LAMP

CC ICT-SUD

Setting up and integrate Apache,
MySQL and PHP on a Linux system

Installation

Simple Alternative
(for development/testing only): Xampp

I will assume MySQL is already installed and configured (see related presentation)

- Arch Linux:
  ```
  # pacman -S apache php php-apache
  ```
- Fedora:
  ```
  # yum install httpd php php-mysql
  ```
- Ubuntu:
  ```
  $ sudo apt-get install php5 apache2
  ```

Start the service

- Arch Linux:
  ```
  # /etc/rc.d/httpd start
  ```
- Fedora:
  ```
  # service httpd start
  ```
- Ubuntu
  ```
  $ sudo invoke-rc.d  apache2 start
  ```

  Test by visiting http://localhost/ in a web browser.

Basic Apache Conf (1)

- Main configuration file
  - Arch Linux, Fedora: /etc/httpd/conf/httpd.conf
  - Ubuntu: /etc/apache2/apache2.conf
- If you set a hostname different from localhost edit /etc/hosts, e.g. for a local hostname:

```
127.0.0.1 localhost.localdomain localhost
myhostname
```

Document root (2)

This is the directory where you should put your web pages

- Arch Linux: /srv/http
- Fedora: /var/www/html
- Ubuntu: /var/www

Related directive: `DocumentRoot ”path”` and `<Directory "path">` to grant necessary privileges

User directories (3)

E.g., ~/public_html on the machine is accessed as http://localhost/~user/.To enable uncomment these lines in /etc/httpd/conf/extra/httpd-userdir.conf:

```
UserDir public_html

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

User directories (3)

You must make sure that your home directory permissions are set properly so that Apache can get there. Your home directory and `~/public_html/` must be executable for others ("rest of the world"). This seems to be enough:

```
$ chmod o+x ~
$ chmod o+x ~/public_html
```

There may be some other, more-secure ways of setting the permissions by creating a special group and allowing only Apache and you to enter there.

Enabling PHP support (4)

- Add these lines in `apache2/httpd.conf`:

  LoadModule php5_module modules/libphp5.so
- Place this at the end of the "Include" list:

  Include conf/extra/php5_module.conf
- Restart the Apache service to make changes take effect.
- Testing: save this file as `test.php` and copy to document root or to `~/public_html`. Go to `http://localhost/test.php` or `http://localhost/~myname/test.php`

  <?php phpinfo() ?>
- If the php instruction is not executed add this to your `apache2/httpd.conf` - normally it's already taken care of:

  AddType application/x-httpd-php .php

  AddType application/x-httpd-php-source .phps

Basic PHP configuration (1)

- PHP configuration file:
  - Arch Linux: /etc/php/php.ini
  - Fedora: /etc/php.ini
  - Ubuntu: /etc/php5/apache2/php.ini
- Restart the Apache service to make changes take effect
- open_basedir:

```
open_basedir=/srv/http:/home/:/tmp/:/usr/sha
re/pear/
```

Basic PHP configuration (2)

- enabling extensions: uncomment each line by removing ;

```
;extensions=mysqli.so
;extension=pdo.so
;extension=pdo_mysql.so
```

- Display errors to help debug your php code:

  ```
  display_errors=On
  ```

  don't do it on production web sites!

PhpMyAdmin (1)

- A PHP web-based tool to administrate MySQL over the WWW
- Features: create and drop databases, create/drop/alter tables, delete/edit/add fields, execute any SQL statement, manage keys on fields, manage privileges, export data into various formats, etc
- Installation:
  - Arch Linux: # pacman -S phpmyadmin php-mcrypt
  - Fedora: # yum install phpmyadmin
  - Ubuntu: $ sudo apt-get install phpmyadmin
- Enable extension=mcrypt.so in php.ini

PhpMyAdmin (2)

- Install configuration file:

```
cp /etc/webapps/phpmyadmin/apache.example.conf
/etc/httpd/conf/extra/httpd-phpmyadmin.conf
```

- Add the following line to /etc/httpd/conf/httpd.conf:

```
Include conf/extra/httpd-phpmyadmin.conf
```

- Ensure there is not a line in **deny from all** in /usr/share/webapps/phpmyadmin/.htaccess

PhpMyAdmin (3)

- Sample httpd-phpmyadmin.conf:

```
Alias /phpmyadmin "/usr/share/webapps/phpMyAdmin"

<Directory "/usr/share/webapps/phpMyAdmin">
        AllowOverride All
        Options FollowSymlinks
        Order allow,deny
        Allow from all
</Directory>
```

- open_basedir in php.ini must include :/usr/share/webapps/:/etc/webapps
- In config.inc.php: $cfg['Servers'][$i]['extension'] = 'mysqli';

PhpMyAdmin (4)

- Advanced features:
  - Query bookmarks
  - Column comments
  - SQL-history
  - Tracking mechanism
  - PDF-generation
  - Field content transformation
  - Relations (even for MyISAM table type)

PhpMyAdmin (5)

- Enabling some advanced features:
  - Define a controluser with the proper rights:

```
GRANT SELECT, INSERT, DELETE, UPDATE ON `phpmyadmin`.*
    TO 'pma'@localhost IDENTIFIED BY 'some_pass';
```

  - Create support database:

