Hanshuo Geng, Dom Spiotta, Joel Meyer, Alex Schieltz

Dr. Noureddine

Network Security

Milestone#2 Report

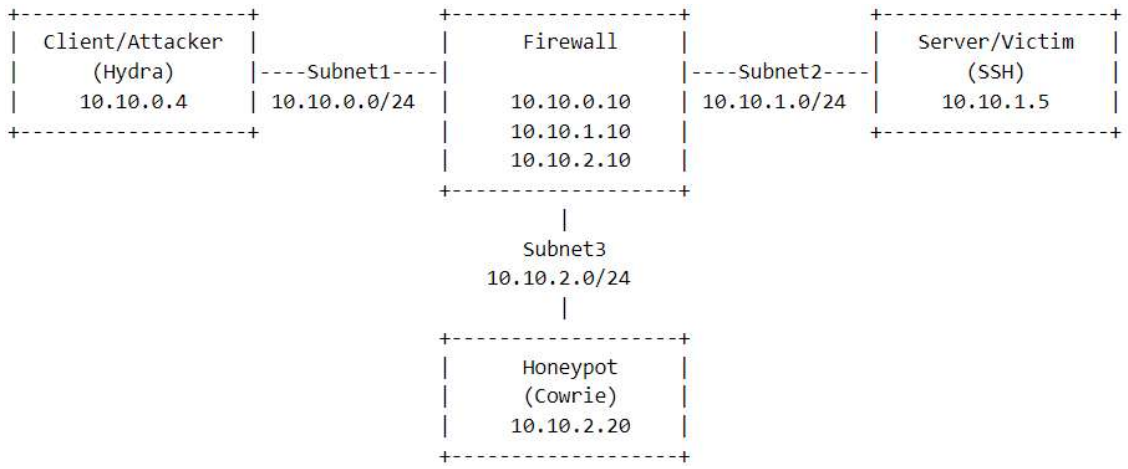
Introduction/Scope

The purpose of this lab is to introduce students to brute-force ssh attacks on a server and potential blocks against them. The lab will start with students analyzing the network and rules on the firewall, then attempting a brute-force hydra attack. Afterwards students will experiment with firewall rules to detect the brute force attacks and a system for redirecting to an outside machine (The outside machine should have all necessary tools to have it run a honeypot, but the setup for the honeypot is not covered in this lab).

Success Criteria:

Success can be determined by successfully implementing firewall rules to stop a brute force attack through routing traffic to an alternate container and understanding the techniques involved with the firewall.

Network Topology



Special Attributes

- Firewall (10.10.0.10, 10.10.1.10, 10.10.2.10):
  - Only forwards ssh attempts, blocks everything else.
  - Doesn't accept any packets destined for the firewall.
  - Should be connected to all containers, hard to test since it is meant to block input by default.
- Server (10.10.1.5):
  - configured for ssh (root:netsec)
  - also running services like telnet - not important for this lab since firewall only forwards ssh
- Client/Attacker (10.10.0.4):
  - no special attributes other than hydra, which is installed on startup
- Honeypot (10.10.2.20):
  - running cowrie - to later be setup as a full honeypot

Test Cases:

- Ping server and ping firewall to make sure no response comes back
- Test ssh root@server to ensure ssh connection over firewall is working
- Run "hydra" on client to make sure hydra is installed
- Only way to test firewall's connection to honeypot is to temporarily delete the firewall table netsec_tbl and try a ping on honeypot.

Lab Walkthrough:

Words marked in red are questions for the students to answer/think about.
Words marked in green are answers that we want the student to get at.
Words marked in gold are commands to be copied/typed by the user.

1. Learning the setup
   a. dcupd
   b. Log in to client.
   c. Attempt a ping to firewall and a ping to server, no response should be received
      i. ping firewall
      ii. ping server
   d. Attempt to ssh into server (if student wants, password is "netsec")
      i. ssh root@server
   e. Why does ssh work and icmp doesn't? (can look at firewall.nft for the firewall rules)
2. Learning about hydra

a. Run “hydra” on client. If it says “command hydra not found” wait 5-10 seconds and try again, if it still doesn’t work, run “sudo apt install -y hydra”
b. What is hydra and why might we want it when connecting to an ssh server?
   How do we specify a specific port/protocol to use it on?
c. Students may analyze hydra man pages to learn about specific flags
d. What do the flags -l and -P do in hydra?
e. Try “ssh root@server” again and this time type in 3 wrong passwords. (it should cancel our shell connection)
f. Now we will attempt a brute-force attack
   i. start a tcpdump on firewall
      “sudo tcpdump -i eth0 -w /volumes/brute_force.pcap”
   ii. attempt to use hydra to brute force into server (from client) over ssh using the password.txt file provided in the volumes directory. (Students should figure the command out on their own, but here it is: “hydra -l root -P password.txt ssh://server”).
g. What is the output from hydra? (should say something about root:netsec being shown as a login for ssh)

```
[22][ssh] host: server   login: root   password: netsec
```

