

# **Lecture**

## **The Web Server**

# Outline

- Available Web Servers
- Server Features
  - Document Root
  - Authentication
  - Proxy Servers
  - Caching
  - CGI Scripting
  - Application Program Interface
- Configuring a Server
- Analyzing a Server's Performance
- Server Log Files

# What Does the WWW Server Do?

- Enables browser requests
- Mainly provides
  - Support for retrieving hypertext documents
  - Manages access to the Web site
  - Provides several mechanisms for executing server-side scripts
    - Common Gateway Interface (CGI)
    - Application Program Interface (API)
    - Direct Module Interfaces (SAPI)
  - provides log files and usage statistics

# **What to Look for in a Web Server**

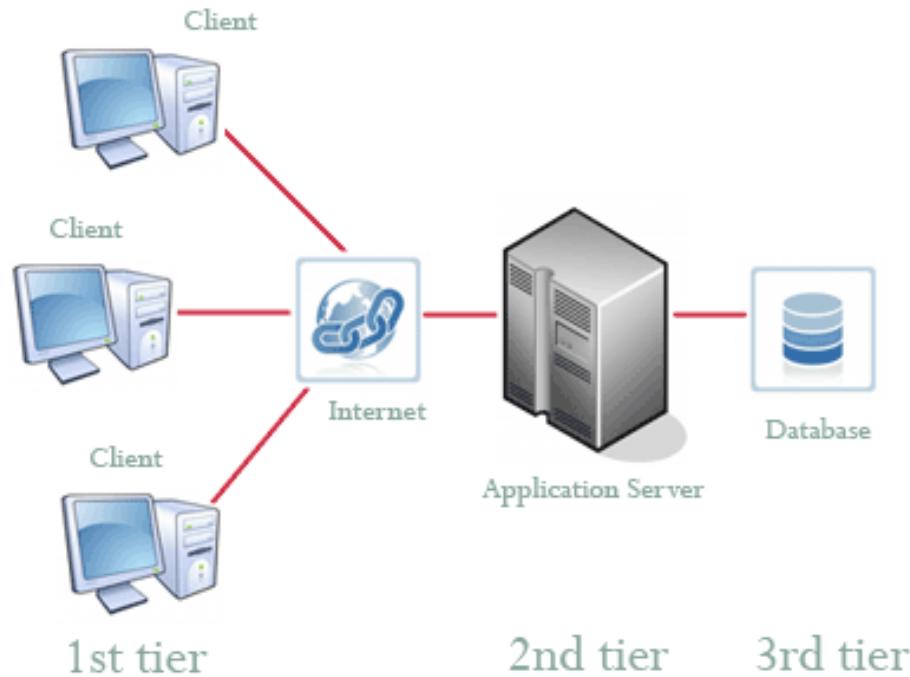
- Main features
  - platform they run on;
  - complete support for HTTP 1.1
  - Multithreading, load balancing
- Security features
  - ability to provide IP address restriction
  - ability to provide domain name restriction
  - Support for secure transactions: SSL
  - Ability to act as a proxy server

# How Servers Handle Multiple Requests

- For each request, a complete copy of the server is made and executed
  - Initially a parent process waits for requests; for each new request a child process is spawned
- Or, a single server program handles many requests simultaneously (multithreaded)
  - the server must keep track of all requests and switch between them as needed
  - writing multithreaded programs is easier in a language that supports multiple threads, e.g. Java

# Application Web Server

- An **application server** is software that typically interfaces one or more databases to convey processed data to and from a user interface such as a web browser



- An application server acts as a set of components accessible through an API
- For web applications, these components are usually performed in the same machine where the web server is running, and their main job is to support the construction of dynamic pages.
  - For example, Apache Tomcat is a popular, light-weight application container, but not a full application server as it doesn't provide the services specified in the J2EE specification. The web modules include servlets and JavaServer Pages. Business logic resides in Enterprise JavaBeans (EJB).

# Some Available Web & Application Servers

| Publisher                  | Product             | Platform  | URL   |
|----------------------------|---------------------|---|---|
| <b>Web Servers</b>         |                     |   |   |
| <i>apache</i>              | apache              | Win32/UNIX  | <a href="http://httpd.apache.org/">http://httpd.apache.org/</a>     |
| <i>Microsoft</i>           | IIS 7-10.0          | Win32   | <a href="http://www.iis.net">http://www.iis.net</a>                 |
| <i>NGINX</i>               | nginx (open source) | Linux, BSD, Win32   | <a href="http://nginx.org">http://nginx.org</a>                     |
| <i>NGINX Software</i>      | NGINX Plus          | Ubuntu, AWS<br>Google Cloud, Microsoft Azure  | <a href="http://www.nginx.com">http://www.nginx.com</a>             |
| <b>Application Servers</b> |                     |   |   |
| <i>Oracle</i>              | GlassFish 4.1       | Solaris/Win32/Linux<br>Java EE Reference Implementation   | <a href="https://glassfish.java.net">https://glassfish.java.net</a> |
| IBM                        | WebSphere V 9       | Win32/UNIX/Linux<br><a href="http://www.ibm.com/software/websphere/">http://www.ibm.com/software/websphere/</a>                               |   |
| <i>Oracle</i>              | WebLogic Server 12c | Win32/UNIX/Linux<br><a href="https://www.oracle.com/middleware/weblogic/index.html">https://www.oracle.com/middleware/weblogic/index.html</a> |   |

