WIND™

Cybersecurity and Secure Deployments

Creating Effective Security with Simulation Technology

WHEN IT MATTERS, IT RUNS ON WIND RIVER

EXECUTIVE SUMMARY

Cybersecurity and secure software deployment are issues that cut across multiple sectors—aerospace, defense, energy, critical infrastructure, industrial automation, medical devices, telecommunications, and more. What these disparate sectors have in common is that a malicious, network-borne intrusion can cause untold damage, whether financial or physical, and even threaten human safety. Fortunately, there are security-strengthening techniques and solutions for cybersecurity research that can apply in all of these areas. This paper explains why creating and testing security solutions is more effectively performed using virtual hardware and system simulation technology, rather than using the live systems that are being subjected to attacks.

TABLE OF CONTENTS

WIND

THE GROWING THREAT

Sophisticated cyberattacks are proliferating globally. Today, with the expansion of the Internet of Things (IoT) and device connectivity, cyberattack targets extend beyond defense and IT to critical infrastructure, aerospace, automotive, healthcare, heavy industry, transportation, and communications—virtually any segment in which there is digital information to steal or misuse, or where there is potential for operational disruption or damage.

Protecting critical systems from network-borne threats and preventing the deployment of infected systems are priorities for both government and industry. Technologies are available today that can give security engineers a considerable advantage in combatting threats. First, though, let's review the current model for cybersecurity research and development.

CYBERDEFENSE: DECONSTRUCTING ATTACKS

Cyberdefense refers to the effort to find ways to protect systems against attacks, including analyzing how attacks happen, how they work, how they play out over time, and their effects, as well as developing countermeasures. Understanding the nature of attacks and uncovering system vulnerabilities is critical in developing effective defense mechanisms.

Defense against cyberattacks involves two primary activities:

- **Defense deployment:** Designing and deploying a coordinated set of protection capabilities, configuring those capabilities to deliver the required protections, verifying the defenses, and maintaining the capabilities with their proper configurations.
- **Forensics:** Investigating how an attack happens, what the attack intends to accomplish, how the intruding element behaves, and how the attack element works. Understanding the nature of an attack in detail is key to developing appropriate cyber countermeasures.

Developing, deploying, and testing effective cyberdefenses in embedded devices is particularly challenging. Embedded devices typically have resource constraints such as limited compute power and processing capacity. They are often designed for a single, unique purpose and employ less widely used busses and interfaces. Setting up test labs to perform system-level cyber testing on a representative set of devices at scale poses logistical and cost challenges. It is also difficult to perform security tests on live systems without "freezing" them entirely, which is not easily accomplished since most systems need to be available at all times. In addition, there is often no backup or redundant service available. While it may be possible to shut down one hardware node and keep the rest of the systems running, this may distort system behavior and therefore not be indicative of how a security measure will perform in a real attack scenario.

Testing cyberdefenses entails such techniques as fuzz testing, or automated testing that injects invalid, unexpected, or random data into a system to determine causes of system failure, and penetration testing (or "pen test"), which involves attacking a system to uncover security weaknesses, gain access to data, and take over or prevent system functions, and then reporting findings to the system owner.

System operators may not even realize they are under attack. Sophisticated attacks can develop over a long period of time, with seemingly random events that in isolation seem harmless, but collectively and over time can cause damage. The cyber chase can be elusive—smart attacks may initially appear as random and simple bugs. Cyberdefense teams must develop countermeasures that are constantly active, that can detect and prevent attacks, and that report attempted attacks to the security team.

Forensics is essentially a form of reverse engineering—investigators work their way backward to identify the root cause of an attack. But many sophisticated attacks are designed to prevent reverse engineering—they burrow and hide below the OS level, in the BIOS or firmware. These attacks may also delete traces of themselves so there is little left for a forensics team to find once the attack becomes exposed. In some cases, attacks can even detect whether they are being analyzed, and change behavior to avoid discovery of their true nature.

INVESTIGATING ATTACKS AND DEVELOPING DEFENSES IN A VIRTUAL ENVIRONMENT

So how can you perform forensics if sophisticated malware is designed to thwart attempts to investigate? How can you detect and remedy vulnerabilities in critical infrastructure systems composed of special-purpose embedded devices? How, in effect, can you become smarter than intruders?

