Practical Network Defense

Second Assignment - University "La Sapienza"

Group 27: Nicola Bartoloni ******* - Valerio Trenta 1856471

2020-13-06

1 Scope and Initial Considerations

The scope of this assignment comprehends the whole target network: a series of services provided by different machines in several subnetworks in the **ACME** environment are listed, and we focus on how to enable these services to the clients of our network by exposing them on a controlled manner through firewall rules specified in the **OPNSense** service provided by the two main routers of the network.
By catching a quick glimpse of the policy proposed by the assignemnt, we can tell the scope is to provide our network of a series of *white-listed* services that we want to be enabled and reachable from certain machines in the network itself, thus reducing the *attack surface* of the network by also excluding from this white list every other service that is not of interest.
Our analysis should begin with the services that we want to provide in the target network, and then on the firewall rules that make it possible to reach each service on each different machine.
Please notice that, since IPv6 addresses are not part of the scope of this assignment, we will ignore them during the configuration of the firewall rules in section 4.

2 Infrastructure Setup

We identify four main services which require machine-to-machine communication and, thus, are susceptible to the firewall rules that we'll define:

- an **Apache** web server reachable through **TCP** ports **80** (HTTP) and **443** (HTTPS) at **100.100.6.2**;
- a **DNS service** reachable through standard **UDP** port **53** at **100.100.1.2**;
- a **syslog server** reachable through standard **UDP** port **514** at **100.100.1.3**;
- a **proxy server** reachable through port **3128** at **100.100.6.3**, to be configured later on Assignemnt 3.

We thus identify two new subnetworks: the **Internal Servers** subnetwork **100.100.1.0/24** and the **DMZ** subnetwork **100.100.6.0/24**, and for the sake of defining the scope of this assignment and the setup of the whole infrastructure, we should also consider two additional details:

- Two of the four services - web server and syslog server - are already setup and running in the system: they do not need to be configured or launched;
- the **proxy server** is to be configured on the next assignment: for now, knowing from the **zentyal** portal that it is provided on port 3128 is enough.

This leaves us with the only **DNS service** to be configured at the moment. This step has been performed by following the instructions provided with the assignment: by accessing the **zentyal** portal at **100.100.1.2** on port **8843** - credentials have been changed - we added as forwarders the suggested IP addresses and specified the domain name - **acme.group27**. Then, the file located at **/etc/zentyal/dns.conf** on the aforementioned machine was modified including the target subnetworks (Clients and DMZ, that are the ones which will exploit the service).
At this point, pairs of IP addresses and hostnames were specified in the portal, so that the well-known machines of the network can now be associated with the following names:

- **kali**.acme.group27;
- **watchdog**.acme.group27;
- **dc**.acme.group27;
- **web**.acme.group27;
- **proxy**.acme.group27;

only the machines in the target subnetworks were actually included - so machines in the External subnet did not receive a hostname, and as suggested in the assignment were provided with an external DNS service such as 8.8.8.8 in their DHCPv4 configuration. Please notice also that the *logserver* was not given a name since it is only meant to be accessed by the SPOCK environment or by SSH and is not offering any browser-related service - and the same reasoning could be applied to the first two machines, *kali* and *arpwatch*, which we could exclude from this pairing list.
Last step was to actually modify the DHCPv4 settings in the two main routers to set the **dc** machine as the **DNS server** in the Internal Servers network, after having disabled the **Service Unbound DNS** in both routers.
The **DNS service** configuration is tested in section 5.

3 Policy Evaluation

The proposed policy targets the four services listed in the previous paragraph and a series of machines which will either exploit or provide the corresponding services.
The best way of interpreting and understanding this policy is given by its fifth line:

- *"Anything that is not specifically allowed has to be denied"*;

which suggests the *white-listed* approach we have to adopt when defining the firewall rules: the policy is a list of **PASS** rules - meaning, rules that when matched will let the packet pass and continue its journey - while everything that doesn't match the rule has to be rejected, i.e. the packet must be dropped. Thus, we can devise a list of rules to apply at each of the two target routers and their corresponding effects on the network.

Internal Router:

- **Clients Interface**: only accept incoming packets with destination ports 80(HTTP), 443(HTTPS), 22(SSH), 53(UDP-DNS). Anything else will be discarder, since clients are not supposed to perform different actions and exploit different protocols than the aforementioned ones. For practical reasons, also packets on port 8443 (for **zentyal** panel service) are allowed - but this might be changed;
- **Servers Interface**: only accept outgoing packets with destination port 53(UDP-DNS) or 514(UDP-SYSLOG, if coming from **DMZ subnet**) and on port 22(SSH) if coming from the **Clients subnet**, otherwise discard. This means that the two servers can only be reached by Clients or DMZ subnetworks, and only for the services that they provide;

Main Router:

- **Internal Interface:** only accept incoming/outgoing packets on ports 80(HTTP), 443(HTTPS), 22(SSH) and 3182(Proxy) between **Clients subnet** and **DMZ subnet**, otherwise discard. This means the Client host will only be able to reach the web server or the proxy to actually reach the WAN, and won't be able to do it directly by itself. Furthermore, SSH is eanbled for Client hosts to reach the **DMZ subnet**;
- **DMZ Interface:** we want the proxy server to be able to reach the internet via HTTP/HTTPS protocol, so in addition to the rules already specified for the Internal Interface, we also add that this interface has to accept incoming packets specifically from the **Proxy server** and with **any** destination address on ports HTTP/HTTPS, and the same gose for the **web server** since we want it to be accessible from the Internet. However, the **proxy service** itself must be onyl available for client hosts in the **Clients subnet**, so we should also specify that this interface should accept incoming/outgoing packets on port 3182 with destination address in the **Clients subnet**, and reject every other packet on the same port with different destination;

- **WAN Interface:** accept incoming/outgoing packets on port 53 from **External services subnet** - the external services need to reach external DNS service on the Internet - and TCP connections on ports 80(HTTP) or 443(HTTPS) if they are coming from the **DMZ**, while all the other incoming/outgoing packets should be rejected since all the other services cannot be reached from the external WAN.

If implemented correctly, this policy should allow the Internal services to be exploited and reachable only by the **Clients or DMZ networks**, and it should also allow the client hosts in the **Clients** network to either reach the **web server** or the Internet via the **Proxy service** to be configured in the next assignment, being these two services in the **DMZ** the only two exposed to (and thus reachable from) the Internet.
In order to make this work correctly, since it is a *white-listed* approach, we have to be careful in specifying every single service that is enabled in every interface: only considering the source or destination subnet, in fact, would not be enough - e.g., we do not want a **client** host to be able to connect through **telnet** on a **DMZ** host, but if we only specify the subnet and not the white-listed service/port, this is what we are going to get.

4 Policy Implementation

5 Tests

5.1 Testing the DNS Service

5.2 Testing the policy

6 Final remarks

References

[1] *Just a placeholder.*