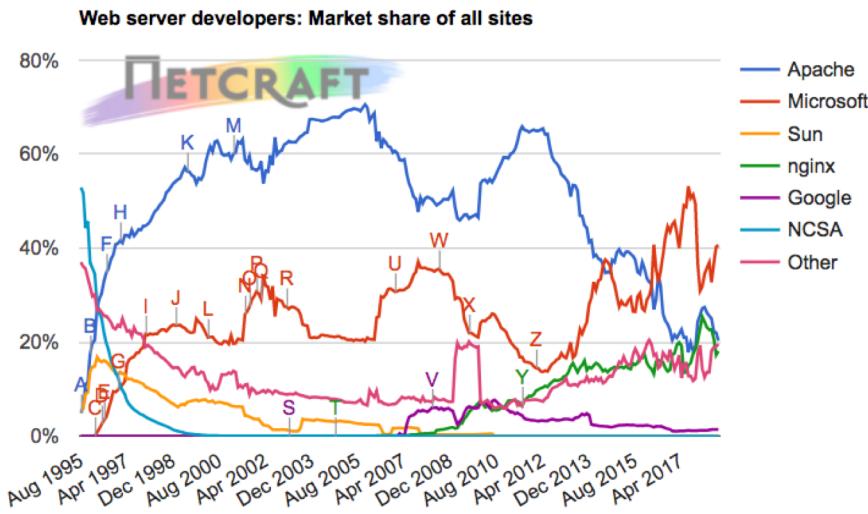
For a comparison of web servers see <http://www.serverwatch.com/>, click on Server Comparison Tool

# Web Server Usage

- Netcraft has identified more than 1,661,000,000 sites (8/2018)
  - Apache no longer dominates Internet servers
  - Microsoft on top, NGINX closing in to Apache at #2
- Statistics on intranets are difficult to determine
  - <http://news.netcraft.com/archives/category/web-server-survey/>

*In July 2014, for the first time since measurements began, Microsoft's IIS had taken the lead, and retook it in July 2016.*

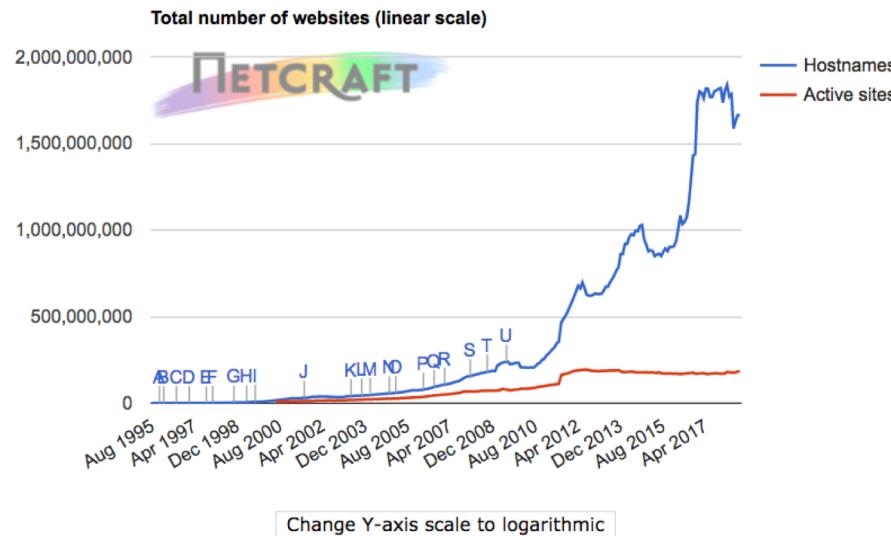
*Almost 200,000 Apache servers (7%) are Western Digital MyCloud servers, consumer devices that can be accessed using public hostnames similar to device1000000-a1b2c3d4.wd2go.com*



| Developer | July 2018   | Percent | August 2018 | Percent | Change |
|-----------|-------------|---------|-------------|---------|--------|
| Microsoft | 670,209,364 | 40.28%  | 675,366,733 | 40.65%  | 0.36   |
| Apache    | 366,023,119 | 22.00%  | 336,937,568 | 20.28%  | -1.72  |
| nginx     | 287,045,450 | 17.25%  | 300,644,222 | 18.10%  | 0.84   |
| Google    | 23,347,290  | 1.40%   | 23,143,610  | 1.39%   | -0.01  |

# Web Server Survey Takeaways

- From Netcraft August 2018 survey we can make these observations
  - Apache, Microsoft, and nginx** have each achieved their widespread popularity in the server market due to their general availability, but many large companies in the web industry choose to roll out their own solutions.
  - Google's custom server software ("GSE")** operates on over 23.1 million hostnames and 2.0 million domains. Google open sourced part of its software, the Google Servlet Engine, at the end of 2008, but the software has not received any official public updates in 9 years.
  - Taobao**, China's largest online marketplace and part of Alibaba Group, developed their own **fork of nginx**. The fork, known as **Tengine**, was released back to the community as open source, leading to wider adoption. Tengine now runs on 56.8 million sites, making it the fourth most popular web server by this metric, despite it having only 1.2 million domains.



# **Web Server Features**

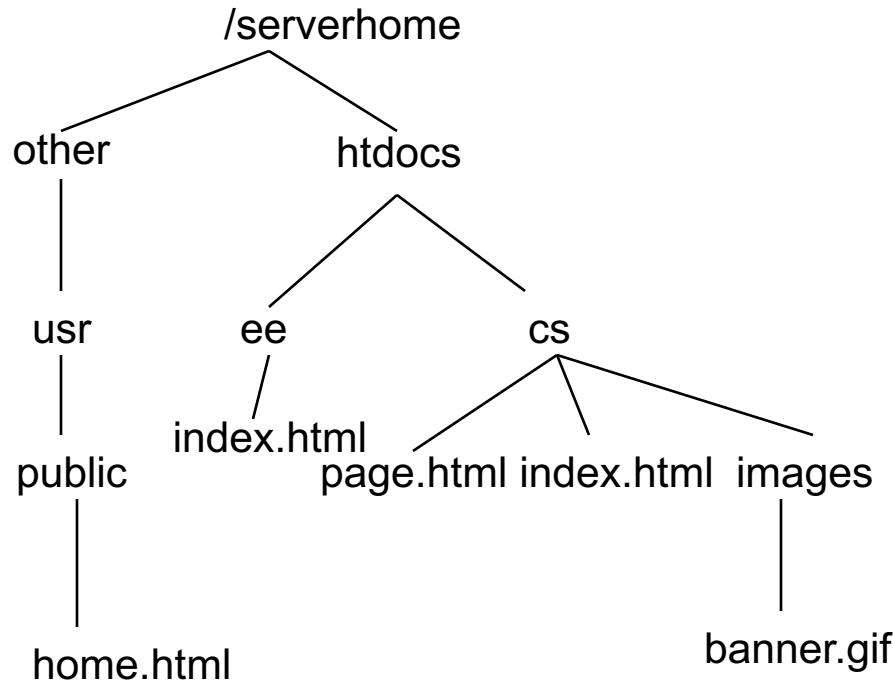
- Document Root
- Authentication
- Proxy Servers
- Caching
- CGI Scripting
- Application Program Interface