If expense were no object, you could build a so-called "cyber range," a completely isolated network of physical computers whose sole purpose is testing cyber malware and countermeasures—comparable to a golf range for swing practice or a firing range for target practice. But this undertaking is usually very expensive, requiring physical equipment—whether that's an entire aircraft cockpit, power plant equipment, or operating room instruments—all wired together in a lab. The cost and physical nature of a cyber range limit its capacity, which is often significantly lower than the actual need. Furthermore, cyber ranges typically require special skills associated with the unique characteristics and interfaces of a particular system. Given these constraints and the resulting value, a physical cyber range is neither sufficient nor cost-effective for many organizations.

A less costly, more flexible, and more effective alternative is to use virtual hardware and full system simulation technology. There are two advantages to using virtual hardware and simulation:

1. Tests can be performed that are not possible on physical hardware—for example, "tricking" malware into behaving in certain ways, thereby exposing itself and making it impossible to hide.
2. A virtual cyber range can be created, fully scaled out as much as necessary, with all the variants needed to explore the systems, and accessible by any engineer on the cyber research and development team.

Wind River® Simics® exemplifies this type of technology. Simics is a full system simulator; it simulates not only processors and boards, but complete networked systems, on which the full software stack runs unmodified, including the BIOS, firmware, operating system, and software applications. Simics virtual platforms simulate target hardware on which the software is intended to run.

Simics has proven to be an effective cybersecurity research and development tool in the aerospace and defense sectors, and that experience is easily transferable to other industries. Simics can be used to support R&D in a variety of ways:

Undetectable Analysis

Software behaves the same way on a Simics virtual platform as it does on physical hardware. All software, from the application level down to the BIOS and firmware, can run unmodified on Simics. Software build systems and development tools do not require modification, and software is loaded in the same way as on physical hardware. This means that from the software's point of view, there is no distinction between Simics and physical hardware. And in contrast to a debug agent, Simics cannot be easily detected. This means you can perform introspection and non-intrusive analysis of a cyberattack, because the malware does not know that it's executing on Simics, making it difficult to hide.

Cyber forensics engineers have many of the same challenges as BIOS developers—they have to understand exactly how low-level software works. One of the primary applications of Simics is in the development of BIOS and firmware code, where the virtual hardware is developed in "high fidelity" to the physical target.

Studying Attacks: Checkpoints and Reverse Execution

Studying an attack likely involves developing a test case that will demonstrate how the attack works. And when bad behavior appears, researchers must be able to reproduce and analyze it. This becomes a fairly simple matter with Simics checkpoints and reverse execution features. With a system checkpoint, a complete state of the system can be saved, from a single device to thousands of devices, which can be replayed and shared among a team. With reverse execution, cyber engineers can simply reverse time and play the same execution over again, with complete determinism.

Moreover, these capabilities are also completely non-intrusive, so the system cannot observe that it is being stopped, paused, reversed, checkpointed, or restored.

Fault Injection

When the "hardware" is actually software, it can be altered as required—for example, hardware faults can be injected programmatically into the system. With complete control over time, engineers can change and modify the system on the fly to behave in certain ways, or take different routes through execution paths. This capability is very useful for both penetration and fuzz testing. Since everything in Simics can be done by scripting, fault injection can be automated and repeated as many times as needed.

Full Inspection

Complementary to ordinary black box analysis and testing, Simics allows full white box testing and analysis. Simics enables complete insight into the whole system and access to physical and virtual

memory. Anything on the target can be read without being noticed or stopped, including MMU contents, registers, and disk content. Every instruction, memory access, device access, and network packet can be traced and logged. And malware has no idea it is being observed.

Observing Future Behavior with Hypersimulation

Since malware is sometimes designed to cause long-term effects after weeks, months, or even years, researchers need to investigate what may happen to a system in the future, and how small errors converge into larger problems over time. With a physical system, there is only one way to do that—let the system run and monitor the effect in real time. Simics can actually speed up time through hypersimulation and project system behavior into the future.

No Source Code Required

When performing forensics, one may encounter situations where only software binaries are available. This lack of source code could potentially slow down or obstruct an investigation. But because Simics runs unmodified software, portions of the system can be available only as machine code and still be executed and analyzed by leveraging the features of Simics. This is a unique characteristic of Simics relative to other system simulation tools.

