Fuzzing Project report December 2015

Hanno Böck

December 2015

1 Bignum library testing

The latest OpenSSL release 1.0.2e contains a fix for a carry propagation bug in the BN_mod_exp() function that I reported in August [1]. In some cases the function produced wrong results. The exact security implications are unclear, but it might be possible to attack Diffie Hellman key exchanges with this vulnerability.

This was one result of tests I did with several bignum libraries. Bignum calculations are an important building block of cryptographic software and incorrect results can lead to security issues.

I looked into this because earlier this year a similar bug was found in the BN_sqr() function of OpenSSL [2]. Ralph-Philipp Weinmann was able to show that it would've been possible to find this bug using american fuzzy lop [3].

There are two potential strategies to use fuzz testing on bignum libraries. One can check for inconsistencies inside a library itself, for example by doing calculations that should always produce the same result for different input. Alternatively one can compare the output of different implementations of the same operation. I did several fuzzing tests with OpenSSL, nss, libgcrypt, gmp and nettle.

Appart from the OpenSSL issue I reported a bug in the gcry_mpi_invm() function [4] in libgcrypt, a function to calculate the multiplicative inverse modulo an integer. According to Werner Koch this was a known issue, a fix is planned for the next version of libgcrypt. It is unlikely that this bug has security implications.

I also reported two bugs in the elliptic curve multiplication operations in nettle, one affecting the NIST P-256 curve operations on all architectures [5] and one affecting the NIST P-384 curve operations on 64 bit systems [6]. Nettle-Developer Niels Möller discovered an additional issue while fixing the first bug.

[1] https://openssl.org/news/secadv/20151203.txt https://blog.fuzzing-project.org/31-Fuzzing-Math-miscalculations-in-OpenSSLs-BN_mod_exp-CVE-2015-3193.html

[2] https://www.openssl.org/news/secadv/20150108.txt

[3] https://www.youtube.com/watch?v=PqW8-MUu09c

[4] https://lists.gnupg.org/pipermail/gcrypt-devel/2015-December/003642.html

[5] https://lists.lysator.liu.se/pipermail/nettle-bugs/2015/003028.html

[6] https://lists.lysator.liu.se/pipermail/nettle-bugs/2015/003024.html

I think it's an important takeaway that fuzz testing can be a successful strategy to test mathematical library functions. This may seem unexpected, as fuzzing is usually associated with memory corruption bugs. It also highlights that carry propagation bugs are a common problem in bignum implementations. I expect more similar bugs to be found in the future.

2 Real-world testing with ASAN-based system

In the last report I mentioned already that I am working on a Gentoo system with almost everything built with Address Sanitizer.

In the mean time I was able to create a bootable version of it that I both used to test a virtualized desktop environment and a server. On the server side this uncovered a use after free error in the MySQL Perl bindings (DBD::mysql) [7], a stack overflow in vnstat [8], another out of bounds read in courier [9], a stack overflow in Monit [10], a global out of bounds read in screen [11], an invalid memory read due to a variable type error in syslog-ng [12] and one more out of bounds read in bash [13].

Testing this with some desktop applications uncovered a use after free in Hexchat [14] and three out of bounds bugs in Claws-Mail [15].

It can generally be seen that simple testing of software with Address Sanitizer (even without any fuzzing) can uncover bugs, even in major software packages that are in widespread use on Linux systems.

3 Network fuzzing

Recently Dough Birdwell released a modified version of american fuzzy lop that supports network input [16]. The lack of network support was a major drawback of american fuzzy lop in the past. I tested the new code on various applications [17], surprisingly I haven't found anything with it yet.

This indicates that networking code is generally much better tested than file parsing code.

