WEB Server Installation and Hardening

Building Secure Web Server

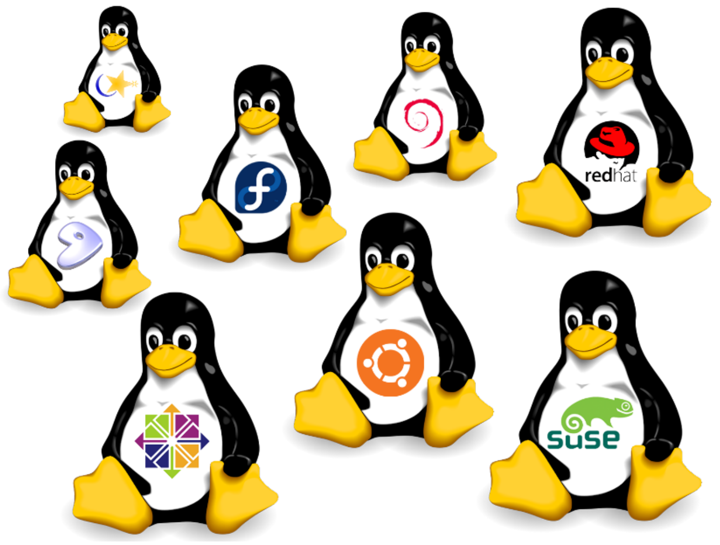
29th November 2016

System Architecture and Administration Workshop 2016

Thilina Pathirana
(University of Kelaniya)

LAMP Service

- What is LAMP
  - Linux - Our Operating System
  - Apache – Web Server Application
  - MySQL – Database Engine
  - PHP – Web Scripting Language



Apache
HTTP Server

MySQL®

php

APACHE Web Server

- World most used web server since 1996
- Developed and Maintained by Apache Software Foundation
- 100 Millionth Web Sites milestone passed in 2009
- The name Apache
    - Chosen out of respect to the Native American tribe Apache and their superior skills in warfare and strategy
- Released under the Apache License, Apache is free and open-source software.
- According to Wikipedia 46% of all active web sites run on Apache

Apache Web Server

- Can be Downloaded From https://httpd.apache.org/
- Installation in Ubuntu is just one command away
- Use sudo apt-get install apache2
- Installed Config Files will be found in
  - /etc/apache2/
- Default Web root
  - /var/www/html
- Default Configs
  - apache.conf

Apache Web Server

- Working Configurations
  - sites-enabled/*
  - confs-enabled/*
- Web roots/ directories must contain a file called index to serve as the configured landing page of that directory.
- All security and fine tuning settings can be changed using .conf files but we need to be more carefull unless apache service will be hanged.
- Default Apache User and Group
  - www-data:www-data (33)