SECURE DEPLOYMENT

Developers need to be sure that new software and the products it enables have not been compromised before being deployed—that the system boots and operates securely initially, as well as after an update.

The simple answer would be to test every part of the software before deployment and at every update. The problem is that security is difficult to scale correctly. The more complex the software and computer system, the larger the test matrix, and the more difficult it becomes to achieve the relevant test variation at production scale. Not testing at full scale can put the production system at risk, and this risk is exacerbated with the unrelenting demand for faster deployments. Unfortunately the solution has often been to forego complete test coverage and test only for the most critical use cases on available platforms. Cyber attackers will find those places that were not fully tested.

Fuzz testing is one method that can be applied to evaluate security prior to deployment. For example, engineers can randomly vary inputs to a device, introduce random communication, apply protocol variations, perform range and boundary checks, or check for buffer and register overflows. Randomized testing, however, requires bandwidth, which again raises the issue of scalability.

SOLVING THE CHALLENGE OF SCALE

Security testing requires scalability. Compromises on test variation and test coverage need to be eliminated. Solving this problem requires two key capabilities: automation and parallelization. It is critical to have as much automation as possible, not only to speed up the testing process, but also to achieve repeatability and to be able to report and log results automatically. Running tests in parallel also helps save time—but parallelization is difficult. Not all types of test software can be run in parallel; some is by nature serial. And test parallelization requires the existence of several instances of the same hardware, which is not always practical or affordable.

INSTANT REPLICATION OF TEST ASSETS

Simulation and virtual hardware solve both the automation and the parallelization problems. When hardware is virtual, any amount of target hardware can be instantiated, in any system configuration, instantly. A virtual hardware lab can complement a physical hardware lab, enabling engineers to create the target systems on demand. An automated test system can also be programmed to create new hardware instances and system setups (of both hardware and software) automatically.

Simics can also significantly accelerate test speed through a "snapshot and restore" feature, meaning it can run a system to a specific point, create a snapshot, then run derivative test cases from the snapshot without having to re-run the system to the snapshot point each time.

Simics enables the instant and unlimited replication of test assets. Parallel testing that requires multiple instances of hardware can be run easily with Simics. Alternate system setups can be created so that boards and software combinations are varied to specific requirements, making it possible to complete the entire test matrix, with any number or combinations of hardware variants, OS configurations, communication protocols, and devices.

AUTOMATING THE IMPOSSIBLE

Since in Simics anything is scriptable, time can be controlled, and the system can be altered in any way, it becomes possible to automate what would not otherwise be possible. For example, hardware can be forced to break repeatedly and deterministically. Fuzz testing can be automated in new ways. Test programs can be set up to automatically create any number of new boards, loaded with pre-defined software stacks and able to execute from any given point in any given state. Combined with the ability to change software loads programmatically on a given set of virtual boards, these capabilities allow varying test combinations to be completely automated.

An important and often overlooked aspect of virtual hardware is that it is more stable and reliable than physical hardware. Hardware labs (with hardware lab equipment) tend to be susceptible to failures and sensitive to disturbance. The larger the lab, the more sensitive it can become as complexity increases. Checking results from an automated test system after overnight testing, one may find that tests were broken or interrupted, costing several hours or days in delays. Engineers may also have to spend time analyzing a reported problem to determine whether the issue lies with the system being developed or the test system itself, which cuts into productivity. With virtual hardware, running on stable servers, the test system becomes more trustworthy, and all test teams, regardless of locations, can save time that might otherwise be lost when test automation is performed on physical hardware only.

CONCLUSION

Increasing automation, digital information, and interconnection of critical systems all raise the complexity of developing and maintaining secure systems. Developers of critical systems need tools that can help them stay a step ahead of increasingly sophisticated attackers. System simulation technology provides an efficient and effective means of researching, analyzing, and testing a wide variety of attack methods and security countermeasures in a flexible and scalable environment, and in ways that would simply not be feasible with physical systems. In a world that is ever more dependent on the safe and reliable performance of interconnected systems, simulation gives cyber professionals a way to gain the upper hand.

WIND

Wind River is a global leader in delivering software for the intelligent edge. Its comprehensive portfolio is supported by world-class professional services and support and a broad partner ecosystem. Wind River is accelerating digital transformation of critical infrastructure systems that demand the highest levels of safety, security, and reliability.