---

[7] https://github.com/perl5-dbi/DBD-mysql/pull/45

[8] https://github.com/vergoh/vnstat/issues/34

[9] https://www.mail-archive.com/courier-users@lists.sourceforge.net/msg38073.html

[10] https://bitbucket.org/tildeslash/monit/commits/58c562b9de25981060486b38a205dfcfa2235ef4

[11] https://savannah.gnu.org/bugs/index.php?46401

[12] https://github.com/balabit/syslog-ng/issues/780

[13] https://lists.gnu.org/archive/html/bug-bash/2015-11/msg00040.html

[14] https://github.com/hexchat/hexchat/issues/1540

[15] http://www.thewildbeast.co.uk/claws-mail/bugzilla/show_bug.cgi?id=3559 http://www.thewildbeast.co.uk/claws-mail/bugzilla/show_bug.cgi?id=3563 http://www.thewildbeast.co.uk/claws-mail/bugzilla/show_bug.cgi?id=3573

[16] https://github.com/jdbirdwell/afl

[17] https://blog.fuzzing-project.org/27-Network-fuzzing-with-american-fuzzy-lop.html

4 Public talks

I held a talk on the Fuzzing Project at the BSides Vienna conference [18]. I plan to do a session on Fuzzing during the 32C3 in cooperation with the Free Software Foundation Europe [19]. I also have submitted a talk on the Address Sanitizer work (see above) for FOSDEM.

5 Notable bugs (dpkg, libxml2, libpcre, mutt)

As usual, I discovered and reported a number of bugs via fuzzing. Notable packages that I discovered bugs via fuzzing were DPKG [20], Libxml2 [21], PCRE [22] and Mutt [23].

One bug I discovered that was interesting, but that wasn't related to Fuzzing, was in old versions of GnuTLS that are still in use by Ubuntu LTS and Debian old stable. They failed to check the first byte of the CBC padding, some tests for the POODLE TLS vulnerability would mark such hosts als vulnerable [24].

Further fuzzing- and asan-related bugs were reported in GNU Indent [25], sed [26], jbig2dec [27], troff [28], fltk [29], nasm [30], flex [31], mawk [32], gawk [33], grep [34] (bug was in gnulib), ZStandard [35].

[18] http://bsidesvienna.at/talks/#18

[19] https://events.ccc.de/congress/2015/wiki/Session:The_Fuzzing_Project

[20] https://blog.fuzzing-project.org/30-Stack-overflows-and-out-of-bounds-read-in-dpkg-Debian.html

[21] https://blog.fuzzing-project.org/28-Libxml2-Several-out-of-bounds-reads.html

[22] https://blog.fuzzing-project.org/29-Heap-Overflow-in-PCRE.html

[23] http://dev.mutt.org/trac/ticket/3776 http://dev.mutt.org/trac/ticket/3787

[24] https://blog.hboeck.de/archives/877-A-little-POODLE-left-in-GnuTLS-old-versions.html

[25] https://savannah.gnu.org/bugs/index.php?46626 https://savannah.gnu.org/bugs/index.php?46627

[26] http://debbugs.gnu.org/cgi/bugreport.cgi?bug=22127

[27] http://bugs.ghostscript.com/show_bug.cgi?id=696452 http://bugs.ghostscript.com/show_bug.cgi?id=696453

[28] https://savannah.gnu.org/bugs/index.php?46401

[29] http://www.fltk.org/str.php?L3263+P0+S-2+C0+I0+E0+V1.

[30] http://bugzilla.nasm.us/show_bug.cgi?id=3392328 http://bugzilla.nasm.us/show_bug.cgi?id=3392329 http://bugzilla.nasm.us/show_bug.cgi?id=3392330

[31] https://sourceforge.net/p/flex/bugs/193/ https://sourceforge.net/p/flex/bugs/194/

[32] https://github.com/ThomasDickey/original-mawk/issues/36

[33] http://git.savannah.gnu.org/cgit/gawk.git/commit/?id=611dc46d8b216c22f05604e8df6bee7aa59e5977

[34] https://debbugs.gnu.org/cgi/bugreport.cgi?bug=21513

[35] https://blog.fuzzing-project.org/26-Two-out-of-bounds-reads-in-Zstandard-zstd.html

6 Misc

There is a new kernel fuzzing tool called syzcaller [36]. I haven't tried it yet, but it looks very promising and much more advanced than previous similar tools. It currently depends on the development code of gcc and requires a patched kernel.

I did some testing with libfuzzer [37], which is a fuzzing tool with similar methods as american fuzzy lop, but it operates on a function level. It is part of LLVM.

[36] `https://github.com/google/syzkaller`
[37] `http://llvm.org/docs/LibFuzzer.html`