RABET-V

Release 0.1.0

RABET-V Team

Nov 28, 2022

INTRODUCTION

CHAPTER ONE

PROGRAM DESCRIPTION

This document is for use through the life of the RABET-V Program. This initial version will also inform the RABET-V Pilot Program which is a trial execution of the Program Description. As such, some sections have additional commentary specific to the pilot. These sections are marked in indented italics like the example below.

Example Pilot comments

1.1 Program Administration and Research

The program will be administered by CIS team with assistance from The Turnout and PG Security Advisors. Dr. Mike Garcia will serve as the Research Lead.

1.2 Pilot Process

The RABET-V Pilot will first establish a RABET-V Program Description (this document), though it will continue to evolve over time as the program matures. The Program Description version will detail how each activity will be conducted. The Program Description will be iteratively reviewed and modified as necessary.

Using the Program Description, the Pilot Program will conduct initial reviews on real products from Pilot Program technology providers. The Pilot Program will work with pilot participants to develop their submission package and security claims.

Each initial review will execute all RABET-V activities resulting in the creation of Testing Rules and initial verification results for each product. The Architecture Review and Process Assessments will be conducted according to the architecture and process review steps detailed in the Program Description, which may be updated as necessary throughout the Pilot Program. This will result in risk-based product-specific testing processes. The Pilot will evaluate the value of its activities, along with the time and cost, and conclude with recommendations on the best approach.

The Pilot will then conduct multiple iterations of RABET-V on product revisions from the participants. Depending on the changes, RABET-V will adapt and conduct only the activities required. This exercise will highlight the effectiveness of RABET-V to create meaningful but streamlined verifications and help determine the effectiveness of the product architecture and process reviews. It will also provide useful time and cost information. After each RABET-V iteration, changes may be made to the testing process and the iteration repeated as necessary.

1.3 Pilot Research Questions

1.3.1 Time and Cost Implications

1. What are the review time implications of the RABET-V approach for:
    - The initial verification of a product from a new vendor?
    - The initial verification of a product from a vendor that have been through the RABET-V process?
    - The re-verification of a product?
2. What are the total cost implications of the RABET-V approach for:
    - The initial verification of a product from a new vendor?
    - The initial verification of a product from a vendor that have been through the RABET-V process?
    - The re-verification of a product?
3. Is there a viable economic model for the RABET-V process? If so:
    - Does it require a government agency to drive the program, similar to voting system certification?
    - Is there a model that suppliers in the market can support?
    - Is there a model that states and localities can support?
4. Will the process be efficient enough to keep costs low enough for vendors to make minor updates?

1.3.2 Market Maturity Implications

1. Is there evidence that products are architected in a manner that is mature enough for the RABET-V process to yield benefits by reducing the extent of re-verification reviews?
    - Will vendors be willing to submit small, frequent updates?
2. Is there evidence that state and local adoption and acceptance processes can leverage the RABET-V process to yield benefits?
    - Can states and localities accept RABET-V verifications quickly enough to make the process worthwhile?
    - Will states and localities be willing to adopt new versions at a rate that maintains incentives to put small, more frequent updates through the process?

1.3.3 Pre-Review Assessment Implications

1. Is there a sufficient correlation between process assessment results and verification outcomes to use those assessments to expedite verification and re-verification under RABET-V?
2. Should process assessments be renewed and, if so, how often or under what circumstances?
3. What party is best equipped to conduct process assessments?
4. Do architecture reviews provide a sufficient understanding of a given product to determine the impact of:
    - De minimus system changes?
    - Minor system changes?
    - Major system changes?
5. Should architecture reviews be renewed and, if so, how often or under what circumstances?

6. What party is best equipped to conduct architecture reviews?

1.3.4 Technical Evaluation Implications

1. For which types of non-voting election technology will the process work?
   - Is it better suited for some types of technology over others?
   - How, if at all, does the process have to be modified to make it more suitable?
   - Are vendors more likely to accept the RABET-V process for certain types of equipment?
   - Are states and localities more likely to accept the RABET-V process for certain types of equipment?

CHAPTER

TWO

RABET-V PROGRAM

RABET-V is a flexible, risk-based, and cost-effective approach to election system verification that will expedite verification of election systems while providing assurances of security and reliability.

> The RABET-V Pilot Program is designed to evaluate the RABET-V process and the potential of the process to improve the speed, security assurances, and cost-effectiveness of non-voting election technology verification.

For more information of the Background and Motivation for RABET-V, see CIS's How to Improve Election Technology Verification White Paper.

2.1 Introduction

This section introduces the program goal and scope. For a list of the definitions as they are used by the Program, see the *RABET-V Glossary*.

2.1.1 Program Goal

The RABET-V Program is a rapid, reliable, and cost-effective approach to verifying the security of non-voting election systems. Its goal is to provide assurances of security and reliability sufficient for stakeholders to have confidence in their use in election administration. Participating organizations will have demonstrated capabilities to effectively build, test, monitor, and maintain their election technology solution through an evidence-based process.

2.1.2 Program Scope

RABET-V is intended for non-voting election technology systems.

An election technology system is an information system that supports an elections administration process.

A "voting system" is defined in the Help American Vote Act (H.R. 3295, Sec 301) as "(1) the total combination of mechanical, electromechanical, or electronic equipment (including the software, firmware, and documentation required to program, control, and support the equipment) that is used—(A) to define ballots; (B) to cast and count votes; (C) to report or display election results; and (D) to maintain and produce any audit trail information; and (2) the practices and associated documentation used—(A) to identify system components and versions of such components; (B) to test the system during its development and maintenance; (C) to maintain records of system errors and defects; (D) to determine specific system changes to be made to a system after the initial qualification of the system; and (E) to make available any materials to the voter (such as notices, instructions, forms, or paper ballots)."

