Network Concepts

Tran Giang Son, tran-giang.son@usth.edu.vn

ICT Department, USTH

Contents

- What's a network?
- Routing
- Network Services

What?

What?

- Collection of nodes and links that connect them
  - Hosts: endpoints

What?

- Collection of nodes and links that connect them
  - Hosts: endpoints (laptop, phone, PC, game console...)

What?

- Collection of nodes and links that connect them
  - Hosts: endpoints (laptop, phone, PC, game console...)
  - Links: cables

What?

- Collection of nodes and links that connect them
  - Hosts: endpoints (laptop, phone, PC, game console...)
  - Links: cables (fiber, copper, radio...)

What?

- Collection of nodes and links that connect them
  - Hosts: endpoints (laptop, phone, PC, game console...)
  - Links: cables (fiber, copper, radio...)
  - Packet switches: forward data

What?

- Collection of nodes and links that connect them
  - Hosts: endpoints (laptop, phone, PC, game console...)
  - Links: cables (fiber, copper, radio...)
  - Packet switches: forward data (switches, routers...)

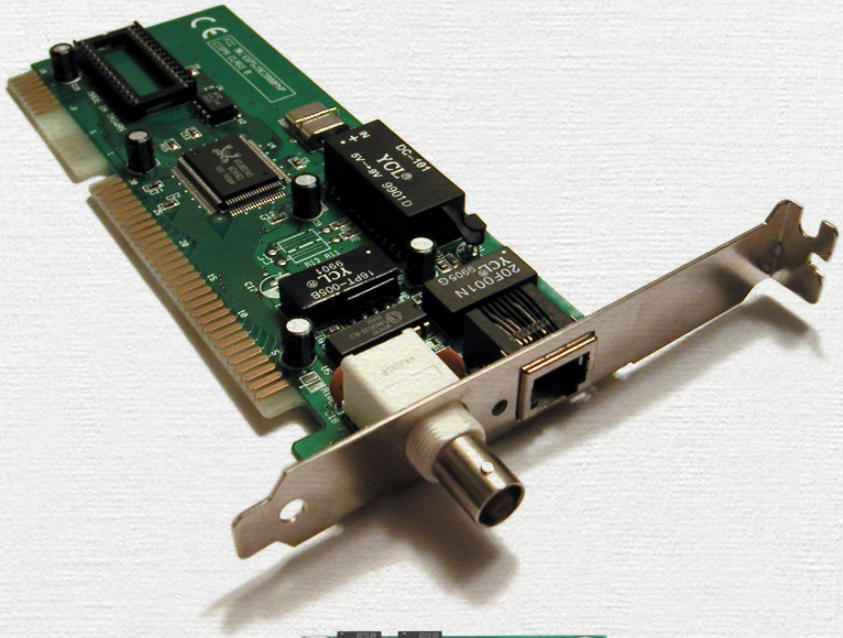
What?

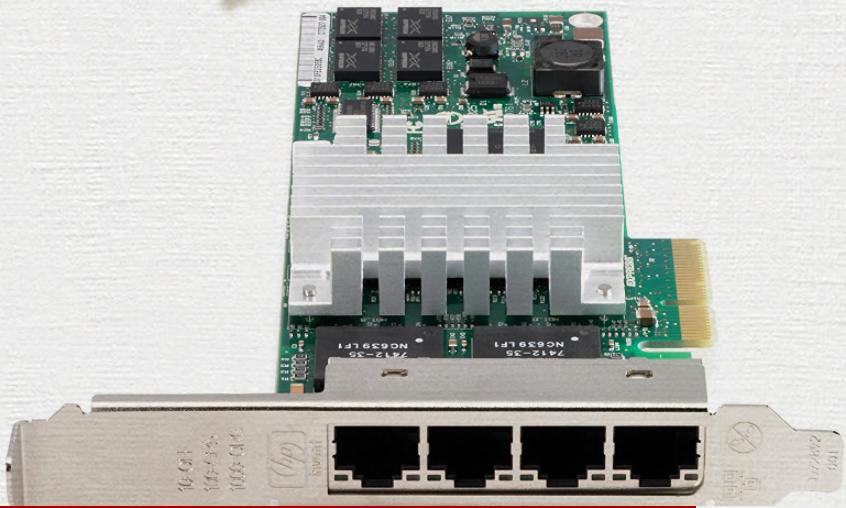
- Collection of nodes and links that connect them
  - Hosts: endpoints (laptop, phone, PC, game console...)
  - Links: cables (fiber, copper, radio...)
  - Packet switches: forward data (switches, routers...)
- A network can belong to another network

What?

- Network interface controller
  - Enables a host to transfer data
  - Examples
    - Ethernet card
    - Infinibard card
    - USB WiFi
    - WiFi card

