簡単なCGI

ほとんどの場合、NGINXのCGIのサポートの欠如は問題ではなく、実際には重要な役得があります: NGINXは直接外部プログラム(CGI)を実行できないため、悪意のある人がシステムに任意のスクリプトをアップロードし実行することができません。There are still ways, of course (uploading a PHP script into a directory where you've got a directive to execute PHP FastCGI scripts for example), but it does in fact make it a bit more difficult (or conversely, easier to secure). でもまだ、サポートしなければならない1個限りのCGIプログラムがあることがあり、これがFastCGIの代わりとしてCGIをNGINXに公開するためのレシピです:

例えば、これがPerlで書かれたCGIプログラムです:

```
#!/usr/bin/perl
print "Content-type: text/html\n\n";
print "<html><body>Hello, world.</body></html>";
```

webサーバはファイルを読むことができ、実行可能としてマークされているようにしてください。

次に、FastCGIとして実行するPerlラッパーが必要で、これを実行します:

> "This one is still pretty ugly, but forwards stdio around correctly (not necessarily efficiently or prettily), and also reads and spits back out STDERR into fcgi STDERR. I am not entirely confident there is no race condition in the event that an app STDERRS out without any output data, but it works for my purposes. これは私の仕事ではなく大部分;99.9%はメーリングリストとこのwikiでのspawnerスクリプトの様々な改良の独力からできています。I just added some more pipe handling to enable communicating through STDERR, and a select loop to properly recieve both data streams." – bryon@spideroak.com

```
#!perl
use FCGI;
use Socket;
use FCGI::ProcManager;
sub shutdown { FCGI::CloseSocket($socket); exit; }
```

```
sub restart  { FCGI::CloseSocket($socket); &main; }
use sigtrap 'handler', \&shutdown, 'normal-signals';
use sigtrap 'handler', \&restart,  'HUP';
require 'syscall.ph';
use POSIX qw(setsid);

END()   { }
BEGIN() { }
{
  no warnings;
  *CORE::GLOBAL::exit = sub { die "fakeexit\nrc=" . shift() . "\n"; };
};

eval q{exit};
if ($@) {
  exit unless $@ =~ /^fakeexit/;
}
&main;

sub daemonize() {
  chdir '/' or die "Can't chdir to /: $!";
  defined( my $pid = fork ) or die "Can't fork: $!";
  exit if $pid;
  setsid() or die "Can't start a new session: $!";
  umask 0;
}

sub main {
  $proc_manager = FCGI::ProcManager->new( {n_processes => 5} );
  $socket = FCGI::OpenSocket( "/var/run/nginx/cgiwrap-dispatch.sock", 10 )
  ; #use UNIX sockets - user running this script must have w access to the
'nginx' folder!!
  $request =
  FCGI::Request( \*STDIN, \*STDOUT, \*STDERR, \%req_params, $socket,
  &FCGI::FAIL_ACCEPT_ON_INTR );
  $proc_manager->pm_manage();
  if ($request) { request_loop() }
  FCGI::CloseSocket($socket);
}
```

```
sub request_loop {
  while ( $request->Accept() >= 0 ) {
    $proc_manager->pm_pre_dispatch();

    #processing any STDIN input from WebServer (for CGI-POST actions)
    $stdin_passthrough = '';
    { no warnings; $req_len = 0 + $req_params{'CONTENT_LENGTH'}; };
    if ( ( $req_params{'REQUEST_METHOD'} eq 'POST' ) && ( $req_len != 0 ) )
{
      my $bytes_read = 0;
      while ( $bytes_read < $req_len ) {
        my $data = '';
        my $bytes = read( STDIN, $data, ( $req_len - $bytes_read ) );
        last if ( $bytes == 0 || !defined($bytes) );
        $stdin_passthrough .= $data;
        $bytes_read += $bytes;
      }
    }

    #running the cgi app
    if (
      ( -x $req_params{SCRIPT_FILENAME} ) &&    #can I execute this?
      ( -s $req_params{SCRIPT_FILENAME} ) &&    #Is this file empty?
      ( -r $req_params{SCRIPT_FILENAME} )       #can I read this file?
    ) {
      pipe( CHILD_RD,   PARENT_WR );
      pipe( PARENT_ERR, CHILD_ERR );
      my $pid = open( CHILD_O, "-|" );
      unless ( defined($pid) ) {
        print("Content-type: text/plain\r\n\r\n");
        print "Error: CGI app returned no output - Executing
$req_params{SCRIPT_FILENAME} failed !\n";
        next;
      }
      $oldfh = select(PARENT_ERR);
      $|     = 1;
      select(CHILD_O);
      $| = 1;
      select($oldfh);
      if ( $pid > 0 ) {
```

```
        close(CHILD_RD);
        close(CHILD_ERR);
        print PARENT_WR $stdin_passthrough;
        close(PARENT_WR);
        $rin = $rout = $ein = $eout = '';
        vec( $rin, fileno(CHILD_O),    1 ) = 1;
        vec( $rin, fileno(PARENT_ERR), 1 ) = 1;
        $ein    = $rin;
        $nfound = 0;

        while ( $nfound = select( $rout = $rin, undef, $ein = $eout, 10 ) )
{
          die "$!" unless $nfound != -1;
          $r1 = vec( $rout, fileno(PARENT_ERR), 1 ) == 1;
          $r2 = vec( $rout, fileno(CHILD_O),    1 ) == 1;
          $e1 = vec( $eout, fileno(PARENT_ERR), 1 ) == 1;
          $e2 = vec( $eout, fileno(CHILD_O),    1 ) == 1;

          if ($r1) {
            while ( $bytes = read( PARENT_ERR, $errbytes, 4096 ) ) {
              print STDERR $errbytes;
            }
            if ($!) {
              $err = $!;
              die $!;
              vec( $rin, fileno(PARENT_ERR), 1 ) = 0
              unless ( $err == EINTR or $err == EAGAIN );
            }
          }
          if ($r2) {
            while ( $bytes = read( CHILD_O, $s, 4096 ) ) {
              print $s;
            }
            if ( !defined($bytes) ) {
              $err = $!;
              die $!;
              vec( $rin, fileno(CHILD_O), 1 ) = 0
              unless ( $err == EINTR or $err == EAGAIN );
            }
          }
```

```
          last if ( $e1 || $e2 );
        }
        close CHILD_RD;
        close PARENT_ERR;
        waitpid( $pid, 0 );
      } else {
        foreach $key ( keys %req_params ) {
          $ENV{$key} = $req_params{$key};
        }

        # cd to the script's local directory
        if ( $req_params{SCRIPT_FILENAME} =~ /^(.*)\/[^\/] +$/ ) {
          chdir $1;
        }
        close(PARENT_WR);
        #close(PARENT_ERR);
        close(STDIN);
        close(STDERR);

        #fcntl(CHILD_RD, F_DUPFD, 0);
        syscall( &SYS_dup2, fileno(CHILD_RD),  0 );
        syscall( &SYS_dup2, fileno(CHILD_ERR), 2 );

        #open(STDIN, "<&CHILD_RD");
        exec( $req_params{SCRIPT_FILENAME} );
        die("exec failed");
      }
    } else {
      print("Content-type: text/plain\r\n\r\n");
      print "Error: No such CGI app - $req_params{SCRIPT_FILENAME} may not
exist or is not executable by this process.\n";
    }
  }
}
```

