

Serious PHP memory_limit remote vulnerability

2004-07-14, xiando

Apache2 users who use php and limites how much memory a script may consume with memory_limit in php.ini should consider upgrading to php version 4.3.7, previous versions may allow remote attackers to execute arbitrary code.

The hole affects all apache2 installations using PHP <= 4.3.7 and PHP5 <= 5.0.0RC3.

The hole was, like most Open Source vulnerabilities, plugged extremely quickly and packages for most distributions were available before it was publicly released.

- Gentoo Users: PHP 4.3.7 has been available in portage since 2004-07-04. `emerge mod_php php` to upgrade.
- Suse users: Updated packages for Suse became available 2004-07-16, [SUSE-SA_2004_021.txt](#).
- The Full [e-matters GmbH Security Advisory: 112004.txt](#)

e-matters GmbH
www.e-matters.de

-- Security Advisory --

Advisory: PHP memory_limit remote vulnerability
Release Date: 2004/07/14
Last Modified: 2004/07/14
Author: Stefan Esser [s.esser@e-matters.de]

Application: PHP <= 4.3.7

Serious PHP memory_limit remote vulnerability (Linux Reviews)

PHP5 <= 5.0.0RC3

Severity: A vulnerability within PHP allows remote code execution on PHP servers with activated memory_limit

Risk: Critical

Vendor Status: Vendor has released a bugfixed version.

Reference: <http://security.e-matters.de/advisories/112004.htm>

Overview:

PHP is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into

According to Security Space PHP is the most popular Apache module and is installed on about 50% of all Apaches worldwide. This includes of course only those servers that are not configured with `expose_php=Off`.

During a reaudit of the memory_limit problematic it was discovered that it is possible for a remote attacker to trigger the memory request termination in places where an interruption is unsafe. This can be abused to execute arbitrary code on remote PHP servers.

Details:

On the 28th June 2004 Gregori Guninski released his advisory about a possible remote DOS vulnerability within Apache 2 (CAN-2004-0001). This vulnerability allows tricking Apache 2 into accepting arbitrarily sized HTTP headers. Guninski and many others rated this bug as "High Risk" for 32bit systems, but they did not take into account that such a bug could have a huge impact on 3rd party modules.

After his advisory was released I reaudited PHP's memory_limit request termination, because this bug made it possible to reach memory_limit at places that were never meant to be interrupted. After a possible exploitation path for Apache 2 servers was discovered and a working exploit was created, similar paths were found and added to the proof of concept exploit that allowed exploitation of NON Apache 2 servers. (f.e. Apache 1.3.31)

The idea of the exploit is simple. When PHP allocates a block

Serious PHP memory_limit remote vulnerability (Linux Reviews)

memory it first checks in the cache of free memory blocks for one of the same size. If such a block is found it is taken from there, otherwise PHP checks if an allocation would violate the memory_limit. In that case the request shutdown is triggered through zend_error() (PHP < 4.3.7 aborts after the violating memory block is allocated). PHP contains several places where such an interruption is unsuitable. An example for such places are those where Zend HashTables are allocated and initialised. This is performed in 2 steps and the first initialisation step itself allocates memory before important data structures are correctly initialised. An attacker that is able to trigger a memory_limit abort within zend_hash_init() and is additionally able to control the heap before the HashTable itself is allocated, is able to supply his own HashTable destructor pointer.

Several places within PHP were found where this action is possible on HashTables that actually get destructed by the request shutdown. One of such places is f.e. within the fileupload code, but is not triggerable on Apache 2 servers that are vulnerable to CAN-2006-2000, another one is only reachable if variables_order was changed to contain the "E" in the end, a third one is within session extension which is activated by default but the vulnerability can not be triggered if the session functionality is not used. A fourth place is within the implementation of the register_globals functionality. Although this is deactivated by default since PHP 4.2 it is activated on nearly all servers that have to ensure compatibility with older scripts. Other places might exist in not default activated or 3rd party extensions.

All mentioned places outside of the extensions are quite easy to exploit, because the memory allocation up to those places is deterministic and quite static throughout different PHP versions. The only unknown entity is the size of the environment vars array. But that is usually small and can be bruteforced with some kind of binary search algorithm. Additionally this information could be leaked to an attacker through an open phpinfo() page. If the above mentioned php.ini-recommended as configuration basis it is irrelevant anyway because the ENV array is not populated in that case.

Because the exploit itself consists of supplying an arbitrary destructor pointer this bug is exploitable on any platform. (Except the system runs with non exec heap+stack protection) This includes systems running Hardened-PHP <= 0.1.2 because t

Serious PHP memory_limit remote vulnerability (Linux Reviews)

have no protection of the HashTable destructor pointer.

As a last word it should be said, that an attacker does not need to send 8/16/64MB (or whatever the memory_limit is) per attack. With POST requests it is quite easy to eat 100 (and more) times the amount of sent bytes.

Proof of Concept:

e-matters is not going to release an exploit for this vulnerability to the public.

Disclosure Timeline:

- 07. July 2004 - Vendor-sec was informed about the fact that this vulnerability was found
- 14. July 2004 - Public Disclosure

CVE Information:

The Common Vulnerabilities and Exposures project (cve.mitre.org) assigned the name CAN-2004-0594 to this issue.

Recommendation:

If you are running PHP with compiled in memory_limit support, it is strongly recommended that you upgrade as soon as possible to the newest version. Disabling memory_limit within your configuration can be considered a workaround, but leaves your site vulnerable to memory hungry PHP scripts or POST requests that create huge vulnerabilities. If you are running PHP with Apache <= 2.0.49 ensure that you have the fix for CAN-2004-0493 applied.

GPG-Key:

http://security.e-matters.de/gpg_key.asc

Serious PHP memory_limit remote vulnerability (Linux Reviews)

pub 1024D/3004C4BC 2004-05-17 e-matters GmbH - Securityteam
Key fingerprint = 3FFB 7C86 7BE8 6981 D1DA A71A 6F7D 572D 30

Copyright 2004 Stefan Esser. All rights reserved.

> [Linux Reviews](#) > [News and headlines](#) > [2004 News archive](#) > [July](#) >
Serious PHP memory_limit remote vulnerability