Traditional AppSec Tools Aren't Enough to Secure Connected Devices & Embedded Systems.

In a comprehensive product security program, traditional AppSec tools and the Finite State Platform complement one another to provide a complete picture of security risk.

While AppSec tools can help an organization scan certain individual components, only scanning the compiled firmware image allows you to see how these components (along with configuration files, drivers, bootloaders, and other parts of the firmware ecosystem) work together and what security issues they introduce.

Think of product security for connected devices like building a house. It's important to assess the quality of lumber before you build the walls. But throughout the construction you need to ensure that the structure is sound. And once it's done you need to test that the windows and doors all lock.

AppSec Vendors	VS	FINITE STATE
Scans application and source code for desktop and server class software.	**Scanning**	Scans devices and firmware.
Piecemeal approach. Vendors have their own proprietary vulnerability databases and rules.	**Vulnerabilities**	**Full context approach.** Detects vulnerabilities in firmware binaries, revealing CVEs, credentials, exploit mitigations, crypto, and more. Leverages open sources and its own database of ~300,000 firmware images. Detects vulnerabilities in operating systems.
Desktop and server class architectures.	**Instruction Set Architecture**	All architectures.
Incomplete SBOM. Doesn't find libraries that were recompiled or modified. Only focuses on visible source packages and misses out on vulnerabilities (which end up in the binary).	**SBOM**	Comprehensive SBOM of open source, custom/first-party, third-party/COTS components.
Depends on the product type. In SAST, there are binary scanners as well as source code scanners. DAST scanners analyze running application. IAST analyzes bytecode.	**Code Analysis**	Analyzes compiled binaries in firmware.
Separate products for Software Composition Analysis (SCA) and custom code analysis.	**Analysis Type**	Performs Device Composition Analysis (DCA). DCA is Finite State's proprietary methodology to unpack the firmware in the entire device.
Only effective for specific languages. Mixed programming languages lead to unrecognized security issues.	**Programming Language**	Language agnostic, captures all security issues as long as binary runs.

By only scanning source code, you are missing vulnerabilities in open source and third party code, which make up 80–95% of device components on average.

DCA can and should be performed throughout the development lifecycle in order to avoid delayed releases and mitigate security issues early in the process.