Prozesse Forken mit PHP

Begriffsdefinition

Beim Forken wird ein zweiter identischer Prozess gestartet, während der erzeugende Prozess (auch Elternprozess genannt) weiter läuft. Alle Daten des ersten Prozesses, beispielsweise auch geöffnete Dateien, werden für den zweiten Prozess kopiert und stehen für diesen nun getrennt zur Verfügung. Jeder Prozess hat seinen eigenen Speicher. Der zweite Prozess ist ein vollwertiger Prozess ohne Einschränkung. Beide Prozesse können dann eigenständig weiterlaufen.

Inhaltsverzeichnis

Einschränkung

PHP muss als CLI oder CGI Modul ausgeführt werden. Bei der Ausführung als Apache Modul sind Threads (sinnvollerweise) nicht möglich.

Anwendungsfälle

Wenn PHP auf einer Webseite abläuft macht es nur bedingt Sinn Prozesse zu forken. Selbst wenn man asynchrone Aufrufe nutzt um, so würde man eine Server-Client-Architektur aus einem PHP-Controller und einem PHP-Modell (Backend) nutzen.
Nutzt man PHP im **Standalone Betrieb** zum Beispiel über den Command Line Interpreter (CLI) gibt es zum Beispiel Netzwerkanwendungen bei denen Forks Sinn machen. So könnte der Elternprozess auf eingehende Verbindungen warten und für jeden verbundenen Client einen Fork erstellen.
Ein weiterer Anwendungsfall ist die Implementierung einer **Job-Queue**. So könnte ein Hauptprozess für jeden Job einen eigenen Prozess abspalten.

Code

Der Programmablauf ist bis zum Aufruf von pcntl_fork() ganz normal. Nach dem Aufruf müsst ihr euch 2 Programme vorstellen, die ab dieser Stelle die Programmausführung fortsetzen.
Das eine "Programm" hat die Variabe $pid auf 0 gesetzt - das ist der Kindprozess. Bei dem anderen Programm ist die $pid im positiven Wertebereich. Die $pid stimmt mit der ProzessID im Betriebssystem überein.
Wir sammeln die Prozess-IDs der Kindprozesse in einem Array $pid_array und fragen deren Status mit waitpid ab. Sobald diese Prozesse terminieren lassen wir sie sterben.

Ohne den Aufruf von waitpid würden mehr und mehr Prozesse entstehen. In einem eigenen Projekt wurde nach 2 Tagen Laufzeit einen Fatal error: pcntl_fork(): Error 12 gemeldet, der vermutlich gesedet wird nachdem die maximale Anzahl möglicher Betriebssystemprozesse erreicht wird.

Quellcode

```
<?php
/**
* as you can see the "main"-process does not sleep
*/
function performSomeFunction($i) {
echo $i."\n";
sleep(5);
}
$i = 0;
$pid_arr = array();
while(true) {
sleep(1);
// any childs waiting to proving the dead?
foreach ($pid_arr as $pid => $i) {
// we are the parent
```

```
18.  $kill = pcntl_waitpid($pid, $status, WNOHANG);
19.  if($kill > 0) {
20.  unset($pid_arr[$pid]);
21.  }
22.  }
23.  $pid = pcntl_fork();
24.  if ($pid == -1) {
25.  die('could not fork');
26.  }
27.  else if ($pid) { // parent
28.  $pid_arr[$pid] = true;
29.  }
30.  else { // child
31.  performSomeFunction($i+1);
32.  exit(0);
33.  }
34.  $i++;
35.  }
36.  ?>
```

Alles anzeigen