# Computer Networking

# Table of Contents

# Internet History

- 1969
  - arpa (arpanet) -US military science network links remote locations
    - packet switching network (as opposed to circuit-switching using a dedicated line between two parties like a telephone)
    - open-architecture network (any network topology can connect)
- 1970's
  - scientific and educational institutions added to ARPA network
  - many advancements in protocols and related technologies
- 1982
  - Internet Protocol Suite (TCP/IP) standardized
  - Internet used as name of global TCP/IP network

- 1980's
  - public access to universities, and research sites
  - news groups introduced to read and post on various subjects
  - text based environment
    - no graphics but fast, no pop-ups and no spam
- 1992
  - www  (world wide web) becomes public with rapid web browser developments
    - web sites with on-line images
    - Internet traffic escalates as the public embraces the new graphical web
    - on-line advertising and spammers are lurking just around the corner
- 2002
  - web 2.0
    - extensive real-time user input/controls (live posts, etc)
    - 3rd party authentication
    - generally better interaction (new or improved) between users and web sites, and between web site to web site

# Network Fundamentals

Computers identify themselves through two types of addresses:

- **Hardware Address** *00:22:15:fb:97:02*
  - Also called a Physical, Ethernet or MAC (Media Access Control) address
  - This is the hard coded address of a NIC or Network Interface Card
  - Systems with multiple nic's have multiple MAC addresses
  - MAC addresses are only used within a local network and are not used when accessing external systems

- **IP** or **Internet Protocol Address** *93.184.216.34*
  - This is used to identify and navigate to remote machines
  - It's structure allows routers and other devices to efficiently navigate to remote networks such as on the Internet

People refer to computers by name not by number so there is a system to match computer name to the computer IP address and vice versa. The initial solution was a shared file on the Internet which you would download to your computer periodically. As the Internet grew and with the introduction of the World Wide Web this method quickly became unmanageable.

That led to the creation of the Domain Name System or DNS. Today, virtually every activity on the Internet requires DNS services.

# Routing

So you have a remote IP you want to connect with but there is no map of how to get to it. Now what?

- If the IP is not on your local network it goes to a Gateway Address which is the default exit path. The Gateway or Default Route is the local router connecting your network to another network. For home users this is usually your ISP's network.
  - *Routers* are devices that connect two or more separate networks
  - *Switches* are devices that connect devices within a single network
  - Most *home routers* are combinations of router, switch, and firewall
  - Large scale routers can connect several networks which in turn connect to other routers, and so on, often creating multiple routes
  - The packet moves from router to router until it reaches it's destination.
  - Routers learn and update each other on how various networks are connected and the best path to get to the next router (called the next hop).
  - Route information is stored in a *routing table* in the routers memory. There are several different routing protocols used to discover the *route path* and *route state* information which populate routing tables.
  - Cable Modems are actually a combination of a Bridge and Modem. They operate at a “lower” level than IP. (see Network Stack pg 20/21)

# DNS – Domain Name System

The Domain Name System is the most important service on the Internet because without it virtually nothing else works.

On the surface all we need do is ask a remote system what the IP address is of the system we want to access. However, not only are computers being added and removed from the Internet constantly but existing system often have to change IP addresses in order to meet various business or technical requirements. Trying to keep track of all those changes requires a dynamic distributed system.

DNS uses different servers to look after each part of the resolution process. First it reads the host name from right to left using the dots as separators, so it starts with the Top Level Domain such as *com org net ca edu*. There are separate servers to resolve the next part of the name, usually but not always the main domain name of the target, and so on until the server that hold the records to match the name to an IP has been reached.

# Recursive DNS

Here is a typical resolution process, called a *recursive DNS look up*:

- Your computers DNS Agent sends a request to the local DNS server (usually at your ISP)
- The local DNS server starts the following sequence of exchanges to find the hosts IP. Each line represents a communication between the local DNS server and the listed server.

1. root domain server -returns the IP of the Top Level Domain DNS server to use
2. top level domain name server -returns the IP of the 2nd Level Domain DNS server to use
3. second level domain name servers -returns the IP of the 3rd Level Domain DNS server to use
4. third level domain name servers -returns the IP of the next DNS server to use
5. etc etc until the server with the host records is found

6. The local DNS server, which was acting on your behalf, now sends the final host to IP mapping back to your computers DNS Agent so that you can check your email, view a web site, etc.

# Visiting a Website in Detail

Here is what happens when you enter or click on a URL in a web browser:  (for www.example.com)

1. a DNS servers IP is read from a local “resolver” file (/etc/resolv.conf in Linux)
2. the “hostname to IP” look up request is sent to the listed DNS server (usually at your ISP)
3. from there the local DNS server sends a request to the Top Level DNS Server (root servers)
4. they send back to the local DNS server which next DNS server to ask (“ask *com* server @ x.x.x.x”)
5. the local DNS server then sends a request to the *com* domain servers to find which DNS server has example.com  records
6. once that IP is returned, a request is sent to example.com's DNS server to find the IP of www.example.com  (in DNS www is simply another computer name to resolve)
7. once that IP is received the IP is passed back to your computer

website details continued...

