

Practical Network Defense - Lab 4

Alessandro Serpi - 1647244

22 March 2019

Contents

1	Introduction	2
2	Setup of the infrastructure	2
2.1	Disabling checksum offload	2
2.2	Enabling recursive DNS queries	2
2.3	Creating and configuring the VMs	2
2.4	Starting the lab	2
2.5	Configuring the interfaces	3
2.6	Setting up OPNsense	3
3	Evaluation of the security policy	3
3.1	DNS	3
3.2	FTP	3
3.3	HTTP/HTTPS	3
3.4	SSH	4
3.5	Syslog	4
4	Policy implementation in OPNsense	4
4.1	Default policy	4
4.2	Address spoofing	4
4.3	Aliases	4
4.4	Packets arriving to <i>mainfw</i> from...	5
4.4.1	... <i>DMZ</i>	5
4.4.2	...outside	5
4.4.3	... <i>intfw</i>	6
4.5	Packets arriving to <i>intfw</i> from ...	6
4.5.1	... <i>mainfw</i>	6
4.5.2	...the <i>client</i> network	7
4.5.3	...the <i>internal server</i> network	7
5	Test of the configuration	7

1 Introduction

The private network of *ACME co.* is composed by four subnetworks living in the shared address space `100.64.0.0/16`. *DMZ* is connected directly to the main firewall-router *mainfw* and offers services accessible from the external network, while *client* and *internal server* networks are behind a second line of defence (represented by the internal firewall-router *intfw*). The latter offers services that may be used only by hosts in the private network, whereas the former does not offer any services.

All hosts are already configured (however, minor adjustment may be made), with the exception of the firewalls, that have yet to be created.

2 Setup of the infrastructure

2.1 Disabling checksum offload

The command `ethtool -K eth0 tx off` was added to every `.startup` file. This disables the checksum computation offload to the network card; instead, the hosts rely on the CPU to calculate the packet checksums. It is slower, but creates less problems with OPNsense. In OPNsense, checksum offload is disabled by default.

2.2 Enabling recursive DNS queries

Recursive DNS queries are not permitted, but this clashes with the fact that client hosts are allowed to retrieve web pages from the internet. Therefore, it has been decided to allow recursive DNS queries adding the clause `recursion yes;` in `dns/etc/named.conf`, section `options`.

It has also decided that hosts in the *DMZ* and in the *internal server* networks should not be interested in retrieving IP addresses of external websites. Hence, recursive DNS queries have been allowed only for *client* network.

2.3 Creating and configuring the VMs

Create in VirtualBox two virtual machine, one for *mainfw* and the other for *intfw*, each with its own disk. Download the ISO installer from OPNsense's [official website](#) and add it to the virtual machines. Install OPNsense in both machine, then power them off and remove the ISO. Finally, add to *mainfw* one NAT network card and two bridge adapters and add to *intfw* three bridge adapters.

2.4 Starting the lab

Every time it is necessary to launch the lab, use `lstart` to initiate Kathará and then execute `start_vms.sh`. The script automatically configures the virtual network cards

and powers up the virtual machines.

2.5 Configuring the interfaces

Power up both virtual machine, login as root and enter interface assignment menu. Use `em0` as *WAN*, `em1` as *OPT1* and `em2` as *LAN*.

Enter the IP address configuration menu. In *mainfw*, configure *WAN* to use DHCP, assign 100.64.254.1/30 to *LAN* and 100.64.6.1/24 to *OPT1*; in *intfw*, assign address 100.64.254.2/30 and gateway 100.64.254.1 to *WAN*, address 100.64.2.1/24 to *LAN* and 100.64.1.1/24 to *OPT1*.

2.6 Setting up OPNsense

In *mainfw*, enter the shell and execute `route add -inet 100.64.0.0/16 -link 100.64.254.2`. This is a temporary measure in order to access the web GUI from the host. From now on, all operations will be executed through the GUI, accessible at <https://100.64.254.1> for *mainfw* and <https://100.64.2.1> for *intfw*. In *mainfw*, add route 100.64.0.0/16 using gateway 100.64.254.2.

Disable outbound NAT in *intfw* (unnecessary, as it is an internal firewall) and outbound DNS in both machines (DNS servers are configured statically in each host). In *mainfw*, set outbound NAT in manual mode and create a rule for NATting the traffic that leaves the private network.

Enable remote logging in both server ticking the corresponding checkbox in System → Settings → Logging. Then, insert 100.64.1.3 as the server and select to send all contents.

3 Evaluation of the security policy

3.1 DNS

All internal hosts should be able to resolve internal domain names. Since devices in the *client* network are the only ones that can access external web services, no other host should be able to resolve external domains,

3.2 FTP

All internal hosts should be able to connect to the internal FTP server, while only devices in *client* network should be able to connect to external ones. External hosts should be able to connect to the internal FTP server.

3.3 HTTP/HTTPS

All internal hosts should be able to connect to the internal web server, while only devices in *client* network should be able to connect to external websites and to the firewall-router administration pages. External hosts should be able to connect to the internal web server.

3.4 SSH

Hosts in *client* network should be able to use the SSH protocol in the internal network. No other use is authorised.

3.5 Syslog

Internal hosts should be able to send logging messages to the syslog server. No other use is authorised.

4 Policy implementation in OPNsense

There are two types of rules in OPNsense: floating and interface. Floating rules apply to all packets and (with `quick` option enabled) have the highest precedence, while interface rules apply only to packets arriving in a specific interface. Unless otherwise stated, enabling a rule automatically creates a reply-to inverse rule that allows reply packets. Since the network topology is quite simple, only interface rules have been used.

4.1 Default policy

Packets not matched by any rule are logged and blocked. Since OPNsense adopts the opposite default policy for LAN interfaces, the chosen behaviour has been manually enforced everywhere.

