Software Engineering Institute

Supporting the Use of CERT® Secure Coding Standards in DoD Acquisitions

Tim Morrow (Software Engineering Institute)
Robert Seacord (Software Engineering Institute)
John Bergey (Software Engineering Institute)
Philip Miller (Carnegie Mellon University and ClickMedix LLC)

July 2012

TECHNICAL NOTE
CMU/SEI-2012-TN-016

Acquisition Support Program
CERT® Program
Research, Technology, and System Solutions Program
http://www.sei.cmu.edu

Carnegie Mellon

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

This report was prepared for the

SEI Administrative Agent
ESC/CAA
20 Schilling Circle, Bldg 1305
3rd floor Hanscom AFB, MA 01731-2125

Table of Contents

Abstract **vii**

1 Introduction **1**
1.1 Background 1
1.2 Document Organization 3

2 The Secure Coding Initiative and Secure Coding Standards **4**
2.1 The SCI 4
2.2 Secure Coding Standards 4

3 An Approach for Implementing CERT Secure Coding Standards in DoD Acquisitions **8**

4 Sample RFP/Contract Language **12**
4.1 Section C: Statement of Work (SOW) 12
4.2 Section L: Instructions to Offerors 13
4.3 Section M: Technical Evaluation Criteria 14
4.4 Section J: Contract Data Requirements List (CDRL) 14
4.5 Impacts on Other Acquisition Documents 15
4.5.1 Acquisition Strategy and Acquisition Plan 15
4.5.2 System Engineering Plan 15
4.5.3 Risk Management Plan 15
4.5.4 Test and Evaluation Plan 16

5 Conclusion **17**

Appendix Mapping of the STIG Guidelines to the CERT Secure Coding Standards **18**

Acronym List **45**

References **47**

List of Figures

Figure 1:	The Software Security Ecosystem	4
Figure 2:	CERT C Secure Coding Standard Wiki: Index Page	6
Figure 3:	CERT C Secure Coding Standard Wiki: Sample Recommendations and Rules	6
Figure 4:	CERT C Secure Coding Standard Wiki: Sample Risk Assessment	7
Figure 5:	Milestone Framework	8
Figure 6:	Contractual Context and Approach for Integrating Secure Coding Standards	9

List of Tables

Table 1: Vulnerability Severity Codes 18

Table 2: Mapping of STIG Guidelines to CERT C Secure Coding Standard 18

Abstract

The United States Department of Defense (DoD) increasingly depends on networked software systems. One result of this dependency is an increase in attacks on both military and non-military systems as attackers look to exploit software vulnerabilities. Program acquisition offices are emphasizing information assurance to address various threats. The Defense Information Systems Agency (DISA) created the Application Security and Development *Security Technical Implementation Guide* (STIG) in response to DoD Directive 8500.IE, which establishes policies and assigns responsibilities for achieving DoD information assurance. That STIG provides guidance for information assurance and security throughout a program's lifecycle, and it is specified as a requirement for DoD-developed, -architected, and -administered applications and systems that are connected to DoD networks.

This technical note provides guidance to help DoD acquisition programs address software security in acquisitions. It provides background on the development of secure coding standards, sample request for proposal (RFP) language, and a mapping of the Application Security and Development STIG to the CERT® C Secure Coding Standard.

1 Introduction

1.1 Background

Increasingly sophisticated exploits of software vulnerabilities are occurring with greater frequency. For example, the Aurora attack launched on Google, Adobe, and several other large companies in January 2010 was designed to retrieve valuable files from compromised machines and featured a different attack approach from what we have generally seen in the past. More recent and more alarming for the nation's security was the Stuxnet malware attack orchestrated through the first publicly known worm to target industrial control systems and take control of real-life physical systems. These attacks have raised awareness within DoD acquisition programs of the need to adequately protect software-intensive systems.

To provide some foundation to this discussion, we use the following definitions (originally provided in *A Structured Approach to Classifying Security Vulnerabilities* [Seacord 2005]) to provide context for this report:

- A *security flaw* is a defect in a software application or component that, when combined with the necessary conditions, can lead to a software vulnerability.
- A *vulnerability* is a set of conditions that allows violations of an explicit or implicit security policy.
- An *exploit* is a piece of software or a technique that takes advantage of a security vulnerability to violate an explicit security policy.

Microsoft's policy of providing patches for its products on the second Tuesday of every month is an example of post-deployment remediation of vulnerabilities. These patches fix security flaws in the software used in Microsoft's applications and operating system—flaws that may have already been exploited. Vulnerabilities are associated with many aspects of a software artifact including, but not limited to, the environment in which software is running, architecture, design, source code, and the machine code to which a source is mapped. The patch process is a necessary but insufficient and expensive means of securing networked systems. One concern is that DoD-acquired systems cannot afford to have patches provided on a monthly or quarterly basis. These systems are safety- and life-critical systems that need to work reliably in order to safeguard our nation. Because it is possible for their software to contain vulnerabilities that adversaries could exploit, the developers of those systems must strive to build software that is free from known code-related vulnerabilities. To reduce the susceptibility of those systems to attacks, the DoD should only acquire systems from contractors whose code conforms to secure coding standards.

To help DoD acquisition programs and organizations acquire more secure software and systems, the DoD issued Directive 8500.1E on information assurance. This directive "establishes policy and assigns responsibilities to achieve DoD information assurance (IA) through a defense-in-depth approach that integrates the capabilities of personnel, operations, and technology, and supports evolution to network centric warfare" [DoD 2007]. This directive applies to all information systems the DoD owns or controls and that receive, process, store, display, or transmit data.

Examples include systems that control weapons, sensors, and enterprise resource planning. The defense-in-depth approach produces layers of technical and nontechnical solutions that

- provide appropriate levels of confidentiality, integrity, authentication, nonrepudiation, and availability
- defend the perimeter of enclaves
- provide appropriate degrees of protection to all enclaves and computing environments
- make appropriate use of supporting IA infrastructures

Section 4.18 of the directive is particularly relevant to this report. It requires all IA and IA-enabled IT products that are incorporated into DoD information systems to be configured in accordance with DoD-approved configuration guidelines. In 2011, the Defense Information Systems Agency (DISA) released Version 3, Release 4 of the Application Security and Development (AS&D) *Security Technical Implementation Guide (STIG)* for use as a DoD-approved security configuration guideline [DISA 2004]. That STIG is designed to help organizations design, develop, test, deploy, and maintain secure applications. It is specified as a requirement for applications and systems that are developed, architected, and administered by the DoD and that are connected to DoD networks.

Based on this guidance, DoD acquisition programs specify IA requirements in requests for proposals (RFPs) that potential bidders must address in their proposals. These requirements impact the bidder’s proposed software development and testing efforts. For example, a DoD contractor might develop coding standards, as a normal part of its software development process, to enable its development teams to follow a uniform set of rules and guidelines. Doing so allows the contractor to produce more consistent and better-documented code and to address its use of particular language features. The use of coding standards is also mandated in AS&D STIG guideline APP2060.1: “Program managers will ensure the development team follows a set of coding standards” [DISA 2004]. These coding standards also need to address the other guidance provided in the AS&D STIG, including the need to identify and mitigate coding practices that are known to produce code that is vulnerable to exploitation. Going forward, coding standards must provide guidance on developing secure alternatives that satisfy the AS&D STIG with the objective of reducing or eliminating vulnerabilities before the code is deployed. This requirement means that secure coding standards need to be developed so that a reliable and repeatable metric for evaluating software security can be used.[1] Later in this report, we present other requirements and artifacts to address the impacts on the software development and testing process.

[1] Software security is related to software safety, reliability, and overall quality. However, these attributes are outside the bounds of this discussion.

The Carnegie Mellon® Software Engineering Institute (SEI) set out to address the need for guidance and support in this area by forming a Secure Coding Initiative (SCI) within its CERT® Program. That initiative coordinates the development of secure coding standards by security researchers, language experts, and software developers using a wiki-based community process.[2] More than 500 contributors and reviewers have participated in the development of secure coding standards on the CERT Secure Coding Standards wiki [SEI 2012a]. The SCI also supports efforts in integrating coding standards into development processes and developing compliance measures.

A secure coding standard is a carefully vetted enumeration of mitigations of security defects that have previously resulted in exploitable vulnerabilities. Faithful application of secure coding standards can eliminate the introduction of known source-code-related vulnerabilities. Achieving this highly desirable result requires a secure coding standard that is sound and complete. To address this need, the CERT Program has released a secure coding standard for C [Seacord 2008] and Java [SEI 2012b], and is readying a standard for C++ [SEI 2012c] and Perl [Seacord 2010].

With the objective of helping acquisition offices acquire software and systems that are free from known vulnerabilities, this report provides guidance for and an approach to satisfying the AS&D STIG requirements with the SCI's products. The report also includes sample RFP and contract language, and a mapping of the STIG to the CERT C Secure Coding Standard.

1.2 Document Organization

This document is organized as follows:

- Section 1 provides an overview of the document and background information.
- Section 2 describes the CERT SCI.
- Section 3 provides an overview of the approach for implementing secure coding standards.
- Section 4 offers sample RFP/contractual language to use in acquisition programs.
- Section 5 summarizes this report.
- the appendix maps the AS&D STIG guidelines to relevant secure coding standards.

® Carnegie Mellon and CERT are registered trademarks owned by Carnegie Mellon University.

2 The CERT C Secure Coding Standard wiki is located at https://www.securecoding.cert.org/confluence/display/seccode/CERT+C+Secure+Coding+Standard, and the CERT Oracle Secure Coding Standard for Java wiki is located at https://www.securecoding.cert.org/confluence/display/java/The+CERT+Oracle+Secure+Coding+Standard+for+Java.

2 The Secure Coding Initiative and Secure Coding Standards

The SCI's mission is to address software vulnerabilities in source code. The CERT Program has been cataloging vulnerabilities and their root causes and mitigations since 1995. Figure 1 illustrates the software security ecosystem in which these activities occur.

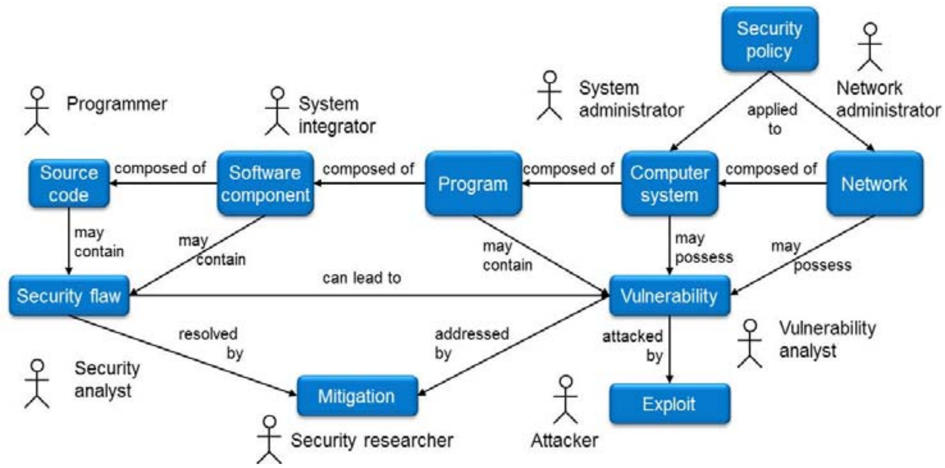


Figure 1: The Software Security Ecosystem

The critical activity loop in the development of a secure coding standard consists of the community at large reporting vulnerabilities to the CERT Program. However, the effort is much broader than the few engineers working at the CERT Program. It includes many users, developers, software companies, international standards organizations, and experts in languages, security, compilers, static analysis tools, and so forth.

2.1 The SCI

The CERT Secure Coding website [SEI 2012d] describes and supports the SCI's activities, and lists the SCI's five major areas of work:

1. secure coding standards
2. international standards development
3. the Source Code Analysis Laboratory (SCALe)
4. development tools and libraries
5. TSP-Secure

2.2 Secure Coding Standards

The SCI's core activity is developing secure coding standards for commonly used programming languages such as C, C++, and Java.[3] Activities two through five above support this core activity,

[3] Going forward, the SCI anticipates taking on additional languages.

promulgate the standards, and help the world's software community apply the standards. The CERT secure coding standards are collections of guidelines for a particular language that, when faithfully applied, allow software developers to write programs without any of the code-related vulnerabilities that are known at the standard's publication time. As of July 2012, the CERT C Secure Coding Standard [Seacord 2008] and the CERT Oracle Secure Coding Standard for Java have been released [SEI 2012b]. The CERT C++ Secure Coding Standard is in the works but not ready for formal release [SEI 2012c].