## **Web Server Features - Document Root**

- The Web server has access to a tree of files that it can deliver
  - the files are created by content providers
  - files may contain text, images, sound, video, other programs, etc.
- the document tree is organized by the web site administrator
- The root of the document tree is given to the web server when it starts

# Managing the Document Tree

http://domain/cs is mapped to /serverhome/htdocs/cs/index.html



The document root is /serverhome/htdocs

index.html

```
<A HREF=page.html>Click here</a>

<A HREF=images/banner.gif>Display Image</a>

<A HREF=..//ee/index.html>Click Here</a>

<A HREF=/serverhome/other/usr/public/home.html>
```



*The last link is an error*

# Web Server Features

## Virtually Hosted Document Roots

- Hosting multiple web sites by the same web server
- It uses the host name or IP address to distinguish the document roots, e.g.

```
GET /index.html HTTP/1.0
```

```
Host: www.hardware.com
```

might return documents from /htdocs/hardware, while

```
GET /index.html HTTP/1.0
```

```
Host: www.antiques.com
```

might return documents from /htdocs/antiques

- This might be configured in, say apache by writing

```
<VirtualHost www.hardware.com>
```

```
<VirtualHost www.antiques.com>
```

```
ServerName www.hardware.com
```

```
ServerName www.antiques.com
```

```
DocumentRoot /htdocs/hardware
```

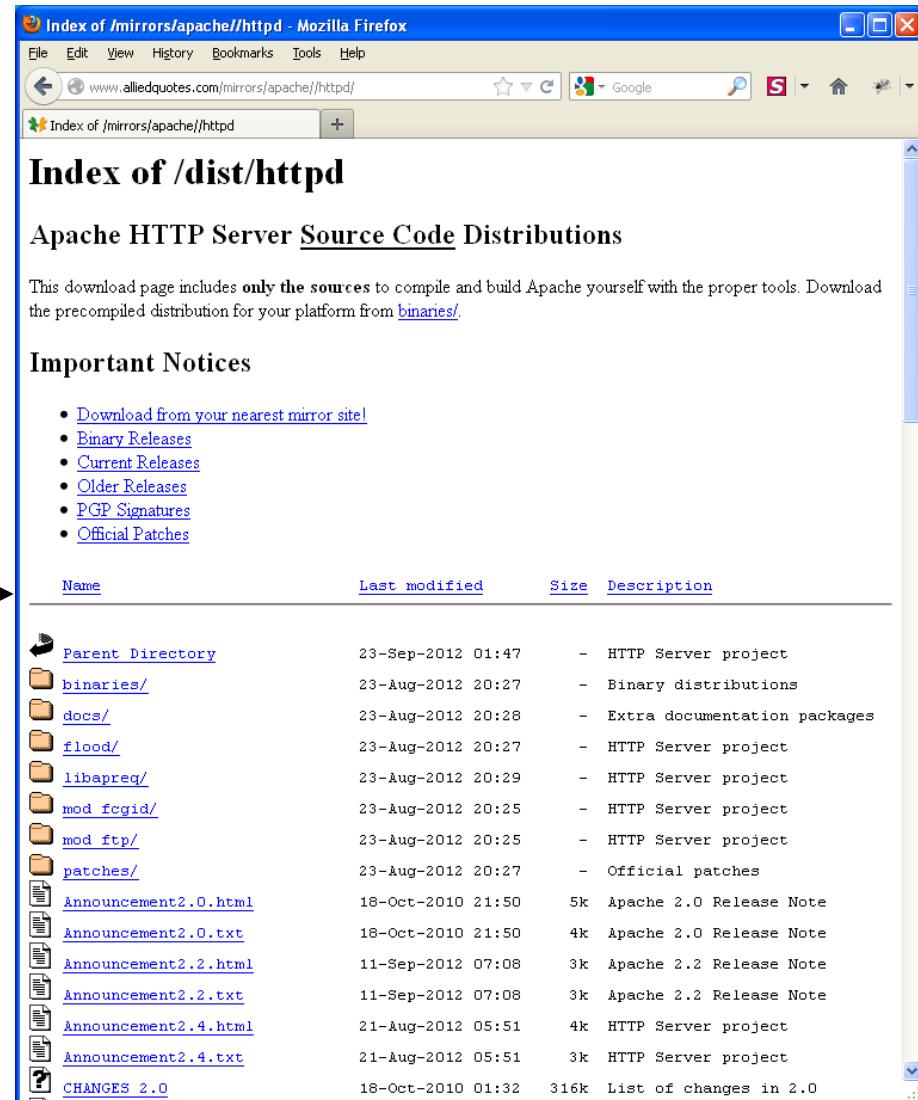
```
DocumentRoot /htdocs/antiques
```

```
</VirtualHost>
```

```
</VirtualHost>
```

# Web Server Features - Directory Listing

- When the URL path resolves to a directory, not to a file, a web server can be configured to return more than just a default file; to the right you see the server returning a list of files in the  directory including special icons for files and folders
- Important Note:**  
Turn off this feature! Remove "Indexes" from **Options Indexes** in httpd.conf



