NETWORK

LEARNING OBJECTIVES

- Understand and use properties in the HTTP protocol, like Headers, Caching, HTTP Sessions and Cookies
- Understand the Same Origin Policy related to REST and how to solve problems using CORS
- Understand, at a conceptual level, the concepts in the underlying structure of the internet
- Understand the Concepts Ports and IP and the Protocols TCP and UDP.
- Know how to implement simple, multi-threaded, TCP (UDP) servers in Java
- Understand the concept Virtualization
- Understand and Implement modern Server Architectures
- Understand and use concepts required to setup a Server with a domain name, Reverse Proxy and SSL

BUSINESS COMPETENCES

- The necessary background to implement and code user defined network protocols on top of TCP or UDP
- The minimal network knowledge, required to setup a Web Server Infrastructure used by a professional web-application

PLAN

Day1 - HTTP
Day2 - Internet
Day3 - Socket programming
Day4 - Server configuration

SUBJECTS

OSI MODEL (OPEN SYSTEMS INTERCONNECTION MODEL)

Conceptual model that characterizes and standardizes communication
Describes how computers communicate with one another over a network
Describes how different software and hardware components involved in a network communication should divide labor and interact with one another
Goal is interoperability of diverse communication systems with standard protocols
Defines networking framework to implement protocols in terms of a vertical stack of seven layers
A layer serves the layer above it and is served by the layer below it
Protocols are used between two endpoints and are fundamental for communication
All data that goes over a network connection passes through each of the seven layers, going from the upper level software oriented services to the lower level more hardware oriented functions

LAYERS

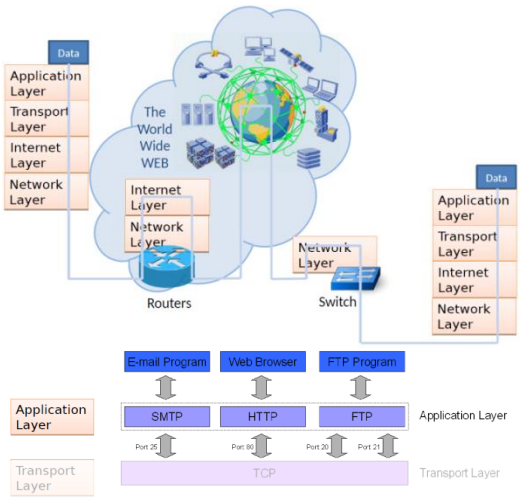
- Layer 7: Application Layer (Protocols: HTTP/FTP/TELNET/SSH/SMTP/POP3/DNS)
- Layer 6: Presentation Layer
- Layer 5: Session Layer
- Layer 4: Transport Layer (Protocols: TCP/UDP)
- Layer 3: Network Layer (Protocols: IP)
- Layer 2: Data Link Layer
- Layer 1: Physical Layer

TRANSMISSION CONTROL PROTOCOL / INTERNET PROTOCOL (TCP/IP)

TCP/IP model is in a way an implementation of the OSI model
TCP defines how applications can create reliable channels of communication across a network and IP defines addressing and routing
Protocol by which data is sent from one computer to another on the Internet, organized into four abstraction layers which specifies how data should be packetized, addressed, transmitted, routed, and received
Each computer/host on the Internet has one IP address that uniquely identifies it from all other computers on the Internet
When data is sent or received, it gets divided into packets, containing both the sender's and the receiver's IP address
During transmission, each layer adds a header to the data that directs and identifies the packet

LAYERS

- Layer 4: Application layer
  - Provides applications the ability to access the services of the other layers and defines the protocols that applications use to exchange data
  - The application layer contains the higher level protocols used by most applications for network communication
  - Data coded according to application layer protocols are encapsulated into a transport layer protocol (TCP / UDP) which in turn use lower layer protocols to do the actual data transfer
  - Application layer protocols generally treat the transport layer and lower protocols as black boxes which provide a stable network connection across which to communicate
- Layer 3: Transport layer
  - Responsible for providing Application layer with session and datagram communication services
  - The transport layer establishes host-to-host connectivity and transmits the data by sending packages
  - Its responsibility includes end-to-end message transfer independent of the underlying network, along with error control, segmentation, flow control, congestion control, and application addressing (port numbers)
- Layer 2: Internet layer
  - Responsible for addressing, packaging, and routing functions
  - Provides logical addressing so data can pass among subnets of different types
  - Provides packet routing, the task of sending packets of data (datagrams) from source to destination by sending them to the next network node (router) closer to the final destination
  - Relates physical addresses (used at the Network Access layer) to logical addresses
- Layer 1: Network layer
  - Responsible for sending and receiving TCP/IP packets on the network
  - Involves encapsulation of IP packets into frames for transmission, mapping IP addresses to physical hardware addresses (MAC Addresses) and physical transmission of data