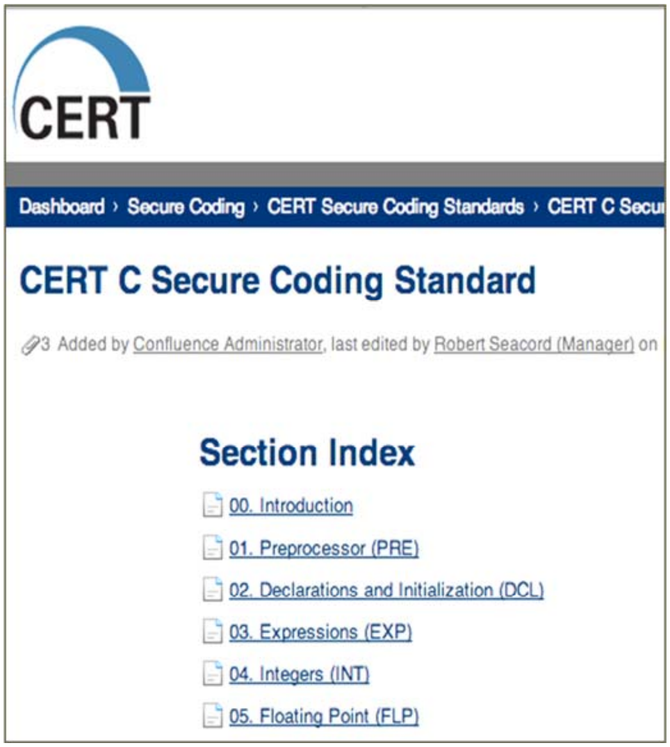
Although developing CERT secure coding standards is the SCI's responsibility, the initiative draws heavily on the experience and expertise of the world's software development community through the CERT secure coding wiki.[4] The wiki incorporates input from hundreds of expert developers, educators, and security researchers, and other industry experts. The general public's access to this wiki is limited to read-only, but they are welcome to submit comments on the overall standards and particular guidelines. The SCI maintains editorial control over each secure coding standard.

The wiki is organized by language, then by subject within each language, and then by specific rule or recommendation. Rules and recommendations include the statement of the guideline, examples of compliant and non-compliant code, implementation details, risk assessment (including likelihood, severity of impact of exploitation, and remediation cost), availability of automatic detection, and so forth. The screenshots in Figure 2, Figure 3, and Figure 4 show the CERT C Secure Coding Standard [SEI 2012d] in successive levels of detail.

The wiki section for a particular language is released as a formal secure coding standard when the SCI determines that

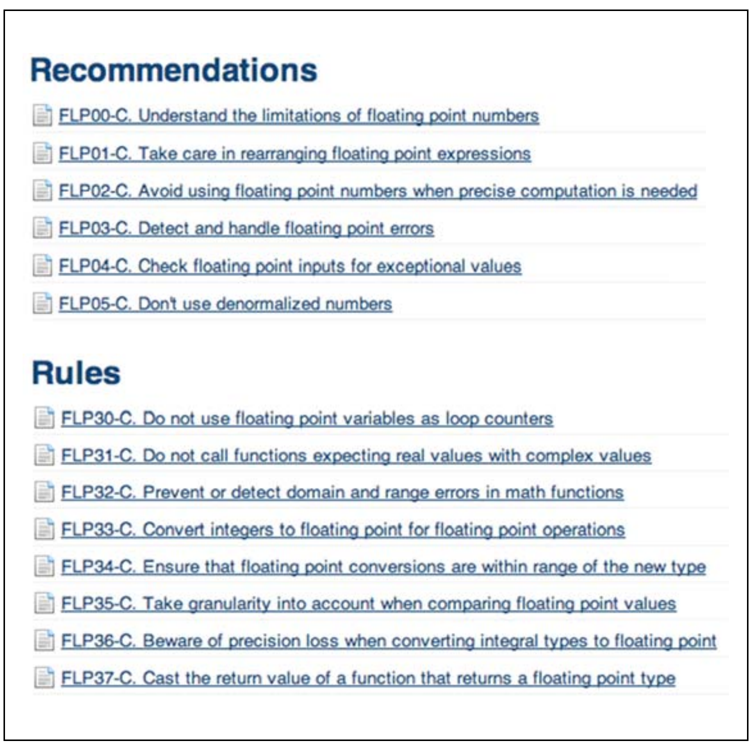
- all known vulnerabilities have been addressed
- input from experts has been included
- tool vendors have had an opportunity to contribute their thoughts
- all meaningful comments have been discussed
- the entire wiki has been thoroughly vetted

4 You can access the wiki from http://www.securecoding.cert.org/confluence/display/seccode/CERT+Secure+Coding+Standards [SEI 2012a].



Each wiki includes an index of sections. The first 6 of 21 sections and appendices are shown here.

Figure 2: CERT C Secure Coding Standard Wiki: Index Page



Here, we drill down into Section 05, the Floating Point guidelines. Headings for all six floating point recommendations and all eight rules are displayed.

Figure 3: CERT C Secure Coding Standard Wiki: Sample Recommendations and Rules

Risk Assessment

The use of floating-point variables as loop counters can result in unexpected behavior.

Rule	Severity	Likelihood	Remediation Cost	Priority	Level
FLP30-C	low	probable	low	P6	L2

Automated Detection

Tool	Version	Checker	Description
LDRA tool suite	V. 7.6.0		
Fortify SCA	V. 5.0		can detect violations of this rule with CERT C Rule Pack
Compass/ROSE			

Here, we see the risk assessment and automated detection parts of FLP30-C, the floating point rule that prohibits the use of floating point variables as loop counters.

Figure 4: CERT C Secure Coding Standard Wiki: Sample Risk Assessment

As noted above, SCI activities two through five support the correct use of secure coding standards in various ways. Below are short descriptions of those activities:

1. international standards development: The SCI participates in the development of international standards for programming languages to improve the security of these languages.
2. Source Code Analysis Laboratory (SCALe): The SCI's SCALe offers conformity assessments of software to CERT secure coding standards. SCALe analyzes existing software to improve confidence that it does not present known, code-related vulnerabilities. SCALe also provides a gap analysis detailing the work that needs to be done to bring software up to the relevant security standard.
3. development tools and libraries: The SCI has developed tools and libraries that help software developers reduce the number of vulnerabilities in their code. Static analysis tools specifically target secure coding guidelines, while runtime tools monitor things that are difficult or impossible to completely assess at compile time, such as writing outside the bounds of an object.
4. TSP-Secure: The SCI and the SEI's Team Software Process[SM] (TSP[SM]) team are collaborating to extend TSP to include the guidance from the secure coding standards. This collaboration brings secure coding standards, and the tools that support their implementation, to the software developer workbench. When organizations implement TSP-Secure, they can efficiently build high-quality, secure software while conforming to Capability Maturity Model Integration[SM] (CMMI[®]) [Davis 2009].

SM Team Software Process, TSP, and Capability Maturity Model Integration are service marks of Carnegie Mellon University.

® CMMI is a registered trademark of Carnegie Mellon University.

3 An Approach for Implementing CERT Secure Coding Standards in DoD Acquisitions

As shown in the previous section, a number of resources support the use of CERT secure coding standards in software development organizations, but there appears to be little incentive to integrate this knowledge into an organization's approach for future DoD acquisitions. For this reason, program offices should specify in their RFPs the use of the CERT secure coding standards in order to improve the security and quality of the software being developed, and then they should analyze the standard's implementation in the software being developed. This approach provides several benefits. It

- provides guidance as to how secure coding standards could impact the milestones and Contract Data Requirements Lists (CDRLs) specified in the RFP
- gives the development organization a chance to evaluate the impact of using the CERT secure coding standards in its development processes
- helps the development organization to better understand the program office's expectations and to create a better estimate and schedule for the program's lifecycle
- enables both the development organization and the program office to obtain training so they can efficiently implement the coding standards into their development process

The Milestone Framework shown in Figure 5 and the Contractual Context and Approach for Integrating Secure Coding Standards shown in Figure 6 can set the acquisition and contractual context.

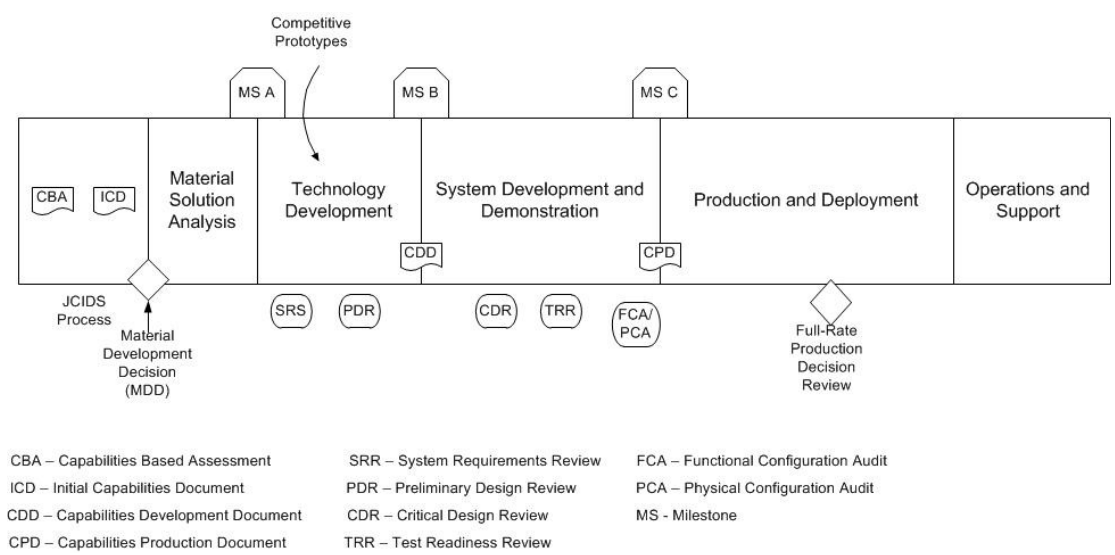


Figure 5: Milestone Framework

In Figure 6, items that the government program office specifies as part of the contract are shown in blue, while items that a contractor would be responsible for producing in the contract are shown in green.

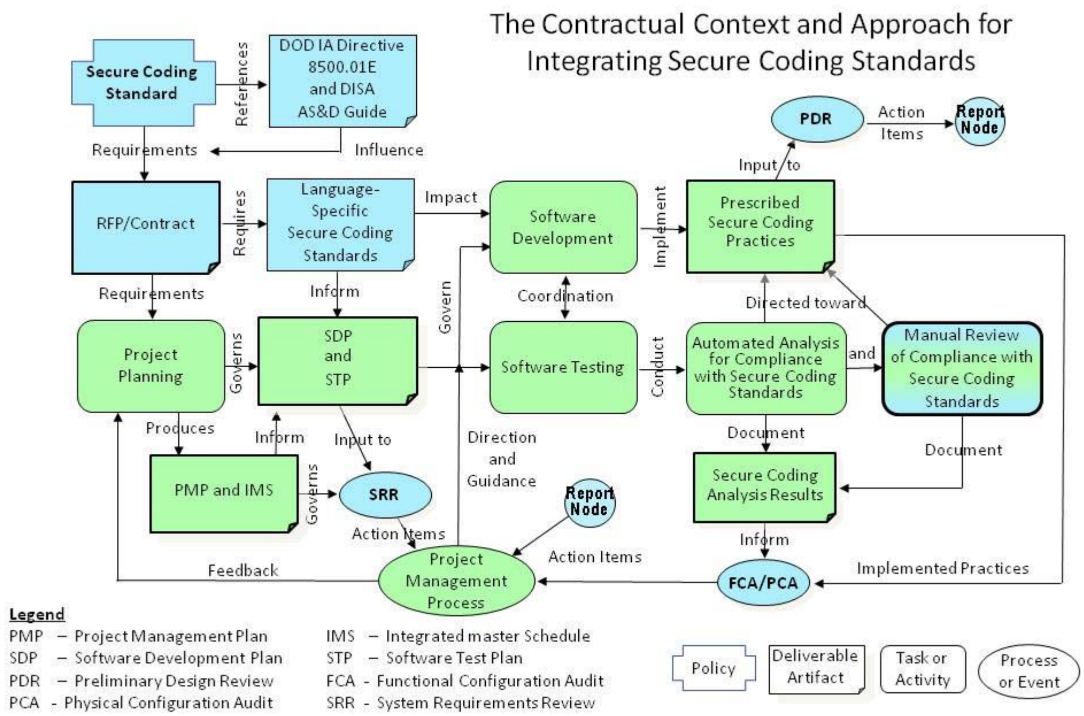


Figure 6: Contractual Context and Approach for Integrating Secure Coding Standards

As shown in the Milestone Framework, one of the key activities in the Technology Development phase is competitive prototyping. The CERT secure coding standards are to be specified in the contract to address the development efforts for the prototypes, and they continue to be used throughout the development lifecycle. As shown in Figure 6, DoD IA Directive 8500.01E and the DISA AS&D STIG specify the need to use coding standards in development efforts. The RFP will identify these two documents as requirements for the acquisition. Further specifying the use of CERT secure coding standards for incorporation into the coding standard will help satisfy a number of other requirements also specified in the AS&D STIG. The appendix provides a mapping of the AS&D STIG guidelines to the CERT C Secure Coding Standard.

Following the approach presented in Figure 6, four CDRLs (which should be included in the RFP and reviewed) assess the understanding and implementation of secure coding standards in the development process. These CDRLs are the Program Management Plan (PMP), Integrated Master Schedule (IMS), Software Development Plan (SDP), and Software Test Plan (STP). It is a good idea to request that draft versions of these CDRLs be included in a bidder's response to the RFP and that updated versions be provided at key milestones in the acquisition.