```
$ mysql -uroot -p <scripts/create_tables.sql
```

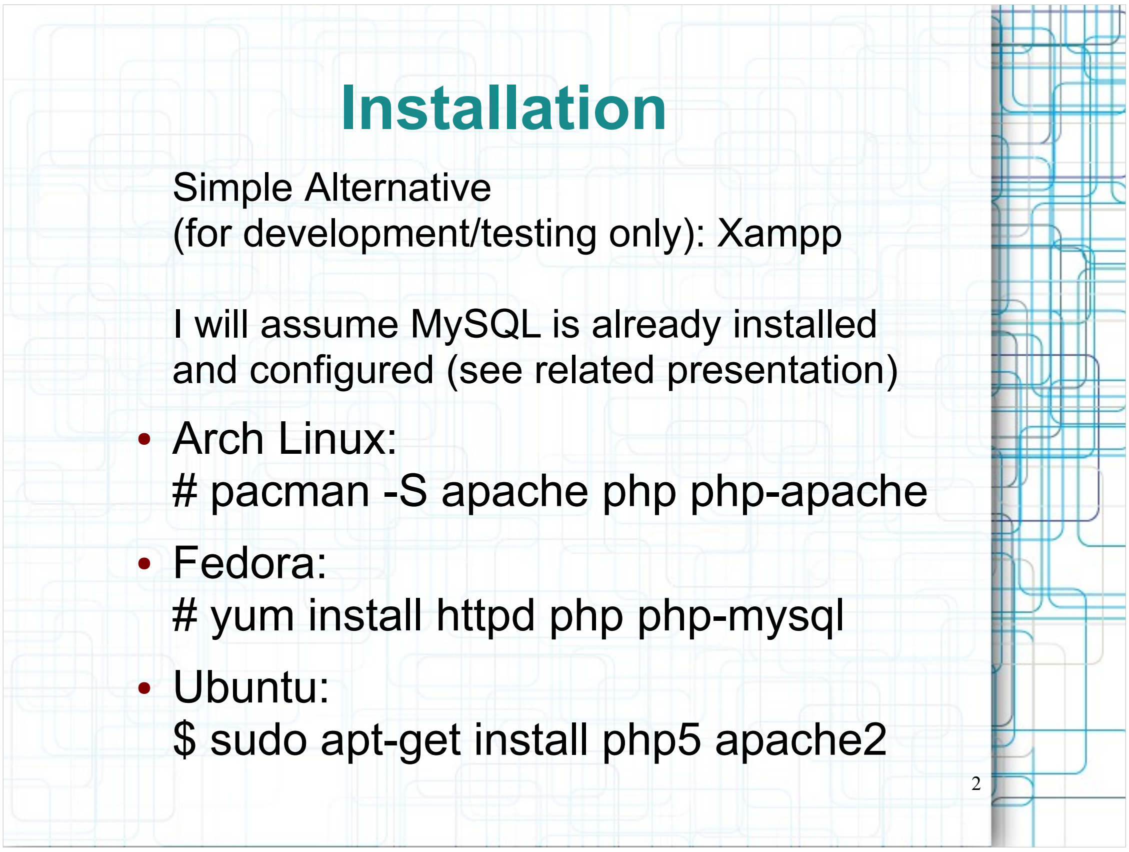
  - Uncomment some lines in config.inc.php:

```
$cfg['Servers'][$i]['pmadb'] = 'phpmyadmin'

$cfg['Servers'][$i]['relation'] = 'pma_relation'

$cfg['Servers'][$i]['controlpass'] = 'some_pass';