INTERNET

Network interfaces
Point of interconnection between a computer and a private or public network
Interface between two pieces of equipment
Computer <-> Router
Router <-> Switch
Typically a network card
A network interface will usually have some form of network address
Network interfaces provide standardized functions such as passing messages, connecting and disconnecting

Switch
Connects devices together on network by using packet switching to receive, process, and forward data to the destination device

Router
Networking device that forwards data packets between computer networks
Routers perform the traffic directing functions on the Internet
A data packet is typically forwarded from one router to another router through the networks until it reaches its destination node

WAN (Wide area network)
Network that spans a large geographic area such as across cities, states, or countries

LAN (Local area network)
Network that interconnects computers within a limited area such as a residence, school, laboratory, university campus or office building

MAC Addresses
Kind of serial number assigned to every network adapter
MAC addresses are assigned at the time hardware is manufactured
Each network adapter has one, including wired and wireless interfaces
Used to direct packets from one device to the next as data travels on a network
Network adapter's MAC address travels the network only until the next device along the way

IP Addresses
Numerical label
Numbers are used by routers and servers to direct requests and get correct responses
IP addresses are assigned as part of connecting to a network
IP address is assigned to every device on a network, so that device can be located on that network
Each ISP or private network administrator assigns an IP address to each device connected to its network, on either static or dynamic basis (*static IP address / dynamic IP address)*

IPv4 Defines an IP address as a 32-bit number
(172.160.254.100)
Number of IPv4 addresses
2^32 = 4,294,967,296

IPv6 Uses 128 bits for the IP address
(2001:0db8:85a3:0000:0000:8a2e:0370:7334)
Number of IPv6 addresses
2^128 = 340,282,366,920,938,463,463,374,607,431,768,211,456

340 undecillion, 282 decillion, 366 nonillion, 920 octillion, 938 septillion, 463 sextillion, 463 quintillion, 374 quadrillion, 607 trillion, 431 billion, 768 million, 211 thousand and 456

Public addresses

Globally routable unicast IP address

Private addresses

If directly connected, a computer will have an IP address that can be reached from anywhere on the internet

If behind a router, router will have the internet-visible IP address, but it will then set up a separate, private network to which the computer is connected, assigning IP addresses out of a private range, that is not directly visible on the internet.

Any internet traffic a computer generates must go through the router and will appear on the internet to have come from the router

Three non-overlapping ranges of IPv4 addresses for private networks

	Start	End	No. of addresses
24-bit block	10.0.0.0	10.255.255.255	16777216
20-bit block	172.16.0.0	172.31.255.255	1048576
16-bit block	192.168.0.0	192.168.255.255	65536

Multiple client devices can appear to share an IP address, either because they are part of a shared hosting web server environment or because an IPv4 network address translator (NAT) or proxy server acts as an intermediary agent on behalf of the client

Subnet mask

Organization of hosts into logical groups

Enables network administrator to further divide the host part of the address into two or more subnets

Subnet mask is a mask used to determine what subnet an IP address belongs to

A subnet mask neither works like an IP address nor exist independently of them. Instead, subnet masks accompany an IP address and the two values work together

IPv4	192	168	33	22
Stored in computer	11000000	10101000	00100001	00010110
Subnet mask	255	255	240	0
Stored in computer	11111111	11111111	11110000	00000000

Gateway

The node that is assumed to know how to forward packets on to other network

DHCP (Dynamic Host Configuration Protocol)

Dynamically distributes addresses, subnet masks and gateway

When a DHCP-configured client connects to a network, it sends a broadcast query requesting necessary information to a DHCP server

The DHCP server manages a pool of IP addresses and information about default gateway, domain name and similar