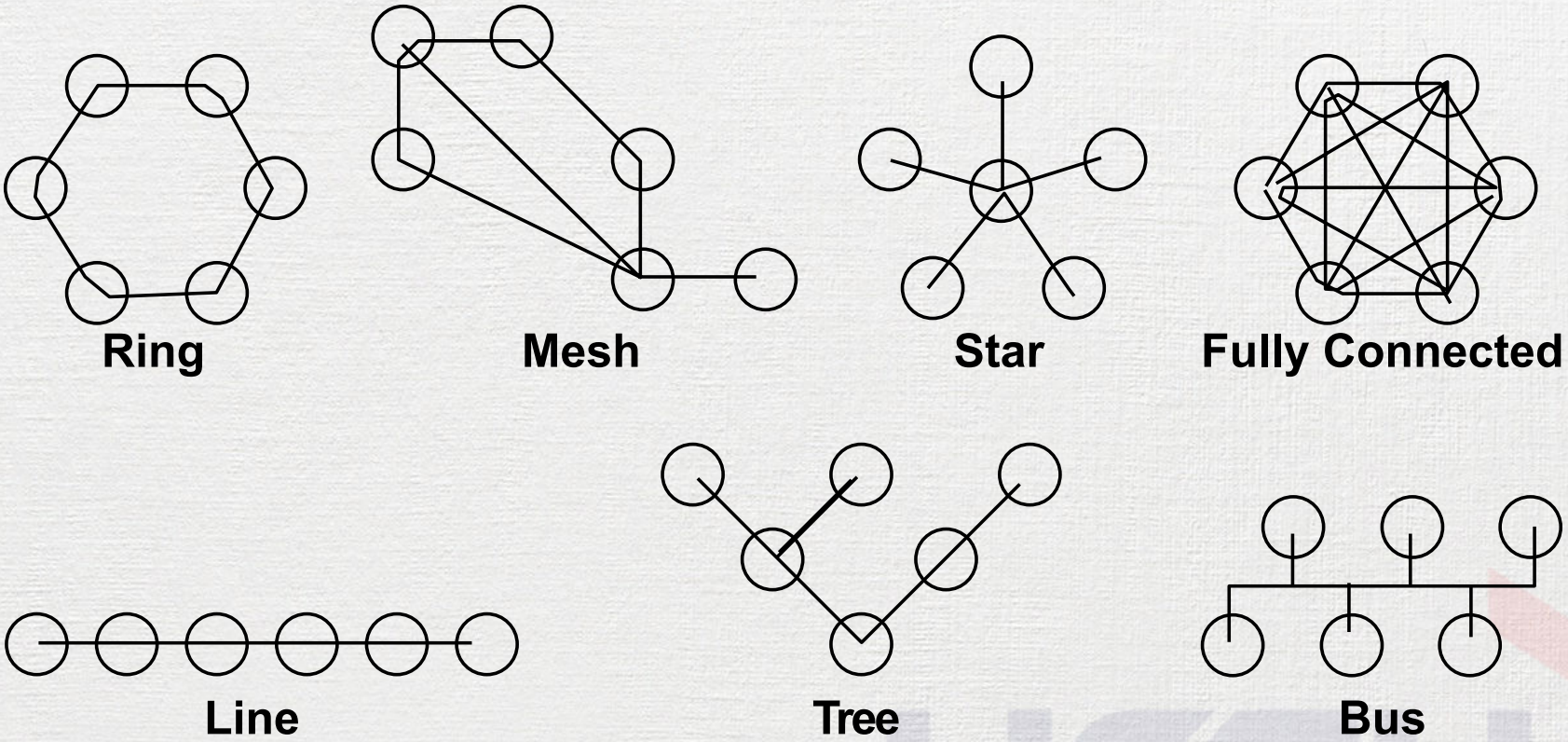
What: NIC examples





What: Network topology

Layout and organization of nodes



What?

Different scales

- Personal...
- Local...
- Metropolitan...
- Wide...

... Area Network

What?

Different scales

- Personal...
- Local...
- Metropolitan...
- Wide...

... Area Network

- Inter-net

The Internet

- An inter-net: a network of networks
  - A set of networks that are connected with each other
  - Networks are connected using routers that support communication in a hierarchical fashion
  - Often need other special devices at the boundaries for security, accounting, ...
- A common set of rules for Inter-operation

Why?

- Keep connected with other people
  - Facebook
  - Flickr
  - Youtube
- Larger set of information

Why?

- "Combine'' a set of separated resources to make something bigger
- Super computers

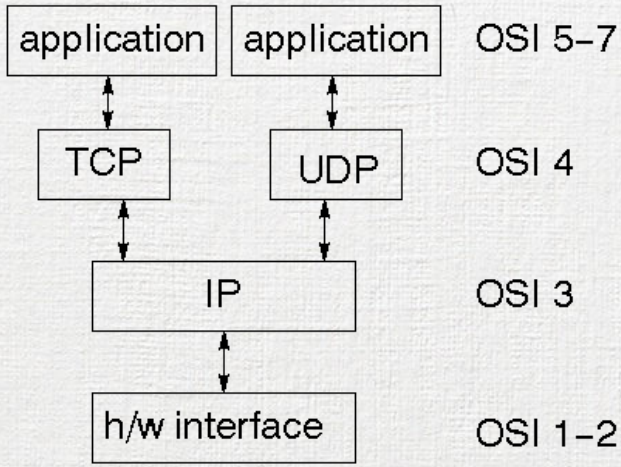
A bit of history

- Who first created it?
  - Military Radar System «Semi Automatic Ground Environment»
  - Early initiatives in 1950s
- ARPANET
  - The base of Internet
  - IP Protocol