The PMP will include the staffing and level of effort required to implement the secure coding standards. This implementation process might include evaluating the impacts to an organization's existing coding standards, training the organization to successfully use and follow the secure coding standard, and estimating the potential impact on development tool evaluations being considered for use in the program.

The IMS reflects the schedule of tasks to satisfy the timeline identified in the RFP, with additional information detailing the activities identified in the PMP. Reviewing how the tasks in the IMS are integrated can help you understand how well the bidder understands the impacts of using secure coding standards.

Two additional documents are affected: the SDP and the STP. In the SDP, the bidder should provide details to address

- the activities that impact or influence the PMP
- the amount of effort needed to tailor the secure coding standard to the bidding organization's processes and thereby satisfy the RFP and contract requirements

The SDP should address how the bidding organization will embrace the secure coding standard so the entire development team follows it faithfully. The tools planned for the development effort could be impacted. The STP should address the compliance testing, including any training needed by the test teams.

These CDRLs should be updated and reviewed again at these key milestones:

- system requirements review (SRR)
- preliminary design review (PDR)
- functional configuration audit (FCA)
- physical configuration audit (PCA)

In the next section, we provide sample language that supports the implementation of a secure coding standard.

At the SRR, the PMP, IMS, SDP, and STP are updated by the selected contractor to reflect its understanding of the contract. Between the initial release of the RFP and the signing of the contract, a number of changes typically occur in the requirements. Updating the PMP, IMS, SDP, and SDP CDRLs to support the SRR provides the contractor with the opportunity to demonstrate its understanding of the contract to the program office and to reflect that understanding in these four documents.

At the PDR, the software architecture has been finalized, and the contractor is in the process of planning software development and software testing. For software development, the secure coding standards influence the selection of development tools. The developers, testers, and IA and quality assurance personnel may undergo training that has been identified as necessary.

For software testing, the STPs should identify how compliance with the secure coding standard is demonstrated. If tools are going to be used to evaluate compliance to the secure coding standards, they will have to be configured and integrated into the development process. A well thought-out approach to testing is available through SCALe [Seacord 2010].

The IA group might include legitimate, documented deviations from the secure coding standard. In that case, those deviations must also be included in the program's IMS.

The final lifecycle phases impacted are the FCA and PCA. Entering these phases indicates that the code has been deemed mature enough to begin acceptance testing. The program's IMS should account for the effort required to analyze the code and ensure that defect removal and late changes have not introduced anything that violates the secure coding standard. After this analysis is complete, the code is ready to be handed off to external organizations for further analysis and compliance to the AS&D STIG.

4 Sample RFP/Contract Language

The sample RFP and contract language provided in this report has been shared with and reviewed by DoD acquisition program personnel, but to date it has not been included in an actual DoD contract. Therefore, the sample language may need to be customized to comply with local contracting requirements, policies, and program-specific requirements.

The purpose of this contract language is to

- specify the contractual requirements needed to ensure that the secure coding guidance is applied properly in DoD acquisition programs
- provide a common and equitable basis for enabling all potential offerors to appropriately respond and estimate the cost of their effort to support the secure coding guidance

The goal of this language is to identify ties to program CDRLs and milestones so the contractors and the acquisition organizations can evaluate and plan for the effort required to support the implementation of secure coding practices.

4.1 Section C: Statement of Work (SOW)

The following language shown in blue italics below is the primary text that an acquisition organization needs to include in an SOW.

For incorporating a secure coding standard:

> *The contractor shall integrate the use of one or more secure coding standard(s) into its development process for the <to be filled in> software.*

For specifying the CERT C Secure Coding Standard:

> *All systems requiring the development of custom software should use a secure coding standard for each selected programming language to promote secure programming practices. As a neutral Federally Funded Research and Development Center (FFRDC), the Software Engineering Institute (SEI) can be used as a source of coding standards for <to be filled in> systems. If custom software is being developed in the C programming language, then Version 1.0 of the SEI CERT® C Secure Coding Standard shall be used as the starting point for a secure coding standard. Information provided on the CERT C Secure Coding Standard should be considered for interpreting Version 1.0 of the CERT C Secure Coding Standard [Seacord 2008].*
>
> For specifying the CERT C++ Secure Coding Standard: *If custom software is being developed in C++, then the CERT® C++ Secure Coding Standard is to be used as the starting point until the standard has been released. The acquisition organization will work with the contractor to develop the secure coding standard to be used on the program [SEI 2012c].*

For specifying the CERT Perl Secure Coding Standard:

> *If custom software is being developed in Perl, then the CERT® Perl Secure Coding Standard is to be used as the starting point until the standard has been released. The acquisition organization will work with the contractor to develop the secure coding standard to be used on the program [Seacord 2010].*

For specifying the CERT Oracle Secure Coding Standard:

All systems requiring the development of custom software should use a secure coding standard for each selected programming language to promote secure programming practices. As a neutral Federally Funded Research and Development Center (FFRDC), the Software Engineering Institute (SEI) can be used as a source of coding standards for <to be filled in> systems. If custom software is being development in Java, then The CERT® Oracle Secure Coding Standard for Java is to be used as the starting point for a secure coding standard. The acquisition organization will work with the contractor to develop the secure coding standard to be used on the program [SEI 2012b].

For incorporating a corresponding SDP:

The contractor shall produce, update, and maintain a Software Development Plan (SDP) document for the <to be filled in> software using the contractor's configuration management control system and deliver the SDP document in accordance with <SDP_CDRL_Identifier>. The Software Development Plan (SDP) shall describe how the secure coding standard is integrated into the development process. The SDP shall indicate the activities that need to be performed prior to the start of development, such as training in secure coding and ensuring the development process will produce source code that conforms to the secure coding standard(s).

For incorporating a corresponding STP:

The contractor shall produce, update, and maintain a Software Test Plan (STP) document for the <to be filled in> software using the contractor's configuration management control system and deliver the STP document in accordance with <STP_CDRL_Identifier>. Test and evaluation of software shall include validation of conformance to the secure coding standard in the STP. It is expected that it will be accomplished with automated analysis tools and manual reviews.

4.2 Section L: Instructions to Offerors

A SDP as part of the RFP:

The Software Development Plan (SDP) should describe how the secure coding standard is integrated into the software development process. The SDP should indicate the activities that need to be performed prior to the start of development, such as training in secure coding and ensuring the development process will produce source code that conforms to the secure coding standard(s).

As a neutral Federally Funded Research and Development Center (FFRDC), the Software Engineering Institute (SEI) is the preferred source of coding standards for <to be filled in> systems. If custom software is being developed in the C programming language, then the SEI CERT® C Secure Coding Standard shall be used. In the case of other programming languages, the program manager will work with the program information assurance system engineers to develop a secure coding standard based on industry best practices, especially in cases where an SEI standard does not exist.

A STP as part of the RFP:

Test and evaluation of software should include validation of conformance with the secure coding standard in the Software Test Plan (STP). If custom software is being developed in the C programming language, the CERT SCALe effort [Seacord 2010] could be consulted for guidance. It is expected that the conformance verification will be accomplished with automated analysis tools and manual reviews.

4.3 Section M: Technical Evaluation Criteria

A SDP as part of the RFP:

> *Does the Software Development Plan (SDP) address the use of a secure coding standard? Does it discuss how the secure coding standard is integrated into the development process? Does the SDP indicate the activities that need to be performed prior to the start of development? Does training in secure coding ensure that the development process will produce source code that conforms to the secure coding standard(s)?*

A STP as part of the RFP:

> *Does the Software Test Plan (STP) include validation of conformance with the secure coding standard? If custom software is being developed in the C programming language, then the CERT SCALe effort could be consulted for guidance [Seacord 2010]. Does the STP discuss the types of validation used (automated analysis tools, manual reviews)?*

4.4 Section J: Contract Data Requirements List (CDRL)

Program Management Plan (PMP)

In Section 16, "Remarks," of the PMP CDRL, the following information should be added as relevant to secure coding standards:

> *The PMP will include the staffing and level of effort required to put the secure coding standard into use. This includes, but is not limited to, any training needed for the development team to understand how to use the secure coding standard and training on additional tools that are unique to secure coding. The PMP will also need to assess new rules and recommendations on a periodic basis to address new threats and mitigations, as well as update the secure coding standard appropriately.*

Integrated Master Schedule (IMS)

In Section 16, "Remarks," of the IMS CDRL, the following information should be added as relevant to secure coding standards:

> *The IMS will identify the tasks and staffing needed to support the secure coding standard as identified in the PMP, SDP, and STP.*

Software Development Plan (SDP)

In Section 16, "Remarks," of the SDP CDRL, the following information should be added as relevant to secure coding standards:

> *The SDP will address the activities identified that impact or influence the PMP, as well as the effort to tailor and integrate the secure coding standard to address the organization's software development lifecycle and processes. The SDP should address how the organization will embrace the secure coding standard such that the entire development team faithfully follows the standard. The secure coding standard will impact the code review process, so the SDP should address any training needed by the development team to be able to understand and apply the secure coding standard.*

Software Test Plan (STP)

In Section 16, "Remarks," of the STP CDRL, the following information should be added as relevant to secure coding standards:

> *The STP will address the activities identified that impact or influence the PMP, as well as the effort to tailor the secure coding standard to address the organization's testing processes. The STP should address how the organization will embrace the secure coding standard such that the entire verification and validation (V&V) team faithfully follows the standard. The tools planned for the V&V effort should be evaluated for compliance with the standard. The STP should address any training needed by the V&V teams to support the standard.*

4.5 Impacts on Other Acquisition Documents

To make sure the use of a secure coding standard is integrated throughout the acquisition process, it must be discussed in the program's Acquisition Strategy, Acquisition Plan, System Engineering Plan, Risk Management Plan, and Test and Evaluation Plan.

4.5.1 Acquisition Strategy and Acquisition Plan

Specifying the use of a secure coding standard and integrating it into the software development lifecycle

- should improve the software's quality
- are risk mitigation efforts to produce code with no known vulnerabilities

The acquisition program office needs to address the costs associated with the effort to integrate secure coding standards into the program's development lifecycle, along with supporting information that indicates how that integration will save money throughout the program in the Acquisition Strategy and Acquisition Plan.

4.5.2 System Engineering Plan

The plan should indicate

- that secure coding standards will be used in the software development lifecycle
- how that use will affect test, evaluation, and security/IA

How the program is planning to reduce the software's vulnerability should play a key part in producing a reliable, more cost-effective system.

4.5.3 Risk Management Plan

The plan should identify

- the process for identifying potential threats as program risks
- the mitigation process for addressing those threats if they are determined to be program risks
- a way to categorize the risk that is relevant to the program

4.5.4 Test and Evaluation Plan

The plan should address how DoD Directive 8500.1E and the AS&D STIG are being handled by the program. The plan should also address how the secure coding standard impacts software development from low-level unit testing and code reviews to the system integration efforts and security considerations.

5 Conclusion

DoD acquisition programs are required to address DoD Directive 8500.1 and the supporting security configuration guideline (AS&D STIG). This requirement has impacts across the DoD acquisition program's lifecycle that are identified and addressed in a contractual context in this document. The CERT C Secure Coding Standard is mapped to STIG guidelines to show how the STIG is being satisfied as related to coding standards. This document also provides guidance to DoD acquisition programs that are addressing Java and C++. CERT secure coding standards provide a starting point for programs to tailor and document possible deviations needed to meet their needs.

Using these standards enables programs to

- define their own secure coding practices that can be used to build software that does not present known vulnerabilities
- train personnel in secure coding practices
- provide a standard that software quality assurance and V&V groups can use to verify that secure code is being developed and to provide metrics to support their efforts

Ultimately, the use of CERT secure coding standards in software acquisition will lead to a reduced number of software defects and software vulnerabilities, resulting in lower maintenance costs for programs because of improved, secure software development practices.

Appendix Mapping of the STIG Guidelines to the CERT Secure Coding Standards

Application Security and Development STIG Guidelines Mapped to CERT C Secure Coding Standard

To help DoD acquisition programs and their contractors develop a secure coding standard, we provide the following two tables that are based on the CERT C Secure Coding Standard and the AS&D STIG. Table 1 identifies the vulnerability severity codes used in the CERT C Secure Coding Standard. In the AS&D STIG, each guideline is given a vulnerability severity code, as defined in Table 1. Table 2 maps the STIG guidelines to the CERT C Secure Coding Standard.