# Visiting a Website in Detail (continued)

8. the web browser now sends the request to the web site

9. the web server now parses the incoming packet looking for the actual URL request. This is because numerous web sites can run off a single IP address.

10. the web page data is sent back to your computer, and if there are no site ads or remote images/content, the process is complete, otherwise the process is repeated for each chunk of remote content.

11. content is displayed in browser window

# Service Ports

Many servers run multiple services like web, ftp, or email.

You may notice that these services are often referred to as servers, such as a "web server". A web server is actually a web service running on a server.

When you use a web browser to access a web site how does the server know that you want to get a web page, and not an ftp service or check your email?

Every service running on a server is associated with a service number, called a Port Number or a Port. These numbers are standardized for all significant network services.

There are 65,535 port numbers available with the lower 1024 reserved for common internet services. Also, there are two primary types of network communication TCP and UDP, which can share port numbers (TCP and UDP are discussed later.)

# Common Service Ports

| | | |
|---|---|---|
| 20 | TCP/UDP | FTP (data transfer) |
| 21 | TCP | FTP control (commands) |
| 22 | TCP | Secure Shell (SSH) |
| 23 | TCP/UDP | Telnet protocol |
| 25 | TCP | Simple Mail Transfer Protocol (SMTP) |
| 80 | TCP | Hypertext Transfer Protocol (HTTP) |
| 110 | TCP | Post Office Protocol v3 (POP3) |
| 443 | TCP | HTTP Protocol over TLS/SSL (HTTPS) |
| 995 | TCP | POP3 Protocol over TLS/SSL (POP3S) |

# IP Addresses

A computers IP (Internet Protocol) address not only represents the computer but also contains information needed to find the network the computer is located on.

- An IP address  has four section:  (spaces added for readability)

  128 . 230 . 192 . 155

- An IP address is a 32 bit value comprised of 4 quadrants separated by a dot, each containing 8 bits.
  - each quadrant can have a value from 0 to 255

    (max value of 8 bits is 0xff = 255)
- The IP address format described is called IP version 4 or IP4. There is a newer format named IP6 which was designed to replace the aging IP4 which was quickly running out of numbers as the world computer population grew. IP6 is extremely versatile and has many advantages but is so complex it can be difficult to work with.

IP Addresses Continued...

- An IP6 address is a 128 bit string in eight 16 bit blocks:

  fe80 : 3a7e : 222b : 15ff : fefb : 9702

  (spaces added for readability)

- IP6 has not made it to mainstream use because of two technologies that emerged during it's development.
  - virtual web hosting
    - Apache web server (the most popular), and possible others, can host hundreds of web sites using a single IP address so web hosting services don't need one IP per customer, just one IP per server (physical or virtual).
  - The adaptation of Non-Routable Networks and NAT (Network Address Translation)
    - There are several IP4 network ranges that are designated as non-routable, meaning they cannot work in public networks. This means any private home or business network can use a non-routable network regardless of who else in the world is using it. Then, at the point of access to the Internet, all internal machines are represented externally by a single IP address. Network Address Translation, the task of tracking and directing requests, is typically done by a firewall or router at the edge of the network.

# Classful IP Address Scheme

Originally IP addresses were divided up into blocks of IP's that could be assigned to various organizations who wanted to access the Internet. This was fine for a while, but soon there were few network blocks left to hand out and many that were assigned were not in use. The concept is that each network was a minimum of and a multiple of 8 bits.

In this addressing scheme there were primarily 3 Classes:

- Class A -8 bit network
  - total of 128 networks each with 16,777,216 IP addresses
  - the largest class with blocks assigned to large organizations like IBM
- Class B -16 bit network
  - total of 16,384 networks each with 65,536 IP addresses
  - handed out to medium-to-large organizations
- Class C -24 bit network
  - total of 2,097,152 networks each with 256 IP addresses
  - assigned to small organizations, or sneaky individuals, with less than 254 hosts

# Classless Inter-Domain Routing

As Classful addressing was wasting precious IP's, in 1993 a new Classless scheme was developed to support more granular IP address block assignments. Instead of the 8 bit network blocks used in classful addressing, CIDR can split network/host at a single bit level.

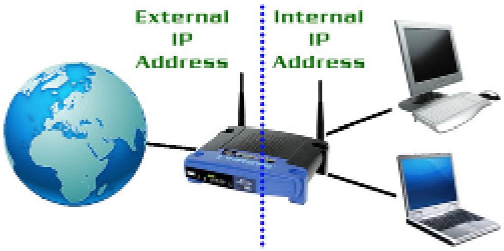
CIDR uses the format:

xxx.xxx.xxx.xxx/nn  where nn indicates the number of network bits

The Classless Inter-Domain Routing Scheme or CIDR has two major advantages:

- Networks of any size can be created reducing waste and adding flexibility to administrators of complex environments.
- CIDR introduced “classless routing” which among other things includes "route aggregation" which greatly increased router efficiency and reduced the size of the ever growing routing tables.