Routing

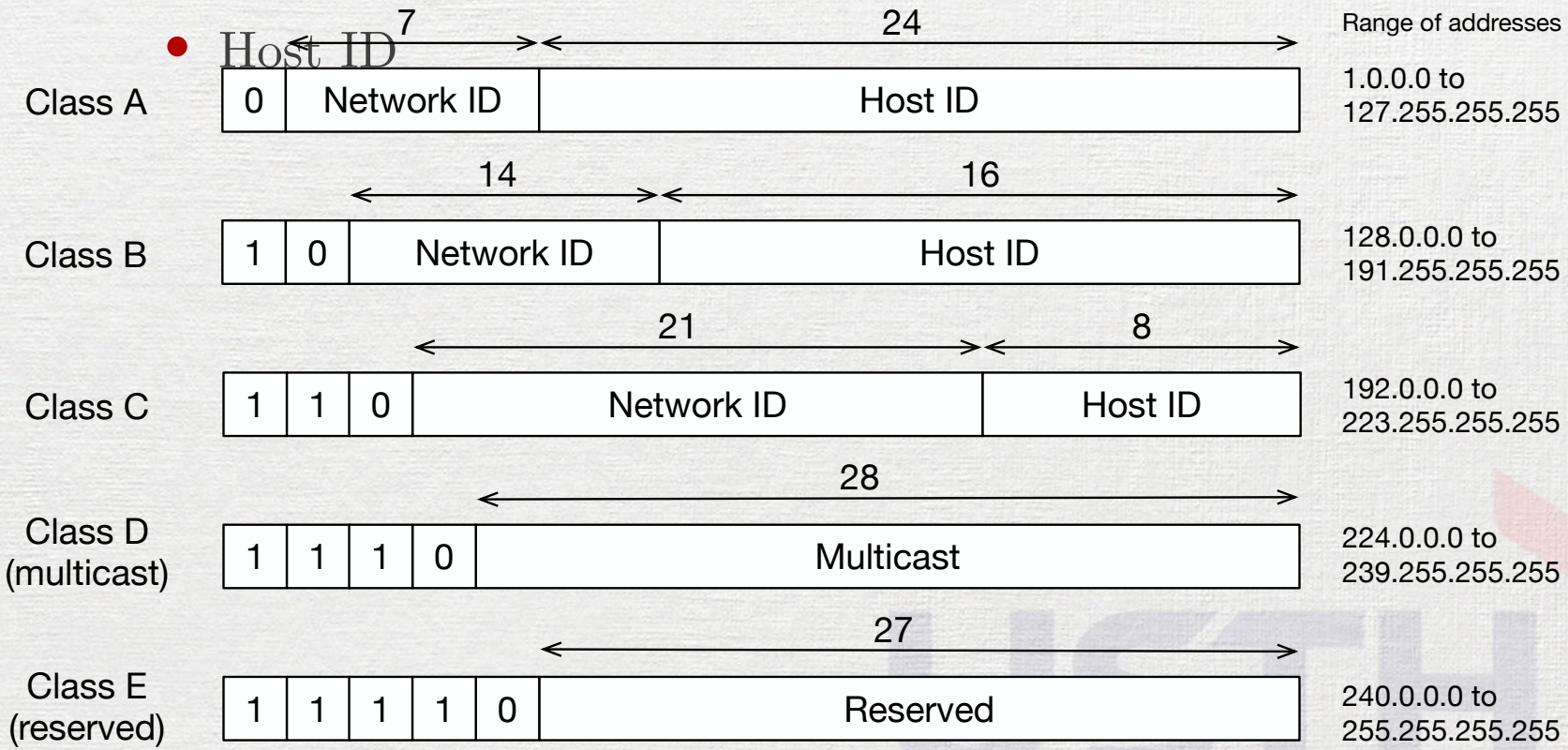
Basic Concept

- Main concepts:
  - MAC Address
  - IP Address: v4 / v6
  - IP Protocols: TCP / UDP
- Routing is transparent



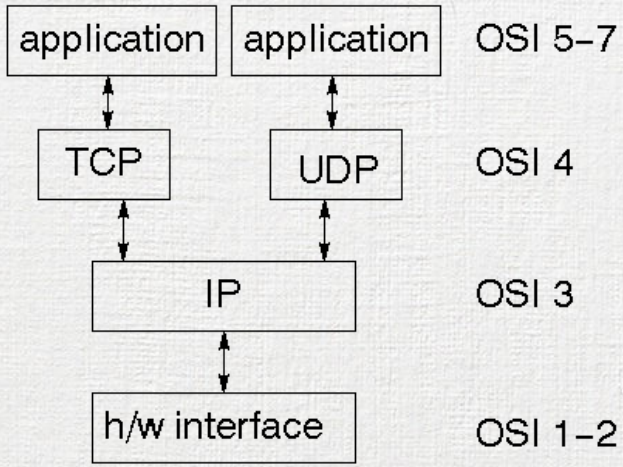
Basic Concept: IPv4

- 32-bit addresses, split to 4 bytes (0-255 each)
- Two parts:
  - Network ID
  - Host ID



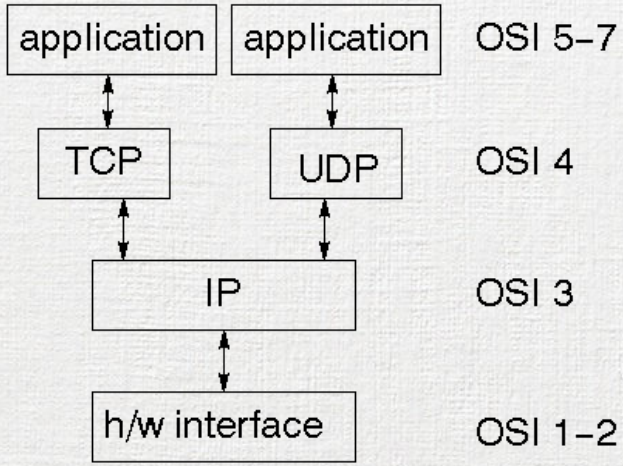
Basic Concept: UDP

- User Datagram Protocol
- Just like any post letter
  - no acknowledgements
  - no retransmissions
  - possible out of order and/or duplicates
  - connectionless: each packet needs destination