Table 1: Vulnerability Severity Codes

Severity Code	Description
Category I (CAT I)	Vulnerabilities that allow an attacker immediate access into a machine, allow super-user access, or bypass a firewall
Category II (CAT II)	Vulnerabilities that provide information that has a high potential of giving access to an intruder
Category III (CAT III)	Vulnerabilities that provide information that potentially could lead to compromise

Table 2: Mapping of STIG Guidelines to CERT C Secure Coding Standard

STIG Guideline	CDRL Guidance
APP2010.1: CAT II – The Program Manager will ensure an SSP is established describing the technical, administrative, and procedural IA program and policies governing the DoD information system, and identifying all IA personnel and specific IA requirements and objectives. (Page 6) ***Secure Coding Guidance*** None	PMP
APP2010.2: CAT II – The Program Manager will ensure all appointments to required IA roles are established in writing to include assigned duties and appointment criteria, such as training, security clearance, and IT designation. (Page 7) ***Secure Coding Guidance*** None	PMP
APP2020.1: CAT II – The Program Manager will provide an Application Configuration Guide to the application hosting providers. (Page 7) ***Secure Coding Guidance*** None	SEP TEP
APP2020.2: CAT II – The Program Manager will provide a list of all potential hosting enclaves and connection rules and requirements. (Page 7) ***Secure Coding Guidance*** None	SEP TEP
APP2020.3: CAT II – The Program Manager will ensure development systems, build systems, and test systems have a standardized environment and are documented in the Application Configuration Guide. (Page 7) ***Secure Coding Guidance*** None	SEP SDP STP
APP2040.1: CAT II – The Program Manager will ensure a Security Classification Guide exists containing data elements and their classifications if the system contains classified information. (Page 8) ***Secure Coding Guidance*** None	PMP SEP

STIG Guideline	CDRL Guidance
APP2050: CAT II – The Program Manager will ensure the system has been assigned specific MAC and Confidentiality levels. (Page 8) ***Secure Coding Guidance*** None	PMP
APP2060.1: CAT II – The Program Manger will ensure the development team follows a set of coding standards. (Page 9) ***Secure Coding Guidance*** Entire standard	PMP SDP STP
APP2060.2: CAT II – The Program Manger will ensure the development team creates a list of unsafe functions to avoid and document this list in the coding standards. (Page 10) ***Secure Coding Guidance*** • PRE31-C Avoid side effects in arguments to unsafe macros • SIG30-C Call only asynchronous-safe functions within signal handlers • MSC34-C Do not use depreciated or obsolescent functions • ENV04-C Do not call system() if you do not need a command processor • SIG32-C Do not call longjmp() from inside a signal handler • SIG33-C Do not recursively invoke the raise() function • SIG34-C Do not call signal() from within interruptible signal handlers • FIO07-C Prefer fseek() to rewind() • FIO12-C Prefer setvbuf() to setbuf() • ERR07-C Prefer functions that support error checking over equivalent functions that don’t	SDP STP
APP2070.1: CAT III – The Program Manager will ensure any IA or IA-enabled products used by the application are NIAP approved or in the NIAP approval process. (Page 10) ***Secure Coding Guidance*** None	PMP
APP2080.1: CAT II – The Program Manager will ensure COTS IA and IA-enabled products, which are used to protect publicly released information, comply with National Security Agency (NSA)–endorsed Protection Profiles. (Page 11) ***Secure Coding Guidance*** None	PMP
APP2080.2: CAT II – The Program Manager will ensure COTS IA and IA-enabled products which are used to protect sensitive information when the information transits non DoD-owned networks, or the system handling the information is accessible by individuals who are not authorized to access the information on the system, comply with NSA-NIAP approved Protection Profiles. (Page 11) ***Secure Coding Guidance*** None	PMP
APP2080.3: CAT II – The Program Manager will ensure COTS IA and IA-enabled products, which are used to protect classified information when the information transits networks, which are at a lower classification level than the information being transported, comply with NSA-NIAP approved Protection Profiles. (Page 11) ***Secure Coding Guidance*** None	PMP
APP2090.1: CAT II – The Program Manager will obtain DAA acceptance of risk for all public domain, shareware, freeware, and other software products/libraries with both (1) no source code to review, repair, and extend, and (2) limited or no warranty, but are required for mission accomplishment. (Page 12) ***Secure Coding Guidance*** None	PMP SDP
APP2120.1: CAT II – The Program Manager will ensure all levels of program management receive security training regarding the necessity, impact, and benefits of integrating secure development practices into the development lifecycle. (Page 12) ***Secure Coding Guidance*** • The SEI provides a Secure Coding in C and C++ training class • The SEI provides training and guidance for organizations to implement TSP-Secure • The SEI CERT Secure Coding website provides additional information	PMP

STIG Guideline	**CDRL Guidance**
APP2120.2: CAT II – The Program Manager will ensure designers are provided training on secure design principles for the entire SDLC and newly-discovered vulnerability types on at least an annual basis. (Page 13) ***Secure Coding Guidance*** • The SEI provides a Secure Coding in C and C++ training class • The SEI provides training and guidance for organizations to implement TSP-Secure • The SEI CERT Secure Coding website provides additional information	SDP
APP2120.3: CAT II – The Program Manager will ensure developers are provided with training on secure design and coding practices on at least an annual basis. (Page 13) ***Secure Coding Guidance*** • The SEI provides a Secure Coding in C and C++ training class • The SEI provides training and guidance for organizations to implement TSP-Secure • The SEI CERT Secure Coding website provides additional information	PMP SDP
APP2120.4: CAT II – The Program Manager will ensure testers are provided training on at least an annual basis. (Page 13) ***Secure Coding Guidance*** • The SEI provides a Secure Coding in C and C++ training class • The SEI provides training and guidance for organizations to implement TSP-Secure • The SEI CERT Secure Coding website provides additional information	PMP STP
APP2140.1: CAT II – The Program Manager will ensure a security incident response process for the application is established that defines reportable incidents and outlines a standard operating procedure for incident response. (Page 14) ***Secure Coding Guidance*** None	PMP
APP2130.1: CAT II – The Program Manager will ensure users are provided with a means of obtaining updates for the application. (Page 14) ***Secure Coding Guidance*** None	PMP
APP2130.2: CAT II – The Program Manager will ensure a mechanism is in place to notify users of security flaws and to provide users with the availability of patches. (Page 14) ***Secure Coding Guidance*** None	TEP SDP STP
APP2130.3: CAT II – The Program Manager will ensure a comprehensive vulnerability management process, including systematic identification and mitigation of software vulnerabilities is in place. (Page 14) ***Secure Coding Guidance*** None	PMP SEP
APP2135: CAT I – The Program Manager will ensure all products are supported by the vendor or the development team. (Page 14) ***Secure Coding Guidance*** None	PMP SDP
APP2150.1: CAT II – The Program Manager will ensure procedures are implemented to assure physical handling and storage of information is in accordance with the data's sensitivity. (Page 15) ***Secure Coding Guidance*** None	PMP SEP
APP2150.1: CAT II – The Program Manager will ensure procedures are implemented to assure physical handling and storage of information is in accordance with the data's sensitivity. (Page 15) ***Secure Coding Guidance*** None	PMP SEP

STIG Guideline	**CDRL Guidance**
APP2160.1: CAT II – The Program Manager will ensure development systems, build systems, test systems, and all components comply with all appropriate DoD STIGS, NSA guides, and all applicable DoD policies. (Page 16) ***Secure Coding Guidance*** None	SEP SDP STP
APP3010: CAT II – The Designer will create and update the Design Document for each release of the application identifying the following: (Page 17) • All external interfaces (from the threat model) • The nature of information being exchanged • Categories of sensitive information processed or stored and their specific protection plans • The protection mechanisms associated with each interface • User roles required for access control • Access privileges assigned to each role • Unique application security requirements • Categories of sensitive information processed or stored and specific protection plans (e.g., Privacy Act, Health Insurance Portability and Accountability Act (HIPAA), etc.) • Restoration priority of subsystems, processes, or information ***Secure Coding Guidance*** None	SEP SDP
APP2020.4: CAT II – The Designer will ensure known security assumptions, implications, system-level protections, best practices, and required permissions are documented in the Application Configuration Guide. (Page 18) ***Secure Coding Guidance*** None	PMP SEP
APP2020.5: CAT II – The Designer will ensure deployment configuration settings are documented in the Application Configuration Guide. (Page 18) ***Secure Coding Guidance*** None	PMP SEP
APP3020.1: CAT II – The Designer will ensure threat models are documented and reviewed for each application release and updated as required by design and functionality changes or new threats are discovered. (Page 18) ***Secure Coding Guidance*** None	SEP SDP
APP3020.2 CAT II – The Designer will identify potential mitigations to identified threats. (Page 18) ***Secure Coding Guidance*** None	SEP RMP
APP3020.3: CAT II – The Designer will ensure appropriate mitigations are implemented to threats based on their risk analysis. (Page 18) ***Secure Coding Guidance*** None	SEP RMP
APP2060.3: CAT II – The Designer will follow the established coding standards established for the project. (Page 23) ***Secure Coding Guidance*** Entire standard	SDP STP

STIG Guideline	CDRL Guidance
APP2060.4: CAT II – The Designer will not use unsafe functions documented in the project coding standards. (Page 23) ***Secure Coding Guidance*** • PRE31-C Avoid side effects in arguments to unsafe macros • SIG30-C Call only asynchronous-safe functions within signal handlers • MSC34-C Do not use depreciated or obsolescent functions • ENV04-C Do not call system() if you do not need a command processor • SIG32-C Do not call longjmp() from inside a signal handler • SIG33-C Do not recursively invoke the raise() function • SIG34-C Do not call signal() from within interruptible signal handlers • FIO07-C Prefer fseek() to rewind() • FIO12-C Prefer setvbuf() to setbuf() • ERR07-C Prefer functions that support error checking over equivalent functions that don't	SDP STP
APP2070.2: CAT III – The Designer will ensure any IA or IA-enabled products used by the application are NIAP-approved or in the NIAP approval process. (Page 23) ***Secure Coding Guidance*** None	PMP
APP2090.2: CAT II – The Designer will document for DAA approval all public domain, shareware, freeware, and other software products/libraries with both (1) no source code to review, repair, and extend, and (2) limited or no warranty, but are required for mission accomplishment. (Page 24) ***Secure Coding Guidance*** None	PMP SDP
APP2100.2: CAT II – The Designer will ensure the application design complies with the DoD Ports and Protocols guidance. (Page 24) ***Secure Coding Guidance*** None	PMP SEP
APP2110.2: CAT II – The Designer will ensure the application is registered with the DoD Ports and Protocols database. (Page 24) ***Secure Coding Guidance*** None	PMP SEP
APP3050: CAT II – The Designer will ensure the application does not contain source code that is never invoked during operation, except for software components and libraries from approved third-party products, which may include un-invoked code. (Page 25) ***Secure Coding Guidance*** • MSC-7-C Detect and remove dead code • MSC12-C Detect and remove code that has no effect • MSC13-C Detect and remove unused values	SDP
APP3060: CAT II – The Designer will ensure the application does not store configuration and control files in the same directory as user data. (Page 25) ***Secure Coding Guidance*** • FIO15-C Ensure that file operations are performed in a secure directory • FIO43-C Do not create temporary files in shared directories • MSC18-C Be careful while handling sensitive data, such as passwords, in program code	SDP
APP3070: CAT II – The Designer will ensure the user interface services are physically or logically separated from data storage and management services. (Page 25) ***Secure Coding Guidance*** • MEM06-C Ensure that sensitive data is not written out to disk	SDP
APP3080: CAT II – The Designer will ensure the application does not contain invalid URL or path references. (Page 25) ***Secure Coding Guidance*** • FIO02-C Canonicalize path names originating from untrusted sources	SDP STP

STIG Guideline	CDRL Guidance
APP3100: CAT II – The Designer will ensure the application removes temporary storage of files and cookies when the application is terminated. (Page 25) ***Secure Coding Guidance*** • MEM03-C Clear sensitive information stored in reusable resources • MEM06-C Ensure that sensitive data is not written out to disk • MSC18-C Be careful while handling sensitive data, such as passwords, in program code	SDP STP
APP3110: CAT II – The Designer will ensure the application installs with unnecessary functionality disabled by default. (Page 25) ***Secure Coding Guidance*** None	SEP
APP3120: CAT II – The Designer will ensure the application is not subject to error handling vulnerabilities. (Page 26) ***Secure Coding Guidance*** • FLP03-C Detect and handle floating point errors • FLP32-C Prevent or detect domain and range errors in math functions • MEM32-C Detect and handle memory allocation errors • FIO04-C Detect and handle input and output errors • FIO07-C Prefer fseek() to rewind() • FIO12-C Prefer setvbuf() to setbuf() • FIO33-C Detect and handle input output errors resulting in undefined behavior • ERR00-C Adopt and implement a consistent and comprehensive error-handling policy • ERR01-C Use ferror() rather than errno to check for FILE stream errors • ERR02-C Avoid in-band error indicators • ERR03-C Use runtime-constraint handlers when calling functions defined by TR2473-1 • ERR04-C Choose an appropriate termination strategy • ERR05-C Application-independent code should provide error detection without dictating error handling • ERR06-C Understand the termination behavior of assert() and abort() • ERR07-C Prefer functions that support error checking over equivalent functions that don't • ERR30-C Set errno to zero before calling a library function known to set errno, and check errno only after the function returns a value indicating failure • ERR31-C Don't redefine errno • ERR32-C Do not rely on indeterminate values of errno • ERR33-C Detect and handle errors • API04-C Provide a consistent and usable error checking mechanism • DCL09-C Declare functions that return errno with a return type of errno_t • MSC31-C Ensure that return values are compared against the proper type	SEP TEP SDP STP
APP3130: CAT I – The Designer will ensure the application follows the secure failure design principle. (Page 27) ***Secure Coding Guidance*** • ERR00-C Adopt and implement a consistent and comprehensive error-handling policy • ERR03-C Use runtime-constraint handlers when calling functions defined by TR24731-1 • ERR04-C Choose an appropriate termination strategy • ERR05-C Application-independent code should provide error detection without dictating error handling • ERR06-C Understand the termination behavior of assert() and abort() • ERR33-C Detect and handle errors	SEP TEP SDP STP
APP3140: CAT II – The Designer will ensure application initialization, shutdown, and aborts are designed to keep the application in a secure state. (Page 27) ***Secure Coding Guidance*** • ERR00-C Adopt and implement a consistent and comprehensive error-handling policy • ERR03-C Use runtime-constraint handlers when calling functions defined by TR24731-1 • ERR04-C Choose an appropriate termination strategy • ERR06-C Understand the termination behavior of assert() and abort()	SEP TEP SDP STP

