Bug 29536 (CVE-2022-39046) - syslog fail to create large messages (CVE-2022-39046)

Status:	RESOLVED FIXED	**Reported:**	2022-08-29 12:26 UTC by Adhemerval Zanella
Alias:	CVE-2022-39046	**Modified:**	2022-09-08 18:59 UTC (History)
		CC List:	6 users (show)
Product:	glibc		
Component:	libc (show other bugs)	**See Also:**	
		Host:	
Version:	2.36	**Target:**	
		Build:	
Importance:	P2 normal	**Last reconfirmed:**	
Target Milestone:	2.37		
Assignee:	Adhemerval Zanella	**Flags:**	siddhesh: security+
URL:			
Keywords:			
Depends on:			
Blocks:			

Attachments

Add an attachment (proposed patch, testcase, etc.)

Note

You need to log in before you can comment on or make changes to this bug.

Adhemerval Zanella **2022-08-29 12:26:46 UTC** **Description**

```
The fallback to use a heap allocated string for large input arguments do not
correctly create the syslog message.  For example the following test fails:

--
$ cat test.c
#include <stdio.h>
#include <syslog.h>

int main (int argc, const char *argv[])
{
  const char *some_very_long_message = "Lorem ipsum dolor sit amet, consectetur
adipiscing elit. Nulla gravida sapien metus, in sagittis ipsum pellentesque ut. In
dui lectus, elementum ut lacus et, mattis ullamcorper nulla. Cras vel arcu laoreet,
fringilla lacus sit amet, scelerisque nisl. Suspendisse nec massa eu erat commodo
mollis. Curabitur imperdiet velit id lectus laoreet auctor. Sed in enim volutpat,
vulputate ipsum quis, tristique nulla. Vestibulum vitae condimentum metus, nec
commodo lacus. Aliquam erat volutpat. Nunc fringilla justo at feugiat elementum.
Aliquam eget nisl vel arcu molestie placerat ut non lectus. Vivamus scelerisque
condimentum felis ut hendrerit. Pellentesque sit amet dui eu erat lacinia gravida
nec vitae nisl. Suspendisse rhoncus sagittis lacus, pharetra porttitor libero
laoreet eu. Proin scelerisque luctus blandit. Maecenas non odio sapien. Vivamus id
euismod lorem, at maximus nisi. Maecenas consectetur et felis at tempus. Etiam ac
laoreet sem, vitae dignissim nulla. Nulla eu pretium nulla. In nec auctor nisl.
Fusce luctus vel dolor id tempus. Nunc varius nunc eros, eget mattis sapien
```

```
efficitur at. Duis dolor est, vestibulum eu interdum a, interdum id augue. Donec
hendrerit, mi non laoreet placerat, nunc turpis scelerisque dui, eu pulvinar dui
dui facilisis diam. Curabitur sapien risus, varius in neque eget, molestie rutrum
dui. Etiam dolor nulla, sollicitudin nec mauris in, blandit pretium nulla. Orci
varius natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus.
Donec lacinia mollis rutrum. Morbi aliquet tempus odio, ac euismod mi fermentum a.
Duis ut facilisis tortor. Curabitur egestas nisi quis pulvinar porta. Sed
consectetur interdum metus, eleifend condimentum massa congue at. Etiam vel rhoncus
enim. Nullam bibendum velit ut ultricies aliquam. Maecenas in varius elit, nec
sollicitudin lectus. Nulla eleifend scelerisque nulla, eu vehicula tortor vulputate
vitae. In consequat vitae ipsum in sollicitudin. Nam rutrum libero mauris, nec
iaculis lectus lobortis vel. Donec eget tempus nibh. Etiam egestas ultrices tortor,
ac condimentum tellus ultricies in. Nulla commodo hendrerit metus nec feugiat.
Donec libero tortor, posuere sit amet metus malesuada, commodo vulputate ipsum. Nam
a auctor augue. Sed vel libero dui. Donec scelerisque dignissim risus, eget aliquet
arcu vestibulum nec. Aliquam nec arcu vel felis sollicitudin lacinia. Curabitur
eget purus nibh. Phasellus rutrum vulputate nunc, sit amet ullamcorper sem congue
eu. Nam interdum nibh turpis, vehicula sagittis quam dictum vel. Curabitur dolor
sem, pulvinar a velit ac, ultrices tincidunt felis. Quisque vitae mollis ipsum.
Morbi quis tortor a metus iaculis elementum.";

  openlog ("MyTest", LOG_PERROR, LOG_DAEMON);
  syslog (LOG_DEBUG, "%s", some_very_long_message);
  closelog ();
}
$ gcc -Wall test.c -o test
$ ./testrun.sh ./test

$
--

Worse, it access invalid memory:

$ ./testrun.sh --tool=valgrind ./test
[...]
==62032==
==62032== Invalid read of size 1
==62032==    at 0x4936537: __vsyslog_internal (syslog.c:230)
==62032==    by 0x4936955: syslog (syslog.c:90)
==62032==    by 0x48011DF: main (in /home/azanella/Projects/glibc/build/x86_64-
linux-gnu/test)
==62032==  Address 0x4a267bf is 1 bytes before a block of size 29 alloc'd
==62032==    at 0x4811899: malloc (in /usr/libexec/valgrind/vgpreload_memcheck-
amd64-linux.so)
==62032==    by 0x49364AB: __vsyslog_internal (syslog.c:206)
==62032==    by 0x4936955: syslog (syslog.c:90)
==62032==    by 0x48011DF: main (in /home/azanella/Projects/glibc/build/x86_64-
linux-gnu/test)
==62032==
==62032== Conditional jump or move depends on uninitialised value(s)
==62032==    at 0x4817D19: strlen (in /usr/libexec/valgrind/vgpreload_memcheck-
amd64-linux.so)
==62032==    by 0x4885B3F: __vfprintf_internal (vfprintf-process-arg.c:397)
==62032==    by 0x48A8964: __vdprintf_internal (iovdprintf.c:54)
==62032==    by 0x4878FB5: dprintf (dprintf.c:30)
==62032==    by 0x4936561: __vsyslog_internal (syslog.c:230)
==62032==    by 0x4936955: syslog (syslog.c:90)
==62032==    by 0x48011DF: main (in /home/azanella/Projects/glibc/build/x86_64-
linux-gnu/test)
==62032==

==62032==
==62032== HEAP SUMMARY:
==62032==     in use at exit: 0 bytes in 0 blocks
==62032==   total heap usage: 9 allocs, 9 frees, 6,567 bytes allocated
```

```
==62032==
==62032== All heap blocks were freed -- no leaks are possible
==62032==
==62032== Use --track-origins=yes to see where uninitialised values come from
==62032== For lists of detected and suppressed errors, rerun with: -s
==62032== ERROR SUMMARY: 2 errors from 2 contexts (suppressed: 0 from 0)
```

