SUBMISSION TO THE OFFICE OF THE NATIONAL CYBER DIRECTOR

For the RFI on Open-Source Software Security

OWASP Foundation, Inc.

8th November 2023

Andrew van der Stock
Executive Director, OWASP Foundation
Colorado Springs

8th November 2023

The OWASP Foundation, Inc. (OWASP) has been at the forefront of application security since its inception in 2001. The OS3I's RFI on *Open-Source Software Security: Areas of Long-Term Focus and Prioritization* is timely and urgent. OWASP is the peak not-for-profit in application security and represents a large global community of application security professionals and developers.

OWASP has dedicated itself to fostering an open environment where the United States can remain ahead of strategic competitors or current and potential adversaries. By concentrating on fields that are research and development-intensive industries, OWASP aligns with the national interest in maintaining technology leadership and peer status among allies and partners. Our commitment is evident through the development of open-source tools and educational resources that contribute to a more secure digital infrastructure.

The OWASP Foundation is volunteer-driven, including the OWASP Board, chapter leaders, project leaders, and members. We support innovative security research with grants and infrastructure, providing activity and investment in the builder, breaker, and defender roles.

All OWASP tools, standards, documents, videos, presentations, and chapters are free and open to anyone interested in improving application security. We advocate approaching application security as a people, process, and technology problem because the most effective approaches to application security require improvements in these areas.

OWASP is not affiliated with any technology company, although we support the informed use of commercial security technology. OWASP produces materials in a collaborative, transparent, and open way. Our freedom from commercial pressures allows us to provide unbiased, practical, cost-effective information about application security.

OWASP is incredibly well-placed to assist the US Government, its agencies, and contractors through various open-source tools, standards, documents, and training. We detail areas for investment that may help achieve the goals and focus areas defined in the RFI.

We eagerly anticipate the outcome of this collaboration and stand ready to provide any further assistance, leveraging our collective expertise and resources to bolster the nation's competitive edge and security posture.

Sincerely,

Andrew van der Stock, Executive Director

Introduction

We at the OWASP Foundation (OWASP) recognize the profound significance of the RFI on Open-Source Software Security initiated by the Office of the National Cyber Director. As proponents of a more secure digital ecosystem for over 20 years, we believe collaborative efforts, especially involving communities like OWASP, are essential for creating robust solutions for software security challenges.

Memory Safety Defined

A 'memory-safe language' ensures a program does not touch memory areas it should not. Imagine you have a set of rules that stop a program from reaching places where it could either grab things that are not there (which slows it down) or peek at something it is not supposed to (which is a security risk).

Memory Safety Vulnerabilities

When a hacker wants to exploit a memory-unsafe language, they will be looking to exploit what is known as a write-what-where primitive, which can allow a malicious actor to put an arbitrary program into an executable location. It is a powerful exploit and often easy to overlook. Memory-safe languages make it impossible for those exploits to get past the compiler.

For optimal software protection, OWASP recommends utilizing programming languages designed with strong security measures. This approach significantly reduces the likelihood of unauthorized code manipulation. However, constant awareness is essential, as exemplified by the Log4j incident, which exposed vulnerabilities within a language typically deemed secure.

OWASP's view of priorities

Which potential areas and sub-areas of focus should be prioritized?

OWASP believes the following areas should prioritized:

- **Financial Support for Open-source**: Secure funding is essential to sustain open-source software education and development.
- **Developer Education and Accreditation**: Strengthen software security by providing comprehensive training and certification for developers.
- **Software Supply Chain Integrity**: Implement systems to help users identify and avoid insecure software components.
- **Broad Vulnerability Reduction**: Adopt standards like OWASP's ASVS and SAMM to eliminate widespread software vulnerabilities.
- **Legacy System Security**: Focus on improving memory safety in applications written in older languages like C, C++, COBOL, or Fortran.

What areas of focus are the most time-sensitive?

OWASP views the acquisition of funds and the provision of developer training as priorities. Without financial support, developing and implementing educational initiatives is impossible. Given the continuous nature of code development, it is essential to produce educational materials that meet

academic standards promptly. This ensures that new code is crafted with higher security and that existing code undergoes thorough vulnerability assessments.

What technical, policy, or economic challenges must the Government consider?

For every Red Hat or Mozilla, with hundreds of paid open-source contributors, most open-source software is produced by a single unpaid volunteer at night for surprisingly large and heavily used projects. Their ability to produce secure code is only as good as their education, desire to move to secure frameworks and languages, and ability to fund their education and certifications. They often will have no formal secure development lifecycle (SDLC) nor tooling to help find common vulnerabilities at compile time.

> **Action:** The Government should provide community organizations such as OWASP the means to help individual contributors, such as funding the creation of open-source static code analysis tools (SAST) or allowing developers access usually expensive resources through free programs with a free or low barrier to entry.