STIG Guideline	CDRL Guidance
APP3150.1: CAT II – The Designer will ensure the application uses FIPS 140-2 validated cryptographic modules if the application implements encryption, key exchange, digital signature, and hash functionality. (Page 27) ***Secure Coding Guidance*** None	PMP SEP
APP3150.2: CAT II – The Designer will ensure the application uses a FIPS 140-2 validated random number generator to support cryptographic functions. (Page 28) ***Secure Coding Guidance*** None	PMP SEP
APP3170: CAT II – The Designer will ensure the application uses encryption to implement key exchange and authenticate end-points prior to establishing a communication channel for key exchange. (Page 28) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3180: CAT II – The Designer will ensure private keys are accessible only to administrative users. (Page 29) ***Secure Coding Guidance*** None	SEP TEP STP
APP3190: CAT II – The Designer will ensure the application does not connect to a database using administrative credentials or other privileged database accounts. (Page 29) ***Secure Coding Guidance*** None	SEP TEP STP
APP3200: CAT III – The Designer will ensure transaction-based applications implement transaction rollback and transaction journaling. (Page 29) ***Secure Coding Guidance*** None	SEP TEP STP
APP3210.1: CAT II – The Designer will ensure NIST-certified cryptography is used to protect stored sensitive information if required by the information owner. (Page 29) ***Secure Coding Guidance*** None	SEP TEP STP
APP3210.2: CAT II – The Designer will ensure NIST-certified cryptography is used to store classified non-Sources and Methods Intelligence (SAMI) information if required by the information owner. (Page 29) ***Secure Coding Guidance*** None	SEP TEP STP
APP3210.3: CAT II – The Designer will ensure a classified enclave containing SAMI data is encrypted with NSA-approved cryptography. (Page 29) ***Secure Coding Guidance*** None	SEP TEP STP
APP3220.1: CAT II – The Designer will ensure sensitive data held in memory is cryptographically protected when not in use if required by the information owner. (Page 30) ***Secure Coding Guidance*** None	SEP TEP STP
APP3220.2: CAT II – The Designer will ensure classified data held in memory is cryptographically protected when not in use. (Page 30) ***Secure Coding Guidance*** • MSC18-C Be careful while handling sensitive data, such as passwords, in program code	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3230.1: CAT II – The Designer will ensure the application properly clears or overwrites all memory blocks used to process sensitive data if required by the information owner. (Page 30) ***Secure Coding Guidance*** • MEM03-C Clear sensitive information stored in reusable resources • MSC18-C Be careful while handling sensitive data, such as passwords, in program code	SEP TEP SDP STP
APP3230.2: CAT II – The Designer will ensure the application properly clears or overwrites all memory blocks used to classified data. (Page 30) ***Secure Coding Guidance*** • MEM03-C Clear sensitive information stored in reusable resources • MSC18-C Be careful while handling sensitive data, such as passwords, in program code	SEP TEP SDP STP
APP3240: CAT II – The Designer will ensure all access authorizations to data are revoked prior to initial assignment, allocation or reallocation to an unused state. (Page 30) ***Secure Coding Guidance*** • POS02-C Follow the principle of least privilege	SEP TEP STP
APP3250.1: CAT I – The Designer will ensure unclassified, sensitive data transmitted through a commercial or wireless network is protected using NIST-certified cryptography. (Page 31) ***Secure Coding Guidance*** None	SEP TEP STP
APP3250.2: CAT I – The Designer will ensure classified data, transmitted through a network that is cleared to a lower level than the data being transmitted, is separately protected using NSA-approved cryptography. (Page 31) ***Secure Coding Guidance*** None	SEP TEP STP
APP3250.3: CAT II – The Designer will ensure information in transit through a network at the same classification level, but which must be separated for need-to-know reasons, is protected minimally with NIST-certified cryptography. (Page 31) ***Secure Coding Guidance*** None	SEP TEP STP
APP3250.4: CAT II – The Designer will ensure SAMI information in transit through a network at the same classification level is protected with NSA-approved cryptography. (Page 31) ***Secure Coding Guidance*** None	SEP TEP STP
APP3260: CAT II – The Designer will ensure the application uses mechanisms assuring the integrity of all transmitted information (including labels and security parameters). (Page 31) ***Secure Coding Guidance*** • FIO09-C. Be careful with binary data when transferring data across systems	SEP TEP SDP STP
APP3270: CAT I – The Designer will ensure the application has the capability to mark sensitive/classified output when required. (Page 31) ***Secure Coding Guidance*** None	SEP TEP STP
APP3280.1: CAT II – The Designer will ensure applications requiring user authentication are PK-enabled. (Page 37) ***Secure Coding Guidance*** None	SEP TEP STP
APP3280.2: CAT II – The Designer will ensure applications requiring user authentication are designed and implemented to support hardware tokens (e.g., CAC for NIPRNet). (Page 37) ***Secure Coding Guidance*** None	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3290.1: CAT II – The Designer will ensure PK-enabled applications are designed and implemented to use approved credentials authorized under the DoD PKI program. (Page 37) ***Secure Coding Guidance*** None	SEP TEP STP
APP3300: CAT II – The Designer will ensure applications requiring server authentication are PK-enabled. (Page 38) ***Secure Coding Guidance*** None	SEP TEP STP
APP3305: CAT I – The Designer will ensure the application using PKI validates certificates for expiration, confirms origin is from a DoD-authorized CA, and verify certificate has not been revoked by CRL or OCSP, and CRL cache (if used) is updated at least daily. (Page 38) ***Secure Coding Guidance*** None	SEP TEP STP
APP3310: CAT I – The Designer will ensure the application does not display account passwords as clear text. (Page 40) ***Secure Coding Guidance*** • MSC18-C Be careful while handling sensitive data, such as passwords, in program code	SEP TEP SDP STP
APP3320.1: CAT II – The Designer will ensure the application has the capability to require account passwords having a minimum of 15 alphanumeric characters in length. (Page 41) ***Secure Coding Guidance*** None	SEP TEP STP
APP3320.2: CAT II – The Designer will ensure the application has the capability to require account passwords contain a mix of upper case letters, lower case letters, numbers, and special characters. (Page 41) ***Secure Coding Guidance*** None	SEP TEP STP
APP3320.3: CAT II – The Designer will ensure the application has the capability to require account passwords be changed every 60 days or more frequently. (Page 41) ***Secure Coding Guidance*** None	SEP TEP STP
APP3320.4: CAT II – The Designer will ensure passwords do not contain personal information such as names, telephone numbers, account names, birthdates, or dictionary words. (Page 41) ***Secure Coding Guidance*** None	SEP TEP STP
APP3320.5: CAT II – The Designer will ensure the application has the capability to limit reuse of account passwords within the last 10 password changes. (Page 41) ***Secure Coding Guidance*** None	SEP TEP STP
APP3320.6: CAT II – The Designer will ensure the application has the capability to limit user changes to their account passwords once every 24 hours with the exception of privileged or administrative users. (Page 41) ***Secure Coding Guidance*** None	SEP TEP STP
APP3320.7: CAT II – The Designer will ensure the application has the capability to require new account passwords differ from the previous password by at least four characters when a password is changed. (Page 41) ***Secure Coding Guidance*** None	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3330: CAT I – The Designer will ensure the application transmits account passwords in a approved encrypted format. (Page 41) ***Secure Coding Guidance*** None	SEP TEP STP
APP3340: CAT I – The Designer will ensure the application stores account passwords in an approved encrypted format. (Page 42) ***Secure Coding Guidance*** • MSC18-C Be careful while handling sensitive data, such as passwords, in program code	SEP TEP SDP STP
APP3350: CAT I – The Designer will ensure the application does not contain embedded authentication data. (Page 42) ***Secure Coding Guidance*** None	SEP TEP STP
APP3360: CAT II – The Designer will ensure the application protects access to authentication data by restricting access to authorized users and services. (Page 43) ***Secure Coding Guidance*** • FIO06-C Create files with appropriate access permissions • POS02-C Follow the principle of least privilege	SEP TEP STP
APP3370: CAT II – The Designer will ensure the application installs with unnecessary accounts disabled or deleted by default. (Page 43) ***Secure Coding Guidance*** None	SEP TEP STP
APP3380: CAT II – The Designer will ensure the application prevents the creation of duplicate accounts. (Page 43) ***Secure Coding Guidance*** None	SEP TEP STP
APP3390: CAT I – The Designer will ensure users' accounts are locked after three consecutive unsuccessful logon attempts within one hour. (Page 43) ***Secure Coding Guidance*** None	SEP TEP STP
APP3400: CAT II – The Designer will ensure locked users' accounts can only be unlocked by the application administrator. (Page 43) ***Secure Coding Guidance*** None	SEP TEP STP
APP3405: CAT I – The Designer will ensure the application supports detection and/or prevention of communication session hijacking. (Page 44) ***Secure Coding Guidance*** None	SEP TEP STP
APP3410.1: CAT II – The Designer will ensure the application provides a capability to limit the number of logon sessions per user. (Page 44) ***Secure Coding Guidance*** None	SEP TEP STP
APP3410.2: CAT II – The Designer will ensure the application provides a capability to limit the total number of logon sessions for the application. (Page 44) ***Secure Coding Guidance*** None	SEP TEP STP
APP3415: CAT II – The Designer will ensure the application provides a capability to automatically terminate a session and logout after a system defined session idle time limit is exceeded. (Page 44) ***Secure Coding Guidance*** None	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3420: CAT II – The Designer will ensure the application provides a capability to terminate a session and logout. (Page 44) ***Secure Coding Guidance*** None	SEP TEP STP
APP3430: CAT I – The Designer will ensure the application removes authentication credentials on client computers after a session terminates. (Page 44) ***Secure Coding Guidance*** None	SEP TEP STP
APP3440: CAT II – The Designer will ensure the application is capable of displaying a customizable click-through banner at logon which prevents further activity on the information system unless and until the user executes a positive action to manifest agreement by clicking on a box indicating "OK". (Page 45) ***Secure Coding Guidance*** None	SEP TEP STP
APP3450.1: CAT II – The Designer will ensure application resources are protected with permission sets which allow only an application administrator to modify application resource configuration files. (Page 46) ***Secure Coding Guidance*** • FIO06-C Create files with appropriate access permissions • FIO15-C Ensure that file operations are performed in a secure directory • POS02-C Follow the principle of least privilege	SEP TEP SDP STP
APP3460: CAT I – The Designer will ensure the application does not rely solely on a resource name to control access to a resource. (Page 46) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3470.1: CAT II – The Designer will ensure the application is organized by functionality and roles to support the assignment of specific roles to specific application functions. (Page 47) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3480.1: CAT I – The Designer will ensure access control mechanisms exist to ensure data is accessed and changed only by authorized personnel. (Page 47) ***Secure Coding Guidance*** • FIO06-C Create files with appropriate access permissions • MSC18-C Be careful while handling sensitive data, such as passwords, in program code • POS02-C Follow the principle of least privilege	SEP TEP SDP STP
AP3480.2: CAT II – The Designer will ensure the access procedures enforce the principles of separation of duties and "least privilege." (Page 47) ***Secure Coding Guidance*** • FIO06-C Create files with appropriate access permissions • POS02-C Follow the principle of least privilege • POS36-C Observe correct revocation order while relinquishing privileges • POS37-C Ensure that privilege relinquishment is successful	SEP TEP SDP STP
APP3500: CAT II – The Designer will ensure the application executes with no more privileges than necessary for proper operation. (Page 47) ***Secure Coding Guidance*** • FIO06-C Create files with appropriate access permissions • POS02-C Follow the principle of least privilege • POS36-C Observe correct revocation order while relinquishing privileges • POS37-C Ensure that privilege relinquishment is successful	SEP TEP SDP STP