Apache Web Server

- Block Directives that limit the application of other directives in the conf files
- They specify by a group like a tag section in html.

  -
    ```
    <VirtualHost host[:port]>
    ...
    </VirtualHost>
    ```

- Some Directives are:
  - User, Group: specify user and group that apache runs on.
  - ServerName: hostname of server
  - Listen: specify the port apache run on
  - ServerAdmin:email addr. for browser to do automatic replies.
  - DocumentRoot:
  - TransferLog, ErrorLog: where access,error logs are located

Apache Web Server

- KeepAlive [on|off](on): keep connection alive for n requests before terminate provided they come in before timeout. n is defined in MaxKeepAliveRequests <n>(100) directive

- KeepAliveTimeout <n>(15): wait for the next request for n seconds before terminate the connections.

- HostNameLookups [on|off|double](off): do reverse DNS lookup for logging the domain name of the request

- MaxClients <n>(256): the limit of # of simultaneous requests

- ??? Questions on Apache…..
- Why Apache ***2***

MySql Server

- One of the mostly used SQL based FOSS DBMS.
- Install from Ubuntu repository by
    - sudo apt-get install mysql-server
- During installation you will be requested to add a password for the default admin account “root”, Make sure you use a strong password as this needs to be protected at all cost unless your data are at danger.
- Once the installation is finished we can do some security hardening by running a script as
    - sudo mysql_secure_installation

PHP Server

- All web programmers love PHP. What is this PHP?
- To support php scripts we must integrate it with apache, therefore need to add apache and MySql library modules while installation
  - sudo apt-get install php7.0 libapache2-mod-php7.0 php-mysql

- Once Installed restart apache and try some php codes in your web root.

Virtual Web Hosting

- One of the mostly There are a few way we can host a web site:
  - Named-based Virtual Hosting
  - IP-based Virtual Hosting
  - Virtual Machine Virtual Hosting
- Name-based Virtual Hosting
  - A set of hostnames shared the same IP address (similar to alias)
  - utilize the HOST: meta header in http request (browser fill in the hostname) to distinguish different web site.
  - Each hostname will have its own site configuration, document root.
  - Require either the set of hostnames that are registered DNS names or the client machines need to configure their IP addresses mapping in host files
- IP-based virtual Hosting:
  - Require a unique IP address for each virtual hosting site
  - Use IP alias to configure the same Network Interface Card (NIC) to listen to different IP address, e.g., ifconfig eth0:1 5.16.19.3
- Use <VirtualHost hostname[:port]> block directives
- Specific ServerAdmin, DocumentRoot, ServerName, ErrorLog, TransferLog for individual virtual host

Virtual Web Hosting

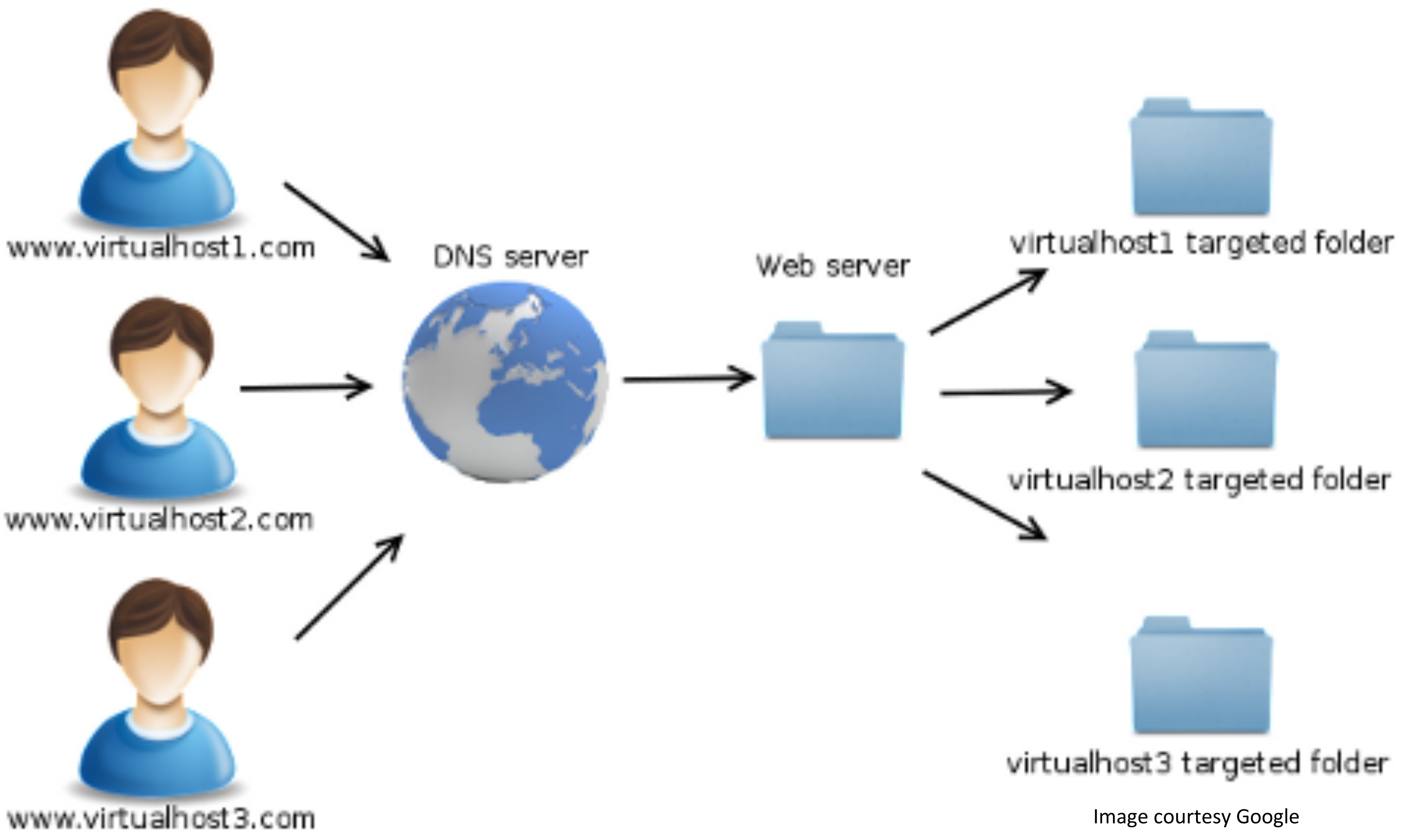

www.virtualhost1.com
www.virtualhost2.com
www.virtualhost3.com
DNS server
Web server
virtualhost1 targeted folder
virtualhost2 targeted folder
virtualhost3 targeted folder

Image courtesy Google

Virtual Web Hosting

- In apache we should create separate configure file to facilitate our virtual hosts. For the simplicity and neatness we will create one conf file for each virtual host we have.
- All virtual host conf's should be included in the sites-available directory in apache working path.
- In each VirtualHostName.conf we are allowed to specify its ServerName, ServerAdmin, Document Root, Log paths Directory permissions etc.
- Once the configurations are done we need to put our web files separately on specified paths and let apache enable them by,
  - sudo a2ensite VirtualHostName.conf
- And restart Apache.
- To check configs we must type in our VH dns entries or specific IP address in the browser, not the actual server address.

Common Gateway Interface (CGI) scripting

- CGI helps to integrate server side scripts written in none web languages to generate HTML capable outputs. Scripts written in Bash, C, Java, Perl, Python, etc languages can be execute using this method.
- CGI scripts needs to be written in a specific location where apache can give it the execute ability defined as its virtual host config.
- General CGI path is /usr/local/cgi-bin/
- To execute CGI scripts we also need to enable CGI modules in apache.
    - sudo a2enmod cgi
- With CGI we can write any server side script even a virus or malware. Therefore make sure you know what you are doing

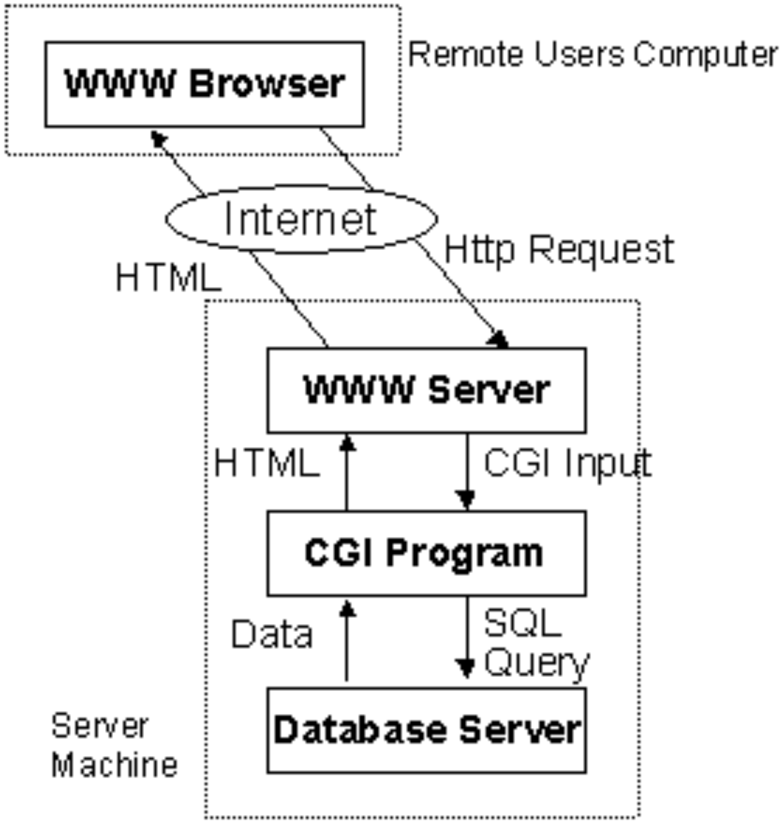
Common Gateway Interface (CGI) scripting

- In apache Virtual Host file, Include:

