

IT452 Advanced Web and Internet Systems

Set 6: Web Servers
(operation, configuration, and security)
(Chapters 16 and 18)

Key Questions

- Popular web servers?
- What does a web server do?
- How can I control it?
 - URL re-writing / re-direction (and why do I care?)
 - Access control / security

Web Server Basics

- <http://www.example.com/products/widget.html>
- What happens? Where does it come from?

- Are we sure?

Web Server Basics

- <http://www.example.com/cgi-bin/search.pl?q=widgets>
- What happens? Where does it come from?

- Are we sure?

- What's not so good about this?

URLs

- Things to avoid
- Things to do
- How to do this?

Aside: How to control a web server?

- Apache – two primary config locations:
 - httpd.conf
 - Whole site
 - Must be root
 - Requires restart
 - .htaccess
 - Per directory
 - Possibly each user (depends on config)
 - Re-read for each request

Content Control

1. Redirection
2. Rewriting
3. Content negotiation

Redirection / rewriting examples

```
# NOTE: this file (.htaccess) is in the 'change' directory

# Using mod_alias
# Redirect file somewhere else (target MUST be absolute URL)
Redirect permanent /change/oldfile.txt http://newplace734.com/test1.txt

# Redirect whole directory
Redirect permanent /change/olddir http://newplace734.com/newdir

# Using mod_rewrite - first must turn on
RewriteEngine On

# Similar redirect -- now to same server
# /change/oldfile3.txt --> /change/test3.txt
RewriteRule ^oldfile3.txt$ /change/test3.txt [R,L]

# Behind the scenes change
RewriteRule ^oldfile4.txt$ /change/test4.txt [L]

# More complex
# rewrite change/stuff/dogs to change/query.pl?q=dogs
# 302 = temp change
RewriteRule ^stuff/([^/]+)/?$ /change/query.pl?q=$1 [R=302,L]
```

Apache Access Control – Options

1. Domain/IP restrictions
2. Password protection: “Basic”
 1. Much relegated to browser – can’t control
 2. Passed in plain text! (okay if using SSL)
 3. Password passed every time!
 4. Okay if using SSL
3. Password protection: “Digest”
 1. Sends “digest” rather than plain password
 2. But hacker could re-use digeest!
4. More advanced modules – keep passwords in DB rather than “flat file”
5. Alternative?

1. Access control: IP-based

```
<LIMIT GET>
order deny,allow
deny from all
allow from .nadm.navy.mil
allow from .usna.navy.mil
allow from .usna.edu
allow from .naps.edu           # Naval Academy Prep School
allow from 192.190.228.       # test bench
allow from 192.190.229.       # test bench
allow from 192.31.8           # test bench
allow from 207.86.40.42       # NAPS
allow from 131.158.248.       # Navy Medical
allow from 131.158.247.       # Navy Medical
allow from 137.225.250.       # Joint Spectrum Command
allow from 12.110.116.250     # Alumni Association
allow from 128.56.
allow from 131.121.
allow from 131.122.
</LIMIT>
```

2. Access Control: “Basic”

- Whole directory

```
AuthType Basic
AuthUserFile c:/wamp/.htpasswd
AuthName "Members Only"
require valid-user
```

- Per file

```
<Files somefile.html>
AuthType Basic
AuthUserFile c:/wamp/.htpasswd
AuthName "Members Only"
require valid-user
</Files>
```

3. Access Control: “Digest”

- Whole directory

```
AuthType Digest
AuthName "myrealm"
AuthUserFile c:/wamp/.htpasswddigest
Require valid-user
```

- Per file

- Use <Files>

- Specific user (also applies to “Basic”)

- Require user lmcdowel schulze

- Groups of users

- See documentation

Making the password file

```
htpasswd -c /usr/local/apache/passwd/digest username
```

```
htdigest -c /usr/local/apache/passwd/digest realm username
```

Notes:

-c makes new file – omit to just add new entry (or update)

Substitute in actual path to the file

Don't store password file in the web space!

AND/OR conditions

From <http://httpd.apache.org/docs/1.3/howto/auth.html>

AuthType Basic

AuthName intranet

AuthUserFile /www/passwd/users

AuthGroupFile /www/passwd/groups

Require group customers

Order allow,deny

Allow from internal.com

Satisfy any