DES352 Lab Summary

Networking Laboratory I SIIT DE Y3T2/2021 – By Paphana Yiwsiw (@waterthatfrozen)

Lab 1 & 2 - Linux Command

`cd [directory]`	enter the directory
`pwd`	show the current directory
`ls`	list out contents in the current directory
	`ls -l` to list out contents in full detail (with permission etc.)
`history`	show history of commands entered
`help [command]`	show help of that command
`man [command]`	show the manual and documentation of that command (press q to exit)
`mkdir [directory]`	create a new directory
`cp [org] [dest]`	copy file from origin to destination
`mv [org] [dest]`	move file from origin path to destination, can also be used to rename a file
`rm [filename]`	remove file from directory
	`rm -r` remove file and directory (-r = recursive)
`less [filename]`	program to read the content in a file.
`echo [text]`	print out text into the terminal
`chmod [mode] [file]`	change the permission of the file (rwxrwxrwx = 777)
`A > B`	redirect standard output from command A to file B (overwrite)
`A >> B`	append the standard output from command A to file B
`A 2> B`	redirect standard error from command A to file B
`A | B`	pipelines, output of command A will be input of command B
`nano`	start text editor running on terminal
`gedit`	start text editor running on desktop
`cat`	concatenate file together
`sort`	sort **lines** in a text file in numerical then alphabetical order
`uniq`	omit repeated lines in a file
	`uniq -c` also count the number of repeated lines
`wc`	show number of lines, words, size in bytes of a file
`ps aux`	show the list of running processes
`CTRL-C`	interrupt a process and terminate it
`[process] &`	run a process run in background
`fg %[job number]`	bring a process (use its job number) to a foreground.
`CTRL-Z`	stop/pause a process, noted that it is **not** terminate a process
`kill [PID]`	terminate a running process (use the PID: Process ID)

```
#!/bin/bash                     begin of the script
varName = 1                     declare variable in script
echo “value is $varName”        use variable with $ / use input argument as $0, $1, …
if [ condition ] ; then         if-else statement (beware space inside [])
      echo “condition true”     conditions: -eq(=)|-ge(>=)|-gt(>)|-le(<=)|-lt (<)|-ne(!=)
else
      echo “condition false”
fi                              end of if-else statement

chmod 755 [filename]            make the script runnable
./[filename]                    run the script
```

Lab 3 - NIC and ARP

`ifconfig` view and change the network interface configuration on the system.

`ifconfig -a`
view **all** network interface configurations (both up and down).
`ifconfig [NICName]`
view a **specific** network interface configuration (specified by the name).
`sudo ifconfig [NICName] [up/down]`
enable or disable the network interface
`sudo ifconfig [NICName] [IP Address]`
assign the **static** IP address of the network interface

`ping [IP Address]` check the connectivity status and quality between source and destination, and estimate round-trip time (RTT) of a response from the network.
ping uses ICMP request/response to send/receive message.
ping can be used to check whether destination is reachable or not.
ICMP header has 8 bytes. IP header has 20 bytes.
TTL specifies the number of hops that a packet can go before being discarded.

`ping -i [interval] [IP Address]`
change the interval between each ping packet. Default is 1 second.
`ping -c [number] [IP Address]`
send the specific amount of ping packet to the destination.
`ping -s [size] [IP Address]`
change the size of ping packet. Default is 56 bytes.
You can combine -i, -c, and -s together in a single command. e.g., `ping -i 2 -c 10 -s 50 192.168.1.100`

`arp` used to find the MAC address of a device from IP address. It is used when one device wants to send packets to another on a local network.
ARP cache is an ARP table which maps MAC address to corresponding IP address.

`arp -a`
show the complete ARP cache, including the host name
`arp -n`
show the ARP cache, excluding the host name
`arp -a [IP Address]`
find the MAC address from the specified IP address in the ARP cache
`arp -i [NICName]`
show the MAC address of IP addresses connected to specific NIC in the ARP cache
ARP entity can be added into ARP cache by ping to that IP address you want to add.

Lab 4 - TCPDUMP and Wireshark

`tcpdump` used to capture packets sent/received via a specific NIC

tcpdump is a CLI: command line interface

`tcpdump -h`

check that tcpdump is installed or not.

`sudo` `tcpdump -D`

view a list of NICs in the system

`sudo` `tcpdump -i [NICName] -w [filename]`

capture network traffic on NIC and write to a file (.dump), press CTRL+C to stop and save.

