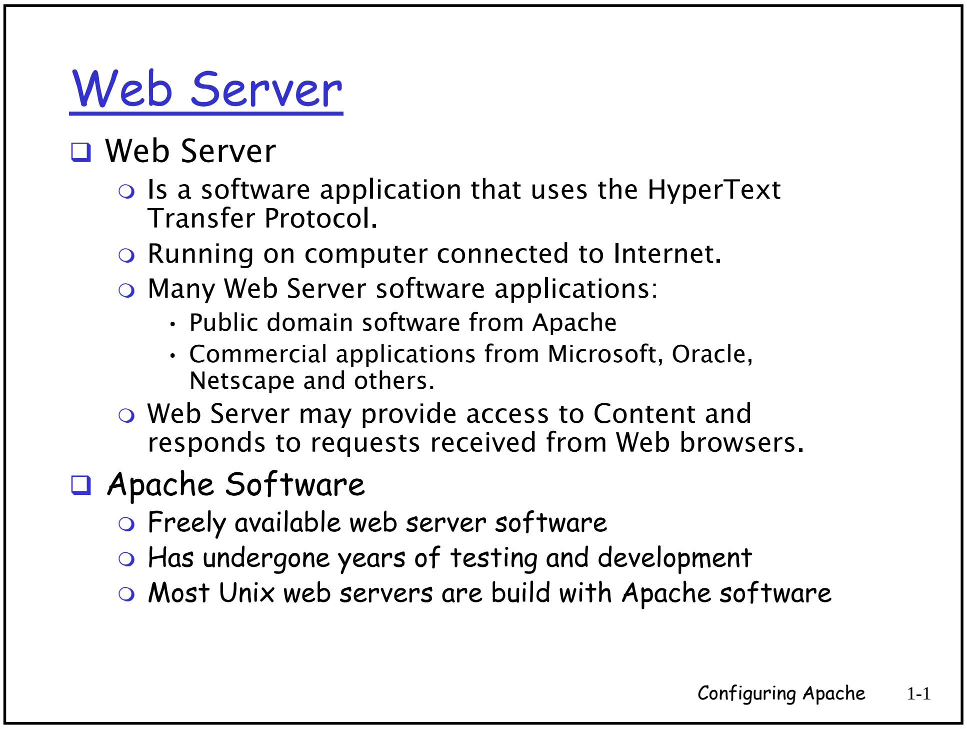
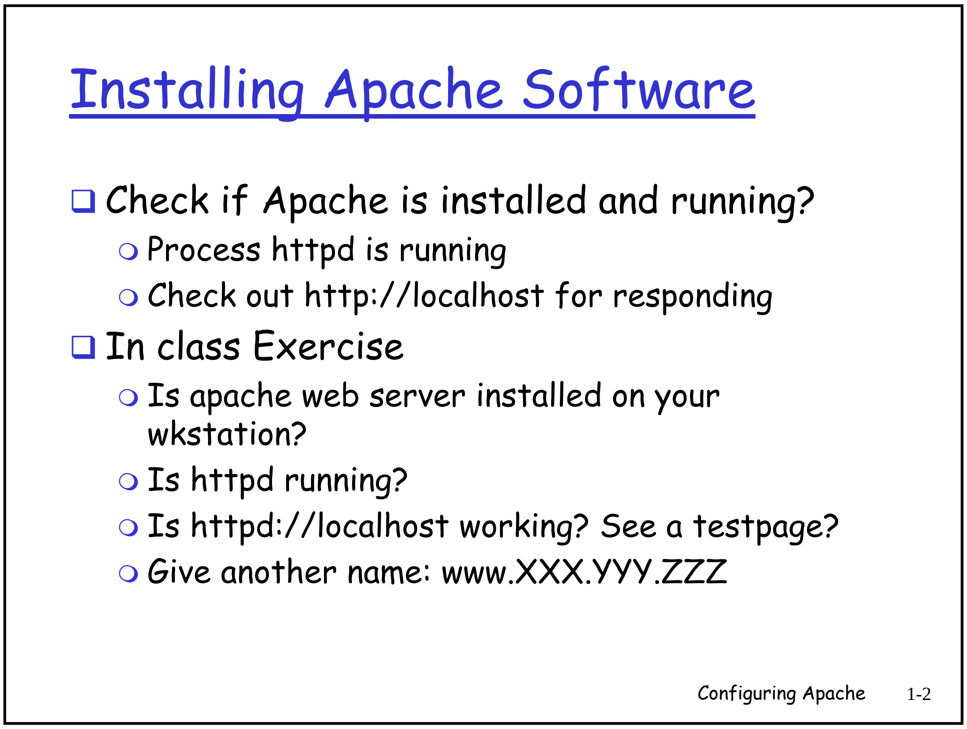


Web Server

- Web Server
  - Is a software application that uses the HyperText Transfer Protocol.
  - Running on computer connected to Internet.
  - Many Web Server software applications:
    - Public domain software from Apache
    - Commercial applications from Microsoft, Oracle, Netscape and others.
  - Web Server may provide access to Content and responds to requests received from Web browsers.
- Apache Software
  - Freely available web server software
  - Has undergone years of testing and development
  - Most Unix web servers are build with Apache software



Installing Apache Software

- Check if Apache is installed and running?
  - Process httpd is running
  - Check out http://localhost for responding
- In class Exercise
  - Is apache web server installed on your wkstation?
  - Is httpd running?
  - Is httpd://localhost working? See a testpage?
  - Give another name: www.XXX.YYY.ZZZ

Install the Package on Linux

- ❑ Use rpm
  - ❍ Get the distribution CD and find the rpm file for apache.
  - ❍ Use rpm with install option and rpm file name
  - ❍ Enable httpd to start automatically during startup
    - Use chkconfig or other to make links to boot process

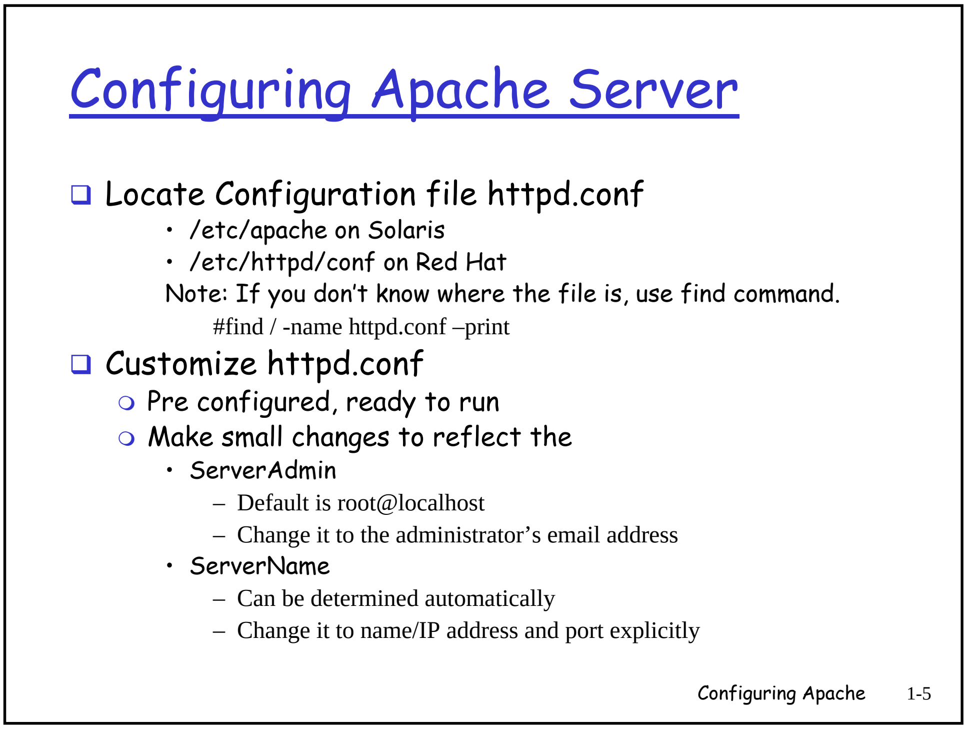
```
#chkconfig –list httpd
#chkconfig –level 35 httpd on
#chkconfig –list httpd
```

