Midterm ch.10-12

> ONLY some concepts including.

The HTTP Protocol

- What Does the WWW Server Do?
  1. Enables browser requests
  2. Mainly provides: Support for retriving hypertext documents; Manage access to the Web site; Provides several machanisms for executing server-side scripts
     - Common Gateway Interface (CGI)
     - Application Program Interface (API)
     - Direct Module Interfaces (SAPI)
  3. produce log files and usage statistics
- How Does a Web Server Communicate?

  HTTP: HyperText Transfer Protocol, lightweight, version 1.1
- MIME(Multipurpose Internet Mail Extensions) Media Types
- Intermediary
  1. proxy: forwarding agent, eg. caching proxy
  2. gateway: receiving agent, eg. convery HTTP traffic to another
  3. tunner: acts as a relay point between 2 connections w/o changing the message, eg. transport non-HTTP data over one or more HTTP connections, w/o looking at the data.
- Persistent Connections
  - TCP uses a three-way handshake: client sends SYN, server replies ACK/SYN, client responds with ACK
  - keep-alive, default is persistent -> connection: close

    permits multiple connections in paralel. but may slower, as they may compete for available bandwidth
- HTTP Request Methods
  - GET(..by the request URL), HEAD, POST(request includes a block of data in the message body), PUT, DELETE, TRACE, OPTIONS
- HTTP Headers
  - general
  - Request
    - Accept, Accept-Charset, Accept-Encoding: compress, gzip, Accept-Language, Authorization, From, Host, Referer,

        - eg. UserAgent(reports the client software name and version and possibly platform)

          userAgent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_3) Apple**WebKit**/604.5.6 (KHTML, like Gecko) **Version**/11.0.3 **Safari**/604.5.6
        - Byte Range Headers: If-Range: "entity-tag"(guarantee that any new byte range responses are generated from the same source object.); Range, Accept-reanges, Content-Range
    - Response
        - Server
        - **Age**(in seconds since response was generated)
        - **Location**(indicates that re-direction is desired)
        - WWW-AUTHENTICATE(sent with **401 Unauthorized** status code, it includes authorization parameters)
        - Retry-after(used with Service Unavailable status, indicates requested data will be available in X seconds)
        -

10 proxy-id	**"Revalidation failed"**
10 Response is stale	11 Revalidation failed
12 Disconnected operation	13 Heuristic expiration
14 Transformation applied	99 Miscellaneous warning

    - Entity
        - for web cache validation, and which allows a client to make conditional requests -> caches to be more efficient, and saves bandwidth
        - An ETag is an opaque identifier assigned by a web server to a specific version of a resource found at a URL
        - An ETag is a serial number or a checksum that uniquely identifies the file

          `If-None-Match` condition header -> if match, `304 Not Modified`
        - Allow, Content-Base, Content-Endoing, Content-Language, Content-Length, Content-Location, Content-MD5(created by the web server), Content-type
        - **Etag**(specifies the entity tag for the object, which can be used for re-validation; tags are unique ids determined by the server; this line is normally sent as a response)
        - **Expires**(specifies the expiration date/time of the object; a cached copy should not be used beyond; Expires 0/now is immediate)
        - **Last-Modified**(...creation/last modification time of the object on the web server)
    - extension

- Status-Line: *(general-header | response-header | entityheader) CRLF [message-body]
  Status-Line = HTTP-Version Status-Code Reason-Phrase CRLF
- Status Codes

1XX	**Informational**
100	Continue, the client may continue with its request; used for a PUT before a large document is sent
101	Swirching Protocols

2XX	**Successful**
200	**OK**, request succeeded
201	Created, result is newly created
202	Accepted, created later
203	Non-authoritative information, info returned from a cached copy and may be wrong
204	**No content**, response is intentionally blank -> not change the page
205	Reset Content
206	Partial content, eg. a byte range response

3XX	**Redirection**
300	Multiple choices, the document has multiple respresentations
301	**Moved permanently**, Location: header
302	Moved temporarily
303	See other, **automativally redirect** the client to a different YRL
304	**Not modified**, the client or procy copy is still up-to-date
305	Use proxy, make the request via the proxy
306	Proxy Redirection
307	Temporary Redirect

4XX	**Client Error**		
400	**Bad request**, server could not undersand	**408**	**Request Timeout**
401	**unauthorized**, authorization changes	409	Conflict
402	**Payment Required**	**410**	**Gone**, requested resource is no longer available
403	**forbidden**, server refuses to fulfill request	411	Length Required
404	**Not found**	412	Precondition failed
405	Method not allowed	413	Request entity too large
406	Not Acceptable	414	Request URL too large
407	**Proxy Authentication Required**	415	Unsupported media type

