Web server and Apache

Francesco Zampognaro

Marco Bonola

History

12/3/2019

- WWW is recent in definition with regard to other protocols and TCP/IP
- First introduced in 1989; first working server in 1990.
- First web page ever back online at its original URL: http://info.cern.ch/hypertext/WWW/TheProject.html
- This page was retrieved from a later 1992 backup, the very original page was lost:
  - 48 copies of the 600 year-old Gutenberg bible exist, yet not one copy of the original first website made just thirty years ago is available.

Declaration

The following CERN software is hereby put into the public domain:

- W 3 basic ("line-mode") client
- W 3 basic server
- W 3 library of common code.

CERN's intention in this is to further compatibility, common practices, and standards in networking and computer supported collaboration. This does not constitute a precedent to be applied to any other CERN copyright software.

CERN relinquishes all intellectual property rights to this code, both source and binary form and permission is granted for anyone to use, duplicate, modify and redistribute it.

CERN provides absolutely NO WARRANTY OF ANY KIND with respect to this software. The entire risk as to the quality and performance of this software is with the user. IN NO EVENT WILL CERN BE LIABLE TO ANYONE FOR ANY DAMAGES ARISING OUT THE USE OF THIS SOFTWARE, INCLUDING, WITHOUT LIMITATION, DAMAGES RESULTING FROM LOST DATA OR LOST PROFITS, OR FOR ANY SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES.

Geneva, 30 April 1993

W. Hoogland
Director of Research

H. Weber
Director of Administration

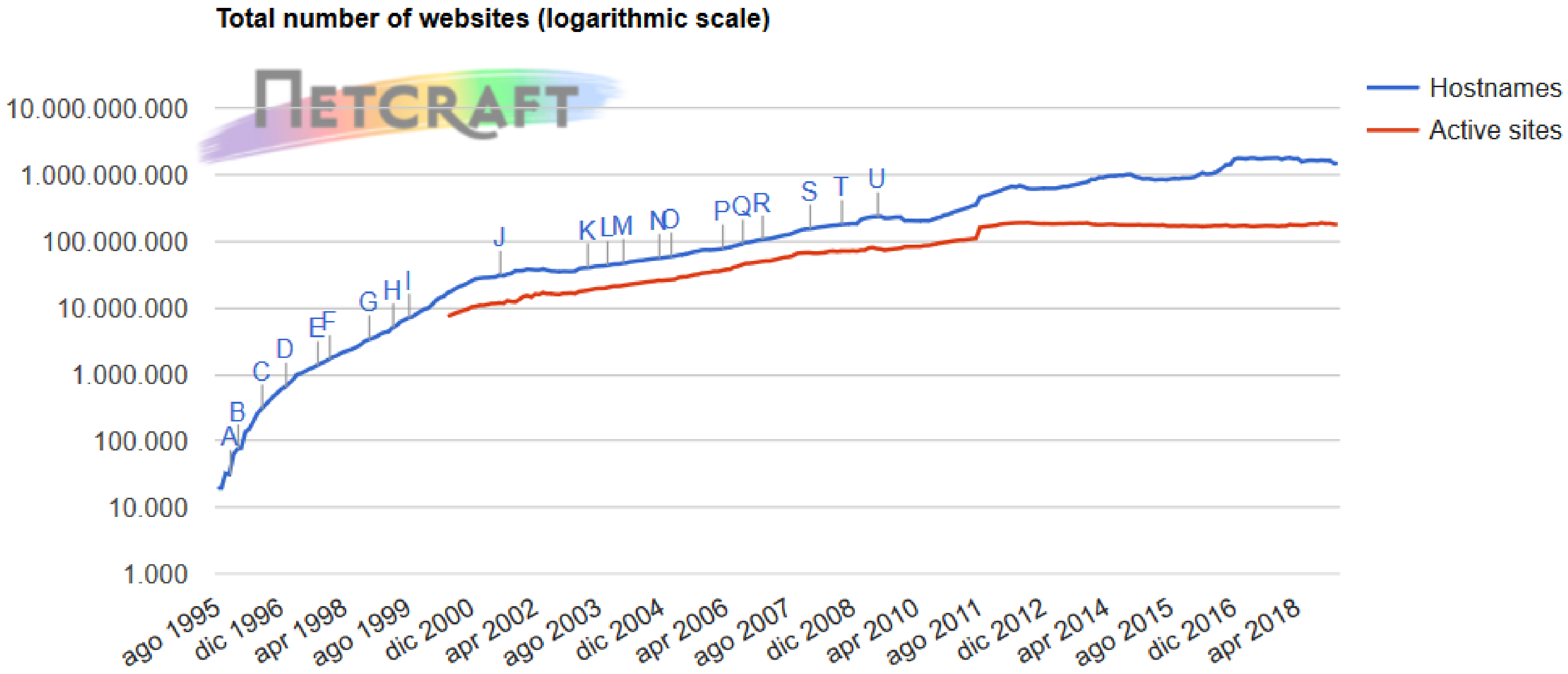
opie certifiée conforme

ait à Genève le 03-05-93

ORGANISATION EUROPÉENNE POUR LA RECHERCHE NUCLÉAIRE
DIVISION
DES
FINANCES
- CERN -

Number of Websites

Src: https://news.netcraft.com/archives/category/web-server-survey/
Methodology: https://www.netcraft.com/active-sites/
2019: response by **1,477,803,927** sites, **229,586,773** unique domains, and **8,366,753** web-facing computers



Apache

- Developed by the *Apache Software Foundation* (v.1 released April 1995)
- Bundled in the (once but still) popular "LAMP" package (**L**inux **A**pache **M**ysql **P**hp)
- in netkit we already have a simple Apache 2.2.9 installed – with fewer options

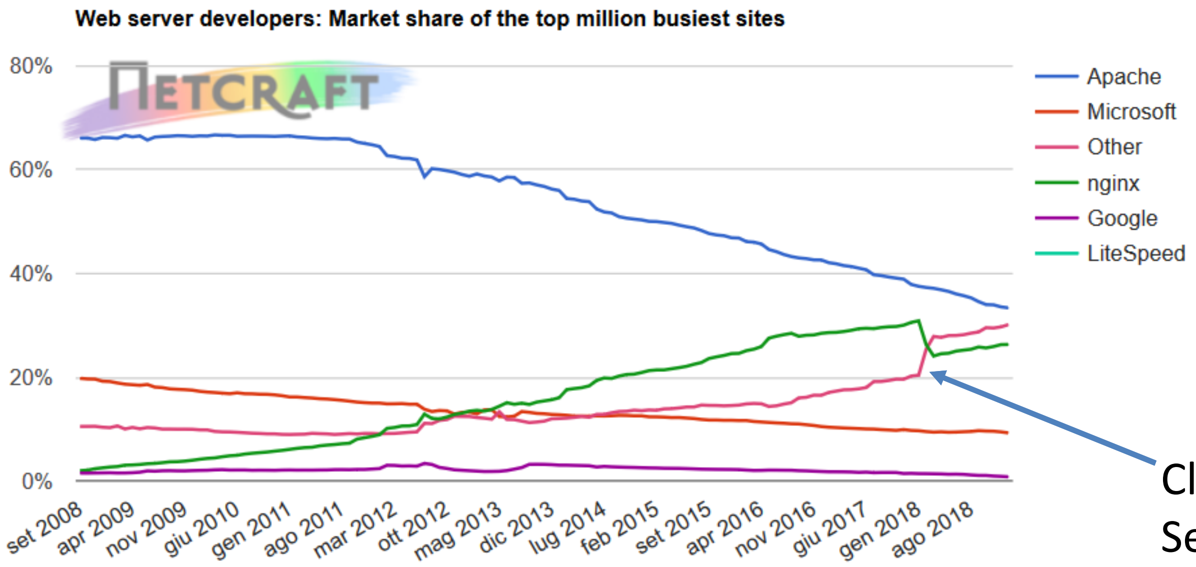
- Install apache2 on the Linux host (available for all platforms).
  - apt-get install apache2

try!

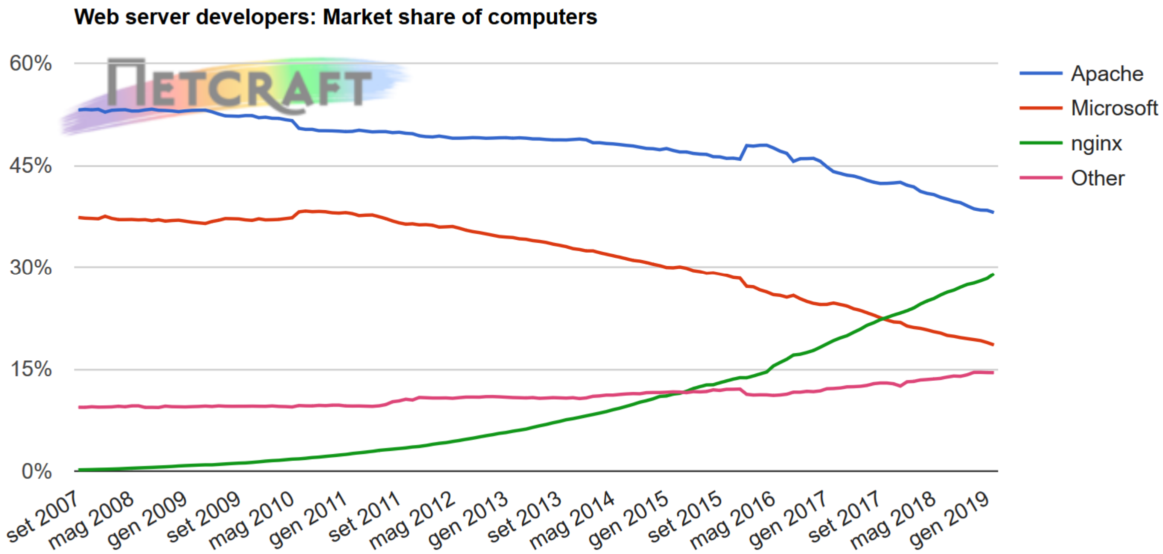
- Start / stop:
  - /etc/init.d/apache2 start
  - /etc/init.d/apache2 stop
- Open browser and go to http://localhost
  - Change the content in /var/www

Src https://news.netcraft.com/archives/category/web-server-survey/

Apache diffusion


Web server developers: Market share of the top million busiest sites

Cloudflare changed Server from nginx to “cloudflare”
(nginx based)


Web server developers: Market share of computers

Where to study

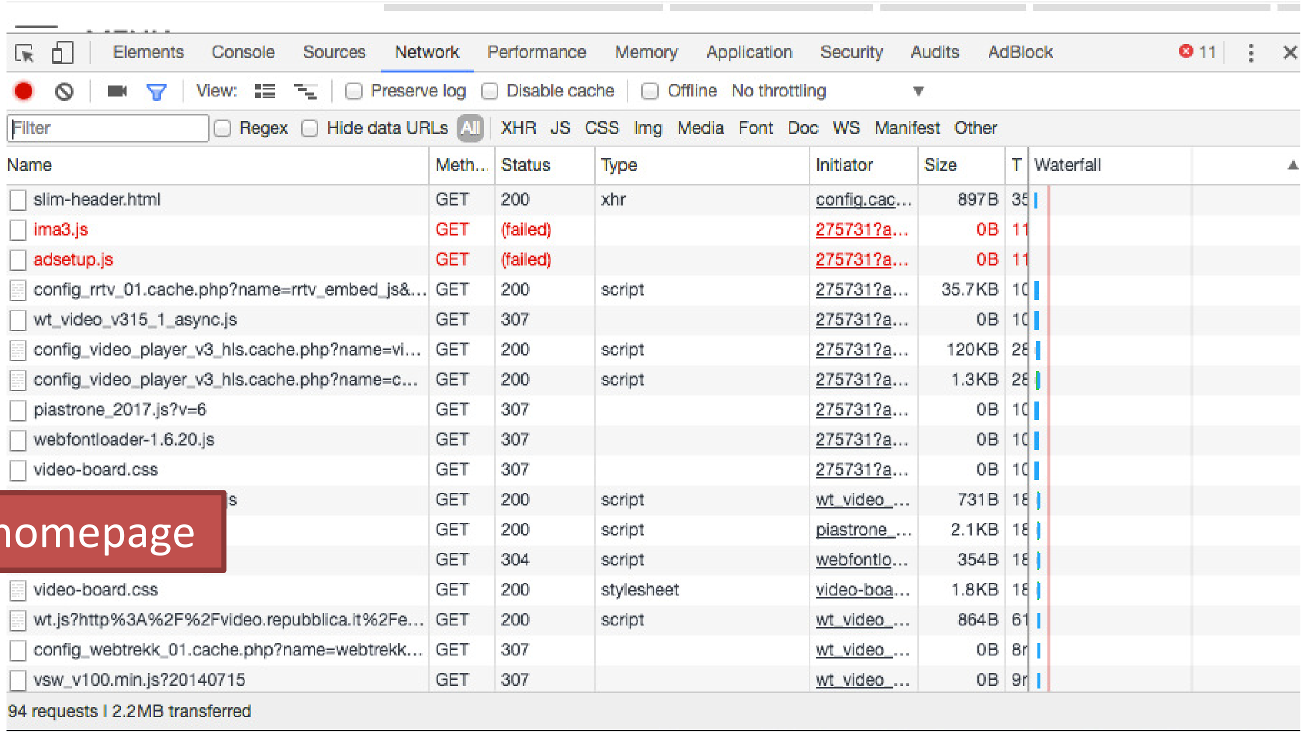
- Apache Server 2 - Mohammed J. Kabir
  - Hungry Minds
- Apache Server 2 Official Documentation
  - http://httpd.apache.org/docs/2.0/

Web servers processing

- Serve many clients
  - Many requests from same client!!
- Parallelism needed

la Repubblica.it

08 Maggio 2017 - Aggiornato alle **11.06**



Name	Meth...	Status	Type	Initiator	Size	T	Waterfall
slim-header.html	GET	200	xhr	config.cac...	897B	35	
ima3.js	GET	(failed)		275731?a...	0B	11	
adsetup.js	GET	(failed)		275731?a...	0B	11	
config_rrtv_01.cache.php?name=rrtv_embed_js&...	GET	200	script	275731?a...	35.7KB	10	
wt_video_v315_1_async.js	GET	307		275731?a...	0B	10	
config_video_player_v3_hls.cache.php?name=vi...	GET	200	script	275731?a...	120KB	28	
config_video_player_v3_hls.cache.php?name=c...	GET	200	script	275731?a...	1.3KB	28	
piastrone_2017.js?v=6	GET	307		275731?a...	0B	10	
webfontloader-1.6.20.js	GET	307		275731?a...	0B	10	
video-board.css	GET	307		275731?a...	0B	10	
s	GET	200	script	wt_video_...	731B	18	
	GET	200	script	piastrone_...	2.1KB	18	
	GET	304	script	webfontlo...	354B	18	
video-board.css	GET	200	stylesheet	video-boa...	1.8KB	18	
wt.js?http%3A%2F%2Fvideo.repubblica.it%2Fe...	GET	200	script	wt_video_...	864B	61	
config_webtrekk_01.cache.php?name=webtrekk...	GET	307		wt_video_...	0B	8r	
vsw_v100.min.js?20140715	GET	307		wt_video_...	0B	9r	