STIG Guideline	CDRL Guidance
APP3510: CAT I – The Designer will ensure the application validates all input. (Page 48) ***Secure Coding Guidance*** • FIO04-C Detect and handle input and output errors • INT04-C Enforce limits on integer values originating from untrusted sources • INT08-C Verify that all integer values are in range • FLP04-C Check floating point inputs for exceptional values • FLP32-C Eliminated Guideline: This guideline has been labeled void and designated for future elimination from the C++ Secure Coding Practices. It has not been erased yet in case it contains information that might still be useful. • ARR30-C Eliminated Practice: This practice has been labeled void and designated for future elimination from the C Secure Coding Standard: It has been superseded by "ARR30-C. Do not form or use out of bounds pointers or array subscripts." The practice has not been erased in case it contains information that might be useful in the future. • ARR32-C Ensure size arguments for variable length arrays are in a valid range • API00-C Functions should validate their parameters	SEP TEP SDP STP
APP3530: CAT II – The Designer will ensure the web application assigns the character set on all web pages. (Page 48) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3540.1: CAT I – The Designer will ensure the application is not vulnerable to SQL injection. (Page 49) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3540.2: CAT II – The Designer will ensure the application uses prepared or parameterized statements. (Page 49) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3540.3: CAT II – The Designer will ensure the application does not use concatenation or replacement to build SQL queries. (Page 49) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3540.4: CAT II – The Designer will ensure the application does not directly access the tables in a database. (Page 49) ***Secure Coding Guidance*** None	SEP TEP SDP STP

STIG Guideline	CDRL Guidance
APP3550: CAT I – The Designer will ensure the application is not vulnerable to integer arithmetic issues. (Page 50) ***Secure Coding Guidance*** • INT00-C Understand the data model used by your implementation(s) • INT01-C Use rsize_t or size_t for all integer values representing the size of an object • INT02-C Understand integer conversion rules • INT04-C Enforce limits on integer values originating from untrusted sources • INT05-C Do not use input functions to convert character data if they cannot handle all possible inputs • INT07-C Use only explicitly signed or unsigned char type for numeric values • INT08-C Verify that all integer values are in range • INT10-C Do not assume a positive remainder when using the % operator • INT12-C Do not make assumptions about the type of a plain int bit-field when used in an expression • INT13-C Use bitwise operators only on unsigned operands • INT14-C Avoid performing bitwise and arithmetic operations on the same data • INT15-C Use intmax_t or uintmax_t for formatted IO on programmer-defined integer types • INT16-C Do not make assumptions about representation of signed integers • INT17-C Define integer constants in an implementation-independent manner • INT30-C ensure that unsigned integer operations do not wrap • INT31-C Ensure that integer conversions do not result in lost or misinterpreted data • INT32-C Ensure that operations on signed integers do not result in overflow • INT33-C Ensure that division and modulo operations do not result in divide-by-zero errors • INT34-C Do not shift a negative number of bits or more bits than exist in the operand • INT35-C Evaluate integer expressions in a larger size before comparing or assigning to that size	SEP TEP SDP STP
APP3560: CAT I – The Designer will ensure the application does not contain format string vulnerabilities. (Page 51) ***Secure Coding Guidance*** • STR02-C Sanitize data passed to complex subsystems • STR03-C Do not inadvertently truncate a null-terminated byte string • STR04-C Use plain char for characters in the basic character set • STR05-C Use pointers to const when referring to string literals • STR06-C Do not assume that strtok() leaves the parse string unchanged • STR07-C Use TR 24731 for remediation of existing string manipulation code • STR08-C Use managed strings for development of new string manipulation code • STR10-C Do not concatenate different type of string literals • STR30-C Do not attempt to modify string literals • STR31-C Guarantee that storage for strings has sufficient space for character data and the null terminator • STR32-C Null-terminate byte strings as required • STR33-C Size wide character strings correctly • STR35-C Do not copy data from an unbounded source to a fixed-length array • STR36-C Do not specify the bound of a character array initialized with a string literal • STR38-C Do not use wide-char functions on narrow-char strings and vice versa • FIO00-C Be careful using functions that use file names for identification • FIO30-C Exclude user input from format strings	SEP TEP SDP STP
APP3570: CAT I – The Designer will ensure the application does not allow command injection. (Page 51) ***Secure Coding Guidance*** • ENV03-C Sanitize the environment when invoking external programs • ENV04-C Do not call system() if you do not need a command processor	SEP TEP SDP STP
APP3580: CAT I – The Designer will ensure the application does not have XSS vulnerabilities. (Page 52) ***Secure Coding Guidance*** None	SEP TEP SDP STP

STIG Guideline	CDRL Guidance
APP3585: CAT II – The Designer will ensure the application does not have CSRF vulnerabilities. (Page 52) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3590.1: CAT I – The Designer will ensure the application does not have buffer overflows. (Page 53) ***Secure Coding Guidance*** • ARR00-C Understand how arrays work • ARR01-C Do not apply the sizeof operator to a pointer when taking the size of an array • ARR02-C Explicitly specify array bounds, even if implicitly defined by an initializer • ARR30-C Do not form or use out of bounds pointers or array subscripts • ARR32-C Ensure size arguments for variable length arrays are in a valid range • ARR33-C Guarantee that copies are made into storage of sufficient size • ARR34-C Ensure that array types in expressions are compatible • ARR36-C Do not subtract or compare tow pointers that do not refer to the same array • ARR37-C Do not add or subtract an integer to a pointer to a non-array object • STR01-C Adopt and implement a consistent plan for managing strings • STR31-C Guarantee that storage for strings has sufficient space for character data and the null terminator • STR35-C Do not copy data from an unbounded source to a fixed-length array • STR36-C Do not specify the bound of a character array initialized with a string literal • STR37-C Arguments to character handling functions must be representable as an unsigned character	SEP TEP SDP STP
APP3590.2: CAT II – The Designer will ensure the application does not use functions known to be vulnerable to buffer overflows. (Page 53) ***Secure Coding Guidance*** • MSC34-C Do not use deprecated or obsolescent functions • STR07-C Use TR 24731 for remediation of existing string manipulation code	SEP TEP SDP STP
APP3590.3: CAT II – The Designer will ensure the application does not use signed values for memory allocation where permitted by the programming language. (Page 53) ***Secure Coding Guidance*** Not addressed	SEP TEP SDP STP
APP3600: CAT II – The Designer will ensure the application has no canonical representation vulnerabilities. (Page 54) ***Secure Coding Guidance*** • FIO02-C Canonicalize path names originating from untrusted sources	SEP TEP SDP STP
APP3610: CAT I – The Designer will ensure the application does not use hidden fields to control user access privileges or as a part of a security mechanism. (Page 55) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3620: CAT II – The Designer will ensure the application does not disclose unnecessary information to users. (Page 56) ***Secure Coding Guidance*** • ERR00-C Adopt and implement a consistent and comprehensive error-handling policy • ERR04-C Choose an appropriate termination strategy • MSC18-C Be careful while handling sensitive data, such as passwords, in program code	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3630.1: CAT II – The Designer will ensure the application is not vulnerable to race conditions. (Page 56) ***Secure Coding Guidance*** • POS38-C Beware of race conditions when using fork and file descriptors • POS44-C Do not use signals to terminate threads • POS47-C Do not use threads that can be cancelled asynchronously • CON00-C Avoid race conditions with multiple threads • CON01-C Acquire and release synchronization primitives in the same module, at the same level of abstraction • CON31-C Do not unlock or destroy another thread's mutex • CON32-C When data must be accessed by multiple threads, provide a mutex and guarantee no adjacent data is also accessed • CON33-C Avoid race conditions when using library functions • CON34-C Declare objects shared between threads with appropriate storage durations	SEP TEP SDP STP
APP3630.2: CAT III – The Designer will ensure the application does not use global variables when local variables could be used. (Page 57) ***Secure Coding Guidance*** • DCL19-C Minimize the scope of variables and functions	SEP TEP SDP STP
APP3630.3: CAT II – The Designer will ensure a multi-threaded application uses thread safe functions when threads are accessing the same object or data. (Page 57) ***Secure Coding Guidance*** • POS38-C Beware of race conditions when using fork and file descriptors • POS44-C Do not use signals to terminate threads • POS47-C Do not use threads that can be cancelled asynchronously • CON33-C Avoid race conditions when using library functions	SEP TEP SDP STP
APP3630.4: CAT II – The Designer will ensure global resources are locked before being accessed by the application. (Page 57) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3640: CAT II – The Designer will ensure the application supports the creation of transaction logs for access and changes to the data. (Page 57) ***Secure Coding Guidance*** None	SEP TEP STP
APP3650: CAT III – The Designer will ensure the application has a capability to notify an administrator when audit logs are nearing capacity as specified in the system documentation. (Page 57) ***Secure Coding Guidance*** None	SEP TEP STP
APP3660: CAT III – The Designer will ensure the application has a capability to notify the user on login of date and time of the user's last unsuccessful logon, IP address of the user's last unsuccessful logon, date and time of the user's last successful logon, IP address of the user's last successful logon, and number of unsuccessful logon attempts since the last successful logon. (Page 58) ***Secure Coding Guidance*** None	SEP TEP STP
APP3670: CAT II – The Designer will ensure the application has a capability to display the user's time and date of the last change in data content. (Page 58) ***Secure Coding Guidance*** None	SEP TEP STP
APP3680.1: CAT II – The Designer will ensure the application design includes audits on all access to need-to-know information. (Page 58) ***Secure Coding Guidance*** None	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3680.2: CAT II – The Designer will ensure the application logs all failed access attempts to need-to-know information. (Page 58) ***Secure Coding Guidance*** None	SEP TEP STP
APP3680.3: CAT II – The Designer will ensure the application's publicly releasable data audit records include: (Page 59) • Userid • Successful and unsuccessful attempts to access security files • Data and time of the event • Type of event ***Secure Coding Guidance*** None	SEP TEP STP
APP3680.4: CAT II – The Designer will ensure the application's sensitive data audit records include: (Page 59) • Userid • Successful and unsuccessful attempts to access security files • Data and time of the event • Type of event • Success or failure of event • Successful and unsuccessful logons • Denial of access resulting from excessive number of logon attempts • Blocking or blacklisting a userid, terminal or access port and the reason for the action • Activities that might modify, bypass, or negate safeguards controlled by the system ***Secure Coding Guidance*** None	SEP TEP STP
APP3680.5: CAT II – The Designer will ensure the application's classified data audit records include: (Page 59) • Userid • Successful and unsuccessful attempts to access security files • Data and time of the event • Type of event • Success or failure of event • Successful and unsuccessful logons • Denial of access resulting from excessive number of logon attempts • Blocking or blacklisting a userid, terminal or access port and the reason for the action • Activities that might modify, bypass, or negate safeguards controlled by the system • Data required to audit the possible use of covert channel mechanisms • Privileged activities and other system-level access • Starting and ending time for access to the system • Security relevant actions associated with periods of activity where security labels or categories of information are processed or changed ***Secure Coding Guidance*** None	SEP TEP STP
APP3680.6: CAT II – The Designer will ensure the application creates an audit trail for addition, deletion, or change of the confidentiality or integrity labels as designated by the information owner. (Page 60) ***Secure Coding Guidance*** None	SEP TEP STP
APP3690.1: CAT II – The Designer will ensure the audit trail is readable only by the application and auditors. (Page 60) ***Secure Coding Guidance*** None	SEP TEP STP
APP3690.2: CAT II – The Designer will ensure the audit trail is protected against modification or deletion except by the application and auditors. (Page 60) ***Secure Coding Guidance*** None	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3700.1: CAT II – The Designer will ensure unsigned Category 1A mobile code is not used in the application. (Page 61) ***Secure Coding Guidance:*** None	SEP TEP STP
APP3700.2: CAT II – The Designer will ensure Category 1A mobile code used in an application is signed with a DoD-approved code-signing certificate. (Page 61) ***Secure Coding Guidance*** None	SEP TEP STP
APP3700.3: CAT II – The Designer will ensure signed Category 1A mobile code used in an application is obtained from a trusted source and is designated as trusted. (Page 61) ***Secure Coding Guidance*** None	SEP TEP STP
APP3710.1: CAT II – The Designer will ensure signed Category 1A mobile code signature is validated before executing. (Page 61) ***Secure Coding Guidance*** None	SEP TEP STP
APP3700.4: CAT II – The Designer will ensure Category 1X mobile code is not used in applications. (Page 61) ***Secure Coding Guidance*** None	SEP TEP STP
APP3720: CAT II – The Designer will ensure unsigned Category 2 mobile code executing in a constrained environment has no access to local system and network resources. (Page 62) ***Secure Coding Guidance*** None	SEP TEP STP
APP3700.5: CAT II – The Designer will ensure signed Category 2 mobile code used in an application is signed with a DoD-approved code-signing certificate. (Page 62) ***Secure Coding Guidance*** None	SEP TEP STP
APP3700.6: CAT II – The Designer will ensure Category 2 mobile code not executing in a constrained execution environment is obtained from a trusted source over an assured channel using at least one of the following measures: (Page 62) 1. The mobile code was digitally signed with a code-signing certificate that was designated as trusted by the recipient's component. 2. The mobile code was downloaded over an SSL connection from a trusted SSL web server using a DoD or trusted commercial SSL server certificate. 3. The mobile code was downloaded over a TLS connection from a trusted TLS web server using a DoD or trusted commercial TLS server certificate. 4. The mobile code was downloaded from a trusted web server over an encrypted IPSec connection that establishes mutual authentication using a DoD or trusted commercial certificate. ***Secure Coding Guidance*** None	SEP TEP STP
APP3710.2: CAT II – The Designer will ensure the signed Category 2 mobile code signature is validated before executing. (Page 63) ***Secure Coding Guidance*** None	SEP TEP STP
APP3730: CAT II – The Designer will ensure uncategorized or emerging mobile code is not used in applications. (Page 63) ***Secure Coding Guidance*** None	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3740: CAT II – The Designer will ensure the application only embeds mobile code in e-mail that does not execute automatically when the user opens the e-mail body or attachment. (Page 64) ***Secure Coding Guidance*** None	SEP TEP STP
APP3750: CAT II – The Designer will ensure development of new mobile code includes measures to mitigate the risks identified. (Page 64) ***Secure Coding Guidance*** None	SEP TEP STP
APP3760: CAT II – The Designer will ensure web services are designed and implemented to recognize and react to the attack patterns associated with application-level DoS. (Page 65) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3770: CAT II – The Designer will ensure the web service design includes redundancy of critical functions. (Page 65) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3780: CAT II – The Designer will ensure web service design of critical functions is implemented using different algorithms to prevent similar attacks from a complete application level DoS. (Page 65) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3790: CAT II – The Designer will ensure web services are designed to prioritize requests to increase availability of the system. (Page 66) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3800: CAT II – The Designer will ensure execution flow diagrams are created and used to mitigate deadlock and recursion issues. (Page 66) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3810: CAT I – The Designer will ensure the application is not vulnerable to XML injection. (Page 66) Relevant Secure Coding Guidance: No relevance to CERT secure coding standards ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3820: CAT I – The Designer will ensure web services provide a mechanism for detecting resubmitted SOAP messages. (Page 69) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP3830.1: CAT II – The Designer will ensure digital signatures exist on UDDI registry entries to verify the publisher. (Page 70) ***Secure Coding Guidance*** None	SEP TEP STP
APP3840.1: CAT II – The Designer will ensure UDDI versions are used supporting digital signatures of registry entries. (Page 70) ***Secure Coding Guidance*** None	SEP TEP STP
APP3850.1: CAT II – The Designer will ensure UDDI publishing is restricted to authenticated users. (Page 70) ***Secure Coding Guidance*** None	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3860: CAT II – The Designer will ensure SOAP messages requiring integrity sign the following elements: (Page 71) • Message ID • Service request • Timestamp • SAML Assertion ***Secure Coding Guidance*** None	SEP TEP STP
APP3870: CAT I – The Designer will ensure when using WS-Security messages use timestamps with creation and expiration times. (Page 72) ***Secure Coding Guidance*** None	SEP TEP STP
APP3880: CAT I – The Designer will ensure validity periods are verified on all messages using WS-Security or SAML assertions. (Page 72) ***Secure Coding Guidance*** None	SEP TEP STP
APP3890: CAT II – The Designer will ensure each unique asserting party provides unique assertion ID references for each SAML assertion. (Page 75) ***Secure Coding Guidance*** None	SEP TEP STP
APP3900: CAT II – The Designer shall ensure encrypted assertions or equivalent confidentiality when assertion data is passed through an intermediary and confidentiality of the assertion data is required to pass through the intermediary. (Page 76) ***Secure Coding Guidance*** None	SEP TEP STP
APP3910: CAT I – The Designer shall use the NotBefore and NotOnOrAfter when using the SubjectConfirmation element in a SAML assertion. (Page 77) ***Secure Coding Guidance*** None	SEP TEP STP
APP3920: CAT I – The Designer shall use the both the NotBefore and NotOnOrAfter elements or OneTimeUse element when using the Conditions element in a SAML Assertion. (Page 79) ***Secure Coding Guidance*** None	SEP TEP STP
APP3930: CAT II – The Designer shall ensure if a <OneTimeUse> element is used in an assertion, there is only one used in <Conditions> element of an assertion. (Page 78) ***Secure Coding Guidance*** None	SEP TEP STP
APP3940: CAT II – The Designer will ensure the asserting party uses FIPS-approved random numbers in the generation of SessionIndex in the SAML Element <AuthnStatement>. (Page 79) ***Secure Coding Guidance*** None	SEP TEP STP
APP3950: CAT II – The Designer shall ensure messages are encrypted when the SessionIndex is tied to privacy data. (Page 79) ***Secure Coding Guidance*** None	SEP TEP STP
APP3960: CAT II – The Designer will ensure the application is compliant with all DISR IPv6 profiles. (Page 81) Relevant Secure Coding Guidance: No relevance to CERT secure coding standards ***Secure Coding Guidance*** None	SEP TEP STP