The screenshot shows a Mozilla Firefox browser window displaying a directory listing for the URL <http://www.alliedquotes.com/mirrors/apache//httpd/>. The title bar reads "Index of /mirrors/apache//httpd - Mozilla Firefox". The page content includes a header titled "Index of /dist/httpd" and a sub-section titled "Apache HTTP Server Source Code Distributions". It states: "This download page includes only the sources to compile and build Apache yourself with the proper tools. Download the precompiled distribution for your platform from [binaries!](#)". Below this is a section titled "Important Notices" with links to various resources. The main part of the page is a table listing files and directories with their names, last modified dates, sizes, and descriptions. Special icons are used for files and folders.

| Name   | Last modified     | Size | Description                  |
|--|-------------------|------|------------------------------|
|  <a href="#">Parent Directory</a>       | 23-Sep-2012 01:47 | -    | HTTP Server project          |
|  <a href="#">binaries/</a>              | 23-Aug-2012 20:27 | -    | Binary distributions         |
|  <a href="#">docs/</a>                  | 23-Aug-2012 20:28 | -    | Extra documentation packages |
|  <a href="#">flood/</a>                 | 23-Aug-2012 20:27 | -    | HTTP Server project          |
|  <a href="#">libapreq/</a>              | 23-Aug-2012 20:29 | -    | HTTP Server project          |
|  <a href="#">mod_fcgid/</a>             | 23-Aug-2012 20:25 | -    | HTTP Server project          |
|  <a href="#">mod_ftp/</a>              | 23-Aug-2012 20:25 | -    | HTTP Server project          |
|  <a href="#">patches/</a>             | 23-Aug-2012 20:27 | -    | Official patches             |
|  <a href="#">Announcement2.0.html</a> | 18-Oct-2010 21:50 | 5k   | Apache 2.0 Release Note      |
|  <a href="#">Announcement2.0.txt</a>  | 18-Oct-2010 21:50 | 4k   | Apache 2.0 Release Note      |
|  <a href="#">Announcement2.2.html</a> | 11-Sep-2012 07:08 | 3k   | Apache 2.2 Release Note      |
|  <a href="#">Announcement2.2.txt</a>  | 11-Sep-2012 07:08 | 3k   | Apache 2.2 Release Note      |
|  <a href="#">Announcement2.4.html</a> | 21-Aug-2012 05:51 | 4k   | HTTP Server project          |
|  <a href="#">Announcement2.4.txt</a>  | 21-Aug-2012 05:51 | 3k   | HTTP Server project          |
|  <a href="#">CHANGES 2.0</a>          | 18-Oct-2010 01:32 | 316k | List of changes in 2.0       |

# Web Server Features

## Basic User Authentication

- Basic authentication is supported by all HTTP servers
  - The server administrator creates secure directories accessible via password files maintained by the server
  - Client requests a resource from a secure directory; e.g., **GET /secure/test.html HTTP/1.0**
  - Server responds with authentication request to the client; e.g., **HTTP/1.0 401 Unauthorized**
  - Browser requests user name and password, scrambles them, and retries the request

**GET http://domain/secure/test.html HTTP/1.0**

**Authorization: Basic 0<V%L:EB.G-E8W) )\$J**

- Server decodes name and password and checks it against its password file

# Web Server Features

## Creating Server-Side Applications

- Web Servers offer several mechanisms
  - Application Programming Interface (API)
  - Common Gateway Interface (CGI)
  - J2EE / .NET interfaces
  - Chrome V8 JavaScript Engine (Google)
    - Node.js
  - **Direct Module Interfaces (PHP)**
    - Apache offers a PHP module
- Microsoft IIS server-side supports:
  - CGI applications
  - API applications compiled as DLL
  - Active Server Pages (written in VBScript)
  - ASP.NET applications (written in C++, VB.NET, C#, or HTML/CSS/JavaScript)

# Configuring a Server

- No matter which operating system or server, you will need to define the
  1. location of the server (server root)
  2. location of documents to deliver (document root)
  3. location of CGI scripts or server-side components to execute
- You may also wish to define
  - Access restrictions
  - Fancy indexing
  - Sys admin e-mail
  - Other features

# Port Numbers

- A typical computer will have thousands of ports
- System ports: 0 - 1023, are reserved for certain functions, typically assigned by IANA (Internet Assigned Number Authority), <http://www.iana.org/>
- Well-known TCP / UDP ports:

[http://en.wikipedia.org/wiki/List\\_of\\_TCP\\_and\\_UDP\\_port\\_numbers#Well-known\\_ports](http://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers#Well-known_ports)

- programs running on these ports must be started under root

- TCP/UDP ports used by Apple products:

<http://support.apple.com/kb/HT6175>

- Dynamic ports 49152 - 65535, ports that are freely available (check OS X possible conflicts)

# Some Well-Known Port Numbers

| <b>Port</b> | <b>Protocol</b> | <b>Purpose</b>                   |
|-------------|-----------------|----------------------------------|
| 20          | ftp             | ftp data connection              |
| 21          | ftp             | ftp control connection           |
| 23          | telnet          | telnet session                   |
| 25          | SMTP            | simple mail transfer protocol    |
| 53          | DNS             | domain name service              |
| 70          | gopher          | gopher protocol                  |
| 79          | finger          | finger protocol                  |
| 80          | http            | hypertext transfer protocol      |
| 88          | kerberos        | kerberos authentication protocol |
| 110         | pop3            | post office protocol             |
| 113         | ident           | remote identity service          |
| 119         | nntp            | network news transfer protocol   |
| 143         | IMAP            | email access                     |
| 161         | snmp            | simple network management proto  |
| 162         | snmp            | snmp traps                       |
| 194         | irc             | internet relay chat              |
| 280         | http mgmt       | http management                  |
| 389         | ldap            | lightweight directory access pr  |

