

# [UNIX] Xinetd Memory Leaks

**Source:** <http://www.derkeiler.com/Mailing-Lists/Securiteam/2003-11/0069.html>

---

**From:** SecuriTeam ([support\\_at\\_securiteam.com](mailto:support_at_securiteam.com))

**Date:** 11/17/03

To: [list@securiteam.com](mailto:list@securiteam.com)

Date: 17 Nov 2003 17:32:08 +0200

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

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

-----

Xinetd Memory Leaks

---

## SUMMARY

A vulnerability in Xinetd allows remote attackers to cause the program to leak memory, by causing enough memory leakage a computer running Xinetd can be brought to its "knees", i.e. becoming unable to process any additional requests.

## DETAILS

### Background:

Xinetd is a popular inetd replacement. Shortly after the 2.3.9 release in September 2002, it was realized that xinetd was leaking file descriptors. That problem turned out to be that file descriptors were not always being closed whenever a connection was rejected. 2.3.10 was released with this fixup among others in January.

Sometime in February, a machine that Steve administrates was hit by an FTP worm. It created > 5000 connections in 1 second. Xinetd promptly keeled over. Xinetd had been running for over a month with no downtime. The machine has next to no FTP traffic and only from 2 sources, so it was configured to be run via xinetd rejecting connections via tcp\_wrappers. The machine had weathered worm attacks in the past, so this puzzled Steve.

## Securiteam: [UNIX] Xinetd Memory Leaks

### Testing:

Eventually, Steve started looking at xinetd with valgrind. I used the following commandline:

```
valgrind --leak-check=yes --leak-resolution=med --num-callers=8 \  
--logfile=fd=9 /usr/sbin/xinetd -d -pidfile /var/run/xinetd.pid \  
-stayalive 9> out.txt
```

Depending on your setup, you may need to use something higher than 9. Xinetd was tested on connections that succeed and connections that are rejected due to configuration settings. The easiest way to test this is to use the following setup for chargen:

```
service chargen  
{  
    type = INTERNAL  
    user = root  
    protocol = tcp  
    wait = no  
    access_times = 2:00-3:00  
# only_from = 192.168.1.3/24  
# no_access = 192.168.1.3/24  
}
```

The point is to set it up in a way that the connection is guaranteed to be rejected. Then do a:

```
telnet localhost chargen  
After a couple seconds "ctl-] quit"
```

Then, /etc/rc.d/init.d/xinetd stop

Valgrind reports the following:

```
==18939== 144 bytes in 1 blocks are definitely lost in loss record 36 of  
45  
==18939== at 0x40160DB8: malloc (vg_clientfuncs.c:103)  
==18939== by 0x804FE22: (within /usr/sbin/xinetd)  
==18939== by 0x805A496: (within /usr/sbin/xinetd)  
==18939== by 0x8053611: (within /usr/sbin/xinetd)  
==18939== by 0x805340D: (within /usr/sbin/xinetd)  
==18939== by 0x40294A46: __libc_start_main (in /lib/libc-2.3.2.so)  
==18939== by 0x804A310: (within /usr/sbin/xinetd)  
==18939==
```

### Problem:

Using objdump -S /usr/sbin/xinetd, the block of code in question comes from service.c:

```
void svc_request( struct service *sp )  
{  
    connection_s *cp ;  
    status_e ret_code;
```

```

cp = conn_new( sp );
if ( cp == CONN_NULL )
    return ;
if (sp->svc_not_generic)
    ret_code = spec_service_handler(sp, cp);
else
    ret_code = svc_generic_handler(sp, cp);

if ( ret_code != OK )
{
    if ( SVC_LOGS_USERID_ON_FAILURE( sp ) )
        if( spec_service_handler( LOG_SERVICE( ps ), cp ) == FAILED ) {
            conn_free( cp, 1 );
            return;
        }
    CONN_CLOSE(cp);
}
}

```

The above code has several problems. One background piece of information is that the sigchld handler in xinetd (child\_exit->server\_end->svc\_postmortem) normally frees the connection's data. If the ret\_code is not OK, the connection was only closed. This is little more than close(cp->co\_descriptor); This does not free cp since sigchld will not be called. It was only if the log service call failed that the connection was freed.

The above code also did not take into account ret\_code == OK if the service was no\_wait or special. In both of those cases, the sigchld handler is not invoked so the memory pointed to by cp is lost when the call returns.

#### Consequences:

The memory area pointed to by cp is 144 bytes. Since the variable goes out of scope, it is permanently lost with no way of finding it again. The memory losses are cumulative, too. It would take little more than while true; do telnet localhost chargen < /dev/null; done; to DOS the services provided by xinetd if you could identify a machine that uses xinetd to reject connections. Xinetd provides a rich set of options for rejecting connections, this includes: tcp\_wrappers, only\_from, no\_access, sensors, access\_times, cps, load\_avg, etc.

It should also be noted that if you DO NOT have any statements in the xinetd.conf file that would cause xinetd to reject a connection, then you are free from this problem.

#### Solution:

Xinetd 2.3.11 fixes the memory leaks as well as other problems discovered

## Securiteam: [UNIX] Xinetd Memory Leaks

since 2.3.10 was released. All users of xinetd 2.3.10 are strongly urged to upgrade ASAP to avoid DOS conditions. Anyone running 2.3.9 is also strongly urged to upgrade since they are leaking file descriptors.

Your xinetd version can be determined by typing "xinetd -version" (that's version with 1 dash).

The new tarball is: [www.xinetd.org/xinetd-2.3.11.tar.gz](http://www.xinetd.org/xinetd-2.3.11.tar.gz)

This problem has been assigned CAN-2003-0211 to track the bug.

This bug was also reported here:

[https://bugzilla.redhat.com/bugzilla/show\\_bug.cgi?id=88537](https://bugzilla.redhat.com/bugzilla/show_bug.cgi?id=88537)

[https://bugzilla.redhat.com/bugzilla/show\\_bug.cgi?id=88537](https://bugzilla.redhat.com/bugzilla/show_bug.cgi?id=88537)

If you are affected, see if your vendor has an updated xinetd for you.

### ADDITIONAL INFORMATION

The information has been provided by <[mailto:linux\\_4ever@yahoo.com](mailto:linux_4ever@yahoo.com)> Steve Grubb.

=====

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](mailto:list-unsubscribe@securiteam.com)

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