Administration of Operating Systems

Apache

Chapter 26

December 07, 2011

Intelligent Systems Lab

Apache

- World's most commonly used web server
  - 48% of all web sites
  - 66% among the million most active ones
- Available for many operating systems
  - maintained by Apache Software Foundation
- Employs modular architecture
  - binary only includes the basic functionality
  - most features in Dynamic Shared Objects
- Current release is 2.2.21 from September 2011
  - 2.0.64 from May 2011
  - 1.3.42 from January 2008
  - 2.3.15 beta from November 2011

HTTP

- Hyper-Text Transfer Protocol (version 1.1)
  - stateless protocol
- RFC 1945, 2616, 2617, ...
- World Wide Web's application-level protocol
  - specifies how data should be exchanged between web browser and web server
  - ... and other applications
- Some terminology
  - a *web page* is a set of *objects*
  - an *object* is, basically, a file
  - one of those files is a *base HTML document*
  - objects can contain *references* to other objects

Uniform Resource Identifier

- Uniform Resource Locator
  - a way of locating things in the network
- http://slawek:***@hh.se:80/index.html?lang=Eng#Staff
  - protocol name (`http`)
  - optional username (`slawek`)
  - optional password (`*****`)
  - host name (`hh.se`)
  - optional port number (`80`)
  - path (`index.html`)
  - optional query (`lang=Eng`)
  - optional fragment (`Staff`)
- URN: *name* resource without specifying location

Communication Flow

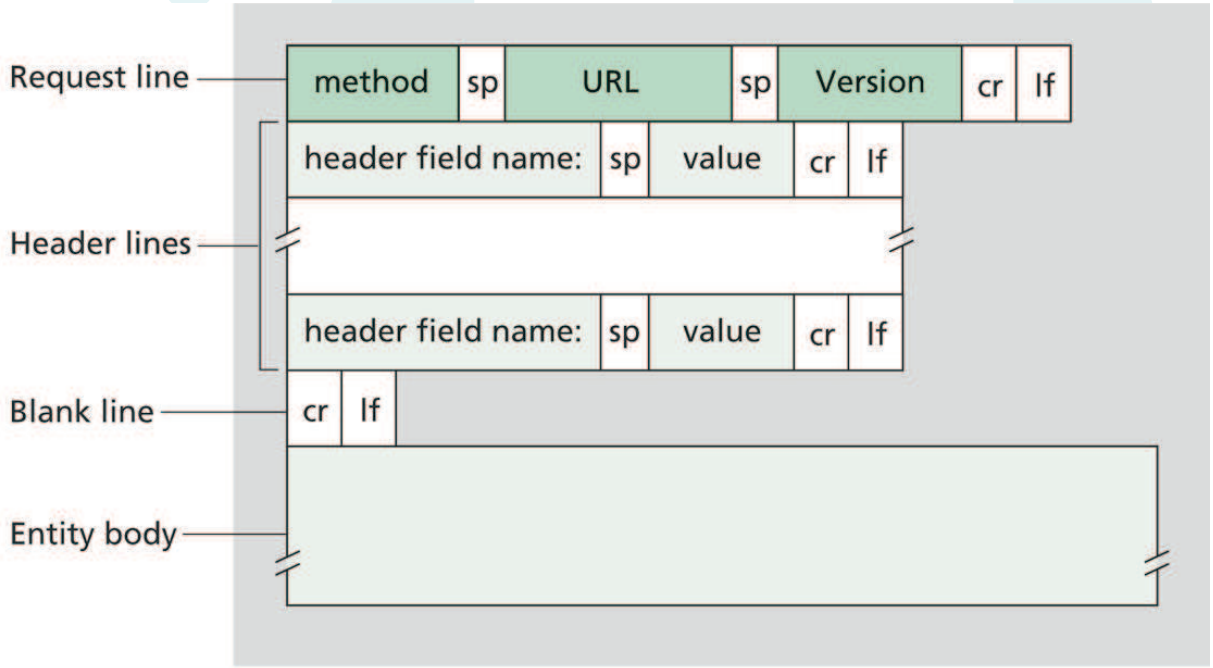
- Getting a web page
  - client opens TCP connection to the server
  - web server acknowledges the connection
  - client sends HTTP request
    - server acknowledges each packet
  - server sends HTTP response
    - requested file or some error message
    - client acknowledges each packet
  - server sends connection close request
  - client acknowledges close request
- Which means $\geq$8 TCP packets
  - for each object

Typical Usage

- Clients wants to get a web page
  - containing some text and, say, 10 images
  - ... or 1000 images
- The simplest idea is to create a separate connection for each of those images
  - *non-persistent connections*
  - it can do it either sequentially or in parallel
- Alternatively, it might use one, single TCP connection for several images
  - *persistent connections*
  - no need to open & close several connections
  - but requires a way to separate data streams

HTTP Request Message

```
GET /nethack/nethack-343-win.zip HTTP/1.1
HOST: www.nethack.org
Connection:  close
User-Agent:  hand.crafted.request
Accept-language:  eng, pl;q=0.9, se;q=0.1
```



HTTP Request Message

- Request line
  - method field GET, POST, HEAD...
  - URL field
  - version field
- Header lines
  - Host
  - User-Agent
  - Accept-language
  - Authorization
  - Referer
  - ...
- Data

HTTP Request Method

- GET:
  - probably the most common one
  - requests an object
  - can pass additional information in the URL
  - `http://www.google.com/search?client=opera&q=test&sourceid=opera`
  - *safe* and *idempotent*
- HEAD:
  - similar to GET
  - mainly for testing
  - server sends HTTP reply
  - does not send the object
  - *safe* and *idempotent*