`sudo` `tcpdump -r [filename]`

read the captured network traffic file (.dump)

`sudo` `tcpdump -i [NICName] -c [number] -w [filename]`

capture a specific number of packets on NIC and write to a file

`sudo` `tcpdump -i [NICName] -c [number] -w [filename] [protocol]`

capture a specific number of packets on NIC that filtered only the specific protocol and write to a file.

You can generate network traffic by open a website, ping, etc.

Basically, -i is interface name, -w is write, -r is read, and -c is count.

`wireshark` GUI for packet capture and analysis tool.

Wireshark and tcpdump are also known as "packet sniffers".

`sudo` `wireshark`

start Wireshark

If you want to open wireshark in IMUNES, you need to right-click on computer then select wireshark.

`Filtering in Wireshark`

You can filter to display packets that fall in the criteria such as protocol, source IP, and destination IP.

Comparison operator: `==, !=, >, >=, <, <=`

equal, not equal, greater, greater or equal, less, less or equal

Logical operator: `&&, ||, !`

and, or, not

IP filtering: `ip.addr, ip.src, ip.dst`

packets from this IP address (both src and dst), send from IP address, send to this IP address

Protocol filtering: `tcp, udp, arp, icmp, dns, …`

use protocol name as filter such as tcp, udp, arp, icmp, dns, etc.

Frame filtering: `frame.len`

packet with specific frame length

Example filter:

`arp || icmp`

show packets from arp or icmp protocol only.

`ip.src == 192.168.1.100 && ip.dst == 192.168.1.1`

show packets sent from IP address 192.168.1.100 and sent to 192.168.1.1

`frame.len <= 100`

show packets that have frame length less than or equal to 100 bytes

`http.host == www.tu.ac.th`

show packets that have host name of www.tu.ac.th

Lab 5 - Essential Networking Commands

`dig`	query DNS name servers
`hostname`	view the host name of your computer
`sudo` `hostname [newName]`	rename the host
`host [domainName]`	find the IP address of the domain name
`nslookup [domainName]`	DNS lookup, find the IP address of the domain name.

`route -n` — show routing table with IP addresses.

If you want as host name, do not include -n.

In routing table, 0.0.0.0 means not specified, flag U is up, flag G is gateway.

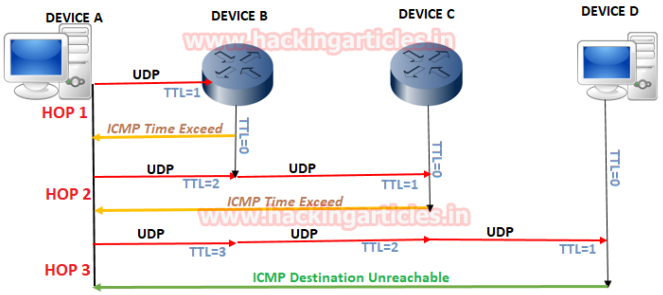
`traceroute [IP address]` — show paths that packets take and number of hops to destination

It works by sending 3 UDP probe packets and listen for ICMP reply packets.

If ICMP reply is ICMP timeout, it means doesn't arrive at destination yet.

If ICMP reply is ICMP destination unreachable, it means it arrived at destination.

Number of hops are TTL in ICMP packets, it gets increasing by 1 after each timeout.



`traceroute -n [IP address]`

normal traceroute but it shows only IP addresses, hide the hostname.

`sudo` `traceroute -T [IP address]`

change from UDP probe packets to TCP probe packets

`netstat` — monitor incoming and outgoing network connections, routing table, interface statistics, etc.

`netstat -s`

show statistics by protocols. Default shows TCP, UDP, ICMP, IP protocols.

`netstat -r`

show routing table

`mtr` — my traceroute, CLI for network diagnostic with both ping and traceroute.

In mtr, you can view traceroute result in real-time.

`mtr -n [IP Address]`

show my traceroute to IP address, show IP addresses in result instead of hostname (-n).