On receiving a valid request, the server assigns the computer an IP address, a lease (length of time the allocation is valid), subnet mask, default gateway and similar

The query is typically initiated immediately after booting and must complete before the client can initiate IP-based communication with other hosts

Upon disconnecting, the IP address is returned to the pool for use by other clients

NAT (Network address translation)

The way that the router translates the IP addresses of packets that cross the internet/local network boundary

Method of remapping one IP address space into another by modifying network address information

Allows a single device, such as a router, to act as an agent between the Internet (public network) and a local (private) network

This means that only a single, unique IP address is required to represent an entire group of computers

1. Local machine sends package
2. Router sees package and replaces source IP with the router's own IP
3. Remote host replies to router
4. Router sees reply and replaces destination IP with local IP

DNS (Domain name system)
Better to use names instead of numbers for Internet
The way that internet domain names are located and translated into IP addresses
Globally distributed, scalable, reliable, dynamic database, used to map between hostnames and IP addresses, and to provide electronic mail routing information

TechnologySite.com <=> 165.23.43.66

DNS mapping is distributed throughout the internet
Access providers, enterprises, governments, universities and other organizations, typically have their own assigned ranges of IP addresses, an assigned domain name and run DNS servers to manage the mapping of those names to those addresses

DNS servers answer questions from both inside and outside their own domains
When a server receives a request from outside the domain for information about a name or address inside the domain, it provides the authoritative answer
When a server receives a request from inside its own domain for information about a name or address outside that domain, it passes the request out to another server

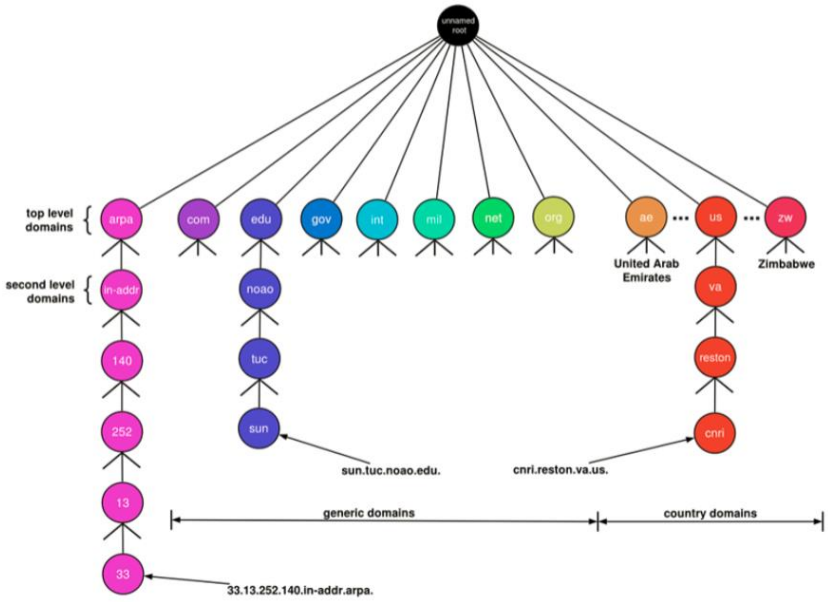
Authoritative DNS server

Satisfies queries from its own data without needing to reference another source
Responsible for providing mapping answers to recursive DNS servers

Recursive DNS server

Answers queries by asking other name servers
Assigned with task of finding IP addresses for domain names
If nothing is cached authoritative DNS hierarchy is asked for answers

The top hierarchy of the Domain Name System is served by the root name servers and top level domain name servers



COMMANDS

ipconfig / ifconfig (Internet Protocol Configuration)

Displays the IP address, subnet mask, and default gateway for all adapters
Displays all current TCP/IP network configuration values and can modify Dynamic Host Configuration Protocol (DHCP) and Domain Name System (DNS) settings
Ipconfig /all gives more detailed information
An important additional feature is to force refreshing of the DHCP IP address of the host computer to request a different IP address
ipconfig /release is executed to force the client to immediately give up its lease by sending the server a DHCP release notification which updates the server's status information and marks the old client's IP address as "available"
ipconfig /renew is executed to request a new IP address

ping

Tests the ability of the source computer to reach a specified destination computer
Verifies that a computer can communicate over the network with another computer or network device
Operates by sending messages to destination computer and waiting for a response
How many responses are returned, and how long it takes for them to return, are the two major pieces of information that the ping command provides
Also gives destination public ip address

tracert / traceroute

Used to show details about the path that packets take from host to destination
Displays route (path) and measure transit delays of packets across network
Identifies network devices all the way to destination, plus delays and packet loss at each stop (hop)
Round-trip time is time that it takes for a packet to get to a hop and back (in milliseconds)

pathping

Combination of ping and tracert

netstat (Network statistics)