A non-voting system is any other information system used to administer an election. Examples include voter registration databases, electronic pollbooks, or the websites of government election authorities.

2.2 RABET-V Administrator

The RABET-V Administrator is a central body responsible for overseeing the RABET-V Program. The responsibility includes (some of the terms listed here are described later in this section):

- Accept requests from and manage list of RABET-V Subscribers
- Accept requests from and manage list of RABET-V Registered Technology Providers (RTPs)
- Host and manage content on the RABET-V Public Portal
- Manage the RABET-V Program Description, making changes as necessary and as approved by the Steering Committee
- Execute RABET-V Iterations when Product Submissions are made by RTPs
- Staff or delegate the execution of RABET-V activities with qualified individuals or organizations

  The Center for Internet Security, its staff, and contractors, are serving as the RABET-V Administrator for the RABET-V Pilot Program

2.3 Registered Technology Providers

A Registered Technology Provider (RTP) is an organization that develops election technology and has met the minimum requirements in this section.

To be a RABET-V Registered Technology Provider, the technology provider must submit a complete RTP request and agree to the RABET-V Program Commitment. An RTP is responsible for submitting their first Product Submission within 3 months of becoming an RTP.

Registered Technology Providers will be listed on the RABET-V Program Portal.

> For the Pilot Program, each pilot participant will be considered an RTP without going through the registration process.

2.3.1 Registered Technology Provider Request Package

Technology providers must submit a completed request package to become an RTP. A complete package will contain the following information:

- Company Name, Legal Address, and Address(es) of all locations
- Sales and Technical Support Points of Contact
- Website URL
- Company Description

2.3.2 Program Commitment

RTPs must agree to the RABET-V Program Commitment. The commitment establishes the ethical and responsible behavior expected by all program providers.

The Program Commitment requires:

- Accurate representation of the product capabilities and its security provisions to RABET-V administrators, RABET-V subscribers, customers, and other stakeholders
- Organization implementation and regular assessment of an organizational security framework like the CIS Controls. The RTP should perform organizational security audits regularly and provide the report to the RABET-V Administrator. The report will be provided to RABET-V Subscribers.
- Continuous product maintenance, including the patching of components within reasonable time frames

2.3.3 Provider Deregistration and Product Delisting

Failure to meet the requirements of the Program Commitment can lead to deregistration of the RTP and delisting of the RTP's products. Activities subject to deregistration are any that breach the Program Commitment. These include, but are not limited to:

- Inaccurate representation: if the vendor is found to have intentionally mislead RABET-V administrators or its customers as to the capabilities of the organization or the product.
- Lacking organization security: if the vendors fails to subscribe to an organizational security framework, like the CIS Controls, and maintain regular audits.
- Lacking product maintenance: if the vendor is no longer properly supporting a product such as with regular monitoring and maintenance.

2.3.4 Deregistration Process

The RTP will be notified of the reason for deregistration and given 90 days to remedy, with a reminder sent at 60 and 30 days. If the breach of Program Commitment has not been remedied within 90 days, the RTP will be deregistered.

2.3.5 Delisting Process

If a product has not been resubmitted to the RABET-V Program in the last three years, the RTP will be notified that the product may be delisted if it is not resubmitted within 90 days. Reminders will be sent to the RTP at 60 and 30 days.

2.4 Subscribers

RABET-V Subscribers are state and local election jurisdictions who intend to use the RABET-V reporting to assist in their certification, approval, or purchase decisions. RABET-V Subscribers will have access to the sensitive information produced by the RABET-V Program. Subscribers must submit a request to the RABET-V Administrator and agree to protect sensitive information. This information will be made available to Subscribers through a secure portion of the RABET-V Public Portal.

An agreement will be drafted for Subscribers to sign prior to being given access.

2.5 RABET-V Public Portal

2.5.1 Purposes

There will be a RABET-V public portal. The public portal serves the following purposes:

- Lists all RTPs
- Provides a product registry which list all submitted products and the product's latest goals, expected usage, and security claims
- Lists all Product Versions submitted by RTPs, the date of submission and date of completion, the outcome of the submission, and the RABET-V report

2.5.2 RABET-V Subscriber Access vs. Public

In an effort to maximize transparency, nearly all documentation about the RABET-V process is public. In addition, there are three broad classes of sharing for RABET-V documents and reports:

- Public: documents and reports that are made fully available to the public
- Shared with Subscribers: documents and reports that are made available to those federal, state, and local election authorities that have requested access and has agreed to treat the information as sensitive
- Shared with the RTP: documents and reports that are shared only with the RTP, though the RTP is free to share documentation with other entities as it sees fit.