HTTP Request Method

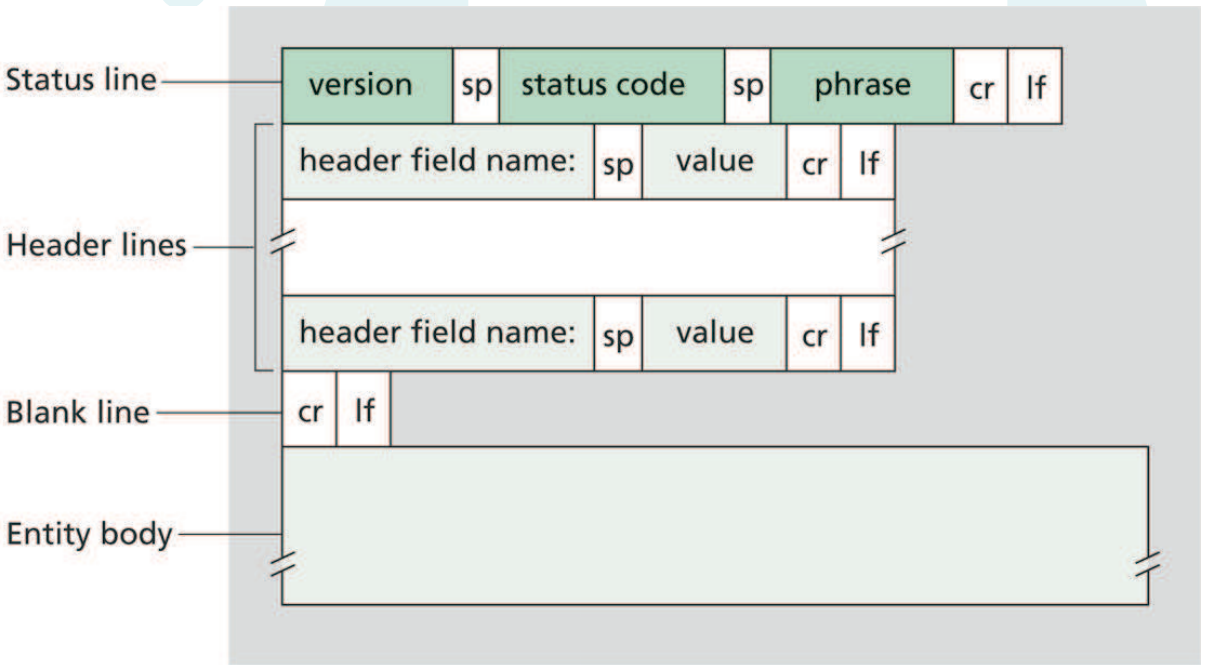
- POST:
  - when data needs to be sent to server
  - in the “entity body” part of the message
  - recommended when some action on the server side is to be taken
  - no specification of what this action may be
  - *not safe* and *not idempotent*

HTTP Request Method

- PUT:
  - places the “entity body” as a resource pointed to by the URL
  - *not safe* but *idempotent*
- DELETE:
  - removes resource pointed to by URL
  - *not safe* but *idempotent*
- OPTIONS
- TRACE
- CONNECT

HTTP Response Message

```
HTTP/1.1 200 OK
Date:  Mon, 12 September 2005, 13:45
Server:  Apache/2.0.54 (Unix)
Last-Modified:  Mon, 12 September 2005, 12:55
Content-Length:  1234
Content-Type:  text/html
```



Intelligent Systems Lab

HTTP Response Message

- Status line
  - version field
  - status code
  - status message
- Header lines
  - Date
  - Server
  - Last-Modified
  - Content-Length
  - Content-Type
  - ...
- Data

HTTP Status codes

200 — OK
202 — Accepted
204 — No Content
300 — Multiple Choices
301 — Moved Permanently
307 — Temporary Redirect
400 — Bad Request
401 — Unauthorized
403 — Forbidden
404 — Not Found
406 — Not Acceptable
500 — Internal Server Error
501 — Not Implemented
503 — Service Unavailable

Cookies

- HTTP is *stateless* protocol
- Server does not remember anything about the client in between requests
- All information necessary for locating the web page and presenting it is explicit in the request
- However, it is often desirable to recognise users
  - customisation
  - authentication
  - session history
  - advertisement
  - context-sensitive help
  - ...

Cookies — the Technology

- Protocol side is very simple
  - “Cookie” header in HTTP request
  - “Set-Cookie” header in HTTP response
- Cookie jar managed by the browser
  - simple storage
- Cookie management on server’s side
  - identification cookies
  - customisation cookies
- Personalising web page according to cookies
  - designer’s job
  - dynamic web page content

Cookies — the Privacy

- Somewhat controversial
- Any server can track any of its users
- *Every* request can send and retrieve a cookie
  - including requests for inline images
  - large advertisement companies can track users across multiple servers
- Browsers can provide some protection
  - refuse cookies from domains other than the web currently being viewed
  - cookie expiration time

HTTP Content

- Web pages — HTML files
- Images, audio, video, ...
  - need a way to specify type of data
  - MIME type: text/html, image/png, video/mpeg, application/binary
- Originally, each object was a disk file
- Nowadays, dynamic content is widespread
  - generated by web server on the fly
  - depending on details of each request
    - cookies
    - data provided in the request
    - ip address, browser type, language, ...