# Non-Routable Networks



Private (non-routable) IP network ranges:

Class A: 10.0.0.0/8

Netmask: (255.0.0.0)    8 network bits, 24 host bits

16,777,216 host addresses

Class B: 172.16.0.0/12

Netmask: (255.240.0.0)    12 network bits, 20 host bits

1,048,576 host addresses

Class C: 192.168.0.0/16

Netmask: (255.255.0.0)    16 network bits, 16 host bits

65,536 host addresses

Note: most home routers use 192.168.1.0/24 for internal network ranges

# Left Side vs Right Side

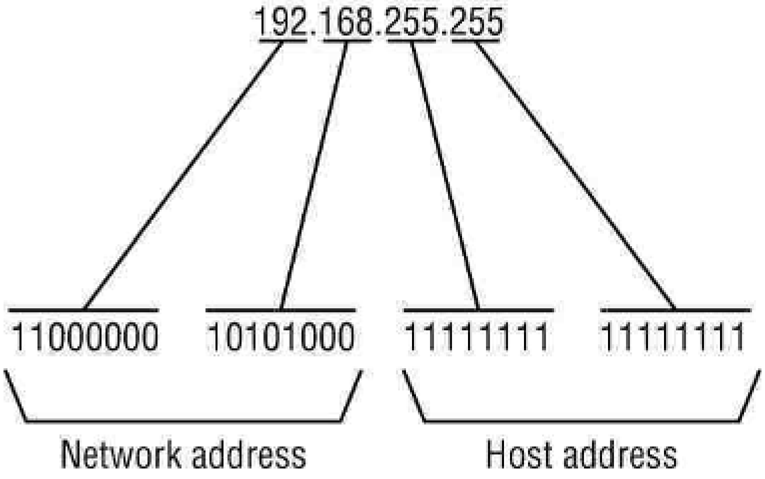
- An IP address has two logical sections, left and right.
  - the left side are the network bits
  - the right side are the host bits (computer)
- A Network Mask or Netmask defines how much of the address represents the network (left side) and the host (right side)
- A Network Mask can be in two formats:

x.x.x.x/n -Host IP appended with /network-bits.

- Called Network CIDR (Classless Inter-Domain Routing)

x.x.x.x n.n.n.n -Host IP followed by Subnet Mask

- Called Classful Address Scheme
- Older style, not used as often but totally valid



# Behind The Mask

Lets break down an IP & Network Mask

192.168.1.101/24 (or 192.168.1.101 255.255.255.0)

- network is 24 bits of the IP address

  (24 bits = 8bits.8bits.8bits.0bits)

- network = 192.168.1.0

  (the 0 (zero) represents the entire 192.168.1 network)

  hosts =192.168.1.1 -> 192.168.1.255

We need a gateway, or way out of the network. Usually this will be either the first or last IP address, and we need a Broadcast Address which is usually set as the highest number in the host range.

- Gateway: 192.168.1.1
  - default gateway sends packets outside the local network
- Broadcast: 192.168.1.255
  - a broadcast sends a packet that is delivered/accepted by every system on the local network.
  - broadcasts do not go through routers
- We can support up to 253 computers on our network using IP's:
  - 192.168.1.2 through 192.168.1.254 inclusive

# Network Bits

Although IP numbers are expressed as four decimal values between 0 and 255 (0x0 to 0xff), they are often calculated using binary.

Binary representation works well as it gives a visual separation of network and host bits.

These complex calculations are usually only of concern to network administrators but a general understanding of the concept is valuable for anyone supporting network services (web servers, email server, etc).

Example:

host:  192.168.1.33     network: 192.168.0.0/18   (CIDR subnet)

Binary:

full address:  11000000 . 10101000 . 00000001 . 00100001 (192.168.1.33)

net mask:      11111111 . 11111111 . 11 000000 . 00000000  (255.255.192.0  /18  first 18 bits)

host “wildcard”: 00000000 . 00000000 . 11 111111 . 11111111    (0.0.63.255  all hosts digits)

network bits:     11000000 . 10101000 . 00

host bits:     0001 . 00100001

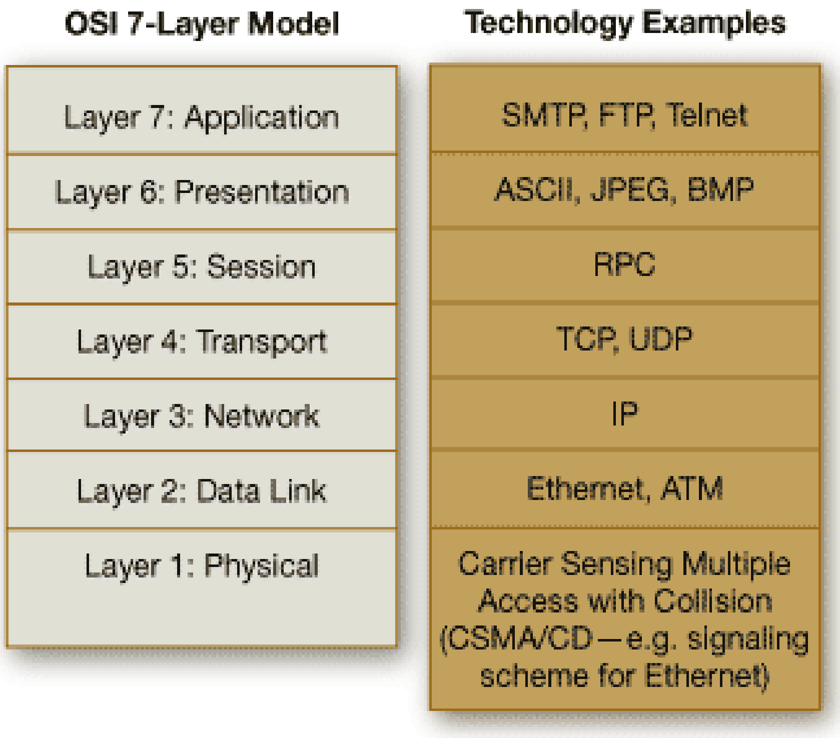
network:     192 . 168     (digits under netmask)