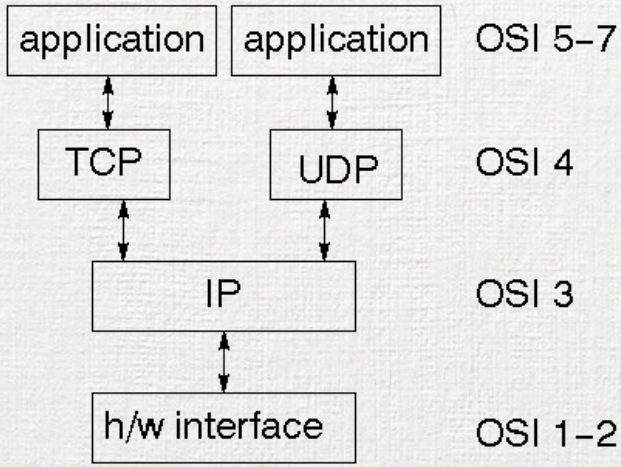
Basic Concept: TCP

- Transmission Control Protocol
- Like a phone call
  - connection-oriented: needs a «connection establish» step
  - bidirectional
  - reliable byte-stream channel
  - in order
  - all arrive
  - no duplicates
- Similar to file access



Basic Concept: TCP vs UDP

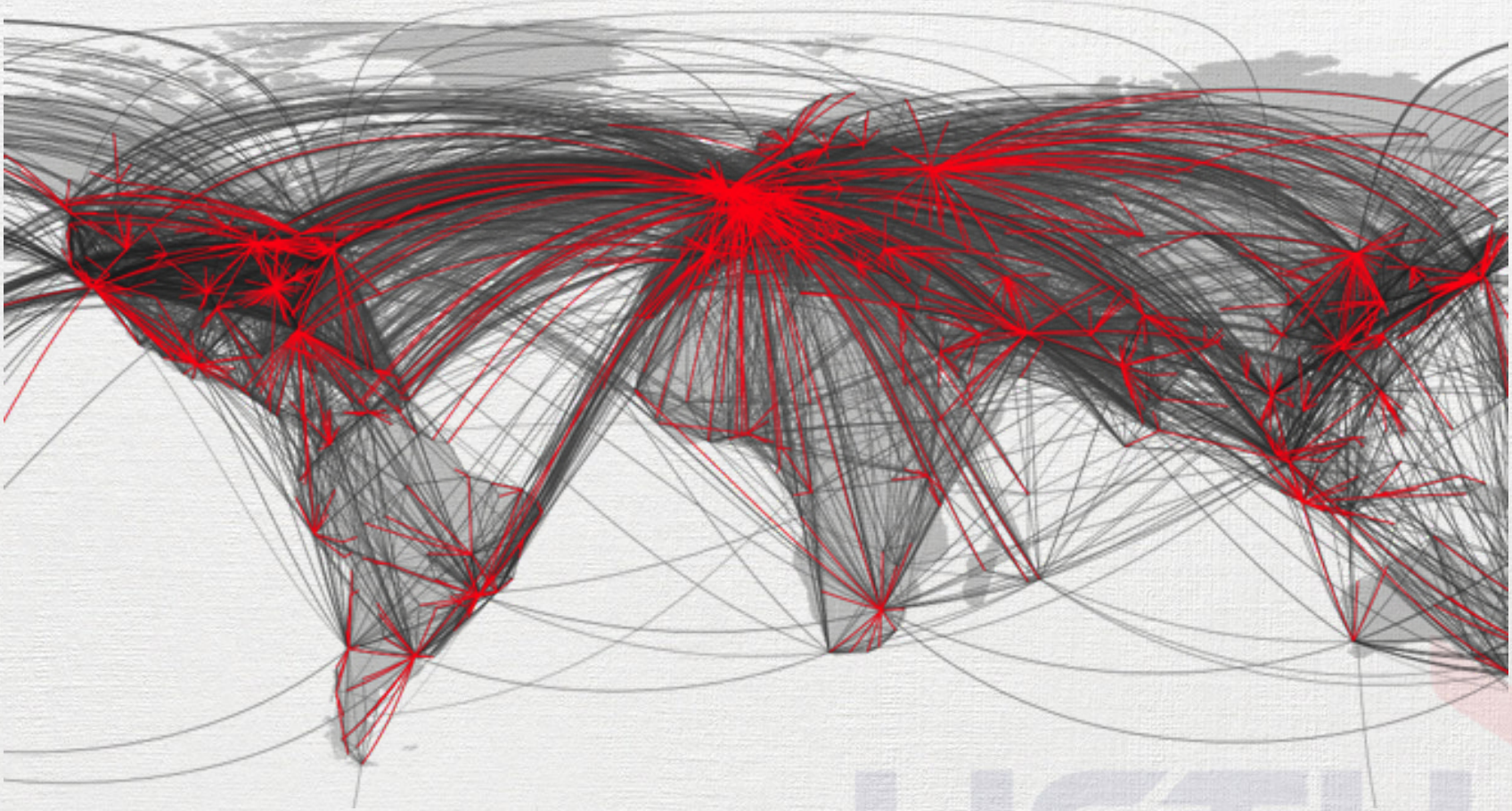
- TCP: slower, suitable for
  - Large data
  - Persistent connection
  - Reliable
- UDP: faster, suitable for
  - Quick lookup
  - Single use query-reply



Routing: What, Why and How

- What: Process of selecting path for a network packet
- Why: Without routing, one cannot send/receive message to another host
  - No direct communication link
- How?
  - Packet forwarding
  - Routing table
- Routing vs Bridging?