STIG Guideline	CDRL Guidance
APP3970: CAT II – The Designer will ensure supporting application services and interfaces have been designed or upgraded for IPv6 transport. (Page 82) ***Secure Coding Guidance*** None	SEP TEP STP
APP3980: CAT II – The Designer will ensure the application is compliant with IPv6 multicast addressing and features an IPv6 network configuration options as defined in RFC 4038. (Page 82) ***Secure Coding Guidance*** None	SEP TEP STP
APP3990: CAT II – The Designer will ensure the application is compliant with the IPv6 addressing scheme as defined in with RFC 1884. (Page 82) ***Secure Coding Guidance*** None	SEP TEP STP
APP4010: CAT III – The Release Manager will ensure the access privileges to the configuration management (CM) repository are reviewed every 3 months. (Page 83) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP4030.1: CAT II – The Release Manager will develop an SCM plan describing the configuration control and change management process of objects developed and the roles and responsibilities of the organization. (Page 83) ***Secure Coding Guidance*** None	PMP
APP4030.2: CAT III – The Release Manager will ensure the SCM plan identifies all objects created during the development process subject to configuration control. (Page 83) ***Secure Coding Guidance*** None	PMP SEP TEP
APP4030.3: CAT II – The Release Manager will ensure the SCM plan maintains procedures for identifying individual application components, as well as, entire application releases during all phases of the software development lifecycle. (Page 83) ***Secure Coding Guidance*** None	PMP
APP4030.4: CAT III – The Release Manager will ensure the SCM plan identifies and tracks all actions and changes resulting from a change request from initiation to release. (Page 83) ***Secure Coding Guidance*** None	PMP
APP4030.5: CAT III – The Release Manager will ensure the SCM plan contains procedures to identify, document, review, and authorize any change requests to the application. (Page 83) ***Secure Coding Guidance*** None	PMP
APP4030.6: CAT III – The Release Manager will ensure the SCM plan defines the responsibilities, the actions to be performed, the tools, techniques and methodologies, and defines an initial set of baseline software components. (Page 84) ***Secure Coding Guidance*** None	PMP
APP4030.7: CAT III – The Release Manger will ensure the SCM plan objects have security classifications labels. (Page 83) ***Secure Coding Guidance*** None	PMP

STIG Guideline	CDRL Guidance
APP4030.8: CAT II – The Release Manager will ensure the SCM plan identifies tools and version numbers used in the software development lifecycle. (Page 83) ***Secure Coding Guidance*** None	PMP SDP
APP4030.9: CAT III – The Release Manager will ensure the SCM plan identifies mechanisms for controlled access of simultaneous individuals updating the same application component. (Page 83) ***Secure Coding Guidance*** None	PMP SDP
APP4030.10: CAT II – The Release Manager will ensure the SCM plan assures only authorized changes by authorized persons are possible. (Page 84) ***Secure Coding Guidance*** None	PMP
APP4030.11: CAT III – The Release Manger will ensure the SCM plan identifies mechanisms for control access and audit changes between different versions of objects subject to configuration control. (Page 84) ***Secure Coding Guidance*** None	PMP
APP4030.12: CAT II – The Release Manager will ensure the SCM plan identifies mechanisms to track and audit all modifications of objects under configuration control. Audits will include the originator and data and time of the modification. (Page 84) ***Secure Coding Guidance*** None	PMP
APP4040.1: CAT II – The Release Manager will establish a CCB managing the CM process. (Page 84) ***Secure Coding Guidance*** None	PMP
APP4040.2: CAT II – The Release Manager will ensure the IAM is a member of the CCB. (Page 84) ***Secure Coding Guidance*** None	PMP
APP4040.3: CAT III – The Release Manager will ensure the CCB meets at least every release cycle or more often. (Page 84) ***Secure Coding Guidance*** None	PMP
APP5010: CAT III – The Test Manager will ensure at least one tester is designated to test for security flaws in addition to functional testing. (Page 85) ***Secure Coding Guidance*** None	TEP
APP2160.2: CAT II – The Test Manager will ensure both client and server machines are STIG compliant. (Page 85) ***Secure Coding Guidance*** None	TEP SEP
APP5030: CAT II – The Test Manager will ensure the application does not modify data files outside the scope of the application. (Page 85) ***Secure Coding Guidance*** None	SEP TEP STP
APP5040: CAT II – The Test Manager will ensure the changes to the application are assessed for IA and accreditation impact prior to implementation. (Page 85) ***Secure Coding Guidance*** None	PMP SEP TEP STP

STIG Guideline	CDRL Guidance
APP5050: CAT II – The Test Manager will ensure tests plans and procedures are created and executed prior to each release of the application or updates to system patches. (Page 85) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP5060: CAT II – The Test Manager will ensure tests procedures are created and at least annually executed to ensure system initialization, shutdown, and aborts are configured to ensure the system remains in a secure state. (Page 85) ***Secure Coding Guidance*** None	SEP TEP STP
APP5100: CAT III – The Test Manager will ensure fuzz testing is included in the test plans and procedures and performed for each application release based on application exposure. (Page 85) ***Secure Coding Guidance*** None	SEP TEP STP
APP5070: CAT III – The Test Manager will ensure code coverage statistics are maintained for each release of the application. (Page 86) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP5080: CAT II – The Test Manager will ensure a code review is performed before the application is released. (Page 86) ***Secure Coding Guidance*** None	SEP TEP SDP STP
APP5090: CAT II – The Test Manager will ensure flaws found during a code review are tracked in a defect tracking system. (Page 86) ***Secure Coding Guidance*** None	PMP SEP TEP SDP STP
APP5110: CAT II – The Test Manager will ensure security flaws are fixed or addressed in the project plan. (Page 86) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP2010.3: CAT II – The IAO will ensure all appointments to required IA roles are established in writing to include assigned duties and appointment criteria such as training , security clearance, and IT designation. (Page 91) ***Secure Coding Guidance*** None	PMP
APP2040.2: CAT II – The IAO will ensure the classification guide for the application data exists and is available to users. (Page 91) ***Secure Coding Guidance*** None	PMP
APP6010: CAT II – The IAO will ensure if an application is designated critical, the application is not hosted on a general-purpose machine. (Page 91) ***Secure Coding Guidance*** None	PMP
APP2020.6: CAT II – The IAO will ensure the application is deployed in a manner consistent with the Application Configuration Guide provided by the developers. (Page 91) ***Secure Coding Guidance*** None	PMP
APP3020.4: CAT II – The IAO will ensure identified mitigations to identified threats are implemented. (Page 91) ***Secure Coding Guidance*** None	PMP RMP

STIG Guideline	CDRL Guidance
APP6020: CAT II – The IAO shall ensure if a DoD STIG or NSA guide is not available, a third-party product will be configured by the following in descending order as available: (1) commercially accepted practices, (2) independent testing results, or (3) vendor literature. (Page 92) ***Secure Coding Guidance*** None	PMP SEP
APP2100.3: CAT II – The IAO will ensure the application is configured to comply with the DoD Ports and Protocols guidance. (Page 92) ***Secure Coding Guidance*** None	PMP SEP
APP2100.4: CAT II – The IAO will ensure mitigations have been applied from the vulnerability assessments for all ports used in the application. (Page 92) ***Secure Coding Guidance*** None	PMP RMP TEP STP
APP2110.3: CAT II – The IAO will ensure the application and all associated PPS are registered with the DoD PPS database. (Page 92) ***Secure Coding Guidance*** None	PMP
APP2150.2: CAT II – The IAO will ensure procedures are implemented to assure physical handling and storage of information is in accordance with the data's sensitivity. (Page 92) ***Secure Coding Guidance*** None	PMP SEP
APP6030: CAT II – The IAO will ensure unnecessary services are disabled or removed. (Page 93) ***Secure Coding Guidance*** None	PMP SEP
APP6040: CAT II – The IAO will ensure at least one application administrator has registered to receive update notifications or security alerts when automated alerts are available. (Page 93) ***Secure Coding Guidance*** None	PMP
APP6050: CAT II – The IAO will ensure the system and installed applications have current patches, security updates, and configuration settings. (Page 93) ***Secure Coding Guidance*** None	PMP SEP
APP6060: CAT I – The IAO will ensure the application is decommissioned when maintenance or support is no longer available. (Page 93) ***Secure Coding Guidance*** None	PMP
APP6070: CAT III – The IAO will ensure provisions are in place to notify users when an application is decommissioned. (Page 94) ***Secure Coding Guidance*** None	PMP
APP2140.2: CAT II – The IAO will ensure a security incident response process for the application is followed. (Page 94) ***Secure Coding Guidance*** None	PMP SEP
APP6080: CAT II – The IAO will ensure protections against DoS attacks are implemented. (Page 94) Relevant Secure Coding Guidance: No relevance to CERT secure coding standards ***Secure Coding Guidance*** None	PMP SEP

