Consumer Technology Association™

1919 S. Eads St.
Arlington, VA 22202
703-907-7600
CTA.tech

May 26, 2021

Consumer Technology Association (CTA) appreciates the opportunity to help NIST with its work under the Executive Order and to participate in the Workshop on Software Supply Chain Security June 2-3.

CTA addresses NIST's request for input on criteria to designate "critical software" as directed in Section 4(g) of the EO. This definition is important because NIST is directed in Section 4(i) to "publish guidance outlining security measures for critical software as defined in subsection (g) of this section, including applying practices of least privilege, network segmentation, and proper configuration." Section 4(h) further directs CISA to "identify and make available to agencies a list of categories of software and software products in use or in the acquisition process meeting the definition of critical software."

We respectfully submit the attached paper as follows:

Title: On Criteria for Designating "Critical Software" (CTA)
Area Being Addressed: 1. *Criteria for designating "critical software"*
Representative:

Michael Bergman
Vice president, Technology & Standards
Consumer Technology Association
mbergman@CTA.tech / +1 (703) 907-4366

Respectfully submitted,

CONSUMER TECHNOLOGY ASSOCIATION

/s/ Michael Bergman
Michael Bergman
VP, Technology and Standards

/s/ Jamie Susskind
Jamie Susskind
VP, Policy and Regulatory Affairs

Consumer Technology Association
1919 S. Eads Street
Arlington, VA 22202
(703) 907-7651

CTA Position Paper On Criteria for Designating "Critical Software"

"Critical Software" Should Be Clearly and Narrowly Defined

NIST should define "critical software" to focus on attributes that have potential to pose a major risk to Government systems and networks if the that software is compromised, and on that software that poses potentially greater risk or consequences if compromised than software designated as "non-critical."

When it comes to criticality, the government is not writing on a blank slate. "Safety-critical software" and "mission-critical software" are categories that the government addresses, in everything from NASA programs to oversight of automotive and aircraft development. These categories are based on a combination of technical attributes of the systems in which software will operate, and the relative risks and consequences of system failures that justify enhanced and more costly development and management. This is appropriate because, without context and related risk, it's practically impossible to define software as "critical." NIST can look to these as examples of how to approach defining "critical software."

In this definition, setting a high bar for "critical" would be sensible to keep too much from being declared "critical." Clearly, when "*everything is critical*," nothing will receive the attention that should be reserved for the smaller number of truly critical elements. It is also appropriate given the other security improvements that the government will use to augment and manage software deployments, like zero trust architecture and the move to secure cloud solutions. These concepts augment the security of the systems in which software is deployed, making it less important to broadly mandate uniform software attributes.

Setting a high bar for "critical software" will also effectuate the EO. The EO directs a particular set of steps be taken for "critical software," and a distinct set of efforts to influence the broader commercial software ecosystem. As such, NIST should articulate functional criteria for "critical software" that target the most sensitive attributes or functions, and which expressly include consideration of the context of deployment and the risks associated with that use. This targeted approach will keep the government's work on "critical software" from becoming de facto regulation of all software.

The software ecosystem is complex and enormous. Much software is acquired as commercial off the shelf items (COTS),[1] and can be used in ways that pose a very broad range of levels of risk. As DoD has said, "'software' can range from off-the-shelf, non-customized software to highly-specialized, embedded software running on custom hardware" making it "critical that the right tools and methods be applied for each type."[2] While a few product categories of COTS may be "critical," generally the approach should be that COTS is not "critical" unless it is in one of a few a priori "critical" categories or possesses certain "critical" attributes, as described below.

In sum, NIST's definition must be tailored to reflect a risk-based approach that identifies a relatively small set of truly "critical software" in particularly high-risk use cases and keeps distinct that which is not truly "critical." This approach will protect Federal information systems and preserve a robust commercial software marketplace.