## Standard Port Numbers (cont'd)

| <b>Port</b> | <b>Protocol</b> | <b>Purpose</b>                |
|-------------|-----------------|-------------------------------|
| 427         | srvloc server   | location protocol             |
| 443         | https           | secure http (SSL)             |
| 465         | smt�ps          | secure email (SSL)            |
| 535         | iiop            | internet inter-orb protocol   |
| 551         | cybercash       | secure money transaction      |
| 563         | snews           | secure news (SSL)             |
| 614         | ssl shell       | ssl shell                     |
| 636         | ldaps           | secure ldap (SSL)             |
| 989         | ftps            | secure ftp data connection    |
| 990         | ftps            | secure ftp control connection |
| 992         | telnets secure  | telnet (SSL)                  |
| 993         | imaps           | secure imap (SSL)             |
| 994         | ircs            | secure irc (SSL)              |
| 995         | pop3s           | secure pop3 (ssl)             |
| 1080        | socks           | socks protocol, circuit level |

# **The Apache Web Server**

# **Apache Web Server**

- Apache is based on code and ideas developed at NCSA in httpd version 1.3 (early 1995)
- The name “Apache” comes from the fact that it is A set of PATCHES to NCSA’s httpd
- Apache is free and is downloadable from <http://httpd.apache.org>
- Apache version 2.4.25 included in OS X Sierra:  
<https://coolestguidesontheplanet.com/get-apache-mysql-php-and-phpmyadmin-working-on-macos-sierra/>
- Using Apache in Windows 8:  
<http://www.techscio.com/install-apache-on-windows-8/>
- Using Apache in Windows 10  
<http://dalaris.com/how-to-install-and-run-apache-web-server-on-windows-10/>

# Apache Home Page

The screenshot shows the Apache Software Foundation's website at apache.org. The page features a navigation bar with links for Home, About, Projects, People, Get Involved, Download, and Support Apache. Below the navigation is the Apache logo, followed by sections for OPEN, INNOVATION, and COMMUNITY. The INNOVATION section includes a sub-section about Apache projects being defined by consensus-based processes. The COMMUNITY section emphasizes the community nature of the foundation. A large section for APACHE PROJECTS highlights Apache Commons Discovery, Apache Lucene.Net, and Apache Etch. On the right side, there's a sidebar with a Google Custom Search box and links to The Apache Way, Contribute, and ASF Sponsors.

Available for  
many  
platforms:  
Win64 (Win7-  
10), Win32,  
Solaris,  
Linux,  
Mac OS X,  
etc.

# **Apache httpd server**

- The Apache httpd server
  - is a powerful, flexible, HTTP/1.1 compliant web server
  - implements the latest protocols, including HTTP/1.1 (RFC2616)
  - is highly configurable and extensible with third-party modules
  - can be customized by writing 'modules' using the Apache module API
  - provides full source code and comes with an unrestrictive license
  - runs on Windows, Netware, Mac OS X, Linux, and most versions of Unix, as well as several other operating systems

# Configuring Apache

- server functionality is available through modules which are either built-into or loaded into the server
- server instructions
  - apache reads its run-time configuration instructions from text files
- no GUI available (visual start/stop available on Windows)
- 182 configuration directives in the basic package
- On UNIX, Apache starts several processes to implement the server and handle requests
- On Windows there are only two processes, a parent and a child that handles all requests. Within the child all requests are handled by separate “threads”
- See <http://httpd.apache.org/docs/2.2/configuring.html>

# How does Apache Work?

- Apache runs under a multitasking operating system, e.g. UNIX, OS/2, Windows
- the binary is called httpd
- Key directories include:
  - conf contains configuration files
  - htdocs contains html files
  - logs contains logging data
  - cgi-bin contains executable scripts
  - icons set of \*.gif files used e.g. in directory listings
  - src contains actual source code if you download the source distribution
- Apache idles, listening to an IP address and port

## **Configuring Apache - conf Directory**

- In the conf directory there is a file provided with model configuration parameters
  - httpd.conf
- This file must be edited
- Partial contents of the mime.types file

text/html

html htm

text/plain

txt

text/richtext

rtx

video/mpeg

mpeg mpg mpe

video/quicktime

qt mov

video/x-msvideo

avi

x-world/x-vrml

wrl vrml

## Apache Settings - httpd.conf

- Directives are keywords followed by their value, e.g.

```
Listen 9637
```

```
ServerAdmin horowitz@usc.edu
```

```
ServerRoot /home/scf-03/csci571/WebServer/apache_2.2
```

```
ErrorLog logs/error_log
```

```
DocumentRoot "/home/scf-17/csci571c/apache2/htdocs"
```

```
ScriptAlias /cgi-bin/ "/home/scf-17/csci571c/apache2/cgi-bin/"
```

- the above are ones that *must* be set

Note: A complete set of instructions for how to install Apache 2 can be found on the class web page under Download Course Software category

- [http://cs-server.usc.edu:45678/hw/hw5/HW5\\_Apache\\_Installation.pdf](http://cs-server.usc.edu:45678/hw/hw5/HW5_Apache_Installation.pdf)

## **Some Apache Settings - httpd.conf**

```
# Timeout: The number of seconds before receives  
and sends time out  
  
Timeout 300  
  
# KeepAlive: Whether or not to allow persistent  
connections (more than one request per  
connection). Set to "Off" to deactivate.  
  
KeepAlive On  
  
# MaxKeepAliveRequests: The maximum number of  
requests to allow during a persistent connection.  
Set to 0 to allow an unlimited amount. They  
recommend you leave this number high, for maximum  
performance.  
  
MaxKeepAliveRequests 100  
  
# KeepAliveTimeout: Number of seconds to wait for  
the next request  
  
KeepAliveTimeout 15
```