94 requests | 2.2MB transferred

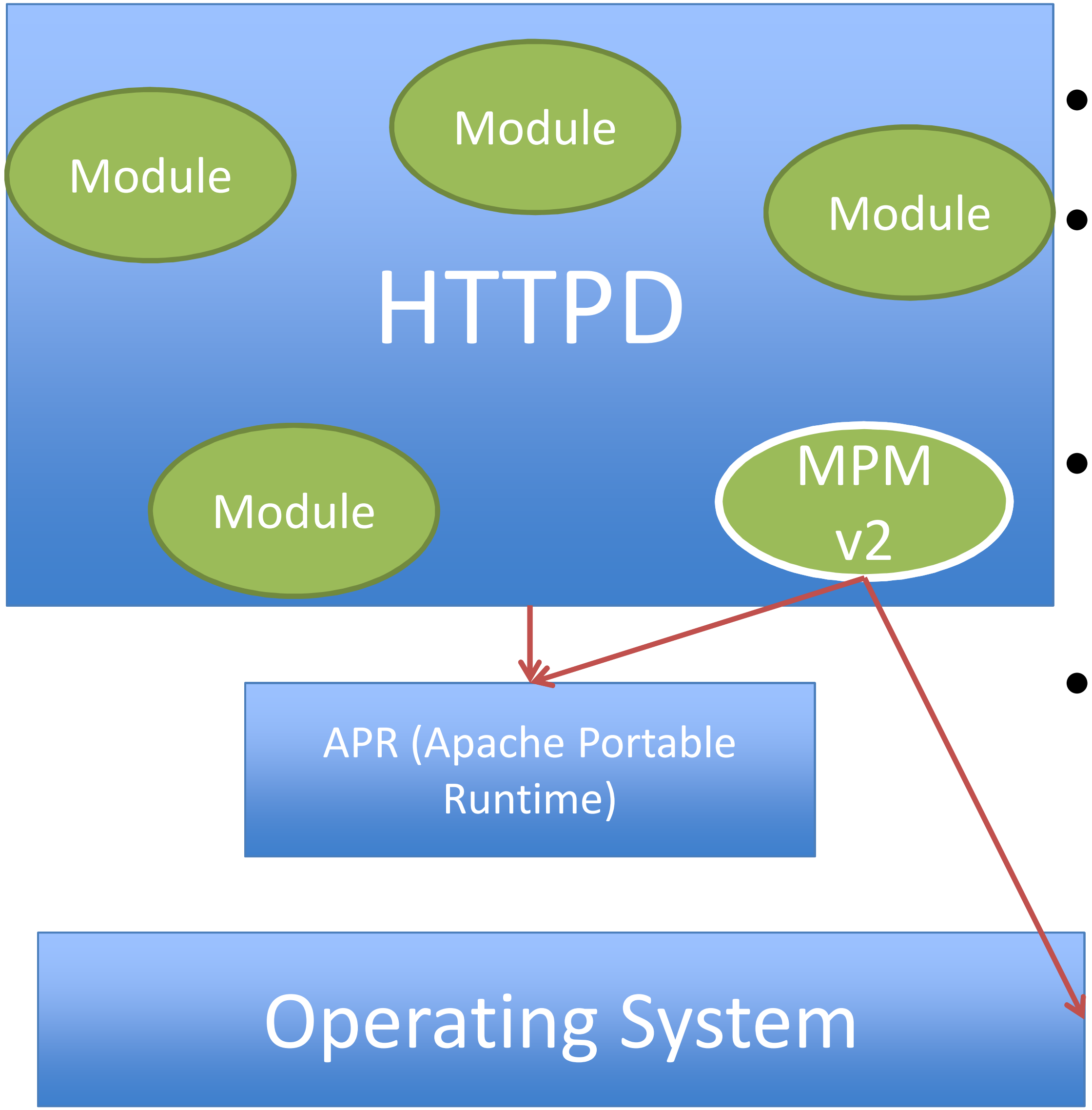
94 requests for just the homepage

Processes vs Threads

- Both threads and processes are methods of parallelizing an application
- **Processes** are independent execution units that contain their own state information, use their own address spaces, and only interact with each other via inter-process communication (IPC) mechanisms
- **Threads** share the same state and same memory space, and can communicate with each other directly, because they share the same variables

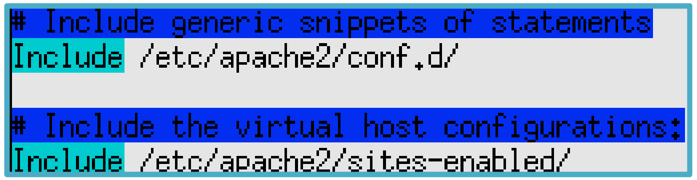
Are your cgi library thread safe?

Apache Architecture



- small core
- several modules
  - compiled statically or loaded dynamically
- Cross platform utilities (APR)
- MultiProcessing Modules
  - Deals with o.s. to handle multiple parallel requests

Apache Web server files



```
# Include generic snippets of statements
Include /etc/apache2/conf.d/

# Include the virtual host configurations:
Include /etc/apache2/sites-enabled/
```

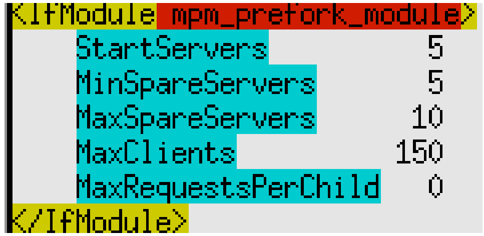
/usr/sbin/apache2	Apache 2 server binary
/usr/sbin/apache2ctl	Apache2 control interface (configtest could help!)
/etc/apache2/apache2.conf	default configuration file (could be overwritten during apache upgrade)
/etc/apache2/httpd.conf	User/legacy configuration files (and files inside /etc/apache2/conf.d)
/etc/apache2/conf.d	other configuration files (included as well in apache2.conf)
/etc/apache2/ports.conf	Listening ports (and virtualhosting) main config
/etc/apache2/sites-available	configuration files for virtual hosting
/etc/apache2/sites-enable	symbolic links to sites-available files (created with a2ensite, a2dissite)
/etc/apache2/mods-available	configuration files for modules
/etc/apache2/mods-enabled	symbolic links to mods-available files (created with a2enmod, a2dismod)
/var/log/apache2	log files

Apache Modules

- Apache has modular architecture:
  - To enable/disable modules : a2enmod / a2dismod MODNAME
  - configurable via commands
    - apache2ctl –M #list of modules
  - mod_so load module at runtime (Dynamic Shared Object (DSO) mechanism) LoadModule

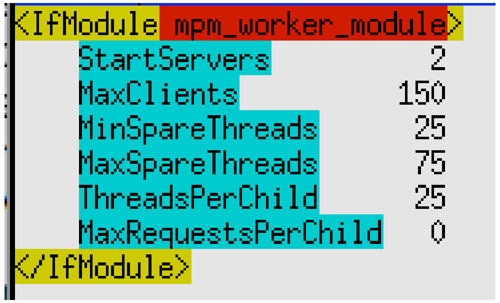
Apache MultiProcessing Module

- MultiProcessing Modules (MPMs) since Apache2:
  - In apache 1.3 uses a preforking architecture
    - the parent creates/destroys children if required
    - does not work well on some platform (such Windows)
  - MPM offers several alternatives (implemented in MPM modules) :
    - prefork MPM (like Apache 1.3)
    - worker MPM (multiple child, each one with several threads)
    - winnt MPM: single process, multithread (specific for windows)
    - event MPM: like worker, improved (dedicated thread to deal with the kept-alive connections)



```
<IfModule mpm_prefork_module>
    StartServers          5
    MinSpareServers       5
    MaxSpareServers      10
    MaxClients          150
    MaxRequestsPerChild   0
</IfModule>
```

We can tune parameters in /etc/apache2/apache2.conf

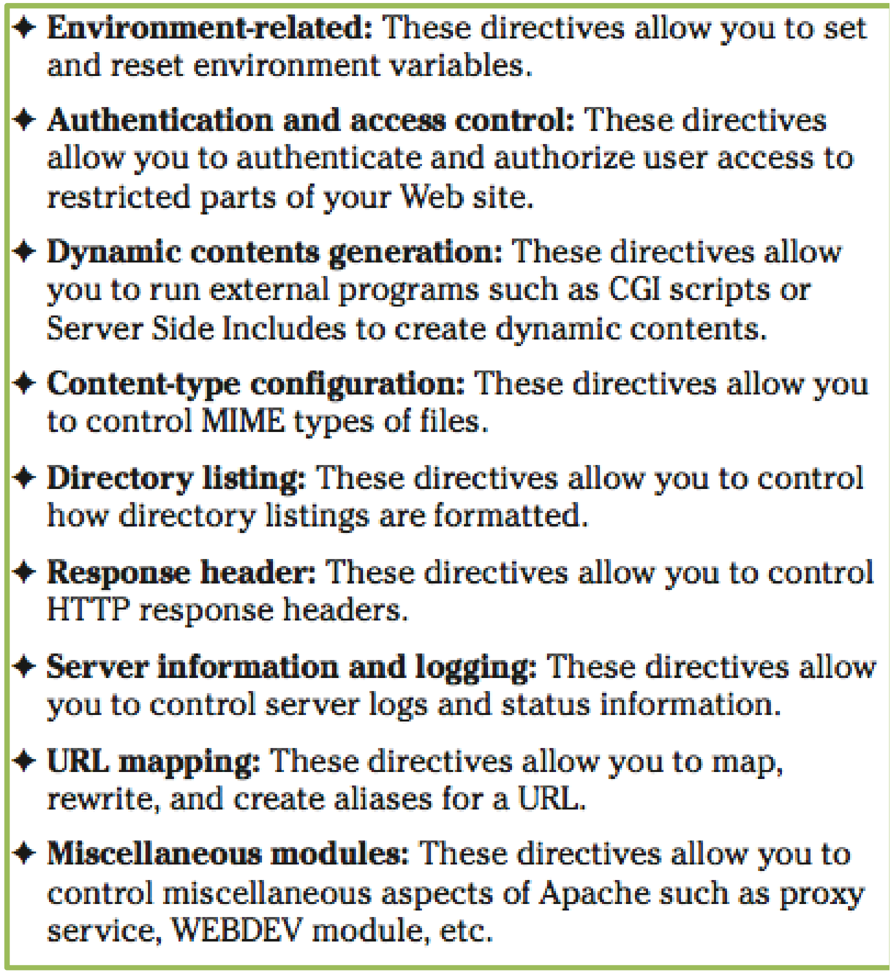


```
<IfModule mpm_worker_module>
    StartServers          2
    MaxClients          150
    MinSpareThreads      25
    MaxSpareThreads      75
    ThreadsPerChild      25
    MaxRequestsPerChild   0
</IfModule>
```

- Check which apache mpm we currently use
  - apache2ctl -V | grep -i mpm
- List Available MPM Modules
  - ls /etc/apache2/mods-available/mpm*
- List enabled MPM modules
  - ls -l /etc/apache2/mods-enabled/mpm*

Configuring Apache

- ~ 360 directives (!!!). Few selected:
  - ServerRoot: path to configuration, error and log files
  - PidFile
  - ServerName: name and port of the server
  - DocumentRoot: where find files to serve
  - ErrorDocument: override standard error messages



- **Environment-related:** These directives allow you to set and reset environment variables.
- **Authentication and access control:** These directives allow you to authenticate and authorize user access to restricted parts of your Web site.
- **Dynamic contents generation:** These directives allow you to run external programs such as CGI scripts or Server Side Includes to create dynamic contents.
- **Content-type configuration:** These directives allow you to control MIME types of files.
- **Directory listing:** These directives allow you to control how directory listings are formatted.
- **Response header:** These directives allow you to control HTTP response headers.
- **Server information and logging:** These directives allow you to control server logs and status information.
- **URL mapping:** These directives allow you to map, rewrite, and create aliases for a URL.
- **Miscellaneous modules:** These directives allow you to control miscellaneous aspects of Apache such as proxy service, WEBDEV module, etc.

Apache “content” folders

- DocumentRoot can be specified in the main config files (obsolete): /etc/apache2/httpd.conf or apache2.conf
- Best practice: available “sites”
  - /etc/apache2/sites-available
- Sites enabled (by command a2ensite) creates a symbolic link into
  - /etc/apache2/sites-enabled
- This was done to support easier adding-removing of a web site and support virtual hosting (more on that later). Simplest site.conf

```
<VirtualHost *:80>
	ServerAdmin webmaster@localhost
	DocumentRoot /var/www/html/site1
</VirtualHost>
```

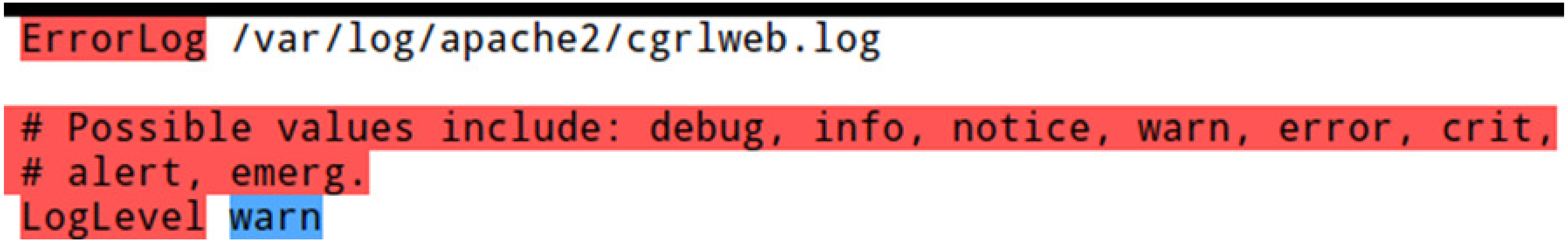
DocumentRoot

- Where apache finds your documents (html files etc)
  - Typically search for: index.html index.cgi index.pl index.php index.xhtml index.htm
  - Defined from DirectoryIndex (order matters)

```
<html>
      <body>
            <h1>
                  HELLO CGRL
            </h1>
      </body>
</html>
```

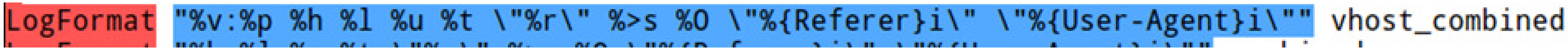
index.html sample file

Logging



```
ErrorLog /var/log/apache2/cgrlweb.log

# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
LogLevel warn
```

