How To LEMP su CentOS 7

How To: Installare Linux, Nginx, MySQL, PHP 7 (LEMP) su CentOS 7

Un ambiente LEMP a differenza dell'ambiente LAMP (Linux, Apache, MySQL, PHP) differisce per il web server ENGINX.

In questo HOW TO andremo a vedere come installare PHP 7 FPM con Nginx per avere le massime prestazioni.

Il primo passo da fare è disabilitare il SELINUX, la cosa migliore sarebbe configurarlo ad hoc, ma questa operazione va eseguite prima della messa in esercizio del server.

Primo step installare Nginx

Come primo passaggio installiamo il repository EPEL e IUS. Utilizziamo il comodissimo script messo a disposizione sul sito ius.io per fare prima. Il curl ci aiuta a scaricare il setup.

```
curl 'https://setup.ius.io/' -o setup-ius.sh
```

ora eseguiamo lo script:

```
bash setup-ius.sh
```

Ora possiamo installare NGINX

```
yum install nginx
```

avviamo il webserver con il comando systemctl

```
systemctl start nginx
```

ora possiamo provare puntando nel nostro browser

```
http://indirizzo_ip_del_server/
```

Se vedi questa pagina allora possiamo abilitare NGINX all'avvio del sistema,

al BOOT:

```
systemctl enable nginx
```

Secondo Step installare MySQL

Abbiamo due possibilità, installare MySQL o MariaDB, sono la stessa cosa nati dallo stesso adre **Ulf Michael Widenius** noto anche come **Monty.**

In questa guida opteremo per MySQL, il comando è il seguente:

```
yum install mysql-server mysql
```

Ora passiamo alla prima configurazione del nostro RDBMS: start del demone:

```
service mysqld status
```

e poi messa in sicurezza di base:

```
mysql_secure_installation
```

Siamo pronti per abilitare anche MySQL al boot:

```
systemctl enable mysqld
```

Terzo Step installazione di PHP-FPM 7

Ora passiamo all'installazione dei PHP-FPM (FastCGI Process Manager) 7, l'ultima versione del php disponibile ad oggi eseguito sulla porta 9000:

```
yum install php70u-fpm-nginx php70u-cli php70u-mysqlnd
```

installato apriamo il file di configurazione e sostituiamo l'utente e i l gruppo d'esecuzione:

```
vim /etc/php-fpm.d/www.conf
```

```
; When POSIX Access Control Lists are supported you can set them using
; these options, value is a comma separated list of user/group names.
; When set, listen.owner and listen.group are ignored
;listen.acl_users = apache,nginx
;listen.acl_users = apache
listen.acl_users = nginx
;listen.acl_groups =
```

a questo punto riavviamo creiamo un vhosts, per prima cosa per tenere in ordine il nostro ambiente posizioniamo i file dei virtual hosts in una directory:

```
mkdir /etc/nginx/sites-available
```

passiamo al file nginx.conf l’istruzione di leggere il contenuto della nuova direcotry

```
vim /etc/nginx/nginx.conf
```

