

# [VulnWatch] AFFLIB(TM): Multiple Format String Injections

---

*Source:* <http://www.derkeiler.com/Mailing-Lists/VulnWatch/2007-04/msg00034.html>

---

- *From:* VSR Advisories <[advisories@xxxxxxxxxxxxxx](mailto:advisories@xxxxxxxxxxxxxx)>
  - *Date:* Fri, 27 Apr 2007 13:36:02 -0400
- 

-----BEGIN PGP SIGNED MESSAGE-----

Hash: SHA1

Virtual Security Research, LLC.  
<http://www.vsecurity.com/>  
Security Advisory

-----  
Advisory Name: Multiple Format String Injections in AFFLIB  
Release Date: 2007-04-27  
Application: AFFLIB(TM)  
Versions: 2.2.0-2.2.5 and likely earlier.  
2.2.6-2.2.8 contain a subset of these vulnerabilities.  
Severity: Medium to Low  
Author: Timothy D. Morgan <[tmorgan@vsecurity.com](mailto:tmorgan@vsecurity.com)>  
Vendor Status: Vendor Notified, Limited Fixes Available  
CVE Candidate: CVE-2007-2054  
Reference:  
<http://www.vsecurity.com/bulletins/advisories/2007/afflib-fmtstr.txt>  
-----

Product Description:

From the [forensicswiki.org](http://forensicswiki.org) website[1]:

"The Advanced Forensics Format (AFF) is an extensible open format for the storage of disk images and related forensic metadata. It was developed by Simson Garfinkel and Basis Technology."

AFFLIB(TM) is the reference implementation of the AFF(TM) format, written primarily by Simson Garfinkel. It comes in the form of an open source library and a set of command line tools used to manipulate

AFF(TM) files.

#### Vulnerability Overview:

In mid-March, 2007 Virtual Security Research, LLC (VSR) performed a security code review of AFFLIB(TM) as a part of an internal tool assessment process. As a result, multiple vulnerabilities of varying severities were discovered. The most significant of these vulnerabilities are being announced publicly to raise awareness and help end-users secure themselves against potential attack.

Several command line utilities included in AFFLIB(TM) pass command line arguments to warn() and err() calls as part of the format string argument. If an attacker could influence these command line parameters, these could be exploited to execute arbitrary code.

Some of the listed vulnerabilities have been fixed in versions 2.2.6 and later, but others remain in the latest release (2.2.8). All line numbers listed below are from version 2.2.0.

#### Vulnerability Details:

The following sections include detailed descriptions of the format string injection vulnerabilities found during the assessment.

##### \* Format String Injection in s3 \*

File: lib/s3.cpp

Line: 207

##### Description:

A command line parameter is used as the format string in the err() call. If an attacker could control this name, a format string injection vulnerability could be exploited. Lines 192–207 are included to illustrate the problem:

```
void s3_cp(const char *fname,string key)
{
struct s3headers meta[2] = {{0,0},{0,0}};
char buf[64];

if(opt_flag){
snprintf(buf,sizeof(buf),"%d",opt_flag);
meta[0].name = AMAZON_METADATA_PREFIX "arg";
meta[0].value = buf;
}

/* Read from fname into a buffer.
```

## [VulnWatch] AFFLIB(TM): Multiple Format String Injections

\* Note that we do this with read, so that we can read from stdin  
\*/

```
FILE *f = fopen(fname,"r");  
if(!f) err(1,fname);
```

An attacker could exploit this problem if the s3 binary were setuid/setgid, or if the s3 program were executed in a CGI script or something similar.

\* Format String Injections in afconvert \*

File: tools/afconvert.cpp  
Lines: 226, 263, and 305

Description:

A command line parameter is used as the format string in three err() calls. If an attacker could control this name, a format string injection vulnerability could be exploited.

\* Format String Injection in afcopy \*

File: tools/afcopy.cpp  
Lines: 202 and 250

Description:

A command line parameter is used as the format string in two err() calls. If an attacker could control this name, a format string injection vulnerability could be exploited.

\* Format String Injection in ainfo \*

File: tools/ainfo.cpp  
Line: 584

Description:

A command line parameter is used as the format string in the err() call. If an attacker could control this name, a format string injection vulnerability could be exploited.