- location and content: CustomLog directive
- Format: LogFormat
  - specified with common logfile format*



```
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" vhost_combined
```

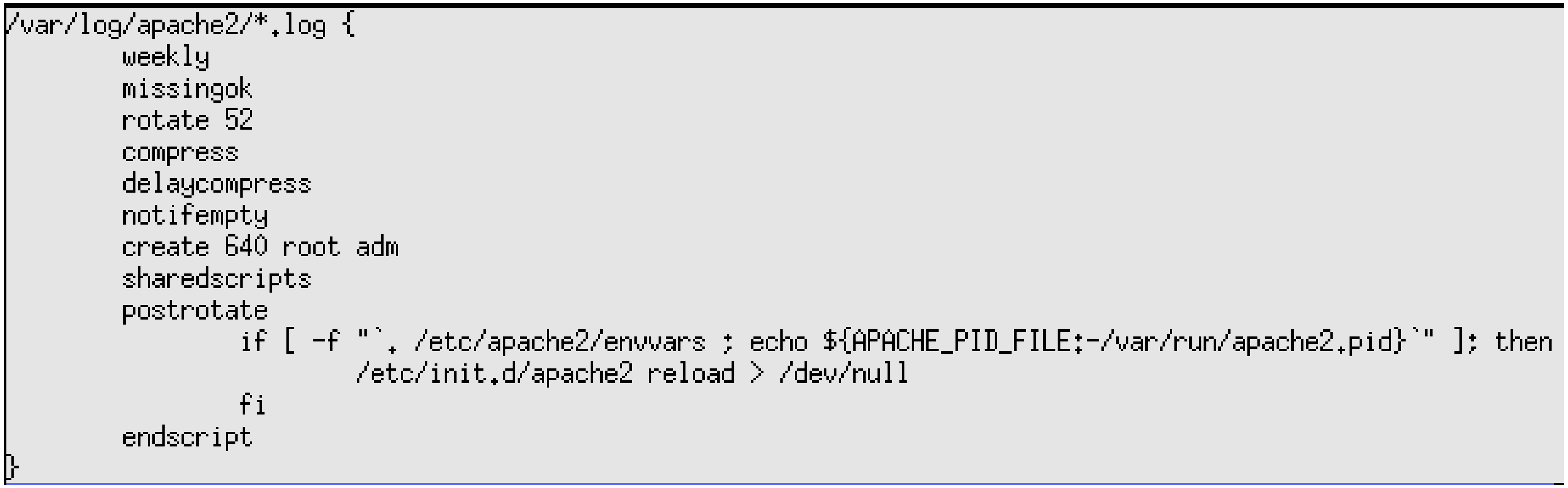
(*) http://www.w3.org/Daemon/User/Config/Logging.html#common-logfile-format

Logrotate

not only for apache!

Log Size: Typically 1 MB for 10000 requests so…

/etc/logrotate.d/apache2



```
/var/log/apache2/*.log {
        weekly
        missingok
        rotate 52
        compress
        delaycompress
        notifempty
        create 640 root adm
        sharedscripts
        postrotate
                if [ -f "`. /etc/apache2/envvars ; echo ${APACHE_PID_FILE:-/var/run/apache2.pid}`" ]; then
                        /etc/init.d/apache2 reload > /dev/null
                fi
        endscript
}
```

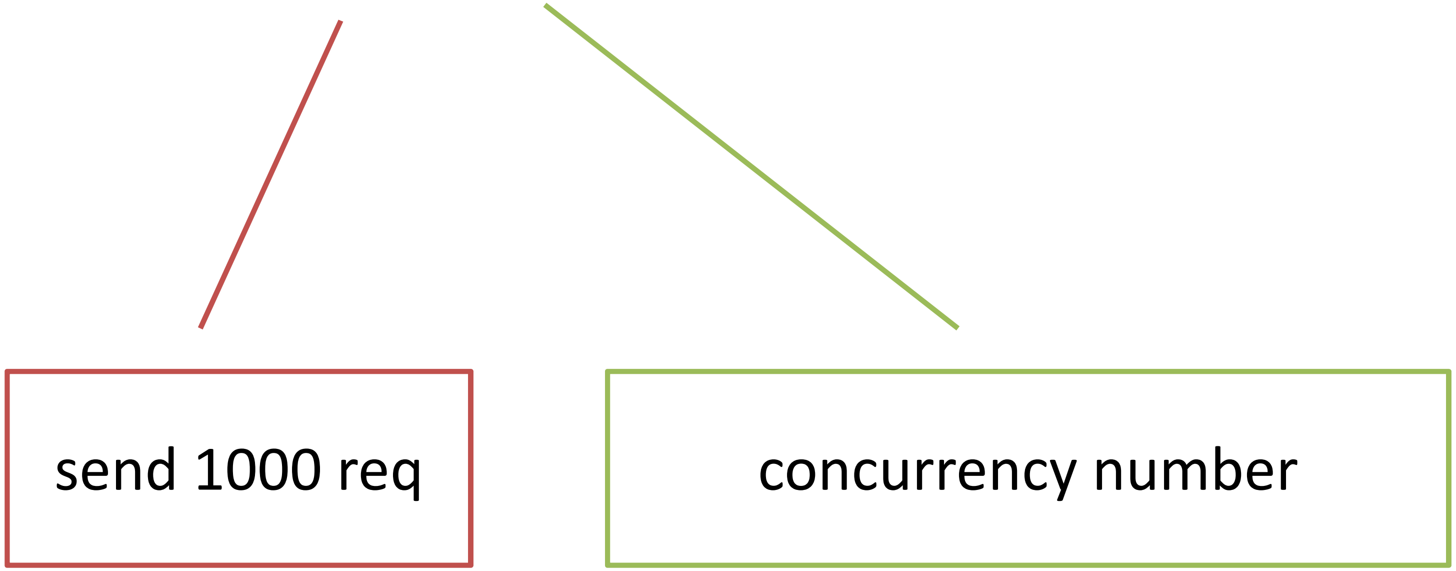
- rotate at most 52 times, weekly
- compress (you can see that using zcat, zless or pipelining gzip and cat/tail)

Mining/security

- Access logs, error logs can be huge files.
- AW statistics https://www.nltechno.com/awstats/awstats.pl?config=destailleur.fr&month=04&year=2018
- SCALP treats identification (post processing): https://code.google.com/archive/p/apache-scalp/ (enable mod_security!)
- fail2ban: scans logfile in realtime to identify malicious access

Apache benchmarking

- ab (Apache HTTP server benchmarking tool)
- ab -n 1000 -c 5 http://URL_TO_TEST/index.html



send 1000 req

concurrency number

Reference documentation:
http://httpd.apache.org/docs/2.2/programs/ab.html

apt-get install apache2-utils

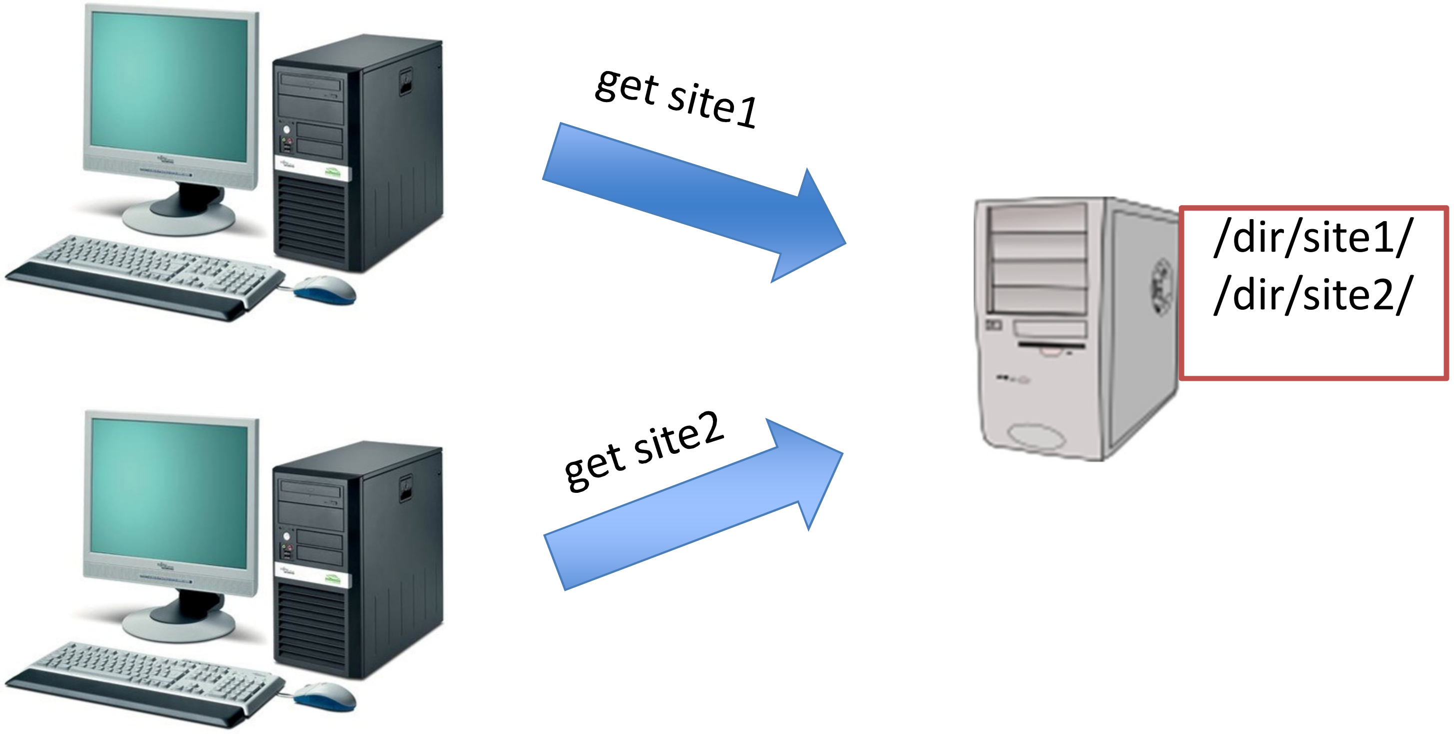
Exercise 1

- Create a lab with two machines, server1 and client1.
  - on server1 startup: `/etc/init.d/apache2 start`
  - by default site "default" is enabled (as 000-default)
- Replace default site (disable it) with a test config
- <u>Optional</u> try apache benchmarking from client1 to server1
    - Warning: page/s is not the only thing to consider
    - Try as well a "larger" file created with dd

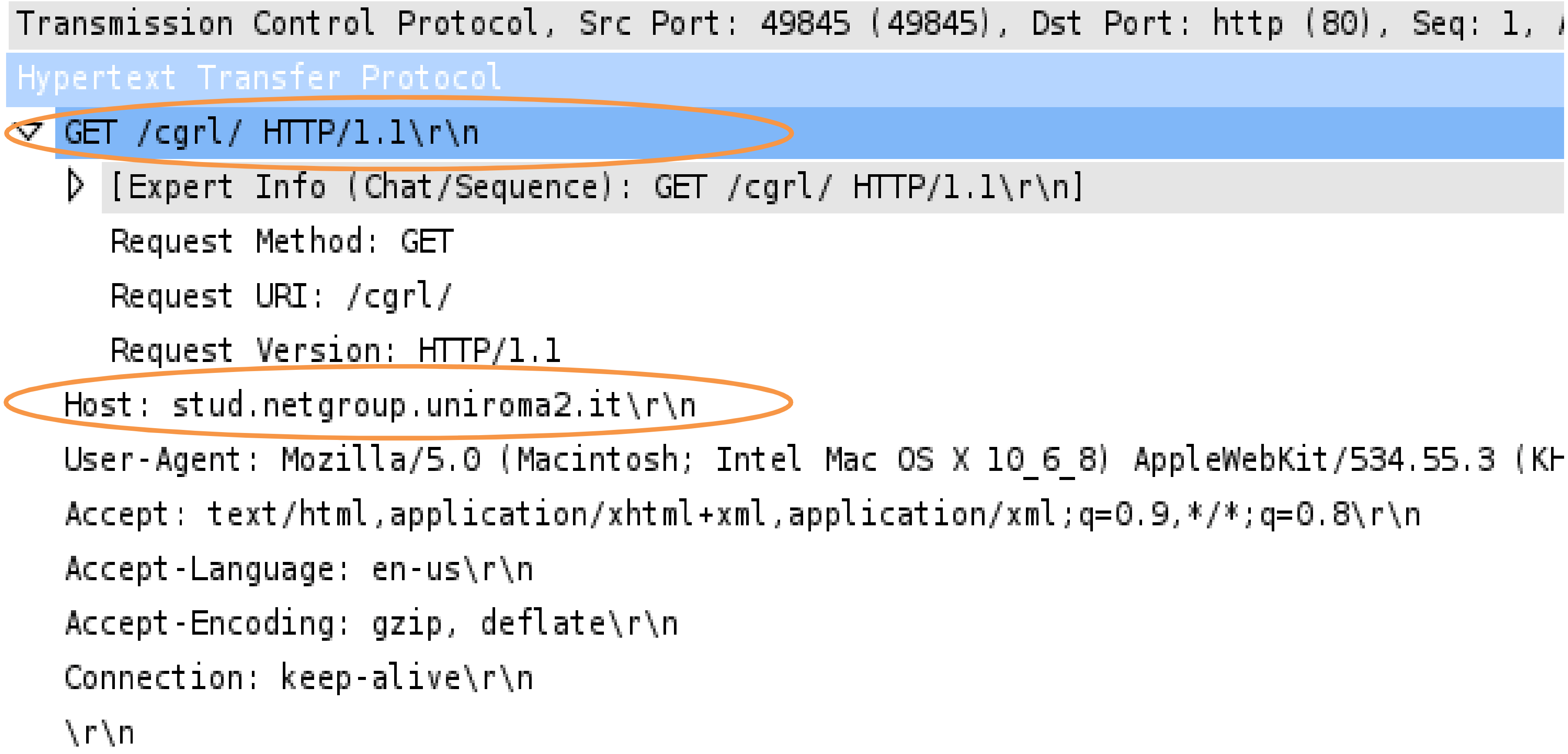
```
dd if=/dev/urandom of=file.txt bs=1k
count=100
```

Virtual Hosting

- Problem: *Several* websites, *one* webserver
  - Typically: *name-based* virtual host (with help of DNS!)