CHAPTER

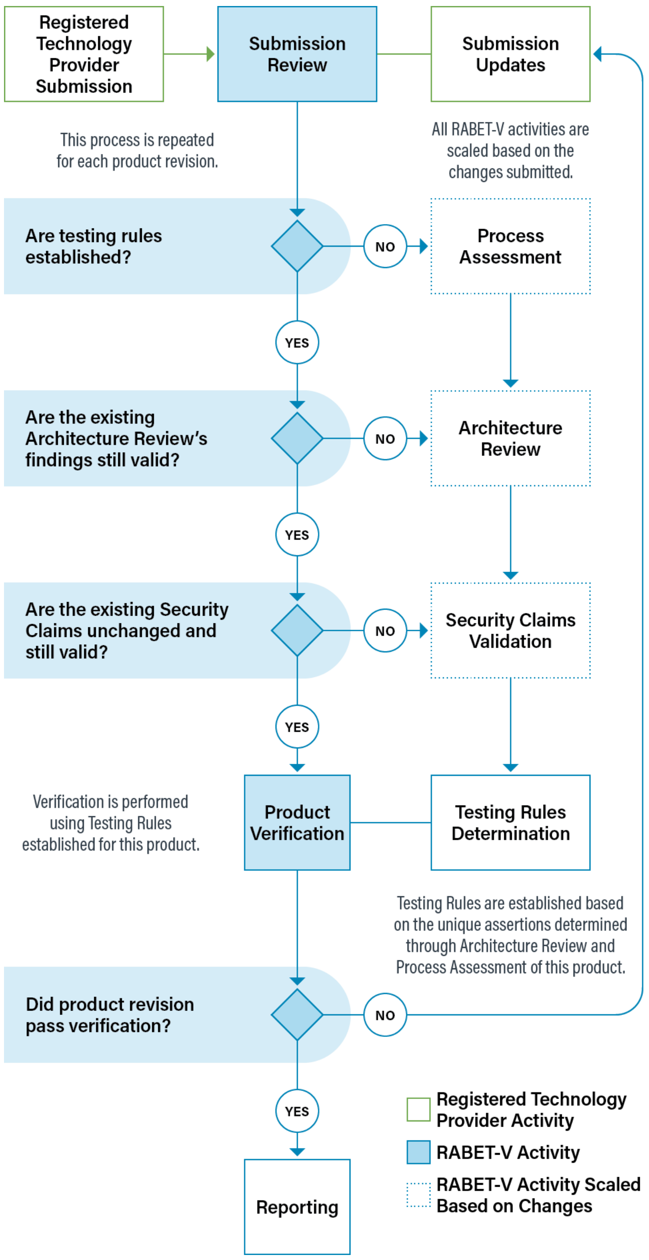
THREE

RABET-V ACTIVITIES

The RABET-V Program consists of eight discrete activities from RTP Submission to Reporting. Five of these activities scale or are eliminated based on risks attributed to the product changes from the previous submission. Risk decisions are informed by the product's architecture, the developer's processes, and their security claims. Each time the RABET-V process is initiated, it is called a RABET-V Iteration.

Throughout the process, certain activities produce scores that are shared with the RTP after the activity is complete. **All scores are considered tentative until the entire RABET-V process is complete.**

1. RTP Submission: A submission from an RTP begins the RABET-V iteration. This submission contains information from the RTP on both its organization and the product under review.
2. Submission Review: The submission is reviewed for completeness and to determine which of the remaining activities are necessary for the submission type.
3. Process Assessment: Assessment of the RTP's approach to developing software to determine its maturity, which will be used throughout the RABET-V process and subsequent submissions by the RTP. A demonstrably high level of maturity can reduce the burden of review across all activities. One can think of this as assessing the general trustworthiness of an RTP to reliably implement any given product feature or capability. A tentative score is provided to the RTP upon completion of the activity.
4. Architecture Review: Assessment of the product's architectural approach to determine its maturity with regard to various security services. A demonstrably high level of maturity can reduce the burden of review for a specific change. One can think of this as assessing the trustworthiness of the product that changes to one product feature or service will not have security implications for other aspects of the product. A tentative score is provided to the RTP upon completion of the activity.
5. Security Claims Validation: Assessment of whether the RTP's stated security approach are appropriate given the goals and expected use of the product. This assessment results in a set of security requirements that will used in producing the Testing Rules.
6. Testing Rules Determination: Produces a first-hit, crosstab decision table based on the outputs from the prior activities.
7. Product Verification: Executes the test plan.
8. Reporting: Produces reports and provides final results to stakeholders.


Registered Technology Provider Submission
Submission Review
Submission Updates
This process is repeated for each product revision.
All RABET-V activities are scaled based on the changes submitted.
Are testing rules established?
NO
Process Assessment
YES
Are the existing Architecture Review's findings still valid?
NO
Architecture Review
YES
Are the existing Security Claims unchanged and still valid?
NO
Security Claims Validation
YES
Verification is performed using Testing Rules established for this product.
Product Verification
Testing Rules Determination
Testing Rules are established based on the unique assertions determined through Architecture Review and Process Assessment of this product.
Did product revision pass verification?
NO
YES
Reporting
Registered Technology Provider Activity
RABET-V Activity
RABET-V Activity Scaled Based on Changes

CHAPTER

FOUR

RTP SUBMISSION

The RABET-V process begins with a product submission from an RTP.

4.1 Submission Types

All product submissions are either an Initial Submission or a Revision Submission.

4.1.1 Initial Submission

The Initial Submission is a first-time submission of product information. It includes statements about the product and the RTP that will be used throughout the RABET-V process. An Initial Submission is required for each unique product an RTP would market and sell independently to an election jurisdiction.

4.1.2 Revision Submission

A Revision Submission is for changes being made to a product that has already been through the RABET-V process. It includes information about changes to the product since the last submission.

An RTP can make a Revision Submission at any time after that product has been verified through an initial RABET-V iteration. It can improve the likelihood of a smooth process by engaging the RABET-V Administrator ahead of the submission about upcoming changes and understanding how any established Testing Rules will be impacted by deviations from the previous version.

A Revision Submission requires only the version change list, artifacts, desired deployment date, and version numbers, as well as any other meaningful changes, such as to the organizational process.

4.2 Submission Items

4.2.1 Product Goals

The Product Goals statement is a description of the product's purpose in non-technical language. It should be brief: a one or two paragraph summary of what the product is designed to do. The RTP can update the Product Goals during any Revision Submission and should always confirm whether there have been any changes.

This description will be used by the RABET-V Administrator in the Submission Review and Security Claims Review activities to determine if the stated security claims align with the product goals. For example, if the Product Goals include managing sensitive voter information, the RABET-V Administrator will expect to see security claims designed to protect sensitive voter information.

The Product Goals will be published in the RABET-V Public Portal.