Press CTRL+C or q to quit my traceroute.

```
                    My traceroute  [v0.85]
netlab09 (0.0.0.0)                              Tue Feb 12 15:03:19 2019
Keys:  Help   Display mode   Restart statistics   Order of fields   quit
                                 Packets               Pings
 Host                          Loss%   Snt   Last   Avg  Best  Wrst StDev
 1. 192.178.18.1                0.0%    10    0.2   0.2   0.2   0.2   0.0
```

`Loss%`	packet loss at each hop, as percentage.
`Snt`	number of packets sent.
`Last`	RTT last packet sent.
`Best`	RTT best/shortest of all packets sent.
`Wrst`	RTT worst/longest of all packets sent.
`Avg`	RTT average of all packets sent.
`StDev`	S.D. of RTT of all packets sent.

RTT = round-trip time (a.k.a. latency)

Lab 6 & 7 – Internetworking I & II

IPv4 Classless Addresses Notation and Calculation

```
128.143.137.144                 Current host address
/20 → 255.255.240.0             Number of 1's in subnet mask bits (short way to write netmask)
128.143.128.0/20                Network Address of this network
                                How? Use and operation on host address and subnet mask
          128.143.137.144 → 10000000 . 10001111 . 10001001 . 10010000
                      /20 → 11111111 . 11111111 . 11110000 . 00000000
          Network Address → 10000000 . 10001111 . 10000000 . 00000000
                          →      128 .      143 .      128 .        0
2^(12) - 2                      Number of available host address on this network
128.143.128.1/20                First available host address on this network.
                                How? It is the first one after the network address
                                10000000 . 10001111 . 10000000 . 00000001
128.143.143.255/20              Broadcast address on this network.
                                How? It is the first one after the network address
                                10000000 . 10001111 . 10001111 . 11111111
128.143.143.254/20              Last available host address on this network.
                                How? It is the one before the broadcast address
                                10000000 . 10001111 . 10001111 . 11111110
```

Online IP calculator: https://www.calculator.net/ip-subnet-calculator.html
My technique of assigning IP addresses in the network: The router or gateway to connect that network to the outside, I will assign to be the first available host address.
For example, If the network address is 192.168.1.0/24. My router host address is 192.168.1.1/24
If more than one gateway, the second one will be the next address, one after, and so on.

Network diagram

- **Logical** network diagram — show just how network connect to each other. Display the network address only.
- **Physical** network diagram — show the exact topology with all devices and the connections. Display all information inside the network: (Host IP, router, etc.)

ENABLE IP FORWARDING AND FIREWALL ON COMPUTER (If connecting with physical computer, not IMUNES)

sudo `sysctl -w net.ipv4.ip_forward = 1 |` **sudo** `iptables -P FORWARD ACCEPT`
enable the IP forwarding firewall so computer can work as a router.

sudo `sysctl -a | grep ip_forward`
check whether IP forwarding is enabled or not.

sudo `iptables -L`
check firewall in FORWARD chain (ACCEPT)

Routing command

sudo `route add default gw [IP address]`
add the default gateway IP address in the routing table

sudo `route add -net [Network address] gw [IP address]`
add the static gateway IP address in the routing table to relay any packets sent to the network address.

sudo `route del default`
delete the default gateway/router

sudo `route del -net [Network address]`
delete the static gateway assigned to the network address from the routing table.

Don't forget how to set the IP address to the NIC (Lab 3/ifconfig eth0 [IP address]), and display the routing table (Lab 5/route -n)

Lab 8 – Transport Layer Protocols

Transport layer protocols responsible for **the delivery of messages to appropriate processes.**

Simulating client-server connection

`nc -luvp [Port number]`

Computer will work as a server and **UDP** connection using the specify port number

`nc -uv [IP address] [Port number]`

Computer will work as a client and makes the **UDP** connection with the server IP address via port number

nc flag options

`-l`	to specify that computer should listen to the incoming connection
`-u`	to use **UDP** connection instead of the default **TCP** connection.
`-v`	to produce more detailed information
`-p`	to specify the port number

You can combine flags into one. For example, -luvp.

Port number

- Representing process-to-process communication, client-server paradigm.
- Important port numbers that you need to know:
  - Echo:7, FTP:21, TELNET:23, SMTP:25, DNS:53, HTTP:80, HTTPS:443

Address

- Physical address/link address: from data link layer, represented by MAC address of NIC. Used for finding the destination host in LAN.
- Logical address: from the network layer, represented by IP address. Used for internetworking and find the destination host in another network.
- Port address: from the transport layer, represented by the port number. Used for finding the destination program or process in the destination host.
- Socket address: Transport layer protocol in TCP need both IP address and port number on each end to make a connection. Then, it is the combination of an IP address and a port number. Ex. 200.23.56.8:69

