TECHNICAL BRIEF

insight.tech

POWERING IoT AND EMBEDDED INNOVATION

Containerized Linux: The Secret to IoT Security

Many software development teams are inclined to base industrial system designs on common Linux distributions, or even develop their own in-house. This is not always advisable from a security perspective, as implementing formal security development practices can be a costly, time-consuming endeavor that distracts from the goal of delivering value-added industrial products.

While the open-source Linux community has a solid history of bug fixes and security patches, this safety net can diminish for development teams over time and as in-house codebases evolve away from the community baseline. Throughout the deployment lifecycle of an industrial embedded device, software development teams will have to assume responsibility for securing not only their core Linux distribution, but also a growing amount and complexity of application code.

Organizations must determine whether the flexibility and agility of open-source Linux-based development environments provide enough value to offset maintaining a secure codebase over the life of their products.

The Value and Cost of Secure Software

Currently, more than 500 active vulnerabilities affect the Linux kernel, ranging from overflows and bypasses to memory and privacy bugs.

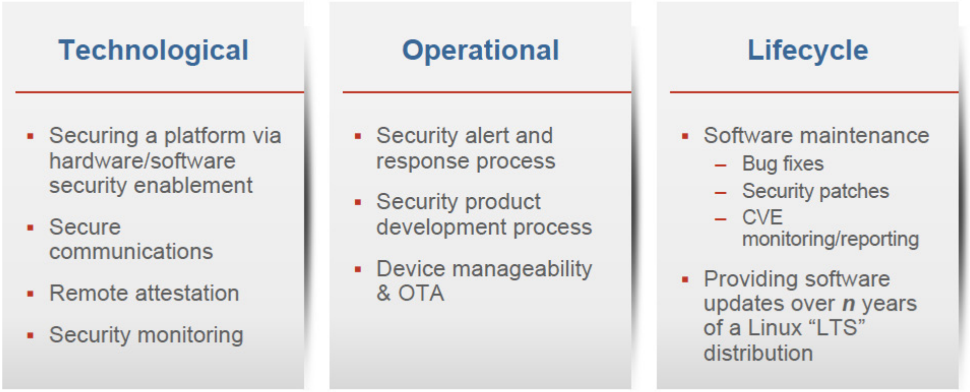
One of the main drawbacks to supporting a secure Linux distribution is that security is difficult to monetize and does not add much value in the eyes of customers. Security is now an expectation, noticed only when it doesn't work.

But from a secure software development lifecycle (SDLC) perspective, there are many high-profile considerations. These include implementing best practices across the assessment, architecture, design, implementation, and deployment phases of the SDLC, as well as provisions for monitoring and maintaining software long after it has been deployed in the field on an IoT device.

Beneath these practices lie strategies for addressing technology, operational, and lifecycle requirements **(Figure 1)**. These include how the software stack interacts with underlying silicon, what development and test tools are used and how, the ways in which third-party services are integrated, encrypted network connectivity, and the device management and update process.

In addition, the nature of IoT devices means that mechanisms for monitoring threats like the common vulnerabilities and exposures (CVEs) listed in MITRE's security database must be in place to safeguard devices in the field as new threats emerge. This requires that development teams are agile enough to identify vulnerabilities quickly, notify clients, and deliver security patches and bug fixes over-the-air (OTA) before damage occurs.

Not only is this infrastructure costly to build and put into practice, it can take years to refine properly.



Technological

- Securing a platform via hardware/software security enablement
- Secure communications
- Remote attestation
- Security monitoring

Operational

- Security alert and response process
- Security product development process
- Device manageability & OTA

Lifecycle

- Software maintenance
  - Bug fixes
  - Security patches
  - CVE monitoring/reporting
- Providing software updates over *n* years of a Linux “LTS” distribution

Figure 1. A secure software development lifecycle (SDLC) requires procedures that address the robustness of code’s technical functionality and operational use over the life of a product. (Source: Wind River)

Bridging Open-Source Flexibility with Industrial-Grade Security

An alternative is to partner with a vendor that already has a secure SDLC and practices in place. Wind River Linux, for example, is a containerized Linux distribution based on Yocto Project tools and processes that provides software engineers with the flexibility of an open-source environment alongside the backing of commercial-grade bug and vulnerability fixes **(Figure 2)**.