\* Format String Injection in aimage \*

File: aimage/aimage.cpp  
Line: 577

Description:

A command line parameter is used as the format string in the err() call. If an attacker could control this name, a format string injection

## [VulnWatch] AFFLIB(TM): Multiple Format String Injections

vulnerability could be exploited. Lines 548–577 are included below to help illustrate the problem:

```
int getlock(class imager *im)
{
/* If the file exists and the PID in the file is running,
* can't get the lock.
*/
char lockfile[MAXPATHLEN];
sprintf(lockfile,"/tmp/aimge.%s.lock",im->infile);
if(access(lockfile,F_OK)==0){
/* Lockfile exists. Get it's pid */
char buf[1024];
FILE *f = fopen(lockfile,"r");
if(!f){
perror(lockfile); // can't read lockfile...
return -1;
}
fgets(buf,sizeof(buf),f);
buf[sizeof(buf)-1] = 0;
int pid = atoi(buf);
if(checkpid(pid)==0){
/* PID is not running; we can delete the lockfile */
if(unlink(lockfile)){
err(1,"could not delete lockfile %s: ",lockfile);
}
}
/* PID is running; generate error */
errx(1,"%s is locked by process %d\n",im->infile,pid);
}
FILE *f = fopen(lockfile,"w");
if(!f){
err(1,lockfile);
}
```

Since the `im->infile` value could be specified by a user, the lockfile string could contain format string characters. An attacker could exploit this problem if the `aimage` binary were `setuid/setgid`, or if the `aimage` program were executed in a CGI script or something similar.

```
* Format String Injection in imager *
File: aimage/imager.cpp
Line: 265
```

### Description:

A command line parameter is used as the format string in the `err()` call. If an attacker could control this name, a format string injection vulnerability could be exploited.

## [VulnWatch] AFFLIB(TM): Multiple Format String Injections

\* Format String Injection in afxml \*

File: tools/afxml.cpp

Line: 101

### Description:

A command line parameter is used as the format string in the err() call. If an attacker could control this name, a format string injection vulnerability could be exploited.

### Vendor Response:

Simson Garfinkel was first contacted on 2007-03-31. The following timeline outlines the responses from the vendor regarding this issue:

- 2007-04-01 – Vendor provided details of all vulnerabilities identified.
- 2007-04-03 – Continued vendor communication.
- 2007-04-05 – Vendor released version 2.2.6, containing multiple security fixes.
- 2007-04-06 – Vendor notified VSR that fixes were released.
- 2007-04-09 – VSR notified vendor that 9 vulnerability instances still remained in latest release.
- 2007-04-12 – Vendor confirmed that remaining vulnerabilities would be fixed in next release.
- 2007-04-25 – Vendor released versions 2.2.7 and 2.2.8. Vendor did not notify VSR.
- 2007-04-27 – VSR discovered new versions were released. VSR inspected version 2.2.8 and found that no additional vulnerabilities were fixed. VSR advisories published.

### Recommendation:

AFFLIB(TM) users should upgrade to the newest version. Third-party projects which rely on AFFLIB(TM) should encourage users to upgrade, and/or incorporate fixes into their distribution of the library.

The update is available via:

<http://www.afflib.org/downloads/>

---

### Common Vulnerabilities and Exposures (CVE) Information:

The Common Vulnerabilities and Exposures (CVE) project has assigned the following name to these issues. This is a candidate for inclusion

[VulnWatch] AFFLIB(TM): Multiple Format String Injections

in the CVE list (<http://cve.mitre.org>), which standardizes names for security problems.

CVE-2007-2054

-----  
References:

- 1. AFF – Forensics Wiki  
<http://www.forensicswiki.org/wiki/AFF>

-----

This advisory is distributed for educational purposes only, and comes with absolutely NO WARRANTY; not even the implied warranty of merchantability or fitness for a particular purpose. Virtual Security Research, LLC nor the author accepts any liability for any direct, indirect, or consequential loss or damage arising from use of, or reliance on, this information.

-----

Vulnerability Disclosure Policy:

<http://www.vsecurity.com/disclosurepolicy.html>

-----  
AFF(TM) and AFFLIB(TM) are trademarks of Simson Garfinkel and Basis Technology Corp.

Included source code excerpts are copyright Simson Garfinkel and Basis Technology Corp.

This advisory is copyright (C) 2007 Virtual Security Research, LLC. All rights reserved.

-----BEGIN PGP SIGNATURE-----

Version: GnuPG v1.4.6 (GNU/Linux)

iD8DBQFGMjSCQ1RSUNR+T+gRAtFVAJ4+d7NZBSefuHg1IoHtBb6RnPA2aACeJ6p3  
SojDUxCo8X43cOE0XXZcxXo=  
=W+7Y

-----END PGP SIGNATURE-----