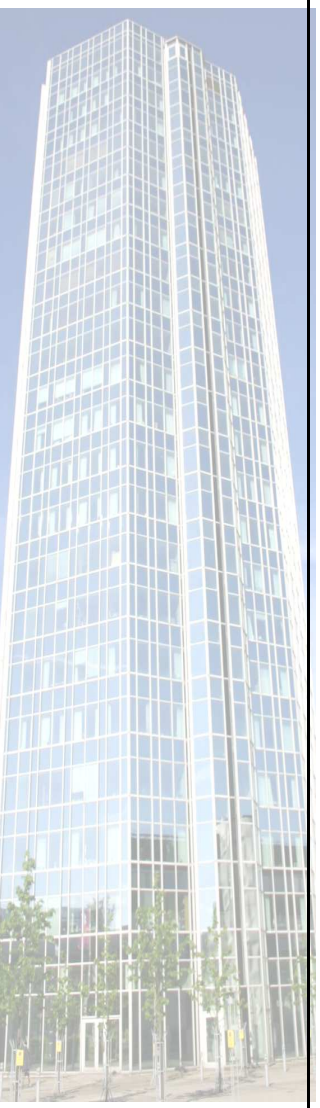
Ubuntu Apache

- Heavily modified configuration
  - different from the default ASF one
- Split into multiple files
  - easy to update
- `apache2.conf`
  - default options
- `httpd.conf`
  - user's customisations
- `sites-available/default`
  - options for each web site
- `mods-available`
  - configuration for various modules

httpd.conf

```
ServerRoot "/usr/local/apache"
DocumentRoot "/var/www"

Listen 80
#Listen 12.34.56.78:80

LoadModule auth_basic_module
                modules/mod_auth_basic.so

ServerAdmin admin@slawek.net
ServerName www.slawek.net:80

User www-data
Group www-data
```

Filesystem



```
<Directory />
    Options FollowSymLinks
    AllowOverride None
    Order deny,allow
    Deny from all
</Directory>

<Directory /var/www/slawek>
    Options Indexes FollowSymLinks Includes
    AllowOverride All
    Order allow,deny
    Allow from all
</Directory>
```

Options

- *ExecCGI* — CGI scripts
- *FollowSymLinks* — symbolic links
- *SymLinksIfOwnerMatch*
- *Includes* — *server-side includes*
- *IncludesNOEXEC* — SSI without `#exec`
- *Indexes* — directory listings
- *MultiViews* — *content negotiation*
- *All* — all (except for *MultiViews*)

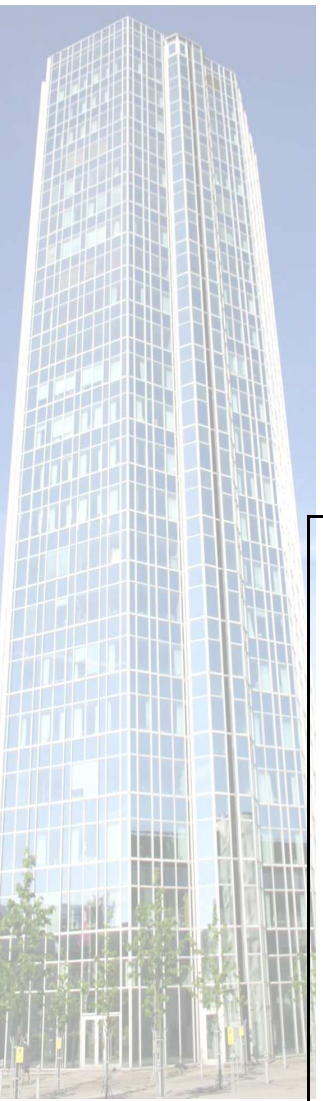
```
Syntax:         Options [+|-]option [[+|-]option]...
Default: Options All
Context: config, vhost, directory, .htaccess
```

AllowOverride

- Limits applicability of `.htaccess` files
- None — nothing can be changed
- AuthConfig — authorisation settings
- FileInfo — types, metadata, *actions* and *rewrites*
- Indexes — directory listing options
- Limit — limiting resource access
- `Options[=Option,+Option,-Option]`

```
Syntax:          AllowOverride All|None|directive...
Default: AllowOverride All
Context: directory
```

Allow/Deny



- *Order* parameter defines the order in which *Allow* and *Deny* directives will be checked
- If a particular request matches both of them, then the *last* directive wins
  - same if neither matches

```
Syntax:          Order ordering
Default: Order Deny,Allow
Context: directory, .htaccess

Syntax:          Allow from all|host|
                          env=[!]env-variable ...
Context: directory, .htaccess
```

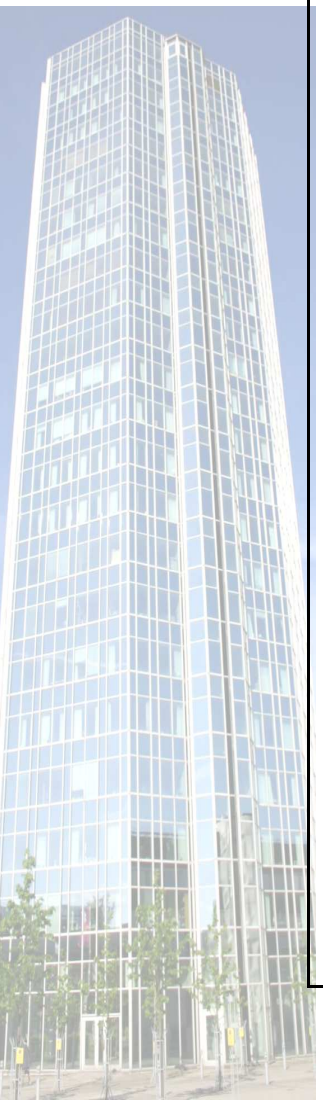
Allow/Deny

```
Order Deny,Allow
Deny from all
Allow from apache.org
```

```
Order Allow,Deny
Allow from apache.org
Deny from foo.apache.org
```

```
Order Deny,Allow
Allow from apache.org
Deny from foo.apache.org
```