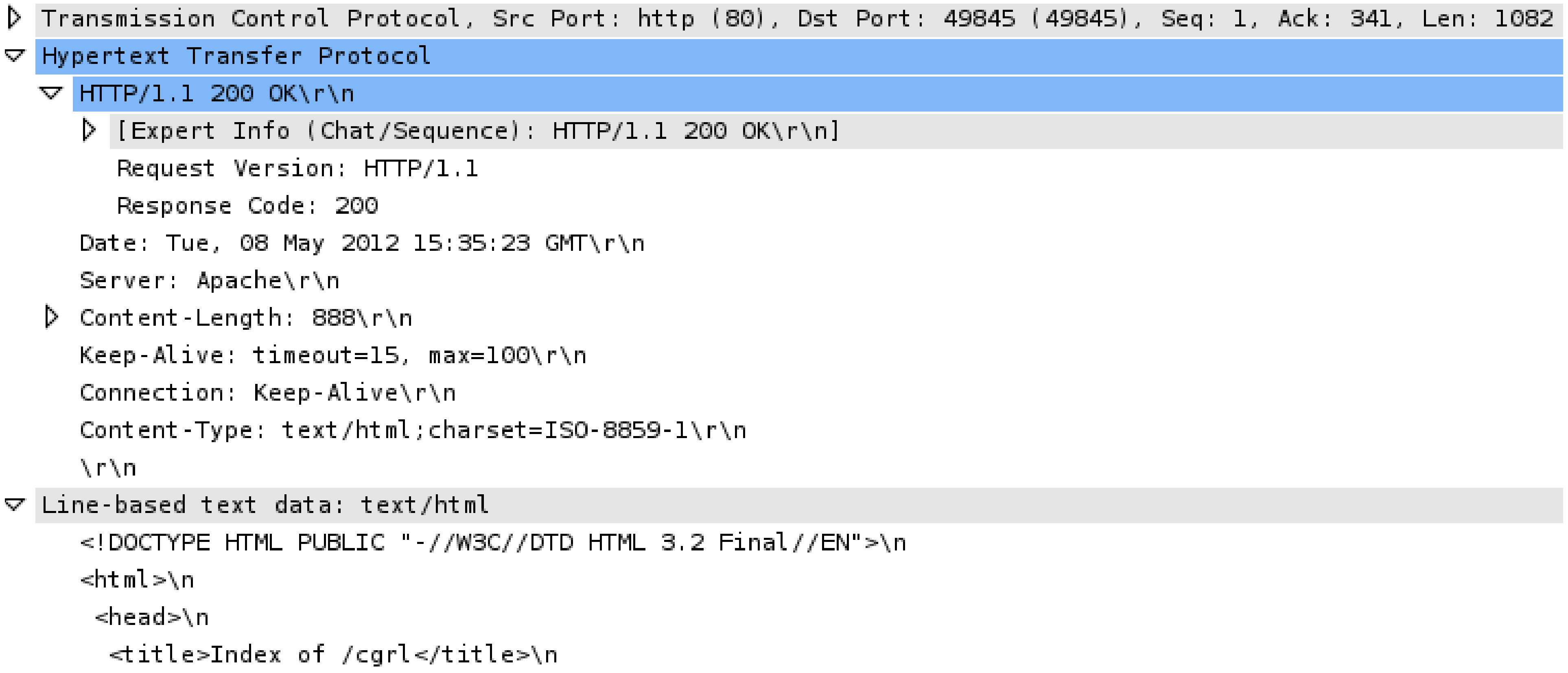


HTTP Request



```
Transmission Control Protocol, Src Port: 49845 (49845), Dst Port: http (80), Seq: 1,
Hypertext Transfer Protocol
▽ GET /cgrl/ HTTP/1.1\r\n
   ▷ [Expert Info (Chat/Sequence): GET /cgrl/ HTTP/1.1\r\n]
     Request Method: GET
     Request URI: /cgrl/
     Request Version: HTTP/1.1
  Host: stud.netgroup.uniroma2.it\r\n
  User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_6_8) AppleWebKit/534.55.3 (KH
  Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n
  Accept-Language: en-us\r\n
  Accept-Encoding: gzip, deflate\r\n
  Connection: keep-alive\r\n
  \r\n
```

HTTP Response



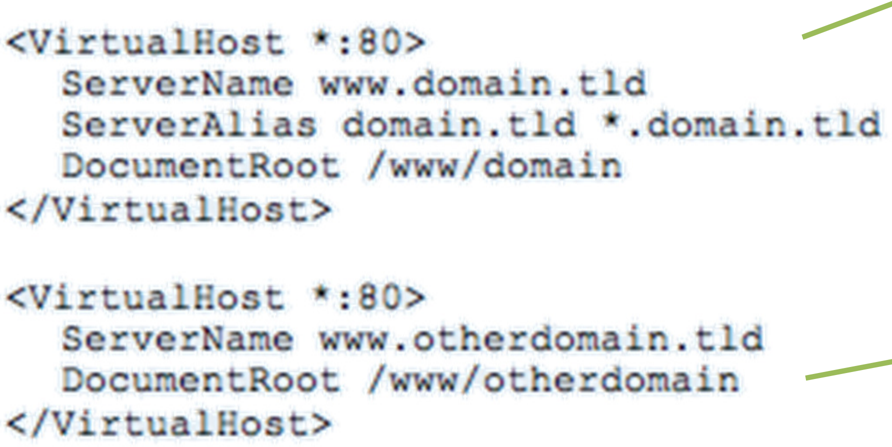
```
▷ Transmission Control Protocol, Src Port: http (80), Dst Port: 49845 (49845), Seq: 1, Ack: 341, Len: 1082
▽ Hypertext Transfer Protocol
  ▽ HTTP/1.1 200 OK\r\n
    ▷ [Expert Info (Chat/Sequence): HTTP/1.1 200 OK\r\n]
      Request Version: HTTP/1.1
      Response Code: 200
    Date: Tue, 08 May 2012 15:35:23 GMT\r\n
    Server: Apache\r\n
  ▷ Content-Length: 888\r\n
    Keep-Alive: timeout=15, max=100\r\n
    Connection: Keep-Alive\r\n
    Content-Type: text/html;charset=ISO-8859-1\r\n
    \r\n
▽ Line-based text data: text/html
    <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">\n
    <html>\n
     <head>\n
      <title>Index of /cgrl</title>\n
```

Virtual Hosting: example

IP/Port for listening requests. Enabling “name-based” VHost. ports.conf

```
NameVirtualHost *:80
Listen 80
```

put virtual hosts configs in **sites-available** dir!



```
<VirtualHost *:80>
  ServerName www.domain.tld
  ServerAlias domain.tld *.domain.tld
  DocumentRoot /www/domain
</VirtualHost>

<VirtualHost *:80>
  ServerName www.otherdomain.tld
  DocumentRoot /www/otherdomain
</VirtualHost>
```

Name of the virtual host (match **host** http field). Put into site1.conf

Second site. Put into site2.conf

use /etc/hosts to test virtual host without DNS modifications

Exercise 2: virtual hosting

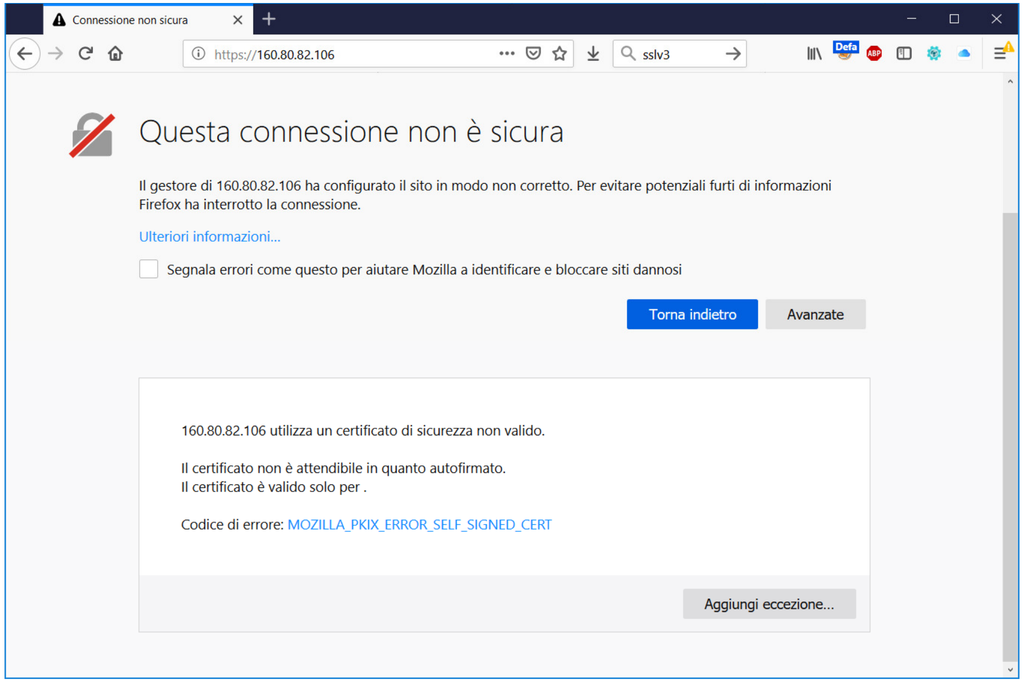
1. Create two dirs “cgrl-web” and “cgrl-mail”. Put in these directories two different index.html files
2. Configure 2 virtual web hosts
   - www.studenti.cgrl.edu
   - webmail.studenti.cgrl.edu
3. enable them on server1 and restart apache2 – disable default site (or edit server1.startup accordingly for auto-startup)
   1. a2ensite cgrl-web
   2. a2ensite cgrl-mail
4. modify nameserver (or /etc/host) configuration
   - Ping to verify they have the same IP
5. View your websites with the text browser from client1 (lynx or links).

HTTPS

- Establish confidentiality (end-to-end) BEFORE actual data transfer (both requests and responses):
  - Use of port 443/TCP
  - Use of TLS protocol (typically TLS1.2, today TLS1.3). Older versions unsupported (SSLv3).
  - Asymmetric encryption, public keys distributed by X.509 certificates.
  - Server hosts a private key, used to cypher the traffic. Should I trust the Server?

HTTPS

- HTTPS is possible with self-signed certificates
  - Traffic is encrypted, leveraging “local” certificate
  - Warning by clients (add exception)
  - I might not be really sure that the server is who says it is
    - Ok for local services (intranet), or testing setups



Questa connessione non è sicura

Il gestore di 160.80.82.106 ha configurato il sito in modo non corretto. Per evitare potenziali furti di informazioni Firefox ha interrotto la connessione.

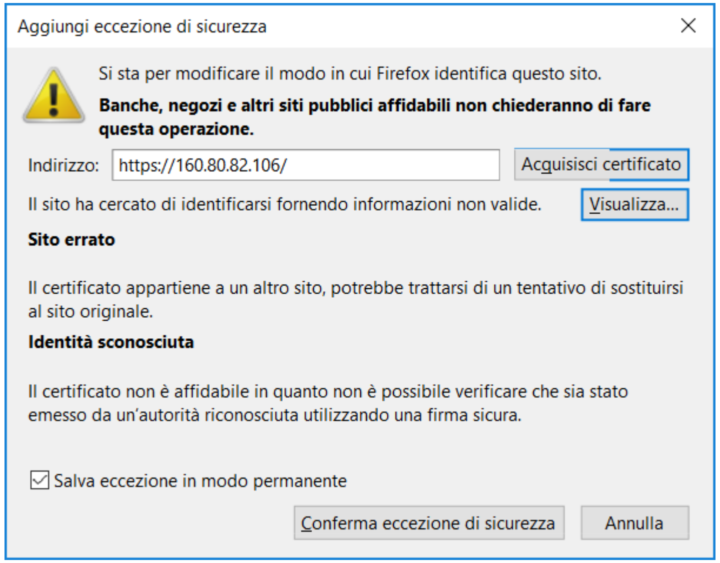
Ulteriori informazioni…

□ Segnala errori come questo per aiutare Mozilla a identificare e bloccare siti dannosi

160.80.82.106 utilizza un certificato di sicurezza non valido.

Il certificato non è attendibile in quanto autofirmato.
Il certificato è valido solo per .

Codice di errore: MOZILLA_PKIX_ERROR_SELF_SIGNED_CERT



Aggiungi eccezione di sicurezza

Si sta per modificare il modo in cui Firefox identifica questo sito.

Banche, negozi e altri siti pubblici affidabili non chiederanno di fare questa operazione.

Indirizzo: https://160.80.82.106/

Il sito ha cercato di identificarsi fornendo informazioni non valide.

Sito errato

Il certificato appartiene a un altro sito, potrebbe trattarsi di un tentativo di sostituirsi al sito originale.

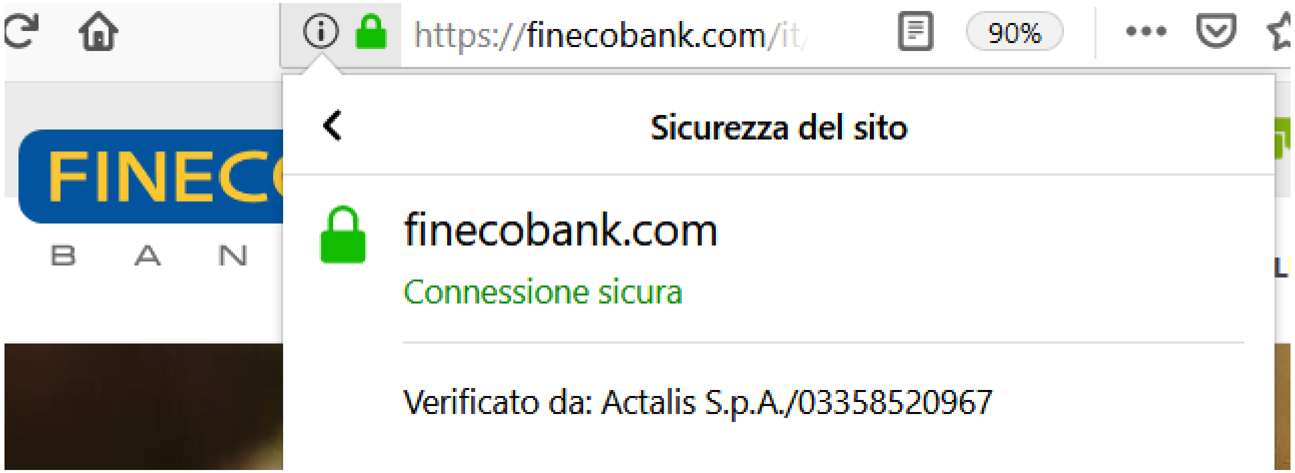
Identità sconosciuta

Il certificato non è affidabile in quanto non è possibile verificare che sia stato emesso da un’autorità riconosciuta utilizzando una firma sicura.