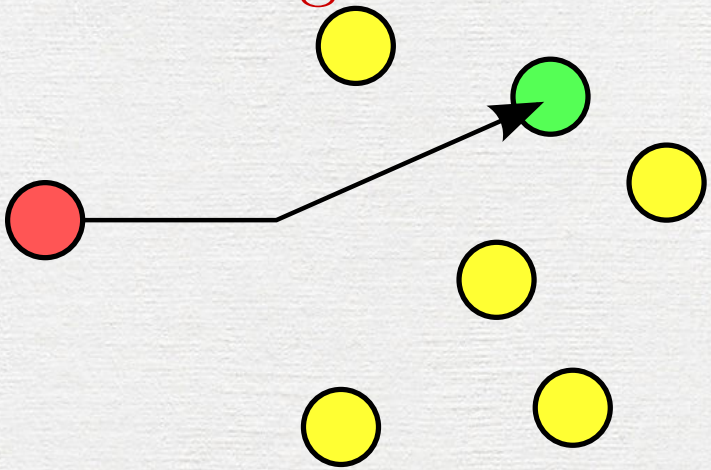
Routing



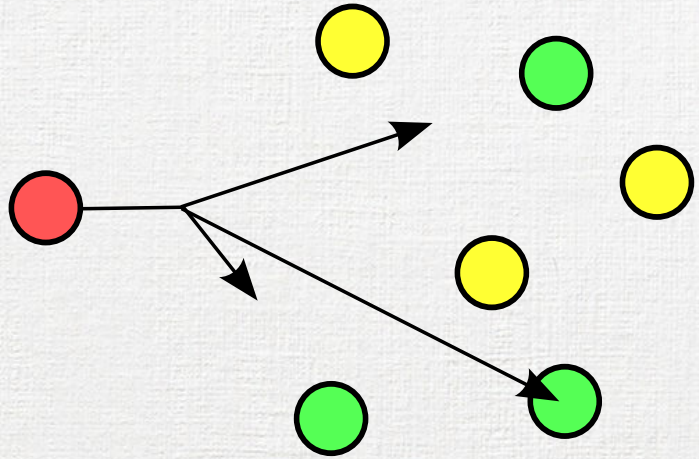
How: Routing Schemes

- Unicast: to a single node
- Anycast: conditional to anyone, typically closest nodes
- Multicast: to many nodes
- Broadcast: to all nodes