Allow/Deny

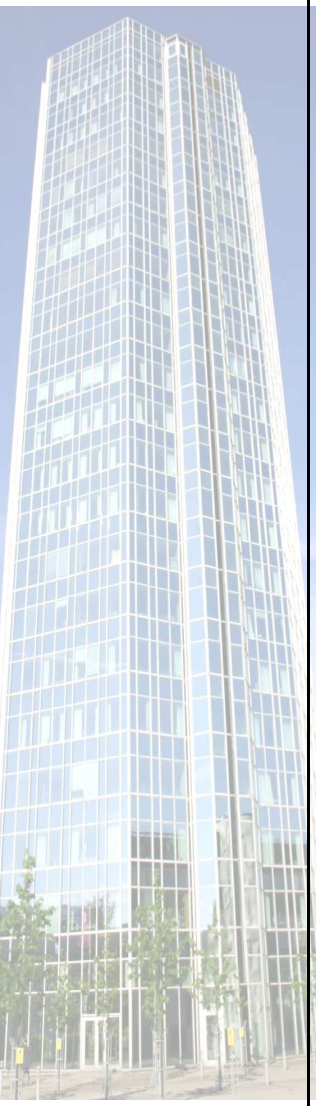


```
SetEnvIf User-Agent
              ^KnockKnock/2\.0
              let_me_in

<Directory /www>
  Order Deny,Allow
  Deny from all
  Allow from env=let_me_in
  Allow from 10.1.0.0/255.255.0.0
  Allow from 10.1.0.0/16
</Directory>
```

.htaccess

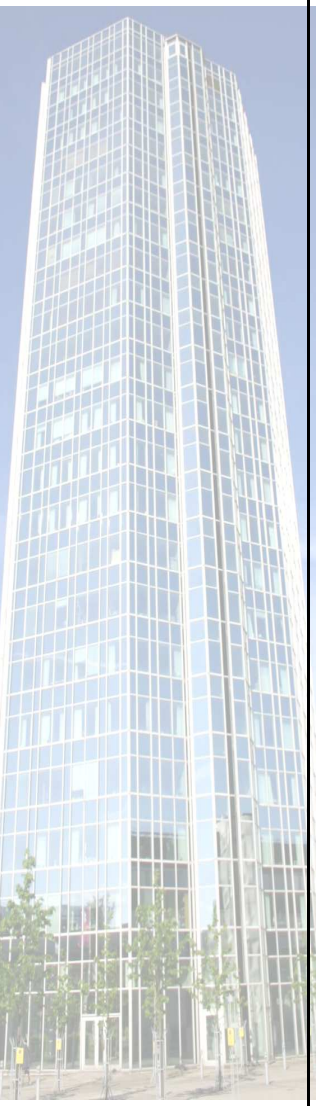
- Move some configuration to the directory level
  - distributed administration
  - can be done by regular users
  - does not require changing `httpd.conf`
- Still, operates within globally-specified limits
  - `AllowOverride` directive
- Apply to the directory they are in
  - and all subdirectories
- Same syntax as `httpd.conf`
  - not all options are available

.htpasswd



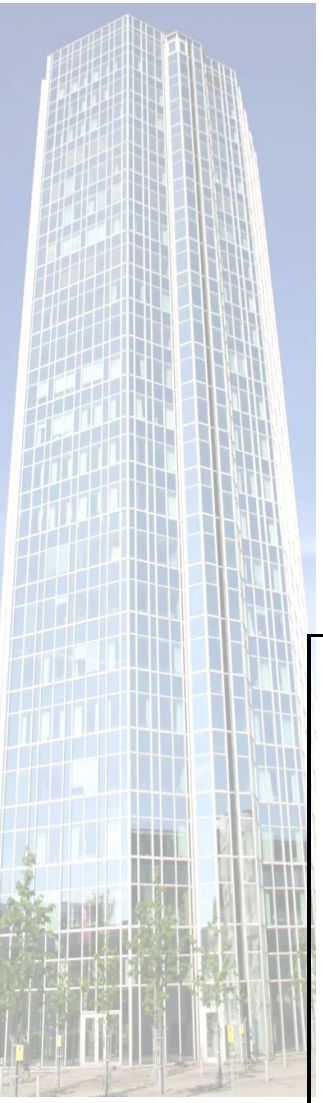
```
.htpasswd
    test1:zSw/6qLAtli3U
    test2:gvnG/pky7S5EU

.htaccess
    AuthName "Secret area"
    AuthType Basic
    AuthUserFile /var/mysite/.htpasswd
    AuthGroupFile /dev/null

    require valid-user
    <Files secret.html>
      require valid-user
    </Files>
```

httpd.conf



```
<DirectoryMatch "/var/www/[0-9]{3}">
</DirectoryMatch>

<Files private.html>
  Order allow,deny
  Deny from all
</Files>

<Directory /var/web/dir>
  <Files private.html>
    Order allow,deny
    Deny from all
  </Files>
</Directory>
```

