Offensive technologies - SYN Flood exercise

Claudio Facchinetti [claudio.facchinetti@studenti.unitn.it]

October 18, 2021

Abstract

In this document it is presented the work done in order to solve the "SYN Flood" exercise on DeterLab.

1 Solving tasks

1.1 Setup the environment

In order to solve the task we first need to install the packages needed on both the server and the attacker machine; to do this a simple script is used. This script connects via ssh to both machines and install required packages. This also disables SYN cookies on the server.

```
#!/bin/sh

# Install flooder
ssh attacker.facchinetti-synflo.offtech
"/share/education/TCPSYNFlood_USC_ISI/install-flooder"

# Install the apache web server and disable SYN cookies
ssh server.facchinetti-synflo.offtech
"/share/education/TCPSYNFlood_USC_ISI/install-server;
sudo sysctl -w net.ipv4.tcp_syncookies=0;
sudo sysctl -w net.ipv4.tcp_max_syn_backlog=10000"
```

A script for the client has also been prepared: this script simply creates another script which will perform the HTTP request every second and gathers data with `tcpdump`.

```
#!/bin/sh

# The interface to use
ETH=eth4

# Write the script
cat << EOF > cl-request.sh
#!/bin/sh
while sleep 1; do
  curl http://5.6.7.8 > /dev/null 2> /dev/null;
done;
EOF

chmod +x cl-request.sh
# Start recording network traffic on the interface
sudo tcpdump -nn -v -s0 -i ${ETH} -w record.pcap &

# Perform requests
./cl-request.sh
```

Please note that the `ETH` variable needs to be changed on every swap in of the machine; in order to find the correct interface simply run the following command and look at the result.

```
ip route 5.6.7.8
```

2 Gathering data

As the tasks required, data has been gathered using the `tcpdump` tool on the client machine and then the analyzed to gather some statistics on connection duration. In particular the following scenarios has been considered:

- `NET1_NOSYN_SPOOF`: the network structure is the original one, the attacker does spoof ip addressed and SYN cookies are disables;
- `NET1_SYN_SPOOF`: the network structure is the original one, the attacker does spoof addresses and SYN cookies are enables;
- `NET1_NOSYN_NOSPOOF`: the network structure is the original one, the attacker does not spoof addresses and SYN cookies are disabled;
- `NET2_NOSYN_NOSPOOF`: the network is the modified one, the attacker does not spoof and SYN cookies are disabled.

In order to measure time the `stopwatch.sh` script has been used: it does only print current date and time, therefore the actual code has been omitted.
In all scenarios the attacker has been configured to send one hundred packets per second.

2.1 `NET1_NOSYN_SPOOF`

By looking at the graph we can notice that there is a massive growth in connection duration right after the attacker starts flooding the server with packets spoofing the IP address of some machines in the client network. When the attacker stops flooding, at second 150, the connection duration returns as at the beginning of the experiment.
By looking at the `tcpdump` result we also notice that a lot of SYNACK packets from the server were received by the client: this is due to the spoofing the attacker is performing.
The high connection durations is due to multiple factors:

- the client is spending a lot of time dropping packets from the server;
- the router is congestioned by the high load of packets;
- the server is allocating a lot of memory, eventually all of it, because of the SYN packets.

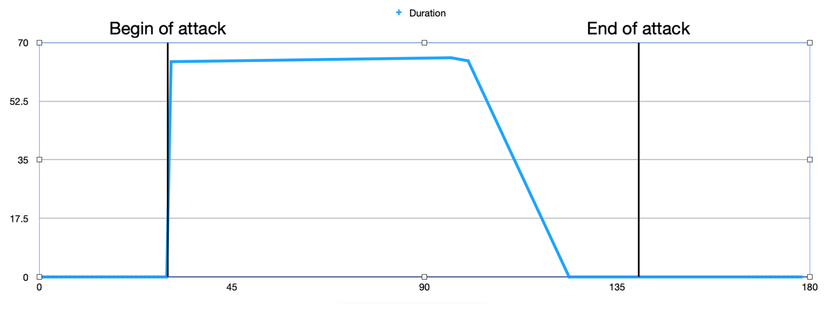

Figure 1: Connection duration of the client over time in the first setup

2.2 NET1_SYN_SPOOF

By looking at the graph we can notice that in this case there is some fluctuation in the connection duration, but just one or two cases where the client actually requires a lot of time to get an answer. This is due to the fact that right now SYN cookies are enabled: right now the server does not occupy space if an ACK containing the correct sequence number is not received.
Even if the client is able to get responses notice that there is a little bit of fluctuation, probably due to the fact that we are congestioning the router and that the client is spending some time dropping server packets caused by attacker packets. By looking at the `tcpdump` results we still see that there are a lot of SYNACK packets from the server received by the client.

