Assignment 2 Report
ACME-6

Francesco Aquino (1851954),
Michele Kryston (1844733),
Ralph Angelo Tancio Almoneda (1837040),
Simone Di Valerio (1835412)

Practical Network Defense, Cybersecurity Master Degree
Sapienza University of Rome
Second Semester 2022/2023

Contents

1 Initial brainstorming

For the assignment we are asked to set a number of security policies regarding the use of services within the network and configure some of them. In particular we looked at how the **DNS server** should be set up and and how to set static DNS addresses to each individual host.

2 Evaluation of the security policy

Before we did anything, we read through all the rules that will need to be set so that we have the big picture.

Because many rules refer to multiple hosts located at different interfaces, to avoid repetition of rules we have grouped them together by defining a new group called **hosts**. The group was created in both firewalls and include interfaces corresponding to the networks of: **DMZ**, **external clients**, **servers** and **clients**. The VPN hosts (in our case Alice, Bob and Charles) were also considered as host, however it was not possible to put the OpenVPN interface in the group, so every rule in the **hosts** group was repeated in the OpenVPN interface.

OPNsense processes the rules in the groups and finally checks the rules in the interfaces, however some rules have to be put to a single interface and if the group's rules are processed first then any rules in the interface would be ignored. To solve it, every group's rules are set with **quick** unchecked while interface's rules are set with **quick** checked. In this way the rules in the interfaces will be processed on the first match, otherwise the last rule matched in the group would go.

Finally the last security policy asks to block anything not made explicit previously, so we decided to whitelist all security policies and block anything else for both incoming and outgoing packets. Incoming packets by default are whitelisted, so we only needed to choose what to do with outgoing packets. To avoid putting a rule in IPsec output for every rule in input in the interface, we let anything goes through IPsec. Anything going outside from the firewall through the WAN is allowed.

3 Policy implementation in opnsense

Since there are two firewalls, we had to add a **host** group for each firewall, but essentially they have the same rules. As mentioned earlier since we could not add OpenVPN to the group, we repeated the same rules as the **Host** group, except that we placed them in the reverse order and with **quick** checked.

All the rules were first implemented on *IPv4* and afterwards by simply changing the addresses to *IPv6* as well. Each rule that doesn't have any specific destination/source address was instead modified with *IPv4+IPv6* in the *field*. *IPv6* addresses were also added in the aliases along with *IPv4*. Has mentioned on the Assigment 1, *Graylog* and *Greenbone* don't have a *IPv6*'s address so their rule are only *IPv4*.

1. **All hosts must use the internal DNS Server as a DNS resolver.** First we have set the *DNS server* by following the video[1] provided by the teacher. Next we set the *DHCPv4 service* to also send the *DNS server* address in *Service⇒DHCPv4* only for the **Clients** and **Clients network** interfaces, all others had static IP's set directly in the configurations of the interfaces within the machine so they don't need it.

   We then modified the interface configuration by inserting the address of the DNS and finally installed *openresolv* so that the configuration would be maintained even after reboot. Static DNS addresses for Graylog were added in the configuration in *netplan*.

[1] Fonte: `https://www.youtube.com/watch?v=zGnpZnxWQ5c`

Host on the VPN have also been configured to automatically accept the DNS address by following a guide on Github[2]. We basically added *push "dhcp-option DNS 100.100.1.2"* in the OpenVPN configuration in *VPN⇒OpenVPN⇒Servers*, installed *openresolv* on the machine and appended the code written in Github on each OpenVPN client configuration.

Rules added:

- Let incoming packets on port 53 in **hosts group** (rule 4 Figure:7).
- Let outgoing packets on port 53 in **server network** (rule 8 Figure:8).

2. **Only the webserver service provided in the DMZ has to be accessible from the Internet.**
   We first disabled the anti-lockout in *Firewall⇒Settings⇒Advanced* because there was a rule in **DMZ** that was letting anything to port 80.

   Rules added:

   - Let incoming packets on port 80/443 to *webserver* in **WAN** (rule 6 Figure:4).
   - Let outgoing packets on port 80/443 to *webserver* in **DMZ** (rule 3 Figure:5).

3. **The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access.**
   Proxy server is already only reachable from internal hosts because by default from the **WAN** everything is blocked.

   Rules added:

   - Let incoming packets from *dns server* to outside in **DMZ** (rule 5 Figure:5).
   - Let incoming packets to *dns server* in **hosts group** (rule 4 Figure:7).
   - Let ougoing packets to *dns server* in **DMZ** (rule 7 Figure:5).

4. **Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.**
   We added a new aliases to group **DMZ** and **Clients network**.

   Rules added:

   - Let incoming packets to *servers network* **DMZ** (rule 9 Figure:5).
   - Let incoming packets to *servers network* **Clients** (rule 4 Figure:9).
   - Let ougoing packets from **clients** and **DMZ network** in **servers** (rule 10 Figure:8).

[2] Fonte: `https://github.com/alfredopalhares/openvpn-update-resolv-conf`

5. **All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.**
   We added a new aliases to group **Graylog** and **Logserver**. Since **DMZ** already has access to **servers network**4 there is no need to add a rule for **DMZ**. Hosts in **clients network** have access to **servers network** and therefore should be blocked if they try to connect to *Logserver* and *Graylog*. This latter rule must come before the pass rule or else would be ignored.

   Rules added:

   - Let incoming packets to *Logserver* and *Graylog* in **external clients network** (rule 3 Figure:6).
   - Let outgoing packets to *Logserver* and *Graylog* in **servers** (rule 7 Figure:8).
   - Block outgoing packets from **clients network** to *Logserver* and *Graylog* in **servers** (rule 5 Figure:8).

6. **The Greenbone server has to access all the hosts of the network.**
   Rules added:

   - Let incoming packets from *Greenbone* in **servers network** (rule 11 Figure:8).
   - Let outgoing packets from *Greenbone* in **hosts group** (rule 8 Figure:7).

7. **All network hosts have to be managed via ssh only from hosts within the Client network.**
   The last rule was added because the security policy5 otherwise did not allow access from **clients networks** to *Logserver* and *Greenbone*, so to let them have *SSH* we need to let the packets pass before the block from rule5.

   Rules added:

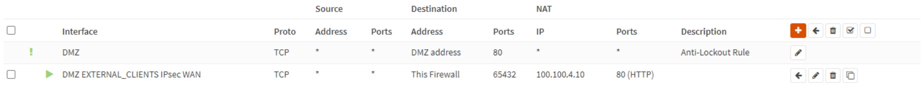
   - Let incoming packets on port 22 to **AMCE's network** in **clients network** (rule 6 Figure:9).
   - Let outgoing packets on port 22 from **clients network** in **hosts group** (rule 9 Figure:7).
   - Let outgoing packets on port 22 from **clients network** in **servers network** (rule 3 Figure:8).

8. **All the Client network hosts have only access to external web services (HTTP/HTTPS).**
   There's no need add an out rule in **WAN** because by default everything pass.

   Rules added:

- Let incoming packets on port 80/443 to outside in **clients network** (rule 8 Figure:9).

9. **Any packet received by the Main Firewall on port 65432 should be redirected to port 80 of the fantasticcoffee host.**
   We added a new rule in *Firewall⇒NAT⇒PortForwarding* to redirect these packets, the rule was applied in any interfaces. The only thing left was to let packets go to *fantasticcoffee*.



				Source		Destination		NAT			
☐		Interface	Proto	Address	Ports	Address	Ports	IP	Ports	Description	
	!	DMZ	TCP	*	*	DMZ address	80	*	*	Anti-Lockout Rule	
☐	▶	DMZ EXTERNAL_CLIENTS IPsec WAN	TCP	*	*	This Firewall	65432	100.100.4.10	80 (HTTP)		

Figure 1: Port forwarding rules.

   Rules added:

   - Let outgoing packets on port 80 to *fantasticcoffee* in **external clients network** (rule 4 Figure:6).

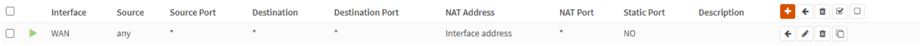
10. **The firewalls should protect against IP address spoofing.**
    On the **WAN**, **DMZ**, **external clients**, **clients**, and **servers network** interfaces, we added a rule that blocks any incoming packet with a wrong source IP. For example, the local **DMZ** network has IP *100.100.6.0/24*, so there should be no packets generated in the **DMZ** with an IP that doesn't belong to that subnet.

    Rules added:

    - Block incoming packets with source IP in *100.100.0.0/16* in **WAN** (rule 3 Figure:4).
    - Block incoming packets with source IP different from *100.100.6.0/24* in **DMZ** (rule 1 Figure:5).
    - Block incoming packets with source IP different from *100.100.4.0/24* in **external clients** (rule 1 Figure:6).
    - Block incoming packets with source IP different from *100.100.2.0/24* in **clients** (rule 2 Figure:9).
    - Block incoming packets with source IP different from *100.100.1.0/24* in **servers** (rule 1 Figure:8).
    - Block incoming packets with source IP different from *100.100.253.0/24* in **OpenVPN** (OpenVPN figure is the same as the Host's one).

11. **All the internal hosts should use the public IP address of the Main Firewall to exit towards the Internet.**

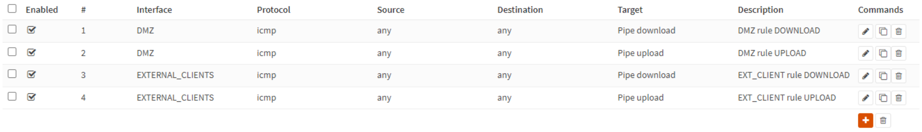
The rule was already active, so the security policy was already working and we didn't have to do anything.



□		Interface	Source	Source Port	Destination	Destination Port	NAT Address	NAT Port	Static Port	Description	
□	▶	WAN	any	*	*	*	Interface address	*	NO		

Figure 2: Port forwarding outbound rules in the **external firewall**.

12. **The rate of ICMP echo request packets should be limited to 10 Kbit/s.**
We followed the OPNsense's doc[3] to limit a bandwidth of a user and remodeled it to our use case. Basically we added two new *pipes* in *Firewall⇒Shaper⇒Pipes* with *10 kb/s* bandwidth (there's two *pipes* because the docs suggests it in order to avoid undefined behaviour). Then we added two rules per interface with those two pipes.



□	Enabled	#	Interface	Protocol	Source	Destination	Target	Description	Commands
□	☑	1	DMZ	icmp	any	any	Pipe download	DMZ rule DOWNLOAD	
□	☑	2	DMZ	icmp	any	any	Pipe upload	DMZ rule UPLOAD	
□	☑	3	EXTERNAL_CLIENTS	icmp	any	any	Pipe download	EXT_CLIENT rule DOWNLOAD	
□	☑	4	EXTERNAL_CLIENTS	icmp	any	any	Pipe upload	EXT_CLIENT rule UPLOAD	

Figure 3: Shaper rules in the **external firewall**.

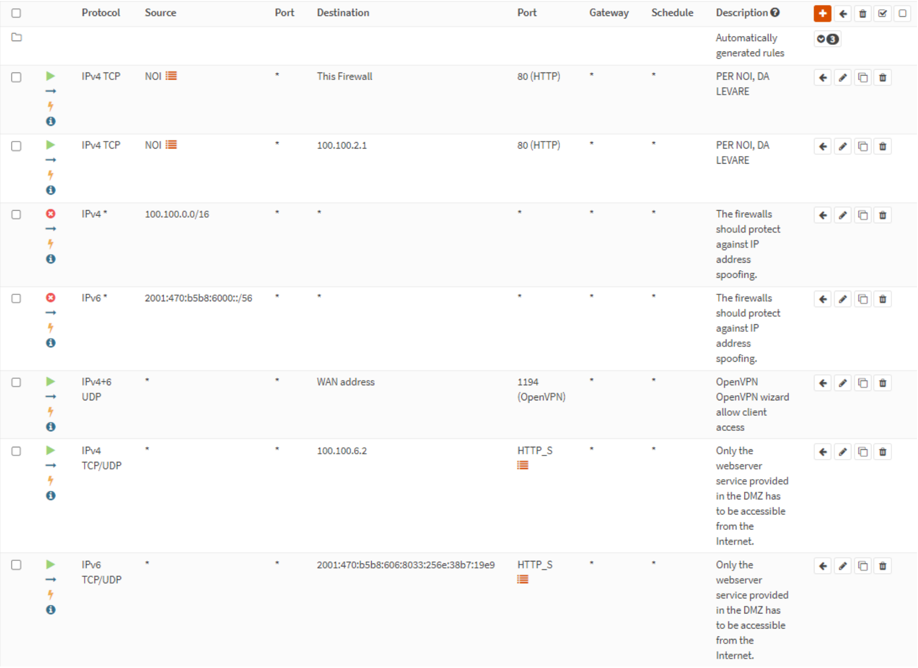
13. **Anything that is not explicitly allowed has to be denied.**
Any undefined incoming packet is blocked by a floating rule, it remains to block outgoing packets, this needs to be the last rule evaluated.

    Rule added:

    - Block any outgoing packets in **Hosts** (rule 1 Figure:7).

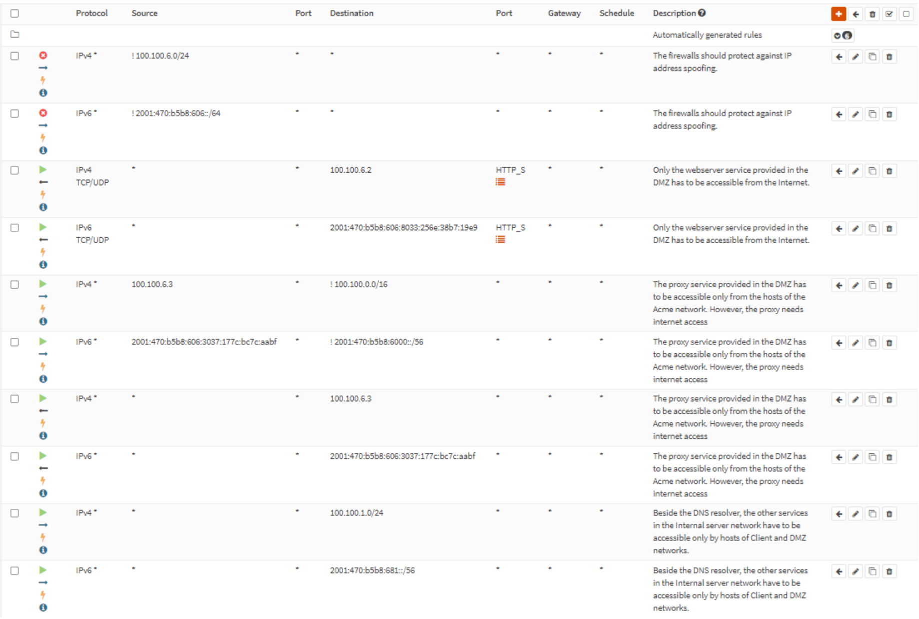
Finally, all the rules applied.

[3] https://docs.opnsense.org/manual/how-tos/shaper_limit_per_user.html



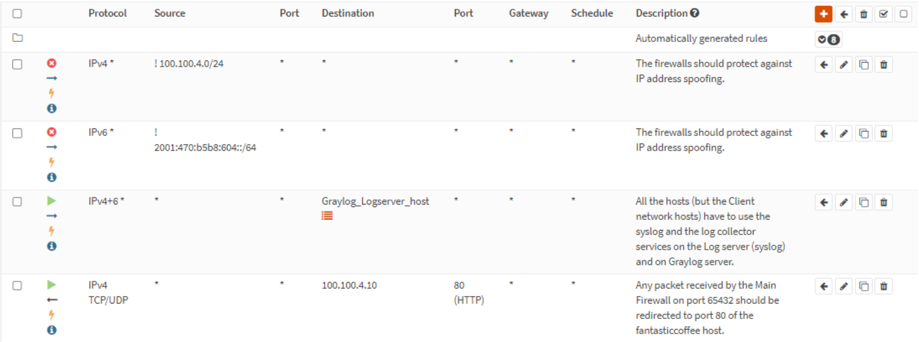
□		Protocol	Source	Port	Destination	Port	Gateway	Schedule	Description
									Automatically generated rules
□	▶	IPv4 TCP	NOI	*	This Firewall	80 (HTTP)	*	*	PER NOI, DA LEVARE
□	▶	IPv4 TCP	NOI	*	100.100.2.1	80 (HTTP)	*	*	PER NOI, DA LEVARE
□	✖	IPv4 *	100.100.0.0/16	*	*	*	*	*	The firewalls should protect against IP address spoofing.
□	✖	IPv6 *	2001:470:b5b8:6000::/56	*	*	*	*	*	The firewalls should protect against IP address spoofing.
□	▶	IPv4+6 UDP	*	*	WAN address	1194 (OpenVPN)	*	*	OpenVPN OpenVPN wizard allow client access
□	▶	IPv4 TCP/UDP	*	*	100.100.6.2	HTTP_S	*	*	Only the webserver service provided in the DMZ has to be accessible from the Internet.
□	▶	IPv6 TCP/UDP	*	*	2001:470:b5b8:606:8033:256e:38b7:19e9	HTTP_S	*	*	Only the webserver service provided in the DMZ has to be accessible from the Internet.

Figure 4: WAN's rules interface



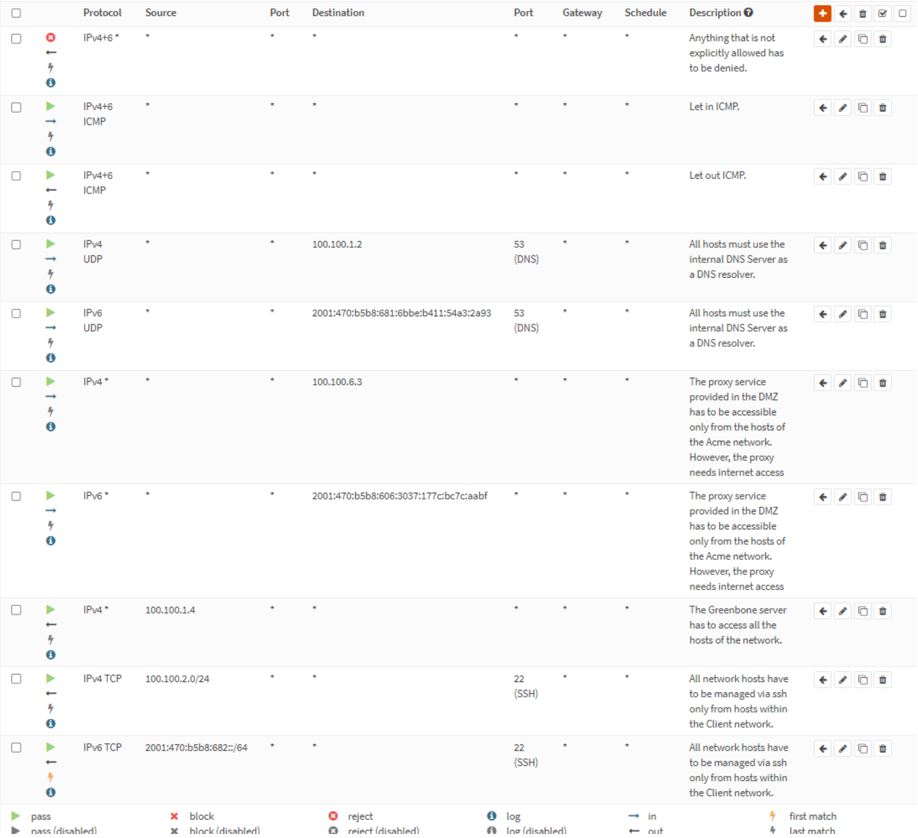
☐		Protocol	Source	Port	Destination	Port	Gateway	Schedule	Description
									Automatically generated rules
☐	⊗ →	IPv4 *	! 100.100.6.0/24	*	*	*	*	*	The firewalls should protect against IP address spoofing.
☐	⊗ →	IPv6 *	! 2001:470:b5b8:606::/64	*	*	*	*	*	The firewalls should protect against IP address spoofing.
☐	▶ ←	IPv4 TCP/UDP	*	*	100.100.6.2	HTTP_S	*	*	Only the webserver service provided in the DMZ has to be accessible from the Internet.
☐	▶ ←	IPv6 TCP/UDP	*	*	2001:470:b5b8:606:8033:256e:38b7:19e9	HTTP_S	*	*	Only the webserver service provided in the DMZ has to be accessible from the Internet.
☐	▶ →	IPv4 *	100.100.6.3	*	! 100.100.0.0/16	*	*	*	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
☐	▶ →	IPv6 *	2001:470:b5b8:606:3037:177c:bc7c:aabf	*	! 2001:470:b5b8:6000::/56	*	*	*	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
☐	▶ ←	IPv4 *	*	*	100.100.6.3	*	*	*	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
☐	▶ ←	IPv6 *	*	*	2001:470:b5b8:606:3037:177c:bc7c:aabf	*	*	*	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
☐	▶ →	IPv4 *	*	*	100.100.1.0/24	*	*	*	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.
☐	▶ →	IPv6 *	*	*	2001:470:b5b8:681::/56	*	*	*	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.

