# NIST Position Paper #2

GitLab

GitLab comments on the *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* in support of the May 12th 2021 Executive Order on Improving the Nation's Cybersecurity, drafted May 26th, 2021.

> *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).

GitLab is excited to partner with NIST and other federal cyber stakeholders on Standards and Guidelines to Enhance Software Supply Chain Security. As a company that holds transparency as a core value, we are happy to share our learned best software security practices to help influence standards that strengthen the cyber resiliency of federal agencies and the broader cyber community.

Given the Executive Order 14028, the application of the created standards and guidelines needs to extend beyond the Federal Government. The NIST 800-53 Supply Chain Risk Management family of controls should become part of the purchase, and continuous assessment process in absence of applicable security certifications evidencing adherence to supply chain best practices. Further expansion of the controls should include extending the breadth of targets. For instance, the existing set of controls focus on supply-chain relating to the origin or location serving the components. The evidence from the SolarWinds breach indicated that the software in question was from the United States.

Secure software development requires people, processes and technologies to function in unison. The Open-Source Software (OSS) community has a rich history of development that follows various standards and major projects are managed by trusted stewards, but from a security standpoint there are a few distinct advantages. Firstly, as the build process is taking place publicly, the code has been examined by a large number of security personnel. Secondly, the build process of OSS projects also can take place in the open, with many eyes from the security community examining the entire process. Third, high quality software is available from the OSS community that has stood the test of time from an unforgiving Internet. As an example, before implementing a library with security features, or a library into a product that has security features or implications, the code is examined to ensure it is securely written and maintained. Not every organization does this, however enough have so that a lot of security flaws have been fixed. These advantages are unique to the OSS community, as closed sourced software lacks the transparency during its development cycles, build cycles, and bug reporting phases after deployment in the field.

For GitLab's supported product offering, GitLab has developed and evolved a rigorous approach to security and secure development, and openly documented them for review and adoption. It is a three-pronged approach that includes multiple layers of monitoring: Securing the Product; Securing the Company; Assuring the Customer (https://about.gitlab.com/handbook/engineering/security/#security-department).

Underpinning all of these is a collaborative approach with transparent feedback loops. Security Teams continuously collaborate with software development to:

a. Develop real-world security use cases
b. Ideation of compelling security features and products
c. Test and provide feedback to development teams

An additional component of this approach is to have the Security organization embedded in highly frequent software development and release cycles. This process is heavily reliant on incorporating standardized security practices during development, thus not slowing down the delivery and deployment of the software. The high release cadence makes it possible for any critical software patches to be delivered quickly so as to limit the impact and potential blast-radius of any discovered vulnerabilities.

**Verify the Materials and Risk**

Having a Software Bill Of Materials (SBOM) is a first step to knowing what is in the software that is being used. Occasionally, though, certain software vendors offer community developed, open or closed source software "plug-ins" that can be downloaded from a marketplace. Vendors offer such a system to enhance their software's capabilities. These "plug-ins" and community provided components also should be scrutinized for vulnerabilities as well as SBOMs. Such architectures contain an inherent risk that has to be assessed.

Furthermore, validation of SBOMs cannot be a singular event only to be executed during the procurement or installation process. The dependencies should be monitored by the software developers and appropriate patches should be deployed based on risk upon issuance. When introducing new dependencies in the software development process strict vetting should be considered, the dependencies should be examined with respect to overall security posture and development practices implemented by the maintainers.

# In Closing

GitLab appreciates the opportunity to offer these positions to NIST in relation to EO 14028. We have proposed approaches that have proven to be fruitful in the modern software delivery model.

**Submitter:**


- Johnathan Hunt
- VP of Security, GitLab
- jhunt@gitlab.com