上のスクリプトを/usr/local/bin/cgiwrap-fcgi.plとして保存します。

上のプログラムを単に実行すると、それを/var/run/nginx/cgiwrap-dispatch.sockでunixソケットにバインドするでしょう。NGINXワーカープロセスのユーザがこのファイルにread/writeアクセスを

持つようにしてください。スクリプトは自身ではforkしないため、それをどうにかしてバックグランドにする必要があります(Bashを使って実行するコマンドの最後にアンパーサンド"&"を追加します)。

これが全てうまく行く場合は、次の部分はNGINXのセットアップです:

```
http {
  root  /var/www/htdocs;
  index index.html;
  location ~ ^/cgi-bin/.*\.cgi$ {
    gzip off; #gzip makes scripts feel slower since they have to complete
before getting gzipped
    fastcgi_pass  unix:/var/run/nginx/cgiwrap-dispatch.sock;
    fastcgi_index index.cgi;
    fastcgi_param SCRIPT_FILENAME /var/www/cgi-bin$fastcgi_script_name;
    fastcgi_param QUERY_STRING     $query_string;
    fastcgi_param REQUEST_METHOD   $request_method;
    fastcgi_param CONTENT_TYPE     $content_type;
    fastcgi_param CONTENT_LENGTH   $content_length;
    fastcgi_param GATEWAY_INTERFACE  CGI/1.1;
    fastcgi_param SERVER_SOFTWARE    nginx;
    fastcgi_param SCRIPT_NAME        $fastcgi_script_name;
    fastcgi_param REQUEST_URI        $request_uri;
    fastcgi_param DOCUMENT_URI       $document_uri;
    fastcgi_param DOCUMENT_ROOT      $document_root;
    fastcgi_param SERVER_PROTOCOL    $server_protocol;
    fastcgi_param REMOTE_ADDR        $remote_addr;
    fastcgi_param REMOTE_PORT        $remote_port;
    fastcgi_param SERVER_ADDR        $server_addr;
    fastcgi_param SERVER_PORT        $server_port;
    fastcgi_param SERVER_NAME        $server_name;
  }
}
```

NGINXを再起動し、ブラウザをCGIプログラムに向けます。The above sample config will execute any .cgi file in `cgi-bin` with the `cgiwrap-fcgi.pl` wrapper, tweak this to your heart's content.

これを使ってPython, Perl, および C++ のGETあるいはPOSTを使うcgi アプリケーションを実行することができました

You may find that `$document_root` does not point to your actual document root (hardcoded upon build), so you can replace the fastcgi param DOCUMENT_ROOT with "/the/real/path". (上でしているように)CGIラッパーによって使われる場合は、SCRIPT_FILENAMEパラメータも置き換えてください