Initial Submission: Always required

Revision Submission: When changed from last submission

4.2.2 Expected Usage

The Expected Usage statement describes how the RTP expects the election office to use the product. While it can communicate this through a number of means, a good approach is through high-level use cases that list the actions and interactions between involved parties and the system to achieve the Product Goals. Usage of the product will be limited to the use cases expressed in the Expected Usage. The RTP can update the Expected Usage during any Revision Submission and should always confirm whether there have been any changes.

This description will be used by the RABET-V Administrator in the Security Claims Review activity to determine if the stated security claims align with the expected usage.

The Expected Usage will be published in the RABET-V Public Portal.

Initial Submission: Always required

Revision Submission: When changed from last submission

4.2.3 Product Security Claims

The Product Security Claims statement is a listing of security requirements met by the product. The security requirements are listed as a part of the Security Service Capability Maturity Model and organized by security control family.

For each requirement, the RTP will describe the implementation approach and whether the requirement is Met, Partially Met, Not Met, or Not Applicable. If the RTP only implements the requirement on certain components, it should provide details and the rationale for excluding it from other components. The RTP should include well-reasoned arguments for the implementation decisions and how they result in the appropriate level of security for the product. This approach allows each product to implement a unique approach to the security of their application that is specific to its goals and usage.

The RTP can update Security Claims during any Revision Submission and should always confirm whether there have been any changes.

Initial Submission: Always required

Revision Submission: When changed from last submission

4.2.4 Process Descriptions

The Process Descriptions statement is about the RTP's development processes and operating environment. These should cover key aspects of software development as described in the OWASP Software Assurance Maturity Model (SAMM), which is used as the basis for the RABET-V Process Review Activity.

A lack of detail in the Process Description statement will not exclude the RTP from participating in the program, though it may slow the pace of the review.

Initial Submission: Always required

Revision Submission: When changed from last submission

> The pilot program will work with the provider to create the necessary descriptions.

4.2.5 Architecture Documentation, Diagrams, and Related Representations

The Architecture Documentation and Diagrams is a set of documents that fully describe the architectural design of the product. The product's architecture can be described using diagrams, narrative, or, ideally, a combination of the two.

The RTP should submit documentation of the architecture of the system as well as the software level. The system architecture should describe deployable subsystems, such as web services, databases, as well as hardware components such as firewalls and tablets. The software architecture should be described in terms of software components.

The term *component* is used generically within RABET-V to describe part of a product. Components can be broken down into subcomponents, as required. The architecture should be deconstructed to the level that exposed functionality (e.g. a particular web service, program API) can be identified.

RABET-V does not dictate a particular notation for submitted diagrams; however, where possible RTP's should follow provided examples, which are based on UML Component Diagrams.

RABET-V uses automated analysis tools to evaluate software architecture without direct access to source code. RTPs will be required to process their source code through such tools in order to make further software level analysis possible.

Initial Submission: Always required

Revision Submission: When changed from last submission

> A lack of architecture documentation and diagrams will not exclude a pilot participant from the program. The pilot program will work with the provider to create documentation and diagrams which are missing.

4.2.6 Third-Party Component Details

The Third-party Component Details describe the RTP's approach to managing supply chain risk. This includes whether the organization has selected third-party software components with a history of known vulnerabilities, and how the organization maintains traceability and assurance of third-party and open source software throughout the lifetime of the software.

When considering parts of the overall solution that are not developed internally, each unique version of the following will be considered an individual component of the system:

1. Operating System
2. Framework
3. 3rd party API
4. Embedded 3rd Party Library
5. Hosting Software/Service (e.g., IIS, Docker, Elastic Beanstalk, Azure App Service)
6. Database (database stored functions and procedures will be treated as a part of the software application)
7. File Storage System/Service
8. Network Appliance (virtual or real)
9. External Device Driver/Firmware

A change to one of these components will be treated a change to the entire component and the version number and change list will describe the entire component.

The RTP should detail initial and ongoing vetting procedures for third-party providers and components (if not covered in the Process Descriptions), to include open source software and libraries. Vetting should include fit for the provider as well as security and reliability. Management of third parties includes the approach to policies, SLAs, reputation, maintenance, and past performance of third-party software and services.

3rd Party Libraries will be processed through automated software bill of material (SBOM) tools. RTPs are required to facilitate the ingestion of software libraries through designated tooling.

Initial Submission: Always required

Revision Submission: When changed from last submission

> A lack of documented third-party component details will not exclude a participant from the program's pilot phase. The pilot program will work with the provider to develop the necessary documentation.

4.2.7 Product Environment and User Documentation

The RTP must provide access to a product environment that can be used by the administrator to conduct the RABET-V iteration. This should be a dedicated environment running the new product version. The administrator must provision user accounts and test data consistent with the Expected Usage statement. Test data should not include sensitive information, but may include data from real elections that is sanitized as necessary to remove personal information, product passwords, etc.

On the initial submission, the RTP should include user documentation and be available for a training session to assist the Administrator in understanding the product usage. Updated documentation and training sessions should be provided when changes are significant enough to warrant the update. User documentation must include the product version number it was written to support.

For many products, the product environment is the deployment of the web application to a sandbox hosting environment. For products like electronic pollbooks with physical devices, the product environment must include deployments of the product revision on physical devices provided to the administrator.

Initial Submission: Always required

Revision Submission: Always required

4.2.8 Revision Submission Artifacts

The RTP can submit a product revision to RABET-V at any time. Engaging the Administrator about upcoming changes and consulting the existing Testing Rules will help the RTP better prepare their submission.

