Avoiding Unsafe Software

My Preferred Options and Suggestions

to Implement CISA Recommendations

Santosh Pandit

London, 9 Nov 2024

Context:

CISA's recent recommendations have started a ferocious debate. They advise shifting from memory-unsafe languages (like C++) to memory-safe ones. In my view, this recommendation makes sense, as relying on developers to follow secure memory practices perfectly is unrealistic.

My Approach:

Between now and 2035, we will see a significant re-write of software code and underlying libraries. **Table A** below outlines CISA's recommendations alongside my approach as a technology architect of my own laboratory.

I do not expect you to agree with me; however, I do expect you to be aware of the risks posed by insecure software to your business and to plan accordingly to ensure a secure technology architecture.

DMs are always open and welcome! 😊

Santosh

Table A: Some ideas for implementing CISA recommendations

Use	**CISA Topic**	**Options** *(and examples / suggestions)* (Do your own research for suitability)
⊗	Memory unsafe languages	☑ **Rust** *(For systems programming, apps and security)* ☑ **Go** *(For network applications, microservices)* ☑ **Swift** *(For Apple platforms)* ☑ **Spark** *(For critical systems)* ? *Other options*: Kotlin, Zig
⊗	SQL injection	☑ **Parameterised queries** *(Rust crate rusqlite)* ☑ **SQL escaping** *(sqlx)* ☑ **Validate and sanitise** input *(server side please!)* ☑ **Prepared statements** *(most important!!!)*
⊗	Command injection	☑ **Parameterised APIs** *(Rust crate rusqlite)* ☑ **Validate and sanitise** input *(server side please!)* ☑ **SELinux / AppArmor / MIC** ? **Escaping** *(combine with other cautions)* ⊗ Direct shell access
⊗	Default passwords	☑ **Mandatory Password Change on First Login** ⊗ Default usernames
☑	Patch known vulnerabilities before release	☑ **Identify assets** *(including underlying libraries)* ☑ **Vulnerabilities scanners** *(debcvescan)* ☑ **Patch management tools** ? **Automatic updates** *(test first before prod!)*

Use	CISA Topic	**Options** *(and examples / suggestions)* (Do your own research for suitability)
☑	Secure open-source dependencies (SBOM)	☑ **Secure trusted resources** *(repositories)* ☑ **Dependency management / audit** *(Snyk, OWASP)* ❓ **Pinning Version and Release** *(careful !)*
☑	Implement multi-factor authentication (MFA)	☑ **Default: must use** *(Do not allow exceptions for applications relating to financial, healthcare, government, cloud, email, and administrative access)* ☑ **Appropriate MFA method** *(SMS can be risky!)* ❓ **Passwordless applications** *(I do not trust them!)*
☑	Enable logging for intrusion detection	☑ **Remote immutable logs** *(all sources, > 6months)* ☑ **Timely response** ❓ **AI driven automation** *(Double-test before use!)*
☑	Timely publishing of CVEs	☑ **Transparent and timely disclosures** ☑ **Dependency management / audit** *(Snyk, OWASP)* ☑ **Patch history management**
☑	Establish a vulnerability disclosure policy (VDP) – **Here I would go beyond what CISA suggests.**	☑ **Use security.txt** *(and security.html)* ☑ **Operate a bug bounty scheme** *(be open!)* ☑ **Be very fair and reward** *(be generous!)* **My strong message based on personal experience of running a private bug bounty for over five years – Do not be penny-wise and pound-foolish, please.**

Disclaimer: All views and opinions are strictly personal. All errors and omissions are solely mine.