☑ Salva eccezione in modo permanente

HTTPS

- Trough a Verification entity, who releases the certificates and verify that private key is entrusted to who effectively hosts the target domain(s).



https://finecobank.com/it/

Sicurezza del sito

finecobank.com

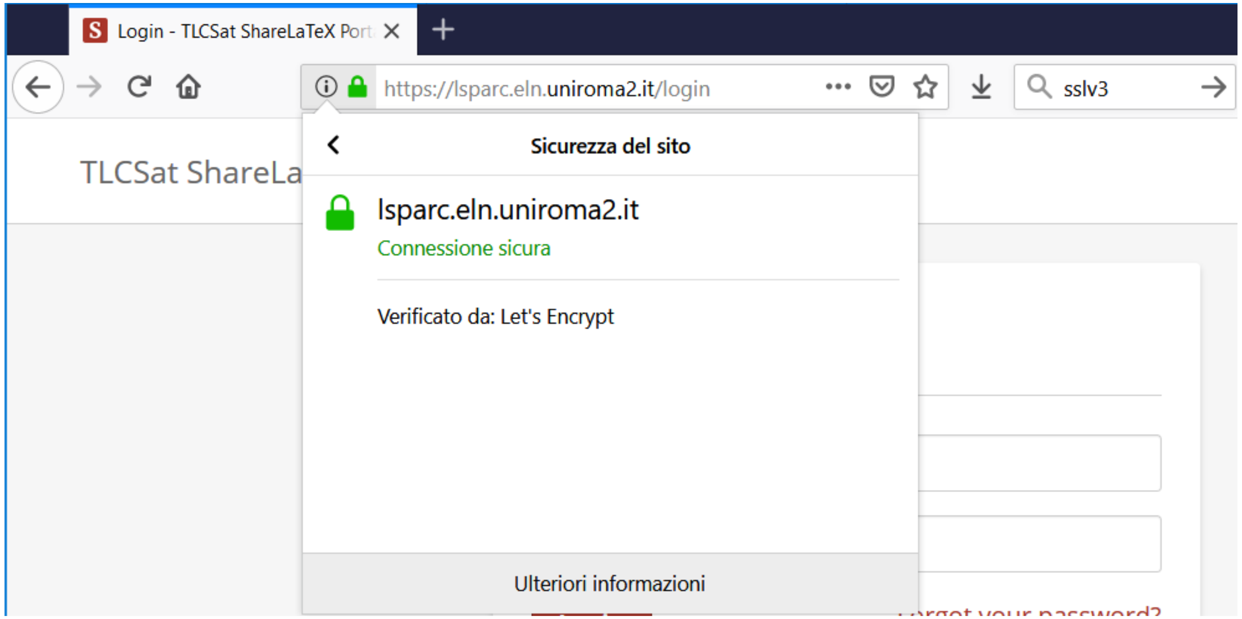
Connessione sicura

Verificato da: Actalis S.p.A./03358520967

- Payed service, several companies available, eg.
  - https://www.cheapsslshop.com/

HTTPS

- Let's encrypt: free. https://letsencrypt.org/
  - Let's make the whole Web secure! https://blog.mozilla.org/security/2015/04/30/deprecating-non-secure-http/
  - Limited applicability (only domain, not IP, not company wide; short duration)
  - Scripts included in Debian



Login - TLCSat ShareLaTeX Port
https://lsparc.eln.uniroma2.it/login
sslv3
TLCSat ShareLa

Sicurezza del sito

lsparc.eln.uniroma2.it

Connessione sicura

Verificato da: Let's Encrypt

HTTPS VirtualHosting

- “GET” request, including target URL, is only available AFTER the TLS handshake. Therefore, server can only use a “default” certificate to start, which might not be the one specifically built for that host name
  - https://wiki.apache.org/httpd/NameBasedSSLVHostsWithSNI
- Solution:
  - use multiple domains certificates (not scalable)
  - use Service Name Identification SNI extension, available since Apache 2.2.12 (RFC 4366). Require also web browser support. Today it is the default for all.

HTTPS lab config

- Self-signed certificate (we can not use Let’s encrypt, since we need a REAL DNS resolution). Virtual HTTPS hosting in netkit’s Apache2 is not available ☹
- A self-signed certificate is already available in netkit (expired in 2018)
- By the way, we can create our own certificate:

```
mkdir /tmp/certificates
cd /tmp/certificates
openssl req -x509 -newkey rsa:4096 -keyout apache.key -out apache.crt -days 365 -nodes

    Country Name (2 letter code) [AU]: IT
    State or Province Name (full name) [Some-State]: RM
    Locality Name (eg, city) []: Rome
    Organization Name (eg, company) [My Company]: CGRL
    Organizational Unit Name (eg, section) []:
    Common Name (e.g. server FQDN or YOUR name) []: www.studenti.cgrl.edu
    Email Address []:webmaster@cgrl.edu
```

HTTPS lab config

Enable SSL (this also enable listening on port 443 at reload; see ports.config):
a2enmod ssl

Create and edit in sites-available «cgrl-sec.conf» file and then enable it:
a2ensite cgrl-sec

Root can be the same of non-HTTPS, or point to a new DocumentRoot!

```
<IfModule mod_ssl.c>
<VirtualHost _default_:443>
        ServerAdmin cgrl-webmaster@cgrl.edu

#       ServerName www.cgrl.edu
        DocumentRoot /root/cgrl-web

        ErrorLog /var/log/apache2/cgrlweb.log

LogLevel warn
        CustomLog /var/log/apache2/access.log
combined
          SSLCertificateFile    /etc/ssl/certs/ssl-cert-
snakeoil.pem
        SSLCertificateKeyFile /etc/ssl/private/ssl-cert-
snakeoil.key

        #   SSL Engine Switch:
        #   Enable/Disable SSL for this virtual host.
        SSLEngine on

</VirtualHost>
</IfModule>
```

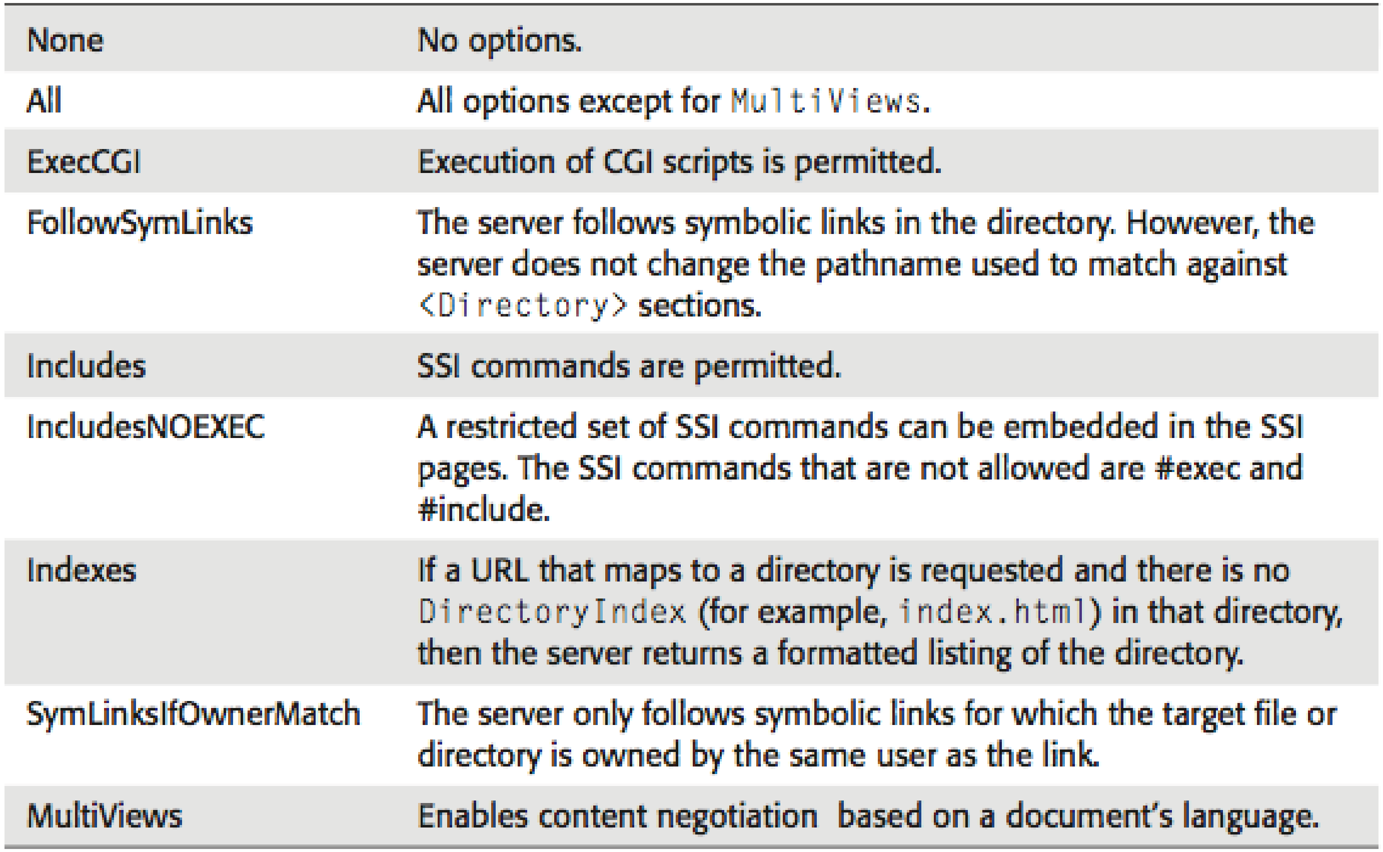
Exercise 3 HTTPS (optional)

- Create custom self-signed certificate
- Adjust the lab config to make server to be reachable by the host machine (tap interface):
  - server1[0]=tap,10.250.1.1,10.250.1.2
- Test both HTTP and HTTPS pointing to the same folder using hosts’ browser (e.g.,Firefox)

Standard Container Directives

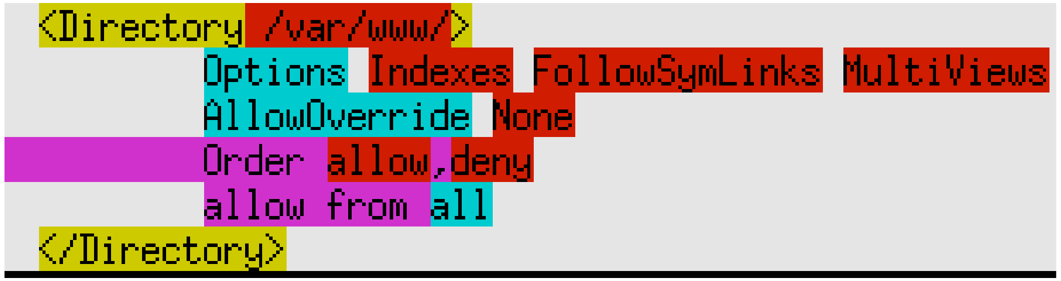
- Many Container contexts:
  - <VirtualHost ...>: already seen...
  - <Directory>: applies one or more directives to a directory
  - <Files>: applies one or more directives to a file
  - <Location>: applies one or more directives to a URL
- AllowOverride: enable/disable directories directives overriding.
- .htaccess: default filename for the per-directory configuration

Options directive



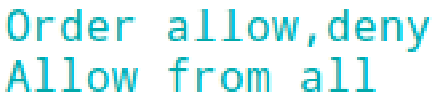
None	No options.
All	All options except for `MultiViews`.
ExecCGI	Execution of CGI scripts is permitted.
FollowSymLinks	The server follows symbolic links in the directory. However, the server does not change the pathname used to match against `<Directory>` sections.
Includes	SSI commands are permitted.
IncludesNOEXEC	A restricted set of SSI commands can be embedded in the SSI pages. The SSI commands that are not allowed are #exec and #include.
Indexes	If a URL that maps to a directory is requested and there is no `DirectoryIndex` (for example, `index.html`) in that directory, then the server returns a formatted listing of the directory.
SymLinksIfOwnerMatch	The server only follows symbolic links for which the target file or directory is owned by the same user as the link.
MultiViews	Enables content negotiation based on a document's language.

Options +Setting1 – Setting2



```
<Directory /var/www/>
        Options Indexes FollowSymLinks MultiViews
        AllowOverride None
        Order allow,deny
        allow from all
</Directory>
```

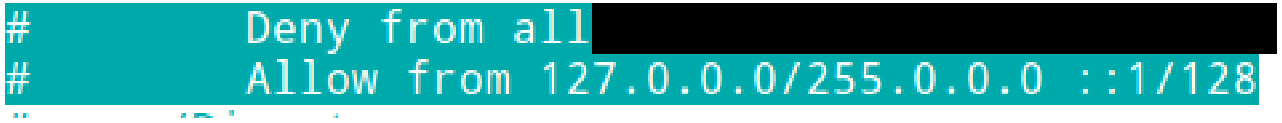
Allow-Deny



```
Order allow,deny
Allow from all
```

- First, all Allow directives are evaluated in order; at least one must match, or the request is rejected (deny). Next, all Deny directives are evaluated. If any matches, the request is rejected. Last, any requests which do not match an Allow or a Deny directive are denied by default.

A domain name, IP, network/netmask (CIDR)



```
#         Deny from all
#         Allow from 127.0.0.0/255.0.0.0 ::1/128
```

Exercise 4: directory listening

1. Take the previous example
2. Create a directory in your DocumentRoot "myfiles" and put some stuff (try a symbolic link) inside that

   ```
   <Directory /your/dir/myfiles>
   Options +Indexes
   </Directory>
   ```

3. Create a directory inside "myfiles": "mysecretfiles":

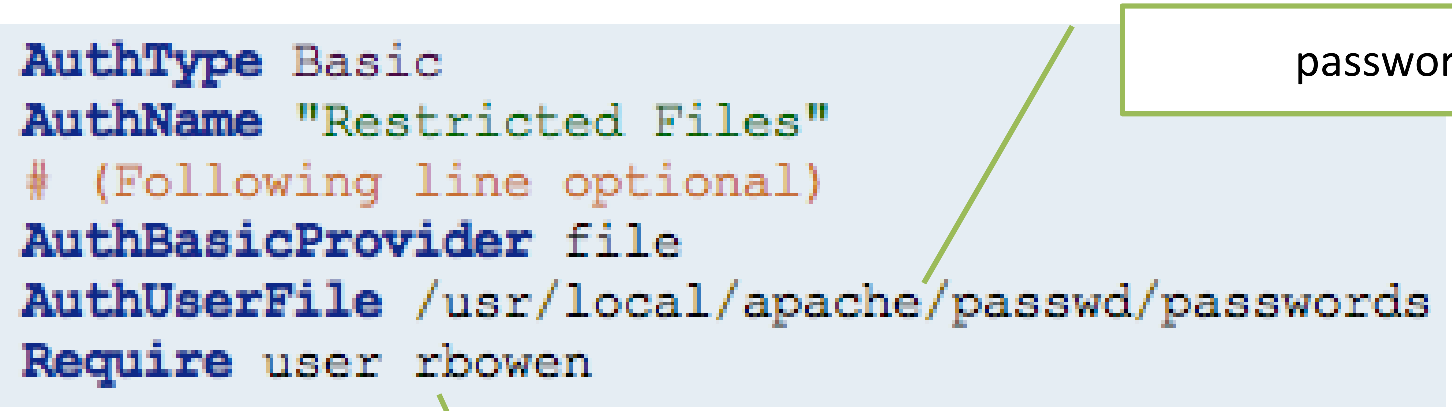
   ```
   <Directory /your/dir/myfiles/mysecrefiles>
   Options -Indexes
   </Directory>
   ```

.htaccess

- Same syntax as the main configuration files
  - so use <Directory> block instead (it’s faster!)
  - Common misconception: not specifically for passwords or rewrite!
- “AllowOverride” : Types of directives that are allowed in .htaccess files (None, All, one or more directive inside these groups: {**AuthConfig, FileInfo, Indexes, Limit, Options**})

Exercise 5: .htaccess password protection

- Let we create a new file with passwords:
  - htpasswd –c *PASSWORDFILE USERNAME*
  - Then put these directives in .htaccess (or <Directory>)



```
AuthType Basic
AuthName "Restricted Files"
# (Following line optional)
AuthBasicProvider file
AuthUserFile /usr/local/apache/passwd/passwords
Require user rbowen
```

password file

users

- Now protect our “secret” file...
  - P.s. passwords are hashed (MD5)

Change the user and group ownership of ***.htdigest*** file to ***apache***.