aggiungendo la riga

```
include /etc/nginx/sites-enabled/*;
```

```
;server {
listen 81.127.13.234:80;
server_name stat.lbit-solution.it;
location / {
try_files $uri $uri/ =404;
root /var/www/vhosts/lbit-solution.it/stat.lbit-solution.it/;
index index.php index.html index.htm;
}

error_page 404 /404.html;
error_page 500 502 503 504 /50x.html;
location = /50x.html {
root /var/www/vhosts/lbit-solution.it/stat.lbit-solution.it/;
}

location ~ \.php$ {
root /var/www/vhosts/lbit-solution.it/stat.lbit-solution.it/;
fastcgi_pass 127.0.0.1:9000;
fastcgi_index index.php;
fastcgi_param SCRIPT_FILENAME /var/www/vhosts/lbit-solution.it/stat.lbit-
solution.it$fastcgi_script_name;
include fastcgi_params;
}
}

server {
listen 81.127.13.234:443 ssl;
server_name stat.lbit-solution.it;

### SSL cert files ###
```

```
ssl_certificate /var/www/vhosts/lbit-solution.it/ssl/stat.lbit-
solution.it.crt;
ssl_certificate_key /var/www/vhosts/lbit-solution.it/ssl/stat.lbit-
solution.it.key;

### Add SSL specific settings here ###

ssl_protocols SSLv3 TLSv1 TLSv1.1 TLSv1.2;
ssl_ciphers RC4:HIGH:!aNULL:!MD5;
ssl_prefer_server_ciphers on;
keepalive_timeout 60;
ssl_session_cache shared:SSL:10m;
ssl_session_timeout 10m;

### SSL log files ###
access_log /var/www/vhosts/lbit-solution.it/logs/stat.lbit-solution.it.ssl-
access.log;
error_log /var/www/vhosts/lbit-solution.it/logs/stat.lbit-solution.it.ssl-
error.log;

location / {
try_files $uri $uri/ =404;
root /var/www/vhosts/lbit-solution.it/stat.lbit-solution.it/;
index index.php index.html index.htm;
}

error_page 404 /404.html;
error_page 500 502 503 504 /50x.html;
location = /50x.html {
root /var/www/vhosts/lbit-solution.it/stat.lbit-solution.it/;
}

location ~ \.php$ {
root /var/www/vhosts/lbit-solution.it/stat.lbit-solution.it/;
fastcgi_pass 127.0.0.1:9000;
fastcgi_index index.php;
fastcgi_param SCRIPT_FILENAME /var/www/vhosts/lbit-solution.it/stat.lbit-
solution.it$fastcgi_script_name;
include fastcgi_params;
}
}
```

Prepariamo le direcotry del virtual hosts:

```
mkdir -p /var/www/vhosts/lbit-solution.it/{ssl,logs,httpdocs,stat.lbit-
solution.it}
```

E ora riavviamo php-fpm e Nginx

```
sudo systemctl restart php-fpm
sudo systemctl restart nginx
```

Vulnerabilità WordPress SEO by Yoast

Vulnerabilità SEO, ma come risolvere i danni fatti?

Come scritto nel blog ufficiale (LINK) il blasonato plugin SEO by Yoast soffre di una gravissima vulnerabilità: **Blind SQL Injection,** file interessato sarebbe il ***class-bulk-editor-list-table.php***.

Cos'è una **Blind SQL Injection** e come possiamo sfruttarla?

Un hacker inserisce una query SQL non valida in un'applicazione, nel nostro caso **WordPress** che avendo un autore, un admin o un editor già autenticati che visitano un URL malformato, il malintenzionato riesce ad accedere e modificare il database **WordPress.**

Vediamo cosa è successo proprio a noi che scriviamo questo articolo.

Il furbo di turno ha sfruttato la vulnerabilità per modificare il database, creare un nuovo utente, concedergli i privilegi amministrativi ed aggiungere delle widget con codice javascript.

Tale codice servire a modificare i link del sito per rimandare a pubblicità, il modo più veloce per monetizzare. Fortunatamente l'hacker aveva un suo scopo ben preciso, quello di monetizzare, per questo motivo non ha fatto danni.

Questo serve a riflettere su quanti usano WordPress per scopi professionali senza affidarsi ad aziende o professionisti del settore.

Come suggerito da hostingtalk.it:

> In casi come questi, il consiglio è di **aggiornare immediatamente** il plugin WordPress SEO by Yoast all'ultima versione disponibile o di **affidarsi a servizi di hosting gestiti**, che eseguono in automatico per l'utenza gli upgrade di sicurezza necessaria. Altra alternativa è **l'autoaggiornamento di WordPress**, sempre che non sia stato disabilitato.

In alternativa un contratto di manutenzione può salvare il proprio business.

Abilitare diverse versioni di PHP in PLESK

Amministrando un web server con PLESK prima o poi arriva la richiesta di installare una seconda versione di PHP e di renderla disponibile ai clienti attraverso il pannello PLESK.

