborate with each other and to enable the newly generated traffic by following the next steps

1. We’ve limited the ICMP traffic to 10 kb/s to avoid network attacks
2. Enabled inside the SERVERS network of Internal firewall communication between graylog and rsyslog to transfer the collected log
3. Enabled the NTP communication inside the Main Firewall and Internal firewall from both firewall to all the network
4. Disabled packet reception containing ICMP redirect, request and unreachable
5. Allowed the HTTPS connection between the webserver and the fantasticcoffe

Protection plan implementation

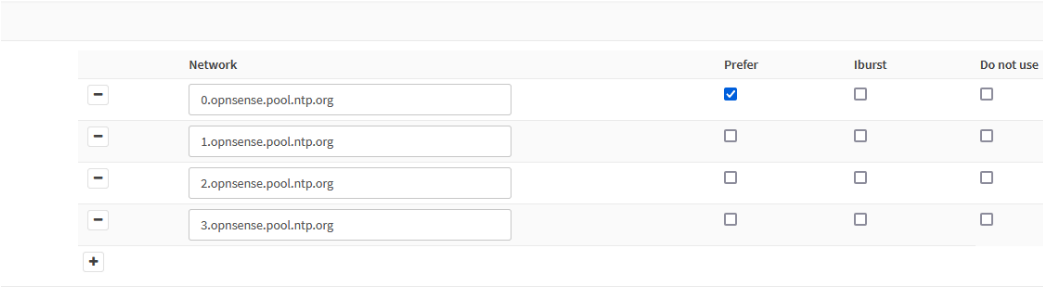
Changing the firewall password

By going into the Password tab of both firewall, we've changed the password to

- **Main firewall:** L0lI0D1Col9o?!
- **Internal firewall:** mAnut0L'1Mbut0

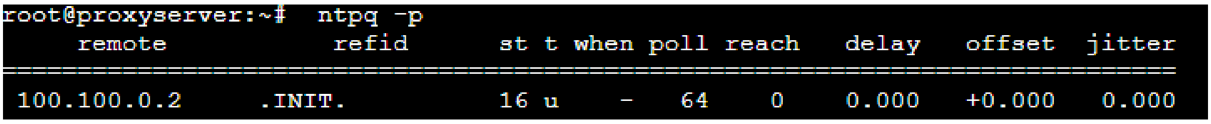
Enabling the NTP server

First the NTP server is activated on the General Tab of Network time for all the network of both the firewall.



	Network	Prefer	Iburst	Do not use
−	0.opnsense.pool.ntp.org	☑	□	□
−	1.opnsense.pool.ntp.org	□	□	□
−	2.opnsense.pool.ntp.org	□	□	□
−	3.opnsense.pool.ntp.org	□	□	□
+				

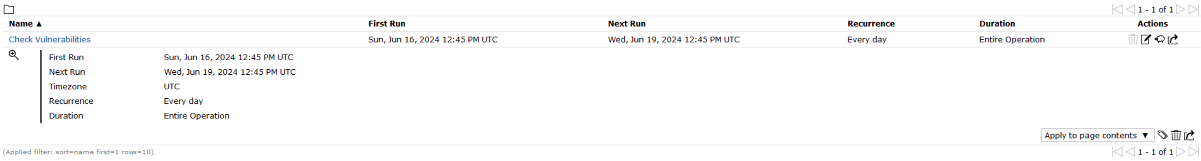
Then we downloaded NPT service with *apt install npt* and put inside */etc/npt.conf* the string needed to make a host with the NPT server of his connected firewall (for the main firewall the string is *100.100.0.2 iburst*). The result is



```
root@proxyserver:~#  ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
 100.100.0.2     .INIT.          16 u    -   64    0    0.000   +0.000   0.000
```

Daily host scan

The greenbon host is setted, via the schedule tab, to scan the host every day at a specific hour. The execution is set to start each day around 12:45 p.m. because we identified this as the typical time of a possible lunch break and a consequent decrease in network traffic



1 - 1 of 1

Name ▲	First Run	Next Run	Recurrence	Duration	Actions
Check Vulnerabilities	Sun, Jun 16, 2024 12:45 PM UTC	Wed, Jun 19, 2024 12:45 PM UTC	Every day	Entire Operation	

First Run	Sun, Jun 16, 2024 12:45 PM UTC
Next Run	Wed, Jun 19, 2024 12:45 PM UTC
Timezone	UTC
Recurrence	Every day
Duration	Entire Operation