5XX	**Server Error**
500	**Internal server error**
501	Not implemented, because server does not support it
502	**Bad gateway**, intermediate proxy server received a bad response
503	Server unavailable, due to **high load or maintenanc**e on the server
504	Gateway timeout, intermediate proxy server timed out waiting for response from another server
505	HTTP version not supported

- Others
  - `META HTTP-EQUIV`: for authors of HTML documents to set HTTP headers, in particular HTTP responses
  - `X-Frame-Options: SAMEORIGIN`: Provides Clickjacking protection. /DENY/ALLOW-FROM https://example.com/
  - **HTTP Strict-Transport-Security (HSTS)**
  - Cross-origin resource sharing (CORS)

    brower send `Origin: url`, server send `Access-Control-Allow-Origin: url` if allows, or the browser will deliver an error

The Web Server

- Web Server Features
  - Main features

    platform they run on; complete support for HTTP 1.1; Multithreading, load balancing
  - Security features

    ability to provide IP address/domain name restriction; Support for secure transactions: **SSL**; Ability to act as a proxy server
  - 1. Document Root

       the document tree is organized by the web site administrator

       use the **host name or IP address**
    2. Directory Listing: Turn off, Remove “Indexes” from Options Indexes in `httpd.conf`
    3. Authentication: Basic 0<V%L:EB.G-E8W))$J

       Server decodes name and password and checks it against its password file
    4. Proxy Servers, Caching, CGI Scripting, Application Program Interface
    5. Creting Server-Side Application:
       - Web Servers offer several mechanisms:
         1. Application Programming Interface (**API**)
         2. Common Gateway Interface (**CGI**)
         3. **J2EE / .NET** interfaces
         4. Chrome V8 JavaScript Engine (Google): **Node.js**
         5. **Direct Module Interfaces (PHP)**
       - Microsoft **IIS** server-side supports:
         1. CGI applications
         2. API applications compiled as **DLL**
         3. Active Server Pages (written in VBScript)
         4. **ASP.NET** applications (written in C++, VB.NET, C#, or HTML/CSS/JavaScript)
- Web Servers: apache, Microsoft, NGINX, NGINX Software
- Application Servers: Oracle, IBM, Oracle
- Configuring a Server

  server root, document root, location of CGI scripts or server-side components to execute, Access restrictions, Fancy indexing, Sys admin e-mail, Other features
- Port Numbers:
  - System ports: 0 - 1023, are reserved for certain functions, typically assigned by **IANA** (Internet Assigned Number Authority)
  - Well-known TCP / UDP ports: 20,ftp,ftp data connection; 21,ftp,ftp control connection;

25,SMTP,simple mail transfer protocol, 53,DNS, domain name service; 80,http, hypertext transfer protocol;110, pop3, post office protocol; 443, https, secure http (SSL)

- Dynamic ports 49152 - 65535

- **Apache** Web Server: A set of PAtCHEs to NCSA□s httpd
  - server functionality is available through modules which are either built-into or loaded into the server
  - 1. On UNIX, Apache starts several processes to implement the server and handle requests
    2. On Windows there are only two processes, a parent and a child that handles all requests.
  - Key directories

    conf(contains **configuration** files); **htdocs**(html files); **logs**(logging data); cgi-bin (**executable** scripts); icons; src
  - Apache idles, listening to an IP address and port
  - Some Settings
    - httpd.conf: `KeepAlive`(On, allow persistentconnections), ...
    - http-mpm.conf(Server-pool size regulation): `DirectoryIndex index.html`(use as a pre-written HTML), Document root, ...
    - scripting: `ScriptAlias fakename realname`
  - Authentication
    1. file `access.cong`
    2. per-directory access rules: `.htaccess`, slows down the server, is initially off
  - `<Limit>` controls which clients can access a directory
  - Virtual Hosting
    1. address-based virtual hosting uses IP addresses
    2. name-based hosting allow a single IP address to have multiple identities. eg.

```
NameVirtualHost 10.0.0.1
<VirtualHost 10.0.0.1>...</VirtualHost>
<VirtualHost 10.0.0.1>...</VirtualHost>
```

  - Content Negotiation: content-type, encoding, language dimensions

- Apache as a Proxy Server
  1. proxy servers are frequently used in **fire walled** environments to provide a single point of contact between the users inside and the Web outside the firewall.
  2. A second function commonly associated with proxy servers is **caching**.

     e.g. `AddModule mod_proxy.c`

     ProxyRequests, ProxyBlock, NoCache ....

