Network Security

Assignment 4A, Wednesday, May 16, 2018, version 1.0

Handing in your answers: Submission via Blackboard (`http://blackboard.ru.nl`)

Deadline: Wednesday, May 23, 23:59:59 (midnight)

The next lecture is in two-and-a-half weeks time. Therefore, assignment 4 is rather large. We split this assignment in three parts so that the grading is distributed somewhat. However, this does not mean that each part has an equal amount of work. When done with one part, move immediately on to the next. Remember that 6 hours per week is the intended time investment for a 3EC course. Don't spend more than 6 hours on part 4a, and make sure that you have 6 hours left for part 4c.

Please turn in all your work in plain text files (program source files are also plain text). If you prefer a document with formatting for whatever reason, like including images, use the PDF format to turn in your work (most editors allow you to export to PDF). Note that it's okay to include images separately and then refer to them from within the text files.

Teaching assistants. Please email *all* of us if you have a question.

- Pol Van Aubel `<pol.vanaubel@cs.ru.nl>`
- Daan Sprenkels `<dsprenkels@science.ru.nl>`
- Wouter Kuhnen `<w.j.a.kuhnen@student.ru.nl>`

In this assignment you will be using the following tools:

- iptables: `http://netfilter.org/projects/iptables/index.html`
- sshuttle: `https://github.com/sshuttle/sshuttle`

Do not compile these programs from source. iptables is likely installed by default. Use your Linux distribution's package manager to install the package sshuttle (e.g. Ubuntu/Debian: `apt-get install sshuttle`, Arch Linux: `pacman -S sshuttle`, Fedora: `yum install sshuttle`, Gentoo: `emerge sshuttle`).

Many commands in this exercise need to be run with root rights. This is denoted by a prefix `#`. When a command should be run without root rights, it will be prefixed with `$`. Do not include the prefix when typing the command.

1. In this exercise you will use iptables to create two firewall configurations: one for a client machine, one for a masquerading server. You are encouraged to test your configuration on your own (virtual) machine.

   You can use the commands `iptables-save` and `iptables-restore` to save and restore iptables rules to and from a file, respectively. Usage: `# iptables-save > filename` stores the firewall configuration in the file `filename`. `# iptables-restore < filename` restores the firewall configuration from `filename`. It might be a good idea to run `iptables-save` on the firewall configuration you have before starting this exercise.

   If you get in unrecoverable trouble, you can completely reset the firewall configuration by running the following commands:

```
# iptables -F
# iptables -X
# iptables -t nat -F
# iptables -t nat -X
# iptables -t mangle -F
# iptables -t mangle -X
# iptables -P INPUT ACCEPT
# iptables -P FORWARD ACCEPT
# iptables -P OUTPUT ACCEPT
```

   This will flush (`-F`) all built-in chains and delete (`-X`) all user-defined chains in the standard tables, and set the default policy (`-P`) to accept.

   Create a folder called `exercise1` to hold the answers to this exercise.

   Note that under some Linux distributions (most notably Ubuntu), you may have to add a rule allowing traffic from localhost to localhost in order to allow some local processes to communicate.

   For this exercise, use the iptables manpage (`man iptables`), the netfilter documentation on http://www.netfilter.org/documentation/index.html#documentation-howto (especially the Packet Filtering HOWTO), the lecture slides, and any other sources you want.

   (a) Build a client firewall that does the following:

   - Allow all outgoing traffic.
   - Deny all incoming traffic, except
     - traffic that belongs to an established connection, and
     - incoming SSH traffic (filter on transport protocol and port).
   - Allow all ICMP traffic, except ICMP redirects.[1]

   If you use tutorials or examples, please make sure you understand what the rules do.

   Write your firewall configuration, preferably dumped by `# iptables-save`, to `exercise1a.fw`

[1] There are other ways to handle ICMP redirects securely. E.g. on Linux, there is the /proc/sys/net/ipv4/conf/*/secure_redirects directive, which tells the kernel to ignore all ICMP redirects that don't redirect to an already known gateway.

(b) If you felt exercise 1a was easy, try your hand at this one. Otherwise, skip to exercise 2 and come back if you have time left over.

Build a firewall for a masquerading server / router with two network cards: `eth0` with IP address 198.51.100.42 and `eth1` with address 10.0.0.1/24. `eth1` is the internal card, the internal network should be obvious from its address. `eth0` is the external card, which is the link to the Internet. The firewall should do the following:

- Masquerade traffic coming from the local network going to the Internet.
- Allow outgoing traffic to the Internet that's forwarded from the local network.
- Block all outgoing traffic from the machine itself to the Internet, except for ICMP and traffic that belongs to an established connection.
- Allow all incoming ICMP traffic.
- Allow all outgoing traffic from the machine itself to the local network.
- Block all incoming traffic from the Internet, except traffic that belongs to an established connection.
- Accept incoming TCP connections on port 80 (let's say that's a webinterface) and port 22 from the Internet.
- Forward TCP and UDP traffic on port 2222 and 8080 to some other host in the local network. Feel free to pick that host yourself.

Since you likely cannot test this, simply give it your best shot. This is the type of configuration you'd use if you use a Linux machine as your home router.

Write your firewall configuration, preferably dumped by `# iptables-save`, to `exercise1b.fw`

2. In this exercise you will use sshuttle to set up a secure tunnel to the lilo login server of the Faculty of Science, and then inspect and analyze the resulting iptables rules. For this, you will need to use your Faculty of Science account. If you do not have one, and do not have access to an alternative SSH server on which sshuttle works, send me an e-mail as soon as possible.

   Use the manpage of sshuttle (`man sshuttle`) to figure out the command to route everything through the VPN. The remote host to use is `<username>@lilo.science.ru.nl`, where `<username>` is your Science login name. Write the command you use to `exercise2`. Remember to run sshuttle as root.

   If the command consistently dies with `c : fatal: server died with error code 255`, you may need to add an exclusion rule for lilo.science.ru.nl.

   Now, view the resulting iptables configuration using either `# iptables -t <table> -L` for each table listed in the manual page, or use `# iptables-save`. Write the rules to `exercise2`, and explain what they do and why they route everything through the VPN. Try e.g. looking for the port number you see in a listing of listening ports.

   Feel free to play with other configurations (e.g. routing only certain networks through the VPN, or using exceptions) and explain what the different firewall rules for these configurations do as well.

3. Place the files and directories `exercise1` and `exercise2`, and all their contents, in a folder called `netsec-assignment4a-STUDENTNUMBER1-STUDENTNUMBER2`. Replace `STUDENTNUMBER1` and `STUDENTNUMBER2` by your respective student numbers, and accommodate for extra / fewer student numbers. Make a `tar.gz` archive of the whole `netsec-assignment4a-STUDENTNUMBER1-STUDENTNUMBER2` directory and submit this archive in Blackboard.