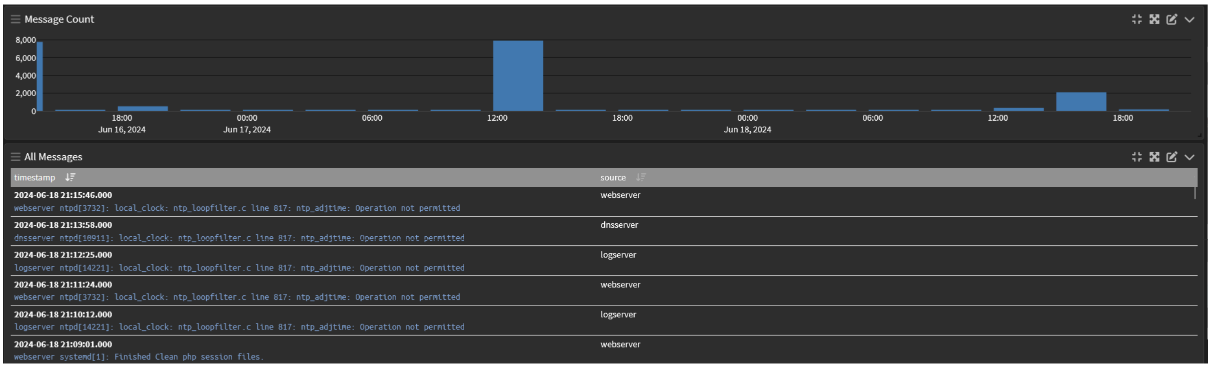
(Applied filter: sort=name first=1 rows=10)

1 - 1 of 1

RSylog and Graylog working together

We had the local log server send data on UDP port 514 to the graylog server, which, listening on that port, would pick up the sent data and use it for log analysis.
The communication was configured by entering, inside */etc/rsyslog.conf* of log server, the line **.* @100.100.1.4:514* and enabling communication between the two within the SERVERS interface of the internal firewall. The result can be viewed inside the Graylog dashboard for the log server

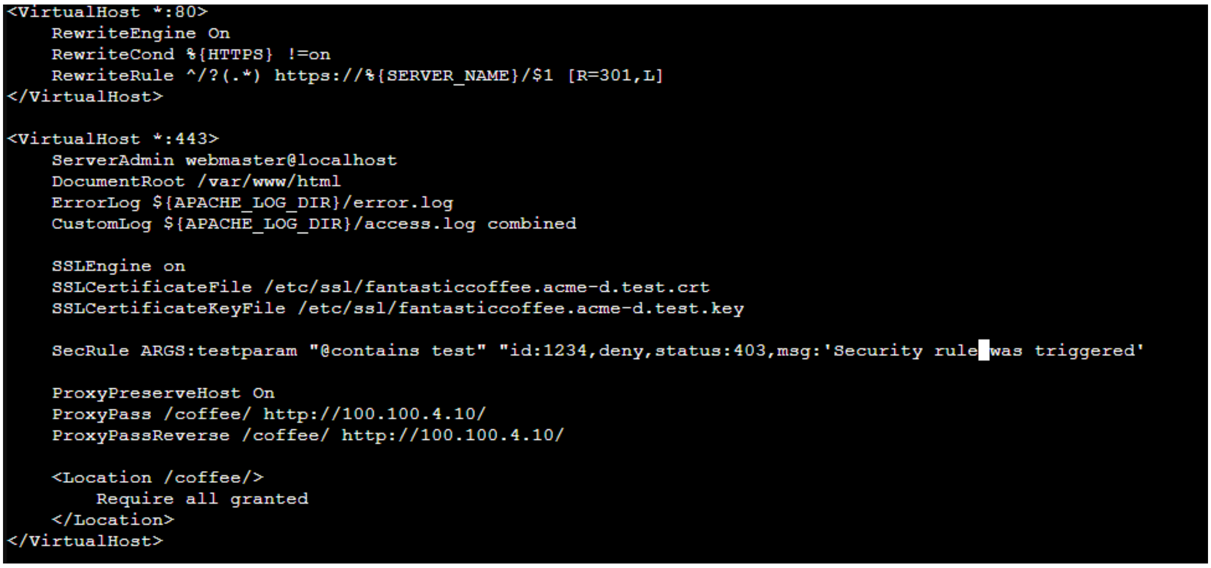
and enabling communication between the two within the SERVERS interface of internal firewall



```
timestamp                                                          source
2024-06-18 21:15:46.000                                            webserver
webserver ntpd[3732]: local_clock: ntp_loopfilter.c line 817: ntp_adjtime: Operation not permitted
2024-06-18 21:13:58.000                                            dnsserver
dnsserver ntpd[10911]: local_clock: ntp_loopfilter.c line 817: ntp_adjtime: Operation not permitted
2024-06-18 21:12:25.000                                            logserver
logserver ntpd[14221]: local_clock: ntp_loopfilter.c line 817: ntp_adjtime: Operation not permitted
2024-06-18 21:11:24.000                                            webserver
webserver ntpd[3732]: local_clock: ntp_loopfilter.c line 817: ntp_adjtime: Operation not permitted
2024-06-18 21:10:12.000                                            logserver
logserver ntpd[14221]: local_clock: ntp_loopfilter.c line 817: ntp_adjtime: Operation not permitted
2024-06-18 21:09:01.000                                            webserver
webserver systemd[1]: Finished Clean php session files.
```