In questa guida l'installazione è stata fatta su una macchina CentOS 6.6 e Plesk 11.5.30:

Per prima cosa creiamo la directory dove potere scaricare il pacchetto PHP

```
cd /usr/local/src
# mkdir php562
# cd php562
wget http://it1.php.net/get/php-5.6.2.tar.gz/from/this/mirror
mv mirror php-5.6.2.tar.gz
tar -xvzf php-5.6.2.tar.gz
cd php-5.6.2
```

Siamo pronti per iniziare, configuriamo per la compilazione:

```
./configure '--with-libdir=lib64' '--cache-file=../config.cache' '--prefix=/usr/local/php562-cgi' '--with-config-file-path=/usr/local/php562-cgi/etc' '--disable-debug' '--with-pic' '--disable-rpath' '--enable-fastcgi' '--with-bz2' '--with-curl' '--with-xpm-dir=/usr/local/php562-cgi' '--with-png-dir=/usr/local/php562-cgi' '--enable-gd-native-ttf' '--without-gdbm' '--with-gettext' '--with-gmp' '--with-iconv' '--with-jpeg-dir=/usr/local/php562-cgi' '--with-openssl' '--with-pspell' '--with-pcre-regex' '--with-zlib' '--enable-exif' '--enable-ftp' '--enable-sockets' '--enable-sysvsem' '--enable-sysvshm' '--enable-sysvmsg' '--enable-wddx' '--with-kerberos' '--with-unixODBC=/usr' '--enable-shmop' '--enable-calendar' '--without-sqlite3' '--with-libxml-dir=/usr/local/php562-cgi' '--enable-pcntl' '--with-imap' '--with-imap-ssl' '--enable-mbstring' '--enable-mbregex' '--with-gd' '--enable-bcmath' '--with-xmlrpc' '--with-ldap' '--with-ldap-sasl' '--with-mysql=/usr' '--with-mysqli' '--with-snmp' '--enable-soap' '--with-xsl' '--enable-xmlreader' '--enable-xmlwriter' '--enable-pdo' '--with-pdo-mysql' '--with-pdo-pgsql' '--with-pear=/usr/local/php562-cgi/pear' '--with-mcrypt' '--enable-intl' '--without-pdo-sqlite' '--with-config-file-scan-dir=/usr/local/php562-cgi/php.d' --enable-shared --enable-zip
```

Ora il classico make e poi make install, mi raccomando non lanciate make test

```
make
make install
```

Copiamo il php.ini sotto nella directory php562-cgi

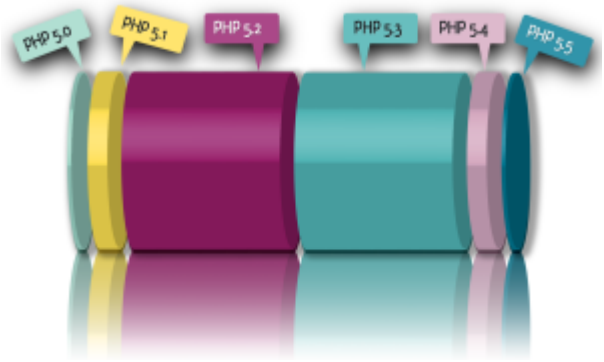
```
cp php.ini-development /usr/local/php562-cgi/php.ini
```

Ora non rimane che censire nel pannello PLESK la nuova versione di PHP

```
/usr/local/psa/bin/php_handler --add -displayname 5.6.2 -path
/usr/local/php562-cgi/bin/php-cgi -phpini /usr/local/php562-cgi/php.ini -type
fastcgi -id 5.6.2
```

Nel mio caso la prima installazione di PHP non è stata così liscia, ho dovuto installare alcuni pacchetti:

```
yum install bzip2-devel.x86_64 bzip2.x86_64
yum install libjpeg*
yum install libpng-devel
yum install freetype
yum install libXpm-devel
yum install libgmp3-dev gmp.x86_64 gmp-devel.x86_64
yum install openssl openssl-devel pam-devel
yum install pam-devel
yum install libicu-devel libc-client-devel.x86_64 libc-client.x86_64
yum install libtomcrypt-devel.x86_64 libmcrypt-devel.x86_64 php-mcrypt.x86_64
yum install unixODBC-devel
yum install postgresql-devel postgresql-libs
yum install pspell php-pspell.x86_64 aspell-devel net-snmp-devel libxslt-
devel libxml2-devel pcre-devel t1lib-devel.x86_64 libtidy-devel php-pecl-zip
```



Visualizzare la struttura di un database mysql da php

Se avete bisogno di “stampare” la struttura di un database mysql esistente e non volete ricorrere a soluzioni come mysql workbench o similari, potete utilizzare un comodissimo script php di David Walsh.