Lists opened connections (sockets) on your machine
Displays network connections, both incoming and outgoing, plus routing tables

-ab	Shows listeners	(Run as administrator)

nslookup (Name server lookup)

Used to obtain information about internet servers
Finds name server information for domains by querying the Domain Name System (DNS)
Can identify which DNS server that the host is currently configured to use for its DNS lookups
A non-authoritative answer refer to DNS record kept on third-party DNS servers
An authoritative address lookup can be performed by specifying one of the domain's registered name servers

set type=a	Specifies a computer's IP address
set type=ns	Specifies a DNS name server for the named zone
set type=any	Specifies all types of data

whois

Search domain name registration records
https://www.whois.com/whois/

whatismyip

Google ip
https://whatsmyip.com/

HTTP (HYPERTEXT TRANSFER PROTOCOL)

Basic protocol of the web
Links documents on different servers via hyperlinks
Stateless application-level protocol
Request / Response
Initiated by a request / Replied with a response

HTTP package

- Request line / Status line
  - Request line: Method / URI / Version
    Request: GET /logo.gif HTTP/1.1
  - Status line: Version / Status code / Status text
    Response: HTTP/1.1 200 OK
- Headers
  Simple key-value pairs
- Empty line (<CR><LF>)
  The request / status line and headers must end with <CR><LF>
- Message
  Optional message body

Methods

GET	Requests a representation of the specified resource Requests using GET should only retrieve data and should have no other effect
POST	Requests that the server accept the entity enclosed in the request as a new subordinate of the resource identified by the URI
HEAD	Asks for the response identical to the one that would correspond to a GET request, but without the response body
PUT	Requests that the enclosed entity be stored under the supplied URI. If the URI refers to an already existing resource, it is modified; if the URI does not point to an existing resource, then the server can create the resource with that URI
DELETE	Deletes the specified resource
TRACE	Echoes back the received request so that a client can see what (if any) changes or additions have been made by intermediate servers
OPTIONS	Returns the HTTP methods that the server supports for specified URL This can be used to check the functionality of a web server

Session

The core HTTP protocol itself is stateless, but web applications built on top of HTTP do not have to be stateless
Different ways to introduce state
Query strings /index.html?user=jimmy&password=1234
Hidden form variables
Cookies
Local storage
Session storage

Cookies

Users sessions for a website exists in temporary memory only while users are navigating the website
Web browsers normally delete session cookies when users close their browsers

Session / Persistent

Caching

Set expiration

POSTMAN

Browser tool

Inspector (F12 -> Network & Storage)
Request / Response / Header / Body / Cookies

CORS

Same Origin Policy

A web browser permits scripts contained in a web page to access data in another web page, but only if both web pages have the same origin

Set in response header with Access-Control-Allow-Origin

Problem

Fetching data from external rest api's might not be allowed

Solution

Extract data via server and make it available locally

SOCKET PROGRAMMING

Sockets

The combination of an IP address and a port number

Two types

Stream socket

Reliable two way connected communication stream

Datagram socket

Protocols

UDP

Datagram communication protocol

Connectionless

Local socket descriptor and receiving socket address must be sent each time

Unreliable protocol

No delivery guarantee

No guarantee datagrams sent will be received in the same order

In UDP, there is a size limit of 64 kilobytes on datagram

TCP

Stream communication protocol

Connection-oriented

Connection must first be established before data can be transmitted

Reliable protocol

Guaranteed delivery

Guarantee packets sent will be received in the order in which they were sent

In TCP there is no size limit for streams

UDP vs. TCP?

