# Sicherheit von PHP-FPM/Nginx in geteilten Hostingumgebungen (Debian/Ubuntu)

4. Februar 2014

**1 Vorbemerkung**

Ich benutze hier einen vhost namens example.com/example.com mit dem Dateipfad /var/www/www.example.com/web.

Sie sollten eine funktionierende LEMP Installation haben, so wie in diesen englischsprachigen Tutorials beschrieben:

Installing Nginx With PHP5 And MySQL Support On Debian Squeeze
Installing Nginx With PHP5 (And PHP-FPM) And MySQL Support On Ubuntu 11.04

*Eine Bemerkung für Ubuntu-Benutzer:*

Da Sie alle Schritte des Tutorials mit Administratorrechten ausführen müssen, können Sie entweder jedem Befehl ein sudo voranstellen oder aber von Anfang an im root-Status verbleiben indem Sie folgenden Befehl benutzen:

sudo su

**2 Was bisher vorhanden ist**

Unter Debian/Ubuntu ist das Poolverzeichnis von PHP-FPM /etc/php5/fpm/pool.d/ - dies ist der Ort an dem neue Pools erstellt werden. Die php.ini die von PHP-FPM benutzt wird ist /etc/php5/fpm/php.ini. Es gibt bereits einen Pool, www.conf - Schauen Sie sich diesen einmal an:

## Quellcode

1. vi /etc/php5/fpm/pool.d/www.conf

```
; Start a new pool named 'www'.
; the variable $pool can we used in any directive and will be replaced by the
; pool name ('www' here)
[www]

; Per pool prefix
; It only applies on the following directives:
; - 'slowlog'
; - 'listen' (unixsocket)
; - 'chroot'
; - 'chdir'
; - 'php_values'
; - 'php_admin_values'
; When not set, the global prefix (or /usr) applies instead.
; Note: This directive can also be relative to the global prefix.
; Default Value: none
;prefix = /path/to/pools/$pool
```

```
; The address on which to accept FastCGI requests.
; Valid syntaxes are:
; 'ip.add.re.ss:port' - to listen on a TCP socket to a specific address on
; a specific port;
; 'port' - to listen on a TCP socket to all addresses on a
; specific port;
; '/path/to/unix/socket' - to listen on a unix socket.
; Note: This value is mandatory.
listen = 127.0.0.1:9000

; Set listen(2) backlog. A value of '-1' means unlimited.
; Default Value: 128 (-1 on FreeBSD and OpenBSD)
;listen.backlog = -1

; List of ipv4 addresses of FastCGI clients which are allowed to connect.
; Equivalent to the FCGI_WEB_SERVER_ADDRS environment variable in the original
; PHP FCGI (5.2.2+). Makes sense only with a tcp listening socket. Each address
; must be separated by a comma. If this value is left blank, connections will be
; accepted from any ip address.
; Default Value: any
;listen.allowed_clients = 127.0.0.1

; Set permissions for unix socket, if one is used. In Linux, read/write
; permissions must be set in order to allow connections from a web server. Many
; BSD-derived systems allow connections regardless of permissions.
; Default Values: user and group are set as the running user
; mode is set to 0666
;listen.owner = www-data
;listen.group = www-data
;listen.mode = 0666

; Unix user/group of processes
; Note: The user is mandatory. If the group is not set, the default user's group
; will be used.
user = www-data
group = www-data

; Choose how the process manager will control the number of child processes.
; Possible Values:
; static - a fixed number (pm.max_children) of child processes;
; dynamic - the number of child processes are set dynamically based on the
; following directives:
; pm.max_children - the maximum number of children that can
; be alive at the same time.
; pm.start_servers - the number of children created on startup.
; pm.min_spare_servers - the minimum number of children in 'idle'
; state (waiting to process). If the number
```

```
; of 'idle' processes is less than this
; number then some children will be created.
; pm.max_spare_servers - the maximum number of children in 'idle'
; state (waiting to process). If the number
; of 'idle' processes is greater than this
; number then some children will be killed.
; Note: This value is mandatory.
pm = dynamic

; The number of child processes to be created when pm is set to 'static' and the
; maximum number of child processes to be created when pm is set to 'dynamic'.
; This value sets the limit on the number of simultaneous requests that will be
; served. Equivalent to the ApacheMaxClients directive with mpm_prefork.
; Equivalent to the PHP_FCGI_CHILDREN environment variable in the original PHP
; CGI.
; Note: Used when pm is set to either 'static' or 'dynamic'
; Note: This value is mandatory.
pm.max_children = 50

; The number of child processes created on startup.
; Note: Used only when pm is set to 'dynamic'
; Default Value: min_spare_servers + (max_spare_servers - min_spare_servers) / 2
;pm.start_servers = 20

; The desired minimum number of idle server processes.
; Note: Used only when pm is set to 'dynamic'

; Note: Mandatory when pm is set to 'dynamic'
pm.min_spare_servers = 5

; The desired maximum number of idle server processes.
; Note: Used only when pm is set to 'dynamic'
; Note: Mandatory when pm is set to 'dynamic'
pm.max_spare_servers = 35

; The number of requests each child process should execute before respawning.
; This can be useful to work around memory leaks in 3rd party libraries. For
; endless request processing specify '0'. Equivalent to PHP_FCGI_MAX_REQUESTS.
; Default Value: 0
;pm.max_requests = 500

; The URI to view the FPM status page. If this value is not set, no URI will be
; recognized as a status page. By default, the status page shows the following
; information:
; accepted conn - the number of request accepted by the pool;
; pool - the name of the pool;
; process manager - static or dynamic;
; idle processes - the number of idle processes;
```

