apache2でHTTP PUTすると、405 Method Not Allowed

どのバージョンから規制されたのか不明ですが、以下の記事によると
defaulthandler()は、GET/POST/OPTIONS以外は、受け付けないようになっているらしい。
https://www.softel.co.jp/blogs/tech/archives/5685

セキュリティの観点から、Limitディレクティブを使って、GET/POTのみ受け付けるように制限を
かける事が言われていて、Googleと、それらの記事も多くHitする。
ただ、どの記事も日付が古い。Apache(httpd)側で対策したから、わざわざ制限を加えなくても
良いとの観点だろう。

今回、ファイルをサーバー側に置いて処理させる要件が出て、HTTPかSCP/SFTPのどちらかで
実装だろうな・・・と思っており、HTTP PUTのほうが単純で(実装が)楽じゃね？と思った次第。
当初は、httpd.confでPUTメソッドを許可する設定を書けば良いと思っていた。
だが、難航極まりない。結局、行き着いたのが上記URLで、設定では許可出来無さそうだった。
その記事で「Scriptディレクティブで設定するのが簡単」とあったので、やってみた。

***** 注意 *****
以下にサンプルスクリプトを掲載するが、あくまでサンプル。
そのままコピペして公開サーバーで実装しないで欲しい。
公開サーバーへ実装を予定されている場合は、実装予定のサーバーの環境に応じて、
有識者がスクリプトのセキュリティ診断を行うなど対応が必要である。

○ 試した環境

・ WEBサーバー (192.168.100.100)
  Debian GNU/Linux 11 Debian 5.10.197-1 (2023-09-29) x86_64
  apache2 2.4.56-1~deb11u2
  perl v5.32.1
・ クライアント (192.168.100.1)
  Rocky Linux 8.9 4.18.0-513.5.1.el8_9.x86_64
  perl v5.26.3

○ インストール

[WEBサーバー]

```
# apt-get install apache2
# a2enmod actions
# a2enmod cgid
# mkdir /var/www/html/test
# chown www-data. /var/www/html/test
# apt-get install libcgi-pm-perl

# vi /etc/apache2/sites-enabled/000-default.conf
  変更箇所は以下の通り

*** 000-default.conf.bak
--- 000-default.conf
***************
*** 27,32 ****
--- 27,39 ----
# after it has been globally disabled with "a2disconf".
#Include conf-available/serve-cgi-bin.conf
+
+
```

```
# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
```

vi /usr/lib/cgi-bin/put.cgi

```
#!/usr/bin/perl -T

use strict;
use warnings;
use CGI;
use File::Basename;
use CGI::Carp qw(fatalsToBrowser);

$CGI::POST_MAX = 1024 * 100;

my $q       = CGI->new;
my $method  = $q->request_method();
my $redirect= $ENV{'REDIRECT_URL'};
my $data    = '';
my $msg     = '';
my $ref_base= '';
my $ref_dir = '';
my $save_nm = '';

unless ( $redirect eq '' ) {
    ($ref_base, $ref_dir) = fileparse $redirect;
    if ( $method eq 'PUT' and $ref_dir =~ m!^/test/! ){
        $data  = $q->param('PUTDATA');
        if ($ref_base =~ /^([\w\.-]+)$/) { $ref_base = $1; }
        else { die 'Unable to extract file name';  }

        $save_nm = "/var/www/html/test/${ref_base}";
        open(OUT, ">$save_nm") or die "File ($save_nm) open Error\n";
        print OUT $data;
        close OUT;
        $msg = "$save_nm write";
    }
    else {
        $msg = 'Method Error';
    }
}
else {
    $msg = 'Direct execution prohibited';
}

# HTML
print $q->header,
  $q->start_html('hello world'),
  $q->p("$msg") . "\n";
print $q->end_html;
```

systemctl restart apache2

[クライアント]

$ vi test.pl

```
#!/usr/bin/perl

use strict;
use warnings;
use HTTP::Request;
use LWP::UserAgent;

my $method = 'PUT';
my $uri    = 'http://192.168.100.100/test/test.txt';
my $header = ['Content-Type' => 'text/plain; charset=UTF-8'];
my $data   = 'Hello Wolrd';

my $req = HTTP::Request->new( $method, $uri, $header, $data );
my $ua  = LWP::UserAgent->new(timeout => 10);
my $res = $ua->request($req);
print $res->content();
```

$ chmod 744 test.pl

○ 実行してみる

(1) クライアントよりコマンド実行
$ ./test.pl

PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

/var/www/html/test/test.txt write

(2) サーバー側にログインしてファイルが出来ている事を確認

(3) クライアントよりブラウザで、http://192.168.100.100/test/test.txt にアクセスしてみる

○ 感想

結局、CGI(put.cgi)を動かしているので、CGI.pm
使っている時点で、POSTでも良かったかも・・・と気づいた。
今回のWEBサーバーはphp入れていないけど、mod_phpなんて入って入たら、test.phpを置くと実行できるのでは？
/var/www/html/test ディレクトリは、PUTメソッドのみ許可にしてブラウジング出来ないようにするとか、
そもそもドキュメントルート以外に保存して閲覧用のCGIを別途用意するとか、対策施さないと危険すぎる。
text.txtファイルのパーミッションは[0644]なので、シェルインジェクションは出来ないと思うけど。

添付ファイル::

一意的なソリューション ID: #1040
製作者: n/a
最終更新: 2023-11-30 08:15