

NetSec Security Advisory: Multiple Vulnerabilities Resulting From Use Of Apple OSX HFS+

Source: <http://www.derkeiler.com/Mailing-Lists/securityfocus/bugtraq/2005-02/0308.html>

From: TAC (tac_at_netsec.net)

Date: 02/16/05

Date: Wed, 16 Feb 2005 04:58:24 -0500

To: <bugtraq@securityfocus.com>

NetSec Security Advisory

VULNERABILITY DETAILS

Name: Multiple Vulnerabilities Resulting From Use Of Apple OSX HFS+

Impact: HIGH

Platform: Apple OS X (Darwin) <= 10.2

Method: Possible unauthorized access to file system data

Identifier: 07012005-01

FORWARD:

In December 2004, NetSec released details of a vulnerability impacting software running on versions of Apple OS X version 10.2 and greater. Under OS X, userland applications are presented with two interface methods to an underlying legacy HFS+ file system: resource and data streams. Access of the individual streams from a file browser or shell application permits users with appropriate access rights to retrieve information from the data fork (content) or resource fork (resources).

The risk associated with any unauthorized file data disclosure to remote users is often significant. This is because users may access the source code of server-side interpreted scripts that may contain embedded database credentials, specify known paths to sensitive files (shell command history files for example), retrieve hidden files, and otherwise retrieve arbitrary file content. All of these exploitation scenarios may bypass default server access controls, unless requests for the data and resource forks are trapped prior to request forwarding.

Subsequent research and testing, conducted by NetSec revealed at least one method to leverage this 'feature' of the legacy HFS+ driver in OS X: web services. The default configuration of several web server

applications does not adequately prevent remote access to these protected file system resources. Testing of other network-enabled applications did not result in the identification of other vectors; however any server application that does not proactively filter requests for local file system resources may expose underlying HFS+ file systems to unauthorized remote access.

In the initial design requirements for the HFS file system, exporting data across the network using a minimal abstraction layer was not considered. The targeted use for Apple systems pre – OS X has largely been desktop publishing, graphics, and other multimedia services. NetSec considers the emergence of this disclosure vulnerability evidence of "growing pains" associated with the recent Apple platform migration to Berkeley Unix (BSD).

The HFS+ file system is not recommended for dedicated servers, but is required to support numerous legacy Macintosh applications. At the time of this technical advisory, NetSec strongly recommends that organizations with public Internet-facing Apple servers consider migration to the Berkeley Fast File System (FFS/UFS) option available in OS X.

The purpose of this security advisory is to increase community awareness of potential risks associated with the Apple HFS+ file system as implemented under OS X. It should be noted that NetSec exercises responsible disclosure policy and has been in close contact with all software vendors referenced herein. Specific examples should not be interpreted as the full extent of affected server applications. Please contact NetSec at info@netsec.net or reply to this message if you have any additional questions about this issue.

SUMMARY:

Apple's HFS and HFS+ file systems allows two separate data streams for each file, referred to as the "data fork" and "resource fork". The classic MacOS operating systems and Carbon API on MacOS X provide separate functions for opening and manipulating the data and resource forks. In MacOS X, however, support for addressing these separate streams has been integrated into the POSIX API. In MacOS X 10.2 and above, opening the file by its pathname opens the data fork, but the data fork or resource fork may also be opened for a given file by respectively appending "../namedfork/data" or "../namedfork/rsrc" to the pathname passed to the open(2) system call. In previous versions, they may be addressed by appending the special pathnames "__Fork/data" or "__Fork/rsrc". The resource fork may also be opened in most versions of MacOS X by appending "/rsrc" to the file pathname.

Due to this feature being available throughout the operating system, via the POSIX API, it is therefore available to any software involved in the opening of file streams via the open() syscall, such as a web server opening an html or PHP file present on the Darwin servers file system.

As a result, server daemons, such as web servers which open file streams, based on user controlled data, may be fooled into opening the respective files resource and/or file fork rather than the absolute file name. This may allow users to view arbitrary data, such as the source code of server interpreted documents (such as PHP and JSP files).

IMPACT:

Remote users may be able to view arbitrary file data, including the source code of server side documents, such as PHP JSP documents. This data may contain sensitive information such as database usernames and passwords and/or disclose vulnerabilities to an attacker which can then be leveraged to further attack the respective web application.

