REVERSINGLABS response to the following areas:

2. *Initial list of secure software development lifecycle standards, best practices, and other guidelines acceptable for the development of software for purchase by the federal government.* This list of standards shall include criteria and required information for attestation of conformity by developers and suppliers. See EO Section 4(e)(i, ii, ix, and x).
4. *Initial minimum requirements for testing software source code* including defining types of manual or automated testing (such as code review tools, static and dynamic analysis, software composition tools, and penetration testing), their recommended uses, best practices, and setting realistic expectations for security benefits. See EO Sections 4(e)(iv and v) and 4(r).
5. *Guidelines for software integrity chains and provenance.* See EO Sections 4(e)(ii, vi, and viii).

Software Security, Reliability and Provenance Validation

Federal Government needs to mandate an independent process that can facilitate Software and Firmware validation of Vendor supplied Security and Quality metrics before acceptance by Consumers. This process needs to apply to:

- the initial software/firmware deliveries
- any updates, upgrades, patch, hotfix, or other components.

The process cannot rely on the availability of source code, debug symbols or any other cooperation from the software/firmware vendors or Open-Source authors.

Validation needs to be performed regardless of the software/firmware platform or software package complexity, programming language or size.

Such solutions are commercially available today and could provide a basis for the future.

Policy Aware scoring using transparent Security Metrics

Security Metrics requested should fall into several categories and provide a scale (e.g., A-F, 1-10) for each critical component surveyed. Each grade needs to describe criteria, remediation recommendation to the developer, and impact/risk assessment for the software/firmware consumer. Grades are rolled up by category to the final risk/grade assessment for a software/firmware package on a worst-case basis. Detailed and describable report can then serve to the software/consumer in applying their custom risk policies (e.g., deploying software with a known vulnerability but lacking a patch).

Critical Security Metric Considered

Some of the key areas that govern whether Software of Firmware is fit (not for purpose) but acceptable for use by Consumers fall in the following categories:

TRANSPARENCY

Validated SBOM (Software Bill of Materials): Third parties need to automatically generate SBOM based on the Software/Firmware package at hand and compare it against the SBOM obtained from a Vendor. NIST can work on a policy framework that should apply when differences are found. Such process needs to have a very broad coverage, automatically

generating a validated SBOM from hundreds of archive, container, installer, update, and package formats. No vendor cooperation needs to be required with a goal to identify all dynamically and statically compiled FOSS and commercial third-party modules. All modules need to be validated, flagging any library misrepresentations (e.g., manually altering version numbers).

TRUST AND RELIABILITY

Counterfeit Software Validation: Each report needs to perform an in-depth validation of all code signing certificates using hundreds of checks (e.g., algorithms, presence of time stamping, revocation, integrity evasions) and producing a risk score to ascertain risk from counterfeit software and assess the trustworthiness of the software/firmware package.

SECURITY QUALITY ASSURANCE AND SOFTWARE RISK

Proactive Software Vulnerability Risk Surface Reduction: Each report needs to survey and score the level of software hardening implemented to reduce the vulnerability risk surface against future/potential software/firmware exploitation. Metrics in this segment intend to harden software against fuzzing and other vulnerability research and provide protection against undiscovered vulnerabilities. Reports needs to survey and score implementation of all available vulnerability mitigation techniques. Measures need to be scored based on the implementation quality and effectiveness. All software should be implementing ASLR, DEP, SEH, and all other (currently) available vulnerability mitigation techniques to inoculate the effect of any undiscovered vulnerabilities.

SENSITIVE AND INAPPROPRIATE CONTENT

Exposed Secrets: Each report needs to itemize a list of possible unwanted components that are present in software/firmware packages: e.g., starting with private crypto keys, debug symbols (revealing source code), PII, access tokens, unwanted software and source code components, default credentials.

MALWARE HYGIENE AND MALICIOUS INTENT

Advanced Supply Chain Malware Canvassing: Each report needs to contain evidence of a total and complete recursive anti-malware analysis for all embedded components regardless of the archive, installer, or platform type. Malware, backdoors, WebShells, red teaming tools, and known CVE Exploits need to be reported along with malware family and actor attribution where applicable and rolled up to the top level regardless of the size of the package and the number of embedded components.

Malicious Intent in Baseline and Incremental Updates/Patches: Starting with a baseline, each Software/Firmware package needs to report on existence of software behavior characteristics used by malware. For Update/Patches such report needs to be reduced just to changes from the last analyzed package. Examples of this type of an analysis applied to Solarwinds Orion can be found here.