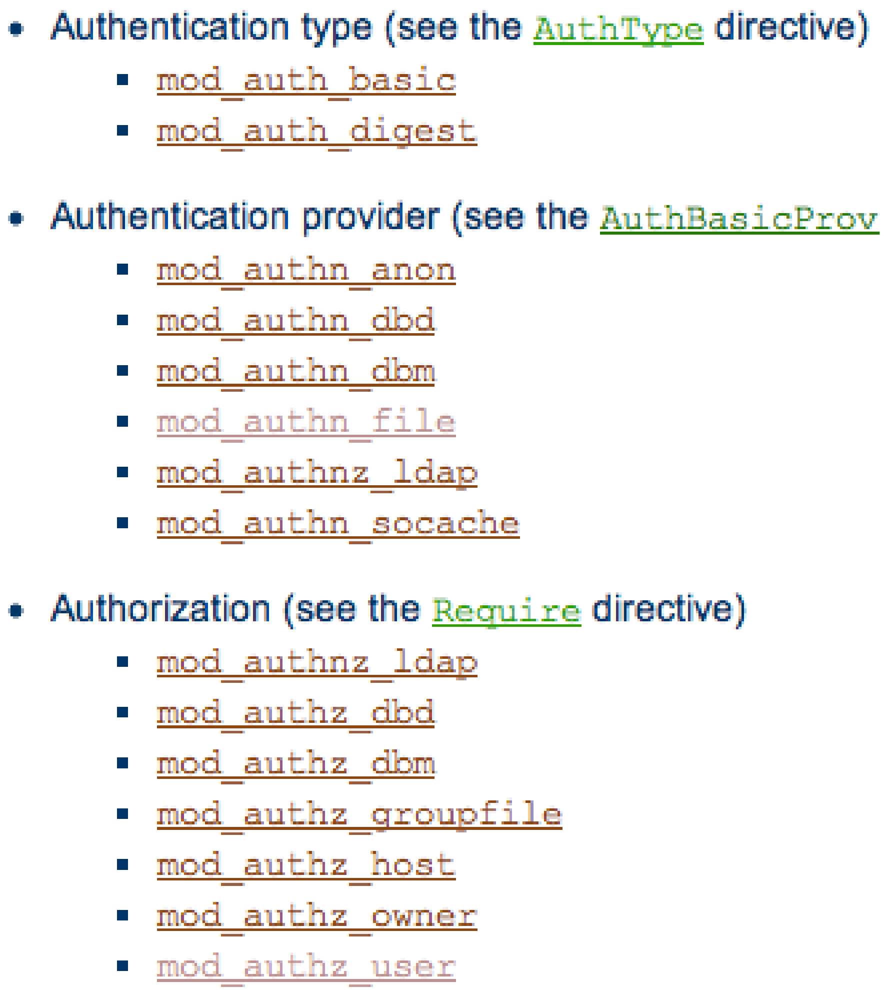
# chown apache:apache /usr/local/apache2/.htdigest

Remove read permissions of others for the ***.htdigest*** file.

chmod o-r /usr/local/apache2/.htdigest

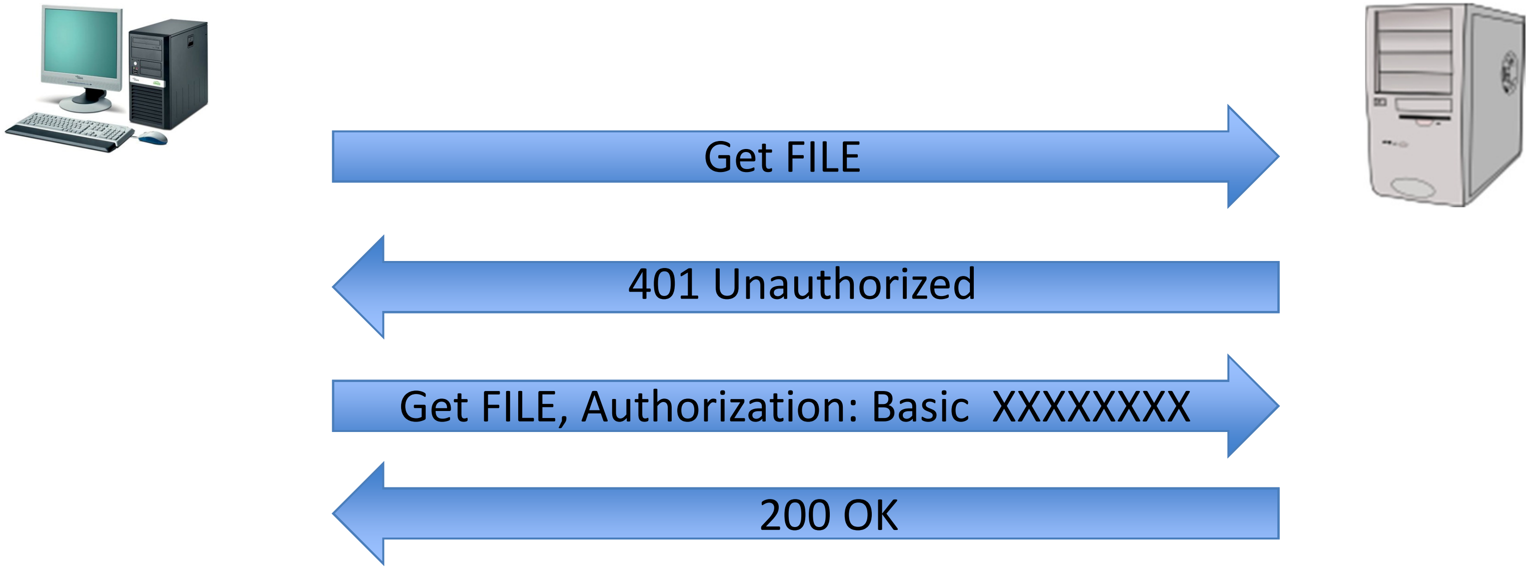
What we did?

- Authentication
  - process by which you verify that someone is who they claim he is
- Authorization
  - someone is allowed to be where they want to go, or to have information that he wants to have



- Authentication type (see the AuthType directive)
  - mod_auth_basic
  - mod_auth_digest
- Authentication provider (see the AuthBasicProv
  - mod_authn_anon
  - mod_authn_dbd
  - mod_authn_dbm
  - mod_authn_file
  - mod_authnz_ldap
  - mod_authn_socache
- Authorization (see the Require directive)
  - mod_authnz_ldap
  - mod_authz_dbd
  - mod_authz_dbm
  - mod_authz_groupfile
  - mod_authz_host
  - mod_authz_owner
  - mod_authz_user

HTTP Basic Authentication

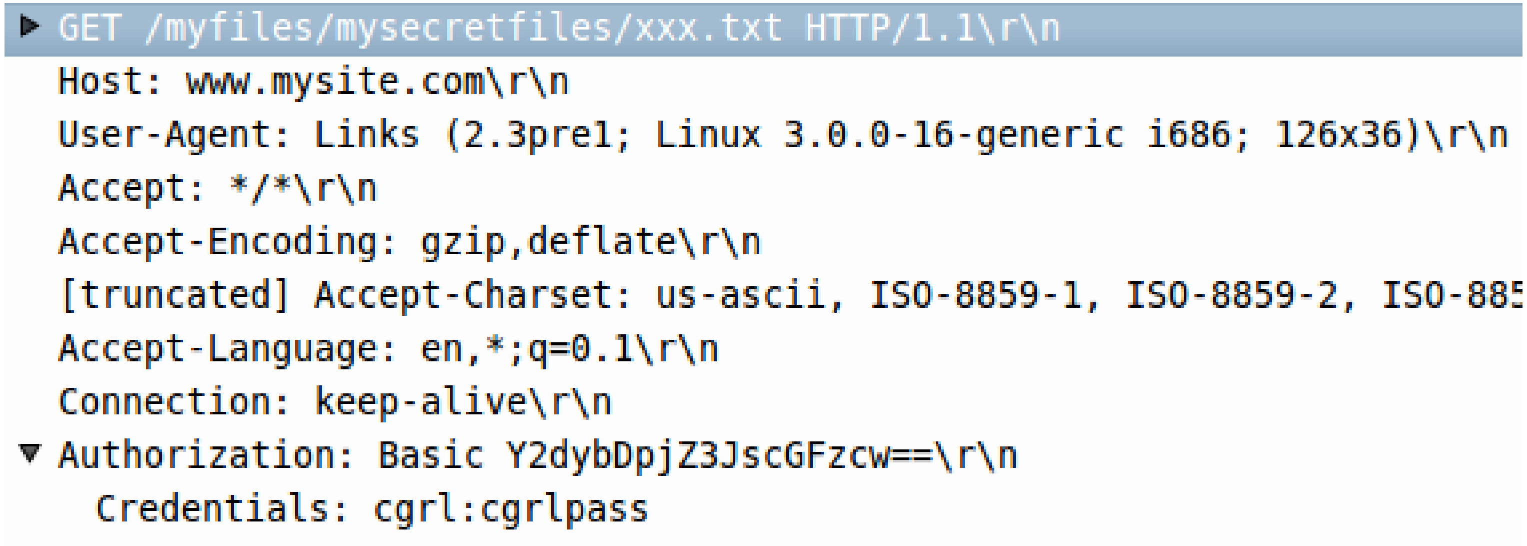


Get FILE

401 Unauthorized

Get FILE, Authorization: Basic XXXXXXXX

200 OK



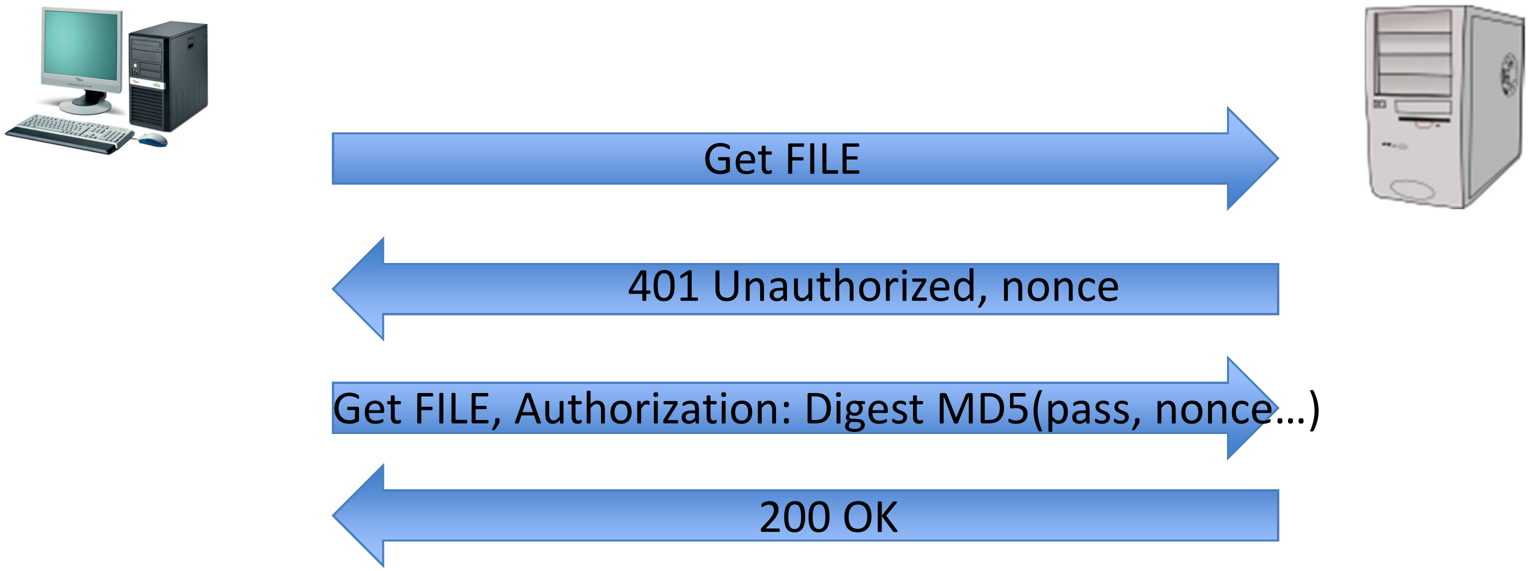
```
▶ GET /myfiles/mysecretfiles/xxx.txt HTTP/1.1\r\n
  Host: www.mysite.com\r\n
  User-Agent: Links (2.3pre1; Linux 3.0.0-16-generic i686; 126x36)\r\n
  Accept: */*\r\n
  Accept-Encoding: gzip,deflate\r\n
  [truncated] Accept-Charset: us-ascii, ISO-8859-1, ISO-8859-2, ISO-885
  Accept-Language: en,*;q=0.1\r\n
  Connection: keep-alive\r\n
▼ Authorization: Basic Y2dybDpjZ3JscGFzcw==\r\n
    Credentials: cgrl:cgrlpass
```

Credentials??

- Client sends passwords in clear text. Maybe ok on HTTPS….
- Let's switch to digest:
  - htdigest –c *PASSWORDFILE* **REALM** *USERNAME*
  - *AuthType Digest*
  - *AuthName REALM*
  - *AuthDigestProvider file*
  - *AuthDigestDomain /*
  - Then update these directives in .htaccess (or <Directory>)
- Realm is the domain of the host performing authentication. I.e., users@example.com or "Private Area".
- Now avoid going sending passwords "in clear". Nonce avoids replay attacks.