Figure 5: DMZ's rules interface



☐		Protocol	Source	Port	Destination	Port	Gateway	Schedule	Description
									Automatically generated rules
☐	⊗ →	IPv4 *	! 100.100.4.0/24	*	*	*	*	*	The firewalls should protect against IP address spoofing.
☐	⊗ →	IPv6 *	! 2001:470:b5b8:604::/64	*	*	*	*	*	The firewalls should protect against IP address spoofing.
☐	▶ →	IPv4+6 *	*	*	Graylog_Logserver_host	*	*	*	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.
☐	▶ ←	IPv4 TCP/UDP	*	*	100.100.4.10	80 (HTTP)	*	*	Any packet received by the Main Firewall on port 65432 should be redirected to port 80 of the fantasticcoffee host.

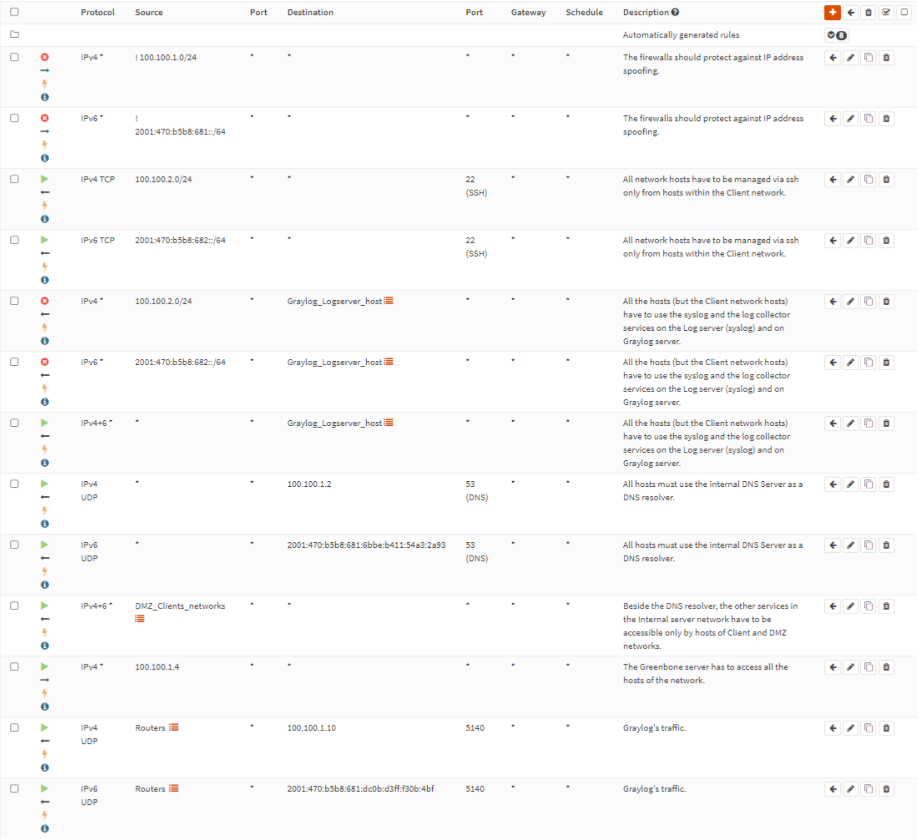
Figure 6: External client's rules interface



☐		Protocol	Source	Port	Destination	Port	Gateway	Schedule	Description
☐	⊗ ← ϟ ℹ	IPv4+6 *	*	*	*	*	*	*	Anything that is not explicitly allowed has to be denied.
☐	▶ → ϟ ℹ	IPv4+6 ICMP	*	*	*	*	*	*	Let in ICMP.
☐	▶ ← ϟ ℹ	IPv4+6 ICMP	*	*	*	*	*	*	Let out ICMP.
☐	▶ → ϟ ℹ	IPv4 UDP	*	*	100.100.1.2	53 (DNS)	*	*	All hosts must use the internal DNS Server as a DNS resolver.
☐	▶ → ϟ ℹ	IPv6 UDP	*	*	2001:470:b5b8:681:6bbe:b411:54a3:2a93	53 (DNS)	*	*	All hosts must use the internal DNS Server as a DNS resolver.
☐	▶ → ϟ ℹ	IPv4 *	*	*	100.100.6.3	*	*	*	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
☐	▶ → ϟ ℹ	IPv6 *	*	*	2001:470:b5b8:606:3037:177c:bc7c:aabf	*	*	*	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
☐	▶ ← ϟ ℹ	IPv4 *	100.100.1.4	*	*	*	*	*	The Greenbone server has to access all the hosts of the network.
☐	▶ ← ϟ ℹ	IPv4 TCP	100.100.2.0/24	*	*	22 (SSH)	*	*	All network hosts have to be managed via ssh only from hosts within the Client network.
☐	▶ ← ϟ ℹ	IPv6 TCP	2001:470:b5b8:682::/64	*	*	22 (SSH)	*	*	All network hosts have to be managed via ssh only from hosts within the Client network.

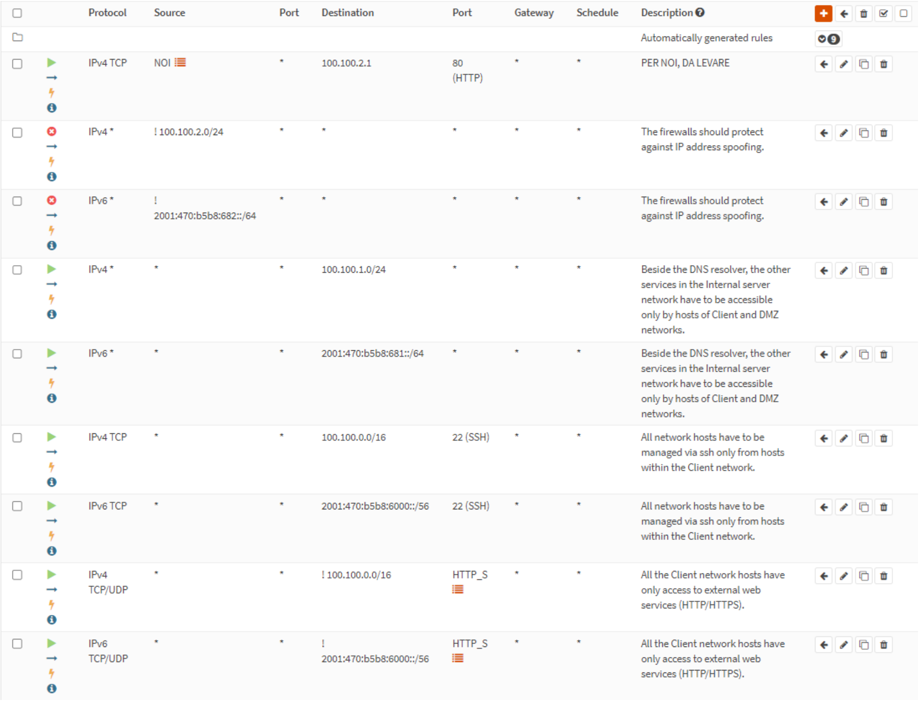
▶ pass ✖ block ⊗ reject ℹ log → in ϟ first match
▶ pass (disabled) ✖ block (disabled) ⊗ reject (disabled) ℹ log (disabled) ← out ϟ last match