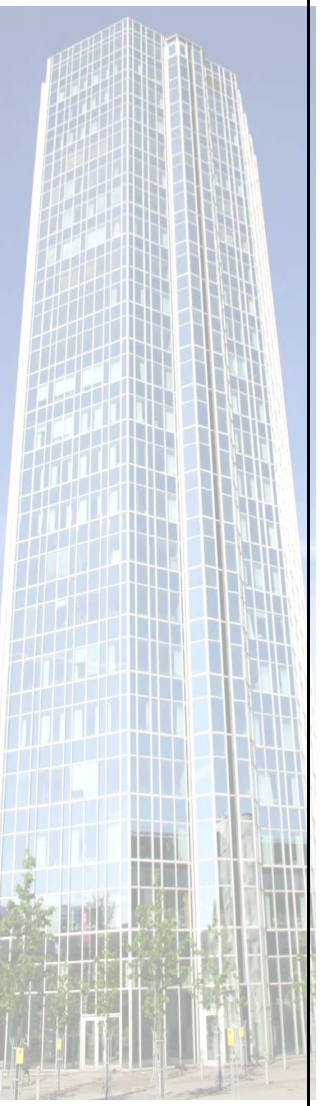
Location



- `Directory` and `Files` directives are applied to specific areas in the *filesystem*
  - not URL addresses
- Apache is often used to serve data that is not related to the filesystem at all
  - *dynamic content*

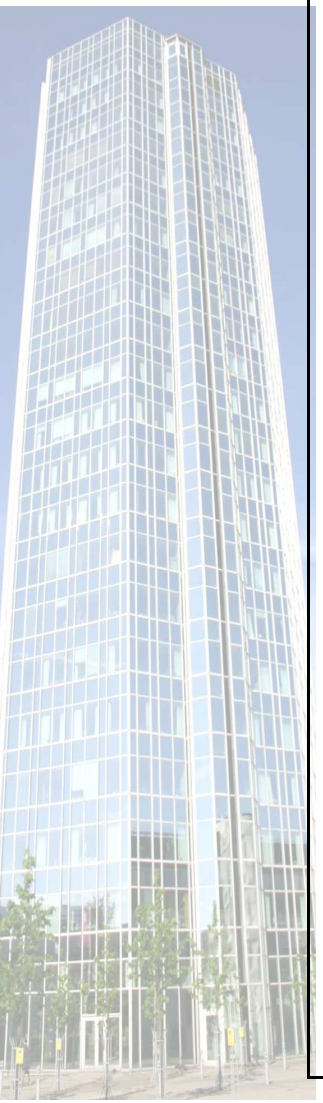
```
<Location /status>
  SetHandler server-status
  Order Deny,Allow
  Deny from all
  Allow from .example.com
</Location>
```

httpd-autoindex.conf



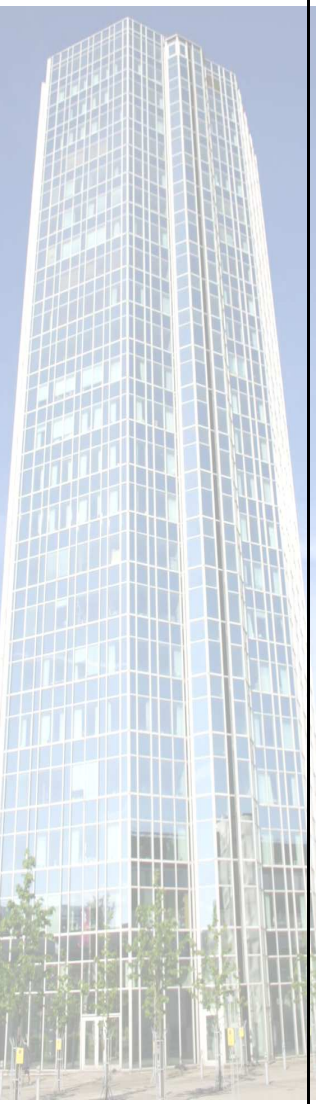
```
<Directory /www>
  Options Indexes
  IndexOptions FancyIndexing
  IndexOptions HTMLTable
  IndexOptions FoldersFirst
  IndexOptions IconsAreLinks
  IndexOptions IgnoreCase
  IndexOptions IgnoreClient
  IndexOptions ScanHTMLTitles
  IndexOptions ShowForbidden
  IndexOptions SuppressColumnSorting
  IndexOptions SuppressSize
  IndexOptions VersionSort
</Directory>
```

httpd-autoindex.conf



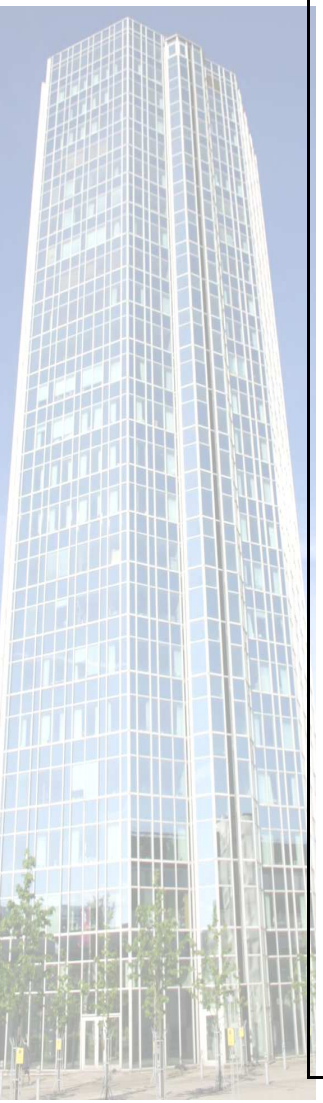
```
ReadmeName README.html
HeaderName HEADER.html

IndexOrderDefault Ascending Name
IndexOrderDefault Descending Size

IndexStyleSheet "/css/style.cs"

IndexIgnore *~ HEADER* README* CVS

IndexHeadInsert "<link rel=\"sitemap\"
                      href=\"/sitemap.html\">"
```

