

Profiling Memory Usage Patterns for Keylogging Detection

Stefano Ortolani¹, Cristiano Giuffrida¹, and Bruno Crispo²

¹ Vrije Universiteit, De Boelelaan 1081, 1081HV Amsterdam, The Netherlands
{ortolani, giuffrida}@cs.vu.nl

² University of Trento, Via Sommarive 14, 38050 Povo, Trento, Italy
crispo@disi.unitn.it

Privacy-breaching malware is an ever-growing class of malicious applications that attempt to steal confidential data and leak them to third parties. A common activity performed by privacy-breaching malware is keylogging, that is the eavesdropping, harvesting, and leakage of user-issued keystrokes. One of the main reasons for this rapid growth is the possibility for unprivileged programs running in user space to eavesdrop and record all the keystrokes of the users of the system. Such an ability to run in unprivileged mode facilitates their implementation and distribution, but, at the same time, allows to understand and model their behavior in detail. To address the general problem of malware detection, a number of models and techniques have been proposed over the years. However, when applied to the specific problem of detecting malware with keylogging behavior, all existing solutions are unsatisfactory.

In [1] we proposed a novel detection technique based on the observation that the general behaviour of a keylogger can be easily modeled and triggered upon request in a well-defined manner. Leveraging this property, we designed a new detection technique that simulates carefully crafted keystroke sequences in input, and monitors the I/O behavior of the keylogger in output to univocally identify it among all the running processes. While we have shown the effectiveness of our technique in detecting most of the existing keyloggers, we however experienced some limitations in detecting malware with keylogging behavior. This is due to the many evasion techniques malware typically adopts: aggressive buffering and disguise of the keylogging activity, to name a few.

To address and overcome these limitations we extend our initial solution by refining the original behavioral model. Rather than measuring the I/O operations a process performs, our new design aims to profile the memory patterns a process exhibits and monitor any correlation with the input. A fine-grained monitoring infrastructure is necessary to carefully identify and model the keylogging behavior, and setting it apart from legitimate applications that perform similar activities (e.g. shortcut managers). Preliminary experimental results are promising and confirm the viability of our approach.

References

1. Ortolani, S., Giuffrida, C., Crispo, B.: Bait your hook: a novel detection technique for keyloggers. Proceedings of the 13th International Symposium on Recent Advances in Intrusion Detection (RAID 2010) pp. 198–217 (2010)