; active processes - the number of active processes;
; total processes - the number of idle + active processes.
; max children reached - number of times, the process limit has been reached,
; when pm tries to start more children (works only for
; pm 'dynamic')
; The values of 'idle processes', 'active processes' and 'total processes' are
; updated each second. The value of 'accepted conn' is updated in real time.
; Example output:
; accepted conn: 12073
; pool: www
; process manager: static
; idle processes: 35
; active processes: 65
; total processes: 100
; max children reached: 1
; By default the status page output is formatted as text/plain. Passing either
; 'html' or 'json' as a query string will return the corresponding output
; syntax. Example:
; foo.bar/status
; foo.bar/status?json
; foo.bar/status?html
; Note: The value must start with a leading slash (/). The value can be
; anything, but it may not be a good idea to use the .php extension or it
; may conflict with a real PHP file.
; Default Value: not set
;pm.status_path = /status

; The ping URI to call the monitoring page of FPM. If this value is not set, no
; URI will be recognized as a ping page. This could be used to test from outside
; that FPM is alive and responding, or to
; - create a graph of FPM availability (rrd or such);
; - remove a server from a group if it is not responding (load balancing);
; - trigger alerts for the operating team (24/7).
; Note: The value must start with a leading slash (/). The value can be
; anything, but it may not be a good idea to use the .php extension or it
; may conflict with a real PHP file.
; Default Value: not set
;ping.path = /ping

; This directive may be used to customize the response of a ping request. The
; response is formatted as text/plain with a 200 response code.
; Default Value: pong
;ping.response = pong

; The timeout for serving a single request after which the worker process will
; be killed. This option should be used when the 'max_execution_time' ini option
; does not stop script execution for some reason. A value of '0' means 'off'.
; Available units: s(econds)(default), m(inutes), h(ours), or d(ays)

```
; Default Value: 0
;request_terminate_timeout = 0

; The timeout for serving a single request after which a PHP backtrace will be
; dumped to the 'slowlog' file. A value of '0s' means 'off'.
; Available units: s(econds)(default), m(inutes), h(ours), or d(ays)
; Default Value: 0
;request_slowlog_timeout = 0

; The log file for slow requests
; Default Value: not set
; Note: slowlog is mandatory if request_slowlog_timeout is set
;slowlog = log/$pool.log.slow

; Set open file descriptor rlimit.
; Default Value: system defined value
;rlimit_files = 1024

; Set max core size rlimit.
; Possible Values: 'unlimited' or an integer greater or equal to 0
; Default Value: system defined value
;rlimit_core = 0

; Chroot to this directory at the start. This value must be defined as an
; absolute path. When this value is not set, chroot is not used.
; Note: you can prefix with '$prefix' to chroot to the pool prefix or one
; of its subdirectories. If the pool prefix is not set, the global prefix
; will be used instead.
; Note: chrooting is a great security feature and should be used whenever
; possible. However, all PHP paths will be relative to the chroot
; (error_log, sessions.save_path, ...).
; Default Value: not set
;chroot =

; Chdir to this directory at the start.
; Note: relative path can be used.
; Default Value: current directory or / when chroot
chdir = /

; Redirect worker stdout and stderr into main error log. If not set, stdout and
; stderr will be redirected to /dev/null according to FastCGI specs.
; Note: on highloaded environement, this can cause some delay in the page
; process time (several ms).
; Default Value: no
;catch_workers_output = yes

; Pass environment variables like LD_LIBRARY_PATH. All $VARIABLEs are taken from
; the current environment.
```

```
; Default Value: clean env
;env[HOSTNAME] = $HOSTNAME
;env[PATH] = /usr/local/bin:/usr/bin:/bin
;env[TMP] = /tmp
;env[TMPDIR] = /tmp
;env[TEMP] = /tmp

; Additional php.ini defines, specific to this pool of workers. These settings
; overwrite the values previously defined in the php.ini. The directives are the
; same as the PHP SAPI:
; php_value/php_flag - you can set classic ini defines which can
; be overwritten from PHP call 'ini_set'.
; php_admin_value/php_admin_flag - these directives won't be overwritten by
; PHP call 'ini_set'
; For php_*flag, valid values are on, off, 1, 0, true, false, yes or no.

; Defining 'extension' will load the corresponding shared extension from
; extension_dir. Defining 'disable_functions' or 'disable_classes' will not
; overwrite previously defined php.ini values, but will append the new value
; instead.

; Note: path INI options can be relative and will be expanded with the prefix
; (pool, global or /usr)

; Default Value: nothing is defined by default except the values in php.ini and
; specified at startup with the -d argument
;php_admin_value[sendmail_path] = /usr/sbin/sendmail -t -i -f www@my.domain.com
;php_flag[display_errors] = off
;php_admin_value[error_log] = /var/log/fpm-php.www.log
;php_admin_flag[log_errors] = on
;php_admin_value[memory_limit] = 32M
```