  - ❍ Reboot or manually start the daemon httpd

```
#/etc/init.d/httpd start
```

Installing Apache

- ❍ If OS does not include Apache, download it first
  - http:/www.apache.org
  - Binaries are listed by operating system, select one that is appropriate to your OS
  - Download the tar and extract it

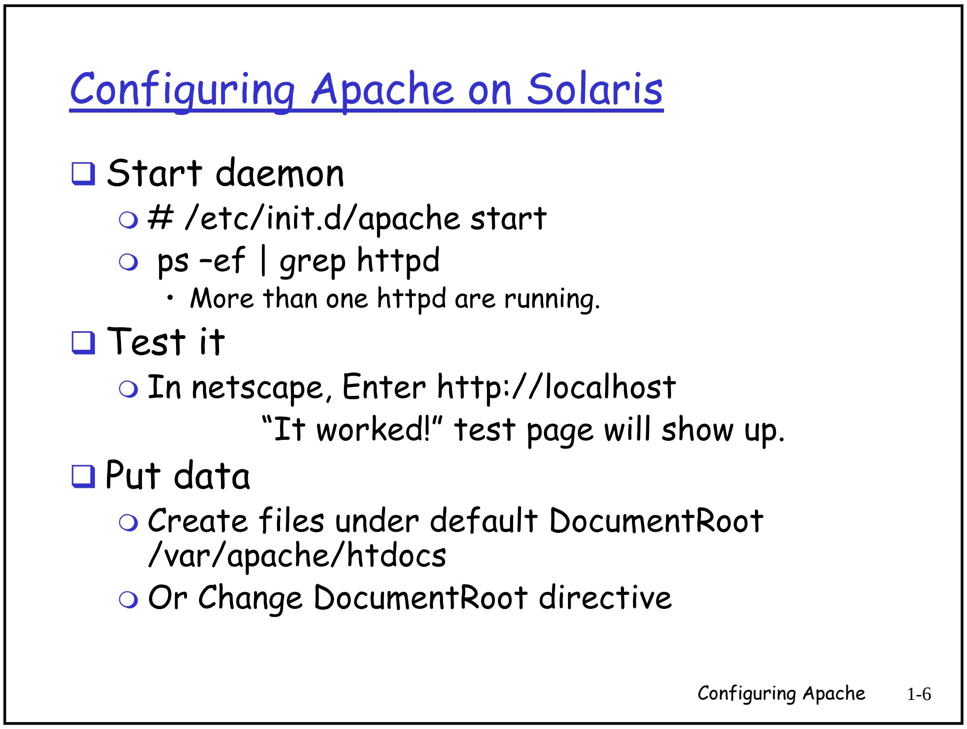


Configuring Apache Server

- ❑ Locate Configuration file httpd.conf
  - /etc/apache on Solaris
  - /etc/httpd/conf on Red Hat

  Note: If you don't know where the file is, use find command.

  ```
  #find / -name httpd.conf –print
  ```
- ❑ Customize httpd.conf
  - ❍ Pre configured, ready to run
  - ❍ Make small changes to reflect the
    - ServerAdmin
      - Default is root@localhost
      - Change it to the administrator's email address
    - ServerName
      - Can be determined automatically
      - Change it to name/IP address and port explicitly



Configuring Apache on Solaris

- ❑ Start daemon
  - ❍ # /etc/init.d/apache start
  - ❍ ps –ef | grep httpd
    - More than one httpd are running.
- ❑ Test it
  - ❍ In netscape, Enter http://localhost

    "It worked!" test page will show up.
- ❑ Put data
  - ❍ Create files under default DocumentRoot /var/apache/htdocs
  - ❍ Or Change DocumentRoot directive

Understanding an httpd.conf File

- We will focus on directives
  - Global environment directives
  - Main server directives
  - Virtual host directives
- Modules must be loaded before the directives they provide can be used in the configuration.
- Loading Dynamic Shared Objects (DSO)
  - Use LoadModule directive
    - List modules compiles into Apache

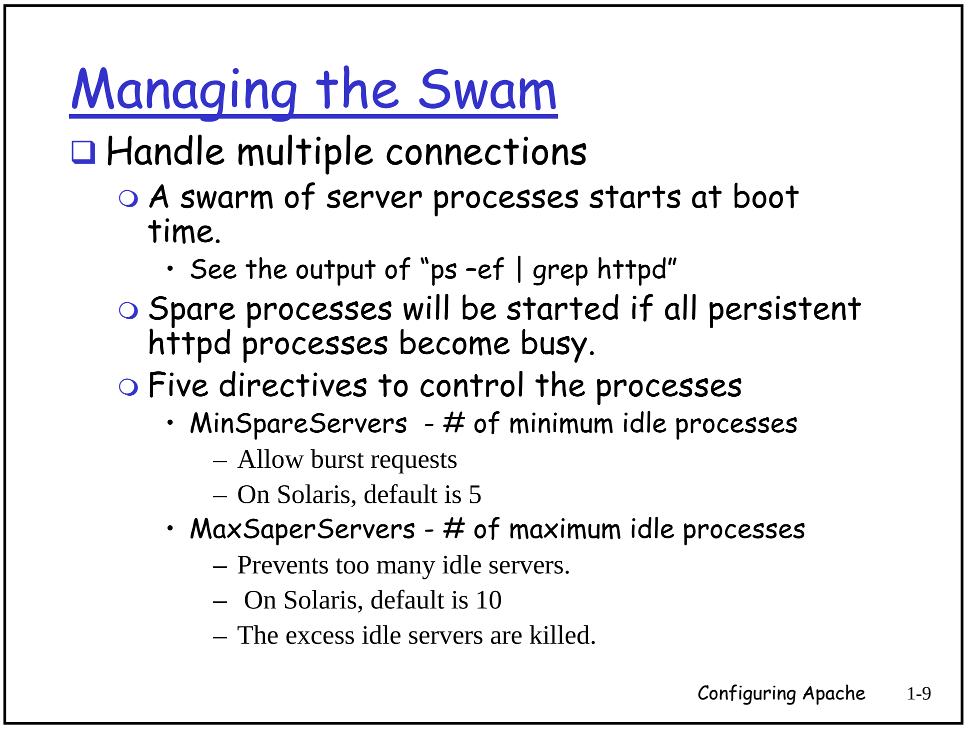
      %httpd –l

      Only http_core.c and mod_so.c have to be compiled into the executables, other modules can be loaded dynamically.
    - LoadModule – specify the .so file
    - AddModule – specify the source file
    - Order is critical

