Position paper is response to the request for NIST Executive Order on Improving the Nation's Cybersecurity workshop topics.

Submitted by the Kantara Work Group on Federated Identifiers for Resilient Ecosystems.
Chair Jim Kragh - kragh65@gmail.com
Editor Tom Jones - thomasclinganjones@gmail.com
Date 2021-05-25

1 Criteria for designating "critical software."

This response is directed not to the large software purchases for server farms, but to the small software applications that sit at the network's edge. It is at the edge where the users engage and where the future security of the internet will first be tested. The following are some of the applications that sit at the edge and are acquired by federal employees, contractors and remote sites:

1. NIST SP 1800-32 Securing the Industrial Internet of Things: Cybersecurity for Distributed Energy Resources describes the distribution of for digital communication, data, and control of cyber-physical grid-edge devices. We have confirmed examples of attacks on remote water plants where the attacker was trying to poison the water supply.
2. DHS REAL ID is now accepting comments on the use of mobile Driver's Licenses for access to nuclear power plants and airline access by both the civilians and government employees.
3. The VA is installing a patient monitoring system starting with Tacoma Washington where veterans and staff will be accessing data that deals with life and death situations using smart phones.
4. The DOC NTIA First Net has a multi-year history of testing apps that are permitted to attaching to the first responder network run by AT&T.
5. FEMA will be in the field using portable devices to evaluate disaster preparedness and response and will share the effort of First Net to get the cellular network to be back is service as the best way to communicate in any disaster.

This is only a small slice of the many ways that the federal employees, contractors and users will take the experience of the federal network into the most critical situation where many people actually experience the true value of the federal safety net. This is where the federal expertise is felt at the most visceral level. This is where resilience of the network is most important.

Any software running at the network's edge requires a reliable means to assure that federal services can be delivered directly to the people when and where they need it most. This is the place where working software in the hands of the people is critical to the understanding that their government is working for them. This is not the place where software running on the smartphones should be tested for its resilience.

If the end user software was recognized as a critical component of the government's ability to deliver service to the point where it is most critically needed. It is most important the any list of critical software include the final delivery platform, the smartphone. This is a proposal for including that software in the critical infrastructure.

The proposed solution is a set of rules that meet the criteria of the EO to be applied to any smart phone app that is to be trusted with critical data. The Kantara Initiative has built a demo site that explains the operation of the trust registry and application attestation network of labs to assure that any app listed in the trust registry meets the criteria that is now in place for Health care apps and should soon be followed by broader federal criteria for apps in any critical area. The demo site can be found at https://trustregistry.org/.

5 Guidelines for software integrity chains and provenance

As a preamble that addresses section 2 of the RFC, it should be clear that not all software that impacts critical government infrastructure is purchased by the federal government. Much of the federal infrastructure depends on end point software that is controlled and purchased by a vendor or even by an end user. At a minimum it is proposed that all software used in critical applications must be secure. This response focuses on the endpoint software but should work equally well for cloud software that is dependent on interoperating with software from other vendors. The security of the deployed infrastructure depends on strict accountability for every action in the entire supply chain. A good security culture starts with a Code of Conduct that is pervasive and actively enforced at the highest levels of management. The following address the points of the RFC.

1) (i) secure software development environment.
   a) Administrative build environments and their secrets need to be controlled by people that are not developers. This assertion strikes at the heart of "devops" culture which pervades the commercial marketplace. But when developers have access to secret credentials, it will inevitably happen that those credentials will get included in code check-ins and from there into the hands of hackers. This was the attack vector that resulted in the release of 57 million customers and 600,000 drivers for Uber. Without automated tools for checking code for vulnerabilities and managing the use of credentials there is no strict accountability for each code module. Only code modules that are approved by a security team should be provided to developers who should not be permitted to download code from any external source for any development purpose whatsoever. All code check-ins must be monitored and approved. Code builds for deployments must never be able to be altered by developers, which typically means that the build tools are created by a security team and cannot be altered by developers.
   b) Auditing trust relationships requires a knowledge of every component that is part of a software package. As an extreme example, if the gcc build tools download provides functions to evaluate a url which loads a metadata package used to evaluate the trust of a

received packet, all of the tools which might be provided by that package must be evaluated to be sure that the right metadata was, in fact downloaded. This is a known condition for that particular package. That appears to be a test which has not been performed for a tool which is used in that manner. This is in addition to the Evaluation of Compiler-Induced Vulnerabilities which has been known to have been completed.

c) Establishing multi-factor, risk-based authentication and conditional access by enabling certificates and W3C Verifiable Credentials to be composed. This will allow (for example) the use of identify proofing from one source and user authentication from a different source. This method is described in the current committee draft: Distributed Assurance Specification 1_0_0 DRAFT 2.

d) Documenting and minimizing dependencies to develop, build, and edit software is a real challenge in an open-source environment, especially one that uses containers (like Docker). The solution would be to fund changes to some of the open-source tools, like docker-compose, so that they would always create a software manifest as a part of the process that would be included in the container for all to see.

e) Encryption for data is will-known. All that is required is a mandate for encryption in transit and encryption at rest.

f) Monitoring operations and alerts and responding to attempted and actual cyber incidents. This item, like the one below, is dependent on a continuing development team. Any code that is subject of a reported vulnerability, include one in any of the dependencies, must create a bug that is triage and resolved expeditiously. This is common in well-funded software teams but is difficult in software that has been abandoned. Whenever support for a software product ceased, use of the product must cease.

2) (ii) generating and providing artifacts that demonstrate conformance to the processes set forth in subsection (e)(i) of this section. It is expected that existing procedures from the First Net plan described above is a good paradigm to follow.

3) (vi) maintaining accurate and up-to-date data, provenance (i.e., origin) of software code or components, and controls on internal and third-party software components, tools, and services present in software development processes, and performing audits and enforcement of these controls on a recurring basis, The federal government can help the process by requiring automated build tools that always generate a detailed bill-of-materials (SBOM) for use in monitoring.

4) (viii) participating in a vulnerability disclosure program that includes a reporting and disclosure process. Security vulnerabilities are reported through communities such as CWE, CVE, and OWASP. Whenever a dependency is found to have a vulnerability, that code needs to be re-evaluated considering the newfound vulnerability. The implication here is that any code used in any trustworthy application must be backed by an active support site whether that code is proprietary or free open source. There is no such thing a secure code without a continuing development tracking process.