Figure 7: Hosts's rules interface



☐		Protocol	Source	Port	Destination	Port	Gateway	Schedule	Description	+ ← 🗑 ☑ ☐
🗀									Automatically generated rules	⌄ 8
☐	⊗ → ⚡ ℹ	IPv4 *	! 100.100.1.0/24	*	*	*	*	*	The firewalls should protect against IP address spoofing.	← ✎ ⧉ 🗑
☐	⊗ → ⚡ ℹ	IPv6 *	! 2001:470:b5b8:681::/64	*	*	*	*	*	The firewalls should protect against IP address spoofing.	← ✎ ⧉ 🗑
☐	▶ ← ⚡ ℹ	IPv4 TCP	100.100.2.0/24	*	*	22 (SSH)	*	*	All network hosts have to be managed via ssh only from hosts within the Client network.	← ✎ ⧉ 🗑
☐	▶ ← ⚡ ℹ	IPv6 TCP	2001:470:b5b8:682::/64	*	*	22 (SSH)	*	*	All network hosts have to be managed via ssh only from hosts within the Client network.	← ✎ ⧉ 🗑
☐	⊗ ← ⚡ ℹ	IPv4 *	100.100.2.0/24	*	Graylog_Logserver_host ☰	*	*	*	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	← ✎ ⧉ 🗑
☐	⊗ ← ⚡ ℹ	IPv6 *	2001:470:b5b8:682::/64	*	Graylog_Logserver_host ☰	*	*	*	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	← ✎ ⧉ 🗑
☐	▶ ← ⚡ ℹ	IPv4+6 *	*	*	Graylog_Logserver_host ☰	*	*	*	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	← ✎ ⧉ 🗑
☐	▶ ← ⚡ ℹ	IPv4 UDP	*	*	100.100.1.2	53 (DNS)	*	*	All hosts must use the internal DNS Server as a DNS resolver.	← ✎ ⧉ 🗑
☐	▶ ← ⚡ ℹ	IPv6 UDP	*	*	2001:470:b5b8:681:6bbe:b411:54a3:2a93	53 (DNS)	*	*	All hosts must use the internal DNS Server as a DNS resolver.	← ✎ ⧉ 🗑
☐	▶ ← ⚡ ℹ	IPv4+6 *	DMZ_Clients_networks ☰	*	*	*	*	*	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.	← ✎ ⧉ 🗑
☐	▶ → ⚡ ℹ	IPv4 *	100.100.1.4	*	*	*	*	*	The Greenbone server has to access all the hosts of the network.	← ✎ ⧉ 🗑
☐	▶ ← ⚡ ℹ	IPv4 UDP	Routers ☰	*	100.100.1.10	5140	*	*	Graylog's traffic.	← ✎ ⧉ 🗑
☐	▶ ← ⚡ ℹ	IPv6 UDP	Routers ☰	*	2001:470:b5b8:681:dc0b:d3ff:f30b:4bf	5140	*	*	Graylog's traffic.	← ✎ ⧉ 🗑

Figure 8: Servers's rules interface



□		Protocol	Source	Port	Destination	Port	Gateway	Schedule	Description
									Automatically generated rules
□	▶	IPv4 TCP	NOI	*	100.100.2.1	80 (HTTP)	*	*	PER NOI, DA LEVARE
□	✖	IPv4 *	! 100.100.2.0/24	*	*	*	*	*	The firewalls should protect against IP address spoofing.
□	✖	IPv6 *	! 2001:470:b5b8:682::/64	*	*	*	*	*	The firewalls should protect against IP address spoofing.
□	▶	IPv4 *	*	*	100.100.1.0/24	*	*	*	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.
□	▶	IPv6 *	*	*	2001:470:b5b8:681::/64	*	*	*	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.
□	▶	IPv4 TCP	*	*	100.100.0.0/16	22 (SSH)	*	*	All network hosts have to be managed via ssh only from hosts within the Client network.
□	▶	IPv6 TCP	*	*	2001:470:b5b8:6000::/56	22 (SSH)	*	*	All network hosts have to be managed via ssh only from hosts within the Client network.
□	▶	IPv4 TCP/UDP	*	*	! 100.100.0.0/16	HTTP_S	*	*	All the Client network hosts have only access to external web services (HTTP/HTTPS).
□	▶	IPv6 TCP/UDP	*	*	! 2001:470:b5b8:6000::/56	HTTP_S	*	*	All the Client network hosts have only access to external web services (HTTP/HTTPS).

Figure 9: Clients's rules interface.

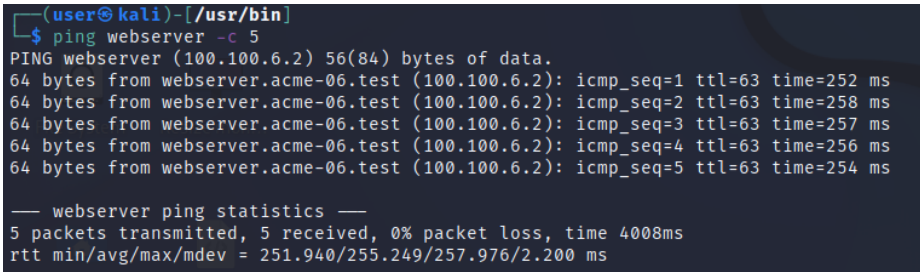
4 Test of the new configuration

For each rule we ran a test and to make sure that the rules were working well we enabled any logs for every rules and checked in *Firewall⇒Log files⇒Live view* that packets were being rejected or accepted by going through the right rules and not some other unexpected rules. In order to have better readability, **Ipsec** interfaces were omitted since every packets don't have any blocking rule.

Tests involving hosts outside the network were tested with the machine with IP *100.101.0.2* which is one of the VPN's IP given to us to do the homework. Most of the time, test were done with *netcat*, this because the ping does pass everywhere inside the network without problem.

We could not test with *fantasticcoffee* because the machine is not accessible.

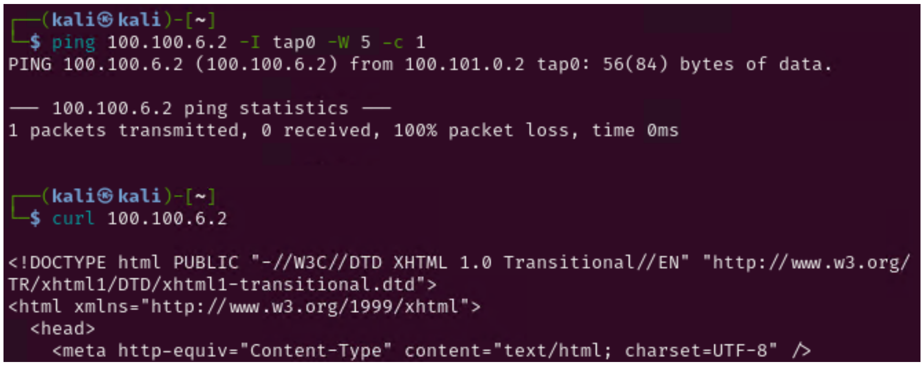
1. **Security Policy 1**
   With a script in each host we pinged all hosts in the network using the hostname instead of the IP.



```
┌──(user㉿kali)-[/usr/bin]
└─$ ping webserver -c 5
PING webserver (100.100.6.2) 56(84) bytes of data.
64 bytes from webserver.acme-06.test (100.100.6.2): icmp_seq=1 ttl=63 time=252 ms
64 bytes from webserver.acme-06.test (100.100.6.2): icmp_seq=2 ttl=63 time=258 ms
64 bytes from webserver.acme-06.test (100.100.6.2): icmp_seq=3 ttl=63 time=257 ms
64 bytes from webserver.acme-06.test (100.100.6.2): icmp_seq=4 ttl=63 time=256 ms
64 bytes from webserver.acme-06.test (100.100.6.2): icmp_seq=5 ttl=63 time=254 ms

--- webserver ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4008ms
rtt min/avg/max/mdev = 251.940/255.249/257.976/2.200 ms
```

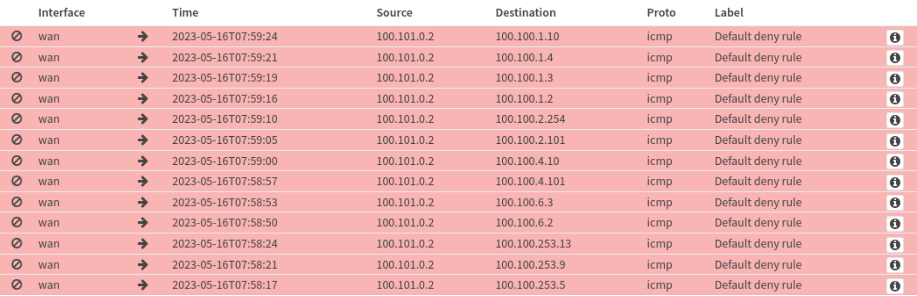
Figure 10: Ping test of one of *webserver* host.

2. **Security Policy 2**
   From the outside using *curl* we can get the *webserver* page without any problems, however you can't ping neither the *webserver* nor reach any other host in the network.



```
┌──(kali㉿kali)-[~]
└─$ ping 100.100.6.2 -I tap0 -W 5 -c 1
PING 100.100.6.2 (100.100.6.2) from 100.101.0.2 tap0: 56(84) bytes of data.

--- 100.100.6.2 ping statistics ---
1 packets transmitted, 0 received, 100% packet loss, time 0ms


┌──(kali㉿kali)-[~]
└─$ curl 100.100.6.2

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/
TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
```

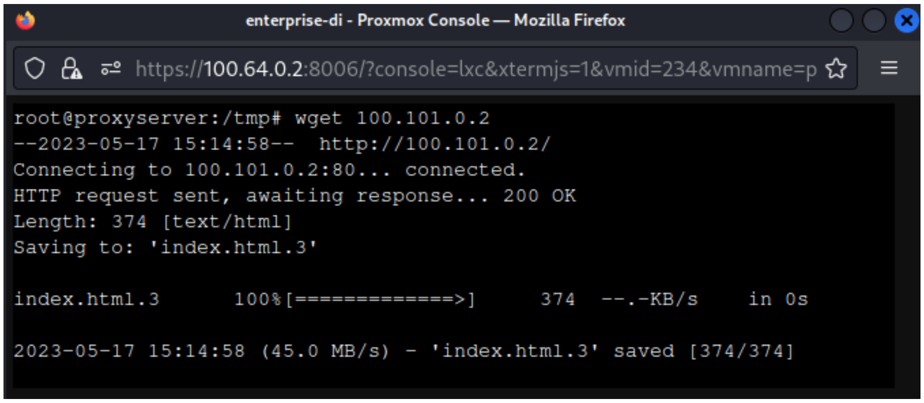
Figure 11: *Curl* goes without any problem but ping is timed out.