Carlos O'Donell 2022-08-29 13:16:48 UTC **Comment 1**

Posted to the public list:
https://sourceware.org/pipermail/libc-alpha/2022-August/141707.html

Siddhesh Poyarekar 2022-08-29 16:51:37 UTC **Comment 2**

Doesn't seem too serious, but depending on the context of the call, the overread could be triggered remotely.

Siddhesh Poyarekar 2022-08-29 19:36:22 UTC **Comment 3**

OK, some more analysis on what's going on:

There are two invalid reads here:

```
    __dprintf (STDERR_FILENO, "%s%s", buf + msgoff,
               "\n" + (buf[bufsize - 1] == '\n'));
```

one that's a byte under the malloc'd block, through buf[bufsize - 1]. The other is a read of uninitialized memory through buf + msgoff, which will end up revealing contents of buf if it has been reused. At best it will reveal a free list pointer, which we mangle, so that's less of a problem. At worst it could be a block of interest for the attacker.

The byte under the malloc'd block is less interesting because it is merely used to decide whether or not to print the '\n'.

So I'm going to leave the security+ in place and file a CVE request. The fix should get backported to 2.36, where the flaw was introduced.

Adhemerval Zanella 2022-08-30 12:02:51 UTC **Comment 4**

Fixed on 2.37.

Florian Weimer 2022-09-06 13:27:21 UTC **Comment 5**

Commit:

```
commit 52a5be0df411ef3ff45c10c7c308cb92993d15b1
Author: Adhemerval Zanella <adhemerval.zanella@linaro.org>
Date:   Sun Aug 28 16:52:53 2022 -0300

    syslog: Fix large messages (BZ#29536)

    The a583b6add407c17cd change did not handle large messages that
    would require a heap allocation correctly, where the message itself
    is not take in consideration.

    This patch fixes it and extend the tst-syslog to check for large
    messages as well.
```

Checked on x86_64-linux-gnu.

Reviewed-by: Siddhesh Poyarekar <siddhesh@sourceware.org>

sjon 2022-09-06 14:52:40 UTC **Comment 6**

it seems glibc 2.36 is completely unable to syslog messages longer than 1024 chars which is very likely related to this issue.

This seems like a pretty major bug and possible reason to release a glibc update quicker then in 6 months

Siddhesh Poyarekar 2022-09-06 14:58:13 UTC **Comment 7**

Please file a bug with your distribution. The fix has been backported to the release branch. We typically don't do upstream point releases from release branches.

For example, the Fedora backport is being tracked here and will eventually be fixed:

https://bugzilla.redhat.com/show_bug.cgi?id=2123395

Michael Brunnbauer 2022-09-08 17:59:59 UTC **Comment 8**

I don't use a distribution. If it has been backported, where can I find the patch?

Siddhesh Poyarekar 2022-09-08 18:51:30 UTC **Comment 9**

(In reply to Michael Brunnbauer from comment #8)
> I don't use a distribution. If it has been backported, where can I find the
> patch?

The patch is on the master and release/2.36/master branches in glibc git.

Michael Brunnbauer 2022-09-08 18:53:29 UTC **Comment 10**

Is it this one? https://git.launchpad.net/glibc/commit/?id=b0e7888d1fa2dbd2d9e1645ec8c796abf78880b9

Siddhesh Poyarekar 2022-09-08 18:59:00 UTC **Comment 11**

Yes, but to quote a more canonical (hah!) source:

https://sourceware.org/git/?p=glibc.git;a=commit;h=b0e7888d1fa2dbd2d9e1645ec8c796abf78880b9

Note that the fix introduces a regression #29544 which was fixed too:

https://sourceware.org/git/?p=glibc.git;a=commit;h=645d94808aaa90fb1b20a25ff70bb50d9eb1d55b