- Anyways, access is normally done with login forms and sessions, using server side programming…

HTTP Digest Authentication

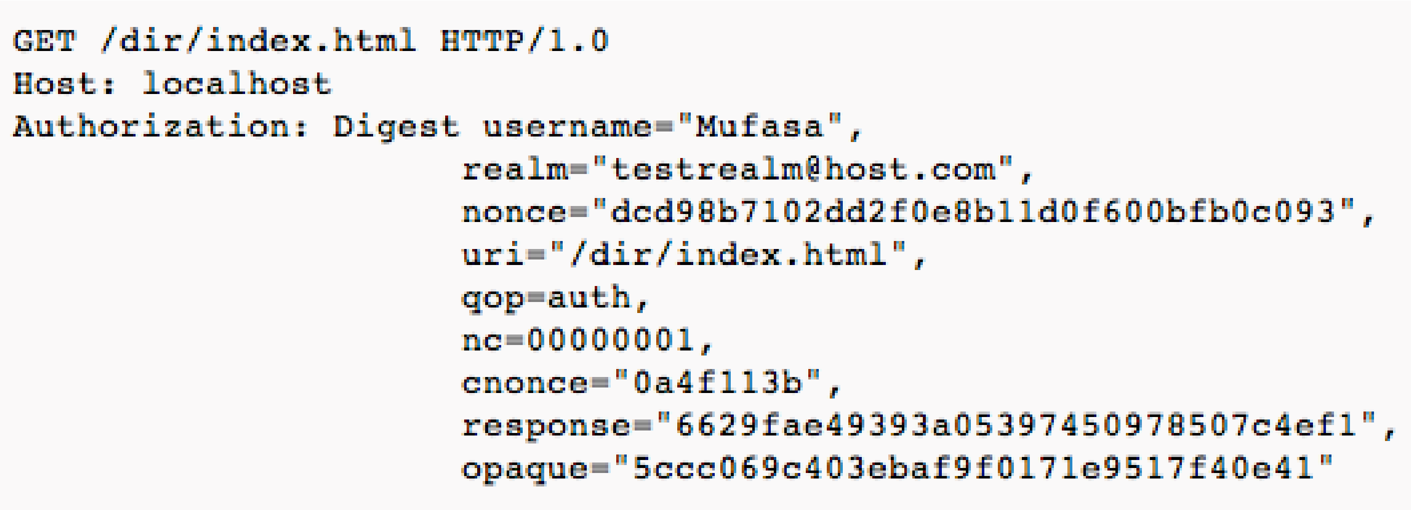


Get FILE

401 Unauthorized, nonce

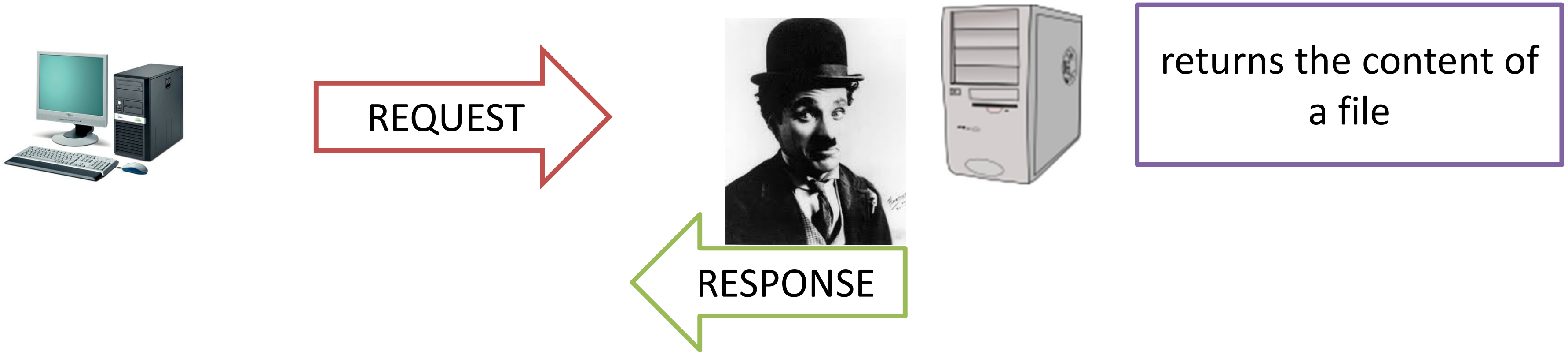
Get FILE, Authorization: Digest MD5(pass, nonce…)

200 OK



```
GET /dir/index.html HTTP/1.0
Host: localhost
Authorization: Digest username="Mufasa",
                     realm="testrealm@host.com",
                     nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
                     uri="/dir/index.html",
                     qop=auth,
                     nc=00000001,
                     cnonce="0a4f113b",
                     response="6629fae49393a05397450978507c4ef1",
                     opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

Static web pages

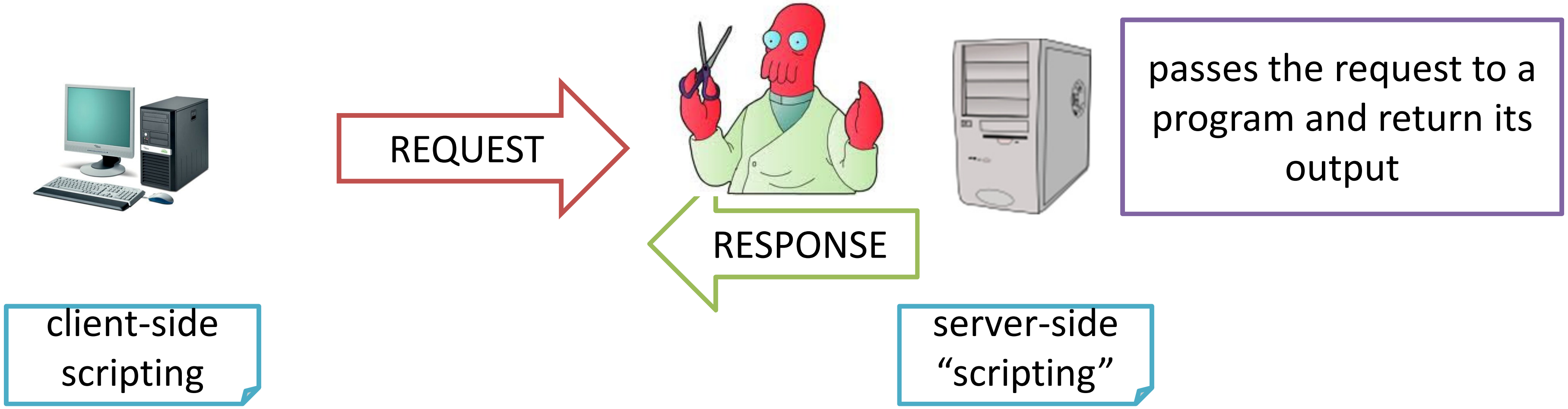


REQUEST

RESPONSE

returns the content of a file

Dynamic web pages



REQUEST

RESPONSE

passes the request to a program and return its output

client-side scripting

server-side “scripting”

Client-side scripting languages

- javascript
- actionscript

YOU CAN NOT DO WHATEVER YOU WANT

Server-side “scripting” languagues

- C/C++
- bash (!)
- Perl
- ASP
- PHP
- Java
- Python
- Lua
- Ruby
- Javascript (!!)
- …

YOU DO WHATEVER YOU WANT

rfc 3875 (2004)
used since 1991

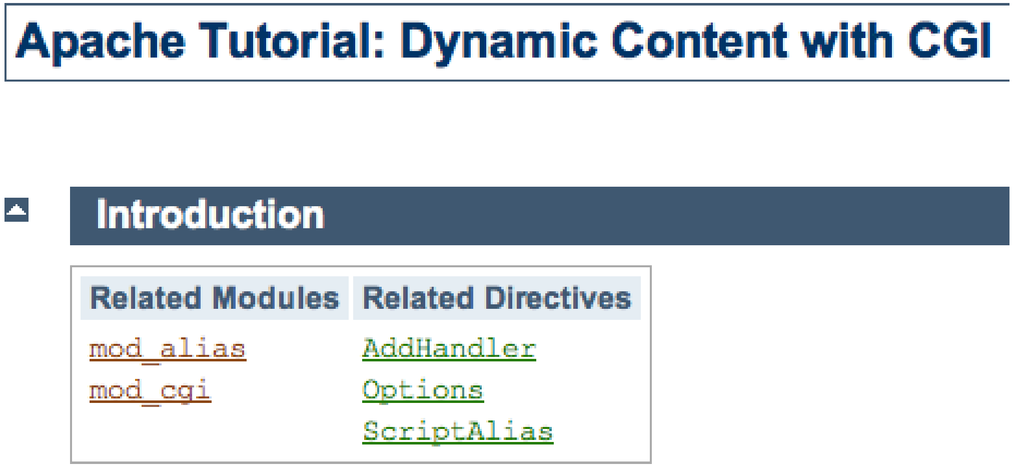
mod_cgi

Common Gateway Interface

- Standard way to delegate the generation of web pages to executable files
- processes isolated from the core Web server

- Check the lab

  for an example!
  - www.cgrl.edu/cgi-bin/env.cgi
  - www.cgrl.edu/cgi-bin/interact.cgi

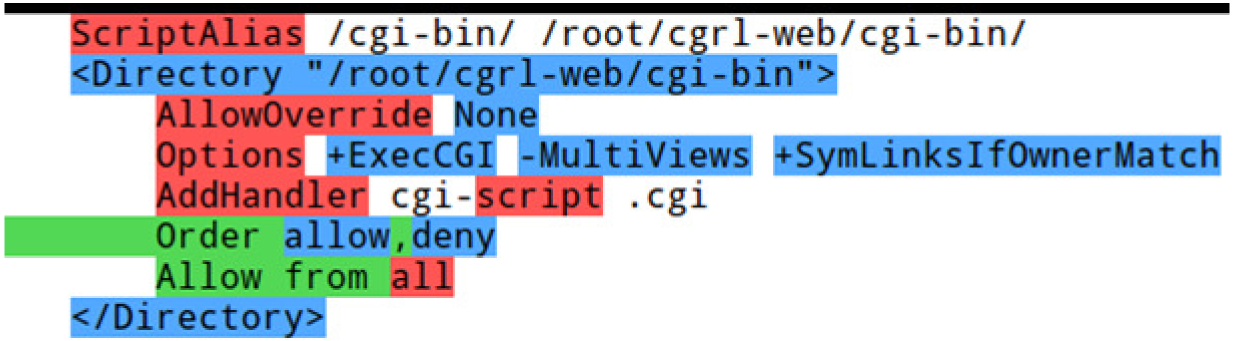


Apache Tutorial: Dynamic Content with CGI

Introduction

Related Modules	Related Directives
mod_alias	AddHandler
mod_cgi	Options
	ScriptAlias

http://httpd.apache.org/docs/2.0/howto/cgi.html



```
ScriptAlias /cgi-bin/ /root/cgrl-web/cgi-bin/
<Directory "/root/cgrl-web/cgi-bin">
    AllowOverride None
    Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
    AddHandler cgi-script .cgi
    Order allow,deny
    Allow from all
</Directory>
```

mod_fcgid

FastCGI

- *CGI: every time you access to a page, you call a program whose output generate the HTTP response*
  - *Launching/Quitting one program per request could cost a lot!*
- *mod_fcgid starts a sufficient number instances of the program to handle concurrent requests, and these programs remain running to handle further incoming requests.*
  - *Significantly faster!*

mod_include

Server Side Include

- They are “directives that are placed in HTML pages, and evaluated on the server while the pages are being served.”

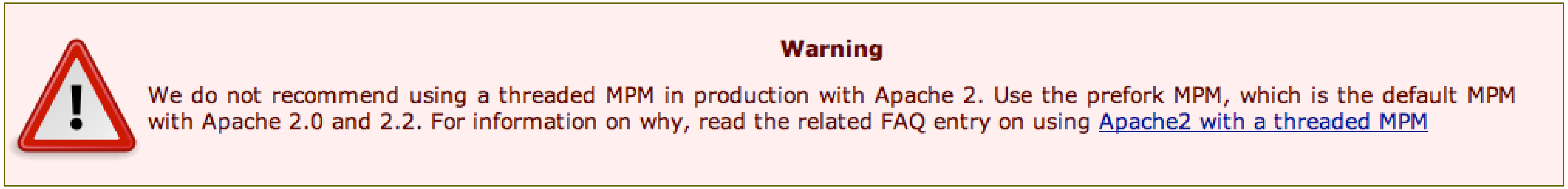
```
Options +Includes

and

AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
```

common directives	examples
include	<!--#include virtual="header.html" -->
exec	<!--#exec cgi="/cgi-bin/foo.cgi" --> <!--#exec cmd="ls -l" -->
echo	<!--#echo var="REMOTE_ADDR" -->
if, elif, else, endif	(control directives)

Installed on more than 20 million Web sites and 1 million web server!

PHP

used by:
wordpress, joomla
facebook, flickr
and many more!

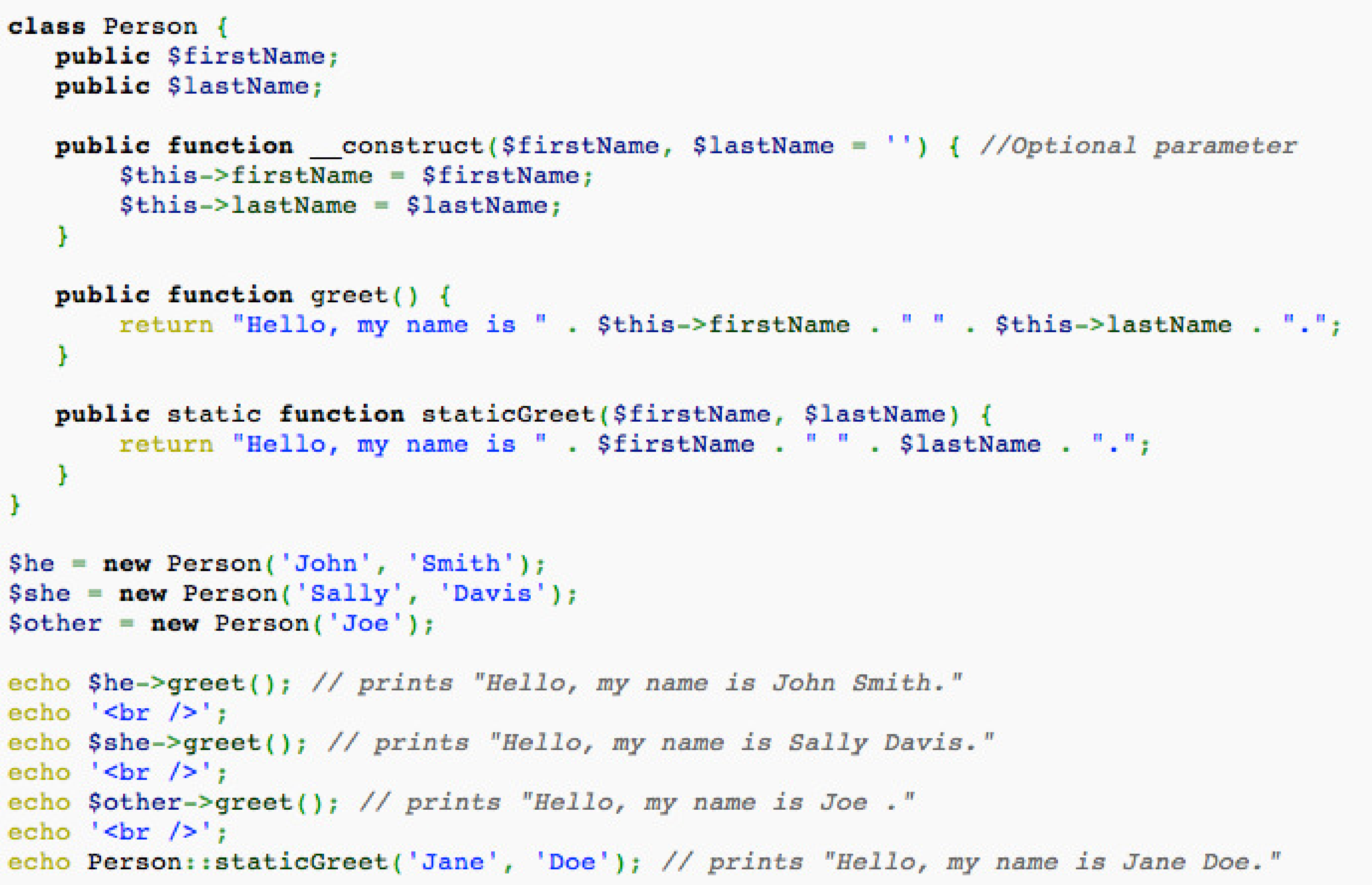


Warning

We do not recommend using a threaded MPM in production with Apache 2. Use the prefork MPM, which is the default MPM with Apache 2.0 and 2.2. For information on why, read the related FAQ entry on using Apache2 with a threaded MPM