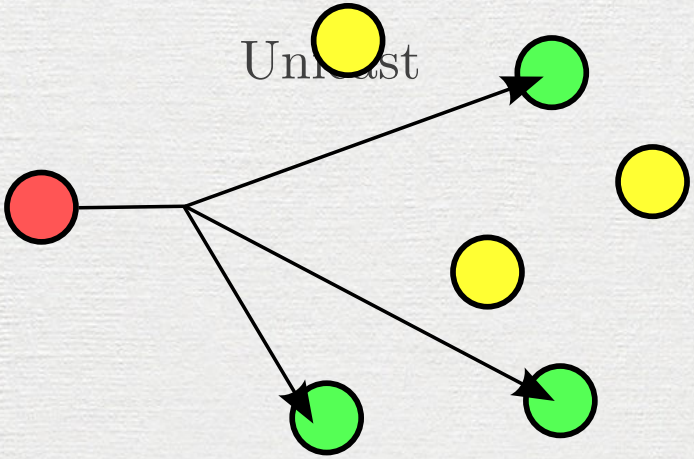
How: Routing Schemes



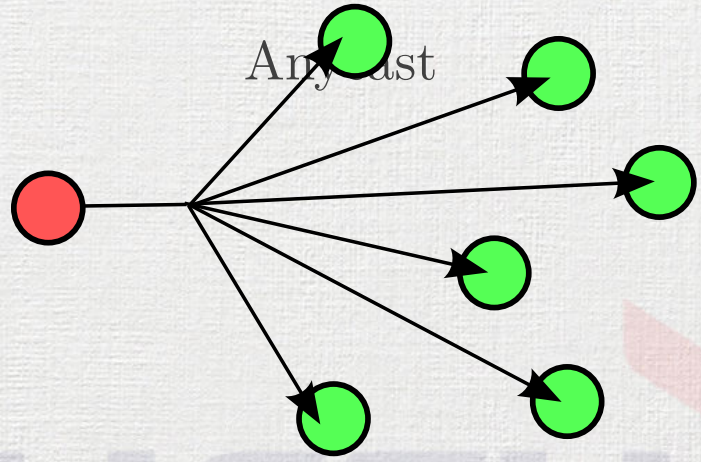
Unicast



Anycast

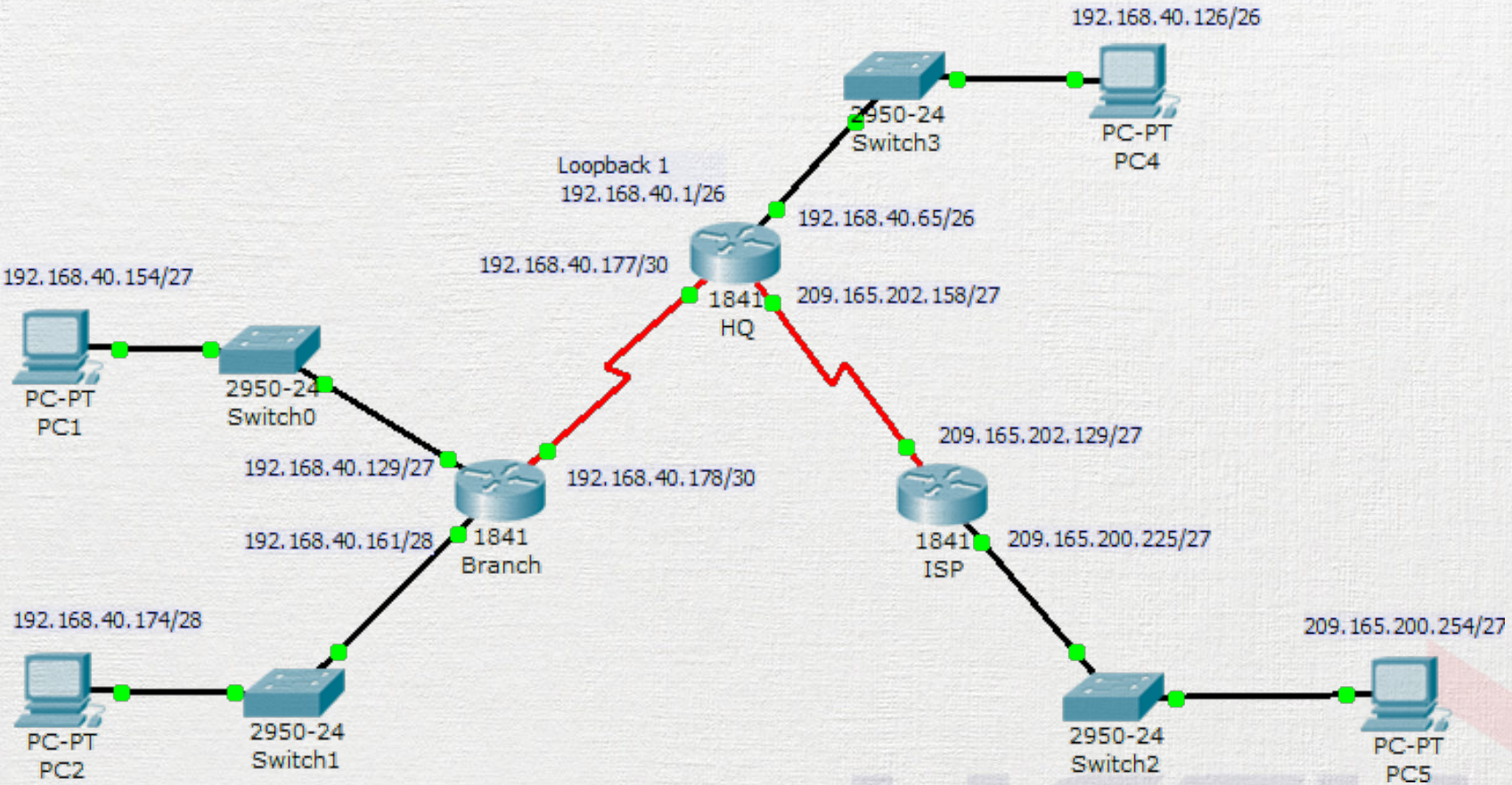


Multicast



Broadcast

Example


192.168.40.126/26
2950-24
Switch3
PC-PT
PC4
Loopback 1
192.168.40.1/26
192.168.40.65/26
192.168.40.177/30
1841
HQ
209.165.202.158/27
192.168.40.154/27
PC-PT
PC1
2950-24
Switch0
209.165.202.129/27
192.168.40.129/27
192.168.40.178/30
192.168.40.161/28
1841
Branch
1841
ISP
209.165.200.225/27
192.168.40.174/28
209.165.200.254/27
PC-PT
PC2
2950-24
Switch1
2950-24
Switch2
PC-PT
PC5

Practical Work 1: VPS Setup & Routing

- Create an account at Google Cloud
- Register for **free** credits
- Once you got your free credits, on Google Cloud Console:
  - Create a new project
  - Create a 1st generation machine instance, type `f1-micro`, Debian/Ubuntu OS.
  - Make SSH key authentication works
- Create a new report named «01.practical.work.vps.routing.md»

Practical Work 1: VPS Setup & Routing

- Write your commands & their corresponding outputs to your report for the following tasks
  - Connect to your shiny & beautiful VPS with `ssh`
  - Install `traceroute` tool
  - Check if `usth.edu.vn` is up or not with `ping` (5 times only)
  - Use `traceroute` tool to find the route from your VPS to `usth.edu.vn`
    - How many hops do you have?
  - Try `traceroute` again, but from your own computer
    - How many hops do you have?
- Push your report to corresponding forked Github repository

Network Services

Network Services

- Any machine can support different TCP/UDP services
- Services are distinguished by "**port number**"
  - 16-bit integer, from 0 to 65535
  - 0-1023 are reserved (need root privileges to `listen`)
- Similar to a hotel hosting many guests
  - Each guest is served in a separated room
  - Each room has an unique Id