httpd-autoindex.conf



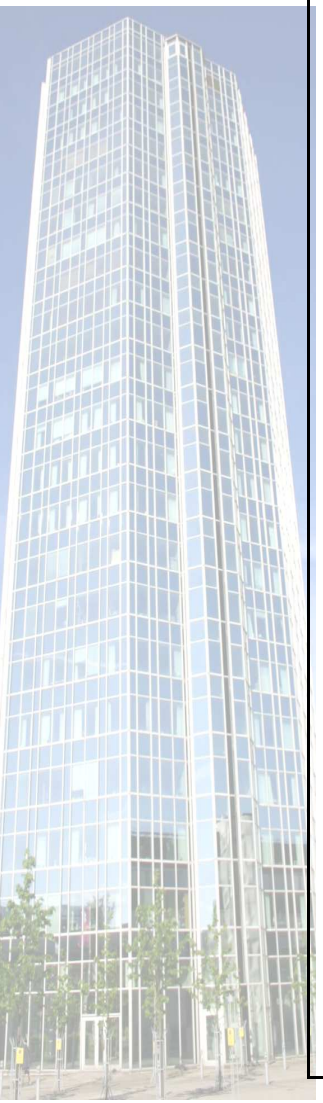
```
AddIconByEncoding (CMP,/icons/compressed.gif)
                  x-compress x-gzip
AddIconByType (TXT,/icons/text.gif) text/*
AddIcon /icons/binary.gif .bin .exe
AddIcon /icons/backup.xbm *~
AddIcon /icons/back.gif ..
AddIcon /icons/hand.right.gif README
AddIcon /icons/folder.gif ^^DIRECTORY^^
AddIcon /icons/blank.gif ^^BLANKICON^^

DefaultIcon /icons/unknown.gif
AddDescription "tar archive" .tar
AddAltByEncoding "gzip archive" x-gzip
```

httpd-status.conf



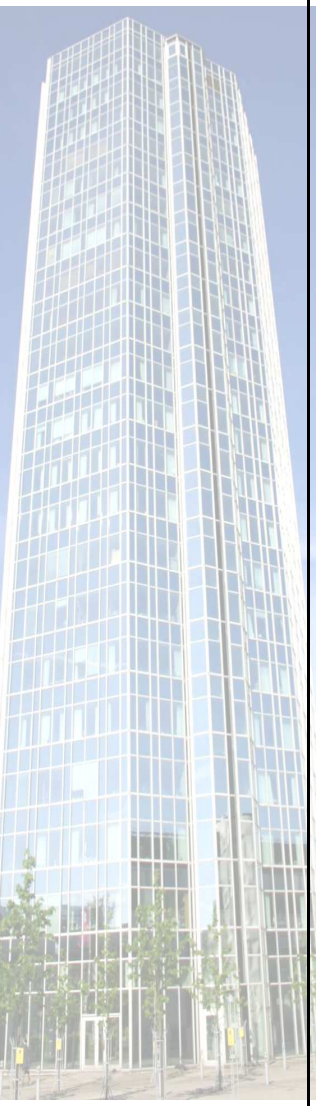
```
<Location /server-status>
    SetHandler server-status
    Order deny,allow
    Deny from all
    Allow from .slawek

    ExtendedStatus On
</Location>

http://www.slawek/server-status?refresh=N

http://www.slawek/server-status?auto
```

httpd-info.conf



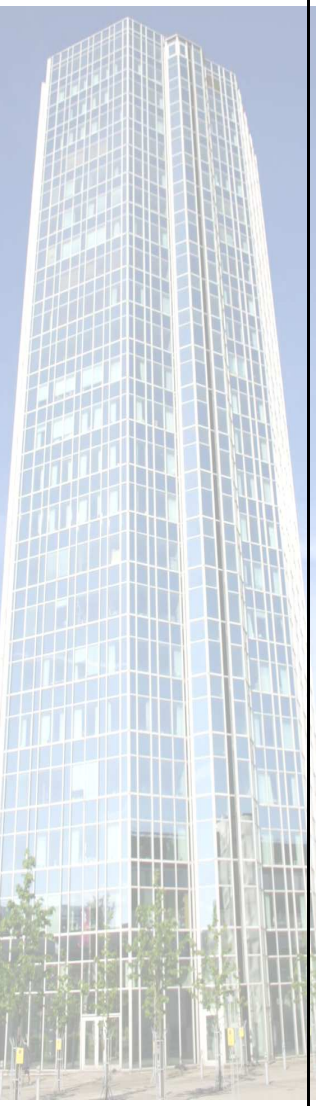
```
<Location /server-info>
    SetHandler server-info
    Order deny,allow
    Deny from all
    Allow from .slawek
</Location>

http://www.slawek/server-info?config

http://www.slawek/server-info?mod_python

http://www.slawek/server-info?server
```

httpd-userdir.conf



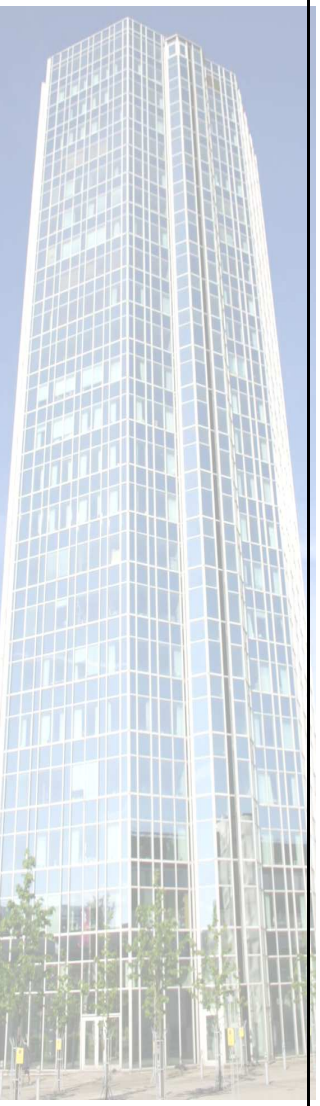
```
http://www.slawek/~slawek/

<IfModule mod_userdir.c>
  UserDir disabled
  UserDir disabled root

  UserDir enabled user1 user2 user3

  UserDir public_html
  UserDir /usr/web
  UserDir /www/*/public
  UserDir http://www.example.com/~*/
</IfModule>
```