4.2 Address spoofing

To avoid IP address spoofing, firewalls reject packets coming from IP addresses not belonging to the subnet the receiving interface is connected to.

4.3 Aliases

To each relevant address (or address range) has been assigned an alias. When formulating firewall or port forwarding rules, we can use the alias instead of the correspondent address, increasing readability and decreasing the possibility of errors.

4.4 Packets arriving to *mainfw* from...

4.4.1 ...DMZ

	Proto	Source	Port	Destination	Port	Gateway	Schedule	Description ⓘ
✗ ⓘ	IPv4 *	! DMZ net	*	*	*	*		DMZ anti-spoofing
▶	IPv4 UDP	*	*	dns_address 📄	53 (DNS)	*		
▶	IPv4 UDP	*	*	syslog_address 📄	514	*		
✗ ⓘ	IPv4+6 *	*	*	*	*	*		DMZ default block

4.4.2 ...outside

	Proto	Source	Port	Destination	Port	Gateway	Schedule	Description ⓘ
✗ ⓘ	*	RFC 1918 networks	*	*	*	*		Block private networks
✗ ⓘ	*	Reserved/not assigned by IANA	*	*	*	*		Block bogon networks
▶	IPv4 TCP	*	*	ftp_address 📄	21 (FTP)	*		NAT
▶	IPv4 TCP	*	*	web_address 📄	80 (HTTP)	*		NAT
✗ ⓘ	IPv4+6 *	*	*	*	*	*		WAN default block

		Source		Destination		NAT	
If	Proto	Address	Ports	Address	Ports	IP	Ports
WAN	TCP	*	*	WAN address	21 (FTP)	ftp_address 📄	21 (FTP)
WAN	TCP	*	*	WAN address	80 (HTTP)	web_address 📄	80 (HTTP)

Since the FTP and the web servers provide public services but have no public IP address, it is necessary to enable port forwarding.

4.4.3 ... *intfw*

	Proto	Source	Port	Destination	Port	Gateway	Schedule	Description ⓘ
✗ ⓘ	IPv4+6 *	! internal_net 📊	*	*	*	*		Internal anti-spoofing
▶	IPv4 TCP	client_net 📊	*	*	80 (HTTP)	*		
▶	IPv4 TCP	client_net 📊	*	*	443 (HTTPS)	*		
▶	IPv4 TCP	client_net 📊	*	internal_net 📊	22 (SSH)	*		
▶	IPv4 TCP/UDP	dns_address 📊	*	*	53 (DNS)	*		
▶	IPv4 TCP	*	*	ftp_address 📊	21 (FTP)	*		
▶	IPv4 TCP	*	*	web_address 📊	80 (HTTP)	*		
✗ ⓘ	IPv4+6 *	*	*	*	*	*		Internal default block

An attacker that has compromised *intfw* (or the link between the firewalls) is able to spoof address in *DMZ*. However, this small risk is greatly surpassed by the additional flexibility: it is possible to add a new low-privilege internal subnet without modifying firewall rules.











4.5 Packets arriving to *intfw* from ...

4.5.1 ... *mainfw*









	Proto	Source	Port	Destination	Port	Gateway	Schedule	Description ⓘ
✗ ⓘ	*	Reserved/not assigned by IANA	*	*	*	*		Block bogon networks
▶	IPv4 UDP	internal_net 📊	*	dns_address 📊	53 (DNS)	*		
▶	IPv4 UDP	internal_net 📊	*	syslog_address 📊	514	*		
✗ ⓘ	IPv4+6 *	*	*	*	*	*		Main default block

Similarly to the previous section, an attacker that has compromised *mainfw* (or the link between the firewalls) is able to spoof addresses in *client* or *internal server* networks.

4.5.2 ...the *client* network

	Proto	Source	Port	Destination	Port	Gateway	Schedule	Description ⓘ
	IPv4+6 *	! client net	*	*	*	*		Client anti-spoofing
	IPv4 UDP	*	*	dns_address 	53 (DNS)	*		
	IPv4 TCP	*	*	ftp_address 	21 (FTP)	*		
	IPv4 TCP	*	*	*	80 (HTTP)	*		
	IPv4 TCP	*	*	*	443 (HTTPS)	*		
	IPv4 TCP	*	*	internal_net 	22 (SSH)	*		
	IPv4+6 *	*	*	*	*	*		Client default block

4.5.3 ...the *internal server* network

	Proto	Source	Port	Destination	Port	Gateway	Schedule	Description ⓘ
	IPv4+6 *	! server net	*	*	*	*		Server anti-spoofing
	IPv4 TCP/UDP	dns_address 	*	*	53 (DNS)	*		
	IPv4 TCP	*	*	ftp_address 	*	*		
	IPv4 TCP	*	*	web_address 	80 (HTTP)	*		
	IPv4+6 *	*	*	*	*	*		Server default block

5 Test of the configuration

In the root directory of each Kathará host there is an executable script `(host).test` that checks the firewall and DNS functionality. The return code is a 4-bit flag:

- 0001: error on DNS resolution of internal hosts
- 0010: error on DNS resolution of external hosts
- 0100: error on internal web connectivity
- 1000: error on external web connectivity.

Given that *expect* is not installed and both *ftp* and *ssh* clients can not be reliably used with simple bash scripts, testing the corresponding protocols must be done manually, checking the connectivity for each host.

6 Final remarks

When checksum offload is activated, all packets passing through the firewall get corrupted. Disabling it on the host's virtual interfaces produce no effects, it must be done from the Kathará hosts.

In theory, outbound NAT should be disabled in *mainfw*, as VirtualBox already takes care of translating the IP addresses. In practice, disabling it conflicts with the configuration script.