Network Services

- Server
  - Has IP address
  - Provides service on a specific port
  - “listen’’ for connections
- Client
  - Needs to know what service it is using
  - Needs to know server’s IP address and port number
  - Connects to server on a specific port

Examples

Normal ones

- HTTP:

Examples

Normal ones

- HTTP: 80

Examples

Normal ones

- HTTP: 80
- POP3:

Examples

Normal ones

- HTTP: 80
- POP3: 110

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP:

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP:

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP:

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS:

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH:

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Secured ones

- HTTPS:

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Secured ones

- HTTPS: 443

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Secured ones

- HTTPS: 443
- POP3S:

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Secured ones

- HTTPS: 443
- POP3S: 995

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Secured ones

- HTTPS: 443
- POP3S: 995
- IMAPS:

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Secured ones

- HTTPS: 443
- POP3S: 995
- IMAPS: 993

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Secured ones

- HTTPS: 443
- POP3S: 995
- IMAPS: 993
- SMTPS:

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Secured ones

- HTTPS: 443
- POP3S: 995
- IMAPS: 993
- SMTPS: 587

Examples

Normal ones

- HTTP: 80
- POP3: 110
- IMAP: 143
- SMTP: 25
- FTP: 21
- DNS: 53
- SSH: 22

Secured ones

- HTTPS: 443
- POP3S: 995
- IMAPS: 993
- SMTPS: 587

Q: How do they get “secured”?

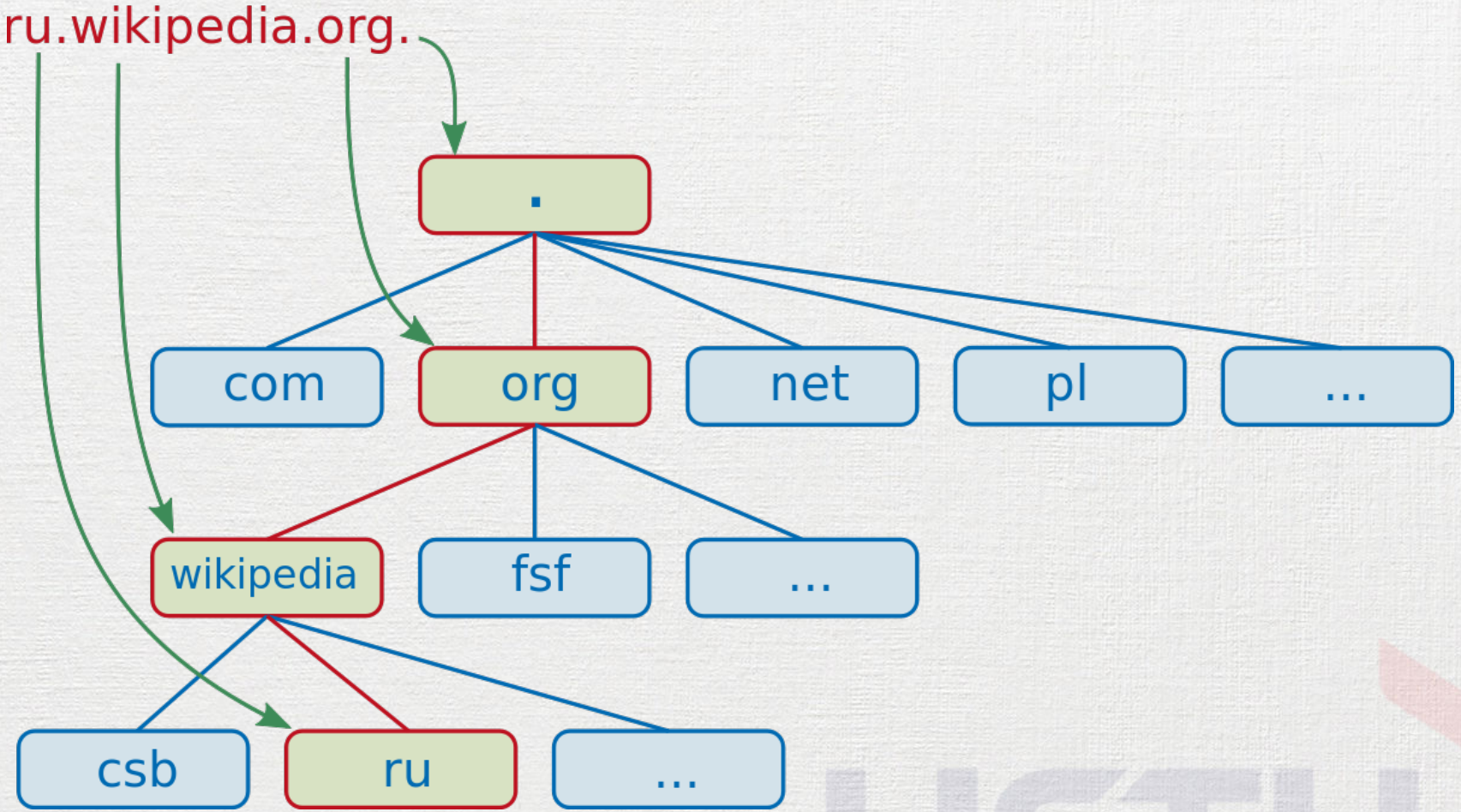
Network Services

- Remind: client
  - Needs to know what service it is using
  - Needs to know **server's IP address** and port number
  - Connects to server on a specific port
- Question: how to know server's IP address?