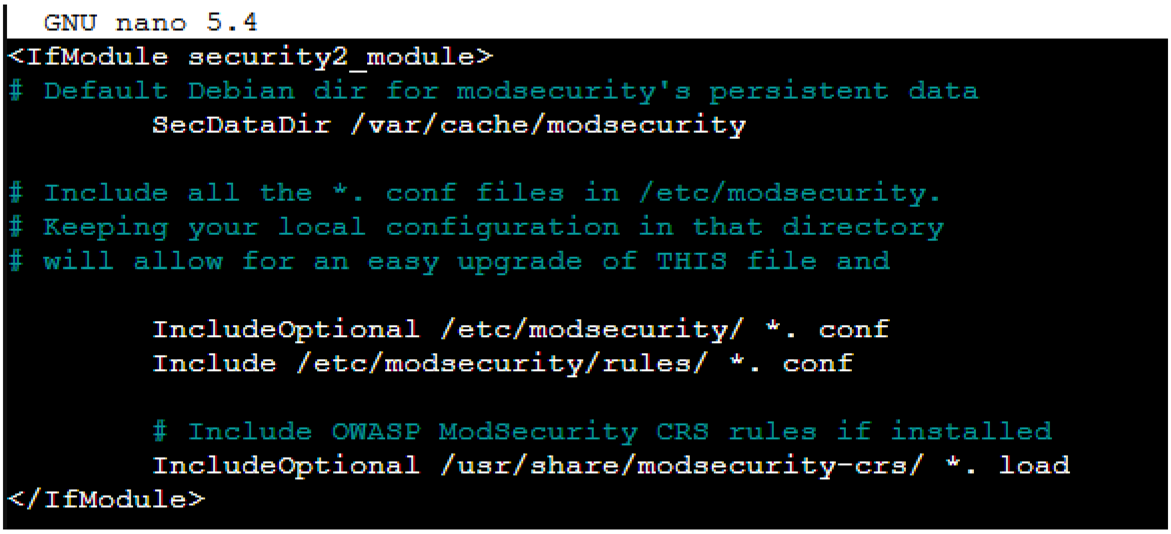
Configuration of reverse proxy

The reverse proxy configuration started by installing and enabling the *ssl, proxy, http_proxy* and *proxy_balancer* module on Apache to enable the configuration of the reverse proxy via the */etc/apache2/sites-available/000-default.conf* by setting the properties for the reverse proxy and by configuring it to reach the fantasticcoffe host on https://100.100.6.2/coffe/



```
<VirtualHost *:80>
    RewriteEngine On
    RewriteCond %{HTTPS} !=on
    RewriteRule ^/?(.*) https://%{SERVER_NAME}/$1 [R=301,L]
</VirtualHost>

<VirtualHost *:443>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/html
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    SSLEngine on
    SSLCertificateFile /etc/ssl/fantasticcoffee.acme-d.test.crt
    SSLCertificateKeyFile /etc/ssl/fantasticcoffee.acme-d.test.key

    SecRule ARGS:testparam "@contains test" "id:1234,deny,status:403,msg:'Security rule was triggered'

    ProxyPreserveHost On
    ProxyPass /coffee/ http://100.100.4.10/
    ProxyPassReverse /coffee/ http://100.100.4.10/

    <Location /coffee/>
        Require all granted
    </Location>
</VirtualHost>
```