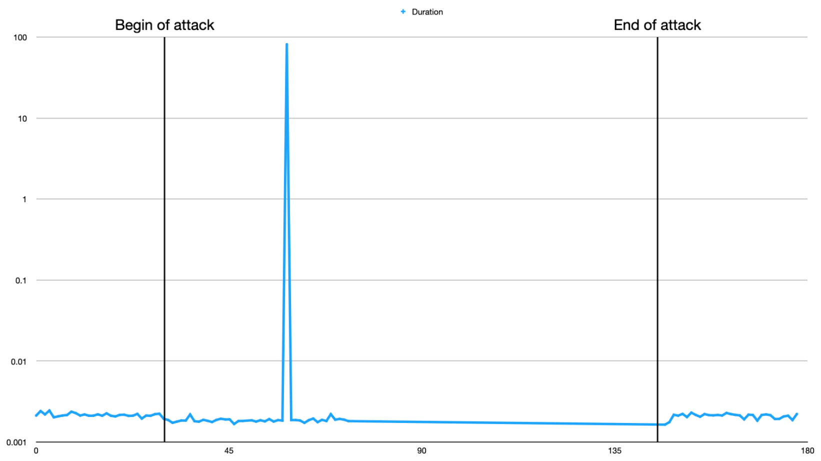

Figure 2: Connection duration of the client over time in the second setup

2.3 NET1_NOSYN_NOSPOOF

By looking at the graph we can notice that in this case there is almost no influence: the client can receive responses in a reasonable amount of time. The only cases where the client takes a larger amount of time to complete the connection may be due to the fact that the router is congestioned with all the packets received.
This happens because the attacker is now no longer sending packets with the IP address of some node on the client network segment, therefore the client is no longer busy to drop packets. A simple way to make the attack work without spoofing would be to increase the number of packets per second: if we manage to completely congestionate the router so that the client cannot communicate with the server. Another way would be to implement a ”distributed fashion” of this attack having multiple machines, like members of a botnet, performing a lot of requests to the server in order to make sure that no packet from the client reaches the server.

2.4 NET1_NOSYN_NOSPOOF

In this case the network configuration has been changed: the router has been removed from the equation and the server now has peer-to-peer connections with both the client and the attacker machines.
By looking at the graph we can notice that the client is slightly affected by the attack, while having the overall connection duration almost halved. This confirms our previous hypothesis: in the previous case the client experienced, from time to time, high connection durations mainly because of the high load on the router.

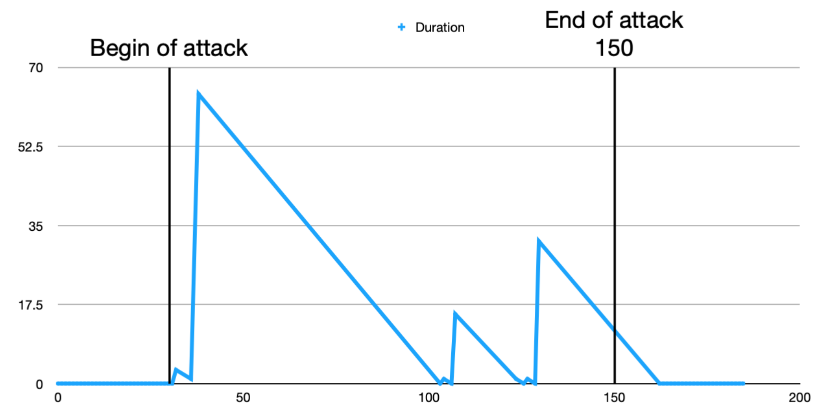

Figure 3: Connection duration of the client over time in the third setup

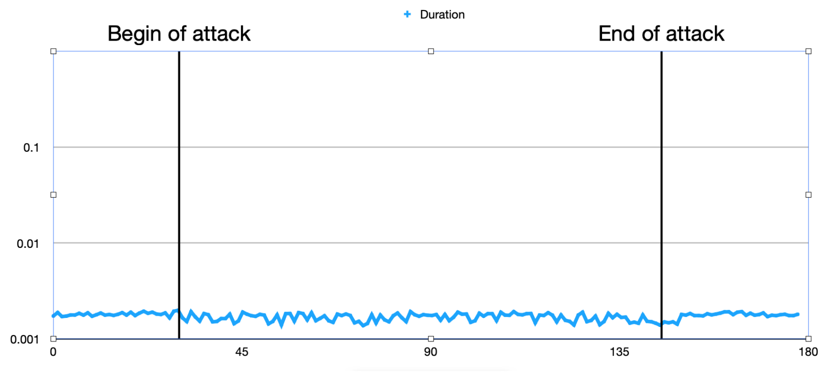

Figure 4: Connection duration of the client over time in the fourth setup