TECHNISCHE UNIVERSITÄT BERLIN
Fakultät IV – Elektrotechnik und Informatik
FG INET
Prof. Dr. Thomas Zinner
Theresa Enghardt, Apoorv Shukla, Damien Foucard, Thomas Krenc

Routerlab SoSe 2018
Worksheet 5: Networking with Linux, ACLs, Packet Filtering, and Traffic Shaping

Just like our Cisco and Juniper routers, a Linux host can be used to forward packets. In this assignment, we will focus on controlling and adjusting what happens in the data plane, that is, packet forwarding. We will examine different approaches to filter traffic, define a firewall, break through the firewall, and control the bandwidth utilization of the network.

Table 1: Assignment of devices to groups

Cloud	Aachen	Köln	Leverkusen
Host A	groupX-lg2	groupX-lg2	groupX-lg2
Linux-Router	groupX-lg3	groupX-lg3	groupX-lg3
Host B	groupX-lg4	groupX-lg4	groupX-lg4
Switch	aac-sc1	cgn-sc1	lev-sc1
Router	aac-rj2	cgn-rj2	lev-rj2
IPv4 range		`10.Z.0.0/16`	

Note: Replace X with the number of your group with leading zero, e.g. $X = 03$ for group 3. Replace Y with the number of your group without leading zero and use hex encoding, eg $Y = 3$ for group 3 and $Y = a$ for group 10. Finally replace Z with the decimal group number without leading zero, e.g. $Z = 3$ for group 3.

Question 1: (2 + 3 = 5 Points) *Linux as a Router*

The Linux kernel provides extensive support for TCP/IP networking (as many other OSes do). For security reasons, *packet forwarding* is often deactivated by default in the Linux kernel. In this assignment however, you will enable Linux packet forwarding among other features. See `https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt` for a list of available flags.

In Questions 1-4 of this assignment, you will build a simple 3-node setup to demonstrate basic IP routing within the Linux kernel, using three loadgens according to Figure 1: `Host A` (groupX-lg2), `Linux-Router` (groupX-lg3), `Host B` (groupX-lg4). Your setup should allow `Host A` to communicate with `Host B` via `Linux-Router`, where `Host A` and `Host B` are in different subnets and VLANs[1]. `Linux-Router` should forward packets on Layer 3 from one (virtual) interface to another.

(a) Draw a topology map that shows your assignment of IP addresses, interfaces and VLANs.

(b) Configure your 3 loadgens and your switch so that `Host A` can communicate with `Host B` in the manner described above. Check with ping or ssh. Remember that you have to enable packet forwarding in the `Linux-Router` for this to work.

[1] Since VLAN encapsulation will happen within `eth1` on the Linux router, you may have to decrease the MTU for the VLAN sub-interfaces as well as on `Host A` and `Host B`.

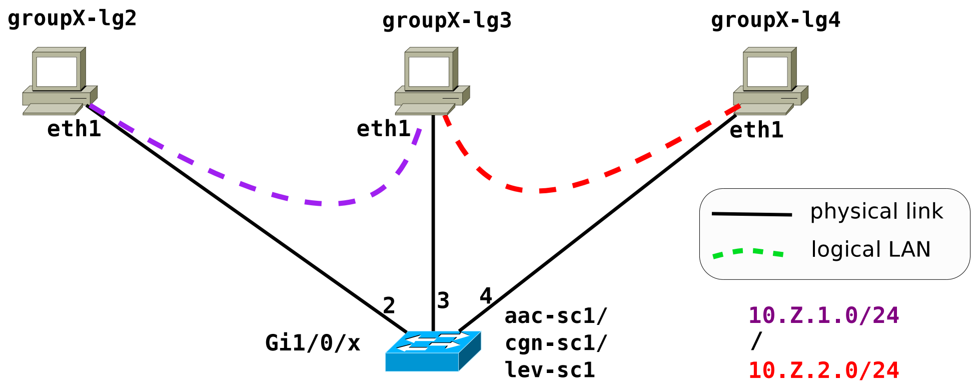

Figure 1: Topology for Questions 1-4.

Question 2: (8 + 4 + 4 + 9 = 25 Points) *Packet Filtering using IPtables*

In general, you don't want to deploy this setup on the edge of the real Internet. After all, the Internet *is* a dangerous place, with all kinds of malicious parties trying to get hold of your precious data. So you may want to control who sends which data to whom. Luckily, Linux is exceptionally well-equipped for this challenge: the Netfilter framework and its front-end tool `IPtables`.

(a) Briefly answer the following questions (around 5 sentences each):

- What is the relationship between Netfilter and IPtables?
- What is the difference between a table and a chain?
- What is a chain policy?
- Which tables are supported by default on your loadgens?

(b) To validate our firewall skills, we will use the simple test service `echo`. `echo` is a very basic TCP service that will just echo back anything that it receives line-by-line. Echo runs on TCP port 7. To activate this service on our loadgens, install the *Internet Superserver* `xinetd` in the usual manner. You will have to tweak the config file `/etc/xinet.d/echo` to enable the service and then restart the `xinetd` (`/etc/init.d/xinetd restart`). Now check with telnet on `Host A` that you can connect to the echo service on `Host B` and provide a few lines of output. Once you can `echo`, continue on.