Understanding an httpd.conf File

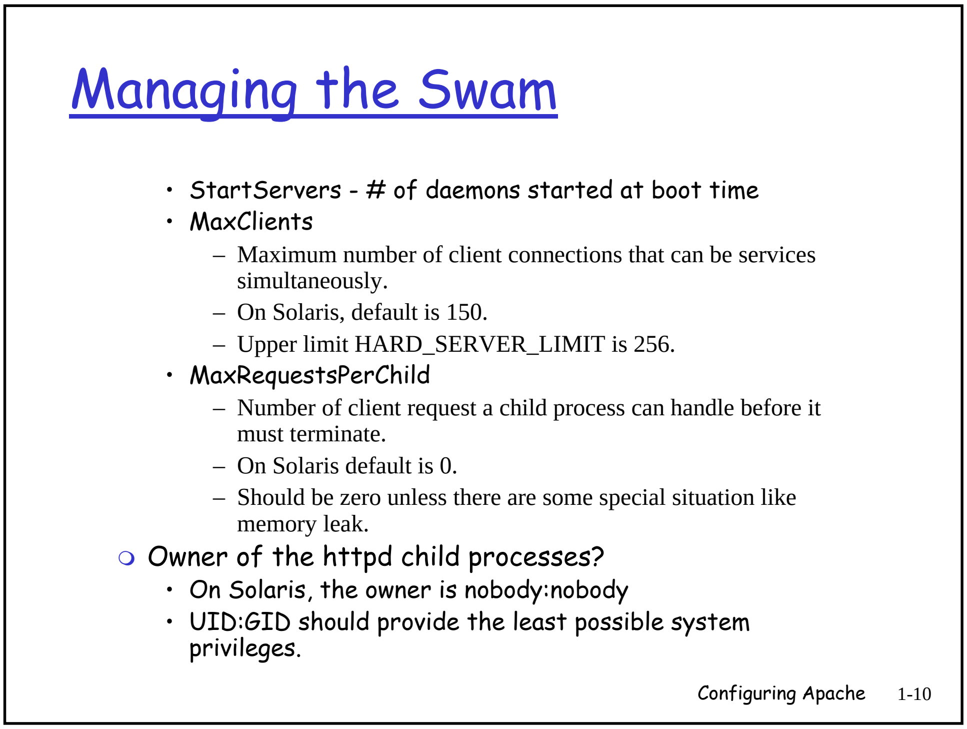
- Basic Configuration Directives
  - Email address of the web server administrator
    - ServerAdmin
  - Hostname returned to clients
    - ServerName
    - Avoid using real server name. Define CNAME in DNS

      For example: www.csl.mtu.edu
  - UseCanonicalName
    - Used to build self-referencing URL
    - If on, ServerName will be used
    - If off, the value that came in the query from client is used.
  - ServerRoot
    - Important files used by httpd.
  - Port
    - Default is 80



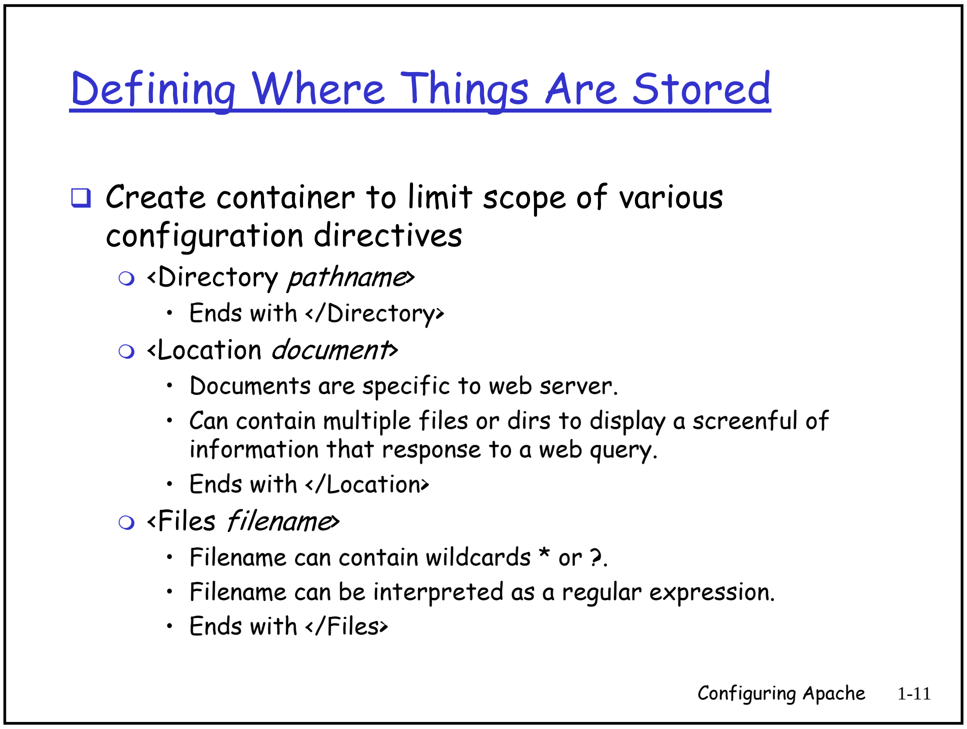
Managing the Swam

- Handle multiple connections
  - A swarm of server processes starts at boot time.
    - See the output of “ps –ef | grep httpd”
  - Spare processes will be started if all persistent httpd processes become busy.
  - Five directives to control the processes
    - MinSpareServers - # of minimum idle processes
      - Allow burst requests
      - On Solaris, default is 5
    - MaxSaperServers - # of maximum idle processes
      - Prevents too many idle servers.
      - On Solaris, default is 10
      - The excess idle servers are killed.



