From:	[redacted] via 800-171comments
To:	800-171comments@list.nist.gov
Subject:	[800-171 Comments] Comments on SP 800-172 r3 draft
Date:	Sunday, January 5, 2025 3:34:11 PM
Attachments:	sp800-172r3-ipd-comments.xlsx

I'm one of the co-authors of the original NIST SP 800-172, and I'm submitting these in my personal capacity. Please find my comments attached.

Best regards,
Ryan

--
Ryan Wagner

Comment #	Submitted By (Name/Org):*	Type (General / Editorial / Technical)	Starting Page # *	Starting Line #*	Comment (include rationale)*	Suggested Change*
		Technical	Througho	Througho	The use of the phrase "process, store, or transmit" omits components that may protect CUI without processing, storing, or transmitting the CUI itself. As an example, asset management utilities or network security scanners may not directly interact with CUI but should be in scope. The CrowdStrike and Solar Winds incidents are reminders of how critical these security-providing systems are to system infrastructure.	Use "process, store, transmit, or protect".
		Technical	11	463	Consider augmenting the number of concurrent sessions with TLS channel binding, which can prevent the reuse of stolen session tokens in other connections.	Add a control enhancement requiring TLS channel binding where the service supports the capability.
		Technical	23	840	Consider including images, binaries, and (for ML) models.	"...baseline configurations, images, binaries, and ML models."

		Technical	24	895	While password management usage is great, this is drifting from one of the original intents for this control, which was the usage of Privileged Account Management (PAM) / Privileged Account and Session Management (PASM). PAM/PSAM have additional capabilities like automated password rotation so components like routers and IOT devices that have only a single account for login can be securely shared amongst multiple administrators without trading off security. A nice writeup of this type of capability can be found here: https://www.oneidentity.com/what-is-privileged-session-management/ Also include other credentials like passkeys here, since they may also be generated and managed.	"Password, Account, and Session Managers. Use...password managers to generate and manage passwords and passkeys. Use...privileged account and session managers to secure and manage credentials for legacy systems in which one login must be shared as well as [organization defined] other critical systems."
		Technical	25	905	The discussion seems to encourage both encrypting passwords *and* storing them offline. It's unclear if the intent is to prohibit centralized password vault storage (e.g., in an on-prem or cloud-based password management server), if this is meant to encourage an offline backup of the password vault, or if the word "or" should be used instead of "and" to indicate a choice between encrypted passwords (stored on a server) or plaintext passwords stored offline.	As one possible option, consider changing "and" to "or."

		General	25	895	This is very focused on passwords. As in a comment above, consider encouraging the use of passkeys or protocols such as Secure Remote Password (SRP) as an alternative to storing passwords on servers. With fewer passwords to store on servers, security is significantly increased.	A new control of "Configure [organization defined] servers to use zero knowledge authentication techniques (e.g., passkeys, Secure Remote Password) instead of sending passwords between client and server as a form of authentication."
		Editorial	38	1328	"Determine" doesn't require further action.	Use "adapt to" rather than "determine."
		Technical	47	1624	Additional techniques to provide partial isolation might be helpful. For example, virtual desktop infrastructure (VDI) or web apps like webmail (instead of a traditional mail client with offline storage) can limit bulk access to data. These partial techniques are worth inclusion.	"In addition to techniques that block specific connections between components, techniques including virtual desktop infrastructure (VDI) and web apps (e.g., webmail instead of traditional mail clients) limit bulk access to CUI."
		General	56	1907	Formal verification appears to be removed. Consider including it again. For example, formal verification of OS kernels or cryptographic implementations can provide significant security benefits. Organizations should prefer the use of software with formally verified critical components. For an excellent example, see: https://www.wireguard.com/formal-verification/	Please re-include a formal verification control so organizations must prefer the use of software with formally verified components.

		Technical	57	1930	The description mentions a key used to create a hash and goes on to mention using a public key to verify a hash. This appears to be a misunderstanding of hashes or a typo. Hashes do not use PKI. Perhaps signatures were meant instead of hashes -- particularly if the intent is signature of hashes.	..."protecting the confidentiality of the private key used to create the signature, and using the public key to verify the signature."
		Technical	58	1943	A hardware root of trust should be explicitly included in the description of methods for maintaining integrity of firmware. Most major operating systems now support hardware roots of trust. A hardware root of trust provides significant security benefits versus software roots of trust, as the hardware is supposed to be nearly immutable.	System components should use hardware roots of trust to verify the integrity of firmware during the boot process.
		Editorial	64	2145	Proper architectural documentation should explain any key decisions (e.g., those that entail significant trade offs). For a security architecture, its alignment with the threat environment should be clear.	Add: 4. Describes key architectural decisions, with explanation of trade offs and rationale for decision. 5. Describes how architecture aligns with anticipated threat environment.
		Editorial	65	2186	Build security in by implementing guard rails to prevent security issues caused by developers. E.g., use of memory-safe languages is a form of "architectural hoisting." For a paper on this, see: https://ieeexplore.ieee.org/document/6834691	Add: d. Intentionally make system architecture and design decisions to constrain developers in ways that prevent the creation of security weaknesses (e.g., through the use of memory safe languages).