STIG Guideline	CDRL Guidance
APP6090: CAT III – The IAO will ensure the system alerts an administrator when low resource conditions are encountered. (Page 95) ***Secure Coding Guidance*** None	PMP SEP
APP3450.2: CAT II – The IAO will ensure application resources are protected with permission sets only allowing application administrator to modify these files. (Page 95) ***Secure Coding Guidance*** • FIO06-C Create files with appropriate access permissions • FIO15-C Ensure that file operations are performed in a secure directory	PMP SEP TEP SDP STP
APP3450.3: CAT II – The IAO will ensure access to format strings used by the application are restricted to authorized users. (Page 95) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP6100: CAT II – The IAO will ensure production database exports have database administration credentials and sensitive data removed before releasing the export. (Page 95) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP3290.2: CAT I – The IAO will ensure the PK-enabled applications are configured to honor only approved DoD PKI certificates. (Page 95) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP6110: CAT III – The IAO will review audit trails periodically based on system documentation recommendations or immediately upon system security events. (Page 95) ***Secure Coding Guidance*** None	PMP
APP6120: CAT II – The IAO will report all suspected violations of IA policies in accordance with DoD information system IA procedures. (Page 96) ***Secure Coding Guidance*** None	PMP
APP6130: CAT III – The IAO will ensure, for classified systems, application audit trails are continuously and automatically monitored, and alerts are provided immediately, when unusual or inappropriate activity is detected. (Page 96) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP6140: CAT II – The IAO will ensure application audit trails are retained for at least 1 year for applications without SAMI data, and 5 years for applications including SAMI data. (Page 96) ***Secure Coding Guidance*** None	PMP
APP3690.3: CAT II – The IAO will ensure the audit trail is readable only by application administrators and auditors. (Page 96) ***Secure Coding Guidance*** None	PMP
APP3690.4: CAT II – The IAO will ensure the audit trail is protected against modification or deletion except by application administrators and auditors. (Page 96) ***Secure Coding Guidance*** None	PMP
APP6160.1: CAT II – The IAO will ensure recovery procedures and technical system features exist so recovery is performed in a secure and verifiable manner. (Page 96) ***Secure Coding Guidance*** None	PMP SEP TEP STP

STIG Guideline	CDRL Guidance
APP6160.2: CAT II – The IAO will document circumstances inhibiting a trusted recovery. (Page 96) ***Secure Coding Guidance*** None	PMP
APP6170: CAT II – The IAO will ensure back-up copies of the applications software are stored in a fire-rated container and not collocated with operational software. (Page 97) ***Secure Coding Guidance*** None	PMP
APP6180: CAT II – The IAO will ensure procedures are in place to assure the appropriate physical and technical protection of the backup and restoration of the application. (Page 97) ***Secure Coding Guidance*** None	PMP
APP6190.1: CAT II – The IAO will ensure data backup is performed at least weekly. (Page 97) ***Secure Coding Guidance*** None	PMP
APP6190.2: CAT II – The IAO will ensure data backup is performed daily and recovery media is stored off-site at a location. (Page 97) ***Secure Coding Guidance*** None	PMP
APP6190.3: CAT II – The IAO will ensure data backup is accomplished by maintaining a redundant secondary system, not collocated, that can be activated without loss of data or disruption to the operation. (Page 97) ***Secure Coding Guidance*** None	PMP
APP6200.1: CAT II – The IAO shall ensure a disaster plan exists providing for the smooth transfer of all mission or business essential functions to an alternate site for the duration of an event with little or no loss of operational continuity. (Page 97) ***Secure Coding Guidance*** None	PMP
APP6200.2: CAT II – The IAO shall ensure a disaster plan exists providing for the resumption of mission or business essential functions within 24 hours of activation. (Page 97) ***Secure Coding Guidance*** None	PMP
APP6200.3: CAT II – The IAO shall ensure a disaster plan exists providing for the partial resumption of mission or business essential functions within 5 days of activation. (Page 97) ***Secure Coding Guidance*** None	PMP
APP6210: CAT II – The IAO will ensure an account management process is implemented, verifying only authorized users can gain access to the application and individual accounts designated as inactive, suspended, or terminated are promptly removed. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP6220: CAT I – The IAO will ensure passwords generated for users are not predictable and comply with the organizations password policy. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP6230: CAT II – The IAO will ensure the applications users do not use shared accounts. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP

STIG Guideline	CDRL Guidance
APP6240: CAT III – The IAO will ensure all user accounts are disabled which are authorized to have access to the application but have not authenticated within the past 35 days. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP6250: CAT II – The IAO will ensure unnecessary built-in application accounts are disabled. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP6260: CAT I – The IAO will ensure default passwords are changed. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP3320.8: CAT II – The IAO will configure the application to ensure account passwords conform to DoD password policy. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP3470.2: CAT II – The IAO will ensure access to privileged accounts is limited to privileged users. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP3470.3: CAT II – The IAO will ensure non-privileged accounts are limited to non-privileged users. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP3470.4: CAT II – The IAO will ensure the application account is established and administered in accordance with a role-based access scheme to enforce least privilege and separation of duties. (Page 98) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP3480.3: CAT II – The IAO will ensure the access procedures enforce the principles of separation of duties and “least privilege.” (Page 99) ***Secure Coding Guidance*** None	PMP SEP TEP STP
APP2160.3: CAT II – The IAO will ensure deployment systems and all components comply with all appropriate DoD STIGS, NSA guides, and all applicable DoD policies. (Page 99) ***Secure Coding Guidance*** None	PMP SEP TEP SDP STP
APP6270: CAT II – The IAO will ensure connections between the DoD enclave and the Internet or other public or commercial wide area networks require a DMZ. (Page 99) ***Secure Coding Guidance*** None	PMP SEP TEP
APP6280: CAT I – The IAO will ensure web servers are on separate network segments from the application and database servers if it is a tiered application. (Page 99) ***Secure Coding Guidance*** None	PMP SEP TEP
APP6290: CAT I – The Designer and the IAO will ensure physical operating system separation and physical application separation is employed between servers of different data types in the web tier of Increment 1/Phase 1 deployment of the DoD DMZ for Internet-facing applications. (Page 99) ***Secure Coding Guidance*** None	PMP SEP TEP

STIG Guideline	CDRL Guidance
APP6300: CAT II – The IAO will ensure an XML firewall is deployed to protect web services. (Page 100) ***Secure Coding Guidance*** None	PMP SEP TEP
APP6310: CAT II – The IAO will ensure web service inquiries to UDDI provide read-only access to the registry to anonymous users. (Page 100) ***Secure Coding Guidance*** None	PMP SEP TEP
APP6320: CAT II – The IAO will ensure if the UDDI registry contains sensitive information, read access to the UDDI registry is granted only to authenticated users. (Page 100) ***Secure Coding Guidance*** None	PMP SEP TEP
APP3830.2: CAT II – The IAO will ensure digital signatures exist on UDDI registry entries to verify the publisher. (Page 100) ***Secure Coding Guidance*** None	PMP SEP TEP
APP3840.2: CAT II – The IAO will ensure UDDI versions are used supporting digital signatures of registry entries. (Page 101) ***Secure Coding Guidance*** None	PMP SEP TEP
APP3850.2: CAT II – The IAO will ensure UDDI publishing is restricted to authenticated users. (Page 101) ***Secure Coding Guidance*** None	PMP SEP TEP

Acronym List

AS&D
Application Security and Development

CDRL
Contract Data Requirements List

CMMI
Capability Maturity Model Integration

DISA
Defense Information Systems Agency

DoD
Department of Defense

FCA
functional configuration audit

FFRDC
Federally Funded Research and Development Center

IA
information assurance

IMS
Integrated Master Schedule

PCA
physical configuration audit

PDR
preliminary design review

PM
program manager

PMP
Program Management Plan

RFP
request for proposal

RMP
Risk Management Plan

SCALe
Source Code Analysis Laboratory

SCI
Secure Coding Initiative

SDP
Software Development Plan

SEI
Software Engineering Institute

SEP
System Engineering Plan

SOW
statement of work

SRR
system requirements review

STIG
security technical implementation guide

STP
Software Test Plan

TEP
Test and Evaluation Plan

TSP
Team Software Process

V&V
verification and validation

References

URLs are valid as of the publication date of this document.

[Davis 2009]

Davis, Noopur; Miller, Philip L.; Nichols, William R.; & Seacord, Robert C. “TSP Secure.” *Proceedings of the Fourth Annual TSP Symposium.* New Orleans, LA, September 21-24, 2009. http://www.sei.cmu.edu/tspsymposium/2009/2009/DAY%203%20315%20PM%20TSP%20Secure.pdf

[DISA 2004]

Defense Information Systems Agency. *Database Security Technical Implementation Guide, Version 7, Release 1*. Defense Information Systems Agency, 2004. http://www.databasesecurity.com/dbsec/database-stig-v7r1.pdf

[DoD 2007]

Department of Defense. *DoD Directive 8500.01E on Information Assurance (IA).* Department of Defense, 2007. http://www.dtic.mil/whs/directives/corres/pdf/850001p.pdf.

[Seacord 2005]

Seacord, Robert C. & Householder, Allen. *A Structured Approach to Classifying Security Vulnerabilities* (CMU/SEI-2005-TN-003). Software Engineering Institute, Carnegie Mellon University, 2005. http://www.sei.cmu.edu/library/abstracts/reports/05tn003.cfm

[Seacord 2008]

Seacord, Robert C. *The CERT C Secure Coding Standard.* Addison-Wesley Professional, 2008 (ISBN: 0-321-56321-2).

[Seacord 2010]

Seacord, Robert; Dormann, Will; McCurley, James; Miller, Philip; Stoddard, Robert; Svoboda, David; & Welch, Jefferson. *Source Code Analysis Laboratory (SCALe) for Energy Delivery Systems* (CMU/SEI-2010-TR-021). Software Engineering Institute, Carnegie Mellon University, 2010. http://www.sei.cmu.edu/library/abstracts/reports/10tr021.cfm

[SEI 2012a]

Software Engineering Institute. *Secure Coding Standards*. https://www.securecoding.cert.org/confluence/display/seccode/CERT+Secure+Coding+Standards (2012).

[SEI 2012b]

Software Engineering Institute. *The CERT Oracle Secure Coding Standard for Java.* https://www.securecoding.cert.org/confluence/display/java/The+CERT+Oracle+Secure+Coding+Standard+for+Java (2012).

[SEI 2012c]

Software Engineering Institute. *CERT C++ Secure Coding Standard.* https://www.securecoding.cert.org/confluence/pages/viewpage.action?pageId=637 (2012).

[SEI 2012d]
Software Engineering Institute. *Secure Coding*. http://www.cert.org/secure-coding/ (2012).

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.			
1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE July 2012		3. REPORT TYPE AND DATES COVERED Final
4. TITLE AND SUBTITLE Supporting the Use of CERT® Secure Coding Standards in DoD Acquisitions			5. FUNDING NUMBERS FA8721-05-C-0003
6. AUTHOR(S) Tim Morrow, Robert Seacord, John Bergey, Philip Miller			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213			8. PERFORMING ORGANIZATION REPORT NUMBER CMU/SEI-2012-TN-016
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) HQ ESC/XPK 5 Eglin Street Hanscom AFB, MA 01731-2116			10. SPONSORING/MONITORING AGENCY REPORT NUMBER
11. SUPPLEMENTARY NOTES			
12A DISTRIBUTION/AVAILABILITY STATEMENT Unclassified/Unlimited, DTIC, NTIS			12B DISTRIBUTION CODE
13. ABSTRACT (MAXIMUM 200 WORDS)			
14. SUBJECT TERMS Statement of Work, SOW, secure coding, acquisitions, request for proposal, RFP			15. NUMBER OF PAGES 58
16. PRICE CODE			
17. SECURITY CLASSIFICATION OF REPORT Unclassified	18. SECURITY CLASSIFICATION OF THIS PAGE Unclassified	19. SECURITY CLASSIFICATION OF ABSTRACT Unclassified	20. LIMITATION OF ABSTRACT UL

13. ABSTRACT (MAXIMUM 200 WORDS)

The United States Department of Defense (DoD) increasingly depends on networked software systems. One result of this dependency is an increase in attacks on both military and non-military systems as attackers look to exploit software vulnerabilities. Program acquisition offices are emphasizing information assurance to address various threats. The Defense Information Systems Agency (DISA) created the Application Security and Development *Security Technical Implementation Guide* (STIG) in response to DoD Directive 8500.IE, which establishes policies and assigns responsibilities for achieving DoD information assurance. That STIG provides guidance for information assurance and security throughout a program's lifecycle, and it is specified as a requirement for DoD-developed, -architected, and -administered applications and systems that are connected to DoD networks.

This technical note provides guidance to help DoD acquisition programs address software security in acquisitions. It provides background on the development of secure coding standards, sample request for proposal (RFP) language, and a mapping of the Application Security and Development STIG to the CERT® C Secure Coding Standard.