After the configuration of the reverse proxy, the next step was to configure the modsecurity by downloading the modules *libapache2-mod-security2*. After the installation we chose to provide the OWAS security rules for modsecurity since we consider it a more secure and complete solution than any custom configuration implemented ad-hoc by us. Placed the file containing the rules in the rules/ directory of OWASP modsecurity folder inside the local within */etc/modsecurity/,* we configured the security2.conf file to reach our rules inside the /etc/modsecurity/rules/ folder



```
  GNU nano 5.4
<IfModule security2_module>
# Default Debian dir for modsecurity's persistent data
        SecDataDir /var/cache/modsecurity

# Include all the *. conf files in /etc/modsecurity.
# Keeping your local configuration in that directory
# will allow for an easy upgrade of THIS file and

        IncludeOptional /etc/modsecurity/ *. conf
        Include /etc/modsecurity/rules/ *. conf

        # Include OWASP ModSecurity CRS rules if installed
        IncludeOptional /usr/share/modsecurity-crs/ *. load
</IfModule>
```

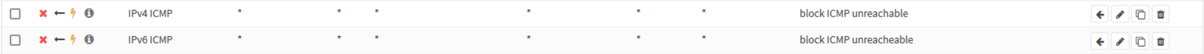
After this configuration and a fast restart of the httpd and apache2 services, the fantasticcoffe host was ready to be used in a secure way!

Note for the professor: due to the problems communicated via we email had with the webserver and apache service the reverse proxy configuration is present on the system but it is not working nor could it be tested. We apologize for the malfunction and hope you can understand us

Firewall rules for dangerous packets

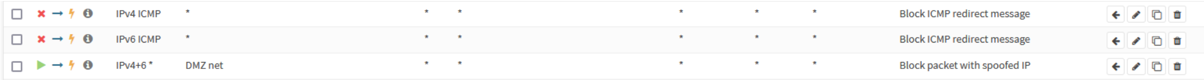
Both the Main and Internal firewall firewall tables have been updated with rules to avoid dangerous packet like:

- **ICMP unreachable:** used to avoid network scanning by malicious user and to avoid DoS attack to the network



□	✗ ← ⚡ ℹ	IPv4 ICMP	*	*	*	*	*	*	block ICMP unreachable
□	✗ ← ⚡ ℹ	IPv6 ICMP	*	*	*	*	*	*	block ICMP unreacheable

- **ICMP redirect & spoofed IP tables:** used by malicious users to manipulate the routing table of a router by declaring a host unreachable. The spoofed IP, on the other hand, can be used for a various range of attacks like DDoS, amplification attack and to elude the firewall



□	✗ → ⚡ ℹ	IPv4 ICMP	*	*	*	*	*	*	Block ICMP redirect message
□	✗ → ⚡ ℹ	IPv6 ICMP	*	*	*	*	*	*	Block ICMP redirect message
□	▶ → ⚡ ℹ	IPv4+6 *	DMZ net	*	*	*	*	*	Block packet with spoofed IP

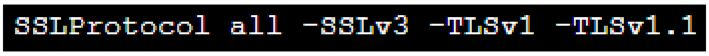
- **Limited ICMP traffic bandwidth:** the ICMP bandwidth traffic has been restricted to 10 kb/s to make more easy the life of the device and to put a limit to the network attacks that use ICMP

OpenSSL version forcing

The OpenSSL v1.1 service is very very dangerous since it has a lot of known vulnerabilities that can be exploited in order to take control of the system that uses it like the **Heartbleed**

vulnerability, an overflow attack on the TLS packet that made it possible to read the memory of a server.