(c) Install and familiarize yourself with the `iptables` tool on your *Linux-Router* loadgen. You should now be able to configure a simple packet filter on *Linux-Router* that controls which traffic is forwarded between *Host A* and Host B (and vice versa). A helpful hint: You can very quickly export and import entire rulesets with the `iptables-save` and `iptables-restore` commands.

(d) Implement the following iptables filters and demonstrate their function by providing a "before/after" tcpdump trace and provide us with the rules you used. Only implement one configuration at a time, i.e., clean the previous configuration before you start working on a configuration.

- *echo* is blocked in both directions
- Block everything except ICMP in both directions.
- Ping only works from Host A to Host B (you will have to filter ICMP types echo-request and echo-reply appropriately).
- Everything except `ssh` connections initiated from `Host A` to `Host B` is blocked.

Question 3: (8 + 4 + 8 = 20 Points) *Stateful Inspection*

We are now able to block specific services based on their transport layer port numbers. However, having a separate rule for every single service in a network is cumbersome, especially for services such as DNS that have a lot of seemingly unrelated UDP packets flying around. Rather we want to allow a *Trusted* network on one side of our firewall to talk to the Internet, while the *Untrusted* side should only be able to answer valid requests from the trusted side. This feature can be achieved quite elegantly with *stateful inspection*. **For each part**, provide us with your iptables configuration using iptables-save.

(a) Let `Host A` be in our trusted home network, and `Host B` be in the untrusted network. Implement a stateful rule set on `Linux-Router` that allows Host A to access `ssh`, DNS and a web server on `Host B`, but NOT vice versa. Use stateful inspection to guarantee that only valid answer packets pass the firewall. That is, any packet not SSH, DNS or HTTP should be blocked by the firewall. Use `IPtables LOG` target to log all dropped packets to the syslog, and provide us with the lines from the syslog which correspond to dropped packets for each of these three services. Keep this firewall state in place for the next part.

(b) Install an ftp server on Host B such as `vsftpd`. Create a text file `test.txt` on the ftp server in the path "/srv/ftp".

Now, connect via *FTP* (using the `ftp` client) from `Linux-Router` to `Host B`. Download (`get`) the file `test.txt` from the default directory and provide us the output from the ftp program.

Next, install an ftp client on Host A and connect via *FTP* (using the `ftp` client) from `Host A` to `Host B`. The download should fail – explain why. Using `iptables`, try to find an elegant solution to this problem that allows an ftp download to succeed, without allowing any other traffic to reach `Host A`. (Hint: look at the different *conntrack* helper modules available.) Provide us with the *iptables-save* output of this configuration.

(c) To make sure that our firewall is as tightly closed as possible, keeping the configuration so far installed, we will now launch a port scan from `Host A` to `Host B`. We will use the tool `nmap` for this task. Launch a port scan from `Host A` against `Host B` that scans all TCP ports up to 1024. Provide us with the output of your nmap port scan and explain what it means. `nmap` can try to guess what OS is running at a certain network address, try this feature against `Host B` and report what nmap finds.

Question 4: (4 + 2 + 3 + 2 + 8 + 3 + 8 = 30 Points) *Traffic Shaping*

That concludes our Linux firewall-specific experiments. Please make sure that you flush all firewall rules and that the chain policies are set to `ACCEPT` before continuing. We will now investigate the traffic shaping features offered by the Linux kernel, i.e., how to limit the amount of bandwidth for individual traffic classes. Familiarize yourself with the traffic shaping features offered by Linux. Take a look at the Linux Traffic Control HOWTO at `http://tldp.org/HOWTO/Traffic-Control-HOWTO/`.

Note: If the file transfer commands used in this exercise stall, check if reducing the MTU size on host A or host B resolves the issue.

(a) In a few sentences, explain what `tc` and qdisc are and what the difference between classful and classless qdisc is. Give an example of each. Explain how iptables and tc can be used together to classify and shape traffic.

(b) Explain how `iperf` can be used to estimate the bandwidth between Host A and Host B.

(c) Estimate the average bandwidth between Host A and Host B.

(d) Estimate the average bandwidth between Host A and Host B using `iperf` while transmitting a 100 Mb file via `scp`. Please ensure that the average is computed using at least three measurements.

(e) We now want to limit the bandwidth between our two test hosts to 300 Mbps. Do not modify `Host A` or `Host B` in this part of the assignment, but explain how to configure the `Linux-Router` using `tc`. Instead of executing `tc`, simply list (in order) the sequence of commands to be executed. (Hint: Use `tc` with queuing discipline `tbf`.)

(f) Now list (in order) the sequence of commands to set up traffic shaping on the `Linux-Router` to emulate the bandwidth characteristics of an ADSL line (`Host A` being the customer). What is the up/downstream bandwidth of a typical ADSL line? Indicate the values you choose.