	Interface		Time	Source	Destination	Proto	Label	
⊘	wan	→	2023-05-16T07:59:24	100.101.0.2	100.100.1.10	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:59:21	100.101.0.2	100.100.1.4	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:59:19	100.101.0.2	100.100.1.3	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:59:16	100.101.0.2	100.100.1.2	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:59:10	100.101.0.2	100.100.2.254	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:59:05	100.101.0.2	100.100.2.101	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:59:00	100.101.0.2	100.100.4.10	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:58:57	100.101.0.2	100.100.4.101	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:58:53	100.101.0.2	100.100.6.3	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:58:50	100.101.0.2	100.100.6.2	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:58:24	100.101.0.2	100.100.253.13	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:58:21	100.101.0.2	100.100.253.9	icmp	Default deny rule	ℹ
⊘	wan	→	2023-05-16T07:58:17	100.101.0.2	100.100.253.5	icmp	Default deny rule	ℹ

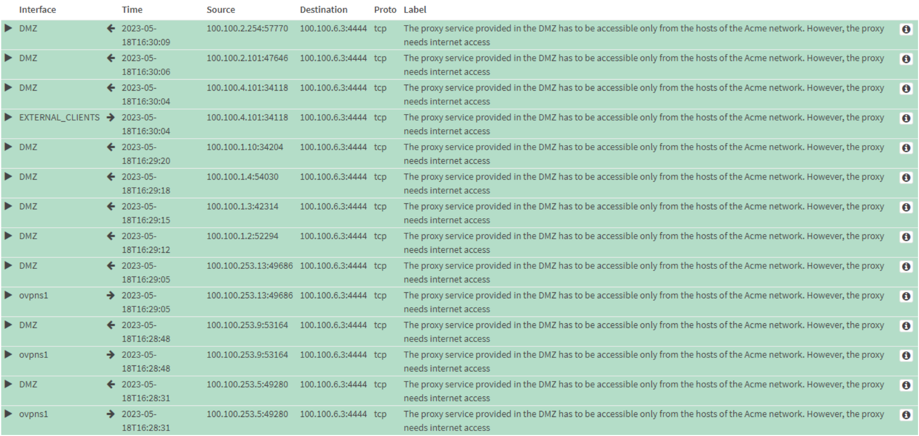
Figure 12: Live view of the **external firewall**. Anything that our host from the outside tries to ping is unreachable.

3. **Security Policy 3**
   We first tested if the *proxy server* has internet with *wget* and then if every hosts on the network can reach it with *netcat*.


enterprise-di - Proxmox Console — Mozilla Firefox
https://100.64.0.2:8006/?console=lxc&xtermjs=1&vmid=234&vmname=p

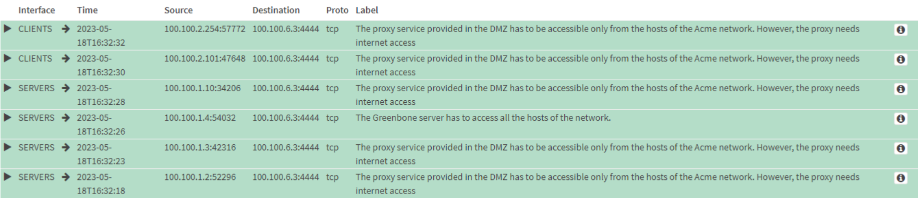
```
root@proxyserver:/tmp# wget 100.101.0.2
--2023-05-17 15:14:58--  http://100.101.0.2/
Connecting to 100.101.0.2:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 374 [text/html]
Saving to: 'index.html.3'

index.html.3      100%[=============>]     374  --.-KB/s    in 0s

2023-05-17 15:14:58 (45.0 MB/s) - 'index.html.3' saved [374/374]
```

Figure 13: *Proxy server* command promt, it can *wget* the *webserver* on our host without problem.



Interface		Time	Source	Destination	Proto	Label
DMZ	←	2023-05-18T16:30:09	100.100.2.254:57770	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
DMZ	←	2023-05-18T16:30:06	100.100.2.101:47646	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
DMZ	←	2023-05-18T16:30:04	100.100.4.101:34118	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
EXTERNAL_CLIENTS	→	2023-05-18T16:30:04	100.100.4.101:34118	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
DMZ	←	2023-05-18T16:29:20	100.100.1.10:34204	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
DMZ	←	2023-05-18T16:29:18	100.100.1.4:54030	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
DMZ	←	2023-05-18T16:29:15	100.100.1.3:42314	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
DMZ	←	2023-05-18T16:29:12	100.100.1.2:52294	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
DMZ	←	2023-05-18T16:29:05	100.100.253.13:49686	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
ovpns1	→	2023-05-18T16:29:05	100.100.253.13:49686	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
DMZ	←	2023-05-18T16:28:48	100.100.253.9:53164	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
ovpns1	→	2023-05-18T16:28:48	100.100.253.9:53164	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
DMZ	←	2023-05-18T16:28:31	100.100.253.5:49280	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
ovpns1	→	2023-05-18T16:28:31	100.100.253.5:49280	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access

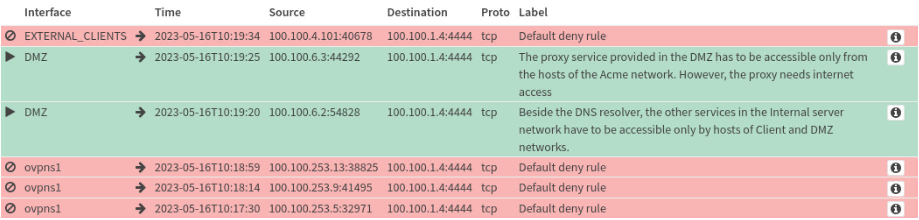
Figure 14: Live view of the **external firewall**. Any host can reach the *proxy server* without any problem.



Interface		Time	Source	Destination	Proto	Label
CLIENTS	→	2023-05-18T16:32:32	100.100.2.254:57772	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
CLIENTS	→	2023-05-18T16:32:30	100.100.2.101:47648	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
SERVERS	→	2023-05-18T16:32:28	100.100.1.10:34206	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
SERVERS	→	2023-05-18T16:32:26	100.100.1.4:54032	100.100.6.3:4444	tcp	The Greenbone server has to access all the hosts of the network.
SERVERS	→	2023-05-18T16:32:23	100.100.1.3:42316	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
SERVERS	→	2023-05-18T16:32:18	100.100.1.2:52296	100.100.6.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access

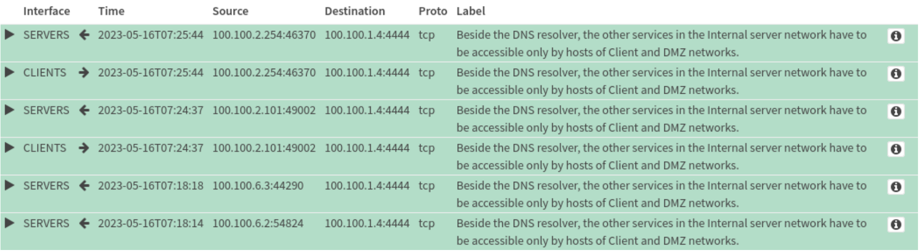
Figure 15: Live view of the **internal firewall**. Any host can reach the *proxy server* without any problem.

4. **Security Policy 4**
   To test this rule, we used *netcat* from different host to *Greenbone*. We selected *Greenbone* because due to other security policy, all of the other host in the **internal server network** are already accessible to some host of the network.



	Interface		Time	Source	Destination	Proto	Label	
⊘	EXTERNAL_CLIENTS	→	2023-05-16T10:19:34	100.100.4.101:40678	100.100.1.4:4444	tcp	Default deny rule	ⓘ
▶	DMZ	→	2023-05-16T10:19:25	100.100.6.3:44292	100.100.1.4:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access	ⓘ
▶	DMZ	→	2023-05-16T10:19:20	100.100.6.2:54828	100.100.1.4:4444	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.	ⓘ
⊘	ovpns1	→	2023-05-16T10:18:59	100.100.253.13:38825	100.100.1.4:4444	tcp	Default deny rule	ⓘ
⊘	ovpns1	→	2023-05-16T10:18:14	100.100.253.9:41495	100.100.1.4:4444	tcp	Default deny rule	ⓘ
⊘	ovpns1	→	2023-05-16T10:17:30	100.100.253.5:32971	100.100.1.4:4444	tcp	Default deny rule	ⓘ

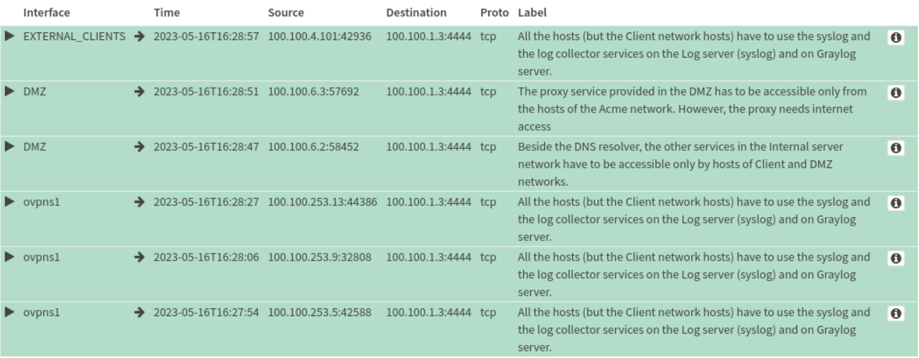
Figure 16: Live view of the **external firewall**. **VPN** and **external client** host are blocked.