host:     1 . 33     (digits under wildcard)

# The OSI Network Stack

- OSI Model
  - 7 layer global model
  - used as a loose framework for developers

- The seven layers:
  - application
  - presentation
  - session
  - transport
  - network
  - data
  - physical

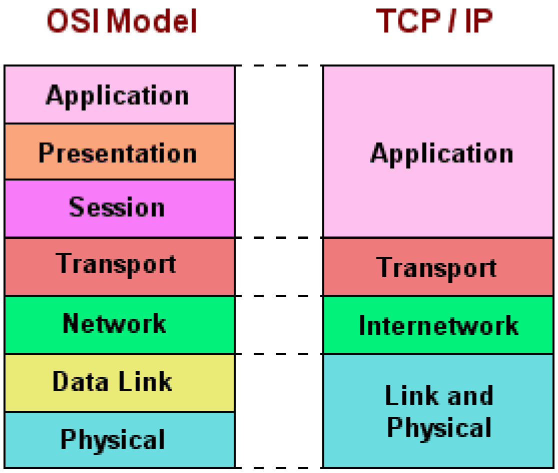


| OSI 7-Layer Model | Technology Examples |
|---|---|
| Layer 7: Application | SMTP, FTP, Telnet |
| Layer 6: Presentation | ASCII, JPEG, BMP |
| Layer 5: Session | RPC |
| Layer 4: Transport | TCP, UDP |
| Layer 3: Network | IP |
| Layer 2: Data Link | Ethernet, ATM |
| Layer 1: Physical | Carrier Sensing Multiple Access with Collision (CSMA/CD—e.g. signaling scheme for Ethernet) |

OSI or “Open Systems Interconnection” is a project to standardize computer networking. It was started in 1977 by the International Organization for Standardization (ISO)

# TCP/IP Network Stack

This is the protocol stack used on the Internet, sometimes referred to as the Internet Protocol Stack or IP Stack. It is based on the OSI 7 layer model but only uses 4 layers. It is used for both TCP and UDP.



TCP/IP Stack example:

Typical home or office

- browser (http) -Application Layer
- TCP -Transport Layer
- IP -Internetwork Layer
- Ethernet -Link and Physical Layer
  - Cable Modems connect Ethernet to DOCSIS (Data Over Cable Service Interface Specification) network segments, at Layer 1 (Link and Physical)

# Frames, Packets and Datagrams

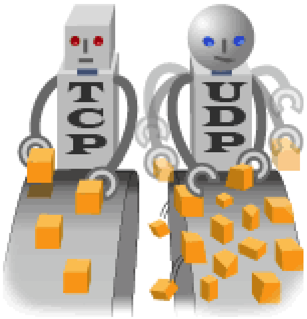
Each layer of the network stack uses it's own wrapper around the data. It is like putting a letter into several envelopes, one with each part of the delivery process, such as country in the outermost, then postal code, then province, then street, etc, with the actual letter in the very inside.

Terminology:

- **Bit**: refers to layer 1 (physical) or layer 2 (data link)
- **Frame**: refers to layer 2 (data link)
- **Packet** or **Datagram**: refers to layer 3 (network)
- **Segments**: refers to layer 4 (transport)
- **Data**: refers to layers 5, 6 or 7

The term *Packet* can also refer to the whole transmission package.

# TCP and UDP



There are two types of IP network layer 4 communications:

- TCP   Transmission Control Protocol
  - connection oriented
    - ongoing two way communication
  - guaranteed, ordered delivery of data
  - slower due to overhead (connecting, tracking, and disconnecting)

- UDP   User Datagram Protocol
  - connectionless
    - transmit only, no further communication
  - best effort delivery only
  - fast because data is sent but never tracked

# TCP vs UDP

With data integrity and error checking TCP packets do not have as big a payload as UDP.

- Layer 2 frame
  - Ethernet  header size: 26 bytes + 1500 for data bytes = 1526 bytes max
- Layer 3 packet
  - IP min header 20 bytes. Leaves 1500 bytes - 20 bytes = 1480 data bytes
- Layer 4 segment
  - TCP  minimum 20 byte header, leaves 1460 bytes of actual data in a TCP/IP over Ethernet packet
  - UDP  minimum 8 byte header, leaves 1472 bytes of actual data in a UDP/IP over Ethernet packet

TCP is a connection based transmission meaning that before any data is sent a connection must be established and once complete, a tear-down or disconnect must occur.