## **Apache Settings – http-mpm.conf**

```
# Server-pool size regulation. Apache dynamically  
adapts to the load it sees --- it tries to  
maintain enough server processes to handle the  
current load, plus a few spare servers to  
handle transient load spikes. It does this by  
periodically checking how many servers are  
waiting for a request. If there are fewer than  
MinSpareServers, it creates a new spare. If  
there are more than MaxSpareServers, some of  
the spares die off.
```

```
MinSpareServers 2
```

```
MaxSpareServers 2
```

```
StartServers      2 #number of servers to start
```

```
MaxClients       2
```

## **Apache Settings - httpd.conf**

```
# DocumentRoot: The directory out of which you will
# serve your documents.
DocumentRoot /home/cscixxx/WebServer/apache/htdocs
# UserDir: The name of the directory which is
# appended onto a user's home directory if a ~user
# request is received.
UserDir public_html
# DirectoryIndex: Name of the file or files to use
# as a pre-written HTML
DirectoryIndex index.html
# FancyIndexing is whether you want fancy directory
# indexing or standard
IndexOptions FancyIndexing
```

## **Apache Settings - Scripting**

```
# ScriptAlias: This controls which directories contain server scripts. Format: ScriptAlias fakename realname
ScriptAlias /cgi-bin/
/home/cscixxx/WebServer/apache/cgi-bin/
# If you want to use server side includes, or CGI outside ScriptAliased directories, uncomment the following lines.
# AddType allows you to tweak mime.types without actually editing it, or to make certain files to be certain types. Format: AddType type/subtype ext1
# To use CGI scripts:
#AddHandler cgi-script .cgi
# To use server-parsed HTML files
#AddType text/html .shtml
#AddHandler server-parsed .shtml
```

# Authentication in Apache

- There are two methods for controlling access to directories
  1. the file access.conf in the conf/ directory can be used
    - placing access directives in a <Directory> block is the preferred method
  2. per-directory access rules can be set by a file placed in a specific directory
    - the name of the file is set by the directive AccessFileName
    - .htaccess is the default name
    - however, using .htaccess slows down the server;
    - NOTE: initially .htaccess is turned OFF
- When the server attempts to retrieve a document, it looks for an access control file in the directory or the parent directory.
- The file it looks for is set by the directive AccessFileName

## Some Sectioning Directives

- The server configuration files use sectioning directives to define special access rights within a section or sub-directory
- Example: <Directory> specifies directory to which directives apply
- Options controls which server features are available
  - None
  - All
  - FollowSymLinks follow symbolic links
  - ExecCGI execution of CGI scripts allowed
  - Includes server-side includes are enabled
  - Indexes will automatically return a list of files in the directory

## **Limit Sectioning Directive**

- <Limit> controls which clients can access a directory; directives within LIMIT include:
  - order in which deny and allow are evaluated
  - deny from host1, host2, ...
  - allow from host1, host2, ...
  - require named-users or group-users or AuthUserFile
  - referer allows access only from this directory
  - satisfy all or any

## Using LIMIT

- Host Filtering is used to limit document trees to certain machines
- Example 1: to limit access to the cscixxx public\_html documents to USC only

```
<Directory /usr/~cscixxx/public_html/>
<Limit GET>
order deny,allow
deny from all
allow from .usc.edu
</Limit>
</Directory>
```

- If someone tries to access documents in this directory from outside of usc they get a 403 Forbidden message

## Using LIMIT (cont'd)

- Example 2: to limit documents so USC people CANNOT access them

```
<Directory /usr/~cscixxx/public_html/>
<Limit GET>
order allow,deny
allow from all
deny from from .usc.edu
</Limit>
</Directory>
```

## Using LIMIT (cont'd)

- Example 3: a directive used to limit access to only USC and ISI domains

```
<limit GET>  
order deny, allow  
deny from all  
allow from 128.125  
allow from 128.9  
</Limit>
```

## Using LIMIT (cont'd)

- Example 4: Suppose you want to restrict files in a directory called secure/ to the user named student1 and password XXXYYYY
- Step 1. create a file called .htaccess in directory secure/ that contains these 7 lines:

```
AuthUserFile /otherdir/.htpasswd      must be a full  
UNIX pathname
```

```
AuthGroupFile /dev/null      there is no group file
```

```
AuthName ByPassword      any name is OK here
```

```
AuthType Basic      Basic or Digest is OK
```

```
<Limit GET>
```

```
require user student1
```

```
</Limit>
```

## Using LIMIT (cont'd)

- Step 2. Next create the password file /otherdir/.htpasswd using the htpasswd program distributed with apache.
- Type

```
htpasswd -c /otherdir/.htpasswd student1  
and enter the password twice.
```

- You are done.
- Step 3. To add other names and passwords run htpasswd without the -c option.

## Using LIMIT (cont'd)

- To generalize this example to include several students,
- Step 4. create a group file called /otherdir/.htgroup which looks like:

my-users: student1 student2 student3 student4

Alter the .htaccess file to look like

AuthUserFile /otherdir/.htpasswd

AuthGroupFile /otherdir/.htgroup      Now it points  
to the group file

AuthName ByPassword

AuthType Basic

<Limit GET>

require group my-users

</Limit>

## **Contents of .htaccess**

- The .htaccess file contains pointers to the users file, the group file, and defines the name and type of authorization

nunki.usc.edu(10) : more .htaccess

```
AuthUserFile /home-scf-03/cscixxx/pwd/.htpasswd
```

```
AuthGroupFile /home-scf-03/cscixxx/pwd/.htgroup
```

```
AuthName OnlineCourse
```

```
AuthType Basic
```

## **Contents of .htpasswd**

- This file contains a set of names and passwords

nunki.usc.edu(13) : more .htpasswd

admin:kjpokufbLdrXU

chungw:2Ds5QooEqT1HM

csci351:ykGZC.bMvV8SY

csci351notes:7814Vi6oJtq9c

csci565:t.OSxBunNhigg

csci571:NUKyNCCLvgLH.