```
ScriptAlias /cgi-bin/ "/usr/local/cgi-bin/"
<Directory "/usr/local/cgi-bin/">
Options +ExecCGI
AddHandler cgi-script .cgi
Require all granted
</Directory>
```

- Here Script Alias will become an alias to url dns/**cgi-bin/**file.cgi

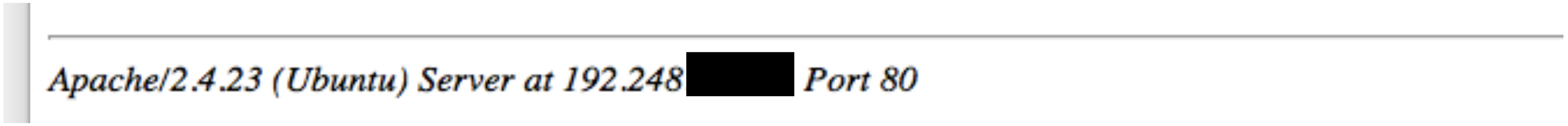
- While Directory derivative will provide execution privileges to apache engine and we must specify file extensions
as Addhandlers. In this case .cgi, but can be any extension such as .pl, .py etc



Apache Hardening

- To protect Apache web server from various attacks we should follow several techniques

❖ Hide Server Signature and details



```
Apache/2.4.23 (Ubuntu) Server at 192.248[redacted] Port 80
```

Edit /etc/apache2/conf-available/security.conf and modify

```
ServerSignature Off
ServerTokens Prod
TraceEnable Off
```

and Restart Apache

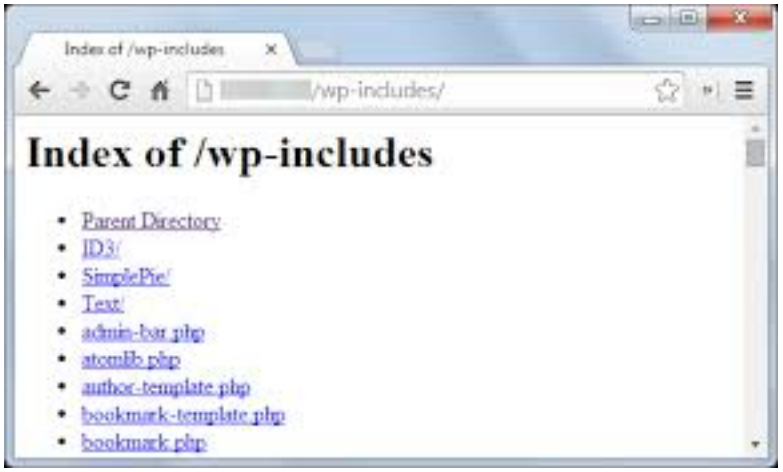
Apache Hardening

❖ Disable Directory Browsing

Edit apache.conf and remove Indexes from directory permissions