$cfg['Servers'][$i]['controluser'] = 'pma'; ...
```

LAMP

CC ICT-SUD

Setting up and integrate Apache,
MySQL and PHP on a Linux system



Installation

Simple Alternative
(for development/testing only): Xampp

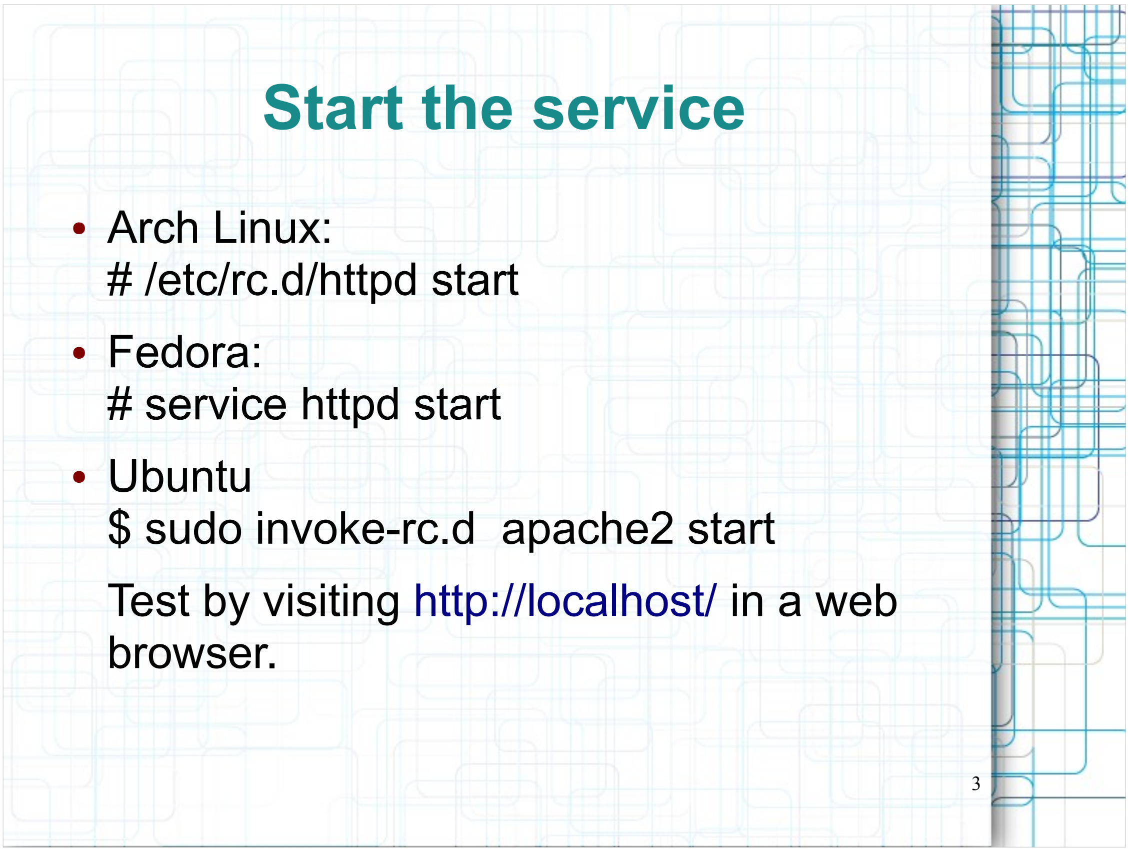
I will assume MySQL is already installed
and configured (see related presentation)

- Arch Linux:
  ```
  # pacman -S apache php php-apache
  ```
- Fedora:
  ```
  # yum install httpd php php-mysql
  ```
- Ubuntu:
  ```
  $ sudo apt-get install php5 apache2
  ```

php: command line PHP interpreter

php-apache: Apache SAPI (Server Application Programming Interface) for PHP

XAMPP is an easy to install and to use Apache distribution containing MySQL, PHP and Perl. It is available for all the most common OS's (Linux, Windows and Mac OS X)



Start the service

- Arch Linux:

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

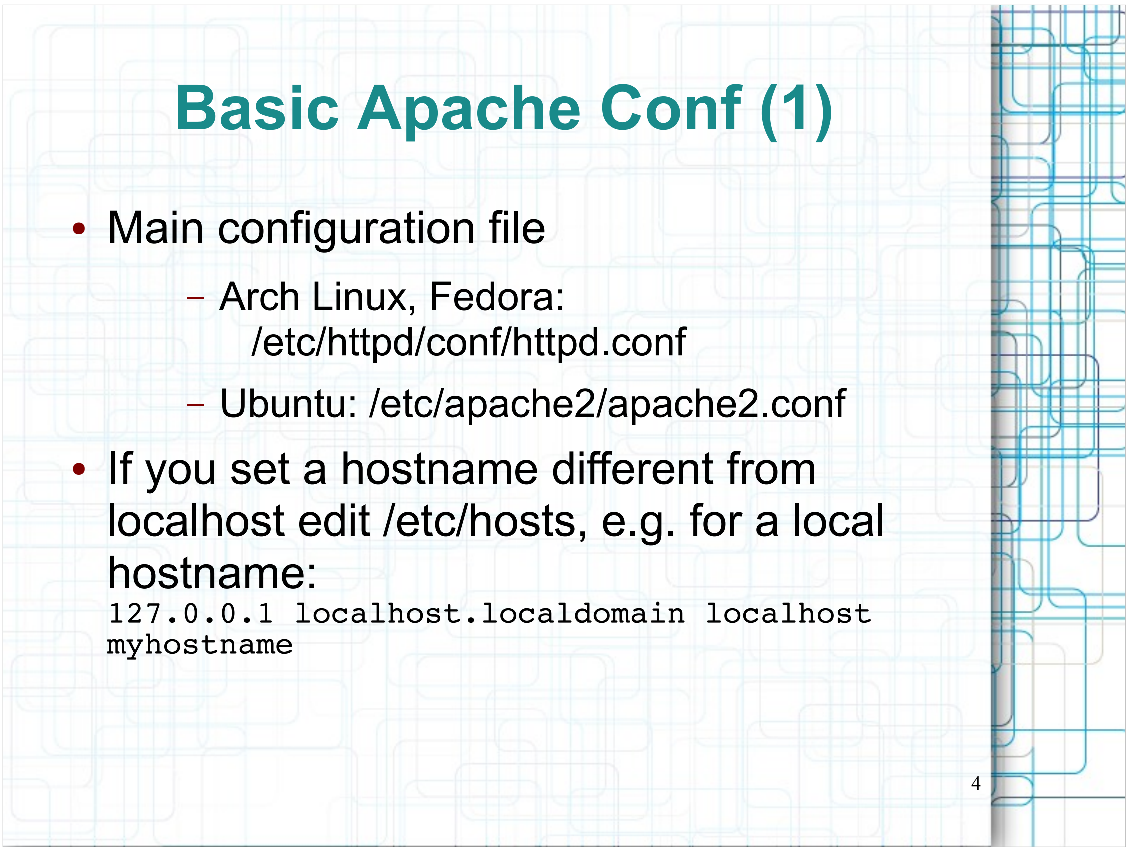
- Fedora:

  ```
  # service httpd start
  ```

- Ubuntu

  ```
  $ sudo invoke-rc.d  apache2 start
  ```

  Test by visiting http://localhost/ in a web browser.

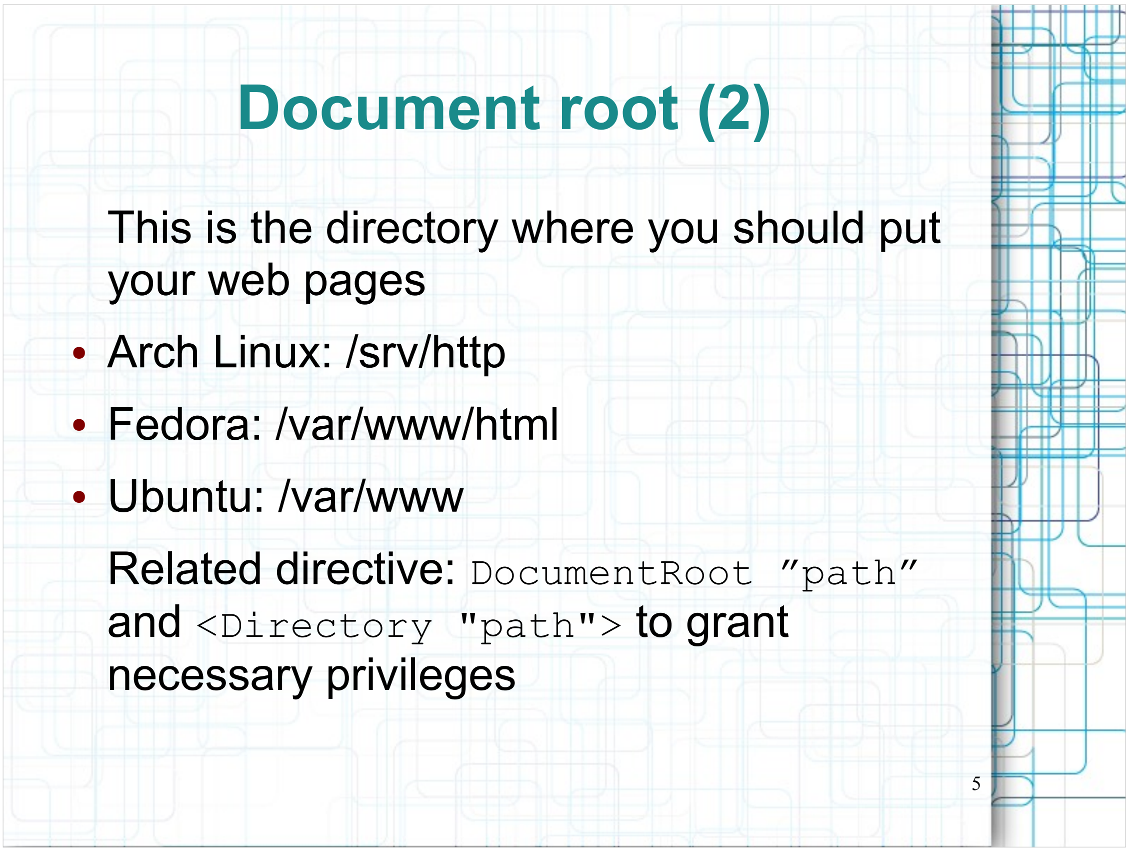


Basic Apache Conf (1)

- Main configuration file
  - Arch Linux, Fedora: /etc/httpd/conf/httpd.conf
  - Ubuntu: /etc/apache2/apache2.conf
- If you set a hostname different from localhost edit /etc/hosts, e.g. for a local hostname:

```
127.0.0.1 localhost.localdomain localhost myhostname
```

If you set a hostname and it does not appear in the static hostnames database, apache will fail to start.

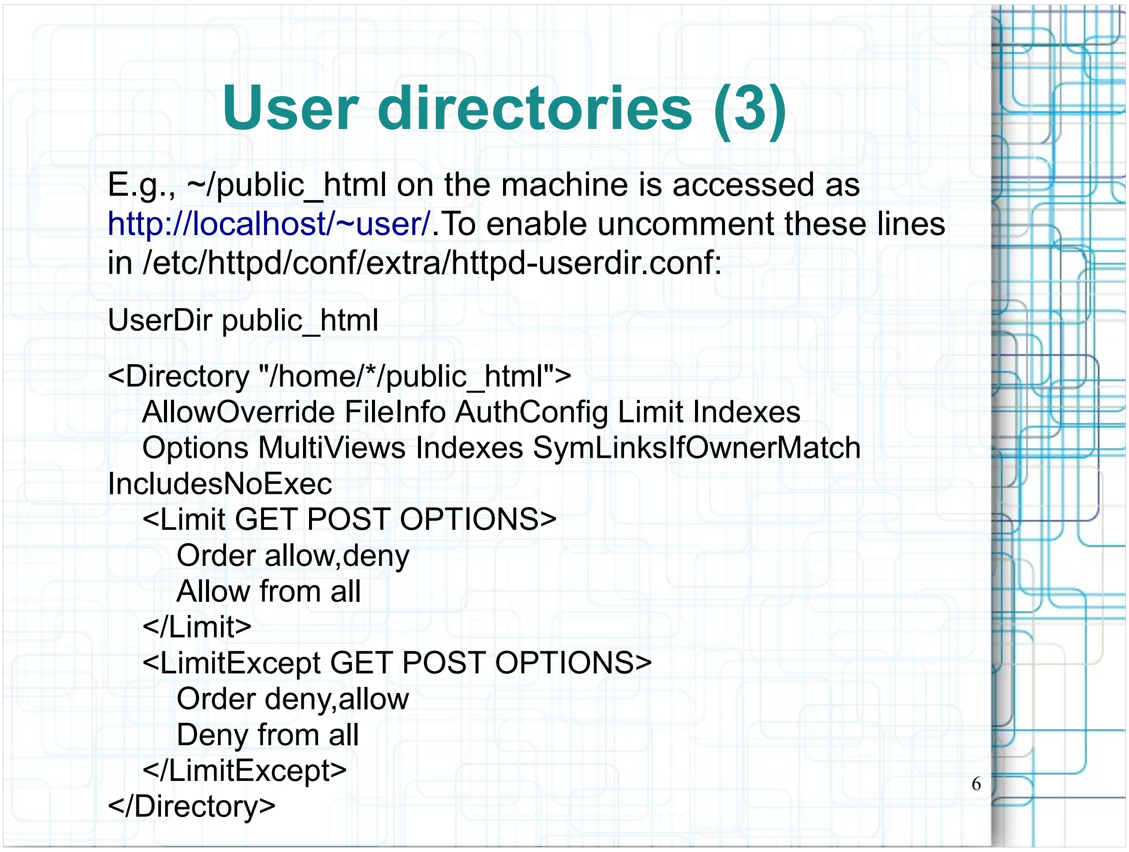


Document root (2)

This is the directory where you should put your web pages

- Arch Linux: /srv/http
- Fedora: /var/www/html
- Ubuntu: /var/www

Related directive: `DocumentRoot ”path”` and `<Directory "path">` to grant necessary privileges



User directories (3)

E.g., ~/public_html on the machine is accessed as http://localhost/~user/.To enable uncomment these lines in /etc/httpd/conf/extra/httpd-userdir.conf:

```
UserDir public_html

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

UserDir: The name of the directory that is appended onto a user's home directory if a ~user request is received. Note that you must also set the default access control for these directories, as in the example below.

Control access to UserDir directories. The following is an example for a site where these directories are restricted to read-only.