3. Analyzing brute-force attacks
   a. Log into the firewall with ./connect_firewall script provided
   b. Run “sudo nft list table netsec_tbl”
   c. What do you notice about the table?
      i. How many chains are there and what are their policies?
      ii. Ignoring the first two rules in the netsec_forward chain, what type of traffic do the other 2 rules seem to be allowing/blocking?
      (tcp dport { 22, 2222 } accept
      tcp sport { 22, 2222 } accept)
   d. Take a look at the pcap from before. What do you notice about the ssh connections? Why might client be repeatedly sending FIN,ACK packets throughout the capture? (indicates hydra making many connections)
   e. If you look closely, you can also see that client attempts to connect to port 22 on server from multiple different ports on client. What does this mean in the context of hydra? (check out -t flag in hydra man page). (Hydra is attempting to sort of pipeline or multi thread login attempts)
   f. Note that there are RST, ACK flags being made to client from “server”. (this is related to the previous question)
   g. Now consider the first 2 rules in the netsec_tbl nft table. The first one is one that we’ve seen before “tcp dport {ssh} ct state established,related accept;”. This allows established ssh connections to pass freely over the firewall.

h. What do you think the other rule is trying to do, and how might it be related to the RST,ACK packets we saw in the capture? (see https://wiki.nftables.org/wiki-nftables/index.php/Meters) for information on how meters work in nftables. (sends a tcp RST when > 3 concurrent connections attempt to be made)

4. Fixing the brute-force detection
    a. Open firewall.nft and uncomment the rule "**tcp dport 22 ct state new tcp flags syn limit rate over 5/minute update @ssh_flood {ip saddr timeout 1m};**"
    b. Now run these 2 commands on firewall:
        i. "sudo nft delete table netsec_tbl"
        ii. "sudo nft -f /volumes/firewall.nft"
    c. Now attempt your hydra command from earlier (this will likely take ~5 minutes).
    d. What happens to the hydra command after adjusting the firewall? (it continues trying but takes longer than before. It eventually finishes and says it found no valid passwords. (it could in theory find the password but the odds are quite rare))
    e. What impact did the new rule have on brute-force attacks and why? (The new rule limited the rate at which a machine can send new connections)
    f. Currently, the rule adds the saddr to a set with a timeout on it. Why not remove the timeout and make it permanent? (Removing timeout would be more effective, but if someone truly did forget their password, they could get locked out and need someone to remove them from the blacklist)
    g. Is there something other than a timeout that we could do to make hydra give up? (Open ended, but we're going to consider rerouting the traffic next)
    h. Now that you've been added to the firewall's set it will keep you around for a while. Try deleting the table and recreating it again.
        i. "sudo nft delete table netsec_tbl"
        ii. "sudo nft -f /volumes/firewall.nft"
    i. Try to ssh to server from client normally. It should work, so your firewall works correctly. It blocks brute-force but allows normal traffic.
5. Alternatives to Dropping: DNAT
    a. Now we're going to consider an alternative to just dropping traffic from a malicious machine. What if you could just send the brute-force attacks to a random machine, or even better - let the attacker think falsely that they succeeded.
    b. Go into firewall.nft and:
        i. Comment out "**ip saddr @ssh_flood update @ssh_flood {ip saddr timeout 1m} drop**"

    ii. Uncomment “**ip saddr @ssh_flood counter dnat to 10.10.2.20**”

c. What do you think changing these 2 rules will do?
d. Delete and recreate the table again
    i. “sudo nft delete table netsec_tbl”
    ii. “sudo nft -f /volumes/firewall.nft”
e. Now start a tcpdump on firewall that listens to every subnet:
    i. “sudo tcpdump -i any -w /volumes/dnat.pcap”
f. Now go to client and run hydra. After 1-2 minutes or once the hydra finishes, stop the tcpdump.
g. Analyze the pcap file. Where is the traffic being sent to? (may be a lot of filler packets in the beginning of the capture)