[TITLE]
Speculative Virtual Verification: Policy-Constrained Speculative Execution

[AUTHORS]
Michael E. Locasto, Stelios Sidiroglou, Angelos D. Keromytis
{locasto,stelios,angelos}@cs.columbia.edu

[CATEGORY]
Research Paper

[JUSTIFICATION]

SVV represents the next step in the evolution of hardware-level security mechanisms. Early approaches to hardware security focused on providing hardware implementations of cryptographic algorithms. Even contemporary approaches to this topic, such as the TCPA/TCG, only provide tamper-resistant hardware modules to store secrets. The state of the art in this field tries to leverage these stored secrets for attestation, and attestation is typically used for the purposes of DRM. Such uses provide a mechanism for a remote entity to control local execution. There are no mechanisms for the local entity to systematically prevent and control a remote entity from executing local code. Our work on SVV is an attempt to provide a unified model for the supervision of machine instructions.

The job of processors has long been to simply execute code, and getting these components to do exactly that at high levels of performance has been the focus of research and industry development. As a result, security is not integrated into the fabric of execution.

One major claim of this paper is that we need to rethink our assumptions about what it means to execute a program. Systems need to have security and monitoring functionality built in at the lowest levels. This change means that the job of performing logical or mathematical operations on binary strings represents a much more complex task than simply performing the operation. This new type of execution recognizes the implications and consequences of an instruction based on its computing context. Such context includes the sources and destinations of the operands at all levels of the system hierarchy as well as the instruction's relation to previous and future statements.

Modern microprocessors provide mechanisms that support both out-of-order and speculative execution of instructions. The constraints on these mechanisms are typically driven by data and structural hazards. Our work on SVV is an extension of these capabilities that enable the hardware to perform speculative execution based on higher--order constraints. In some

ways, SVV can be thought of as a firewall or filter for machine instructions.

One major advantage of adopting this approach would be the ability to re-write potentially harmful instruction streams. This capability would provide the foundation for automatic remediation.