Wie Sie sehen benutzt der Pool den Port 9000 unter Localhost (127.0.0.1) und wird von dem Benutzer und der Gruppe www-data ausgeführt.

**Schauen Sie sich nun die PHP Konfiguration Ihres vhosts an:**

## Quellcode

1. vi /etc/nginx/sites-available/example.com.vhost

```
server {
[...]
location ~ .php$ {
try_files $uri =404;
fastcgi_pass 127.0.0.1:9000;
fastcgi_index index.php;
fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
fastcgi_param PATH_INFO $fastcgi_script_name;
```

include /etc/nginx/fastcgi_params;
}
[...]
}

Der wichtige Teil ist die Zeile fastcgi_pass 127.0.0.1:9000; - diese bewirkt, dass nginx PHP Anfragen an den PHP-FPM Prozess weitergibt, welcher den Port 9000 auf Localhost (127.0.0.1) benutzt - wie Sie sich sicher erinnern, ist das der in /etc/php5/fpm/pool.d/www.conf definierte Pool, was bedeutet dass Ihre PHP Skripte als Benutzer und Gruppe www-data ausgeführt werden.

**3 Einen individuellen Pool für jede Webseite definieren**
Meine Webseite example.com gehört dem Benutzer web1 und der Gruppe client0, deshalb möchte ich, dass meine PHP Skripte als dieser Benutzer und als diese Gruppe ausgeführt werden. Hierfür definiere ich einen neuen Pool /etc/php5/fpm/pool.d/example.com.conf:

**vi /etc/php5/fpm/pool.d/example.com.conf**

[example.com]

listen = 127.0.0.1:9001

listen.allowed_clients = 127.0.0.1

user = web1
group = client0

pm = dynamic
pm.max_children = 50
pm.start_servers = 20
pm.min_spare_servers = 5
pm.max_spare_servers = 35

chdir = /

Wie Sie sehen lasse ich diesen Pool den Port 9001 anstatt 9000 benutzen, außedem definiere ich als Benutzer web1 und als Gruppe client0.
Sie können so viele Pools wie Sie wollen definieren, stellen Sie jedoch sicher, dass Sie einen unverwendeten Port für jeden Pool benutzen (9002, 9003, etc.).

Laden Sie PHP-FPM neu:

**Quellcode**

1. /etc/init.d/php5-fpm reload

Ändern Sie nun Ihre vhost Konfiguration, sodass diese den gerade

erstellten Pool anstatt des alten benutzt. Alles was Sie dafür tun müssen ist, den Port in der Zeile fastcgi_pass zu ändern:

**Quellcode**

1. vi /etc/nginx/sites-available/example.com.vhost

```
server {
[...]
location ~ .php$ {
try_files $uri =404;
fastcgi_pass 127.0.0.1:9001;
fastcgi_index index.php;
fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
fastcgi_param PATH_INFO $fastcgi_script_name;
include /etc/nginx/fastcgi_params;
}
[...]
}
```

Laden Sie danach nginx neu:

**Quellcode**

1. /etc/init.d/nginx reload

Das war es. PHP Skripte werden nun als Benutzer web1 und als Gruppe client0 ausgeführt.