httpd-userdir.conf



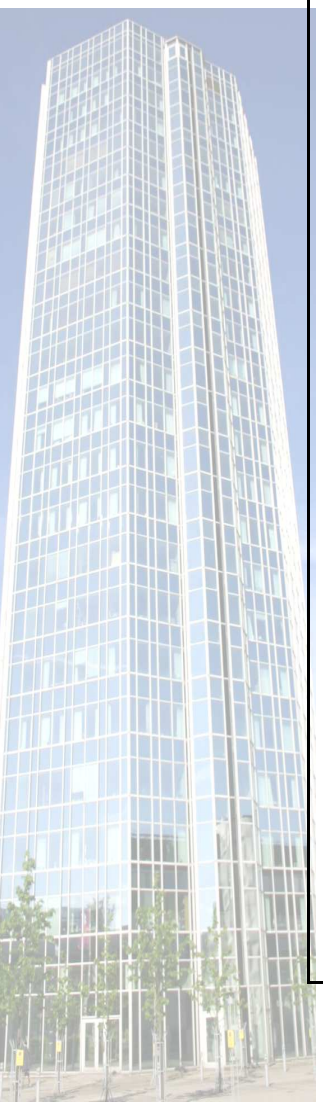
```
<Directory "/home/*/public_html">
    AllowOverride FileInfo AuthConfig Indexes
    Options Indexes IncludesNoExec
                        SymLinksIfOwnerMatch
    <Limit GET POST OPTIONS>
        Order allow,deny
        Allow from all
    </Limit>
    <LimitExcept GET POST OPTIONS>
        Order deny,allow
        Deny from all
    </LimitExcept>
</Directory>
```

httpd-vhosts.conf



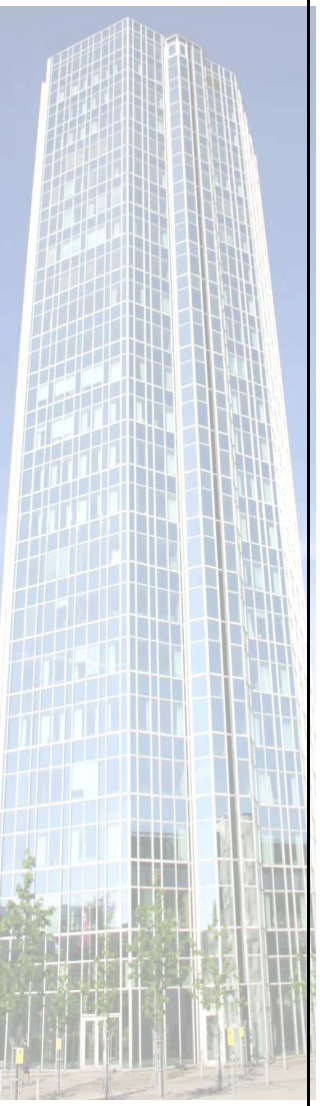
```
NameVirtualHost 11.22.33.44:80
NameVirtualHost *:80
NameVirtualHost *

<VirtualHost *:80>
    ServerAdmin webmaster@dummy-host.net
    DocumentRoot "/www/dummy-host.net"
    ServerName dummy-host.net
    ServerAlias www.dummy-host.net
    ErrorLog "logs/dummy.net-error.log"
    CustomLog "logs/dummy.net-access.log"
                  "%h %l %u %t \"%r\" %>s %b"
</VirtualHost>
```

httpd-vhosts.conf



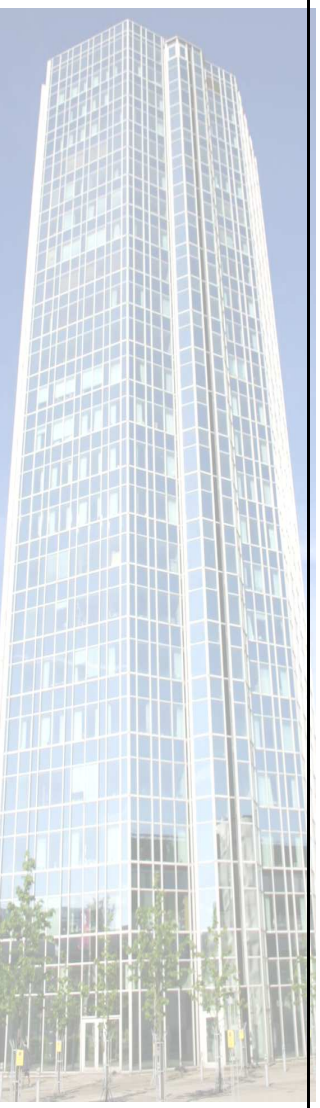
```
<VirtualHost *:80>
    ServerAdmin webmaster@other-host.net
    DocumentRoot "www/other-host.net"
    ServerName other-host.net
    ServerAlias *.other-host.net
    ErrorLog "logs/otherhost.net-error.log"
    LogFormat "%h %l %u %t \"%r\" %>s %b"
                        other-host-log-format
    CustomLog "logs/otherhost.net-access.log"
                        other-host-log-format
</VirtualHost>
```