Si tratta di un file .php, lo copiate sotto la DocumentRoot del vostro

webserver, lo editate con db_name, user e password e lo aprite da web. Il risultato sarà del tipo:

admin_roles

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI		auto_increment
nome	varchar(255)	NO			

admin_user

Field	Type	Null	Key	Default	Extra
iduser	int(11)	NO	PRI		auto_increment
username	varchar(255)	NO			
password	varchar(255)	NO			
salt	varchar(255)	NO			
nome	varchar(50)	NO			
cognome	varchar(50)	NO			

Lo script lo potete consultare in questo post del blog di David Walsh oppure scaricarlo da questa pagina su Mr.Webmaster

---

Symfony2: Error SecurityDataCollector

Se in seguito all'aggiornamento del PHP, la vostra webapp Symfony2 presenta il seguente errore:

```
FatalErrorException: Error: Call to a member function getRole() on
a non-object in
C:\xampp\htdocs\AppApartamentos\vendor\symfony\sym
fony\src\Symfony\Bundle\SecurityBundle\DataCollect
or\SecurityDataCollector.php line 60
```

dovete aggiungere un metodo "serialize" alla vostra classe User.

Aprite l’Entità User.php:

modificate la dichiarazione in:

```
class User implements UserInterface, \Serializable
```

ed aggiungete i seguenti metodi:

```
public function serialize()
{
return json_encode(
array($this->username, $this->password, $this->salt,
$this->user_roles, $this->id));
}

/**
* Unserializes the given string in the current User object
* @param serialized
*/
public function unserialize($serialized)
{
list($this->username, $this->password, $this->salt,
$this->user_roles, $this->id) = json_decode(
$serialized);
}
```

La vostra istallaizione è salva!

Symfony - 01 - Installazione e configurazione del framework

Credo sia utile una mini guida su come installare e utilizzare il framework php Symfony2.

Innanzitutto preparatevi un ambiente LAMP, Linux + Apache + Mysql + Php

E prepariamoci un nuovo database mysql, che chiamiamo symfony , che dovremmo utilizzare per abilitare l’accesso securizzato.

Spostiamoci sotto la DocumentRoot di apache (se avete l'installazione di default, è /var/www/html)

Da qui, scarichiamo il composer, che serve per gestire le dipendenze del framework

```
curl -s https://getcomposer.org/installer | php
```

Ed ora scarichiamo il framework (ci vorrà qualche minuto).
Ho scelto la versione 2.3 che è supportata fino a Maggio 2016.

```
php composer.phar create-project symfony/framework-standard-edition symfony/ "2.3.*"
```

Ora vi chiede le informazioni sul database creato per configurare il file "##DOCUMENT_ROOT##/symfony/app/config/parameters.yml" (che potete configurare in un secondo momento).
Ve le riassumo qui:

```
database_driver (pdo_mysql):
database_host (127.0.0.1):
database_port (null):
database_name (symfony): symfony
database_user (root): myuser
database_password (null): mypassword
mailer_transport (smtp):
mailer_host (127.0.0.1):
mailer_user (null):
mailer_password (null):
locale (en): it
secret (ThisTokenIsNotSoSecretChangeIt): 2Cdq235nau
```

Il token potete generare una qualsiasi stringa.

Settiamo i permessi ai file scaricati:

```
chown -R apache.apache symfony
```

```
chmod -R 775 symfony
```

Ultimiamo l'installazione con:

```
cd symfony
mv ../composer.phar .
php composer.phar update
chmod 777 app/cache/ app/logs/
```