	Interface		Time	Source	Destination	Proto	Label	
▶	SERVERS	←	2023-05-16T07:25:44	100.100.2.254:46370	100.100.1.4:4444	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.	ⓘ
▶	CLIENTS	→	2023-05-16T07:25:44	100.100.2.254:46370	100.100.1.4:4444	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.	ⓘ
▶	SERVERS	←	2023-05-16T07:24:37	100.100.2.101:49002	100.100.1.4:4444	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.	ⓘ
▶	CLIENTS	→	2023-05-16T07:24:37	100.100.2.101:49002	100.100.1.4:4444	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.	ⓘ
▶	SERVERS	←	2023-05-16T07:18:18	100.100.6.3:44290	100.100.1.4:4444	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.	ⓘ
▶	SERVERS	←	2023-05-16T07:18:14	100.100.6.2:54824	100.100.1.4:4444	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.	ⓘ

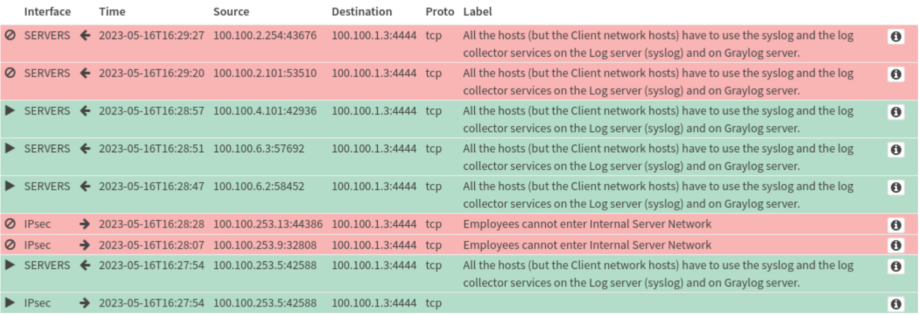
Figure 17: Live view of the **internal firewall**. Everything pass without problem to the **internal servers network**.

5. **Security Policy 5**
   We basically performed the same test as the previous one but this time we used the *Logserver*. **IPsec** interface is not omitted because some packets are blocked in the interface, this because in the first assignment we setted that Bob and Charles cannot reach the **internal Network**.



	Interface		Time	Source	Destination	Proto	Label	
▶	EXTERNAL_CLIENTS	→	2023-05-16T16:28:57	100.100.4.101:42936	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ
▶	DMZ	→	2023-05-16T16:28:51	100.100.6.3:57692	100.100.1.3:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access	ℹ
▶	DMZ	→	2023-05-16T16:28:47	100.100.6.2:58452	100.100.1.3:4444	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.	ℹ
▶	ovpns1	→	2023-05-16T16:28:27	100.100.253.13:44386	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ
▶	ovpns1	→	2023-05-16T16:28:06	100.100.253.9:32808	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ
▶	ovpns1	→	2023-05-16T16:27:54	100.100.253.5:42588	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ

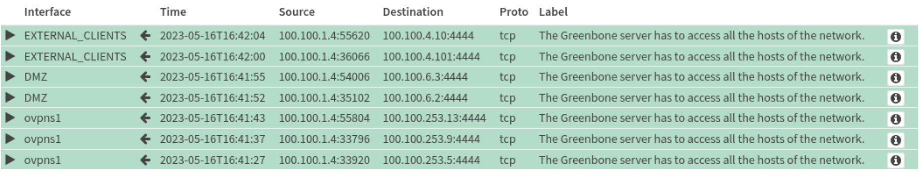
Figure 18: Live view of the **external firewall**. *Netcat* pass without any problem.



	Interface		Time	Source	Destination	Proto	Label	
⊘	SERVERS	←	2023-05-16T16:29:27	100.100.2.254:43676	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ
⊘	SERVERS	←	2023-05-16T16:29:20	100.100.2.101:53510	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ
▶	SERVERS	←	2023-05-16T16:28:57	100.100.4.101:42936	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ
▶	SERVERS	←	2023-05-16T16:28:51	100.100.6.3:57692	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ
▶	SERVERS	←	2023-05-16T16:28:47	100.100.6.2:58452	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ
⊘	IPsec	→	2023-05-16T16:28:28	100.100.253.13:44386	100.100.1.3:4444	tcp	Employees cannot enter Internal Server Network	ℹ
⊘	IPsec	→	2023-05-16T16:28:07	100.100.253.9:32808	100.100.1.3:4444	tcp	Employees cannot enter Internal Server Network	ℹ
▶	SERVERS	←	2023-05-16T16:27:54	100.100.253.5:42588	100.100.1.3:4444	tcp	All the hosts (but the Client network hosts) have to use the syslog and the log collector services on the Log server (syslog) and on Graylog server.	ℹ
▶	IPsec	→	2023-05-16T16:27:54	100.100.253.5:42588	100.100.1.3:4444	tcp		ℹ

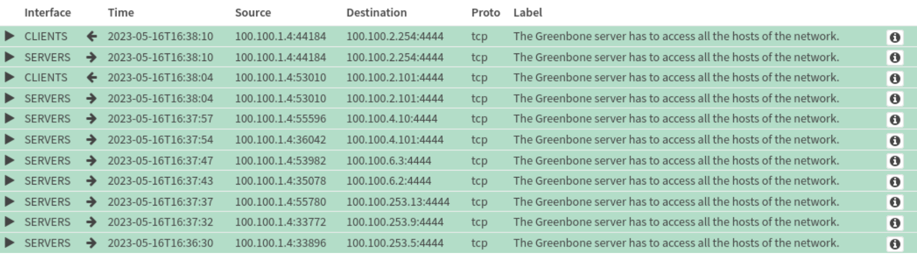
Figure 19: Live view of the *internal firewall.* The packets that are being blocked are from **clients network** and the two *Employees* of the **OpenVPN**.

6. **Security Policy 6**
   For this test we used *netcat* from *Greenbone* to the other hosts.



	Interface		Time	Source	Destination	Proto	Label	
▶	EXTERNAL_CLIENTS	←	2023-05-16T16:42:04	100.100.1.4:55620	100.100.4.10:4444	tcp	The Greenbone server has to access all the hosts of the network.	ℹ
▶	EXTERNAL_CLIENTS	←	2023-05-16T16:42:00	100.100.1.4:36066	100.100.4.101:4444	tcp	The Greenbone server has to access all the hosts of the network.	ℹ
▶	DMZ	←	2023-05-16T16:41:55	100.100.1.4:54006	100.100.6.3:4444	tcp	The Greenbone server has to access all the hosts of the network.	ℹ
▶	DMZ	←	2023-05-16T16:41:52	100.100.1.4:35102	100.100.6.2:4444	tcp	The Greenbone server has to access all the hosts of the network.	ℹ
▶	ovpns1	←	2023-05-16T16:41:43	100.100.1.4:55804	100.100.253.13:4444	tcp	The Greenbone server has to access all the hosts of the network.	ℹ
▶	ovpns1	←	2023-05-16T16:41:37	100.100.1.4:33796	100.100.253.9:4444	tcp	The Greenbone server has to access all the hosts of the network.	ℹ
▶	ovpns1	←	2023-05-16T16:41:27	100.100.1.4:33920	100.100.253.5:4444	tcp	The Greenbone server has to access all the hosts of the network.	ℹ

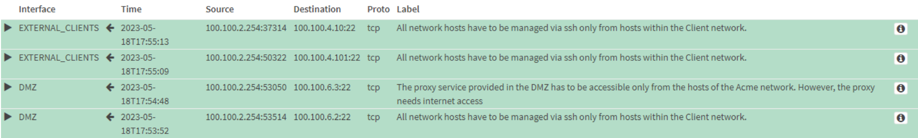
Figure 20: Live view of the **external firewall**. *Netcat* goes without any problem.



	Interface		Time	Source	Destination	Proto	Label
▶	CLIENTS	←	2023-05-16T16:38:10	100.100.1.4:44184	100.100.2.254:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	SERVERS	→	2023-05-16T16:38:10	100.100.1.4:44184	100.100.2.254:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	CLIENTS	←	2023-05-16T16:38:04	100.100.1.4:53010	100.100.2.101:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	SERVERS	→	2023-05-16T16:38:04	100.100.1.4:53010	100.100.2.101:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	SERVERS	→	2023-05-16T16:37:57	100.100.1.4:55596	100.100.4.10:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	SERVERS	→	2023-05-16T16:37:54	100.100.1.4:36042	100.100.4.101:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	SERVERS	→	2023-05-16T16:37:47	100.100.1.4:53982	100.100.6.3:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	SERVERS	→	2023-05-16T16:37:43	100.100.1.4:35078	100.100.6.2:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	SERVERS	→	2023-05-16T16:37:37	100.100.1.4:55780	100.100.253.13:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	SERVERS	→	2023-05-16T16:37:32	100.100.1.4:33772	100.100.253.9:4444	tcp	The Greenbone server has to access all the hosts of the network.
▶	SERVERS	→	2023-05-16T16:36:30	100.100.1.4:33896	100.100.253.5:4444	tcp	The Greenbone server has to access all the hosts of the network.

