Technical Disclosure Commons

Defensive Publications Series

January 21, 2019

Using build identifiers to fingerprint ELF binaries and link to build information without having access to source code

Armijn Hemel

Using build identifiers to fingerprint ELF binaries and link to build information without having access to source code

Abstract

Finding out where a software program or library comes from and how it was built without having direct access to the source code is not a trivial problem to solve. While versions of programs can be fairly accurately guessed[1][2] this is a lot more difficult for build configuration.

By comparing build identifiers from binaries of which nothing is known with build identifiers extracted from binaries for which source code and build information is available it is in certain cases possible to find out what source code and build information was used for a binary.

Keywords

ELF, Linux, Unix, executable, fingerprinting, provenance, software composition

Method

Modern compiler suites on Linux and other Unix(-like) operating systems (FreeBSD, NetBSD, OpenBSD and others) such as GCC + GNU binutils and LLVM allow recording a so called build-id[3] in the binary in an ELF note section called ".note.gnu.build-id" with type "SHT_NOTE" [4]. This information can be extracted from ELF binaries, such as readelf[5] as well as other tools.

The idea behind the build-id is that it will be the same every time the same configuration is used to build the binary, even though the binary itself might be slightly different because of for example different time stamps, which makes comparisons with cryptographic hashes (MD5, SHA1, SHA256, etcetera) less useful.

The major Linux distributions like Fedora Linux en Debian GNU/Linux have large archives[6][7] with binaries plus corresponding source code and build information. To create a mapping between the build-ids and the source code the following steps are taken:

1. packages with binary files are downloaded from a Linux distribution, together with corresponding source code and build information
2. the packages with binaries downloaded in step 1 are analyzed to see if there are ELF files inside and if these ELF files have a build-id embedded in the binary
3. for all files analyzed in step 2 that have a build-id, the build-id is extracted from the binary
4. the build-id is stored together with information about the source code archive and the build information

The mapping between build-id and information about the source code archive and build information could multiple forms, such as a relational database table:

build-id	source code archive	build information
abcd-1234-efgh	test-0.1.src.rpm	test-0.1.spec
hgfe-4321-dcba	test-0.2.orig.tar.gz	test-0.2.debian.tar.gz

but other storage mechanisms could also work, such as a directory with symbolic links with the name of the uuid to a directory with the source code and build information, or a spreadsheet file.

When an unknown ELF binary is scanned the following steps are taken:

1. the binary is examined to see if the ".note.gnu.build-id" section is present in the binary
2. if the section ".note.gnu.build-id" is present the build-id is extracted using readelf or a similar tool or mechanism
3. the extracted build-id from step 2 is compared to the known build-ids in the database
4. if there is a match in step 3 the corresponding source code and build information is reported.

References

[1] Finding Software License Violations Through Binary Code Clone Detection, IP.com Disclosure Number: IPCOM000214472D, https://priorart.ip.com/IPCOM/000214472

[2] Hemel, Armijn & Trygve Kalleberg, Karl & Vermaas, Rob & Dolstra, Eelco. (2011). Finding software license violations through binary code clone detection. Proceedings - International Conference on Software Engineering. 63-72. 10.1145/1985441.1985453.

[3] http://web.archive.org/web/20101030115235/https://fedoraproject.org/wiki/RolandMcGrath/BuildID

[4] Linker and Libraries guide, section "Note Section", http://web.archive.org/web/20150223114310/https://docs.oracle.com/cd/E19683-01/816-1386/6m7qcoblj/index.html#chapter6-18048

[5] GNU binutils manual, https://sourceware.org/binutils/docs-2.31/binutils/readelf.html

[6] Fedora Linux archive, https://archive.fedoraproject.org/pub/

[7] Debian archive, http://archive.debian.org/