Open-source communities like OWASP exist all over the world. So do open-source developers. A US-centric or US-only approach will be doomed to failure.

> **Action:** The US Government should lead in promulgating secure software for all, regardless of country, because, in the end, the US Government, its agencies, and contractors will benefit from the result.

Trying to force mandatory requirements under US laws or regulations may fail simply because the compliance burden is too high.

> **Action:** OWASP advocates that any necessary regulations or laws are aimed at those who can afford to comply, such as contractors, large ISVs, agencies, etc.

Secure Open-Source Foundations

Fostering the adoption of memory-safe languages

OWASP strongly supports moving from C and C++ to memory-safer languages such as Rust, Go, Java, C#, Swift, etc. The days of needing the absolute speed of near-assembly language are far outweighed today by the speed of code generated by modern memory-safe compilers.

Moving To Memory Safety

The solution is simple: a transition towards memory-safe languages can be gradual. However, all existing unsafe architectures that would be hard to refactor immediately must be “wrapped” in a memory-safe language, utilizing a foreign function interface.

To mitigate ongoing memory safety in existing codebases, we recommend referencing the memory-safe matrix, guiding the intentional movement from memory-unsafe languages to safe languages based on their ease of syntactic transition (meaning their code patterns are often similar enough to keep similar architecture through transition).

The aim is to provide a reference for developer organizations to choose a clear path forward to transition critical infrastructure and provide the optimal use case for each popular memory-safe language. We expect this reference to evolve as memory safety patterns and languages evolve.

Mitigating Composite Vulnerabilities in Memory Safe Languages

Modern software comprises custom code, third-party libraries or frameworks, and open-source components. An accurate inventory of all components enables organizations to identify risk, allows for greater transparency, and enables rapid impact analysis. CycloneDX was created for this purpose. Importantly, CycloneDX uniquely verifies several layers into a codebase's dependency tree, making it easier to detect memory unsafe patterns when the composite vulnerability is known. Simply put, identifying if Log4j and JNDI are inside your codebase, even if one is several layers "deep," is now easier to scan and mitigate.

Additional Considerations

Shifting to memory-safe programming languages reduces exposure to a subset of common security vulnerabilities that hackers favor due to their simplicity and widespread occurrence.

However, it is crucial to recognize that this does not equate to complete immunity from attacks. While these languages present more complexity for would-be attackers, they are not impervious to hacking attempts. Memory-safe languages are designed with security as a priority, which helps guard against a range of known exploits.

For instance, the Rust programming language exemplifies the balance between security and efficiency. It is less susceptible to specific exploits because it enforces strict data handling requirements, like needing strings in UTF-8 format. This aids in the computer's swift and secure data processing. This illustrates the synergy between maintaining robust security measures and achieving operational efficiency.

Reducing Entire Classes of Vulnerabilities at Scale

Secure Development Practices – OWASP SAMM

OWASP SAMM, a software development maturity model and a benchmarking methodology, comprises fifteen secure development lifecycle activities that organizations should aspire to. SAMM is simple enough that an individual developer working independently can do most SAMM activities but complex enough to make a real difference in producing secure software at any organization. Many governments and organizations have adopted SAMM. As a free and open standard, SAMM has a low cost of compliance, which is a crucial issue for many organizations facing hefty requirements, such as PCI DSS compliance audits.

> **Action:** OWASP recommends the US Government review and adopt OWASP SAMM, particularly if their agencies or contractors have no defined secure development lifecycle or practices. It can be adopted progressively as funds or resources allow.

What does memory safety not address?

Memory safety is not a panacea. Memory safety helps with a bug class not highly present in web application languages, frameworks, and APIs - buffer overflows. OWASP notes that memory-safe languages do not address the following bug classes, including insecure, insufficient, or missing:

- Architecture
- Authentication and Session Management
- Authorization
- Input validation and output encoding
- Cryptographic Flaws
- Error Handling
- Data Protection and Privacy
- Secure communications
- Malicious code checks
- Business Logic Flaws
- File and Resource Handling
- API Security
- Configuration

These are the principal areas of OWASP's Application Security Verification Standard not addressed by memory safety, covering some 280 controls.

Simply re-coding an insecure application into a memory-safe language, such as Rust, does not address these issues. This is why OWASP will be recommending the OWASP Application Security Verification Standard Level 1 as the base minimum for all applications to be considered even somewhat secure. This applies to systems, line of business, web or mobile applications, and APIs equally.

Web application languages and frameworks are memory-safe(r)

Most web programming languages and tools like C#, Node.js, TypeScript, and Java are built to be safe from memory errors. Usually, you do not have to change your code to make it safe.