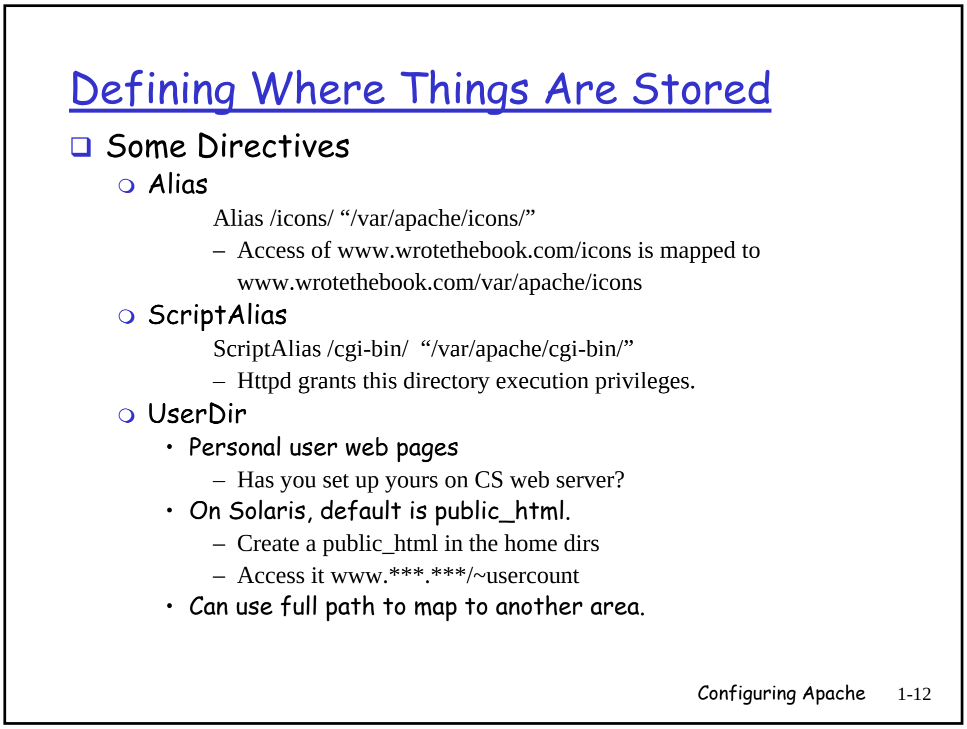
Managing the Swam

- StartServers - # of daemons started at boot time
- MaxClients
  - Maximum number of client connections that can be services simultaneously.
  - On Solaris, default is 150.
  - Upper limit HARD_SERVER_LIMIT is 256.
- MaxRequestsPerChild
  - Number of client request a child process can handle before it must terminate.
  - On Solaris default is 0.
  - Should be zero unless there are some special situation like memory leak.
- Owner of the httpd child processes?
  - On Solaris, the owner is nobody:nobody
  - UID:GID should provide the least possible system privileges.



Defining Where Things Are Stored

- Create container to limit scope of various configuration directives
  - <Directory *pathname*>
    - Ends with </Directory>
  - <Location *document*>
    - Documents are specific to web server.
    - Can contain multiple files or dirs to display a screenful of information that response to a web query.
    - Ends with </Location>
  - <Files *filename*>
    - Filename can contain wildcards * or ?.
    - Filename can be interpreted as a regular expression.
    - Ends with </Files>

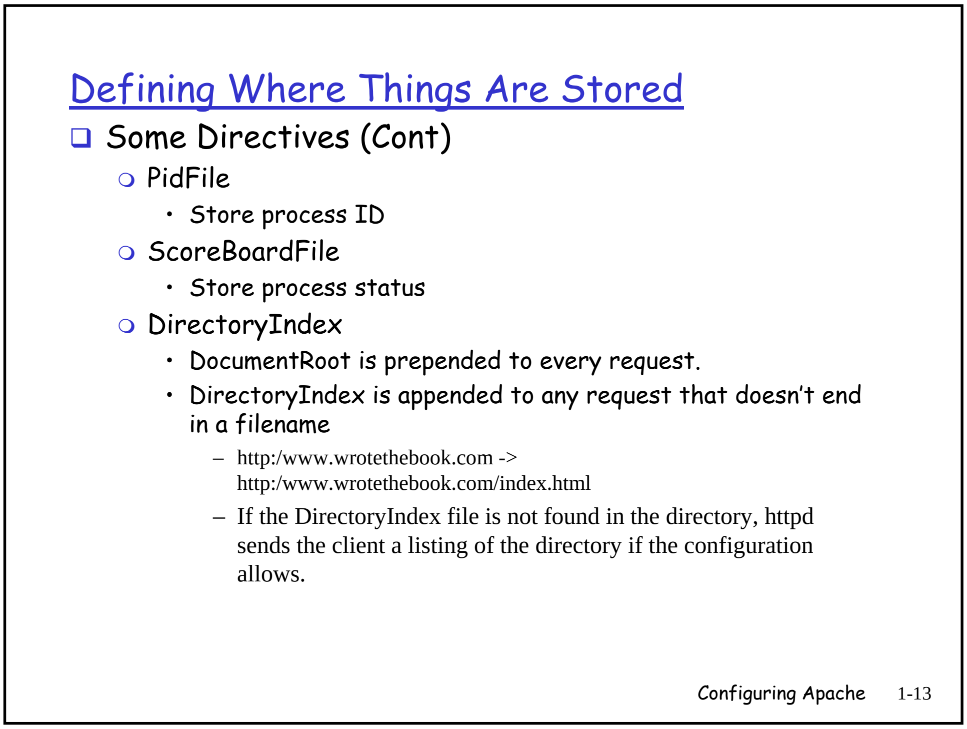


Defining Where Things Are Stored

- Some Directives
  - Alias

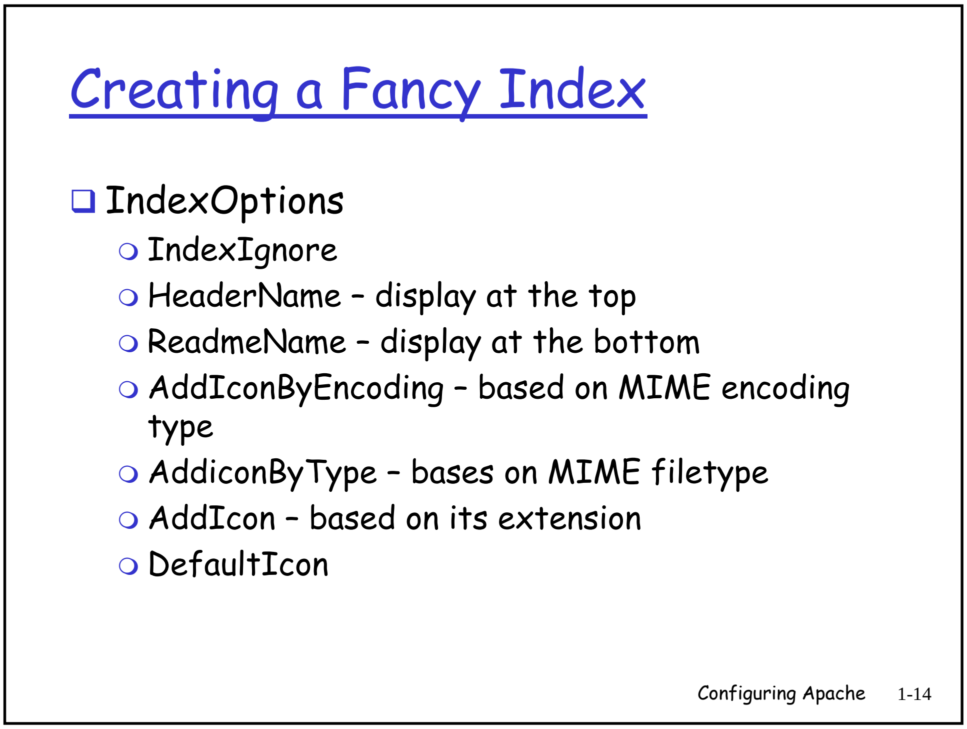
    Alias /icons/ “/var/apache/icons/”
    - Access of www.wrotethebook.com/icons is mapped to www.wrotethebook.com/var/apache/icons
  - ScriptAlias

    ScriptAlias /cgi-bin/ “/var/apache/cgi-bin/”
    - Httpd grants this directory execution privileges.
  - UserDir
    - Personal user web pages
      - Has you set up yours on CS web server?
    - On Solaris, default is public_html.
      - Create a public_html in the home dirs
      - Access it www.***.***/~usercount
    - Can use full path to map to another area.