Depends on client / server application needed

Socket programming

Server…

1. Create a ServerSocket object
   ```
   ServerSocket serverSocket = new ServerSocket();
   ```
2. Put server into waiting state
   ```
   Socket clientSocket = serverSocket.accept();
   ```
3. Set up input and output streams
   ```
   PrintWriter toClient = new PrintWriter( clientSocket.getOutputStream(), true );
   BufferedReader fromClient = new BufferedReader( new InputStreamReader( clientSocket.getInputStream() ) );
   ```
4. Send and receive data
   ```
   toClient.println("");
   fromClient.readLine();
   ```
5. Close connection
   ```
   toClient.close();
   fromClient.close();
   clientSocket.close();
   ```

Client…

1. Establish connection to server
   ```
   Socket clientSocket = new Socket(ip, port);
   ```
2. Set up input and output streams
   ```
   PrintWriter toServer = new PrintWriter(clientSocket.getOutputStream(), true);
   BufferedReader fromServer = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
   ```
3. Send and receive data
   ```
   toServer.println("");
   fromServer.readLine();
   ```
4. Close connection
   ```
   toServer.close();
   fromServer.close();
   clientSocket.close();
   ```

java.net.Socket

Implements client sockets (also called just “sockets”).
An endpoint for communication between two machines.
Constructor and Methods

Socket(String host, int port):

Creates a stream socket and connects it to the specified port number on the named host.

InputStream getInputStream()
OutputStream getOutputStream()
close()

java.net.ServerSocket

Implements server sockets.
Waits for requests to come in over the network.
Constructor and Methods

ServerSocket(int port)
Socket Accept():

Listens for a connection to be made to this socket and accepts it. This method blocks until a connection is made

Never hardcode IP and Port numbers

```
Socket clientSocket = new Socket(ip, port);
ServerSocket serverSocket = new ServerSocket();
serverSocket.bind(new InetSocketAddress(ip, port));
```

Telnet

Installation

Control panel -> Programs -> Features -> Telnet client

Usage

telnet localhost 10007

PacketSender

Program for sending and receiving TCP and UDP packets

https://packetsender.com/

Multi-threaded server

Multiple clients
Communication between clients

SERVER CONFIGURATION

DigitalOcean

JAR

Upload jar

FileZilla

Start jar

(java –cp filename.jar package.class &)

Example: *java -cp SOCKET.jar tcpserverclient.TcpServerTest &*

Find jar processid

netstat –lptu

Stop jar

(kill processid)

Example: *kill 1680*

Snapshot

Power down droplet

sudo poweroff

images -> Snapshots -> Choose droplet & Image name -> Take snapshot

Swap file

https://www.digitalocean.com/community/tutorials/how-to-configure-virtual-memory-swap-file-on-a-vps

1. free
2. cd /var
3. touch swap.img
4. chmod 600 swap.img
5. dd if=/dev/zero of=/var/swap.img bs=1024k count=1000
6. mkswap /var/swap.img
7. swapon /var/swap.img
8. echo "/var/swap.img none swap sw 0 0" >> /etc/fstab
9. reboot

Log in again and check status with free

Domain name

Update name servers

- ns1.digitalocean.com
- ns2.digitalocean.com
- ns3.digitalocean.com

Security

Cryptography

Secure communication to prevent third parties from reading private messages

Keys

A key is a value that works with a cryptographic algorithm to produce a specific cipher text

In public key cryptography, the bigger the key, the more secure the cipher text

Decryption by brute force is always possible given enough time and computing power, it is therefore important to pick keys of the right size; large enough to be secure, but small enough to be applied fairly quickly

Symmetric key cryptography

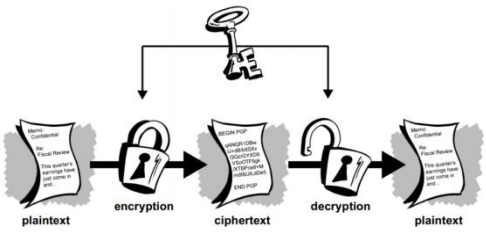
Same key is used by sender and receiver

Algorithms are public, but keys are secret

Algorithms: DES (Data encryption standard) / 3DES / AES (Advanced encryption standard)