```
<Directory /var/www/>
# Options Indexes FollowSymLinks
Options -FollowSymLinks
AllowOverride None
Require all granted
</Directory>
```

and Restart Apache



Index of /wp-includes

- Parent Directory
- ID3/
- SimplePie/
- Text/
- admin-bar.php
- atomlib.php
- author-template.php
- bookmark-template.php
- bookmark.php

Apache Hardening

- Correct File Permissions

Web documents must run under the www-data or default apache user. Make sure to remove ownership from root to any files within web root directory.

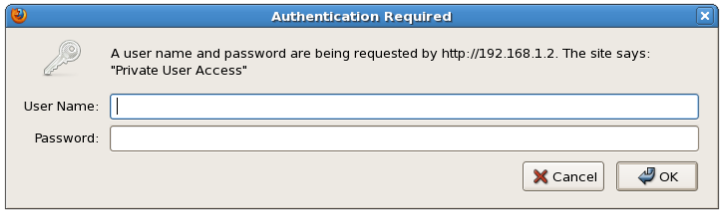
```
sudo chown –R www-data:www:data /var/www/
```

If you have any user data directories, make sure to remove executable permissions from them

```
sudo chmod 664 /var/www/uploads/
```

Apache Hardening

❖ Password Protected Directories



A user name and password are being requested by http://192.168.1.2. The site says: "Private User Access"

User Name:

Password:

Private Directories can be protected by introducing a user and pass to the apache instances. This is achieved by modifying virtual host config or creating a .htaccess file within the required directory path.

.htaccess should contain,