OWASP recommends that US Government agencies and critical infrastructure providers regularly perform secure code reviews and static code analysis, as well as consider other testing methodologies or tools, such as IAST, to ensure that the application has fewer avoidable vulnerabilities. Developers should be trained in secure coding methodologies. Agencies and contractors should have a software security maturity program to benchmark their secure software delivery lifecycle (SDLC).

Strengthening the Software Supply Chain

OWASP strongly believes that securing the supply chain is everyone's responsibility – from compilers and build tools emitting software bill of materials (SBOM's), to end consumers having free and open access to details of what software they are running so they can make informed decisions.

OWASP has several significant tools in the software supply chain area, all open source and free to download, use, and contribute.

> **Action:** OWASP encourages adopting and using OWASP's already existing tools and standards by the US Government, its agencies, and contractors.

OWASP CycloneDX – Software Bill of Materials Standard

OWASP CycloneDX is the premier method of describing a software bill of materials. This is an interchange format that allows vendors and authors to create JSON and XML documents containing all

the known components and their versions in an easy to consume format. The vast majority of SBOM tooling is CycloneDX enabled. If all software came with SBOM's, it would make identifying software with faulty or vulnerable components trivial. OWASP supports CISA's SBOM-everywhere initiative.

OWASP Dependency Track – Dependency Monitoring

OWASP Dependency Track is an intelligent Component Analysis platform that allows organizations to identify and reduce risk in the software supply chain. Dependency-Track takes a unique and highly beneficial approach by leveraging the capabilities of Software Bill of Materials (SBOM). This approach provides capabilities that traditional Software Composition Analysis (SCA) solutions cannot achieve.

OWASP Dependency Check – Dependency Discovery

OWASP Dependency Check is a Software Composition Analysis (SCA) tool that attempts to detect publicly disclosed vulnerabilities contained within a project's dependencies. It does this by determining if there is a Common Platform Enumeration (CPE) identifier for a given dependency. If found, it will generate a report linking to the associated CVE entries.

OWASP Software Component Verification Standard

The Software Component Verification Standard (SCVS) is a community-driven effort to establish a framework for identifying activities, controls, and best practices, which can help identify and reduce risk in a software supply chain. Managing risk in the software supply chain is important to reduce the surface area of systems vulnerable to exploits and to measure technical debt as a barrier to remediation.

Developer Education

OWASP has a great deal of freely available training videos and presentations. However, developers need a way to consume secure software development education in a guided manner. One of the member benefits of OWASP was through third-party companies such as SecureFlag, AppSec Engineer, Secure Code Warrior, and we45, all of whom provided free access to their tools.

As detailed in the next section, OWASP is currently undertaking an effort to create an open-source tertiary secure software education syllabus and framework, as well as industry education, certification, and more. OWASP strongly believes that developers need to know how to code securely and would if only they had open, free access to that education.

Case Study - OWASP Juice Shop – Developer Training

OWASP Juice Shop is the latest in a long line of OWASP-developed deliberately vulnerable web applications. Juice Shop is a modern application written in a memory-safe language (Angular / TypeScript), with a node.js API backend. Juice Shop helps teach application security professionals and developers how their code can be exploited and how to fix the code. As Juice Shop is written in a memory-safe language with extensive end-to-end testing, considering that it has over 180 known vulnerabilities demonstrates that simply transitioning to a memory-safe language or framework is insufficient to protect applications from exploitable application vulnerabilities.

Sustaining Open-Source Communities and Governance

OWASP Foundation, Inc. as a 501 (c) 3 not-for-profit entity, supports a community of over 65,000 application security professionals and developers. We are actively looking for funding to help deliver the following programs:

OWASP Education Committee **Tertiary Academic Syllabus and Framework** **$750k**

The OWASP Education Committee has been working on an open-source tertiary syllabus and framework that would allow any university or tertiary institution to adopt OWASP's academic syllabus for secure development aimed at software engineers and computer scientists. The funds would fund workshops, completion of the syllabus, the use of full-time academic researchers for a year, and a train-the-trainer program.

OWASP Education Committee **Industry Syllabus for Developer Education** **$350k**

As most developers have not received a single day of secure coding education, OWASP intends to make a standard curriculum for third-party training providers to adopt and give worldwide. The funds would be used to develop materials of high quality within OWASP in concert with industry providers, both in memory safe languages such Rust and Swift, but also in other commonly used languages.

OWASP Education Committee **Developer Education Certification** **$1.25m**

Once we have the materials for industry training, the next step is to create certification programs for both delivery providers (to ensure that they are accredited and qualified to deliver the materials to a high standard) and developers. OWASP will use the funds to develop the certification programs with an external certification provider, such as Pearson Vue or similar, and ongoing certification maintenance costs.

OWASP CycloneDX **ECMA Standardization** **$700k**

OWASP CycloneDX is already the premier software bill of materials standard in use worldwide (the other is SPDX, which is supported by some but not all tools). The next step is to fund the four primary developers to produce a standardized version of CycloneDX to promote interoperability and adoption via ECMA standardization. We envisage this to take six months of initial work, followed by a commentary period per ECMA standardization processes.