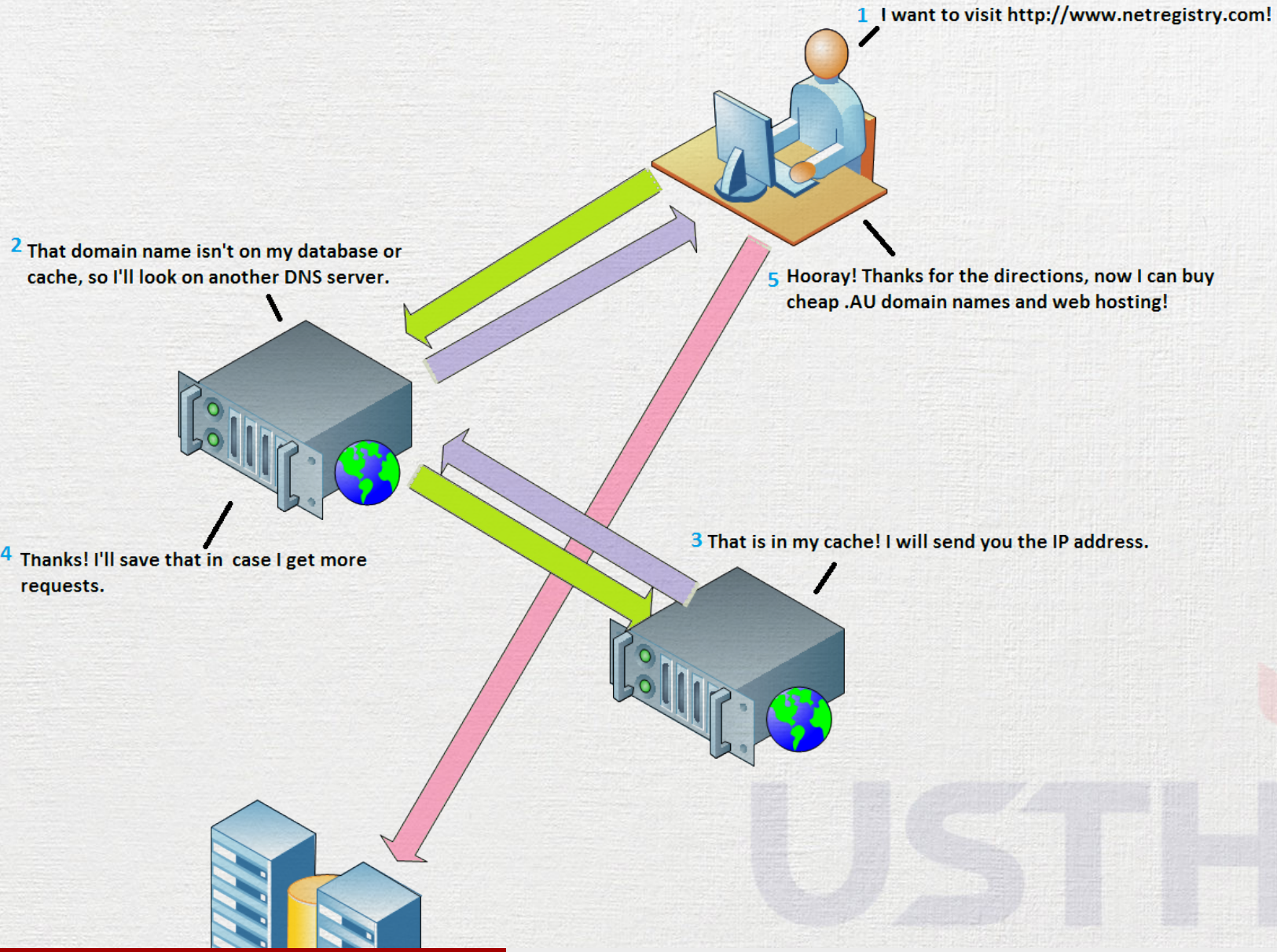
Domain Name Service

- «Convert» domain names to IP addresses
  - Just like an address book
- Use anycasting
- Hierarchical distributed naming system for computers
  - 13 root servers “‘[a-m].root-servers.net
  - Multi level
  - 2011: 18.5 million DNS servers
- UDP 53

Domain Name Service



Domain Name Service



Domain Name Service

```
struct hostent *gethostbyname(const char *name);
```

- Provides access to domain name services
- Convert domain names to IP addresses
- Connects to DNS server, if necessary

Domain Name Service: Important `struct`

```
struct hostent {
    char  *h_name;          /* official name of host */
    char **h_aliases;       /* alias list */
    int    h_addrtype;      /* host address type */
    int    h_length;        /* length of address */
    char **h_addr_list;     /* list of addresses, NULL-terminated */
}
```

Practical Work 2: gethostbyname()

- Write a new program in C
  - Name it « 02.practical.work.gethostbyname.c »
  - Use gethostbyname() to resolve a domain name
    - From program's arguments
    - From STDIN if there's no argument
  - Show the resolved IP address
- Use `man` for help
  - `gethostbyname()`
  - `inet_ntoa()`, using `hostent->h_addr_list` with proper cast

Practical Work 2: gethostbyname()

- Write a report named « 02.report.gethostbyname.tex »
  - Deploy your above practical work to your VPS
  - Run your practical work several times on your laptop and on VPS
  - Compare the resolved IP address on your laptop and on VPS
  - Explain the result
- Push your C program to corresponding forked Github repository