```
AuthType Basic
AuthName "Restricted Content"
AuthUserFile /etc/apache2/.htpasswd
Require valid-user
```

Then create specific users as,

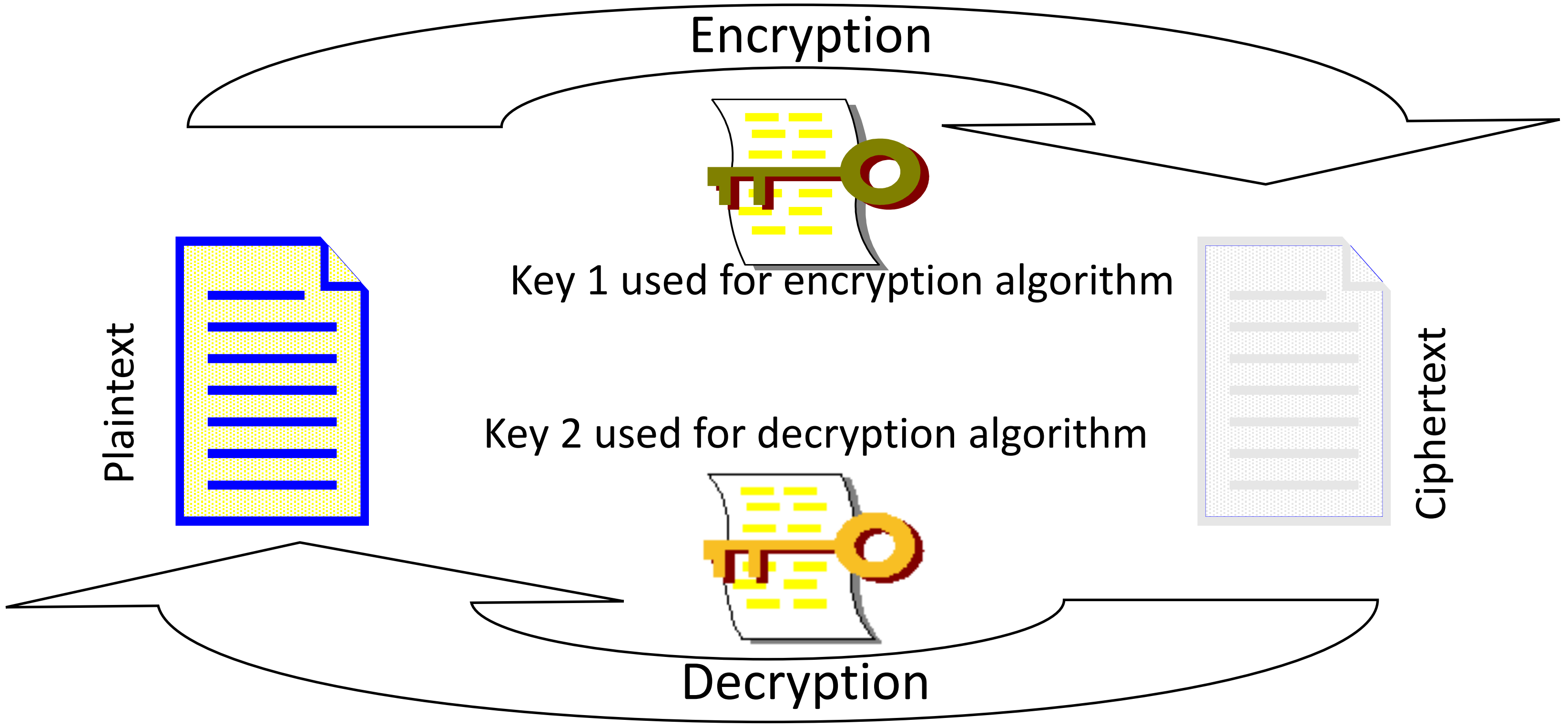
```
sudo htpasswd -c /etc/apache2/.htpasswd your_user
```

Apache Secure HTTP Configuration

- Secure HTTP or enabling https makes all traffic between server and client encrypted so that any middle party cannot see the clear contents.
- HTTPS is possible as it uses Public Key Infrastructure (PKI) along with Asymmetric Encryption to create secure channel between two parties.
- Authentication and Authorization is done by associating Digital Certificates and the process is handled by PKI guidelines.
- As a standard HTTPS uses port 443 by default and if needed we can change it to any non-standard port.
- Best Practice is to disable port 80 or plain HTTP to secure traffic between client and server eliminating MITM attacks

Asymmetric Encryption

- Whitfield Diffie and Martin Hellman (1975)
- Each party has two keys (public and private)
- Anything encrypted with key1 can only be decrypted with key2 ---- Asymmetric!!!


Encryption
Key 1 used for encryption algorithm
Plaintext
Ciphertext
Key 2 used for decryption algorithm
Decryption

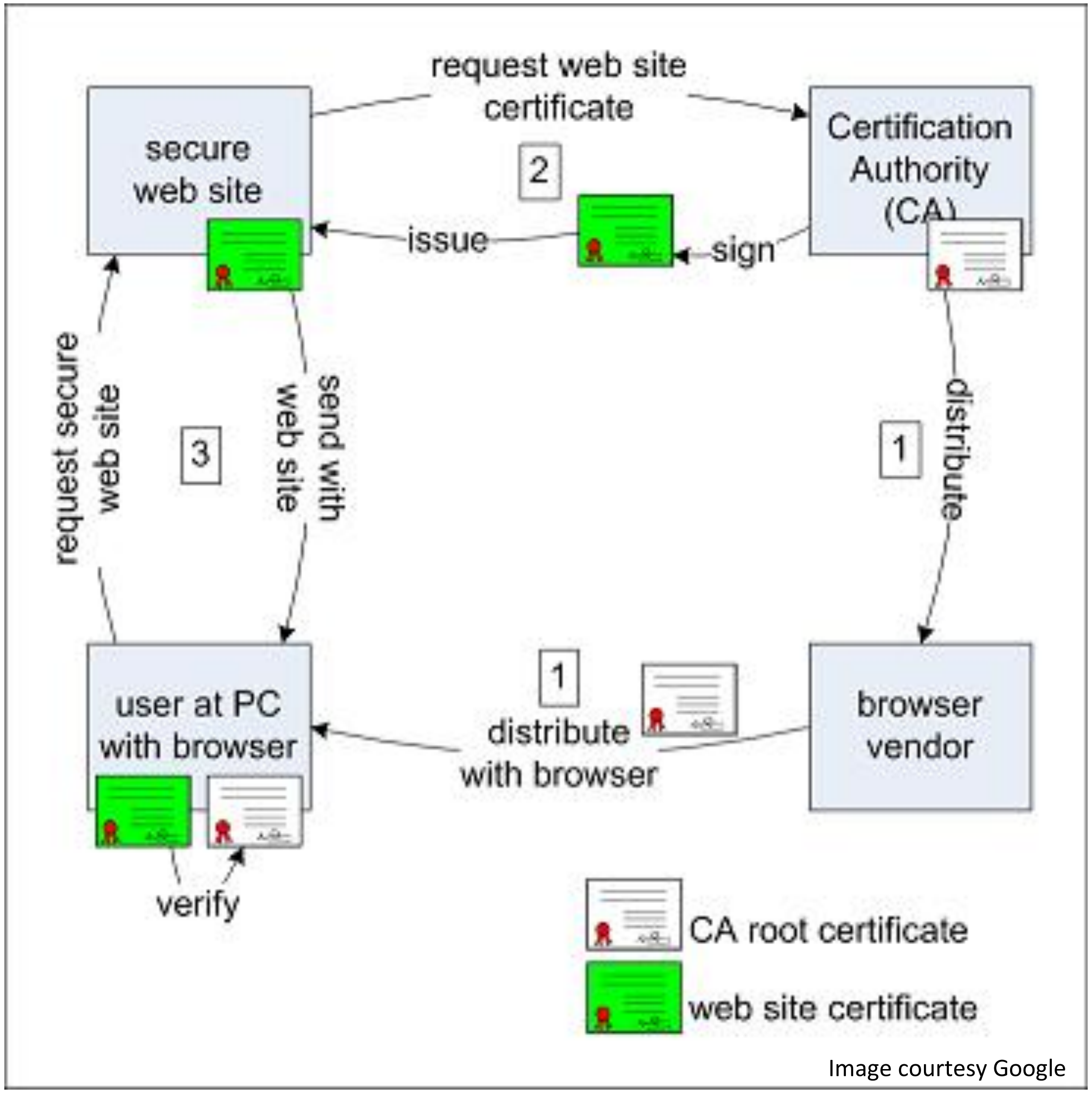
Digital Certificates

- Introduced in 1978 in a Bachelor's Thesis by Kohnfelder
