SUBMITTED ELECTRONICALLY VIA SWSUPPLYCHAIN-EO@NIST.GOV

May 26, 2021

Subject: Position Paper on Standards and Guidelines to Enhance Software Supply Chain Security

U.S Department of Commerce
National Institute of Standards and Technology
100 Bureau Drive
Gaithersburg, MD 20899

To Whom It May Concern:

UL appreciates the opportunity to submit this position paper **on Standards and Guidelines to Enhance Software Supply Chain Security** and applauds NIST's effort to enhance the security of the software supply chain and to fulfill the President's Executive Order (EO) 14028 Improving the Cybersecurity of the Federal Government, issued on May 12, 2021. In line with the goals of the workshop scheduled for June 2-3rd , UL is happy to provide input on software-related standards, tools, best practices and other guidelines to enhance software supply chain security, called for by Section 4 of the EO, with focus on the following two areas: 'Initial list of secure software development lifecycle (SDL) standards, best practices, and other guidelines' and 'Initial minimum requirements for testing software source code'.

Consistent with OMB Circular A-119 and the National Technology Transfer and Advancement Act, UL believes that any effort to create initial requirements for a secure SDL and testing software should be based on existing efforts in the private sector to develop such standards. Furthermore, accredited, third- (3rd) party assessment and verification or certification is crucial to demonstrating security.[1] The ability of software and product developers and manufacturers to credibly demonstrate the performance, safety and security of their systems is critical to establishing trustworthiness and should serve two purposes: 1) to help developers and manufacturers improve the security posture of their systems and solutions by leveraging proven 'security baseline best practices', and 2) to rate the security posture of software systems and solutions in order to make security more transparent and accessible to end users.

UL has been leading efforts to develop several cybersecurity/risk management standards and frameworks to address software supply chain security, with focus on embedded software, and taking these forward in assessment, verification, certification and labeling programs or solutions that are voluntary, rely on market driven mechanisms, and are risk management-based and internationally aligned. Among these, and with applicability for IoT products' SDL and embedded software, is *UL Methodology for Marketing Claim Verification: Security Capabilities Verified to level Bronze/Silver/Gold/Platinum/Diamond, UL MCV 1376* and UL's IoT Security Rating and Secure IoT Component Qualification based on UL MCV 1376.[2] UL MCV 1376's security capabilities are mapped to relevant 'security baseline' standards, frameworks and best practices, including but not limited to NIST SP 800-213, NISTIR 8259 (A-D), CSDE/CTA C2 Consensus, CTA-2088, ETSI EN 303 645, ETSI TS 103 701, draft ISO 27402, the UK Code of Practice for Consumer IoT Security, ENISA Baseline

[1] Software testing and verification or certification by independent 3rd parties – along with the attachment of a visible well-known mark, provides confidence that software, whether application or embedded software, will function according to the developer's or manufacturer's intentions and instructions, as well as comply with industry and government specifications and requirements.
[2] Additionally, UL has for example led development of UL 5500: Standard for Safety for Remote Software Updates and evaluation and certification to the standard, which is designed to set criteria for OTA safety software updates. UL has also led or been instrumental in developing UL 2900 and ISA/IEC 62443 series of standards and evaluation and certification schemes, looking at a secure SDL and OT/IoT system/product security.

UL LLC
1850 M St. N.W., Suite 1000, Washington, DC 20036-5833 USA
T: 202.296.1435 / F: 202.872.1576 / W: UL.com

Security Recommendations for IoT, the CTIA Cybersecurity Certification requirements, NIST SP 800-22 and -57, PCI standards, ARM PSA Certified, Amazon Alexa Voice Service security requirements, among others.

UL's IoT Security Rating measures the security of IoT products and embedded software. Products are tested and classified into one of five security levels, ranging from the lowest level, bronze, to the highest level, diamond. Each level represents a set of security capabilities that level-to-level increases in number and becomes more advanced. To get rated, and listed in UL's Verify database, products undergo an efficient and comprehensive 3rd party assessment process that evaluates and verifies products can meet a 'security baseline' to protect against common attack methodologies and known IoT vulnerabilities.[3] An IoT Security Rating assessment involves technical testing and inspection, but aspects of SDL process review are also included. Every product submitted for assessment is run though a set of tests that – whereas the requirements and levels are risk-based overall, are as prescriptive as possible so that an objective and consistent conformity assessment is ensured. Assessments also consist of review of manufacturer's secure software development lifecycle (SDL) policies, procedures and processes for IoT product security, for example for vulnerability management, software composition (SBOM) and software updates.

UL's IoT Security Rating and the corresponding UL Verified Mark only apply to IoT products. However, IoT products often rely on security functionality provided through specific components.[4] UL's Secure IoT Component Qualification assesses components to any applicable UL MCV 1376 requirements. Results are captured in a Qualification Letter and a Secure IoT Component Qualification Listing in UL's Product iQ database. The Qualification Letter lists which UL MCV 1376 clauses are met by the component stand-alone, and can be met for the product by the component when integrated with the product, if applicable. UL also lists if qualification of the component is based on any other certifications and review of those, e.g. PSA Certified. As a result, through UL's Qualification component vendors can differentiate their component by making security capabilities transparent to customers, industry and end users. Component vendors can demonstrate adherence to 'security baseline best practices', validated by a 3rd party. With qualified components, IoT product manufacturers or OEMs have a more seamless path to meeting 'security baseline best practices' and UL's IoT Security Rating for their products.