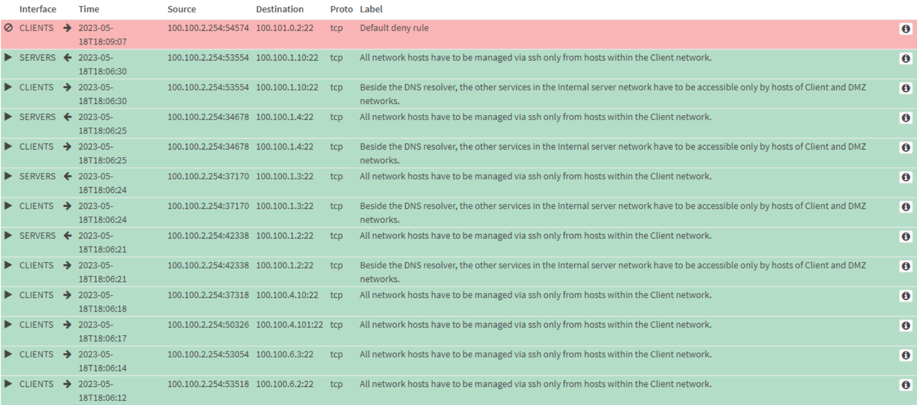
Figure 21: Live view of the **internal firewall**. *Netcat* goes without any problem.

7. **Security Policy 7**
   We test if a host in *clients network* (in this case we used *arpwatch*) can connect with *SSH* to every host inside the network.



	Interface		Time	Source	Destination	Proto	Label
▶	EXTERNAL_CLIENTS	←	2023-05-18T17:55:13	100.100.2.254:37314	100.100.4.10:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.
▶	EXTERNAL_CLIENTS	←	2023-05-18T17:55:09	100.100.2.254:50322	100.100.4.101:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.
▶	DMZ	←	2023-05-18T17:54:48	100.100.2.254:53050	100.100.6.3:22	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access
▶	DMZ	←	2023-05-18T17:53:52	100.100.2.254:53514	100.100.6.2:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.

Figure 22: Live view of the **external firewall**. *SSH* goes without any problem.

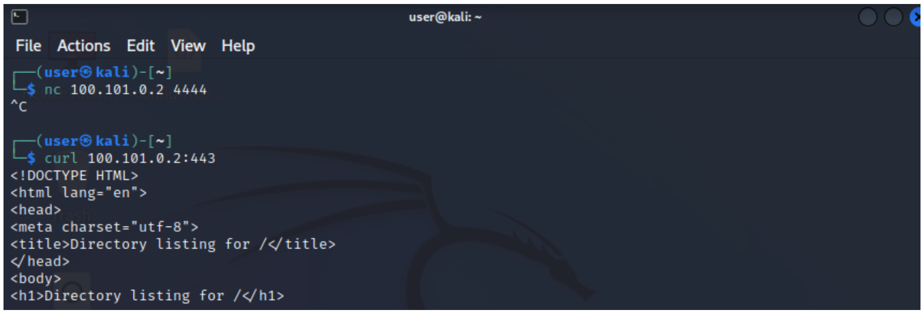


	Interface		Time	Source	Destination	Proto	Label
⊘	CLIENTS	→	2023-05-18T18:09:07	100.100.2.254:54574	100.101.0.2:22	tcp	Default deny rule
▶	SERVERS	←	2023-05-18T18:06:30	100.100.2.254:53554	100.100.1.10:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.
▶	CLIENTS	→	2023-05-18T18:06:30	100.100.2.254:53554	100.100.1.10:22	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.
▶	SERVERS	←	2023-05-18T18:06:25	100.100.2.254:34678	100.100.1.4:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.
▶	CLIENTS	→	2023-05-18T18:06:25	100.100.2.254:34678	100.100.1.4:22	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.
▶	SERVERS	←	2023-05-18T18:06:24	100.100.2.254:37170	100.100.1.3:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.
▶	CLIENTS	→	2023-05-18T18:06:24	100.100.2.254:37170	100.100.1.3:22	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.
▶	SERVERS	←	2023-05-18T18:06:21	100.100.2.254:42338	100.100.1.2:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.
▶	CLIENTS	→	2023-05-18T18:06:21	100.100.2.254:42338	100.100.1.2:22	tcp	Beside the DNS resolver, the other services in the Internal server network have to be accessible only by hosts of Client and DMZ networks.
▶	CLIENTS	→	2023-05-18T18:06:18	100.100.2.254:37318	100.100.4.10:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.
▶	CLIENTS	→	2023-05-18T18:06:17	100.100.2.254:50326	100.100.4.101:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.
▶	CLIENTS	→	2023-05-18T18:06:14	100.100.2.254:53054	100.100.6.3:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.
▶	CLIENTS	→	2023-05-18T18:06:12	100.100.2.254:53518	100.100.6.2:22	tcp	All network hosts have to be managed via ssh only from hosts within the Client network.

Figure 23: Live view of the **internal firewall**. *SSH* goes without any problem except for the one that tries to connect to an outside host.

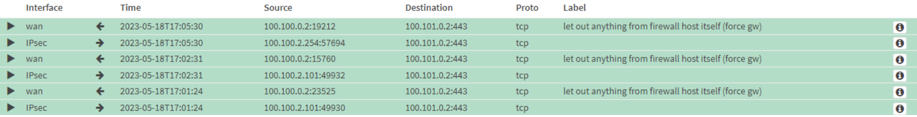
8. **Security Policy 8**

We tested *netcat* and *netcat* from the **clients network** to our host, there's only two hosts which are *Kali* and *Arpwatch*. The **IPsec** interface this time was not omitted.



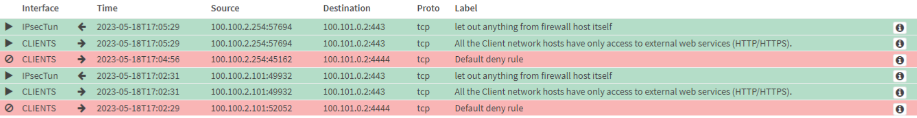
```
┌──(user㉿kali)-[~]
└─$ nc 100.101.0.2 4444
^C

┌──(user㉿kali)-[~]
└─$ curl 100.101.0.2:443
<!DOCTYPE HTML>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Directory listing for /</title>
</head>
<body>
<h1>Directory listing for /</h1>
```

Figure 24: Command line of the *Kali* host. *Netcat* is blocked while *curl* goes without any problem.



	Interface		Time	Source	Destination	Proto	Label	
▶	wan	←	2023-05-18T17:05:30	100.100.0.2:19212	100.101.0.2:443	tcp	let out anything from firewall host itself (force gw)	ⓘ
▶	IPsec	→	2023-05-18T17:05:30	100.100.2.254:57694	100.101.0.2:443	tcp		ⓘ
▶	wan	←	2023-05-18T17:02:31	100.100.0.2:15760	100.101.0.2:443	tcp	let out anything from firewall host itself (force gw)	ⓘ
▶	IPsec	→	2023-05-18T17:02:31	100.100.2.101:49932	100.101.0.2:443	tcp		ⓘ
▶	wan	←	2023-05-18T17:01:24	100.100.0.2:23525	100.101.0.2:443	tcp	let out anything from firewall host itself (force gw)	ⓘ
▶	IPsec	→	2023-05-18T17:01:24	100.100.2.101:49930	100.101.0.2:443	tcp		ⓘ

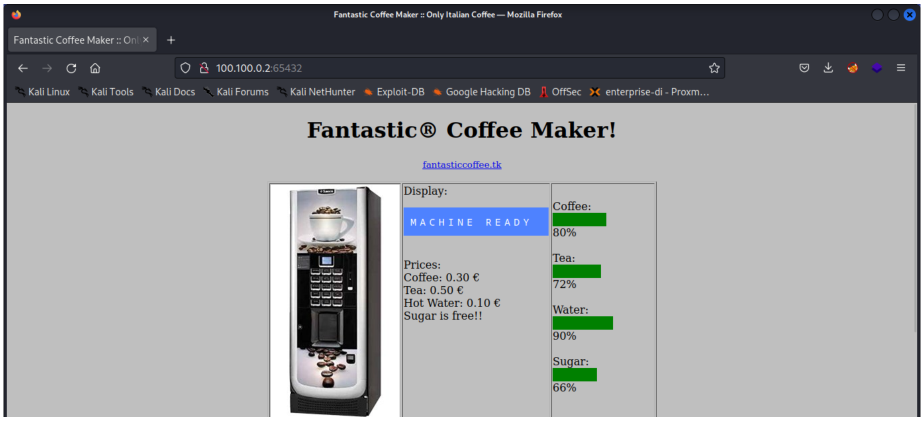
Figure 25: Live view of the **external firewall**.



	Interface		Time	Source	Destination	Proto	Label	
▶	IPsecTun	←	2023-05-18T17:05:29	100.100.2.254:57694	100.101.0.2:443	tcp	let out anything from firewall host itself	ⓘ
▶	CLIENTS	→	2023-05-18T17:05:29	100.100.2.254:57694	100.101.0.2:443	tcp	All the Client network hosts have only access to external web services (HTTP/HTTPS).	ⓘ
⊘	CLIENTS	→	2023-05-18T17:04:56	100.100.2.254:45162	100.101.0.2:4444	tcp	Default deny rule	ⓘ
▶	IPsecTun	←	2023-05-18T17:02:31	100.100.2.101:49932	100.101.0.2:443	tcp	let out anything from firewall host itself	ⓘ
▶	CLIENTS	→	2023-05-18T17:02:31	100.100.2.101:49932	100.101.0.2:443	tcp	All the Client network hosts have only access to external web services (HTTP/HTTPS).	ⓘ
⊘	CLIENTS	→	2023-05-18T17:02:29	100.100.2.101:52052	100.101.0.2:4444	tcp	Default deny rule	ⓘ

Figure 26: Live view of the **internal firewall**.

9. **Security Policy 9**
   From our machine we tried to access using *100.100.0.2:65432* as the link. We cannot see that our host tried to use port 65432 because I think that the *NAT* is applied before the rules are evaluated.