Criteria for "Critical Software" Should Explicitly Include Consideration of Use by Agencies

The key challenge for NIST is to refine and use these factors to categorize software. To put this in context, SolarWinds' Orion network management and monitoring platform typically ran with powerful administrative privileges had the capability to configure the network and servers, track user devices, and more.[3] Criteria that included any platform with elevated privilege or access would have covered the Orion platform. But just because SolarWinds was a network monitoring tool does not mean all network monitoring tools should be treated as critical, because the category would capture far too much. For example, a read-only network device monitoring system or a browser-based speed test program might fall under a broad category of "network monitoring" tools but they can be fully "sandboxed" in execution, with no

[1] "COTS software is sold in substantial quantities in the commercial marketplace and is purchased without modification, or with minimal modification, to its original form." https://www.gao.gov/assets/gao-21-182.pdf In fact, NIST may want presumptively exclude from the definition of "critical software" all or most COTS software.
[2] Defense Innovation Board, Ten Commandments of Software (Version 0.14, 15 April 2018) https://media.defense.gov/2018/Apr/22/2001906836/-1/-1/0/DEFENSEINNOVATIONBOARD_TEN_COMMANDMENTS_OF_SOFTWARE_2018.04.20.PDF
[3] https://www.solarwinds.com/orion-platform

elevated privileges or access rights. This is why use case and context are so important. A vendor may not know to what uses its software or service will be put by a government purchaser, so NIST's functional criteria must account for the intended deployment.

In a few cases, product categories will be helpful. This approach should be reserved for cases where the type of software has no realistic usage that is not in a sensitive or powerful way. But caution is appropriate here. Product categories might use recognized terms for an entire package, such as "network management software" or "browser." Using product categories has substantial chance of casting too wide a net. Attribute-based categories would instead use capabilities or requirements inherent in the design of the software, such as "must run as root" or provides privileged user management in addition to use case context.

CTA offers some specific comments on each factor identified by the EO.

- The "*level of privilege or access required*" should be interpreted to require elevated privilege or access to function at all. Software in commercial products that does not always require high-risk privilege or access should not be considered "critical." A definition of a Critical Software category that addresses *"level of privilege or access required to function"* must refer to a truly functional requirement (as in, the software will not do useful work without it), and not refer simply to a way in which a user or administrator might choose to configure and use the software. Otherwise the definition will capture far more product than intended or helpful. Otherwise, for example, a simple browser plug-in intended to block web advertising would be considered "critical" because it could be installed and used by a user with powerful administrative privileges.

- As for "*integration and dependencies with other software*," NIST needs to explain what kind of integration and dependency, and with what sort of other software will make something "critical." Every piece of software has some level of integration and dependency on other software. This factor should be described to work in tandem with relevant privilege and access rights, and in light of the protections established in the interrelated software. The browser-based speed test mentioned above will depend on the browser, for example. The browser would have greater rights than the speed test application, but can enforce a "sandbox" policy on the app. As a result, the speed test application should not trigger this criterion because it does not receive additional rights or access via the interdependency.

- Likewise, the criteria for "*access to networking and computing resources*" should be carefully circumscribed. All software will have some access to such resources. The criticality relates to the kind of access and level of control that may be required. This "critical software" definition relates most closely to product design. NIST's definition should enable agencies to clearly determine what networks and resources are sensitive from that product design perspective, ahead of installation and zero trust networking considerations.

- Regarding "*a function critical to trust,*" the definition of "trust" will be important here. NIST should use a strict cryptographic definition of "trust" and "trusted systems."

- "*Potential for harm if compromised*" should be defined in terms of privilege and access, with a focus on risk management in the deployment. It's possible to envision some form of harm from nearly any kind of software compromise, depending on where and how and by whom the software is executed. CTA urges NIST to heed definitions of harm in the security context, for example, and describe the harm required to render software "critical" as akin to "a debilitating effect on security, national economic security, national public health or safety, or any combination thereof."

Conclusion

CTA urges NIST to take a targeted, risk-informed approach to defining "critical software," using a combination of a few unavoidable but well-defined product categories, and a set of required functional attributes. This approach is consistent with the longstanding risk-based approach taken in federal IT security policy and in cybersecurity policy generally.