All Revision Submissions require the following artifacts:

1. Change list - Indicates which components have changed and what level of change was made. It should reference the components identified in the architecture review.
2. Artifacts - The product development artifacts identified in the existing Process Review. These artifacts provide the necessary information on product changes to conduct a review of the changes in the Change List
3. Desired Deployment Date - Target date for deploying the product revision in a production environment.
4. Version number - The version number of the current product revision. It must indicate and correspond to code branches and change size (i.e. minor version number changes must correspond to minor changes).

A provider may change any of the Initial Submission items during a Revision Submission by providing updated information and alerting the Administrator. If they are not submitting updates for any given artifact, the RTP will have to attest to there being no change.

Initial Submission: Not applicable

Revision Submission: Always required

4.3 Submission

Once the Initial Submission or Revision Submission package is complete, it should be submitted electronically to the RABET-V Administrator.

For the pilot, CIS will provide a method to submit this information.

CHAPTER

FIVE

SUBMISSION REVIEW PROCESS

Once the RTP has made a submission, the RABET-V Administrator will review the submitted information and determine which RABET-V activities are necessary for this iteration.

5.1 Inputs

- The RTP's submission package
- The RTP's Process Assessment
- Prior reviews, if a Revision Submission

5.2 Outputs

- Submission Review Checklist indicating submission type, change type (for a revision submission), and which RABET-V activities should be performed in this iteration

5.3 Workflow

5.3.1 Review package for completion

See *RTP Submission* for submission requirements.

Initial submission

All RABET-V activities are required in order to generate the Testing Rules. Ensure all items on the Submission Review Checklist are included in the submission. For each step, indicate on the Submission Review Checklist if the respective item is present or missing.

Revision submission

Some RABET-V activities may not be required. Complete the remainder of the steps in this process to determine which activities are required for this submission. For each step, indicate on the Submission Review Checklist if the respective item is present, missing, or not required.

5.3.2 Validate change list

The approach to validating the change list will vary based on the findings of the prior Process Review:

1. Reliable: change list validation can be skipped or limited to high-level spot checking
2. Otherwise: validate the change list by manual or automated means

Record the result in the Submission Review Checklist.

5.3.3 Determine change type

(For revision submissions only)

Given the validated change list, determine which change types apply to the revision. Change types are listed below:

Change Type Number	Change Type Description
1	Other major or multiple change(s) to security service component(s)
2	Source code change to security service component(s)
3	Major configuration change to security service component(s)
4	Security patch of security service component(s)
5	Dependency updates for security service component(s)
6	Minor configuration change to security service component(s)
7	Source code change interfacing with security service component(s)
8	Source code change unrelated to security service component(s)
9	3rd party software patch to a non-security service component(s)
10	Operating system patch
11	Other software or configuration change

5.3.4 Determine if Process Assessment activity is necessary

The Process Assessment is required when one of the following conditions is true:

1. The submission is an Initial Submission
2. The RTP has requested a new Process Assessment in order to generate a new set of Testing Rules or update Software Development Maturity (SDM) scores
3. It has been more than 18 months since the last Process Assessment was performed
4. Artifacts provided by the RTP indicate a significant process change has occurred.

Record the result in the Submission Review Checklist.

5.3.5 Determine if Architecture Review activity is necessary

The Architecture Review is required when one of the following conditions is true:

1. The submission is an Initial Submission
2. The RTP has requested a new Architecture Review in order to generate a new set of Testing Rules or update Security Services Architectural Maturity (SSAM) scores
3. The change list indicates the addition, removal, or modification of major architectural components since the last Architecture Review

Record the result in the Submission Review Checklist.

5.3.6 Determine if Security Claims Validation activity is necessary

The Security Claims Validation activity is required when one of the following conditions is true:

1. The submission is an Initial Submission
2. The RTP has updated the product goals, expected usage, or security claims.
3. The RTP has requested a new Security Claims Validation in order to generate a new set of Testing Rules or update Security Services Capability Maturity (SSCM) scores
4. The change list indicates that prior Security Claims Validation findings need to be reviewed

Record the result in the Submission Review Checklist.

CHAPTER

SIX

PROCESS ASSESSMENT

The RABET-V Process Assessment measures the *Software Development Maturity (SDM)* of the RTP. It uses the OWASP Software Assurance Maturity Model (SAMM) as the basis for its evaluation. The SAMM determines a maturity score for the RTP in 5 areas across 15 principles. The RABET-V Process Assessment extends the SAMM by including principles for usability and accessibility to create the SDM. These maturity scores are used to help determine the types of testing conducted by RABET-V for product revisions.

In addition to providing the maturity scores, the SDM evaluation will determine the reliability of RTP-generated artifacts that can be used by RABET-V. By using RTP-generated artifacts, the RABET-V process will not have to reproduce these artifacts (i.e. test results). OWASP maintains a list of SAMM evaluators. Unless not practical, the SDM evaluation should be performed by one of these evaluators. These evaluators will review documentation and perform interviews with the RTP in order to complete the evaluation. Evidence of artifacts – such as historical version of reports – will be required.

The OWASP SAMM project makes a toolkit available. This toolkit provides an interview option for evaluating the RTP's processes according to SAMM.

6.1 Inputs

- Process descriptions
- Interviews with RTP

6.2 Outcomes

- Software Development Maturity Scores
- List of software development artifacts usable for verification

6.3 Workflow

6.3.1 Review Existing Documentation

The RTP submits existing documentation during the RTP Submission activity. The type of documentation requested includes:

1. Policy and Compliance documents that are related to or help define efforts related to acquiring, managing, designing, developing, testing, and supporting software at the organization.
2. Process related documents that help define which processes the RTP follows related to software activities.

3. A representative sample of artifacts from completed activities related to the above policy and compliance or process related activities.

