Transparent and Extensible Malware Analysis by Combining Hardware Virtualization and Software Emulation

Lok Kwong Yan [†‡] Manjukumar Jayachandra[†] Mu Zhang[†] Heng Yin[†]
[†]Syracuse University [‡]Air Force Research Laboratory
{*loyan, mjayacha, muzhang, heyin*}*@syr.edu*

Malware is actively making efforts to evade analysis. In particular, anti-emulation techniques have been deployed to defeat fine-grained dynamic analysis. Our evaluation of 150 real world malware samples revealed that 14 could not be analyze by any of three popular emulation based analysis tools, Anubis [1], CWSandbox [3] and TEMU [5]. While these samples operated normally in KVM using hardware virtualization, they either crashed or exhibited no malicious behaviors on these three analysis platforms. Apparently, emulation-based malware analysis is extensible to provide good instrumentation support, but its downside is lack of transparency.

In contrast, leveraging hardware virtualization, Ether [4] achieves ideal transparency because the malicious code is executed on bare metal hardware and the in-guest changes caused by analysis can be intercepted and hidden by the hypervisor. However, Ether is not extensible, because it incurs a prohibitive performance penalty to conduct instruction-level analysis. Our experiment shows an approximately 3000 times slowdown by enabling single-step, not to mention the extra heavy instrumentation needed by in-depth malware analysis. Moreover, it is far more challenging to implement the code instrumentation tools within a hypervisor (i.e. Ring -1) than an emulator (i.e. Ring 3).

Therefore, it is still an unresolved problem to build an extensible and transparent malware analysis platform. We aim to tackle this problem by combining hardware virtualization and software emulation. The essence is *precise heterogeneous replay*. That is, we record malware execution using hardware virtualization for transparency, and then replay and analyze the malware's execution using dynamic binary translation for flexibility and efficiency of in-depth analysis.

The idea of heterogeneous replay was first proposed and implemented in Aftersight [2], which records the virtual machine execution from VMWare and replays it in QEMU, for heavyweight analyses (such as bug detection) on production workloads. In contrast to Aftersight, our platform needs to work under the malicious context: the emulator should exactly replay the execution recorded from the hardware virtualization platform in spite of the fact that malware is trying to detect every possible heterogeneous property in these two systems.

We carefully classify various operations and instructions into several categories and handle them properly to ensure precise replay. More specifically, we choose to emulate basic integer-based instructions for efficiency, directly pass floating point instructions to the FPU, and record and replay the remaining complex instructions, exceptions, interrupts and device I/O. With the assumption that integer-based instructions are easy to emulate correctly, this design choice achieves transparency, analysis efficiency, and extensibility.

We have implemented a prototype in KVM and TEMU. KVM has been modified to transparently record malware execution using hardware virtualization, and TEMU has been enhanced to precisely replay the execution via dynamic binary translation. With minimum changes, the existing analysis plugins (such as taint analysis, unpacker, and tracing) work properly, achieving the advantages of transparency and greater analysis efficiency. Our experiment on the 14 real world emulation-resistant malware samples has demonstrated that our prototype is able to defeat emulation-resistant malware and conduct in-depth analysis with acceptable performance overhead.

References

[1] Anubis: Analyzing Unknown Binaries. `http://anubis.iseclab.org/`.

[2] J. Chow, T. Garfinkel, and P. Chen. Decoupling dynamic program analysis from execution in virtual environments. In *Proceedings of 2008 Usenix Annual Technical Conference*, June 2008.

[3] CWSandbox::Behavior-based Malware Analysis. `http://mwanalysis.org/`.

[4] A. Dinaburg, P. Royal, M. Sharif, and W. Lee. Ether: malware analysis via hardware virtualization extensions. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, pages 51–62, 2008.

[5] H. Yin and D. Song. Temu: Binary code analysis via whole-system layered annotative execution. Technical Report UCB/EECS-2010-3, EECS Department, University of California, Berkeley, Jan 2010.