

[REVS] Comparing Binaries with Graph Isomorphism

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2004-04/0017.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 04/07/04

To: list@securiteam.com

Date: 7 Apr 2004 11:12:05 +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>

Comparing Binaries with Graph Isomorphism

SUMMARY

The paper (linked at the end) presents a method and algorithms for finding differences between two versions of a binary executable file, based on graph isomorphism. One possible application is to discover the differences in a security patch, and a couple examples in that vein are shown. A brief comparison is also made to Halvar Flake's function signatures approach.

DETAILS

Introduction:

The problem of comparing two different versions of a binary has several possible applications. Some applications could be

- * The nature of some build processes may rebuild some files that did not "really" need to be rebuilt [1]. But after the fact, figuring out which files those are (so that they can not be shipped in an update), is non-trivial. The referenced paper has a heuristic solution to that. Binary comparisons would be another possibility.

- * During incident response, you find a modified version of some normal file on the system, and want to know what modifications have been made to

it.

* Suppose a third party product uses undocumented interfaces in an OS. A new version or patch comes out that breaks the third party app. The developer wants to both: a) fix their application to work with the new version, and b) figure out the reason behind the change, and if it was done just to break their application.

* Probably, the most provocative application is to reverse engineer security patches to determine the source of the vulnerability. There are several reasons you might want to do this:
A vendor releases a patch but claims the impact is a denial of service. The researcher claims it is exploitable to run code.

o However, neither party offers details. As a customer, your policy is that you will patch sooner if it is exploitable than if it is just a denial of service. How do you figure out what to do? (You may not be capable of doing the analysis, but could use the results of others.)

o You want to find a method of checking whether the patch has been installed, without inspecting the files on the system, and without fully exploiting the vulnerability.

o Lastly, you may want to find the vulnerability so that you can write an exploit for it. Blackhats, of course, are interested in this, but it is also necessary for penetration testing, for example.

Recently, Halvar Flake has presented some work in this area at various conferences, including BlackHat Windows 2004 [2]. This paper will present an alternative method of attacking the problem.

ADDITIONAL INFORMATION

The information has been provided by <mailto:tsabin@razor.bindview.com>
Todd Sabin.

The complete article can be found at:
<<http://razor.bindview.com/publish/papers/comparing-binaries.html>>
<http://razor.bindview.com/publish/papers/comparing-binaries.html>.

=====

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

=====
=====

Securiteam: [REVS] Comparing Binaries with Graph Isomorphism

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.