UDP is a best effort, no tracking or error reporting. It's fast but unreliable.

You may think of sending by UDP like mailing a regular letter, while TCP is like sending a more costly registered letter, signature required.

# TCP

TCP is used when reliability and data integrity outweigh speed concerns.

It is the dominant layer 4 protocol used on the Internet. Even though social media services such as Twitter have significantly increased UDP traffic due to the high volume of messages, the packet count is tiny compared to loading a media rich web page over TCP.

TCP Transmission:

1. there is an initial 3 way handshake to establish the connection
    1. Sender sends a SYN (synchronize) packet to receiver.
    2. The receiver replies with an ACK (to acknowledge the SYN) and a SYN both in one packet.
    3. The sender sends an ACK to acknowledge receipt of the SYN from the receiver. The connection is now set up and data transfer can start.
2. each TCP packet has a sequence number which needs to be tracked by the sender and acknowledged by the receiver
3. the sender must time each frame and check for an acknowledgment, resending the frame if no acknowledgment is received on time

TCP Continued...

# TCP (continued)

4. TCP frames must be buffered by the receiver so they can be reordered if necessary before passing them up the stack to the application

5. after the transmission is complete the connection must be torn down:

    1. The sender sends a FIN packet

    2. The receiver send a packet with an ACK and a FIN

    3. The sender sends an ACK packet and the session is closed.

# UDP

Most communication uses TCP because of the reliability but it is not always needed or even desired in which case UDP is used. An application can also do some higher level data tracking but this is not the norm.

**UDP Transmission:**

- sending side send a UDP based packet which may or may not be received at the far end

**Common uses of UDP:**

- DNS requests (not server to server record updates)
    - dns is a tiny data exchange but is in constant use so it makes no sense to burden things with TCP, although some DNS servers are set up for TCP as well
- message services
    - IM, Twitter, etc
- most streams
    - music, VoIP, movies (sometimes they will use TCP for Quality of Service requirements)

**“I'd tell you a UDP joke, but you might not get it”**

# Web Pages and Packets Counts

Lets look at how many transmissions a typical web page request requires.

This is for the web data transfer only and does not include DNS related traffic.

- First we set tcpdump to capture traffic to or from GRSS's site:

```
# tcpdump -i p34p1 -w /tmp/grss_count 'tcp port 80 and host www.grss.sd84.bc.ca'
```

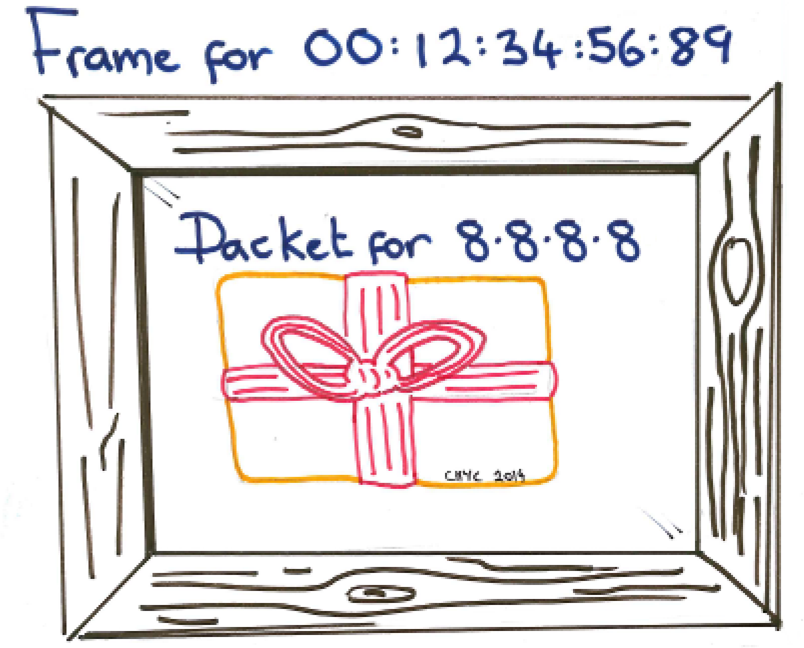
- Next we use wget to retrieve the main index without images:

```
$ wget http://www.grss.sd84.bc.ca

index.html    [ <=> ]  33.17K  --.-KB/s   in 0.1s

2015-02-14 16:55:51 (330 KB/s) - ‘index.html’ saved [33961]
```

So we downloaded a single web page 33.17 Kilobytes in size (no images)

- Our tcpdump output file shows 58 packets were exchanged
- Including images (via browser) it's about **13 MB** and uses **18,425 packets**
- packet counts may vary slightly depending on network conditions
- around half the packets are for handshakes and acknowledgments