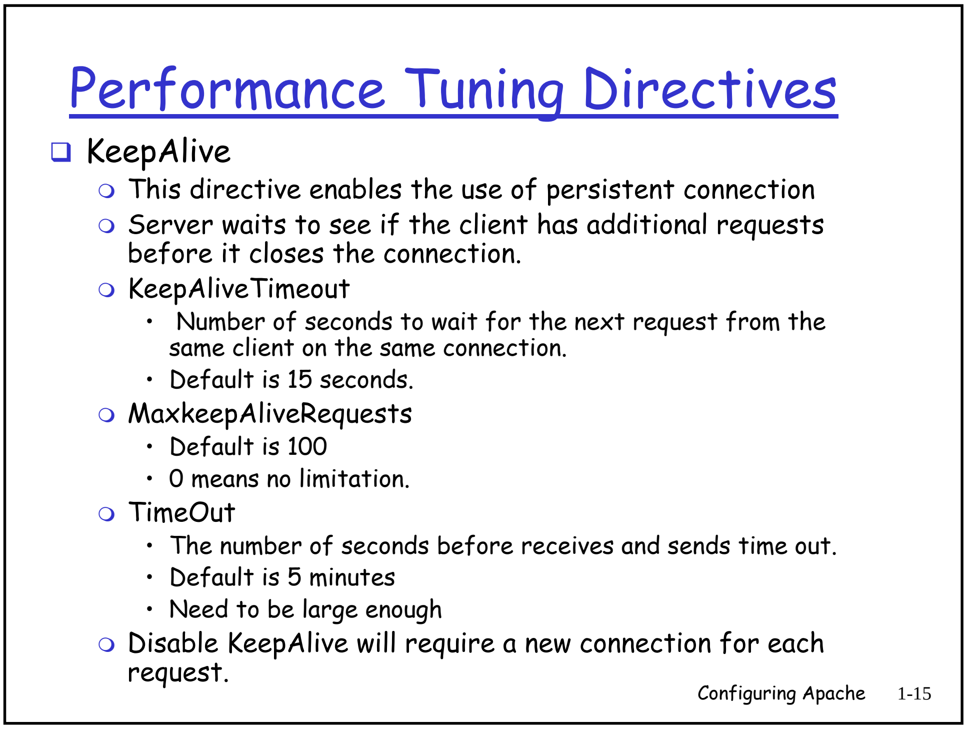
Defining Where Things Are Stored

- Some Directives (Cont)
  - PidFile
    - Store process ID
  - ScoreBoardFile
    - Store process status
  - DirectoryIndex
    - DocumentRoot is prepended to every request.
    - DirectoryIndex is appended to any request that doesn't end in a filename
      - http:/www.wrotethebook.com -> http:/www.wrotethebook.com/index.html
      - If the DirectoryIndex file is not found in the directory, httpd sends the client a listing of the directory if the configuration allows.



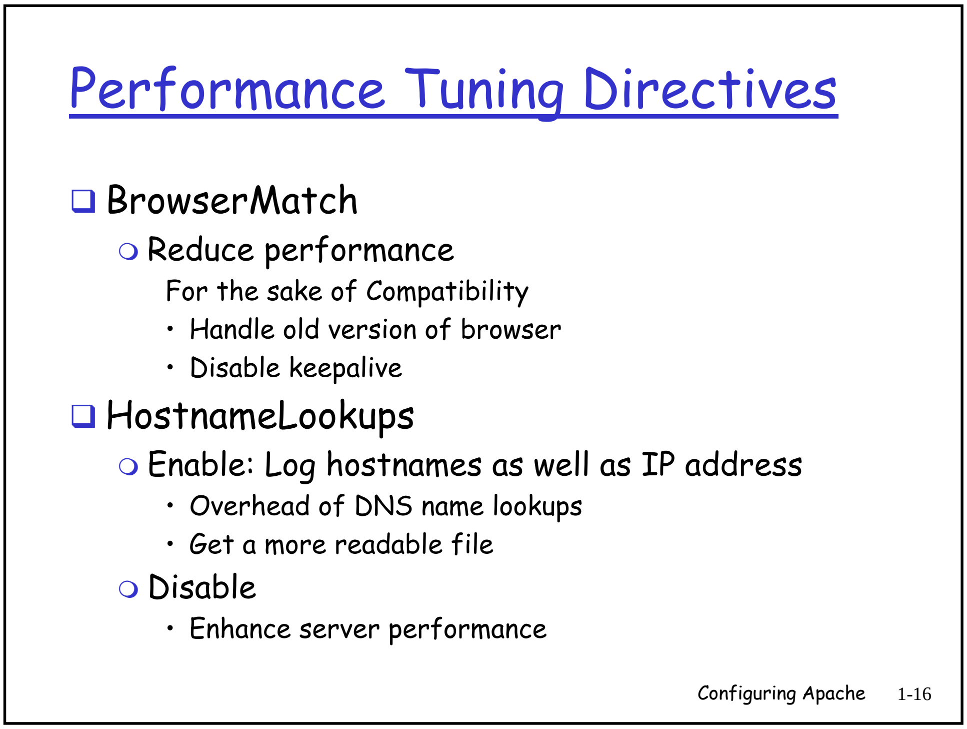
Creating a Fancy Index

- IndexOptions
  - IndexIgnore
  - HeaderName – display at the top
  - ReadmeName – display at the bottom
  - AddIconByEncoding – based on MIME encoding type
  - AddiconByType – bases on MIME filetype
  - AddIcon – based on its extension
  - DefaultIcon



Performance Tuning Directives

- KeepAlive
  - This directive enables the use of persistent connection
  - Server waits to see if the client has additional requests before it closes the connection.
  - KeepAliveTimeout
    - Number of seconds to wait for the next request from the same client on the same connection.
    - Default is 15 seconds.
  - MaxkeepAliveRequests
    - Default is 100
    - 0 means no limitation.
  - TimeOut
    - The number of seconds before receives and sends time out.
    - Default is 5 minutes
    - Need to be large enough
  - Disable KeepAlive will require a new connection for each request.



Performance Tuning Directives

- BrowserMatch
  - Reduce performance

    For the sake of Compatibility
    - Handle old version of browser
    - Disable keepalive
- HostnameLookups
  - Enable: Log hostnames as well as IP address
    - Overhead of DNS name lookups
    - Get a more readable file
  - Disable
    - Enhance server performance

Logging Configuration Directives

- Logging configuration
  - ErrorLog
    - Define the path of the error log
    - Monitor the log regularly or on real time
      - » “tail –f filename”
  - LogLvel
    - Eight levels:
      - – debug, info, notice, warn, error, crit, alert, emerg
      - – Default level is warn – right amount of info in most cases

Logging Configuration Directives

- LogFormat
  - Define the format of log file entries
    - – Conforms to the Common Log Format (CFL) standard, so log analyzer can be used
    - – Includes a layout and a label
      - » LogFormat “%h %u %t \”%r\” %>s %b” common
      - » LogFormat “{User-agent}” agent
- CustomLog
  - Bind the label of LogFormat with a file
    - – CustomLog /var/apache/logs/access_log combined
    - – CustomLog /var/apache/logs/agent_log agent

Logging Configuration Directives

- Using conditional logging
  - Log when certain status codes are returned
    - Log browser name only if the browser requests a service that is not implemented in your server
      - Status code 501:Not implementation
      - %501{user-agent}I
      - %!200,302,304{Referer}i

Proxy Servers and Caching

- What are proxy servers?
  - Servers that sits between a client application, such as a Web browser, and a real server.
  - Two main purposes:
    - Improve Performance
      - It saves the results of all requests for a certain amount of time and sends back the result to client from its cache if not expired or updated since then.
        - » Reduce network traffic
        - » Reduce user's wait time, improve response time
        - » Relieve network bandwidth bottleneck
      - Major online services such as Compuserve and America Online employ array of proxy servers.
    - Filter Requests
      - Prevent to access certain set of web sites.
      - Increate network security

Options that control Caching

- CacheNegotiateDocs
  - Allow caching
- ProxyRequests
  - Turn your server into a proxy server
- ProxyVia
  - Enables or disable the use of Via: headers, which aid in tracking where cached paged came from
- CacheRoot
  - The directory where cached web pages are written
- CacheSize
  - Maximum size of the cache in kilobytes
- CacheGcInterval
  - Time interval at which the server prunes the cache.
- NoCache
  - Defines a list of servers whose pages you do not want to cache.