- Today a ITU-T Standard called as X.509
- Certificates are implemented using Signatures
- therefore Authentication can be implemented using Certificates
- In https communication Public Key Certificates are very essential. These are Certificates created with the public key and signature of server and its details such as FQDN.
- Note to remind is that public keys are not a secret at all. They can be used to prove someone's identity another. But those public keys must be verified by a 3rd party who is already trusted by both parties.

Public Key Infrastructure (PKI)

- Certificates containing public keys are certified by a trusted Certificate Authorities (CA).
- PKI procedure:
  - Generate key pair
  - Include one key as the public key and sends a Certificate Signing Request (CSR) to the CA
  - Registration Authority (RA) checks your identity and let the CA know that is really you.
  - Once you are validated, CA signs your certificate and send you the certified public certificate.
- Once you receive you public certificate you can send it to some other party to communicate with each other. When the other party gets your Public Certificate they will validate it if the signed CA is also a trusted party by them.
- Now they can encrypt anything with the key and decryption is only possible with your Private key. Therefore make sure you keep your private keys a very Secret.

Public Key Infrastructure (PKI)



TLS/SSL with Apache

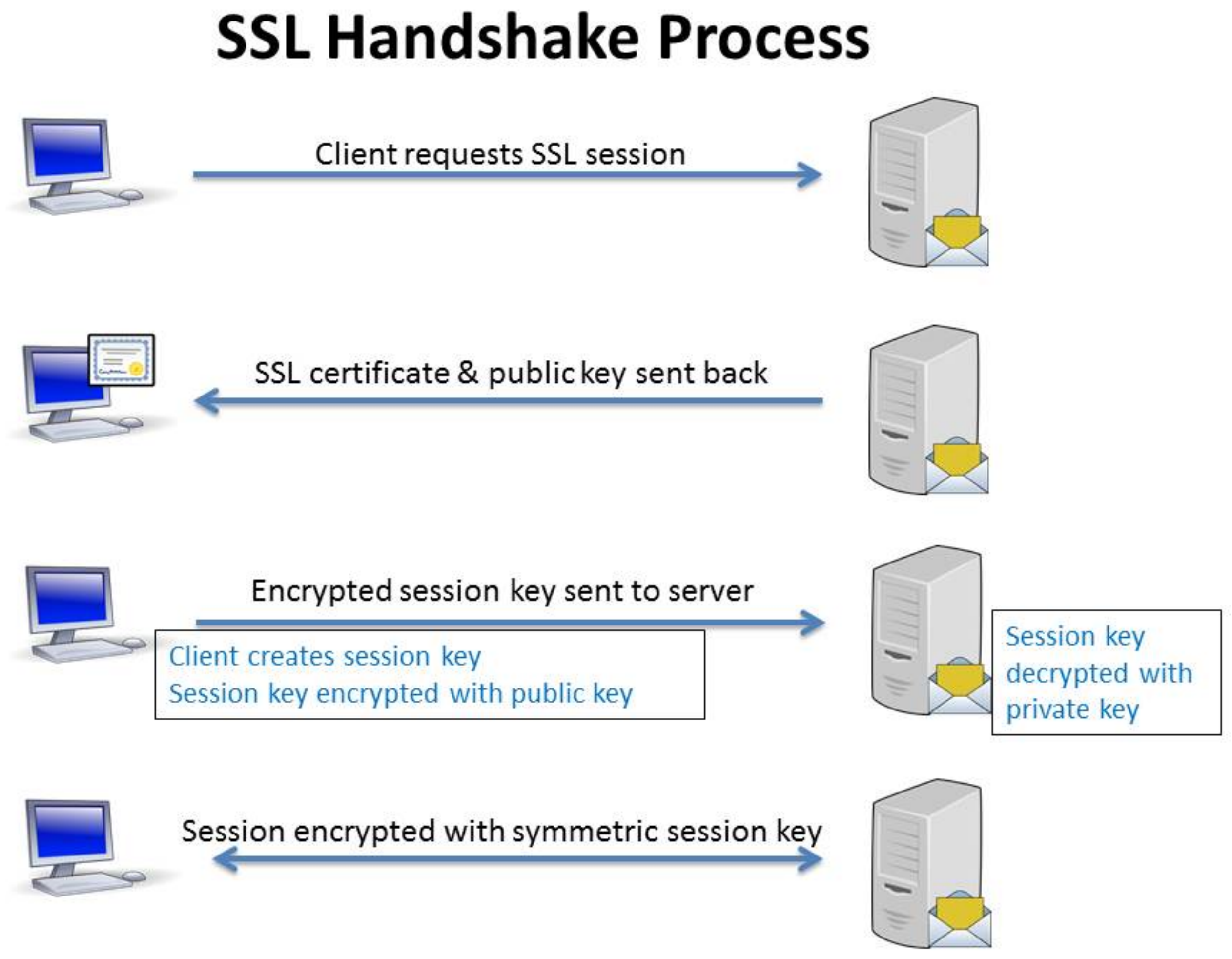

SSL Handshake Process
Client requests SSL session
SSL certificate & public key sent back
Encrypted session key sent to server
Client creates session key
Session key encrypted with public key
Session key decrypted with private key
Session encrypted with symmetric session key

Image courtesy Google

Enable HTTPS

- First step : Generate Key Pair
- Next: Create a CSR and send it to a selected CA.
- Once Received the signed Certificate associate it in apache virtual host configuration.
- Enable SSL modules for apache.
- Redirect all port 80 traffic to port 443

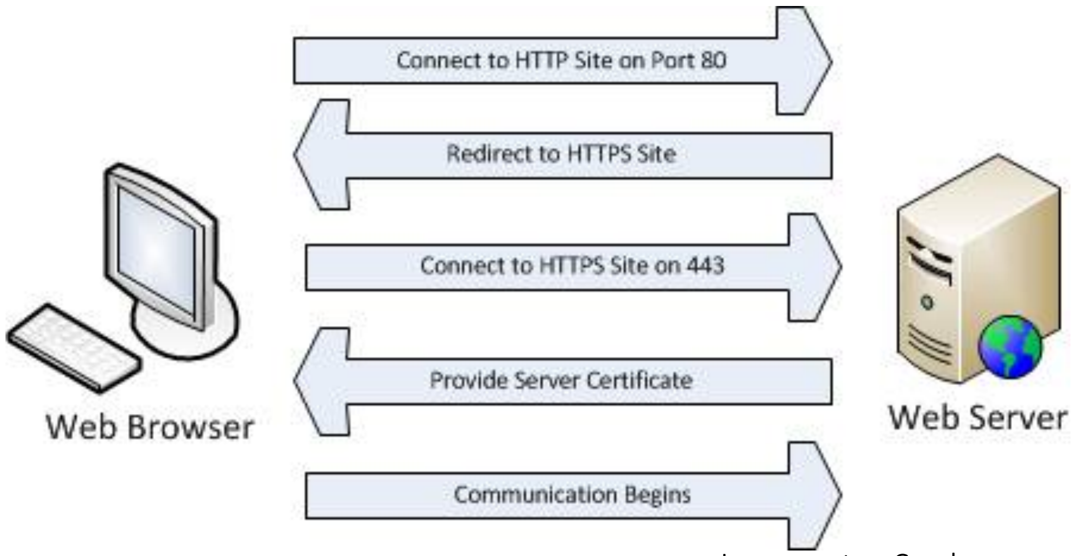

Connect to HTTP Site on Port 80
Redirect to HTTPS Site
Connect to HTTPS Site on 443
Provide Server Certificate
Communication Begins
Web Browser
Web Server

Image courtesy Google

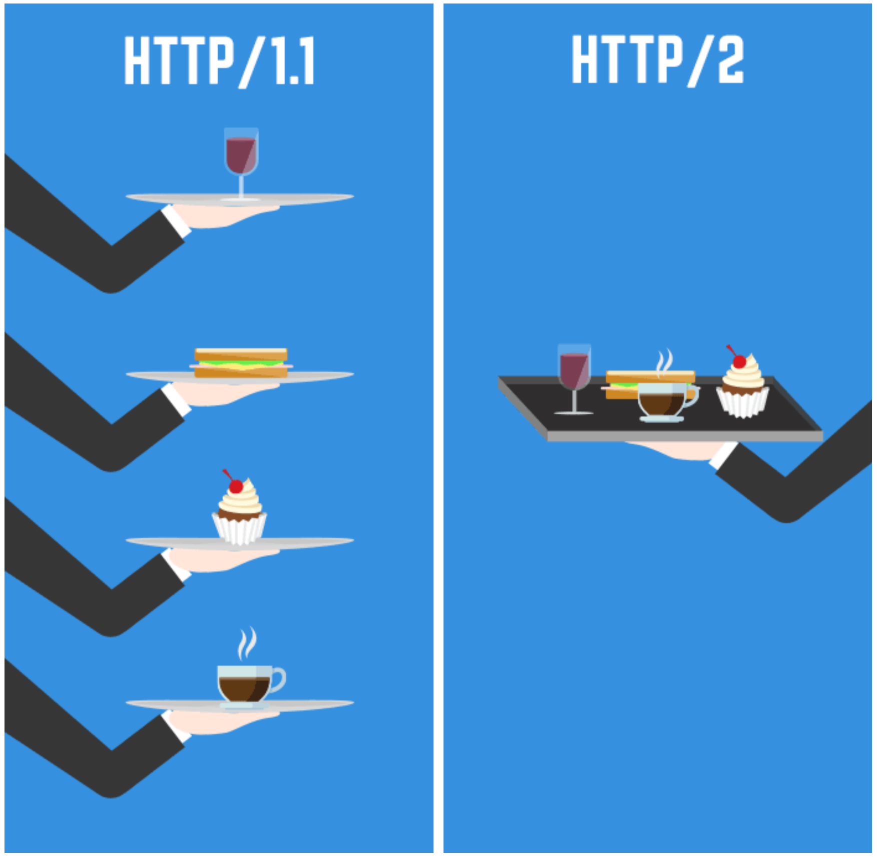
Building an Encrypted Internet.

- Let's Encrypt Certificate Authority launched their free service in 12th April 2016.
- This is a major project headed by Internet Security Research Group (ISRG) and is sponsored by Mozilla, Google, Cisco, Akamai and many more giants in Internet.
- They have issued more than 10.7M certificates as of Nov 7th and all major browsers also support Let's Encrypt CA.