- Web Server Log Files

- Host(client hostname or IP address); ident(remote identity, often a dash)

  usr, time(date/time request is received), req(TTP request)

  s1(server's HTTP response status code), c1(Content-length)
- eg. marmot.usc.edu - - [17/Feb/2008:23:21:09 -0800] "GET / HTTP/1.0" 200 1316
- error_log
  - No file matching URL -> The file was not found
  - user csci571 not found -> authorization entered is incorrect
  - Premature end of script headers -> Perl script fails
  - -> Undefined subroutine
- Analyze Log Files: Google Analytics

PHP Tutorial

- 'PHP: Hypertext Preprocessor' (recursive backronym)
  - Open-source, server-side scripting language, ...
  - Interpreted language, scripts are parsed at run-time
  - Executed on the server-side
  - Source-code not visible by client
  - Various built-in functions allow for fast development
  - Compatible with many popular databases
  - Provides
    - SAPI, Server Application Programming Interface (Direct Module)
    - ISAPI, The Internet Server Application Programming Interface (IIS)
    - CGI interface
    - CLI (Command Line Interface)
- Variables
  1. Global variables can be used anywhere (declared outside a function)
  2. Local variables restricted to a function or class
  3. Variable declared "static" do not disappear when a function is completed
  4. Superglobals: predefined variables available in all 'scopes'

     eg. Form variables($_POST, $_GET)

     Server variables ($_SERVER)

     State variables ($_COOKIE, $_SESSION)

     Environment variables ($_ENV)
  - eg. **PHP_SELF**, an index variable that returns the current script being executed, including its name and path (Ex: $_SERVER['PHP_SELF'])

    $_GET, an associative array of variables passed to the current script

$_POST, an associative array of variables passed to the current script

- Variable names are case sensitive

- Constants

  Start with letter or underscore (_) followed by letters, numbers or underscores

  Constant names use all uppercase letters

  Use the define() function to create a constant `define("CONSTANT_NAME", value);`

  cannot be changed, global scope

- others
  - Strings in single quotes (‘) are not interpreted or evaluated by PHP
  - function names are not case sensitive

- eg.

```php
//Source of displayfile.php Script
<?php
$ourFileName = "apple.txt";
$ourFileHandle = fopen($ourFileName, "r") or die("can't open file");
$file = fread($ourFileHandle, filesize($ourFileName));
fclose($ourFileHandle);
header("Content-type: text/plain");
echo $file;
?>

// rediction.php Script
<?php if($_POST["submit"]): ?>
<?php header("Location: $_POST[url]"); ?>
<?php else: ?>

// Creating a File List
if ($handle = opendir('/home/scf-22/csci571/public_html/Special/php_ex/translated')) {
    while (false !== ($entry = readdir($handle))) {
        if ($entry != "." && $entry != "..") {
            echo "<a href='$entry'>$entry</a><br/>\n";
        }
    }
    closedir($handle);
}
```

```php
// Example Using POST
<?php
    include_once("inc.php");
    if(isset($_POST["submit"])):
?>
<?php print_array($_POST); ?>
<?php echo isset($_POST["first_name"]) ?$_POST["first_name"] : "" ?>
```

```php

// Echoing Inputs Scripts
<?php
    foreach($_GET as $key => $value) {
        if ($key !== "submit") {
            echo $key . " = " . $value . "\n";
        }
    }
?>
```

- PHP Sessions
  - Sessions store their identifier("**Session ID**") in a cookie in the client's browser
  - `session_start()` , Session variables are then set and retrieved by accessing the global `$_SESSION[]`, this is virtually invisible tothe user.
  - `session_destroy();`, `unset($_SESSION);`
- Connecting to a Database

```php
<?php
// setup a query
$link = mysql_connect($host, $user, $password);
mysql_select_db($database); //should be $link
$query = "SELECT * FROM $table WHERE id = '$id';";
$result = mysql_query($query);

//Display a table
echo "<table>";
While ($row = mysql_fetch_array($result)) {
    echo "<tr>";
    foreach ($row as $key => $value) {
        echo "<td>$value</td>";
    }
    echo "</tr>";
}
echo "</table>";
mysql_close($link);
?>
```