It should be noted that this issue extends to any server software running on the Darwin operating system, which is involved in the opening of file streams, based on user input.

VENDOR STATUS:

NetSec have been in touch with several software vendors, whose products are affected by the OSX kernel feature.

Apache Foundation (Apache Web Server)

The Apache Foundations HTTPD server project is known to be vulnerable to the issues resulting from the use of the Apple OSX HFS+ file system. Users of OSX, who have not modified the default Apache configuration file (httpd.conf) can install a mod_rewrite work around through installing the OSX update, made available by Apple in December 2004. See: <http://docs.info.apple.com/article.html?artnum=300422> for more information.

Users using Apache to serve files from an HFS+ file system, who have modified the configuration file by hand (such as default users of OSX 10, who have made changes to the Apache configuration) can add the following mod_rewrite rule to their httpd.conf as a work-around to the issues inherent to the use of an HFS+ file system as a web root:

```
<Files "rsrc">  
    Order allow,deny  
    Deny from all  
    Satisfy All  
</Files>
```

```
<DirectoryMatch ".*\\.\\.namedfork">  
    Order allow,deny  
    Deny from all  
    Satisfy All
```

</DirectoryMatch>

The above mod_rewrite rules, reflect those recommended by Apple, and installed by the Apple update, in cases where the Apache configuration file has not been modified.

Prior to the use of mod_rewrite, users of versions of Apache prior to 1.3.29 should ensure that they are not vulnerable to the stack overflow conditions, as described in CAN-2003-0542.

4D (WebStar Web Server V)

Vulnerable: 4D WebStar 5.3.4 and below.

4D's WebStar web server was found to be vulnerable to the issues resulting from the use of the OSX HFS+ file system. 4D have acknowledged the existence of the issue and have implemented a fix in the latest version of their product.

Further fix information for WebStar's product is available at:

http://www.4d.com/products/downloads_4dws.html

and:

ftp://ftp.4d.com/ACI_PRODUCT_REFERENCE_LIBRARY/4D_PRODUCT_DOCUMENTATION/PDF_Docs_by_4D_Product_A-Z/4D_WebSTAR/About_5.3.4_Web_Security_Update.pdf

Roxen (Roxen Web Server)

Vulnerable: Roxen Web Server Version 4.0.172 and below.

Roxen's freely available web server product was found to be vulnerable to the issues resulting from the use of the OSX HFS+ file system. Roxen have acknowledged that the issues result in an exploitable condition and are currently working on a fix to in order to remedy the issues. Release 4 of the web server software is due for release in mid-January 2005 and will be available from: <http://download.roxen.com/4.0/>

Roxen (Roxen Content Management System)

Vulnerable: Roxen Web Server Version 4.0 Release 3 and below.

Roxens Content Management System (Roxen CMS), which is based around the freely available Roxen web server, is also affected by the vulnerability caused by the use of an OSX HFS+ file system. Research concluded that, although specific, customised applications running on Roxen CMS may be affected in a more severe manner, default configurations are not exploitable in the same manner in which these issues have found to be exploitable on other web server suites (see technical information for details). This said, attempts to access HFS+ named fork files via the CMS interface result in an error being thrown by the CMS software, which may cause the application to behave in an undesired manner.

Roxen have also commented that (as previously discussed in this advisory), the fact that the HFS+ file system behaves in a case-insensitive manner; remote users may also be able to access sensitive files through bypassing the case-sensitive access controls that Roxen provides users with. Roxen have stated that a work around for this issue will also be provided in release four of the CMS product.

As with the freely available Roxen web server, Roxen CMS 4, R4 will be made available to customers by mid-January 2004. The fixed, trial version of the Roxen CMS product will be made available from: http://www.4d.com/products/downloads_4dws.html

TECHNICAL DETAIL:

Side note:

It should be noted that NetSec will not release technical information until the point at which its responsible disclosure policy has been satisfied and/or NetSec believes that sufficient technical and/or exploitation vector related information already exists within the public domain. In the case of the issues described herein, both of the above were found to be true.

The following extract of code is taken from the forkcomponent() function, which is part of the OSX HFS+ file system kernel module:

```
/*
 * There are only 3 valid fork suffixes
 * "../namedfork/rsrc"
 * "../namedfork/data"
 * "/rsrc" (legacy)
 */
```

As you can see, the HFS+ file system implements three fork named file system forks, two of which pertain to the resource fork (suffixed with rsrc) and the third, pertaining to the file data named fork. Because of the case-insensitivity of the HFS+ file system, these can be accessed in a number of ways (such as /rsrc or /rSrC), hence, any programmatic checks for attempts to access resource forks must also occur in a case insensitive manner (via constructs such as tolower() for example).

