Available online at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/cose

Computers & Security

A practical off-line taint analysis framework and its application in reverse engineering of file format

Baojiang Cui [a,b,*], Fuwei Wang [a,b], Tao Guo [c], Guowei Dong [c]

[a] Beijing University of Posts and Telecommunications, Beijing, China
[b] National Engineering Laboratory for Mobile Network Security, Beijing, China
[c] China Information Technology Security Evaluation Center, Beijing, China

ARTICLE INFO

Article history:
Received 14 August 2014
Received in revised form
12 February 2015
Accepted 15 February 2015
Available online 5 March 2015

Keywords:
Taint analysis
Data flow tracking
Binary instrumentation
Format reverse engineering
Fuzzing test
Virtualized execution
Parallelization

ABSTRACT

This paper presents FlowWalker, a novel dynamic taint analysis framework that aims to extract the complete taint data flow while eliminating the bottlenecks that occur in existing tools, with applications to file-format reverse engineering. The framework proposes a multi-taint-tag assembly-level taint propagation strategy. FlowWalker separates taint tracking operations from execution with an off-line structure, utilizes memory-mapped files to enhance I/O efficiency, processes taint paths during virtual execution playback, and uses parallelization and pipelining mechanisms to achieve speedup. Based on the semantic correlations implied by the taint path information, this paper presents an algorithm for extracting the structures of unknown file formats. According to test data, the overall program runtime ranges from 92.98% to 208.01% of the length of the underlying instrumentation alone, while the speed enhancement is 60% compared to another well-featured tool in Windows. Medium-complexity file formats are correctly partitioned, and the constant fields are extracted. Due to its efficiency and scalability, FlowWalker can address the needs of further security-related research.

1. Introduction

Over the past decade, dynamic taint analysis (DTA) has become a popular technique in the field of software security analysis. Fundamentally, DTA entails tagging specific user input sources as original taint data and monitoring their propagation during the entire process runtime. Thus, a taint data flow path is extracted, which can be used for further analyses on program semantics and smart fuzzing, among other applications. Data flow tracking is also necessary to secure local servers and clients against privacy leaks, which is critical to cybercrime prevention and digital forensics.

In recent years, DTA theory has been studied in-depth and implemented by many researchers in numerous tools. The basic taint propagation strategy was first introduced by J. Newsome and D. Song in their tool TaintCheck (Newsome and Song, 2005), which aims to perform automatic detection and analysis of exploits in commodity software. DTA algebra was later discussed systematically and theoretically in (Schwartz et al., 2010). Since then, DTA applications have increased in number. Many DTA techniques have been implemented and

[*] *Corresponding author*. No.10 Xitucheng Road, Haidian District, Beijing, China. Tel.: +86 13611330827.
E-mail address: cuibj@bupt.edu.cn (B. Cui).

http://dx.doi.org/10.1016/j.cose.2015.02.006

widely utilized in various research areas related to binary analyses and vulnerability exploitation, such as Temu (Yin and Song, 2010), Panorama (Yin et al., 2007), Minemu (Bosman et al., 2011), libdft (Kemerlis et al., 2012) and TaintScope (Wang et al., 2010). A number of high-level applications have been designed on the basis of DTA tools. Examples of these applications include taint-aided data format reverse engineering and data relevance assessment using taint analysis. REWARDS (Lin et al., 2008) is an outstanding implementation of this kind, which has achieved automatic network protocol format reverse engineering through context-aware monitored execution.

However, the practicability of DTA prototypes is subject to some limitations. The most challenging problem is the excessive overhead associated with these tools and platforms. DTA can consume an excessive amount of extra storage and CPU resources, apart from the inherent overhead of binary instrumentation, which makes these tools incapable of executing even normal-scale programs. In particular, I/O bottlenecks in the recording of the huge amount of information in a typical database or set of disk files for analysis purposes severely hinder the execution speed. To realize the true power of DTA from its redundant form, some researchers have attempted to improve the implementation of various techniques. At both the NDSS (Jee et al., 2012a) and CCS (Jee et al., 2013) security conferences over the past two years, there were published papers arguing for possible enhancements from either theoretical or technical perspectives.

In this paper, we present *FlowWalker*, a novel taint analysis framework. The DTA function is performed off-line by separating the taint tracking logic from the execution process. Two stand-alone modules control recording and analysis: the dynamic module works on a binary instrumentation platform to instrument and record the trace of the target process, and a static analysis module or trace-replaying virtual machine replays the process and tracks the taint propagation with each executed instruction. Additionally, a file-format reverse engineering extension is designed and implemented by analyzing the implicit taint data correlations.

The original aspects and contributions of this framework are threefold:

- *Enhanced execution performance*. The overhead attached to running processes is maintained at an applicable level. The off-line analysis architecture removes all workload associated with maintaining and tracking taint status from real processes. A virtual machine replaying recorded traces can carry out complicated multi-tag taint tracking and parallelize the entire workload. Moreover, with the improvement of techniques such as those related to memory-mapped files, several bottlenecks are eliminated.
- *Comprehensive and adoptable taint propagation logic*. Multi-tag taint attributes and strategies are applied. Several sequences of specific instructions that can produce particular semantic effects are identified and monitored. Most importantly, support for MMX, SSE and SSE2 supplementary instruction sets is added for the taint analysis logic.
- *Innovative application to file format cognition*. Currently, format reverse engineering with the aid of taint analysis mainly targets network protocols that are relatively uncomplicated compared to the more complex formats typically encountered in file-format reverse engineering. FlowWalker extracts more semantic information from taint analysis results and makes a significant attempt to deduce file formats from taint information, yielding a promising result.

Moreover, with the ultimately different architecture and techniques, FlowWalker is a brand-new project, not just improvements or modifications based on some former codebases. In order to let our practical framework be verified and adopted in the projects in demands of an efficient DTA base, we have published our project on GitHub under modified BSD license. We would direct anyone who is interested in testing or adopting FlowWalker to visit our project page.[1]

This paper is organized as follows: Section 2 provides a summary of taint analysis and an overview of the architecture of FlowWalker. Section 3 introduces the design and implementation of the off-line taint analysis function of FlowWalker, including the detailed taint propagation logic. As a demonstration of practicability, Section 4 presents an extensive description of the application of taint analysis results to grey-box file-format reverse engineering. Finally, in Section 5, the methods used to evaluate FlowWalker and the results of that evaluation are presented.

2. Background and overview

In this section, we present the principles of the DTA technique, its general uses in the scope of security analysis, and the limitations of existing implementations. Then, we present the architecture of FlowWalker.

2.1. *Basis of DTA*

The behavior of a process is profiled using two main categories of analysis: control flow and data flow. Control flow (Allen, 1970) integrates all runtime execution information, such as sequences of invocations among binary modules, classes, functions and basic blocks. In the scope of binary analysis and vulnerability mining, the basic principle is to determine the pivotal conditional jump instructions that compose a branching diagram of the target process; with knowledge of each possible branch, which is actually an elemental node within the control flow, it is possible for finite combinations of process states to be enumerated. Therefore, considerable research effort has been applied to control flow analysis.

Notably, by means of control flow analysis, we can enumerate the states of the process, but we cannot traverse them. Only a specific subset of the pivotal conditional branching is controllable by external inputs to the program; among these inputs, only the input data from the user interface can be carefully constructed to influence the specific branching. The relationship between the external data and the process behaviors is the core component of data flow analysis.

[1] https://github.com/forward-wfw/FlowWalker.