mod_include.conf



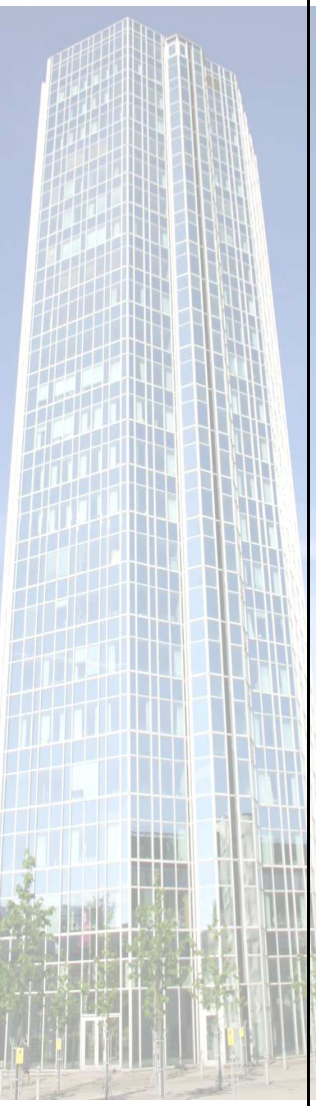
```
AddType text/html .shtml
AddOutputFilter INCLUDES .shtml
Options +Includes

<!--#exec cgi="/cgi-bin/example.cgi" -->
echo, exec, fsize, flastmod, include, set,...

DATE_GMT, DATE_LOCAL,
DOCUMENT_NAME, DOCUMENT_URI, ...

<!--#if expr="test_condition" -->
<!--#elif expr="test_condition" -->
<!--#else -->
<!--#endif -->
```

mod_rewrite



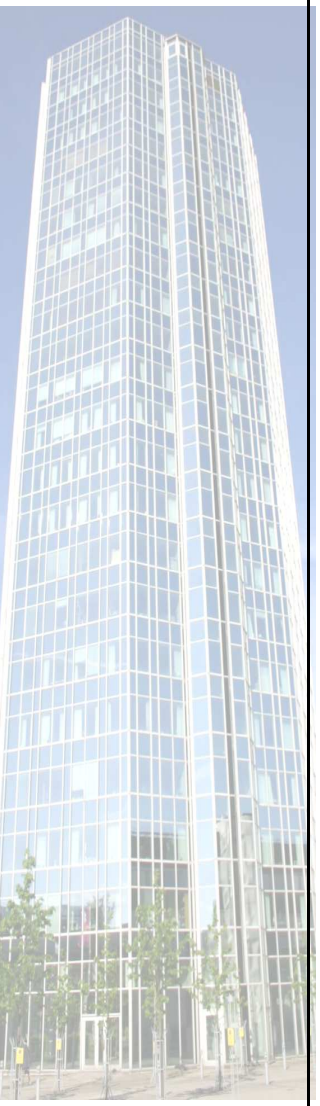
```
RewriteEngine on

RewriteRule  ^/~(.+) http://new.tld/~$1 [R,L]

RewriteRule  ^/$     /about/            [R]

RewriteCond  %{HTTP_USER_AGENT}  ^Mozilla.*
RewriteRule  ^/$    /homepage.max.html  [L]
RewriteCond  %{HTTP_USER_AGENT}  ^Lynx.*
RewriteRule  ^/$    /homepage.min.html  [L]

RewriteCond %{HTTP_HOST} ^[^.]+\.host\.com$
RewriteRule ^(.+)   %{HTTP_HOST}$1      [C]
RewriteRule ^([^.]+)\.host\.com(.*) /usr/$1$2
```

mod_python.conf



```
AddHandler mod_python .py
PythonHandler mod_python.publisher
PythonDebug On

<html>
  Please provide feedback below: <p>
  <form action="form.py/email" method="POST">
  Name:   <input type="text" name="name"><br>
  Email: <input type="text" name="email"><br>
  Comment: <textarea name="comment"
                  rows=4 cols=20></textarea><br>
  <input type="submit">
  </form>
</html>
```

form.py



```
import smtplib
def email(req, name, email, comment):
    if not (name and email and comment):
        return "A parameter is missing"
    msg ="From: %s\n" % email
    msg+="Subject: feedback\nTo: admin\n"
    msg+="I have the following comment:\n"
    msg+=comment+"\nThank You,\n\n%s" % name
    conn = smtplib.SMTP(SMTP_SERVER)
    conn.sendmail(email, "admin", msg)
    conn.quit()
    resp ="<html> Dear %s,<br>" % name
    resp+="Thank You for your comments."
    return resp+"</html>"
```

Apache Module Management

- List of enabled modules: `apache2 -l`
- Install additional modules in a regular way

```
apt-get install libapache2-mod-python
```

- Installed modules
  - `/etc/apache2/mods-available/`
  - `mod-ruby.load` & `mod-ruby.conf`
- Enabled modules
  - `/etc/apache2/mods-enabled/`
  - symbolic links
  - `a2enmod` & `a2dismod`
- Reload new configurations
  - `/etc/init.d/apache2 reload`

LA{M|P}P

- A stack of programs for creating web applications
  - GNU/Linux — operating system
  - Apache — web server
  - MySQL/PostgreSQL — database system
  - Python (Perl,PHP) — scripting language
- *Free* and/or *open source* software
- Consistent, cross-platform tool
  - high flexibility
  - fast prototyping
  - scalability

Intelligent Systems Lab

Questions?