- Very easy to get a certificate to any use in internet, the only condition is we had to have a valid DNS that can be reached through Internet.

HTTP version 2

- We are using HTTP 1.1 since 1997 / 1999
- HTTP2 was published (RFC 7540) in May 2015 and as of the end of last year all major browsers supports.
- In October 2016, 10Millionth supported web sites milestone passed
- Version 2 is more fast and supports encrypts – all supporting browsers supports http2 on TLS 1.2 and above.
- Major Web servers support http2 and for Apache it supports your versions from 2.4.17.
- Most Major CDN's support http2 and that is because;
- https://http2.akamai.com/demo

HTTP version 2



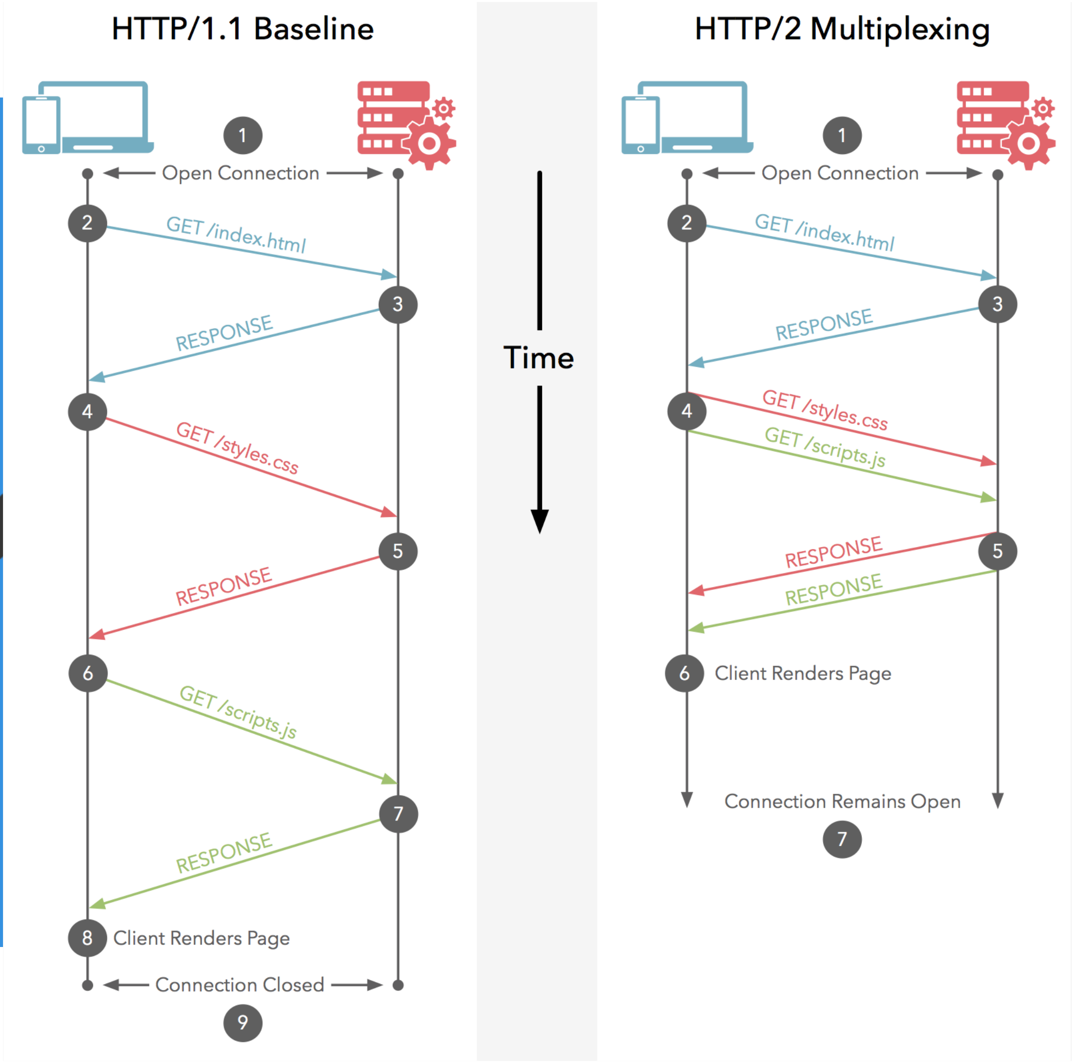


Image courtesy Google

UFW

- Uncomplicated Firewall is the built-in Ubuntu Firewall that can be used to protect services from External access
- UFW is an extension of Linux IP-Tables and very easy to understand and create rules.
- By default UFW comes with status OFF but we can enable it by
  - sudo ufw enable
- But as we configure our servers mostly by SSH make sure you allow SSH access before enabling ufw. As default firewall action is to deny all incoming traffic SSH connections will be blocked unless allowed.
- More UFW in Hands-on….

Questions

Thank You

Thilina Pathirana/UoK

Email: thilina@kln.ac.lk