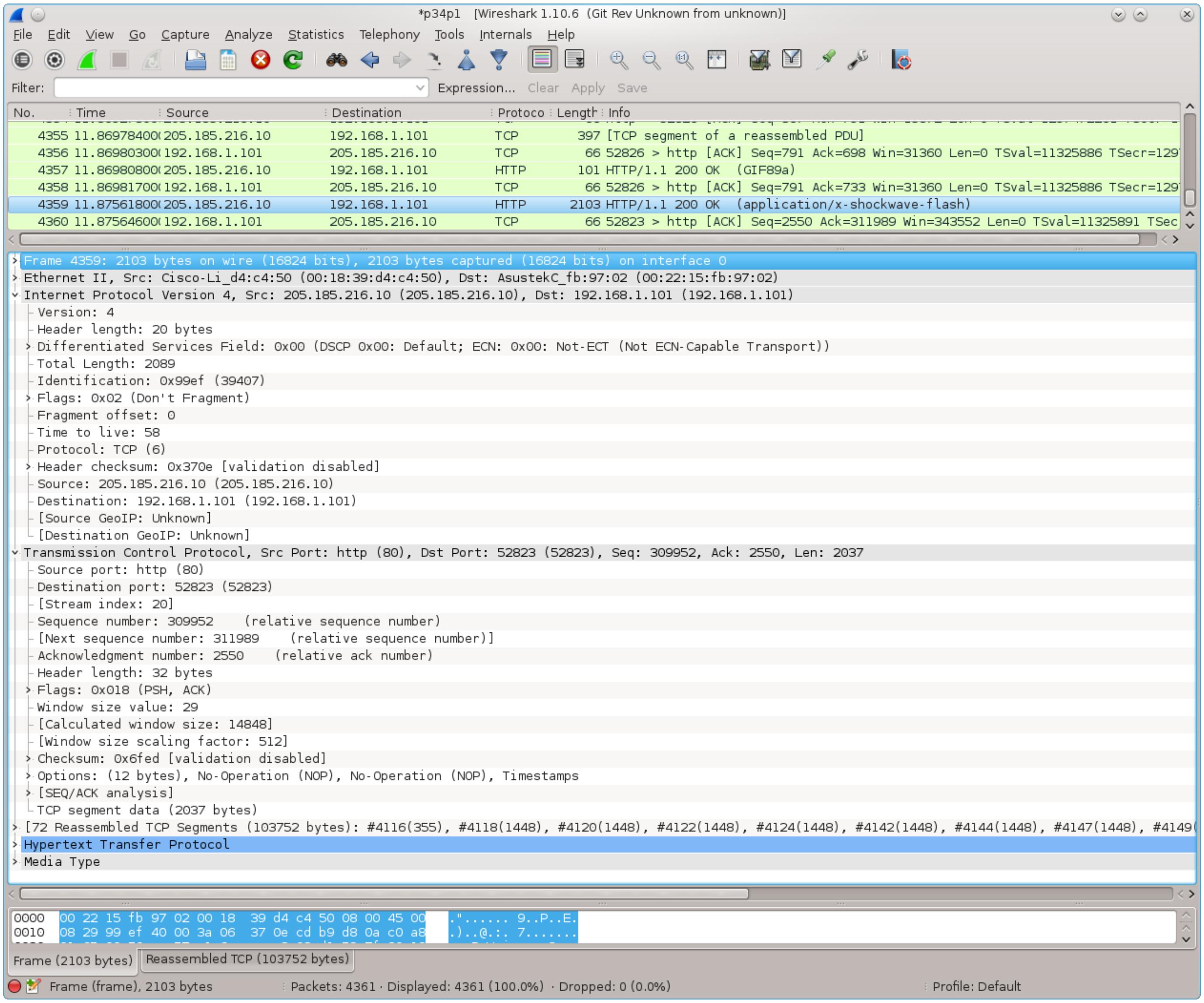
# Looking Inside the Packet



- Wireshark
  - network protocol analyzer
  - provides both text based and graphical views of inside packets and frames
  - www.wireshark.org (free, Linux and Windows)

- tcpdump
  - command-line packet analyzer for network traffic capture
  - very powerful and flexible but complex
  - part of base install on all Linux systems