Transport layer protocols (in this lab)

- UDP: User Datagram Protocol
  - Connectionless, unreliable transport protocol.
  - Suitable for sending message without reliability concern.
  - No connection establishment and connection termination.
  - Each datagram sent by the UDP is an independent datagram.
- TCP: Transmission Control Protocol
  - Connection-oriented, reliable protocol.
  - Requires:
    - Establishment: 3-way handshaking
      - SYN (no carry, one seq), SYN+ACK (no carry, one seq), ACK (if no carry, no seq)
    - Data transfer: bidirectional data transfer can take place, both send data and ACK.
    - Termination: 3-way handshaking
      - FIN (if no carry one seq), FIN+ACK (if no carry, one seq), ACK (no carry, no seq)

Lab 9 – Application Layer Protocols

This lab mostly uses wireshark

DNS: Domain Name System

- Domain name is the name of a website
- Universal resource locator: URL, complete web address to find web page

HTTP: Hypertext Transfer Protocol

- Client will send a HTTP request message; Server send HTTP response message.
- HTTP request message: 4 sections
    - Request line: method, URL, version. Ex. `GET ./index.html HTTP/1.1`
      Most of the time, client uses the GET method to send a request. Other method is POST, etc.
    - Header line: header name (User-agent, Accept, Auth, Host, Date, etc.)
    - Blank line
    - Body
- HTTP response message: 4 sections
    - Status line: Version, Status Code, Phrase. Ex. `HTTP/1.1 200 OK`
        - 1xx Info. response / 2xx Successful / 3xx Redirection / 4xx Client error / 5xx Server error
        - 200 OK / 201 Created / 202 Accepted
        - 301 Moved Permanently / 303 See Other / 304 Not Modified / 307 Temporary Redirect
        - 400 Bad Request / 401 Unauthorized / 403 Forbidden / 404 Not Found / 410 Gone
        - 500 Internal Server Error / 503 Service Unavailable.
    - Header line: Date/Upgrade/Server/Location/Last-modified/etc.
    - Blank line
    - Body: with webpage source code.

FTP: File Transfer Protocol

- Standard protocol provided by TCP/IP for copying a file from one to another.
- FTP is a better choice to transfer large files or using different formats
- Security is the issue about the FTP protocol since it requires password but sent in plaintext.

Lab 10 – Basic Linux Firewall

- Firewall is a device (mostly router or computer) install between the internal network and the internet. Designed to **forward** some packets and **filter** others.
- A firewall can be used as **packet filter**, to forward or block packets based on the information in the network/transport layer headers.
- A filter table will list the rules and policies to drop or accept packets according to source or destination IP address and/or port, protocol, network interface.
- Filter table has 3 "chains": INPUT (incoming) /FORWARD (relay) /OUTPUT (outgoing)

Linux Firewall Commands

Show the filter table:
```
sudo iptables -nvL
```
-n: display in numeric format / -v: verbose output (show details) / -L: list all firewall chain

Delete all existing firewall rules:
```
sudo iptables -F
```

Set the default rule to specific chain:
```
sudo iptables -P [chain name] [ACCEPT/DROP]
```

Save the current firewall rules into the file:
```
sudo iptables-save > [file name]
```

Restore the firewall rules from the file:
```
sudo iptables-restore < [file name]
```

Add/insert/delete a specific rule to the filter table:
```
sudo iptables [action on chain] [chain name] [rule] [action on packet]
```

`Action on chain:`

-A	Append a new rule at the end of the chain
-I [chain-name] [position]	Insert a rule at the specified position
-D	Delete a rule
-P	Reset the default rule

`Chain name:` **INPUT / FORWARD / OUTPUT**

`Rule:`

-s [IP address / network address]	Packet is sent from this source IP or network address
-d [IP address / network address]	Packet is sent to this destination IP or network address
-p [protocol name]	Packet is using this protocol (tcp, udp, icmp,…)
-sport [port number]	Packet is sent from this source port number
-dport [port number]	Packet is sent to this destination port number
-i [NIC]	Packet is entering this NIC
-o [NIC]	Packet is sent out through this NIC

`Action on packet:`

-j ACCEPT	Accept this packet if the rule matches
-j DROP	Drop this packet if the rule matches

Firewall Strategies
- Blacklisting strategy: default is ACCEPT, insert rules to drop packets.
- Whitelisting strategy: default is DROP, insert rules to accept packets.