Sie können PHP noch sicherer machen, indem Sie einige Einstellungen individuell für jeden vhost vornehmen. Schauen Sie sich das Ende der Datei /etc/php5/fpm/pool.d/www.conf an, dort sind einige Beispiele angeführt, wie dies zu erreichen ist.

Zum Beispiel könnten Sie die Optionen open_basedir oder disable_functions im /etc/php5/fpm/pool.d/example.com.conf Pool ausnutzen.

**Quellcode**

1. vi /etc/php5/fpm/pool.d/example.com.conf

```
[example.com]

listen = 127.0.0.1:9001

listen.allowed_clients = 127.0.0.1

user = web1
group = client0

pm = dynamic
pm.max_children = 50
```

```
pm.start_servers = 20
pm.min_spare_servers = 5
pm.max_spare_servers = 35

chdir = /

php_admin_value[open_basedir] =
/var/www/www.example.com:/usr/share/php5:/tmp:/usr/share/phpmyadmin:/etc/phpmyadmin
:/var/lib/phpmyadmin
php_admin_value[disable_functions] =
dl,exec,passthru,shell_exec,system,proc_open,popen,curl_exec,curl_multi_exec,parse_ini_file
,show_source
```

Laden Sie dann PHP-FPM neu:

**Quellcode**

```
1. /etc/init.d/php5-fpm reload
```

3.1 Sockets anstatt von TCP Verbindungen benutzen
Bisher haben wir TCP Verbindungen für unseren PHP-FPM Pool benutzt (127.0.0.1:9000, 127.0.0.1:9001,
etc.). Dies verursacht einigen Overhead. Glücklicherweise können Sie für
Ihre Pools Unix Sockets anstatt von TCP Verbindungen benutzen und den
Overhead somit loswerden. Unix Sockets sind hier leistungsfähiger als
TCP Verbindungen.

Ich will, dass meine Sockets im /var/run/php5-fpm Verzeichnis erstellt werden, deshalb muss dieses Verzeichnis ersteinmal erstellt werden:

**Quellcode**

```
1. mkdir /var/run/php5-fpm
```

Um einen Unix Socket zu benutzen ändern Sie einfach die listen Zeile in Ihrer Pooldefinition, kommentieren die Zeile listen.allowed_clients aus oder löschen sie (diese macht nur für TCP Verbindungen Sinn) und fügen die Zeilen listen.owner (definiert den Besitzer des Sockets), listen.group (definiert die Gruppe des Sockets) und listen.mode (definiert die Zugriffsrechte des Sockets) hinzu:

**Quellcode**

```
1. vi /etc/php5/fpm/pool.d/example.com.conf
```

```
[example.com]

listen = /var/run/php5-fpm/example.com.sock

;listen.allowed_clients = 127.0.0.1
listen.owner = web1
listen.group = client0
```

listen.mode = 0660

user = web1
group = client0

pm = dynamic
pm.max_children = 50
pm.start_servers = 20
pm.min_spare_servers = 5
pm.max_spare_servers = 35

chdir = /

Laden Sie danach PHP-FPM neu:

**Quellcode**

```
1. /etc/init.d/php5-fpm reload
```

Schauen Sie sich nun das /var/run/php5-fpm Verzeichnis an:

**Quellcode**

```
1. ls -l /var/run/php5-fpm
```

Dort sollten Sie den Socket example.com.sock mit den Zugriffsrechten 0660 finden, der dem Benutzer web1 und der Gruppe client0 gehört:
root@server1:~# ls -l /var/run/php5-fpm

total 0

srw-rw---- 1 web1 client0 0 2011-09-21 11:08 example.com.sock

root@server1:~#

Zum Schluss müssen SIe noch die fastcgi_pass Zeile in Ihrem nginx vhost in fastcgi_pass unix:/var/run/php5-fpm/example.com.sock; umändern:

**Quellcode**

```
1. vi /etc/nginx/sites-available/example.com.vhost
```

server {
[...]
location ~ .php$ {
try_files $uri =404;
fastcgi_pass unix:/var/run/php5-fpm/example.com.sock;
fastcgi_index index.php;
fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
fastcgi_param PATH_INFO $fastcgi_script_name;
include /etc/nginx/fastcgi_params;

}
[...]
}

Starten Sie dann nginx neu:

**Quellcode**

1. /etc/init.d/nginx reload

Das war's!