Response to NIST: Call for Position Papers on Standards and Guidelines to Enhance Software Supply Chain Security

Area: (4) Initial minimum requirements for testing software source code. See EO Sections 4(e)(iv and v) and 4(r).

Submitted by: Doug Britton, CTO, RunSafe Security, doug@runsafesecurity.com, 571.250.5941

Summary

NIST 800-53 revision 5 recommends Runtime Application Self-Protection (RASP) as an approach to increase the security of the software supply chain [see Section SI-7(17), p.339]. The Executive Order and subsequent analyses are an opportunity to require automated RASP for vulnerability remediation, the presence of which is validated in a software testing phase.

Currently available RASP approaches have wide coverage, from web applications all the way to low-level compiled code. “Software is never done” is a common expression. One reason it’s never done is that it always has vulnerabilities. A brute force, find and fix approach to vulnerabilities is impossible to get right all the time, every time, and does not scale because it requires extremely skilled people. Instead, RASP provides dynamic, moving target defense, protecting against both known and unknown vulnerabilities without the need for human analysts. As a proof point only and not a sales pitch, RunSafe’s Alkemist RASP approach provides an example of the productization of RASP, deployed in the commercial and DoD environments.

Scanning and Patching Leave Gaps

RASP addresses the exploitable security gap left by static scanning and patching in the area of memory vulnerabilities in compiled code. Current research on vulnerabilities and scanning reveals:

- Memory vulnerabilities are the most frequent CVEs reported in the NIST database, making up 40% of the total, and are ranked as the #1 software weakness in MITRE’s current CVE rankings.
- 59% of these memory-related CVEs have exploits available [1].
- Memory CVEs produce the highest CVSS consequence scores, in the 7.2 to 9.8 out of 10.0 (the High and Critical ranges), in NIST’s Common Vulnerability Scoring System. In layman’s terms these are the vulnerabilities that, if exploited, can cause the most damage to an organization.
- Only 2.5% (15 of 592) of all Linux CVEs are currently detected by scanning tools, leaving a massive 97.5% security gap [1].

As an example, memory related vulnerabilities were 55% of VxWorks Urgent/11 discovered in 2019. These vulnerabilities enabled attacks on medical devices, weapons systems, and critical infrastructure.

Patching, while important, is not done in a timely enough matter. The Colonial pipeline attack was successful in part due to lack of patching. More generally, studies say companies aren't patching on time due to lack of knowledge, effort to coordinate change, process slows progress, fear of breaking current setup, and others.

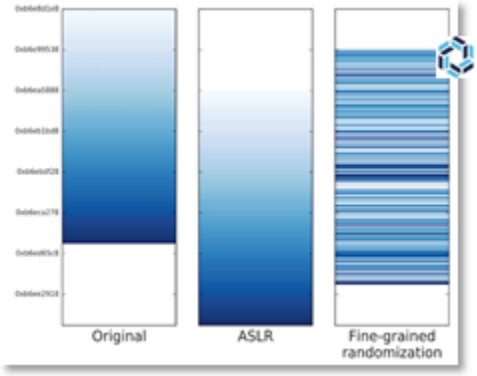
RASP Is Easily Implemented and Has Recognized Value

In addition to unreliable performance, scanning testing tools take time to implement, involving tweaking and tuning, as well as significant time to execute the manual remediation indicated, if even possible. Conversely, RASP tools are added quickly and easily at build-time. RunSafe Alkemist is an example of the productization of RASP theory. Alkemist is successfully integrated into both commercial and DoD CI/CD tool chains for C/C++ source code.

Avocent, a provider of software to server OEMs, recognized the value of automated mitigation of memory corruption vulnerabilities through Alkemist RASP. Avocent cyber hardened their OpenBMC (base management controller) product ACI (Avocent Core Insight) with RunSafe Alkemist. ACI is built on Yocto, and Avocent used RunSafe's Yocto integration process to seamlessly apply protections across the entire ACI image. ACI Firmware is a family of embedded Linux packages that includes a commercial-ready implementation of the OpenBMC open-source project for BMCs. It is highly modular, scalable and includes premium enhancements for added security, performance, and functionality.

Similarly, two Air Force PlatformOne programs - SkiCAMP (Hill AFB) and ThunderCAMP (Tinker AFB) - recognized the need for cyber security beyond compliance through static scanning. They decided to implement Alkemist RASP. They integrated Alkemist into their C/C++ pipelines easily and quickly. ThunderCAMP integrated Alkemist in under a week after a single initial conversation with RunSafe, making Alkemist available to their entire team via a Docker image on their cloud dev tool, Coder Enterprise.

How Alkemist RASP Works



Alkemist provides run-time protection to a wide range of systems without changing original functionality. Alkemist application makes each instance of software functionally identical but logically unique, meaning attackers can't access vulnerabilities with fileless ROP/JOP attacks. Target software includes in-house developed, COTS, GOTS, and open source running on IT enterprise equipment, OT and IoT devices, and specialized embedded systems. Alkemist protected software components are immunized against memory corruption attacks, including zero-days. Alkemist disrupts memory exploitation by varying the attack surface (the layout of your code, see figure). This entropy makes writing reliable exploits extremely difficult for hackers.

Reference: [1] https://arxiv.org/abs/2104.04385