Lab - Firewall setup

Managing and securing computer networks

Floriane Magera (S111295)
Fabrice Servais (S111093)

may 13, 2015

1 Overview

In order to protect our subnet, we used extended access lists and rate limitations. We placed our firewall rules on the outer interface of the router. We made this choice because we refuse a lot of incoming traffic, in this way, the non-conform traffic is dropped before being routed.

We based ourselves a lot on the standard ports in order to filter the traffic according to each protocol. This could be a big limitation if the standard ports were not used, but when we scanned the ports of the server, we saw that these ports were used in almost every case (sometimes there is other ports dedicated in addition to the standard ones).

2 Authorized inbound traffic

ICMP We decided to limit the ICMP traffic to less than 5 percent of the bandwidth of the link. We did not differentiate according to the type of `ICMP` as there are so many of them and as `ICMP` is not absolutely necessary to the proper function of a network.

HTTP(S) We accepted connections to the standard ports 80 and 443 of our web server. We also authorized traffic in the other direction for established connections.

SSH We authorized any connection from a machine of the subnet towards another machine (of the subnet) and the responses.

3 Authorized outbound traffic

Essential services The two essential services we identified was the routing protocol in this case OSPF and also DHCP. We authorized any traffic for OSPF. For DHCP, we authorized the discovery process and requests again on the standard ports. In order to design the rules for DHCP, we had to observe the traffic on our subnet.

DNS We authorized traffic towards Queen with the standard port.

ICMP The outbound ICMP traffic was limited to 5% of the bandwidth too. As for the inbound traffic, we did not allow differentiate the filtering according to the type of ICMP packet.

LDAP We authorized traffic towards ant on the port 2049.

NFS we authorized traffic towards ant on the port 389. If we had more time we would have used the port mapper in order to allow other services of NFS proposed on ant.

HTTP(S) We authorized traffic to the internet on the standard ports of HTTP and HTTPS.

SMTP We authorized traffic to the SMTP server on the standard port of SMTP.

FTP

SSH We authorized outbound traffic to the network on the standard SSH port.

LOGS The problem we face when storing our logs on bee is that bee is outside our subnet and we don't know how bee is protected against intrusions. It would be annoying for us if an attacker had access to our logs, because he could learn a lot of things about our firewall and adapt his attacks to the properties of our firewall. It would be easier for him to attack us with the informations provided by the log.

4 General rules

As asked, we denied all traffic that was not explicitly authorized. For the response, we used the keyword `established` in our access lists in order to permit these rules with only established connections. To forbid IPv6, we made only one IPv6 access list which denied all kind of traffic. Now let's focus on the mechanism we set up against attacks.

Spoofing First we denied all kind of incoming traffic which had Ladybug or Hornet as source so that we refuse any traffic spoofing Hornet or Ladybug from outside our network. This measure is limited : Hornet could spoof Ladybug, and anyone connecting in our subnet could do it too. In order to prevent a machine in our subnet to spoof another machine's ip which is not belonging to the subnet, we were as precise as possible for the rules outgoing our router. This means that we always used the subnet mask as a source for outgoing packets when the packet for the service aimed by the access list could not originate from owl. In this way, someone in our subnet can not impersonate someone outside the subnet.

Smurf attack The principle of the smurf attack is to broadcast a message from a spoofed source in order to return a huge number of responses to that victim. In order to prevent our subnet to take part in that kind of attack, we forbid the transmission of any broadcast message. In that way, it is impossible dor our network to serve as an amplifier for a smurf attack.

SYN flooding In order to avoid syn flooding, we set up some threshold : the maximum number of half-open connections is 200 connections, and once this threshold reached, we reset one half-open connection for each new connection until another threshold of 100 connections is reached. We decided to let the high threshold to 200 as Hornet is a web server, it can have a larger number of connection towards it. This measure does not prevent from any syn flooding, some harm can still be done as the number of authorized incomplete connection is quite high, but we need to balance between efficiency of the web server and attack prevention. A too strict threshold may be harmful too and prevent useful traffic.

5 Test

We tested first each basic service : ssh, icmp (by ping), smtp (by sending mails), http(s). Due to several problems during our last session, we were not able to test LDAP, FTP nor NFS.

Concerning the tests of the attacks, we first tested the icmp flooding, and as planned, we drop the packets when a certain threshold (5% of the bandwidth) is reached.

Then we tested some broadcast packets, which were well dropped.

Finally, we could not test the SYN flooding measure due to the lack of time. It would have been nice for us to be able to test the relevance of our thresholds.

6 Conclusion

We are disappointed to be unable to test more our configuration : during our last session, we spent hours on silly problems: we encountered problems to reload the initial configuration of owl, also as the lab was overloaded and we had to switch to the router owl. Happily we had tested the basic functionalities before and we were able to perform quickly other tests.