UserDir: Directory-filename: location of the user-specific directories

<Directory>: enclose a group of directives that apply only to the named file-system directory and sub-directories

AllowOverride: type of directives that are allowed in .htaccess files

Options: Configures what features are available in a particular directory

<Limit>: Restrict enclosed access controls to only certain HTTP methods

<LimitExcept>: Restrict access controls to all HTTP methods except the named ones

User directories (3)

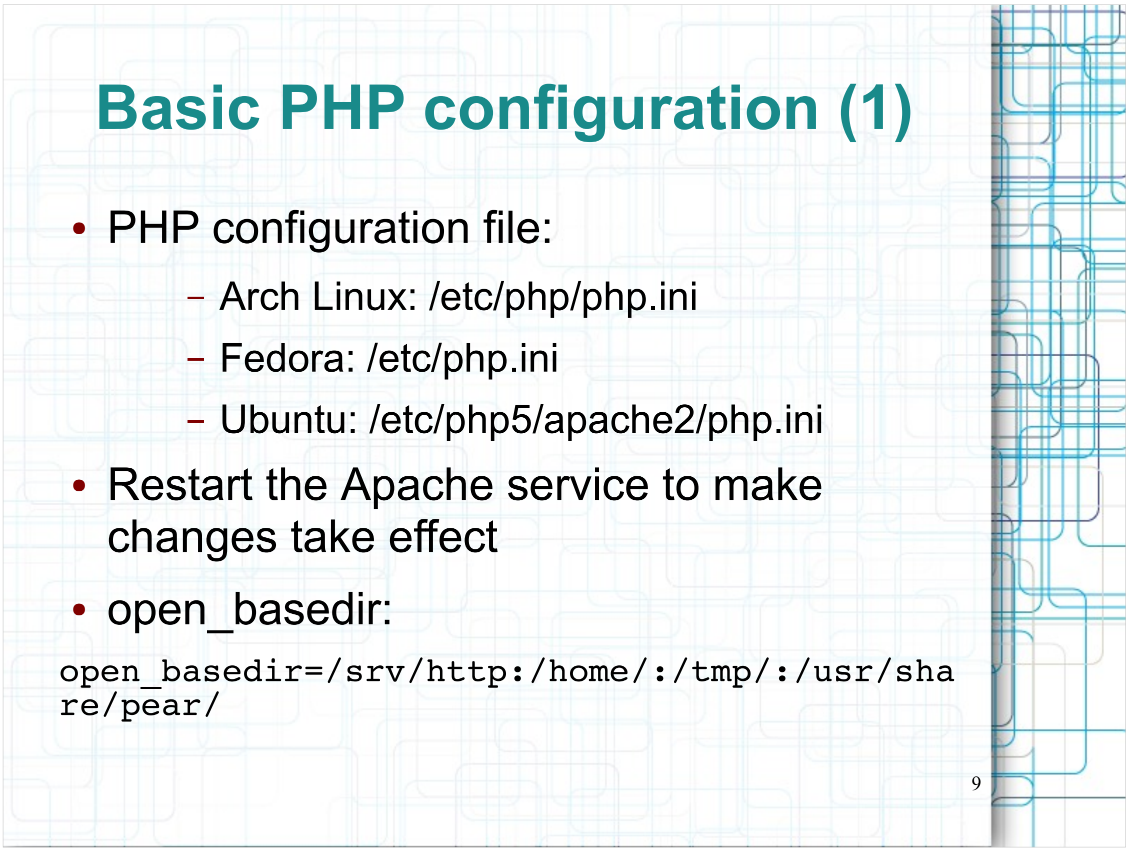
You must make sure that your home directory permissions are set properly so that Apache can get there. Your home directory and `~/public_html/` must be executable for others ("rest of the world"). This seems to be enough:

```
 $ chmod o+x ~
 $ chmod o+x ~/public_html
```

There may be some other, more-secure ways of setting the permissions by creating a special group and allowing only Apache and you to enter there.

Enabling PHP support (4)

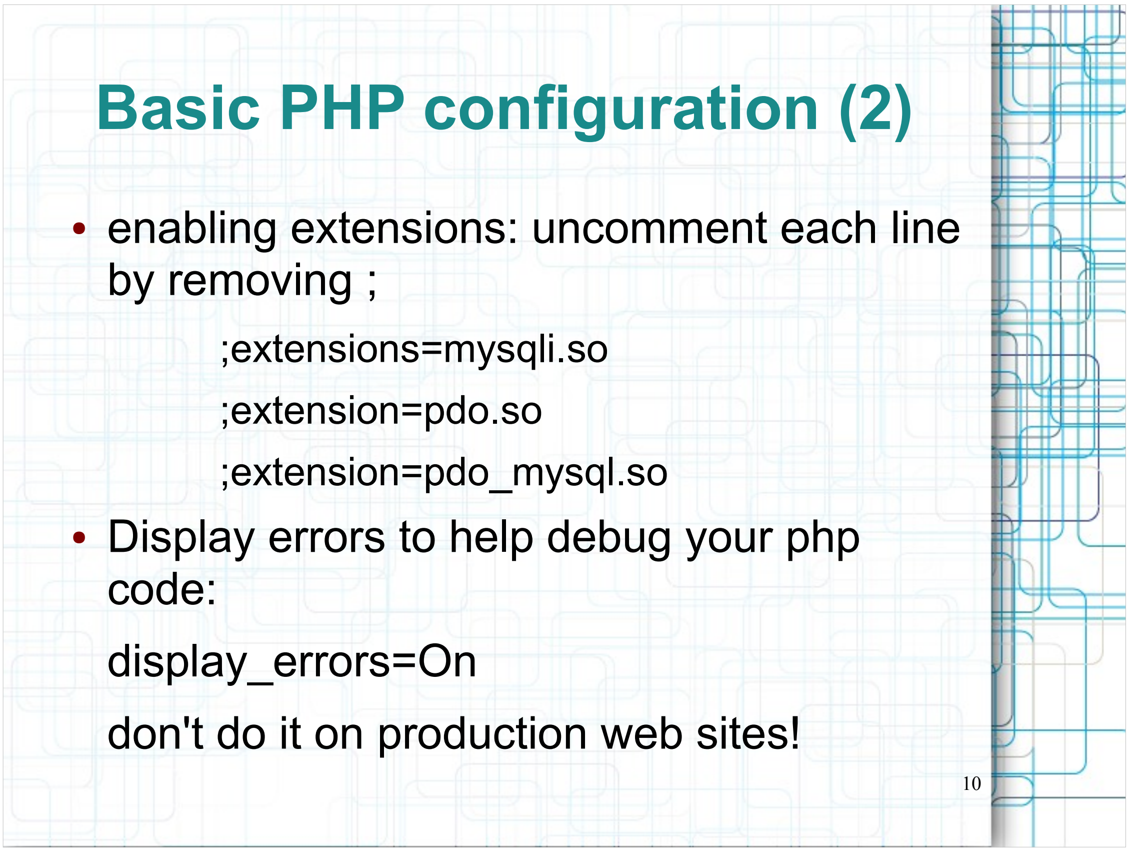
- Add these lines in `apache2/httpd.conf`:
  LoadModule php5_module modules/libphp5.so
- Place this at the end of the "Include" list:
  Include conf/extra/php5_module.conf
- Restart the Apache service to make changes take effect.
- Testing: save this file as `test.php` and copy to document root or to `~/public_html`. Go to `http://localhost/test.php` or `http://localhost/~myname/test.php`
  <?php phpinfo() ?>
- If the php instruction is not executed add this to your `apache2/httpd.conf` - normally it's already taken care of:
  AddType application/x-httpd-php .php
  AddType application/x-httpd-php-source .phps



