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 uses 2 access-group, one `in` and one `out` to enhance the lisibility of the rules and thus also the ease of the maintenance.

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 cases (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 10 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

LDAP

NFS

HTTP(S)

SMTP

FTP

SSH

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 traffic which had Ladybug or Hornet as source on our incoming interface. We refuse 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. Also as there is no established subnet mask for the subnet, the rules in order to protect the internet and the rest of our network from our network are complicated. Ideally we should drop any traffic outgoing which is not from Hornet or Ladybug, but this would prevent any traffic if a machine was added in our network and it would not be easy to maintain in this case. So for now we don't protect the world from our network.

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 moderate he effect of such an attack, we limit the number of broadcast packets that can be emitted towards our network. In that way the number of packets in our network is limited and as there are no many machines, it would be difficult to be an efficient amplifier. We could not simply forbid any broadcast packets as it is used by DHCP. Note that as the icmp packets are limited, a smurf attack using icmp would fail.

SYN flooding In order to avoid syn flooding, we set up some threshold : the maximum number of half-open connections is 300 connections, and once this threshold reached, we reset one half-open connection for each new connection until another threshold of 200 connections is reached. We decided to let the high threshold to 300 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, smtp, http(s), etc. Then tried some more advanced tests: