

[NT] Security Risk When Using the CGI Binary (PHP.EXE) Under Apache

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2002-01/0009.html>

From: support@securiteam.com

Date: 01/04/02

From: support@securiteam.com

To: list@securiteam.com

Date: Fri, 4 Jan 2002 00:27:14 +0100 (CET)

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

When was the last time you checked your server's security?

How about a monthly report?

<http://www.AutomatedScanning.com> -- Know that you're safe.

Security Risk When Using the CGI Binary (PHP.EXE) Under Apache

SUMMARY

As advised in the installation text that comes with all versions of PHP, when installing PHP.EXE for use on a windows machine installed with Apache, the user should insert a few lines of code into the Apache "httpd.conf". These exact lines are shown here:

```
ScriptAlias /php/ "c:/php/"
AddType application/x-httpd-php .php
Action application/x-httpd-php "/php/php.exe"
```

A security vulnerability arises when placing the ScriptAlias line above. This line effectively maps the alias /php/ to your web document root such that typing "<http://www.example.com/php/>" will actually try to access in this case "c:\php\". Please note that the last "/" on the end of the URL has to exist for this to work ("<http://www.example.com/php>" will not work). At this point your server will respond with "Access Denied", however if you now specify the URL "<http://www.example.com/php/php.exe>", you will see the error "No input file specified". This error is actually returned by php.exe, which you have just executed on the server.

Securiteam: [NT] Security Risk When Using the CGI Binary (PHP.EXE) Under Apache

There are many exploits that can happen with this setup (some very serious, which could be used to gain root access).

DETAILS

Exploit 1:

It is possible to read any file remotely on the server, even across drives with the following URL construct:

```
"http://www.example.com/php/php.exe?c:\winnt\repair\sam"
```

PHP.EXE will parse the sam file "c:\winnt\repair\sam" and return it to the browser for download (this is the Windows NT password file).

```
"http://www.example.com/php/php.exe?d:\winnt\repair\sam"
```

PHP.EXE will return the same file on the D: drive.

The above SAM file can then be used to decrypt all the Account Passwords for the Server.

Exploit 2:

If you specify a file that exists in the php directory (different files exist depending on the version of PHP), the web server will try to execute this file and will throw back an error reporting the install directory of php. So in PHP4, for example, you would specify the following line:

```
"http://www.example.com/php/php4ts.dll"
```

The error returned by the web server would be: " couldn't create child process: 22693: C:/php/php4ts.dll " showing the install path of PHP.

ADDITIONAL INFORMATION

The information has been provided by
<mailto:brereton_paul@btopenworld.com> Paul Brereton.

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:

list-unsubscribe@securiteam.com

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====
=====

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.

- *Previous message:* support@securiteam.com: "[NT] Internet Explorer GetObject() Problems"
- *Messages sorted by:* [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#) [\[attachment \]](#)