(g) Linux offers traffic prioritization in order to minimize the impact that different traffic classes have on each other. Familiarize yourself with HTB *(Hierarchical Token Bucket)* by reading the HOWTO at `http://tldp.org/HOWTO/Traffic-Control-HOWTO/classful-qdiscs.html`.

List the sequence of commands (in order), purely on `Host A`, to set up class-based traffic shaping that minimizes the impact of the concurrent upload on the download without interfering with the upload bandwidth.

Question 5: (8 + 6 + 6 = 20 Points) *Cisco and Juniper ACLs*

Many mid- to high-end switches and routers provide traffic filtering and access control list functionality as well. We will now explore some of these features in our devices. As we are introducing new devices, we need to modify our topology slightly. Take a moment to reconfigure your topology according to Figure 2. You may use static routing as well as default routes where possible. Provide a traceroute to ensure that you can ping groupX-lg4 from groupX-lg2 through rj2.

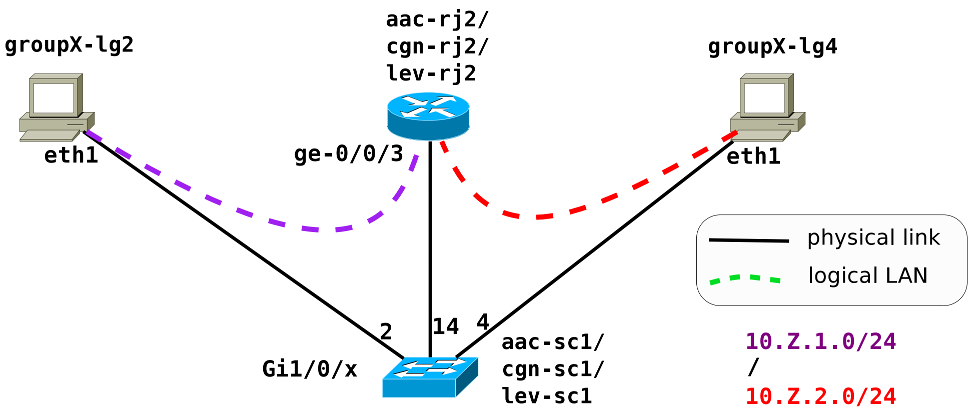

Figure 2: Topology for Question 5.

(a) To familiarize yourself with access control lists on Cisco devices, have a look at the first chapter of the Cisco Security and ACL configuration document at http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/sec_data_acl/configuration/xe-3s/sec-data-acl-xe-3s-book.pdf.

Cisco switches (depending on the IOS version, of course) support many different ACL mechanisms for different purposes. Among these are Standard, Extended, and Reflexive ACLs, Lock and Key Dynamics, and CBAC. Provide a table or list giving each of these types, a short description of its function and abilities and an example command you could use to configure it on our switches.

(b) In this question, we look into the configuration of a reflexive IP access list that would be necessary on aac-sc1 (cgn-sc1 / lev-sc1) to block all IP traffic except for ssh connections established from Host A to Host B.

Unfortunately, aac-sc1 (cgn-sc1 / lev-sc1) does not support outbound access-groups, a feature necessary for what we intend to do. Hence, exceptionally, simply write down the necessary succession of commands that you would need to run on aac-sc1 (cgn-sc1 / lev-sc1) to obtain the correct reflexive IP access list but do not attempt to apply these commands onto aac-sc1 (cgn-sc1 / lev-sc1).

(c) Now, to familiarize yourself with access control lists on Juniper devices, typically referred to as stateless firewalls, consult the Juniper Firewall Configuration documentation at

http://www.juniper.net/documentation/en_US/junos12.3/information-products/pathway-pages/config-guide-firewall-filter/config-guide-firewall-filter.pdf.

On your Juniper router, configure a simple firewall to block all IP traffic, except for icmp requests from Host A to Host B and icmp replies from Host B to Host A.

Coming up next week: Tunnels and VPNs.

Submission details (more in ISIS):
Please submit an archive (.tar.gz or .zip) containing *a directory*, which contains all files you want to submit. Please have *your group number* in the file name and the directory name.
A report (one single PDF file, named *worksheet(num)-group(num).pdf*) containing the following elements is mandatory:

- Your group number on the first page
- Topology map with relevant routers, switches, *loadgens*, and interfaces, IPs and subnet masks (CIDR).
- For each question, the written answers with the **relevant** portions of output from all commands such as `ping`, `tcpdump`, etc in a text format. **No** screenshots of terminal windows are accepted. For ping 3-4 lines of ping requests are usually sufficient.
- For each question all commands needed to configure the *loadgens*.
- For each question all **changed parts** in the configuration of routers and switches (differences to the default config).
- **Never** include the full verbatim switch or router configuration in the pdf report.
- For all questions, state your assumptions, say what you did, describe what you observed, explain your conclusions.

Additionally, please include your config files in the archive.
For each question, please provide the full switch and router configuration in a separate text file named after the device and question, e.g.: *q01-config-sc1.txt*. This makes it easier for us to reproduce your configuration and understand what you did.
We can only grade what we find in your submission and what we understand. Please state your assumptions and observations as clearly as possible.
Due Date: Thursday, June 7th, 11:55pm