Basic PHP configuration (1)

- PHP configuration file:
  - Arch Linux: /etc/php/php.ini
  - Fedora: /etc/php.ini
  - Ubuntu: /etc/php5/apache2/php.ini
- Restart the Apache service to make changes take effect
- open_basedir:

```
open_basedir=/srv/http:/home/:/tmp/:/usr/sha
re/pear/
```



Basic PHP configuration (2)

- enabling extensions: uncomment each line by removing ;

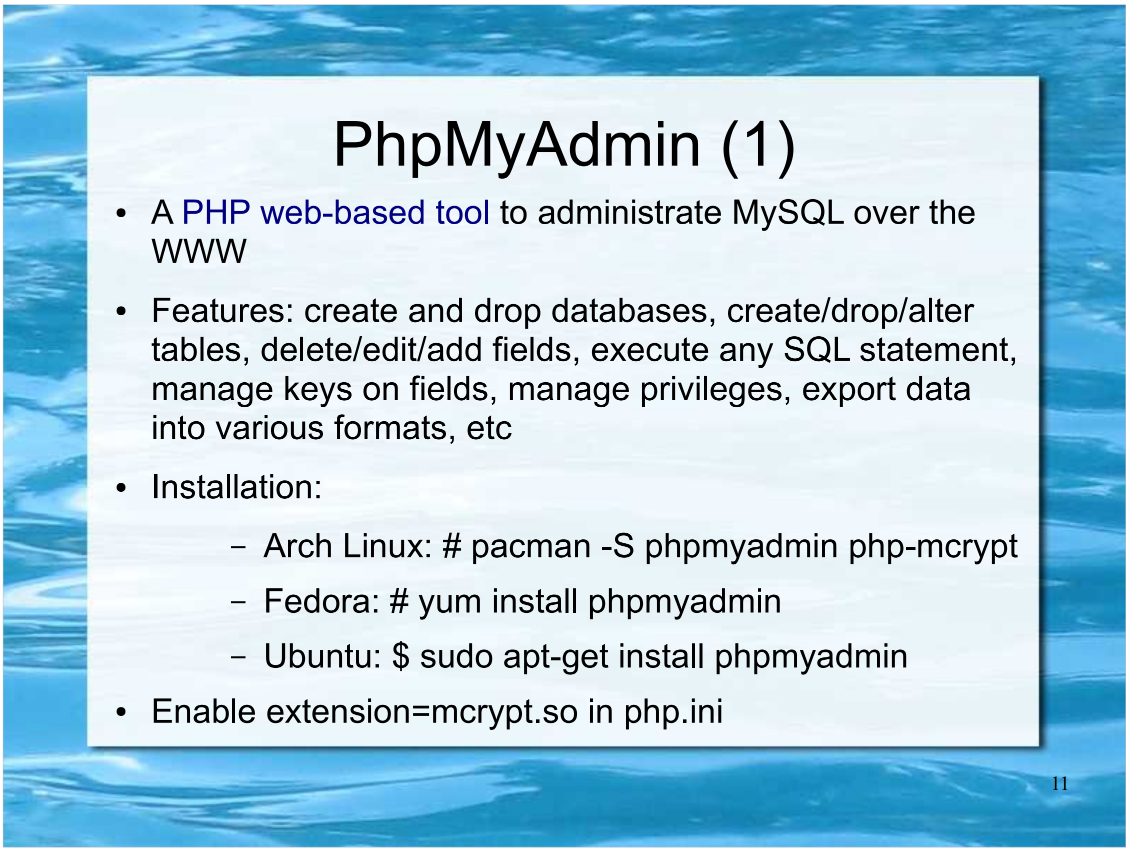
```
;extensions=mysqli.so
;extension=pdo.so
;extension=pdo_mysql.so
```

- Display errors to help debug your php code:

