Background

The Apache HTTP Server is an open-source HTTP server for modern operating systems including UNIX, Microsoft Windows, Mac OS/X and Netware. The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services observing the current HTTP standards. Apache has been the most popular web server on the Internet since April of 1996.

Problem Description

During routine testing, an integer overflow was found in apache2-mpm-worker 2.2.19 in the function ap_pregsub called from mod-setenvif. The issue affects all versions from XXXX up to 2.2.20, not depending on the mode of operation (worker, prefork, ..). When a header field is mangled using *SetEnvIf*, the new environment variable data can be multiples of the size of the submitted header field. When *ap_pregsub* from *server/util.c* calculates the buffer size using

```
        else if (no < nmatch && pmatch[no].rm_so < pmatch[no].rm_eo) {
            len += pmatch[no].rm_eo - pmatch[no].rm_so;
        }
```

the length value overflows and is used in a subsequent allocation call of buffer too small:

```
    dest = dst = apr_pcalloc(p, len + 1);
```

The subsequent filling of the buffer with user-supplied data leads to buffer overflow. Even without overflowing, the allocation of significant amounts of server memory for excessivly large environment variables should be considered a problem also.

Impact

Depending on the input data, exploitation of this issue leads to:

- allocation of large quantities of server memory, killing processes due to out-of-memory conditions or reducing system performance to crawl due to massive swapping.

- invalid memory access when copying more than 4GB of data into the much smaller buffer. Since the loop copying the data uses only stack and libc-heap, not the apr pool, for source and destination addresses, copy process is linear, starting at low address and pool is separated by unaccessible memory pages for protection on linux. Usually this will only cause termination of the apache process, which is restarted automatically. The impact is increased system load and DOS-condition while under attack.
- At least with multi-threaded server (worker), arbitrary code execution is proven, on single-threaded varians, the use of crafted stop-sequences might allow code execution even on these systems. On many systems ASLR will reduce the efficiency of the attack, but even with ASLR enabled, the automatic restart of processes allows to probe for all possible mappings of libc. An attacker, that has already access to another account on the machen, might be able to use ApacheNoFollowSymlinkTimerace to learn the memory map of the process, thus having the posibility to reach nearly 100% efficiency.

To trigger this issue, mod_setenvif must be enabled and the attacker has to be able to place a crafted *.htaccess* file on the server. Since the triggering of the exploit might depend on a *magic header field*, the malicious *.htaccess* might be placed as backdoor in web-content .zip files or could be stored dormant on the server until activation by the corresponding magic request.

Workaround

On workaround is to disable the module when not needed. Another one is to disable use of *.htaccess* on all user-modificable locations by setting *AllowOverride None* on all corresponding directories.

Solution

The fix for the integer overflow in ap_pregsub (server/util.c,) is tivial, e.g.

```
        oldlen=len=0
        else if (no < nmatch && pmatch[no].rm_so < pmatch[no].rm_eo) {
            len += pmatch[no].rm_eo - pmatch[no].rm_so;
// Return NULL on error
```

```
            if(len<oldlen) return(NULL);
        }
```

Apart from the overflow, a discussion on reasonable sizes for environment variables should be started to deal with the memory consumptin problem. Without limits, another problematic section is in ap_pregsub (server/util.c,) should be fixed also. At the moment, the return value from from apr_pcalloc is not checked, leading to a NULL-pointer dereference. Fix

```
    dest = dst = apr_pcalloc(p, len + 1);
    if(!dest) return(NULL);
```

Timeline

- 20110715: Initial discovery, report to apache security
- 20110716: Ubuntu bug report 811422
- 20111011: Feedback from apache security, impact low, go ahead with preparation of advisory.
- 20111031: Feedback from apache security on advisory, OK for public release
- 20111102: Public release via bugtraq and full disclosure

References

- CVE: CVE-2011-3607
- Ubuntu bug report: 811422
- apache bug ID?
- Demo exploit: http://www.halfdog.net/Security/2011/ApacheModSetEnvIfIntegerOverflow/DemoExploit.html

Last modified 20111102
Contact e-mail: me (%) halfdog.net