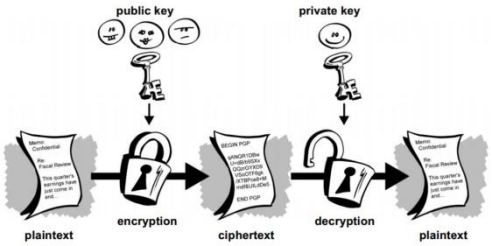
Advantage: Relatively fast

Disadvantage: Sharing the key / More damage if compromised



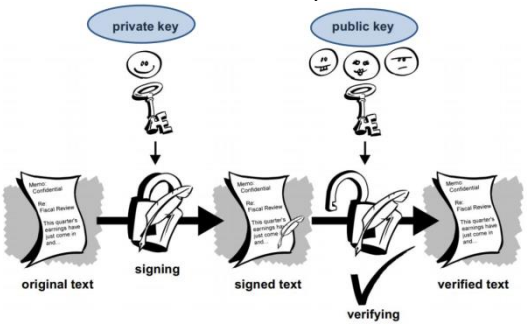
Public key cryptography

Two keys, a public and a private
Public key is available so anyone can encrypt message, but not decrypt message
Only private key can decrypt message
Public and private keys are mathematically related, but it's very difficult to derive the private key given only the public key
Algorithms: RSA / DSA (Digital signature algorithm) / Diffie-Hellman
Advantage: More secure / Less damage if compromised
Disadvantage: Slower than symmetric key cryptography

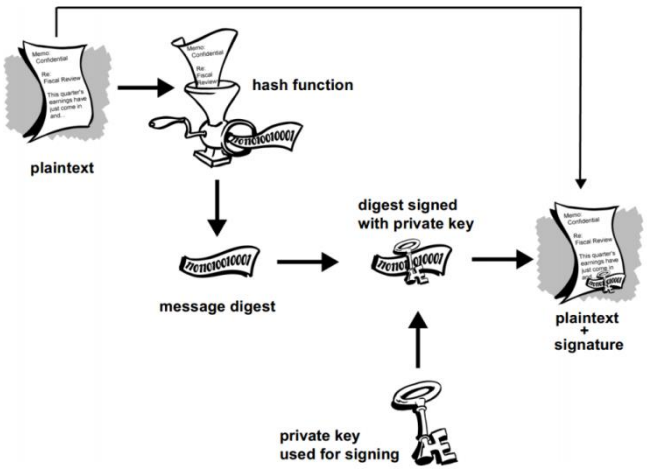


Digital signature

A major benefit of public key cryptography is that it provides a method for employing digital signatures
Enable the recipient of information to verify the authenticity of the information's origin, and also verify that the information is intact
Prevents the sender from claiming that he or she did not actually send the information



Signing whole document (Slow / Large)



Signing a digest (Fast / Small)

Digital certificate

A digital certificate is data that functions much like a physical certificate

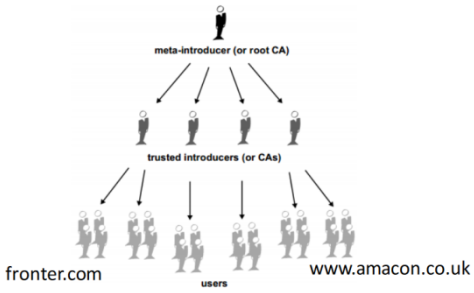
A digital certificate is information included with a person's public key that helps others verify that a key is genuine or valid

A digital certificate consists of three things:

A public key

Certificate information (Information about the user)

One or more digital signatures



Number of "root" certificates from which trust extends. These certificates may certify certificates themselves, or they may certify certificates that certify still other certificates downwards

View installed certificates

Start -> mmc -> add/remove snap-in -> certificates -> finish

TLS/SSL

Enables two parties to identify and authenticate each other and communicate with confidentiality and data integrity

TLS/SSL protocol adds a new layer

As with all the other layers TLS/SSL protocols are independent of the protocols above and below, but the layers "speak the same language" as the same layer on the other side of the communications channel

Not only ensures full compatibility with all the network technologies based on TCP/IP, but it also enables them to easily "switch" to their secure versions

HTTP became HTTPS without having to modify its specifications, FTP became FTPS, and so on

