Detecting the Onset of Infection for Secure Hosts

Kui Xu[1], Qiang Ma[2], and Danfeng (Daphne) Yao[1]

[1] Department of Computer Science, Virginia Tech
xmenxk,danfeng@cs.vt.edu
[2] Department of Computer Science, Rutgers University
qma@cs.rutgers.edu

Abstract. Software flaws in applications such as a browser may be exploited by attackers to launch drive-by-download (DBD), which has become the major vector of malware infection. We describe a host-based detection approach against DBDs by correlating the behaviors of human-user related to file systems. Our approach involves capturing keyboard and mouse inputs of a user, and correlating these input events to file-downloading events. We describe a real-time monitoring system called *DeWare* that is capable of accurately detecting the onset of malware infection by identifying the illegal download-and-execute patterns.

Analysis based on the arrival methods of top 100 malware infecting the most number of systems discovered that 53% of infections are through download [1]. In another study, 450,000 out of 4.5 millions URLs were found to contain drive-by-download exploits that may be due to advertisement, third-party contents, and user-contributed contents [2]. Drive-by-download (DBD) attacks exploit the vulnerabilities in a browser or its external components to stealthily fetch malicious executables from remote malware-hosting server without proper permission of the user.

We present *DeWare* – a host-based security tool for detecting the onset of malware infection at real time, especially drive-by-download attacks. Deware is application independent, thus it is capable of performing host-wide monitoring beyond the browser. DeWare's detection is based on observing stealthy download-and-execute pattern, which is a behavior virtually all active malware exibits at its onset.

However, the main technical challenge to successful DBD detection is to tell DBDs apart from legal downloads. Our solution is based on monitoring relevant file-system events and correlating them with user inputs at the kernel level. In contrast to DBDs, legitimate user download activities are triggered by explicit user requests. Also, browser itself may automatically fetch and create temporary files which are not directly associated with user actions. To that end, we grant browser access to limited folders with additional restrictions.

Security and attack models We assume that the browser and its components are not secure and may have software vulnerabilities. The operating system is assumed to be trusted and secure, and thus the kernel-level monitoring of file-system events and user inputs yields trusted information. The integrity

of file systems defined in our model refers to the enforcement of user-intended or user-authorized file-system activities; the detection and prevention of malware-initiated tampering.

DeWare Archietecture Overview The DeWare monitoring system is designed to utilize a combination of three techniques, including input logger, system monitor, and execution monitor. Following are the main components.

- *Input logger* that intercepts user inputs at the kernel level with timestamp and process information (i.e., to which process the inputs go to). User inputs are viewed as *trusted seeds* in our analysis, which are used to identify legitimate system behaviors.
- *System logger* which intercepts system calls for file creations, and probes kernel data structures to gather process information. Timestamps can be obtained from input logger at runtime to perform temporal correlation.
- *Access control framework* that specifies (1)accessible area: where an application is allowed to make file creations, (2)downloadable area: places a user can download files into via an application.
- *Execution monitor* which gives additional inspection to areas where access is granted to an application or user downloads.

Capturing all file-creation events related to processes generates an overwhelmingly large number of false alarms. The purpose of our access control framework is to reduce the white noise, by granting a process access to certain folders, which are defined as accessible area. For example, Temporary Internet Files folder is modifiable by IE – in contrast, system folder is not. *Execution monitor* is to prevent malware executables from being run at accessible area.

Prototype Implementation in Windows Our implementation and experiments are built with Minispy, a kernel driver for Windows operating systems. It is able to monitor all events where system is requesting to open a handle to a file object or device object, and further find out the file creations. Logged activities are reported to user mode where the access control policy, input correlation, file extension check are performed. We record user inputs at the kernel level through hooks SetWindowsHookex provided by Windows OS. The *execution monitor* is realized with Microsoft PsTools and the process tracking in local security settings. We have carried out a study with 22 users to collect real-world user download behavior data. We will also use DeWare to evaluate a large number of both legitimate and malware-hosting websites for testing its detection accuracy.

References

1. Macky Cruz. Most Abused Infection Vector. http://blog.trendmicro.com/most-abused-infection-vector/
2. N. Provos, D. McNamee, P. Mavrommatis, K. Wang, and N. Modadugu. The ghost in the browser analysis of web-based malware. In Hot-Bots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets, Berkeley, CA, USA, 2007. USENIX Association.