Available online at www.sciencedirect.com

ScienceDirect

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

Computers & Security

Automated analysis of freeware installers promoted by download portals

Alberto Geniola[a], Markku Antikainen[b,*], Tuomas Aura[a]

[a] Aalto University, Finland
[b] Helsinki Institute for Information Technology, University of Helsinki, Finland

ARTICLE INFO

Article history:
Received 24 October 2017
Accepted 26 March 2018
Available online 31 March 2018

Keywords:
Potentially-unwanted program
Pay-per-install
UI-automation
Man-in-the-middle Malware

ABSTRACT

We present an analysis system for studying Windows application installers. The analysis system is fully automated from installer download to execution and data collection. The system emulates the behavior of a lazy user who wants to finish the installation dialogs with the default options and with as few clicks as possible. The UI automation makes use of image recognition techniques and heuristics. During the installation, the system collects data about the system modification and network access. The analysis system is scalable and can run on bare-metal hosts as well as in a data center. We use the system to analyze 792 freeware application installers obtained from popular download portals. In particular, we measure how many of them drop potentially unwanted programs (PUP) such as browser plugins or make other unwanted system modifications. We discover that most installers that download executable files over the network are vulnerable to man-in-the-middle attacks. We also find, that while popular download portals are not used for blatant malware distribution, nearly 10% of the analyzed installers come with a third-party browser or a browser extension.

1. Introduction

Most computer users download and install some freeware applications from the Internet. The source is often one of the many download portals, which aggregate software packages and also offer locations for hosting them. It is common concern that the downloaded software might be infected with malware or have other unwanted side effects. Freeware installers are also known for dropping of potentially unwanted programs (PUP) to the user's computer. PUP and other unwanted system modifications to desktop computers can also be considered a security threat (Emm et al., 2016; Wood et al., 2016). This phenomenon is partly caused by the *pay-per install* (PPI) business model where freeware software developers can monetize their software effectively by bundling it with other third-party applications or by promoting some software and services by changing the user's default settings. This business model is not always illegal as the application installer may inform the user about the third-party software and even allow her to opt-out from installing third-party applications. However, this is often done in a way that the user is not completely aware of the choices he makes.

In this paper, we set out to analyze how prevalent are the security and PUP problems among the software obtained from

[*] Corresponding author.
E-mail addresses: alberto.geniola@studenti.polito.it (A. Geniola), markku.antikainen@helsinki.fi (M. Antikainen), tuomas.aura@aalto.fi (T. Aura).

https://doi.org/10.1016/j.cose.2018.03.010

download portals. For this, we create an automated analysis system that downloads and installs the applications in a sandbox while monitoring the target system. The sandbox emulates the behavior of a lazy user who tries to complete the installation process with the default settings of the installer. It does this with the help of image recognition on screenshots and heuristic rules. During the whole process, we record network traffic and modifications to the target system. We demonstrate the capabilities of the system by analyzing nearly 800 popular software installers from eight different download portals.

As hinted, we have two distinct goals. First, we create a scalable and fully automated tool for analyzing a large number of application installers. Unlike other existing application analysis sandboxes (e.g. Cuckoo Sandbox by Guarnieri et al., 2012), our tool is not only a plain sandbox but can also interact with application installers. Our second goal is to use the system to analyze large quantities of software from different download portals in order to better understand the prevalence of any security problems in them. Unlike earlier research on PPI and PUP, such as those presented by Caballero et al. (2011) and Thomas et al. (2016), we do not try to differentiate between legitimate and malicious actions but try to cover all potentially unwanted changes to the system. This not only gives insight to the prevalence of any problems but also teaches up about the software installers and download portals in general.

More specifically, our contributions are the following:

- We create a scalable, fully automated, sandboxed analysis system for software installers. The system uses UI automation to emulate user interaction and monitors the installation process. The system supports virtualized as well as bare-metal sandboxes. The system has been published as open source.[1]
- To show the capabilities of the system, we use it to analyze 792 popular freeware installers crawled from eight popular download portals. The analysis covers file system access, registry modifications, and network traffic. We look for indications of unwanted software drops, other potentially unwanted changes to the system, and vulnerabilities in the network communication of the installers. Our main findings include that while the download portals do not distribute malware, 1.3 % of the installers led to the installation of a well-known potentially unwanted application (PUP) and nearly 10 % of the installers came with a third-party browser (e.g. Chrome) or a browser extension. Furthermore, we found that the installers often download the application binaries over HTTP and that over half of these are do not verify the integrity of the binary and are thus vulnerable to man-in-the-middle (MitM) attacks. While some of the analysis results have been published earlier (Geniola et al., 2017), the results and discussion presented in this paper are more comprehensive than what has been published earlier.

The rest of this paper is organized as follows. Section 2 reviews related work. Section 3 describes the overall architecture of the analysis system. Section 4 explains how we were able to automatically interact with the UI's of the installers. Section 5 moves towards using the analysis system and describes how the system was used to analyze a large number of freeware installers. Analysis results are presented in Section 6 and further discussed in Section 7. Section 8 concludes the paper.

2. Background

This section describes the related work and ideas on which our research is based.

2.1. *Potentially unwanted programs*

Downloading applications from the Internet can be dangerous, and this also applies to download portals (Heddings, 2014; 2015). The applications might come with unwanted features that range from clearly malicious, such as bundled malware and spyware, to minor nuisances like changing the browser's default search engine. Such software is often referred to as *potentially unwanted programs* (PUP)[2]. We use the broad definition of Goretsky (2011), which states that a PUP is an application or a part of an application that installs additional unwanted software, changes the behavior of the device, or perform other kinds of activities that the user has not approved or does not expect. PUP often functions in a legal and moral gray area. The threat of legal action from PUP authors has been suggested as the reason why antimalware labels it as "potentially unwanted" rather than "malicious" (Boldt and Carlsson, 2006; McFedries, 2005).

Recently, Kotzias et al. (2016) have shown that freeware installers only rarely come bundled with critical malware. More often, the system modifications are just unnecessary and unexpected. The user may even be informed about them, for example, in the end-user licence agreement (EULA), and the installer may allow a careful user to opt out of unwanted features. However, as pointed out by Böhme and Köpsell (2010); Motiee et al. (2010), users do not always read EULAs and may be habituated to accept default settings and ok any warnings. This *rushing-user* behavior leads the user to giving *uninformed consent* to the system modifications. Moreover, PUP installers often come with a complex EULAs (Good et al., 2005), which users are more likely to accept blindly (Bruce, 2005). Solutions to this problem have been proposed (Boldt and Carlsson, 2006). For example, Boldt et al. (2008) showed that it is possible to detect some classes of spyware can be detected by analyzing the EULAs. However, none of the proposals has been widely adopted.

On mobile platforms, the problem of uninformed consent has been solved so that the operating system informs the user about the permissions given to each application. This may happen either at the install time (e.g. Android 5 and earlier), or when the application requests access to restricted

[1] The analysis system is available at https://github.com/albertogeniola/TheThing/tree/master.

[2] Potentially Unwanted Application (PUA) is another often used term.