nunki.usc.edu(13) : more .htgroup

admin: admin csci571 csci565

student: admin csci571 csci565

# Adding a Smart Error Page

- When someone requests a page or directory that does not exist it is better to return an error page than a 404 Not Found message
- For an example try: `http://www.usc.edu/x.html`
- Look at `/conf/httpd.conf`
- find the section that describes the properties of your domain, e.g.

```
<VirtualHost www.test.com>
ServerAdmin webmaster@test.com
DocumentRoot /web/test
ServerName www.test.com
ErrorLog /log/test/error_log
transferLog /log/test/access_log
AgentLog /log/test/agent_log
RefererLog /log/test/refer_log
</VirtualHost>
```

# Adding a Smart Error Page

- Insert the line

```
ErrorDocument 404 /error-page.html
```

- place the error page at

DocumentRoot/ErrorDocument or in this case  
/web/test/error-page.html

Sample contents of error-page.html:

```
<HTML><HEAD><TITLE>Test Error Page</TITLE></HEAD>
<BODY>Error. You have requested a page that
cannot be found. You may wish to return to the
<A HREF=http://www.test.com/>home page by
clicking here.</A>
</BODY></HTML>
```

# Apache Approaches to Server-Side Processing

- CGI - the standard way to invoke an external process, e.g. perl & perl.exe
- Fast CGI, Java Servlets - Apache passes a request to another running server which executes the request and returns the result
  - see mod\_fastcgi at <http://www.fastcgi.com>
- Interpreters embedded in the server itself. There are roughly two categories
  - Modules that answer or modify requests directly, e.g. mod\_perl (Perl)
  - Modules aimed to process commands embedded in HTML pages before serving it to the client. e.g. mod\_php5 (PHP)

# **Virtual Hosting**

- *Virtual hosting* is a way of setting up your server so that it can appear to be multiple Web sites at once, depending upon how it's accessed by clients
- ISPs do this a lot
- The Apache Web server software allows you to create as many such virtual site identities as your system's limits will allow;
- Example of a virtual host definition.

```
<VirtualHost 10.0.0.1>  
    ServerName WWW.Mydomain.Com  
    ServerAdmin Webmaster@Mydomain.Com  
    DocumentRoot /home/httpd/mydomain  
    CustomLog logs/mydomain_log CLF  
</VirtualHost>
```

## Virtual Hosting (cont'd)

- there are two ways to do virtual hosting
  - *address-based* virtual hosting uses IP addresses
  - *name-based* hosting allow a single IP address to have multiple identities, e.g.

```
NameVirtualHost 10.0.0.1          this example assigns  
<VirtualHost 10.0.0.1>          two domain names to the  
    ServerName WWW.Mydomain.Com    same IP address  
    ServerAlias WWW.X.Com WWW.Y.Org  
    ServerAdmin Webmaster@Mydomain.Com  
    DocumentRoot /home/httpd/mydomain  
    CustomLog logs/mydomain_log_CLF  
</VirtualHost>  
<VirtualHost 10.0.0.1>  
    ServerName WWW.Other.Org  
    ServerAlias WWW.Z.Net WWW.A.Org  
    ServerAdmin Webmaster@Other.Org  
    DocumentRoot /home/httpd/other  
    CustomLog logs/other_log_CLF  
</VirtualHost>
```

# Content Negotiation

- The client provides a set of preferences concerning the preferred attributes of a document, and the server tries to come up with the best match from the versions available
  - If there's only one version of a document, then that's what the server is going to send
- The set of all versions of a document that are available to the server is called the resource's list of *variants*.
- The ways in which they differ, such as by language or encoding, are called its *dimensions*.
- The Apache server can negotiate along the
  - content-type,
  - encoding, and
  - language dimensions

# Content Negotiation

- The primary use of negotiation on the Web today is to select translations of text.
  - Typically a work is prepared in a default language such as English, and the result put in a file named `text.html` or some such. Then translated versions are prepared, and each is put into an appropriate-named file: `text.html.da` for Danish, `text.html.fr` for French, `text.html.en-gb` for English (Great Britain dialect), and so on.
- You can either prepare a list of variants yourself, or you can allow Apache's `mod_negotiation` module to calculate the list itself by scanning the directory.
- The latter method is more common, since it is easier and automatically pick up new variants as soon as they are available, but it does impose a slight performance penalty due to the directory scan.

# **Apache as a Proxy Server**

- 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.
  - The users ask for a page, the browser sends the request to the proxy server, the proxy server sends the request to the origin server, and the data flows back the same way.
- A second function commonly associated with proxy servers is caching.
- To use the proxy functions in your Apache Web server, the mod\_proxy module must be activated, either by being statically linked in (less common than formerly), or turned on at run-time with directives, e.g.

```
AddModule mod_proxy.c
```

## **Apache as a Proxy Server, Proxy Directives**

- `ProxyRequests` – This is the 'master directive,' which enables and disables proxy operation overall. It takes a single keyword argument, which must be either On or Off
- `ProxyRemote` – tells the Apache server that certain requests are to be forwarded to yet another system rather than being handled locally, allowing for cascading of proxies
- `ProxyBlock` – allows you to selectively block the fetching of certain resources through the proxy. It takes a list of space-separated case-insensitive phrases or URI substrings which should be blocked.
  - "`ProxyBlock foo`" would block all requests in which "foo" appeared, in any combination of upper- and lower-case, including "`http://foo.com/bar/`" and "`http://bar.org/zardofoo.html`". You can block all proxy requests with "`ProxyBlock *`"

# Controlling the Apache Cache

- CacheRoot – Defines the directory that's the top of the cache hierarchy
  - Apache uses its own naming conventions for the stuff it stores.
- CacheSize – Specifies the largest number of kilobytes of space you want to allocate to the proxy cache.
- CacheDirLevels – Controls how deep the cache directory tree is allowed to grow
- CacheDirLength – Used to define the number of characters used in naming each level of cache subdirectory
- NoCache – Identifies proxy-accessed documents that shouldn't be cached. The format is the same as for the ProxyBlock directive.