The security and flexibility of Wind River Linux begin with its container architecture, which isolates critical software components such as the Linux kernel and user space libraries from applications that would be deployed on an industrial device. Applications are also isolated from one another in these containers, effectively segregating vulnerabilities that may arise in one application from affecting other sensitive code or resources.

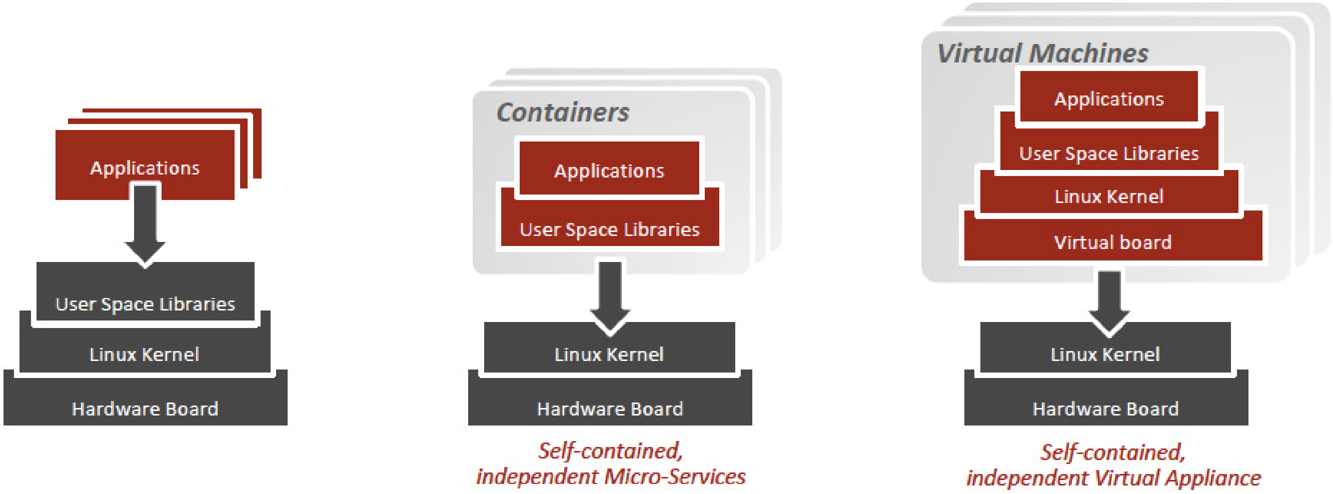


Figure 2. Wind River Linux relies on a container-based architecture that securely isolates software components, eases application integration, and enables targeted application updates. (Source: Wind River)

Not only does this architecture facilitate application integration from other popular Linux distributions like Red Hat or Ubuntu, it eases codebase updating and management by separating software components and reducing dependencies.

The OS offers a variety of select packages and middleware from traditional Wind River Linux, including market-specific profiles for security, virtualization, and carrier-grade functionality. The OS was developed using Wind River's security framework, which takes advantage of security features available on Intel® processors such as Intel® Trusted Execution Technology (Intel® TXT), secure boot, and hardware acceleration of encryption algorithms with Intel® Advanced Encryption Standard New Instructions (Intel® AES-NI).

But the primary benefit of Wind River Linux for industrial organizations and developers is as a turnkey software infrastructure platform. Rather than creating and maintaining their own distribution, customers can use a certified binary image on select hardware targets and maintained by Wind River and the OEM.

As part of this maintenance, Wind River monitors security databases like MITRE for vulnerabilities that could impact software components and also issues patches that can be downloaded directly from an online support system repository by the user. These patches can then be installed on any applicable OS containers.

Open-Source Flexibility, Commercial-Grade Security, or Both?

As the industrial IoT technology market continues to mature, the value of engineering is increasingly at the application layer and not in a system's underlying infrastructure.

Wind River Linux takes this concept to the next level by packaging a commercial-grade, secure distribution with certified hardware platforms that allow engineering teams to focus on application development in a familiar open-source environment. The combined solutions are available as a single bill of materials (BOM) item with maintenance included.

According to a Wind River study, commercial Linux solutions can save industrial organizations up to 53 percent of the total cost of ownership (TCO) of developing, deploying, and maintaining a secure Linux distribution over the life of an IoT product—not to mention the potential ramifications of security exploits.

For more detail on how Wind River Linux makes software engineering organizations more agile, watch the "Providing Secure IoT Platforms at the Edge" webinar.