Multi-homed Server Options

- Web servers with more than one IP address are said to be multi-homed
- Which address it should listen to for incoming server requests?
  - Listen
    - Specifies addresses and port in addition to the default port and address.
      - listen 192.168.1.1:80
      - Listen 216.180.25.168:443

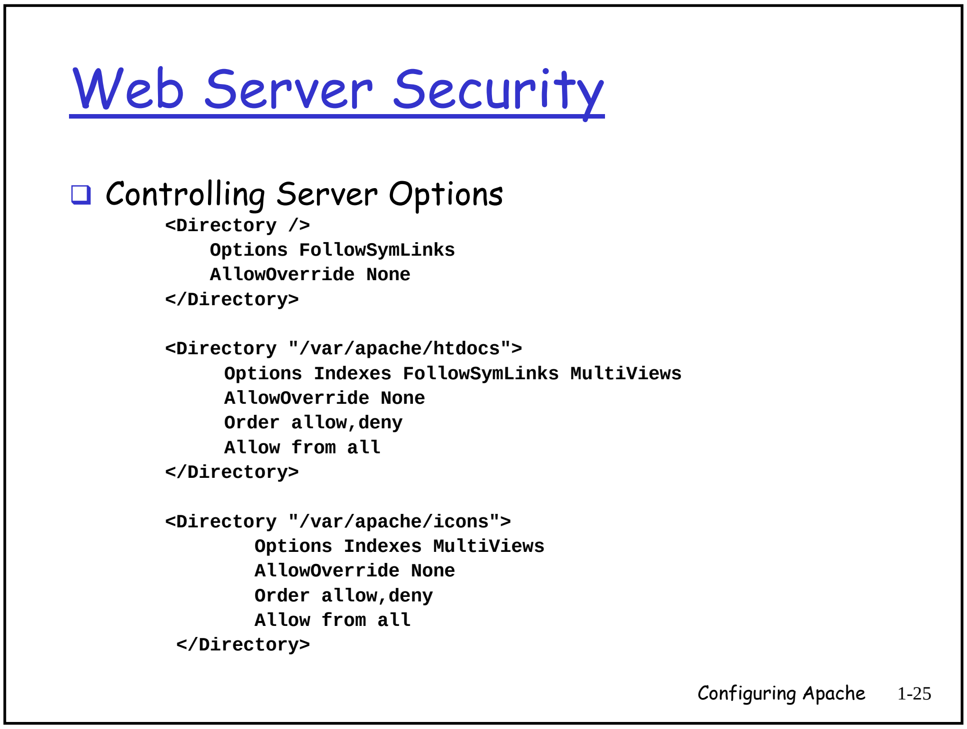
Defining Virtual Hosts

- Use Virtual Host directives to hosts multiple web sites.
  - For example, on crab.wrotethebook.com, host two web site
    - fish.edu and mammal.com

```
<VirtualHost www.fish.edu>
DocumentRoot /var/apache/fish
Servername www.fish.edu
</VirtualHost>
<VirtualHost www.mammal.edu>
DocumentRoot /var/apache/mammals
Servername www.mammals.edu
</VirtualHost>
```

Web Server Security

- Protect the integrity of the information
  - Access controls defined in httpd.conf
    - Host level
    - User level
  - Unix file permissions
    - Read permission for all files
    - Execution permission for some files
      - CGI and SSI
      - potential security threat.
        - Buffer overflow
        - Passing shell commands
  - Control executable files
    - One Location
    - Review the CGI files
    - Limit the use of some SSI commands



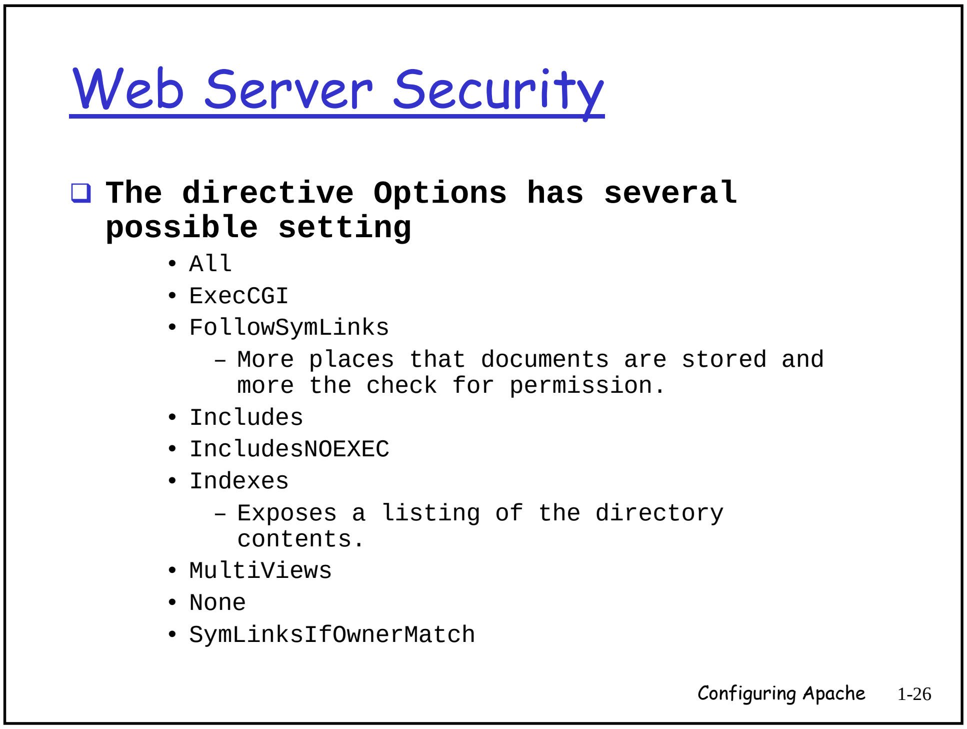
Web Server Security

- Controlling Server Options

```
<Directory />
    Options FollowSymLinks
    AllowOverride None
</Directory>

<Directory "/var/apache/htdocs">
     Options Indexes FollowSymLinks MultiViews
     AllowOverride None
     Order allow,deny
     Allow from all
</Directory>

<Directory "/var/apache/icons">
        Options Indexes MultiViews
        AllowOverride None
        Order allow,deny
        Allow from all
 </Directory>
```