## Proxy Host Example

- Usually a proxy server is set up as a completely separate virtual host from the rest of the server's operation (which is likely to be handling normal requests).
- You can do this with a <VirtualHost> stanza. The <IfModule> container will only be evaluated if the proxy module is part of the running server

```
<IfModule mod_proxy.c>
    Listen 10.0.0.1:8080
    <VirtualHost 10.0.0.1:8080>
        ProxyRequests On
        DocumentRoot /home/httpd/htdocs/proxy
    </VirtualHost>
</IfModule>
```

# Analyzing Web Server Log Files

- A sample entry from a log file

```
Host           date/time           method file HTTP ver  status  bytes
marmot.usc.edu - - [17/Feb/2009:23:21:09 -0800] "GET"   /    "HTTP/1.0" 200    1316
```

- There are different log file formats
  - W3C Extended Log File Format
    - Date/time, server port, URL and query, protocol version, user agent and cookie, referrer, etc.
  - Microsoft IIS log file format
    - IP, user, date/time, server name, server IP, bytes received, bytes sent, protocol, method, etc.
  - Other file formats
    - Cerfnet, apache extended, CERN common, Lotus Domino, Netscape Common, etc.

## Definition of Terms

- A *hit*: a request for a page, image, cgi script, etc.
- A *page view*: a request for an html page
- Unique host vs. unique visitor: aol.com is a unique host, so this statistic may not be useful; cookies are best way to track unique visitors
- An *impression*: is a view of an advertisement
- A *click-through* is when an advertisement is actually clicked on

## Common Log Format

- Each visit to your site produces a record of the request in log file
- log files in Apache are typically stored in the logs directory
- Some fields include:
  - Host client hostname or IP address
  - ident remote identity, often a dash
  - usr authenticated user name
  - time date/time request is received
  - req HTTP request
  - s1 server's HTTP response status code
  - c1 Content-length
- Example entries from access log

marmot.usc.edu - - [17/Feb/2008:23:21:09 -0800]  
"GET / HTTP/1.0" 200 1316

128.125.229.195 - - [23/Feb/2008:10:32:09 -0800]  
"GET / HTTP/1.1" 200 1316

## Other Entries in access Log

- ciscoserv-229-180.usc.edu - - [17/Feb/2008:23:16:51 -0800] "GET /cgi-bin/secure/admin/student.pl HTTP/1.1" 401 362
- 128.125.229.195 - admin [23/Feb/2008:10:56:21 -0800] "POST /cgi-bin/secure/admin/student.pl HTTP/1.1" 200 807
- 209.85.18.67 - - [27/Feb/2008:01:12:38 -0800] "GET /ftp/templates.zip HTTP/1.1" 200 30780
- nunki.usc.edu - - [28/Feb/2008:00:48:22 -0800] "GET / HTTP/1.0" 200 955
- nunki.usc.edu - - [28/Feb/2008:00:48:26 -0800] "GET /cgi-bin/Count.cgi?df=count.dat HTTP/1.0" 200 207

## access\_log Contents

- Here are some lines from a single user session

```
lvl-sun702.usc.edu - - [09/Feb/2008:19:15:36  
-0800] "GET /newsdigest/index.html HTTP/1.0"  
200 3630
```

```
lvl-sun702.usc.edu - - [09/Feb/2008:19:15:37  
-0800] "GET / HTTP/1.0" 200 955
```

```
lvl-sun702.usc.edu - - [09/Feb/2008:19:15:39  
-0800] "GET /newsdigest/index.html HTTP/1.0"  
200 3630
```
- A *user session* includes all hits and requests made by a user terminated by “n” minutes of inactivity

## Sample error\_log Contents

- Here are some lines from the error\_log file

- The file was not found**

[Mon Jan 29 11:47:36 2008] httpd: access to /home/scf-13/csci665/www/pleader/horohome.html failed for internet-gw.watson.ibm.com, reason: No file matching URL: /pleader/horohome.html from -

- **authorization entered is incorrect**

[Thu Sep 24 10:17:41 2008] [error] [client 128.125.50.47] user csci571 not found: /cgi-bin/secure/test-cgi

- **Perl script fails**

[Sun Sep 30 01:15:01 2008] [error] [client 71.140.65.229] Premature end of script headers: /auto/home-scf-22/csci571/WebServer/Apache\_1.3.36/cgi-bin/cgipm/test6.pl

## More Examples of Error Log File Entries

- Undefined subroutine &main::Export\_to\_Excel called at /home/scf-03/csci571/WebServer/apache\_1.2.5/cgi-bin/admin/student\_db.pl line 75.
- [Tue Dec 29 02:07:47 2008] access to /auto/home-scf-03/csci571/WebServer/apache\_1.2.5/htdocs/robots.txt failed for 193.189.227.21, reason: File does not exist
- [Tue Dec 29 20:11:22 2008] send directory lost connection to client 128.125.223.237

# **Performance and Security Considerations**

- Don't log more than needed
- Look for 401 (unauthorized) and 403 forbidden status codes
- Look at the response times
  - Which parts of the web site have the longest response times?
  - Where are the bottlenecks
- Look for failed page hits (404, 5xx)
- Look for cached hits
- Look for browsers and platforms of clients

# Tools to Analyze Log Files

- Some commercially available tools for IIS
  - WebTrends  
<http://www.webtrends.com>  
See also  
[http://cs-server.usc.edu:45678/resources/web\\_servers/webtrends\\_report2.html](http://cs-server.usc.edu:45678/resources/web_servers/webtrends_report2.html)
  - Marketware, webmanager, MS site server, LiveStats
- For any web server
  - Google Analytics. See:  
<http://www.google.com/analytics/>

# Google Analytics