Two attack impacts have been identified, through the above semantics of HFS+:

- 1) Software may be tricked into mis-interpretating files, through accessing their respective data stream directly, for example, `/filename/..namedfork/data` as opposed to `/filename`.
- 2) Software, implementing an internal access control systems may be bypassed through either implementing access controls in a case sensitive manner (when HFS+ behaves case-insensitivity) and/or accessing the files respective files data steam directly. For example, a file access control system preventing access to `/filename`, may be bypassed through constructing a request to `/filename/..namedfork/data`. Such is the case, when an attempt is made to access an Apache ".htaccess" file, via the request construct:

```
GET /path/.htaccess/..namedfork/data HTTP/1.0
```

Where access to .htaccess files should otherwise be denied.

As noted above, the following http server suites have been tested to be affected by the HFS+ issues in this manner:

4D (WebStar Web Server V)
Roxen (Roxen Web Server)
Apache Foundation (Apache Web Server)

Although Roxens's Content Management System (CMS) is largely based around the open source Roxen HTTP Server, it was not found to be vulnerable to the HFS+ arbitrary data access issue in this manner. This was for the following reasons:

- i) By default, the CMS system will only permit access to files ending with certain file extensions (such as .xml). Note: This can be bypassed through issuing requests to files using a null byte, such as `/path/filename.forbidden/..namedfork/data%00.xml`
- ii) Prior to serving a file to a web client, the CMS interface will attempt to `chdir()` to the respective directory, containing the file requested. In the case of a request to:

```
/path/filename.xml/..namedfork/data
```

the CMS daemon will attempt to `chdir()` to the path:
`/path/filename.xml/..namedfork/` which of course will not exist. Unfortunately, the `chdir()` is the second of two operations which the CMS will perform in order serve the file, the first being a `stat()` of the respective file to be served. Due to the `stat()` succeeding and the `chdir()` failing, an exception will be thrown by the CMS daemon.

Although no obvious way exists to leverage this, such unexpected behaviour may result in the manifestation of a security flaw in customised applications, running on the CMS.

Whilst the more obvious attack vectors pertain to HTTP servers (as above), it is important to note that HTTP servers are not the only group of server software which will be impacted by the use of the HFS+ file system on Apple OSX. In essence, any software attempting to access files on an HFS+ file system may be affected in the two ways noted above. To this end, developers should thoroughly test their software and analyse the affects of attempts to access HFS+ named forks by either remote or local users. NetSec are aware of several non-http server daemons which are negatively impacted by the use of HFS+, but do not intend to release details, given that the respective software publishers have yet to remedy these issues.

ATTACK DETECTION:

As described above, flaws resulting from the HFS+ named fork issue can be exploited through http servers to disclose the contents of files, intended for server-side interpretation, such as files intended for interpretation by the hyper-text pre-processor – PHP. The following request may be made to a vulnerable web server, to disclose the potentially sensitive contents of "test.php":

Expected behaviour:

```
# curl http://127.0.0.1/path/test.php  
hello world<br>  
#
```

Expected behaviour if vulnerable:

```
# curl http://127.0.0.1/path/test.php/.namedfork/data  
<? print "hello world<br>\n"; ?>  
#
```

To this end, web logs will display a request for a file, similar to the following:

```
a.b.c.d -- [02/Jan/2005:13:33:37 +0100] "GET  
/path/test.php/.namedfork/data HTTP/1.0" 200 <size>
```

As part of a comprehensive risk management strategy, NetSec recommends routine evaluation of anomalous web application log entries to detect these types of exploitation attempts. Customer security devices managed under NetSec's Managed Security Services received custom signatures to detect this attack as of mid-November 2004.

Please contact the NetSec Security Operations Center if you have any questions regarding this security issue.

--

NetSec Security Operations Center
13525 Dulles Technology Drive
Herndon, VA 20171
866.444.6762 - Toll Free (North America)
+001.703.561.9042 - Phone (International)
+001.703.561.0426 - Fax
Corporate Site: <http://www.netsec.net>
Managed Security | Business Relevance