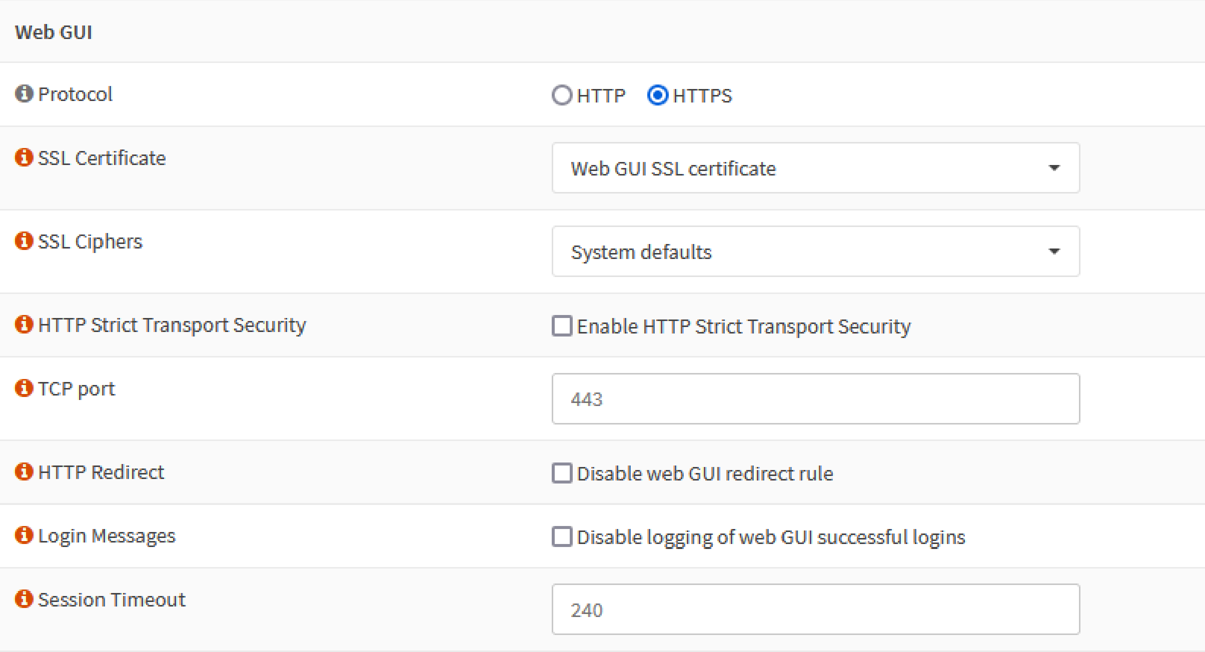
To avoid this kind of problem we configured the */etc/apache2/mods-available/ssl.conf* in the webserver to force using encryption protocol that aren’t the OpenSSL v1.1



```
SSLProtocol all -SSLv3 -TLSv1 -TLSv1.1
```

HTTPS connection between both firewall

By going in the Administration tab of the firewall’s setting it’s been possible configure the HTTPS connection between the two firewall in order to encrypt the communication between the two device and make it more secure



Web GUI	
Protocol	○ HTTP ● HTTPS
SSL Certificate	Web GUI SSL certificate
SSL Ciphers	System defaults
HTTP Strict Transport Security	□ Enable HTTP Strict Transport Security
TCP port	443
HTTP Redirect	□ Disable web GUI redirect rule
Login Messages	□ Disable logging of web GUI successful logins
Session Timeout	240

Protection plan evaluation

At the end of the configuration we are aware of two important points. The first is that the implemented configurations are the basis of at least sufficient protection from the most basic attacks, the most known vulnerabilities, and above all, allowing an easy and complete view of the current state of the sensitive hosts in the network, the actions they take and how these affect the entire system.

The second is that although the implementations made are sufficient for adequate protection, these may not be enough to deal with all the types of attacks today. For this reason, we have identified a number of technologies that, as future implementations, may be useful in improving the security management of the ACME network.

- Implementation of an **IPS (Intrusion Prevention System)** such as Suricata that would allow automatic actions to be taken upon detection of hybrid signature-based (i.e., known attack paths) and anomaly-based (where activities are analyzed and classified into normal and anomalous) cyber attacks. The implementation of this system, in conjunction with the Graylog log analysis server, allows the system to be continuously monitored and, in the case of anomalies, follow a series of preventive actions without human action (which may not be present in the case of an attack occurring during the night)

- Data **backup systems** for sensitive services. We all know how frustrating it can be to lose data, even more so in a business environment. One could think of implementing backup systems that would save important data from systems such as the webserver, proxy and fantasticcoffe and which, in addition, would act as a security support if it was implemented to save system logs giving the possibility to recognize any unauthorized changes to logs. The solution that we believe is most suitable for the system is that of a hybrid cloud between "local" and "remote" where the local cloud would maintain the highly critical data of the ACME network while the remote one, entrusted to a third-party company, would save all those less critical data and, for example, also the system logs in such a way that, in the event of a compromised network, in order to alter log backups, any users must also attack other systems
- outside the ACME network
- Still with regard to the evolution of the network and its security, one could think, in the case of possible growth of the network, the implementation of a **Bastian host** to replace (or support) one of the two firewalls or, in the case, as a security point of a possible new section of the subnet