```
LoadModule php5_module modules/libphp5.so


<FilesMatch \.php$>
   SetHandler application/x-httpd-php
</FilesMatch>
```

A "handler" is an internal Apache representation of the action to be performed when a file is called



```php
class Person {
   public $firstName;
   public $lastName;

   public function __construct($firstName, $lastName = '') { //Optional parameter
       $this->firstName = $firstName;
       $this->lastName = $lastName;
   }

   public function greet() {
       return "Hello, my name is " . $this->firstName . " " . $this->lastName . ".";
   }

   public static function staticGreet($firstName, $lastName) {
       return "Hello, my name is " . $firstName . " " . $lastName . ".";
   }
}

$he = new Person('John', 'Smith');
$she = new Person('Sally', 'Davis');
$other = new Person('Joe');

echo $he->greet(); // prints "Hello, my name is John Smith."
echo '<br />';
echo $she->greet(); // prints "Hello, my name is Sally Davis."
echo '<br />';
echo $other->greet(); // prints "Hello, my name is Joe ."
echo '<br />';
echo Person::staticGreet('Jane', 'Doe'); // prints "Hello, my name is Jane Doe."
```

source: http://en.wikipedia.org/wiki/PHP#cite_note-54

Model View Controller (MVC) frameworks

mod_passenger (aka mod_rails)

```
LoadModule passenger_module ...
PassengerRoot ...
PassengerRuby ...
```

mod_wsgi

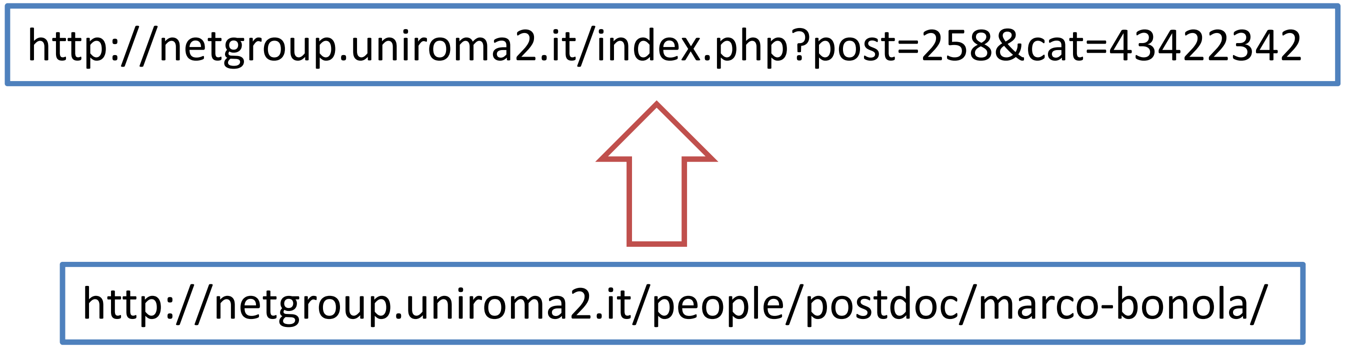
WSGI: python standard to communicate with web server

```
WSGIScriptAlias /
/path/to/mysite.com/mysite/wsgi.py
```

not necessarily in .htaccess

Mod Rewrite

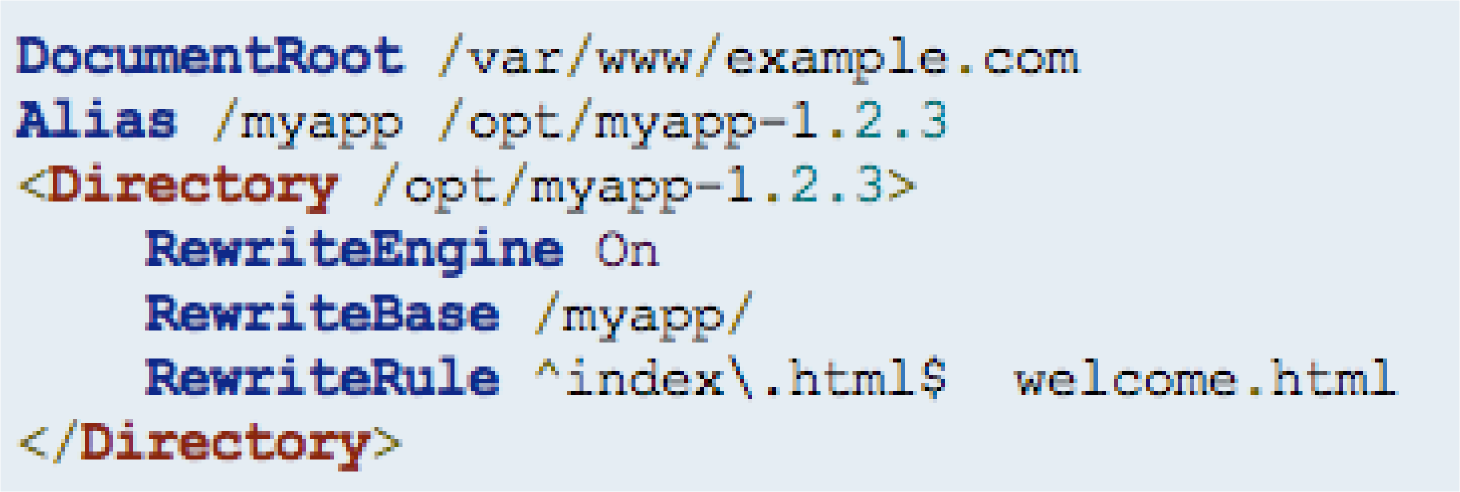
- Goal: rewrite an URL to another
- Why? typical: user friendly URL
- How?
  - LoadModule rewrite_module modules/mod_rewrite.so
  - AddModule mod_rewrite.c
  - RewriteEngine on



http://netgroup.uniroma2.it/index.php?post=258&cat=43422342

http://netgroup.uniroma2.it/people/postdoc/marco-bonola/

Mod Rewrite (example)



```
DocumentRoot /var/www/example.com
Alias /myapp /opt/myapp-1.2.3
<Directory /opt/myapp-1.2.3>
    RewriteEngine On
    RewriteBase /myapp/
    RewriteRule ^index\.html$  welcome.html
</Directory>
```

- Change /myapp/index.html with welcome.html

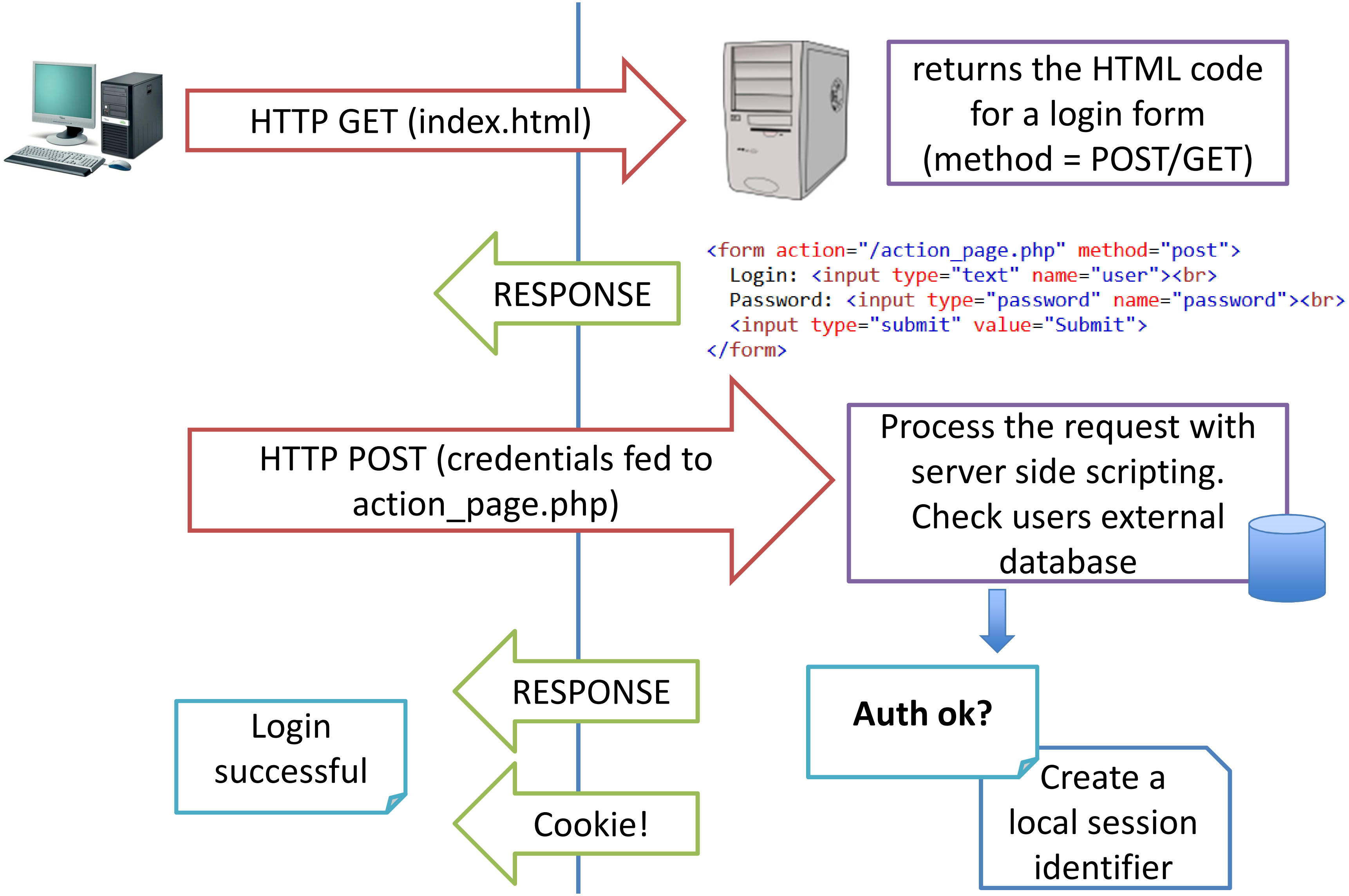
Mod Rewrite

RewriteRule PATTERN SUBSTITUTION [FLAGS]	Define a rule: if find a pattern in the URL, then substitute. Flags: send headers to browsers (e.g. 401)
RewriteCond %{HTTP_USER_AGENT} ^Mozilla.*	Apply the next rule only in this condition is true. Rules are applied only if ALL the previous conditions are true

Examples of rules:

```
RewriteRule ^/shortcut$ /complicated/and/way/too/long/url/here
RewriteRule /products/([0-9]+) /siteengine/products.php?id=$1
RewriteRule  ^/products/([0-9]+),([ad]*),([0-9]{0,3}),([0-9]*),([0-9]*$)
      /test/index.php?id=$1&sort=$2&order=$3&start=$4
```

“non-HTTP” authentication



HTTP GET (index.html)

returns the HTML code for a login form (method = POST/GET)

RESPONSE

```
<form action="/action_page.php" method="post">
  Login: <input type="text" name="user"><br>
  Password: <input type="password" name="password"><br>
  <input type="submit" value="Submit">
</form>
```

HTTP POST (credentials fed to action_page.php)

Process the request with server side scripting. Check users external database

Auth ok?

Create a local session identifier

RESPONSE

Cookie!

Login successful

Note: Any forms involving sensitive information like passwords should be served over HTTPS.

Client side scripting

- Actions performed by the web browser. Very limited for security reasons (ie. no access to local disk)
- HTML5 allows some more degree of actions
- In practice **all** current web HTML pages require client side scripting to perform some dynamic action:
  - Send asynchronous message to servers (AJAX)
    - Display values on a chart in realtime
    - Fast-forward of a video
  - https://www.w3schools.com/js/tryit.asp?filename=tryjs_timing_clock
  - We may MD5 the password before sending it… **warning:** the hash becomes the password! Needs some “salt” (nonce)

Server side

```
login.php
<script>
function hash_pswd_nonce()  { ….
   […]
return true
} </script>
<?php
session_start();
//Check nonce against session
if(isset($_POST) && $_POST["nonce"] === $_SESSION["csrf"]){
   //use nonce+password for MD5 check
   //redirect to private page
}

//generate new nonce for form
$_SESSION["csrf"] = uniqid(mt_rand(),true);
?>
<form method="post" action="login.php" onsubmit= "return hash_pswd_nonce();" >
   <input type="hidden" name="nonce" value="<?php echo $_SESSION['csrf']; ?>"/>
   Login: <input type="text" name="user"><br>
   Password: <input type="password" name="password"><br>
  <input type="submit" value="Submit">
</form>
```

Client side

Avoid **Cross-site request forgery (CSRF)**