Fantastic® Coffee Maker!

fantasticcoffee.tk

Display:

MACHINE READY

Prices:
Coffee: 0.30 €
Tea: 0.50 €
Hot Water: 0.10 €
Sugar is free!!

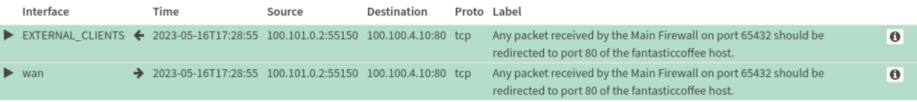
Coffee:
80%

Tea:
72%

Water:
90%

Sugar:
66%

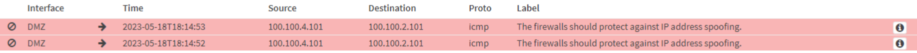
Figure 27: Page from the browser.



	Interface		Time	Source	Destination	Proto	Label
▶	EXTERNAL_CLIENTS	←	2023-05-16T17:28:55	100.101.0.2:55150	100.100.4.10:80	tcp	Any packet received by the Main Firewall on port 65432 should be redirected to port 80 of the fantasticcoffee host.
▶	wan	→	2023-05-16T17:28:55	100.101.0.2:55150	100.100.4.10:80	tcp	Any packet received by the Main Firewall on port 65432 should be redirected to port 80 of the fantasticcoffee host.

Figure 28: Live view of the **external firewall**.

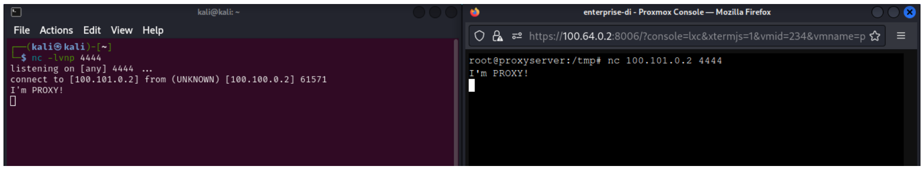
10. **Security Policy 10**
    We tried changing the *webserver* ip to *100.100.4.101* and sending ping to another host.



	Interface		Time	Source	Destination	Proto	Label
⊘	DMZ	→	2023-05-18T18:14:53	100.100.4.101	100.100.2.101	icmp	The firewalls should protect against IP address spoofing.
⊘	DMZ	→	2023-05-18T18:14:52	100.100.4.101	100.100.2.101	icmp	The firewalls should protect against IP address spoofing.

Figure 29: Live view of the **external firewall**. Rule block the IP-spoofing attempt.

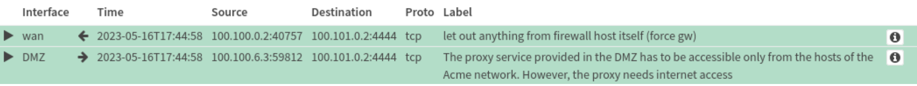
11. **Security Policy 11**
    For this test we used the *proxy server* because is the only host with the complete access to the internet. We placed a listener on our host and waited for the connection.


kali@kali: ~
File Actions Edit View Help
enterprise-di - Proxmox Console — Mozilla Firefox
https://100.64.0.2:8006/?console=lxc&xtermjs=1&vmid=234&vmname=p

```
┌──(kali㉿kali)-[~]
└─$ nc -lvnp 4444
listening on [any] 4444 ...
connect to [100.101.0.2] from (UNKNOWN) [100.100.0.2] 61571
I'm PROXY!
```

```
root@proxyserver:/tmp# nc 100.101.0.2 4444
I'm PROXY!
```

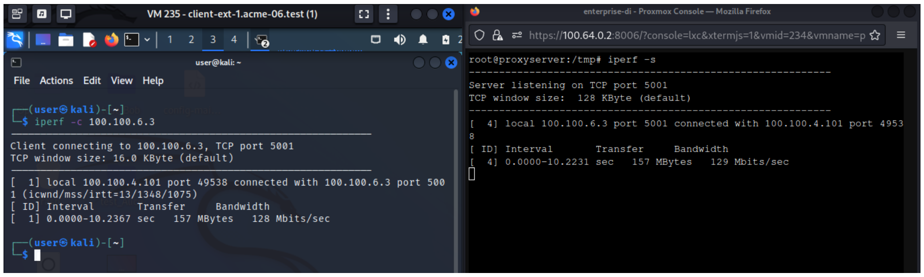
Figure 30: On the left we have our host, while on the right we have the *proxy server*. Even if we connected using the *proxy server*, the IP that is displayed is the firewall's IP.



	Interface		Time	Source	Destination	Proto	Label	
▶	wan	←	2023-05-16T17:44:58	100.100.0.2:40757	100.101.0.2:4444	tcp	let out anything from firewall host itself (force gw)	ℹ
▶	DMZ	→	2023-05-16T17:44:58	100.100.6.3:59812	100.101.0.2:4444	tcp	The proxy service provided in the DMZ has to be accessible only from the hosts of the Acme network. However, the proxy needs internet access	ℹ

Figure 31: Live view of the **external firewall**. From here you can see that the source IP has been replaced with the firewall IP.

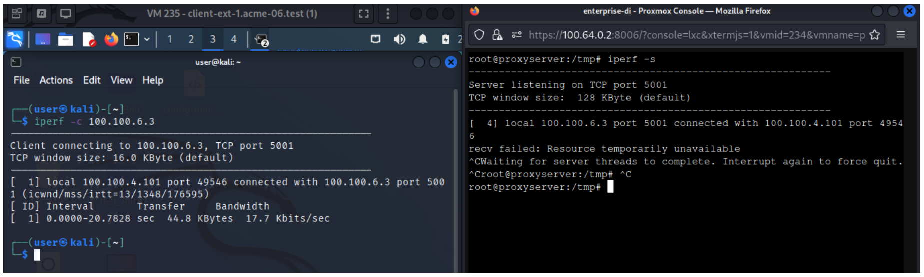
12. **Security Policy 12**
    For testing, we used *Iperf*, a tool for measuring bandwidth between two hosts. *Iperf* doesn't have a specific command to test only *ICMP* packets so we changed the rule to filter all of the IP packets (instead of only *ICMP*) in *Firewall*⇒*Shaper*⇒*Rules*.


VM 235 - client-ext-1.acme-06.test (1)
1
2
3
4
user@kali: ~
File Actions Edit View Help
enterprise-di - Proxmox Console — Mozilla Firefox
https://100.64.0.2:8006/?console=lxc&xtermjs=1&vmid=234&vmname=p

```
┌──(user㉿kali)-[~]
└─$ iperf -c 100.100.6.3
------------------------------------------------------------
Client connecting to 100.100.6.3, TCP port 5001
TCP window size: 16.0 KByte (default)
------------------------------------------------------------
[  1] local 100.100.4.101 port 49538 connected with 100.100.6.3 port 500
1 (icwnd/mss/irtt=13/1348/1075)
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-10.2367 sec   157 MBytes   128 Mbits/sec

┌──(user㉿kali)-[~]
└─$ 
```

```
root@proxyserver:/tmp# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 100.100.6.3 port 5001 connected with 100.100.4.101 port 4953
8
[ ID] Interval       Transfer     Bandwidth
[  4] 0.0000-10.2231 sec   157 MBytes   129 Mbits/sec
```

Figure 32: On the left the client, on the right the server. This is was before we applied the filter.


VM 235 - client-ext-1.acme-06.test (1)
user@kali: ~
File Actions Edit View Help
enterprise-di - Proxmox Console — Mozilla Firefox
https://100.64.0.2:8006/?console=lxc&xtermjs=1&vmid=234&vmname=p

```
┌──(user㉿kali)-[~]
└─$ iperf -c 100.100.6.3
------------------------------------------------------------
Client connecting to 100.100.6.3, TCP port 5001
TCP window size: 16.0 KByte (default)
------------------------------------------------------------
[  1] local 100.100.4.101 port 49546 connected with 100.100.6.3 port 500
1 (icwnd/mss/irtt=13/1348/176595)
[ ID] Interval       Transfer     Bandwidth
[  1] 0.0000-20.7828 sec  44.8 KBytes  17.7 Kbits/sec

┌──(user㉿kali)-[~]
└─$ 
```

```
root@proxyserver:/tmp# iperf -s
------------------------------------------------------------
Server listening on TCP port 5001
TCP window size:  128 KByte (default)
------------------------------------------------------------
[  4] local 100.100.6.3 port 5001 connected with 100.100.4.101 port 4954
6
recv failed: Resource temporarily unavailable
^CWaiting for server threads to complete. Interrupt again to force quit.
^Croot@proxyserver:/tmp# ^C
root@proxyserver:/tmp# 
```

Figure 33: On the left the client, on the right the server. This is was after we applied the filter, we can see that the bandwidth has decreased significantly from before.

13. **Security Policy 13**

    As for the last security policy, which requires all packet not included in the previous rules must be blocked, we can state that this requirement is already satisfied otherwise, in the previous tests, all these packets should have been passed. However we verified that are all blocked.

5 Final remarks

The rules were initially implemented individually for each interface. However, we opted for a more efficient solution. In order to reduce redundancy by repeating the same multiple times we decided to group the rules and use the group in both firewalls, including the interfaces corresponding to the networks in the topology.

Another aspect to take into account is the in which order to apply the rules. This was a sensitive task because we needed to define the correct order to apply the rules in order to prevent any conflict between them and avoid packets to be treated in the wrong way.

We used Live View to check how the rules performed in our firewalls. It was not immediate to understand how Live View behaves but we figured that we needed to activate the logs on each rule we added. The output shows how packets are treated by our firewalls. We used labels to identify the flow of the packets and understand which rule it matched and its output: the green ones are accepted and the red ones are blocked.