Web Server Security

- **The directive Options has several possible setting**
  - All
  - ExecCGI
  - FollowSymLinks
    - More places that documents are stored and more the check for permission.
  - Includes
  - IncludesNOEXEC
  - Indexes
    - Exposes a listing of the directory contents.
  - MultiViews
  - None
  - SymLinksIfOwnerMatch

Web Server Security

- ❑ Directory-level Configuration Controls
  - ❍ Access control information for each directory
  - ❍ Enable it:
    - AccessFileName .htaccess
  - ❍ Limit the control given to individual's directories
    - AllowOverride
      - – ALL
      - – NONE
      - – Individual directive

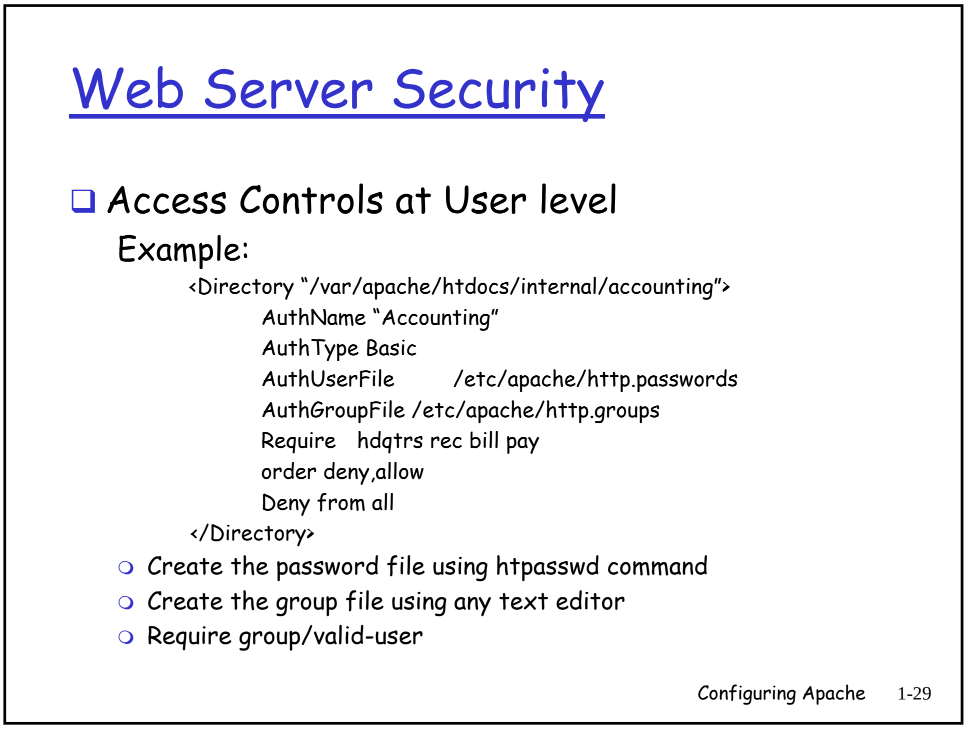
Web Server Security

- ❑ Defining Access Controls from host level

  Example:

```
<Directory "/var/apache/htdocs/internal">
        order deny,allow
        Deny from all
        Allow from wrotethebook.com
</Directory>
```

  - ❍ Three access control directives
    - Order
    - Deny from
    - Allow from



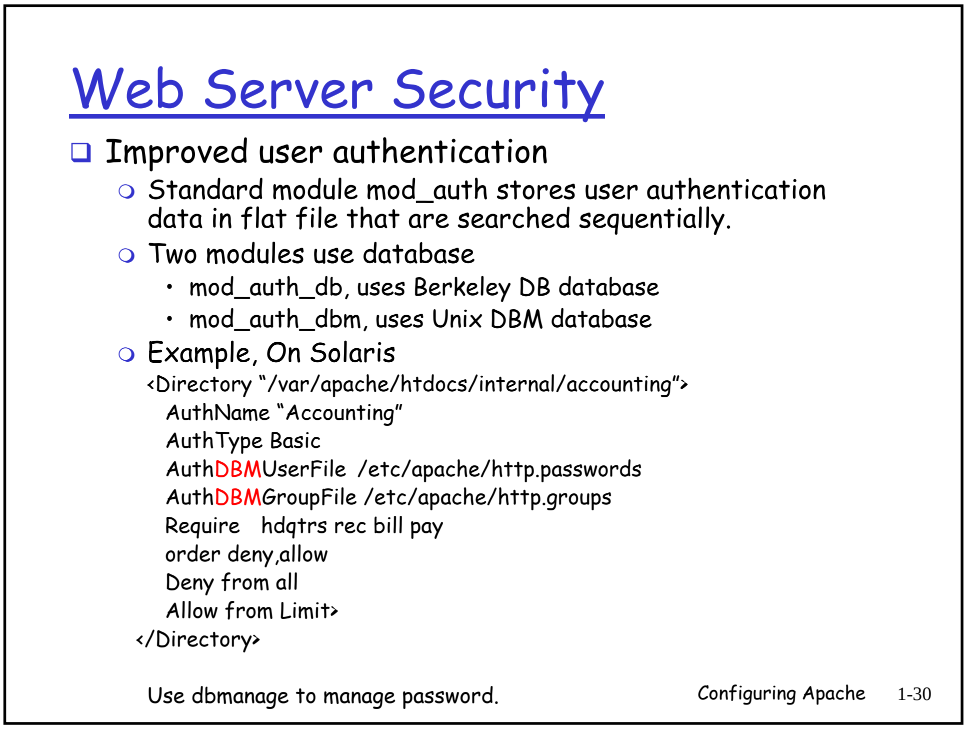
Web Server Security

- Access Controls at User level

  Example:

```
<Directory "/var/apache/htdocs/internal/accounting">
        AuthName "Accounting"
        AuthType Basic
        AuthUserFile        /etc/apache/http.passwords
        AuthGroupFile /etc/apache/http.groups
        Require   hdqtrs rec bill pay
        order deny,allow
        Deny from all
</Directory>
```

  - Create the password file using htpasswd command
  - Create the group file using any text editor
  - Require group/valid-user