  display_errors=On

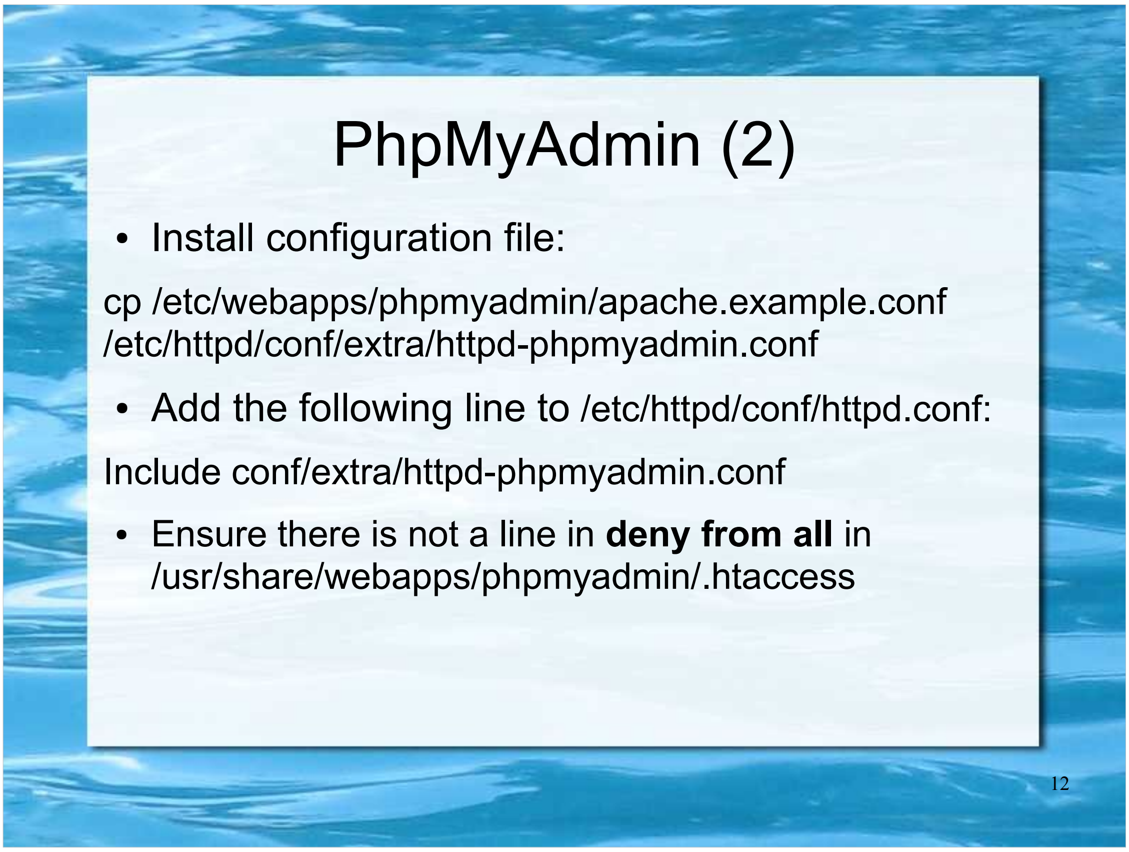
  don't do it on production web sites!

You're strongly advised to use error logging in place of error displaying **on production web sites because they could be useful to someone attempting to crack your server.**



PhpMyAdmin (1)

- A PHP web-based tool to administrate MySQL over the WWW
- Features: create and drop databases, create/drop/alter tables, delete/edit/add fields, execute any SQL statement, manage keys on fields, manage privileges, export data into various formats, etc
- Installation:
  - Arch Linux: # pacman -S phpmyadmin php-mcrypt
  - Fedora: # yum install phpmyadmin
  - Ubuntu: $ sudo apt-get install phpmyadmin
- Enable extension=mcrypt.so in php.ini



PhpMyAdmin (2)

- Install configuration file:

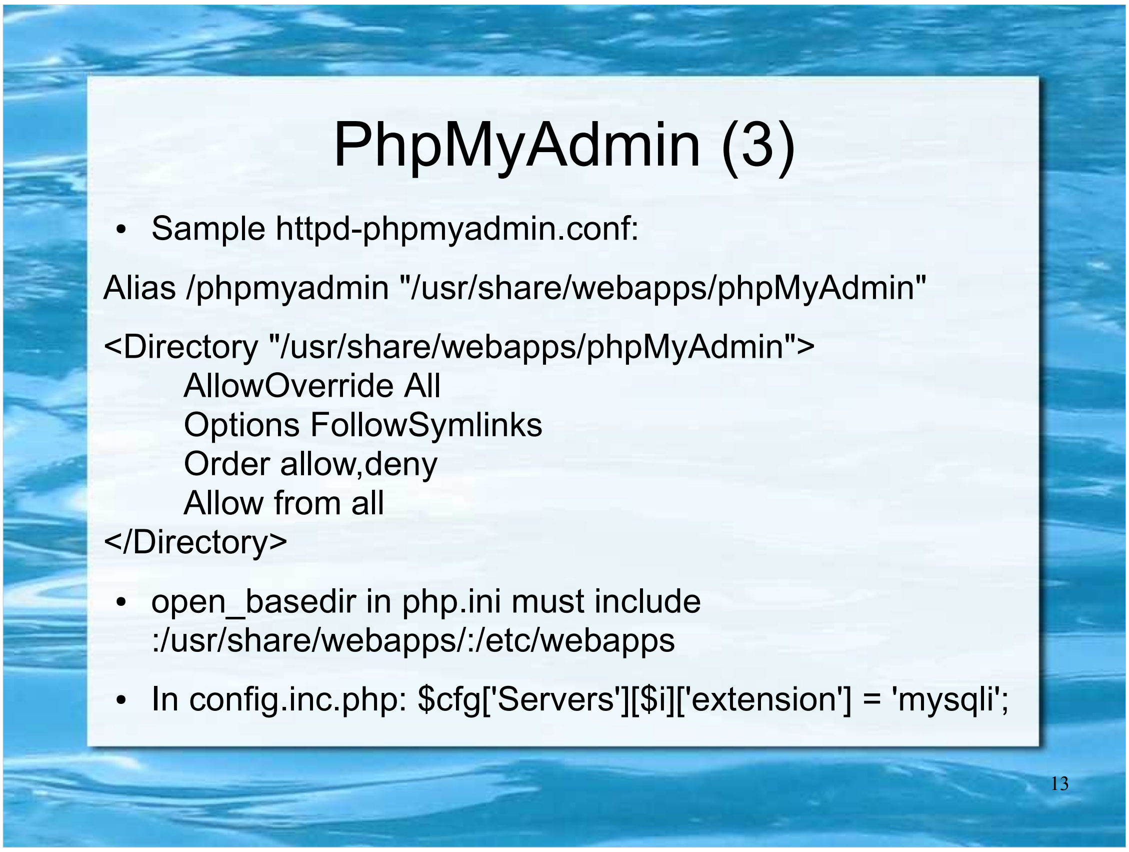
```
cp /etc/webapps/phpmyadmin/apache.example.conf
/etc/httpd/conf/extra/httpd-phpmyadmin.conf
```