# Wireshark



*p34p1 [Wireshark 1.10.6 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression...
Clear
Apply
Save

```
No.  Time              Source          Destination     Protoco Length Info
4355 11.86978400( 205.185.216.10  192.168.1.101   TCP      397 [TCP segment of a reassembled PDU]
4356 11.86980300( 192.168.1.101   205.185.216.10  TCP       66 52826 > http [ACK] Seq=791 Ack=698 Win=31360 Len=0 TSval=11325886 TSecr=129
4357 11.86980800( 205.185.216.10  192.168.1.101   HTTP     101 HTTP/1.1 200 OK  (GIF89a)
4358 11.86981700( 192.168.1.101   205.185.216.10  TCP       66 52826 > http [ACK] Seq=791 Ack=733 Win=31360 Len=0 TSval=11325886 TSecr=129
4359 11.87561800( 205.185.216.10  192.168.1.101   HTTP    2103 HTTP/1.1 200 OK  (application/x-shockwave-flash)
4360 11.87564600( 192.168.1.101   205.185.216.10  TCP       66 52823 > http [ACK] Seq=2550 Ack=311989 Win=343552 Len=0 TSval=11325891 TSec
```

```
> Frame 4359: 2103 bytes on wire (16824 bits), 2103 bytes captured (16824 bits) on interface 0
> Ethernet II, Src: Cisco-Li_d4:c4:50 (00:18:39:d4:c4:50), Dst: AsustekC_fb:97:02 (00:22:15:fb:97:02)
v Internet Protocol Version 4, Src: 205.185.216.10 (205.185.216.10), Dst: 192.168.1.101 (192.168.1.101)
    Version: 4
    Header length: 20 bytes
  > Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
    Total Length: 2089
    Identification: 0x99ef (39407)
  > Flags: 0x02 (Don't Fragment)
    Fragment offset: 0
    Time to live: 58
    Protocol: TCP (6)
  > Header checksum: 0x370e [validation disabled]
    Source: 205.185.216.10 (205.185.216.10)
    Destination: 192.168.1.101 (192.168.1.101)
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
v Transmission Control Protocol, Src Port: http (80), Dst Port: 52823 (52823), Seq: 309952, Ack: 2550, Len: 2037
    Source port: http (80)
    Destination port: 52823 (52823)
    [Stream index: 20]
    Sequence number: 309952    (relative sequence number)
    [Next sequence number: 311989    (relative sequence number)]
    Acknowledgment number: 2550    (relative ack number)
    Header length: 32 bytes
  > Flags: 0x018 (PSH, ACK)
    Window size value: 29
    [Calculated window size: 14848]
    [Window size scaling factor: 512]
  > Checksum: 0x6fed [validation disabled]
  > Options: (12 bytes), No-Operation (NOP), No-Operation (NOP), Timestamps
  > [SEQ/ACK analysis]
    TCP segment data (2037 bytes)
> [72 Reassembled TCP Segments (103752 bytes): #4116(355), #4118(1448), #4120(1448), #4122(1448), #4124(1448), #4142(1448), #4144(1448), #4147(1448), #4149(
> Hypertext Transfer Protocol
> Media Type
```

```
0000  00 22 15 fb 97 02 00 18  39 d4 c4 50 08 00 45 00   ."...... 9..P..E.
0010  08 29 99 ef 40 00 3a 06  37 0e cd b9 d8 0a c0 a8   .)..@.:. 7.......
```

Frame (frame), 2103 bytes   Packets: 4361 · Displayed: 4361 (100.0%) · Dropped: 0 (0.0%)   Profile: Default

# tcpdump

```
 # tcpdump -vv -i p34p1

tcpdump: listening on p34p1, link-type EN10MB (Ethernet), capture size 65535 bytes

17:02:40.288726 IP (tos 0x0, ttl 64, id 38279, offset 0, flags [DF], proto UDP (17), length 68)

    localhost.37415 > google-public-dns-a.google.com.domain: [bad udp cksum 0xd25e -> 0xfe82!] 32590+
A? securemail.webnames.ca. (40)

17:02:40.289088 IP (tos 0x0, ttl 64, id 38280, offset 0, flags [DF], proto UDP (17), length 66)

    localhost.45402 > google-public-dns-a.google.com.domain: [bad udp cksum 0xd25c -> 0x8c89!] 54830+
PTR? 8.8.8.8.in-addr.arpa. (38)

17:02:40.289110 IP (tos 0x0, ttl 64, id 38281, offset 0, flags [DF], proto UDP (17), length 68)

    localhost.37415 > google-public-dns-a.google.com.domain: [bad udp cksum 0xd25e -> 0x9502!] 59571+
AAAA? securemail.webnames.ca. (40)

17:02:40.326255 IP (tos 0x0, ttl 57, id 36756, offset 0, flags [none], proto UDP (17), length 84)

    google-public-dns-a.google.com.domain > localhost.37415: [udp sum ok] 32590 q: A?
securemail.webnames.ca. 1/0/0 securemail.webnames.ca. A 65.39.140.89 (56)

17:02:40.361143 IP (tos 0x0, ttl 57, id 36785, offset 0, flags [none], proto UDP (17), length 119)

    google-public-dns-a.google.com.domain > localhost.37415: [udp sum ok] 59571 q: AAAA?
securemail.webnames.ca. 0/1/0 ns: webnames.ca. SOA ns1.webnames.ca. postmaster.webnames.ca.
201410281 1800 900 2419200 3600 (91)

17:02:40.361363 IP (tos 0x0, ttl 64, id 19821, offset 0, flags [DF], proto TCP (6), length 60)

    localhost.39417 > securemail.webnames.ca.pop3s: Flags [S], cksum 0x8fbc (incorrect -> 0x9ada), seq
3145535269, win 29200, options [mss 1460,sackOK,TS val 12599880 ecr 0,nop,wscale 7], length 0
```

# Network Hardware

Devices that would have at least one connection to a network (a *Network Node)* include:

- desktop computers, servers, network storage, network printers, switches, routes, etc.
- Most servers, switches, routers etc have multiple nodes (network connections)

**<u>Common Network Devices</u>:**

**router** -layer 3, connects multiple networks

**switch** -layer 2 or 3, connects devices within a single network

**hub** -layer 2, connects devices within a single network

**bridge** -layer 2, connects 2 networks of the same topology. Routes using hardware address, very fast

**firewall** -software running on a router, server, or dedicated device to apply rules for network traffic, usually ingress/egress

**load balancer** -at the edge of the network, spreads inbound requests across server farm (see below)

**web farm or server farm** -groups of servers configured exactly the same and often sharing contend from backend storage

**patch panel** -not a device. A panel to plug in network cables in order to move nodes between sub-nets or other business requirements without rewiring.