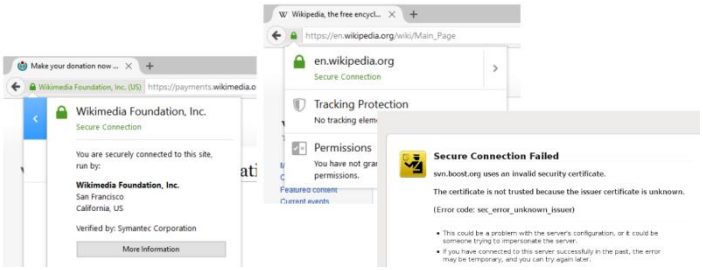
An SSL or TLS connection is initiated by an application, which becomes the SSL or TLS client

The application which receives the connection becomes the SSL or TLS server

Every new session begins with a handshake, as defined by the SSL or TLS protocols

For the duration of the session, the server and client can now exchange messages that are symmetrically encrypted with the shared secret key

HTTPS



Port 443
HTTP inside a tunnel

Requirements

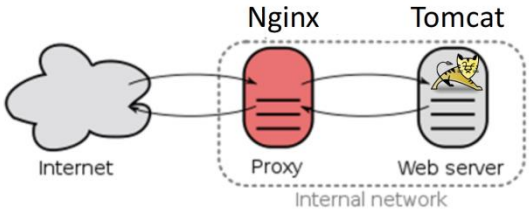
Web server should support TLS/SSL encryption (Tomcat / Nginx)
Unique IP address + Domain Name
SSL Certificate from an SSL certificate provider

Certificate provider

https://letsencrypt.org/ (Free)

Reverse proxy

A proxy server is an Intermediary server that forwards requests for content from multiple clients to different servers across the Internet
Those making requests to the proxy may not be aware of the internal network existence and characteristics of an origin server or servers



A reverse proxy server is a type of proxy server that typically sits behind the firewall in a private network and directs client requests to the appropriate backend server
A reverse proxy provides an additional level of abstraction and control to ensure the smooth flow of network traffic between clients and servers
Common uses for a reverse proxy server include load balancing, web acceleration, security and anonymity

A reverse proxy can distribute the load from incoming requests to several servers, with each server serving its own application area
A reverse proxy can reduce load on its origin servers by caching content
A web server may not perform SSL encryption itself, but instead offload the task to a reverse proxy

Nginx

Handle all SSL Traffic
Provide better frontend security than Tomcat alone

Configuration

/etc/nginx/nginx.conf

Change tomcat manager upload size

```
http {
    client_max_body_size 500M;
    ...
}
```

/etc/nginx/sites-available/default

Virtualization
Manual computing -> Mainframes -> Racks -> Virtualization

Virtual machine
Emulated operating systems

The cloud
Huge data centers running virtualized environments

Virtualization techniques
- X as a service
  - Infrastructure
  - Platform
  - Software

REFERENCES

HTTP

https://en.wikipedia.org/wiki/OSI_model
https://en.wikipedia.org/wiki/Internet_protocol_suite
https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol
https://en.wikipedia.org/wiki/List_of_HTTP_status_codes
https://en.wikipedia.org/wiki/List_of_HTTP_header_fields
https://en.wikipedia.org/wiki/HTTP_cookie
http://www.tutorialspoint.com/http/http_tutorial.pdf

CORS

https://en.wikipedia.org/wiki/Same-origin_policy
https://en.wikipedia.org/wiki/Cross-origin_resource_sharing

IP / DNS

https://www.youtube.com/watch?v=ub1o0M_DizM
http://www.thegeekstuff.com/2011/11/tcp-ip-fundamentals/
https://en.wikipedia.org/wiki/Domain_Name_System

SOCKETS

https://www.youtube.com/watch?v=3YoKVswJsKI
http://www.baeldung.com/a-guide-to-java-sockets
https://www.cs.uic.edu/~troy/spring05/cs450/sockets/socket.html
https://docs.oracle.com/javase/tutorial/networking/sockets/
https://www.tutorialspoint.com/design_pattern/observer_pattern.htm
https://packetsender.com/

SERVER CONFIGURATION

https://www.ibm.com/developerworks/cloud/library/cl-cloudservices1iaas/index.html
https://www.ibm.com/developerworks/cloud/library/cl-cloudservices2paas/index.html
https://youtu.be/ERp8420ucGs
https://youtu.be/LRMBZhdFjDI
https://en.wikipedia.org/wiki/Reverse_proxy
https://www.nginx.com/resources/glossary/reverse-proxy-server/