- Add the following line to /etc/httpd/conf/httpd.conf:

```
Include conf/extra/httpd-phpmyadmin.conf
```

- Ensure there is not a line in **deny from all** in /usr/share/webapps/phpmyadmin/.htaccess

/etc/webapps/phpmyadmin/ is the location for configuration files pertaining to phpMyAdmin



PhpMyAdmin (3)

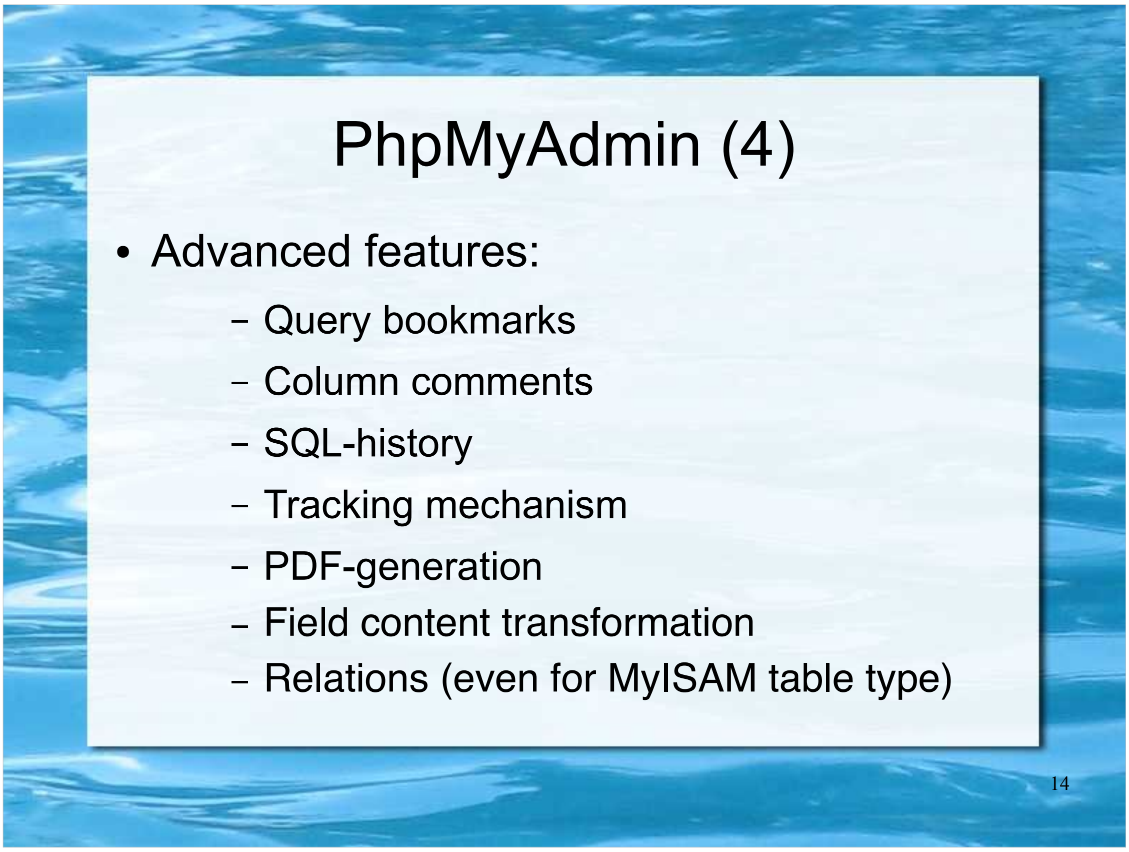
- Sample httpd-phpmyadmin.conf:

```
Alias /phpmyadmin "/usr/share/webapps/phpMyAdmin"

<Directory "/usr/share/webapps/phpMyAdmin">
        AllowOverride All
        Options FollowSymlinks
        Order allow,deny
        Allow from all
</Directory>
```

- open_basedir in php.ini must include :/usr/share/webapps/:/etc/webapps
- In config.inc.php: $cfg['Servers'][$i]['extension'] = 'mysqli';

Note that if you include a trailing / on the *url-path* of the Alias directive then the server will require a trailing / in order to expand the alias. That is, if you use `Alias /phpmyadmin/ /usr/share/webapps/phpmyadmin/` then the url `/phpmyadmin` will not be aliased.

Select mysqli as extension to use for making phpMyAdmin communicating with MySQL, because it is faster than the old mysql extension.



PhpMyAdmin (4)

- Advanced features:
  - Query bookmarks
  - Column comments
  - SQL-history
  - Tracking mechanism
  - PDF-generation
  - Field content transformation
  - Relations (even for MyISAM table type)

14

Bookmarks are useful for queries you often run. Any query you have executed can be stored as a bookmark on the page where the results are displayed. A bookmark is related to the database you run the query on and single variable support is available.

You can store comments to describe each column for each table. These will then be shown on the "printview" or as tool-tips above the column names on the table property pages.

SQL history means all queries you entered manually into the phpMyAdmin interface are saved across navigation sessions.

Tracking allows creating versions of tables and logs both DDL and DML SQL commands.

You can create PDF graphics of your Database layout.

Transform stored data into any format using a set of predefined functions, like displaying BLOB-data as image or download-link

PhpMyAdmin (5)

- Enabling some advanced features:
  - Define a controluser with the proper rights:

    ```
    GRANT SELECT, INSERT, DELETE, UPDATE ON `phpmyadmin`.*
        TO 'pma'@localhost IDENTIFIED BY 'some_pass';
    ```

  - Create support database:

    ```
    $ mysql -uroot -p <scripts/create_tables.sql
    ```

  - Uncomment some lines in config.inc.php:

    ```
    $cfg['Servers'][$i]['pmadb'] = 'phpmyadmin'

    $cfg['Servers'][$i]['relation'] = 'pma_relation'

    $cfg['Servers'][$i]['controlpass'] = 'some_pass';

    $cfg['Servers'][$i]['controluser'] = 'pma'; ...
    ```