   For the pilot, the Administrator will work with the RTP on documentation

6.3.2 Discussion Sessions

These sessions are for interactive discussions with the different roles supporting the efforts related to the RTP's software development process. The discussions will normally involve two interviewers and will last approximately 60-90 minutes. While sessions are driven by topics found in the SAMM toolkit, they will not be checklist-based, but discussions on how processes and procedure are implemented and conducted throughout the organization. Below are some of the common organizational roles that would be interviewed, however representatives from the appropriate business units are also useful candidates for interviewing:

1. Application/software security lead or equivalent party with responsibilities for defining and managing the integration of security into software
2. Business analyst or similar role with responsibilities related to requirements, user stories, etc.
3. Project manager or similar role with responsibilities for guiding teams through the processes to develop, acquire, and maintain software
4. Application architect or similar role with responsibilities to help ensure good design and architecture for applications is defined
5. Developer or similar role that has responsibilities to write code and some testing
6. Quality assurance/test or similar role that handling the primary testing for software or applications
7. DevOps engineer or similar role with responsibilities related to build and deployment processes for software
8. Incident response/support or similar roles with responsibilities for helping support, triage, respond to issues in production systems

6.3.3 Determine Artifact Reliability

RABET-V can expedite product verification if certain software development artifacts are found to be reliable. When artifacts are found to be reliable, the RABET-V process may use them instead of reproducing similar tests. However, this does not mean RABET-V must use them. In fact, from time to time, RABET-V may reproduce the results submitted by the RTP in order to validate the artifacts are still reliable.

The Process Assessment is used to help determine if the following artifacts are accurate and consistently available for RABET-V iterations. If the RTP has additional software development artifacts which they believe are reliable and beneficial to streamlining the RABET-V process, they may request those artifacts to be evaluated and the testing rules updated to account for the artifacts.

Change List

This is the most important software development artifact used by RABET-V when performing verification of a Product Revision. It is critical that the list is accurate, detailed, and complete. While RTPs can submit manually generated change lists, they may take longer to process than automated change lists built from the central source code repository and reviewed by system architects and product owners.

During the Process Assessment, the method used for building change lists will be discovered and sample change lists will be reviewed for accuracy and completeness. If the change list is determined to be reliable, the RABET-V process will use the RTP's change list and not generate its own. If the change list is not reliable, the RABET-V process will explore other ways to produce an accurate change list—which may take additional time and resources.

Automated Configuration Assessments

Security configurations are a major part of ensuring that systems contain properly implementing security controls. Using configuration guidance, such as the CIS Benchmarks, leads to consistent security outcomes. Automated configuration assessment tools, such as the CIS configuration assessment tool (CIS-CAT), can ensure guidance is being followed for every release.

During the Process Assessment, the reviewer will determine if the RTP is subscribed to configuration guidance and if they are using a reliable assessment tool. If so, the results of the assessment tool will be used during RABET-V iterations to verify certain requirements. If this artifact is not present or reliable, the Product Verification activity will have to perform additional testing to verify secure configurations.

Automated Vulnerability Assessments

Automated vulnerability assessments check system components for known vulnerabilities. These assessments primarily check third party components for known vulnerable versions of software. RTPs that are regularly performing automated vulnerability scans on the product networks and software will have their results used during the Product Verification activity in lieu of RABET-V reviewer performing new scans. During the Process Assessment, reviewers will investigate the scope, frequency, and tooling used by the technology provider to determine if there is sufficient coverage and accuracy.

Automated Unit Testing

Automated unit testing is a way to regression test large and complex applications efficiently. It takes significant investment on the part of the RTP to build test suites that are robust and accurate. For RTPs that have invested in this capability, the results of their internally testing can be used to offset of the RABET-V verification. The Process Assessment will look at the coverage and depth of the current automated testing routines, as well as the RTP's commitment to maintaining its test suites.

3rd Party Security Analysis

RABET-V strongly encourages RTPs to receive regular, in-depth security audits on their systems. For example, there are audits that focus on hosting security and application security. These audits, if performed against a reliable standard and performed recently, can be used in RABET-V in lieu of repeating similar evaluations.

6.3.4 Analysis and Reporting

Analysis of the provided documentation (if any) along with the captured session notes will be used to complete a SDM assessment for the organization. At the conclusion of the analysis, the following artifacts will be delivered as part of the work product for the organization:

1. High level executive summary of the process, findings, SDM maturity score and tailored recommendations
2. Completed SDM Toolbox
3. Interview session notes

6.4 Technical Guidance

1. OWASP SAMM
2. NIST Mitigating the Risk of Software Vulnerabilities by Adopting a Secure Software Development Framework (SSDF)
3. Managing Security Risks Inherent in the Use of Third-party Components

CHAPTER

SEVEN

ARCHITECTURE REVIEW METHODOLOGY

The RABET-V Architecture Review is designed to evaluate the solution's architectural support for the *RABET-V security control families*. This evaluation produces an architectural maturity score for each security control family and identifies the components that provide each *security service*. This score does not measure how well the *product* executes the security service (i.e., its capability level), just how mature the architecture is that supports the security service. The *Security Services Capability Maturity* level is a separate metric determined in the *Security Claims Validation* that indicates how well the product provides the security services.

The Architectural Maturity scores and component mappings are used to help assess the risk that changes to the product will negatively impact the security services. These are used in the *Testing Rules Determination Activity* to identify how to test the product changes. Higher architectural maturity scores, in conjunction with process maturity scores; may indicate the need for less testing required to validate that changes have not created increased risk in the product.

The Architecture Review activity is supplied with architecture diagrams, architecture descriptions, and interview sessions to confirm the architectural details and threat model of the product.

For more information about what is expected for the architecture diagrams and description, see the *Provider Submission* activity.

The Architecture Review will identify the product components at the system and software levels that expose functionality, and the security services that *protect* those functions.

This activity will also complete the system and software architectural viewpoints. The system level diagram identifies the larger components of the environment used to host and manage the software application(s). The software level diagrams identify the components a layer deeper into the software application(s).