UL welcomes and supports NIST's efforts to enhance software supply chain security and also in EO Section (4)(s) initiate pilot programs informed by *existing* consumer product labeling programs to educate the public on the security capabilities of Internet-of-Things (IoT) products and software development practices and to consider ways to incentivize developers and manufacturers to participate in these programs. UL is keen to participate in such pilot programs. We would be honored to represent UL during the upcoming two-day workshop to elaborate further on the value of private sector standards and conformity assessment, the technical aspects of UL's IoT Security Rating/UL MCV 1376, or any of the other efforts that UL has been engaged in to provide transparency into the security of the software supply chain.

Respectfully,

Gonda Lamberink
Senior Business Development Manager, UL LLC

[3] From UL's and industry's experience, today, most successful IoT product attacks are the result of exploitation of common and known weaknesses and vulnerabilities. Taking an 80-20(%) view, a lot of attacks can be prevented by focusing protection on common and proven 'security baseline best practices'.

[4] For example, a hardware chip for secure data storage, a TEE for executing security-sensitive software and functions, a software agent to detect vulnerabilities and attacks on the product in real-time, a library that offers cryptographic functions, or full-stack system software.

APPENDIX: Initial minimum requirements for testing embedded software

UL below provides a breakdown of its security assessment approach for *testing* software, incl. source code, and how this is included within UL's IoT Security Rating assessment and verification based on UL MCV 1376.

We note upfront that some requirements from UL MCV 1376 pertaining to privacy/Personally Identifiable Data (PII) and SDL process and documentation checks are not referenced here, as those requirements in part are validated by inspection of documentation, not through hands-on security assessment activities. Other than that, all the remaining UL MCV 1376 sections/requirements are addressed.

In summary, a structured security assessment based on IoT and cybersecurity standards and best practices involves attempting to discover and exploit software vulnerabilities with extensive hacking techniques – including embedded product firmware and source code evaluation.

Vulnerability Assessment

Vulnerability assessment of embedded products, components and software to identify weaknesses that may be exploitable. Some testing criteria include:

- Vulnerability scanning and binary analysis (UL MCV 1376: 3.4, 3.5, 3.7)
- Examining security controls and circumventing features (UL MCV 1376: 1.3, 1.4, 2.1-2.3, 2.7, 2.8, 4.4)
- Protocol and Packet analysis of communications (UL MCV 1376: 6.1-6.5)
- Cryptography attacks (UL MCV 1376: 2.4, 2.5, 2.6)
- Denial of Service tests (UL MCV 1376: 4.1)
- Hardware vulnerabilities (UL MCV 1376: 1.5, 3.1)

Penetration Testing

Next level assessment, building on the vulnerability assessment, evolving into penetration testing with more complex hacking techniques, including cutting-edge capabilities and vulnerability exploits to validate security hygiene of embedded products and components. Some testing criteria include:

- Spoofing communications (UL MCV 1376: 4.1, 6.1)
- Replay attacks (UL MCV 1376: 6.1, 6.3)
- Credential attacks and exposure (UL MCV 1376: 6.2, 6.4)
- Attempts to elevate privilege (UL MCV 1376: 2.1, 2.2, 3.9, 4.1, 4.5, 6.1, 6.2)
- Customized exploitation

Wireless Communications Protocols

Radio Frequency (RF) protocols and implementation assessments to determine security robustness. Some testing criteria include:

- Design protocols review (UL MCV 1376: 1.3, 2.4, 2.6, 6.1-6.5)
- RF hacking attacks to detect implementation flaws (UL MCV 1376: 6.2)
- Data interception (UL MCV 1376: 6.1, 6.5)
- Information-leakage vulnerabilities (UL MCV 1376: 3.6, 3.8, 4.2)
- Denial-of-Sleep attacks (UL MCV 1376: 4.5)

Testable protocols (examples):

- Bluetooth/BLE
- Wi-Fi, WiMAX
- TLS/SSL
- ZigBee
- Z-wave
- LoRa
- Cellular (GSM family), CDMA, LTE
- Software defined radio (SDR)
- Custom RF

Cryptanalysis

Security assessment of the design, implementation and use of cryptography. Some testing criteria include:

- Manually exploit, credential attacks and cryptography assessment (UL MCV 1376: 2.1, 2.2, 2.4, 2.5, 2.6)
- Identification and assessment of cryptographic techniques for communication and storage of sensitive data (UL MCV 1376: 2.3, 2.4, 2.5, 2.6, 6.1-6.5)

Component and Binary Analysis

Incl. black box reverse engineering if firmware is not provided. Some testing criteria, based on automated (leveraging a variety of tools and hardware) and manual analysis include:

- Binary analysis on a wide variety of architectures, including:
  - Commercial off-the-shelf processors (Intel, ARM, etc.)
  - Custom microcontroller cores with vendor-specific Instruction Set Architectures
  - Writing of custom Linux kernel drivers to be able to fuzz test domain-specific proprietary equipment

Source Code Analysis

As a testing methodology which achieves the highest level of assurance and confidence, UL performs source code analysis through both automated (commercial and open source tools) and manual analysis, to identify known vulnerabilities, common weaknesses, and to evaluate the implementation of specific security controls related to the attack scenarios listed previously.

- Coding languages (examples):
  - C
  - C++
  - Python
  - Java
  - PHP
  - ECMAScript / JavaScript
  - SQL