Review “TCG”

Name: Wenhui Zhang
Email: wuz49@ist.psu.edu

Paper Name：

Sailer, R., Zhang, X., Jaeger, T., & Van Doorn, L. (2004, August). Design and Implementation of a TCG-based Integrity Measurement Architecture. In *USENIX Security Symposium* (Vol. 13, pp. 223-238).

Contribution:

This paper proposes an architecture which measures systems to prove they are running correctly to remote verifiers.This architecture restrict kernel to only execute approved code through monitoring kernel operations to enforce security.

There are 3 main contributions in this paper: (1) they build a “non-intrusive and verifiable remote software stack attestation mechanism” on commodity hardware; (2) they build an “efficient measurement system for dynamic executable content”; (3) they build a “tractable software stack attestation mechanism” on original system without requiring extra hardware components.

Motivation:

Secret and sensitive data was processed along with open data on software stack that is running on remote systems. To make sure that the secret and sensitive data are processed correctly, integrity of the process and data should be monitored and filtered.

Related works:

There are many methods to verify integrity of process of sensitive data calculation on commodity hardwares: (1) it could be done through validating the source of messages from the remote system; (2) it could be done through constraining the system to run only trusted softwares; (3) it could be done through inspecting the runtime state.

Methodology:

In this paper, operating systems support for measuring the integrity of code and structured data is defined. TCG has four steps: (1) isolating secrets belonging to an isolated engine; (2) reporting the identity and behavior of an isolated engine; (3) analyzing integrity of reported identity and behavior of an isolated engine; (4) stop or remove process which is processing secrets if integrity is violated.

In this implementation, hooks are put throughout Linux kernel; TPM PCR is extended using PCR = SHA1(File || PCR) at file load-time; kernel-stored measurement list is extended using a list of SHA1 hashes; attestation are then generated using TPM hardware using $S(K^{-}_{TPM}$, PCR+nonce).

Results:

A design and implementation of a secure integrity check system is accomplished. The kernel implementation is quite light weight, with 4755 LOCs.

Take away:

This approach could be used to mitigate the damage caused by modern ransomware attacks on filesystems. By measuring entropy of read/write on filesystems, we can verify some actions that are violating integrity of normal processes, and stop those processes which are out of spec of kernel-stored measurement list.