Listiamo i file installati:

```
ls -l
```

E avremo in seguente output:

```
totale 116
drwxrwxr-x 6 apache apache 4096 29 ago 11:44 app -> contiene log, cache,
```

```
configurazioni, risorse
drwxrwxr-x 2 apache apache 4096 29 ago 11:38 bin -> binari del framework
-rwxrwxr-x 1 apache apache 2085 8 lug 16:17 composer.json
-rwxrwxr-x 1 apache apache 56987 29 ago 11:38 composer.lock
-rwxrwxr-x 1 apache apache 1065 8 lug 16:17 LICENSE
-rwxrwxr-x 1 apache apache 5736 8 lug 16:17 README.md
drwxrwxr-x 3 apache apache 4096 8 lug 16:17 src -> file sorgenti del progetto
-rwxrwxr-x 1 apache apache 1308 8 lug 16:17 UPGRADE-2.2.md
-rwxrwxr-x 1 apache apache 1962 8 lug 16:17 UPGRADE-2.3.md
-rwxrwxr-x 1 apache apache 8499 8 lug 16:17 UPGRADE.md
drwxrwxr-x 13 apache apache 4096 29 ago 11:44 vendor -> framework
drwxrwxr-x 3 apache apache 4096 29 ago 11:44 web -> documentRoot del
framework
```

Ora se apriamo la seguente URL, modifichando ##MYHOST## con il nome del server, localhost se siamo in locale:

http://##MYHOST##/symfony/web/config.php

Si aprirà una pagina di verifica della configurazione, se non avete errori segnalati potete cliccare su
“Bypass configuration and go to the Welcome page”

Benvenuti nel vostro nuovo framework!

Se come errori segnalati, riscontrate il seguente:

> Set the “date.timezone” setting in php.ini* (like Europe/Paris).

Aggiungete la stringa
date.timezone = Europe/Berlin
nel file php.ini (di default sotto /etc) e restartare apache.

Nel prossimo post vedremo come configurare l’accesso securizzato.

Richiamare bash da php

Gli script bash possono aiutarci ad interfacciarci al S.O. dal php

Per fare un esempio completo, create uno script che faccia una echo:

vi /home/roberto/script.sh

e dentro incollateci:

```
#!/bin/bash
echo "Ciao, sono la echo!"
```

E fate una pagina php con il seguente contenuto:

```
<?php
$result=shell_exec("/home/roberto/script.sh");
echo($result);
?>
```

Per sistemi unix/linux, se siete sotto Windows ovviamente i path saranno nel formato “C:\…”

Floating Widget Social Network

Sito pulito o widget dei social network?

Non esiste sito internet senza widget social o connessioni ai propri profili o pagine aziendali, il cliente vuole html pulito ma senza rinunciare al badge di google plus piuttosto che al “mi piace” di facebook.

La soluzione è metterli a scomparsa in modo da lasciare il sito pulito senza widget fastidiose e pacchiane.

Guarda la demo http://blog.lbit-solution.it//prova.html

Scarica il file di prova al seguente link:
http://blog.lbit-solution.it/wp-content/uploads/2014/04/prova.html.gz

La personalizzazione del codice è molto semplice, per facebook e google plus basta sostituire il nome account e l’ID del profilo:

```
likebox.php?href=http%3A%2F%2Ffacebook.com%2FLbitSoluzioniInformatiche&width
```

Al posto si “LbitSoluzioniInformatiche” inserisci l’account della pagina facebbok, per quanto riguarda google plus sostituisci l’ID del profilo:

```
g-page" data-width="240" data-
href="https://plus.google.com/109087407088989811737" data-rel="publisher">
```

in questo caso il mio ID è “109087407088989811737“, è sufficiente inserire il proprio per avere il badge configurato.

Twitter invece richiede la creazione di una nuova widget direttamente dal sito twitter.com, nel codice che hai scaricato sostituisci tutto questo:

```
<!-- Start Twitter Badge -->

<a class=”twitter-timeline” href=”https://twitter.com/LinuxLBiT” data-widget-id=”459091732685021184″>Tweets di @LinuxLBiT</a>

<script>!function(d,s,id){var js,fjs=d.getElementsByTagName(s)[0],p=/^http:/.test(d.location)?’http’:’https’;if(!d.getElementById(id)){js=d.createElement(s);js.id=id;js.src=p+”://platform.twitter.com/widgets.js”;fjs.parentNode.insertBefore(js,fjs);}}(document,”script”,”twitter-wjs”);</script>

<!– // END Twitter Badge –>
```

Pochi passi per avere l’effetto desiderato.