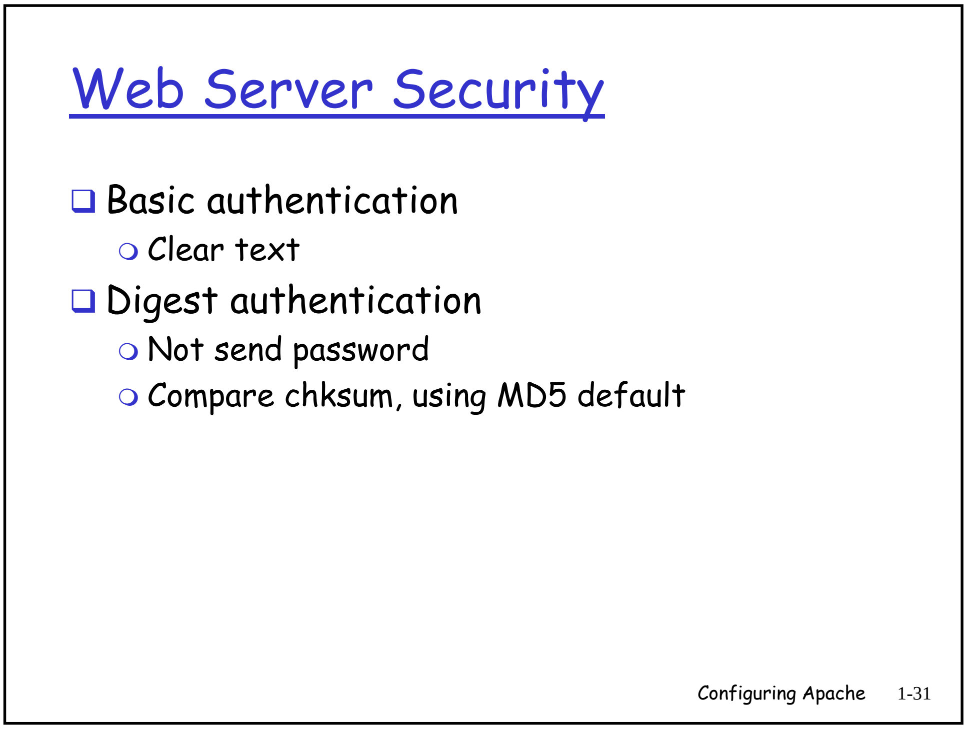


Web Server Security

- Improved user authentication
  - Standard module mod_auth stores user authentication data in flat file that are searched sequentially.
  - Two modules use database
    - mod_auth_db, uses Berkeley DB database
    - mod_auth_dbm, uses Unix DBM database
  - Example, On Solaris

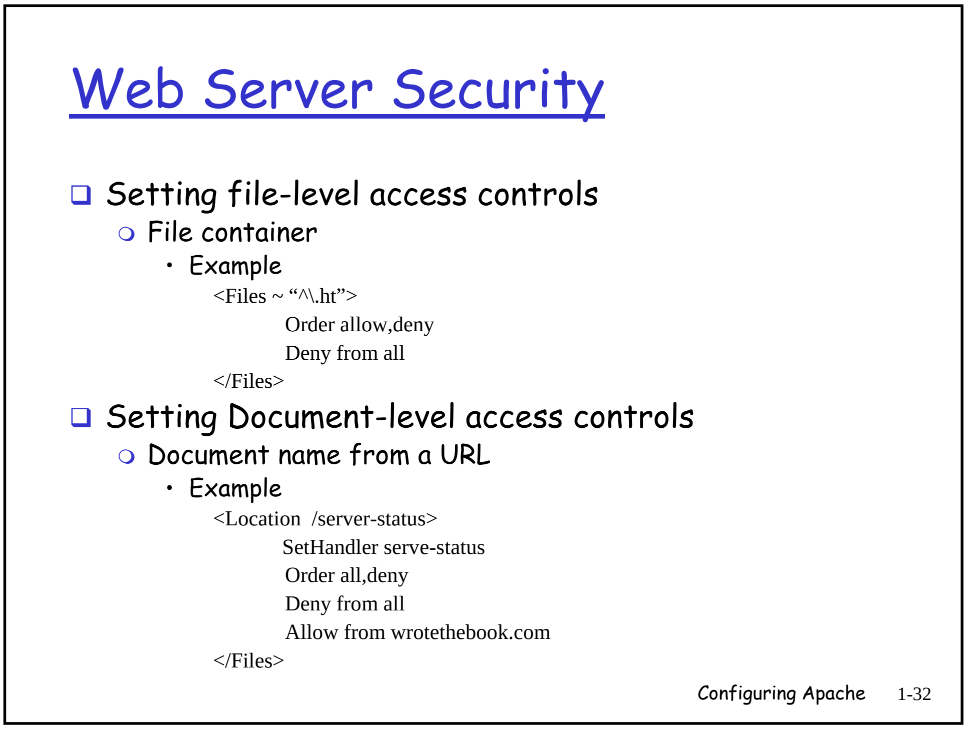
```
<Directory "/var/apache/htdocs/internal/accounting">
  AuthName "Accounting"
  AuthType Basic
  AuthDBMUserFile  /etc/apache/http.passwords
  AuthDBMGroupFile /etc/apache/http.groups
  Require   hdqtrs rec bill pay
  order deny,allow
  Deny from all
  Allow from Limit>
</Directory>
```

Use dbmanage to manage password.



Web Server Security

- Basic authentication
  - Clear text
- Digest authentication
  - Not send password
  - Compare chksum, using MD5 default



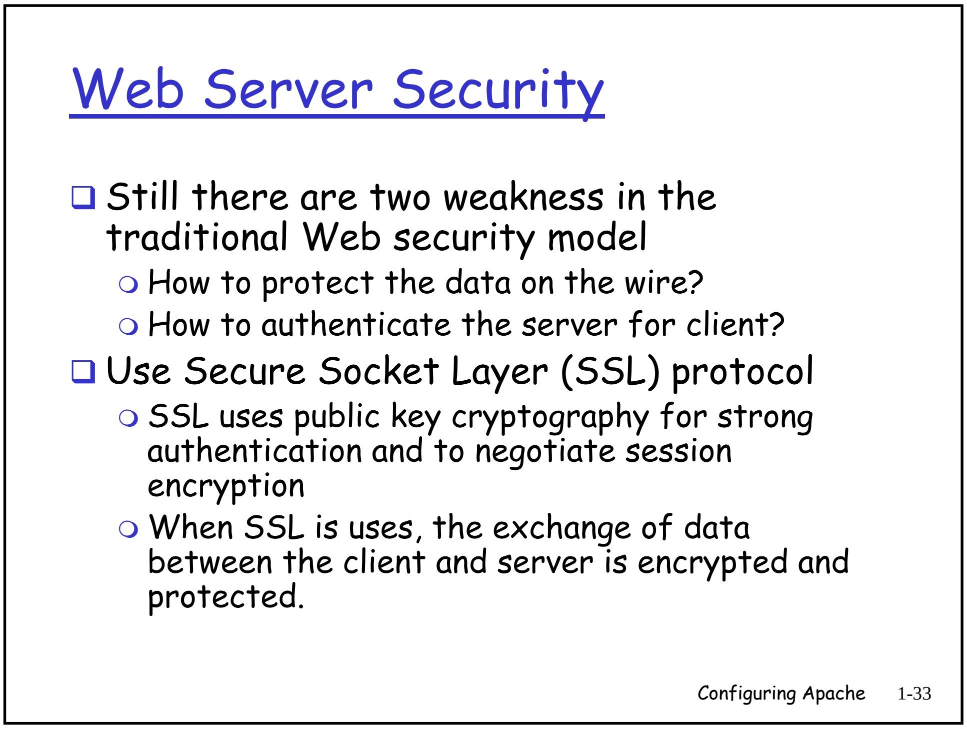
Web Server Security

- Setting file-level access controls
  - File container
    - Example

```
<Files ~ “^\.ht”>
        Order allow,deny
        Deny from all
</Files>
```

- Setting Document-level access controls
  - Document name from a URL
    - Example

```
<Location  /server-status>
        SetHandler serve-status
        Order all,deny
        Deny from all
        Allow from wrotethebook.com
</Files>
```



Web Server Security

- Still there are two weakness in the traditional Web security model
  - How to protect the data on the wire?
  - How to authenticate the server for client?
- Use Secure Socket Layer (SSL) protocol
  - SSL uses public key cryptography for strong authentication and to negotiate session encryption
  - When SSL is uses, the exchange of data between the client and server is encrypted and protected.