# Email

Email, or Electronic Mail, is one of the original Internet applications. It actually was implemented on the ARPA network, predating public use of the Internet.

Using email is fairly straightforward, but behind the scenes the process is more involved and can be extremely complex in large enterprise environments.

It's important to recognize that the process of sending email and the process of receiving email are not related, they are two completely separate activities and two separate services often on different servers.

# Sending Email

**SMTP (Simple Mail Transfer Protocol)**

- when you send email, from an email client or web interface, it is sent to your networks SMTP server which sends it to the SMTP server with the addressees mailbox
- servers involved in sending email run SMTP protocol but local SMTP servers can also be referred to as MTA's to discern them from the remote SMTP

**Sending Details:**

1. use a MUA (Mail User Agent) to compose email and click send (to trigger delivery)
2. this hands the message off to an MSA  (mail sender agent) which sends it to your local MTA (SMTP mail server)
    - home users local SMTP server (MTA) is usually at the local ISP
    - larger environments typically have one or more SMTP servers within the network
3. the MTA server extracts the destination domain from the destination email address which is in the *message envelope*
4. the MTA server does a DNS lookup for the MX record (SMTP server) of the target domain

Sending Email continued...

# Sending Email (continued)

5. the MTA server makes a TCP connection to the remote SMTP server, handing the message off for delivery
6. the remote SMTP server copies the message to the users mailbox, setting the NEW flag
7. the users MUA checks and acts upon (usually downloads) the contents of the mailbox

# Receiving Email

**POP (Post Office Protocol)**

- the original and still widely used protocol
- logs into a mail server, checks users mailbox, downloads any new messages, deletes message from server

**IMAP (Internet Message Access Protocol)**

- mail is managed on and remains on the server
- by default, mail is not downloaded to the users computer
- provides shared simultaneous access to mailboxes
- access email from any device with a IMAP email client
- access can also be achieved through webmail which is IMAP mailbox access via a web interface

- see POP and IMAP processes on following page

Receiving Email (continued)

# Receiving Email (continued)

**The POP process:**

1. email clients MUA connects to POP server over TCP
2. after authentication (usually automated through client) new messages are downloaded and by default, deleted from the users mailbox file on the POP server.

**The IMAP process:**

1. a TCP connection is established between a IMAP client and the IMAP server
2. from there the user can act upon the messages as if they were local

# Web Proxy

A proxy is a service that sits at the edge of the network and tunnels either inbound or outbound traffic, typically web traffic, acting as a cache to reduce network traffic, increase response times, and possibly apply restrictions. A proxy may be configured as a *transparent proxy,* meaning it is not visible to other devices or users. The open source software Squid is the most popular proxy server.

The two types of proxies are:

-*forward proxy* (works on outbound traffic) -internal users have outbound traffic tunneled through the proxy server

-*reverse proxy* (works on inbound traffic) -external traffic is sent to the proxy server which caches data from back-end servers.

**forward:**

- reduce outbound bandwidth and increase responsiveness for internal workers
- restrict or limit sites available to internal users
- a remotely hosted private proxy server can provide staff with a geolocation based view from the hosting service location

**reverse:**

- adds security by restricting access to back-end servers
- reduce ingress traffic into network, and improves response time for content on slow internal networks or servers

# Other Web Technologies

**Geolocation**

- used for redirection of sites for performance or content (.ca  vs .com) or legal/business reasons
- usually access an external service for location determination

**CDN** (Content Distribution Network or Content Delivery Network)

- caches copies of static text, image, audio, and video content
- content is available on a nearby server as opposed to the primary server which could be half way around the world and very busy.
- Akamai is the leader with over 100,000 servers in over 80 countries around the world.

# Google Public Services

- search engine
  - the default search is “AND” but you can use “OR” (upper case)

    “blue OR green”
  - use “site:“ to limit search to a list of sites

    “election site:ctv.ca OR site:.gov.ca”
  - quotes to search for the specified string

    ' mountains “blue sky” '
  - a minus or hyphen will exclude a keyword

    “bike -motorcycle”
  - try *google.ca/advanced_search* for advanced search options
- searchable world maps and Google Earth desktop application
- Gmail public web or client based email
- Google Plus (similar to Facebook)
- DNS open access to public DNS servers
- Numerous analytical and data tracking services

- SEO (Search Engine Optimization) and site administration tools

Google continued...

# Google Public Services (continued)

- Advertising services
- Publishing tools
- lots of other tools
- Google Search Appliance – a Google server usually sold to large organizations for indexing internal data.

Although the exact number is unknown, it is estimated that Google uses **over one million commodity-class x86** servers running customized versions of Linux.