NSPHD: The Polyglot Computer [*]

Daniel Medeiros Nunes de Castro
Department of Computer Science
University of Calgary
2500 University Drive N.W.
Calgary, AB, Canada T2N 1N4
dmncastr@ucalgary.ca

ABSTRACT

Performing security verification on a compromised system can give a false sense of security. If compromised, a computer system would likely return false information, which could "deceive" any verification process. Our motivation for this work is straightforward: Computers should not be trusted, at least not when they are attesting their own integrity.

In our project Babel, this problem is addressed by, quite literally, thinking outside the box. Babel takes into consideration the advances in computer networks and cloud computing and moves the verification process to outside the physical limits of the computer.

With Babel, we propose an approach that we call "secure co-dependency", that consists of making the user's computer physically unable to execute any program by itself. In this approach, the computer then becomes dependent on an external entity that helps it to execute programs. This external entity, a "security provider", is responsible for verifying each portion of code prior to its execution, assuring that code that is executed by the computer will not be harmful for the computer.

Babel architecture incorporates three main components: a local component in the user's computer; a remote component, that is managed by the security provider; and the communication channel used to perform communication between the user's computer and the security provider.

The local component consists of the actual operating system that runs in the user's computer. We envision it to take the form of a micro-kernel based operating system, in order to reduce the amount and complexity of code that is required to be trusted by the user's computer. Secure co-dependency is enforced by implementing program diversity at the instruction level. Each program is executed within an application VM that is incompatible with the program itself, i.e., its instruction set is different than the one used to create the program. In order to execute the program, the VM needs to incrementally translate portions of it, as it becomes necessary.

That translation happens off-site, and is performed by the remote component of Babel at the security provider, which consists basically of a service running in the network. Along with the translation, the security provider will also carry out its main task: perform security checks to verify if the code is safe to be executed.

With the physical separation between code execution and code verification, Babel introduces an extra challenge for attackers, as a malicious process will not be able to affect the code verification.

Babel could be mistaken for yet another instance of extant approaches, such as remote computing. In this talk, we revisit the Babel architecture with the twofold intention of clarifying what Babel is and showing how Babel differs from previous work.

[*]A full version of this paper is available as a technical report in [1].

NSPW'13, September 9–12, 2013, Banff, AB, Canada.
ACM 978-1-4503-2582-0/13/09.
http://dx.doi.org/10.1145/2535813.2535827.

Categories and Subject Descriptors

C.0 [**Computer Systems Organization**]: General – System architectures; D.4.6 [**Operating Systems**]: Security and Protection; D.4.7 [**Operating Systems**]: Organization and Design

General Terms

Security, design

Keywords

Diversity, client-server, remote computing, virtualization, secure co-dependency

1. REFERENCES

[1] D. M. N. de Castro. The polyglot computer. Technical Report TR-2013-1048-16, Department of Computer Science, University of Calgary, Calgary, Alberta, Canada, October 2013.