7.1 Inputs

7.1.1 RTP Submission

The Technology Provider will supply architecture diagrams, architecture descriptions, and access to a functioning version of the solution. The Technology Provider process their source code through designated SBOM and software architecture analysis tools (currently Mend and Lattix). The architecture review will use the architecture tools and functioning solution to validate or fill-in missing pieces from the architecture diagrams and descriptions. For more information about what is expected for the architecture diagrams and description, see the *Provider Submission* activity.

7.1.2 Required Security Control Families

The Ten *Security Control Families* provide guidance as to the needed controls to help protect the product and related data.

7.1.3 Security Architecture Rubric

The rubric was created to help score the product architecture in the categories of Reliability, Manageability and Consistency, Maintainability (Modularity and Isolation), and Depth of control coverage (i.e., defense-in-depth)

7.2 Outputs

7.2.1 Product Security Architecture Maturity Workbook

These scores will be included in an architecture maturity workbook that will contain an executive summary tab, threat model results, and architecture scoring.

Product Security Architecture Maturity Scores

Based on the maturity scoring rubric, the architecture will be assigned scores at various levels for each security control family which corresponds to how well it supports the mitigations within that family. These scores are calculated at five layers, starting at the most detailed level of security service implementation per component or interface and rolling up to result in a master architecture score.

List of issues or concerns

Included in the workbook will be a list of threat modeling findings and any additional issues or concerns from the more detailed review of the software level architecture.

7.2.2 Software Architecture Report

The Architecture Review will identify the components of the system and how the security services are used in relation to those components.

Tasks

Perform Threat Modeling

Threat modeling takes the provider submitted architectural documentation as input along with interview sessions with individuals that possess knowledge about the system and software architecture. The security control families provided by the application are enumerated using the threat modeling methodology.

Outputs:

- SAMM Presentation
- Security Service Listing
- System Level Scores
- Threat Model

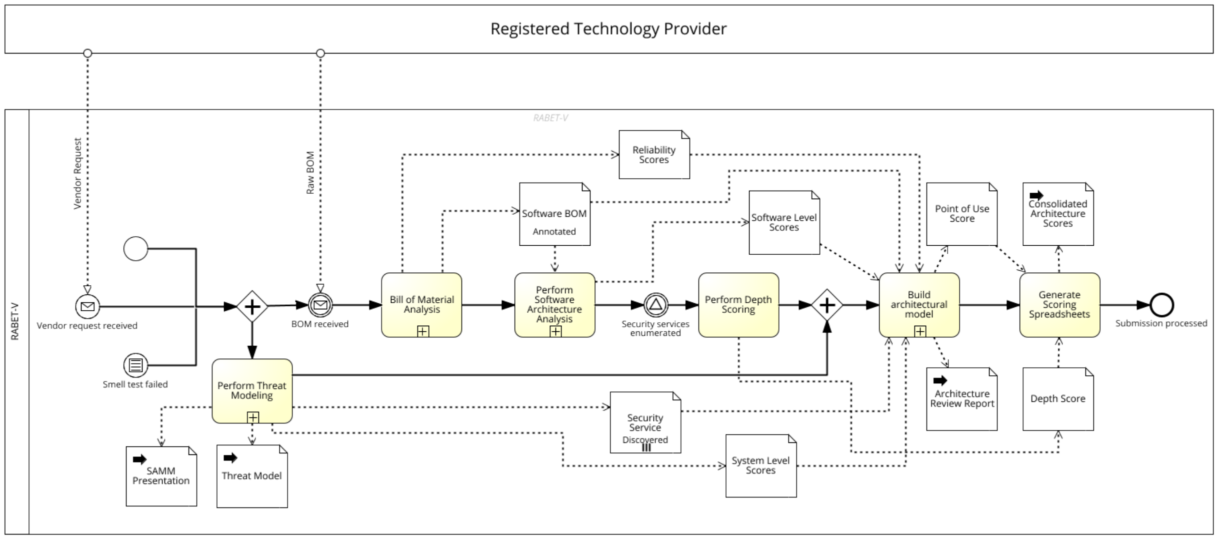


Fig. 1: A *BPMN* process model of the architecture review process

Bill of Material Analysis

Analyze the third party libraries used by the product, including licenses, maintainers, and known vulnerabilities.

Outputs:

- Reliability Scores
- Software BOM

Perform Software Architecture Analysis

Analyze the software architecture using tools and interviews.

Inputs: Software BOM

Outputs: Software Level Scores

Perform Depth Scoring

Outputs: Depth Score

Build architectural model

Create an architectural model containing the components, trust boundaries, and interfaces.

Inputs:

- Reliability Scores
- Security Service
- Software BOM
- Software Level Scores
- System Level Scores

Outputs:

- Architecture Review Report
- Point of Use Score

Generate Scoring Spreadsheets

Inputs:

- Depth Score
- Point of Use Score

Outputs: Consolidated Architecture Scores

CHAPTER

EIGHT

SECURITY CLAIMS VALIDATION

This security claims validation activity reviews whether the RTP's statements of security are sufficient for the product's goals and expected usage. Not all applications pose the same security risks. Even similar products can have different risk profiles based on the type of data they manage and how the product is used. This activity reviews the particulars of the product to ensure the security claims match its specific risk profile. This validation activity determines if the RTP's claims make sense given the product environment and data sensitivity, and if the claims are sufficient for the given context.

Security claims are submitted by RTP's in their submission package. The Initial RTP Submission must include claims for each security requirement. Subsequent Revision Submissions can add, remove, or modify any previous security claim.