OWASP has other funding opportunities, but these would fall outside the scope of the RFI.

International Collaboration

OWASP, as the peak web application security not-for-profit, welcomes a collaboration between OWASP and the US Government, and other similar organizations, such as the Linux Foundation's OpenSSF, SecureCode, and other not-for-profit cyber security organizations.

All too often, governments, agencies, organizations, and contractors are required to adhere to many conflicting standards. OWASP has been aligning its core standards where it makes sense to do so:

- OWASP Application Security Verification Standard 4.0 deliberately aligned with NIST 800-63 and some elements of NIST 800-53, to lighten the burden of compliance with the ASVS.
- OWASP was recently inducted into ECMA, the global standards organization, to help standardize OWASP CycloneDX.

We have several ready-to-go initiatives (or are already underway and seeking additional funding). With over one thousand leaders, 6,500 financial members, and over 65,000 regular participants, OWASP is the best-placed organization to help the US Government improve the security of open-source projects and tooling.

Many OWASP projects can serve as foundational tools and resources in achieving the objectives set out by this RFI for low to no cost that are also seeking funding to accelerate their development:

- **OWASP MAS**: *RFI Focus Area Addressed:* Secure Open-Source Software Foundations; Developer Education. *Connection:* Serves as a guideline for mobile application security, supporting adoption of secure configurations and practices in mobile open-source software development and providing training resources for developers. (**$43,240**)
- **OWASP BLT**: *RFI Focus Area Addressed:* Behavioral and Economic Incentives. *Connection:* Aids developers in managing vulnerabilities, incentivizing secure development practices through streamlined vulnerability management tools. (**$426,000**)
- **OWASP Dependency-Track**: *RFI Focus Area Addressed:* Strengthening the Software Supply Chain. *Connection:* Ensures a risk-free software supply chain by tracking dependencies, supporting secure and privacy-preserving attestations, and automated tracking of complex code dependencies. (**$416,000**)
- **OWASP SAMM**: *RFI Focus Area Addressed:* Secure Open-Source Software Foundations. *Connection:* Provides a framework integrating security within development, promoting secure programming practices, and reducing vulnerabilities, essential for fostering the adoption of memory-safe languages and secure configurations in open-source software. (**$801,000**)
- **OWASP Juice Shop**: *RFI Focus Area Addressed:* Developer Education. *Connection:* An educational tool for developers with practical examples and training opportunities on prevalent software vulnerabilities, supporting security and open-source software education and training. (**$17,200**)
- **OWASP Core Rule Set**: *RFI Focus Area Addressed:* Secure Open-Source Software Foundations. *Connection:* A comprehensive set of generic attack detection rules for web application firewalls strengthening defense against emerging attacks on web applications, used by many providers of critical infrastructures and secure web applications. (**$332,000**)
- **OWASP Coraza:** *RFI Focus Area Addressed:* Secure Open-Source Software Foundations. *Connection:* An open-source engine for web application firewalls, strengthening defense against emerging attacks on web applications. ($80,000)
- **OWASP ASVS**: *RFI Focus Area Addressed:* Secure Open-Source Software Foundations. *Connection:* Establishes universal security controls for web application development, supporting secure by default configurations and fostering best practices in open-source software development. (**$90,000**)
- **OWASP OpenCRE**: *RFI Focus Area Addressed:* Significantly improve International Collaboration on Security Standards. *Connection:* As a universal resource, it unifies security standards globally,

aiding in identifying and harmonizing international priorities and dependencies in open-source software security. **($40,000)**

- **OWASP Dependency Check**: *RFI Focus Area Addressed:* Strengthening the Software Supply Chain. Connection: Dependency Check discovers vulnerable components in both the build and delivered phases. Keeping Dependency Check up to date with the thousands of new vulnerabilities each year is not easy for one volunteer. Funding would allow for a full-time contractor to maintain Dependency Check and improve the accuracy of results. (**$250,000**)

Conclusion

The OWASP community has the potential for further innovations. We encourage an expansive collaboration, inviting the US Government to contribute additional projects and tools to OWASP or assisting OWASP in delivering our existing and new programs. By leveraging OWASP's worldwide expertise, we can ensure a comprehensive approach to enhancing open-source software security.

Through the investment of approximately $5.5 million of US Government funds, OWASP will project manage and disburse funds to a set of projects that have a proven track record, including the development of free developer education around Rust and Swift, developer certification and strategic improvements to mature projects to take them to the next level.

OWASP's commitment to fostering a secure web environment resonates deeply with the goals of the RFI. As we progress, embracing and integrating insights from such esteemed communities will be paramount in elevating open-source software security.