For each requirement, the RTP must include:

1. Whether the requirement is:
    1. Met,
    2. Met, User Dependent,
    3. Partially Met,
    4. Not Met, or
    5. Not Applicable
2. Which component or sub-systems implements the requirement, and whether it is all or a sub-set of components
3. Rationale for any Not Applicable claims
4. Documentation on how the security service is to be configured for any Met, User Dependent claims
5. Implementation details
6. Explanation for why the requirement is only partially met or partially applied to the system. In some cases simple explanations will suffice (e.g., planned for future development, lack of resources). If the RTP believes that partial implementation is sufficient, a longer explanation is necessary.

The Product Verification activity performs the verification of these claims, based on the Testing Rules created in the Testing Rules Determination activity. The Testing Rules Determination accounts for the security claims made by the RTP For instance, the testing rules will exclude requirements that are Not Applicable.

8.1 Inputs

- Product goals (included in RTP's submission package)
- Product expected usage (included in RTP's submission package)
- Product security claims (included in RTP's submission package)
- Product demonstration or access
- Security Service Component Mapping (from architecture review)

8.2 Outputs

- Validation or rejection of security claims sufficiency
- List of applicable security requirements

8.3 Workflow

8.3.1 Review Product Goals, Expected Usage, and Product Functionality

This first step will review the written goals and usage from the RTP. This step should be augmented with a product demo or access to the product in a test environment. Reviewers should obtain a good sense of the high-level product functionality and validate the goals and usage consistent with the product functionality. For example, if there are use cases related to product administration, the reviewer should be able to access the administration module and exercise a few use cases.

8.3.2 Review Requirements listed as Not Applicable

The requirements marked Not Applicable are reviewed to ensure that for this product the requirements are not relevant and thus the SSCM scores should not reflect a non-conformity. This is done with the aid of the threat analysis and security service component mapping from architecture review. Using the component mapping and knowledge from the product demo and expected usage, the reviewer should be able to make a determination on whether or not the requirement is valid for this product. Often times, the decision comes down to the use of certain technologies in the system. For example, if the product disabled all wireless, the requirements on using encrypted wireless are not applicable.

8.3.3 Review Remaining Requirements

Once the list of applicable requirements has been determined, the reviewer will go through the applicable requirements. Using the security service component mapping and the implementation details, the reviewer can validate if the stated implementation is fully, partially, or not meeting the requirement. Requirements are fully met when the implementation of the requirement covers all the relevant components. If the implementation is only covering a portion of the relevant components, the requirement is partially met. The determination of which components are relevant is made by the reviewer.

8.3.4 Determine Claim Sufficiency

In this final step, the reviewer will analyze the product's use cases, the list of applicable requirements, and the RTP's validated claims. If there are sensitive use cases that are not mitigated to a minimally acceptable level, the reviewer may determine that the claims are not sufficient.

> Until more guidance can be developed on what is minimally acceptable, the guidance is that any product which claims to meet all applicable maturity level 1 requirements will pass this step.

8.4 Risk Considerations

When determining whether requirements are applicable and which components are relevant, the following considerations are used to help determine risk. This is not an exhaustive list.

8.4.1 Data Criticality

Different types of data carry different levels of criticality. The following sections review some of the most critical election data elements, identify the typical lifecycle for that data, and discuss the points where its value is the highest.

- Jurisdictional: The jurisdictional data used to assign ballot contents to the correct geographical districts and polling places is most at risk when it is used to build the ballots and assign them. This puts the entire supply chain of that data leading up to ballot generation at risk.
- Voter: While voter data varies between jurisdiction, some–especially non-public voter data–is a target for identity thieves and other criminals.
- Election definition: A combination of jurisdictional information, candidate filing information, and other attributes. Once the election definition is created and approved by the jurisdiction, its integrity has critical value.
- Blank Ballot: The collection of ballot contents into ballot styles. Modification of blank ballots can disenfranchise voters or manipulate how their intent will be read by the voting machine tabulators; they are most at risk from the time they are approved by the election jurisdiction to when they are presented to the voter for marking.
- Election Results: The aggregated totals generated from voting system tabulation functions. Election night results are a form of unofficial election results and are at significant risk of tampering and manipulation. The outcome of such tampering would lead to widespread confusion and distrust in the correct result produced by the voting system.

8.4.2 Election Operations

This consideration reviews how critical the product, and its services, are to an election's operations. Is the product a single point of failure? What options are available as backups for election officials? Is the product used during non-peak times or peak times?

CHAPTER

NINE

TESTING RULES DETERMINATION

This activity takes the results from previous activities and builds a unique set of Testing Rules for each product. These Testing Rules stay valid as long as none of the previous activities - Architecture Review, and Process Assessment - has changes. If any of those activities are performed during the current RABET-V Iteration, the Testing Rules Determination must be performed again.

The Testing Rules are structured as a first-hit, crosstab decision table. Artifacts from earlier activities, such as Submission Review, Process Assessment and Architecture Review serve as inputs to the table. The output of the Testing Rules Determination activity is a set of test methods to be used during Product Verification. A test method is determined for each security control family.

These test methods are `Full`, `Basic`, and `Streamlined`. The names reflect the rigour that each test method applies to confirm the effectiveness of the control family, with `Full` applying the most rigour and `Streamlined` the least.

The chosen test method for a given security control family is based on the Change Type(s) identified for the product's iteration and the SAMM and SSAM scores for the product. Change Types that indicate changes to security service components will require higher SAMM and SSAM scores to receive `Basic` or `Streamlined` testing. Minor changes may receive less testing even with relatively lower SAMM and SSAM scores.

9.1 Inputs

- Change Type
- Security Service Architectural Maturity Scores
- Software Development Maturity Scores

9.2 Outputs

- Product Testing Rules Matrix

9.3 Workflow

9.3.1 Pull test scores

Software Assurance Maturity Model (SAMM) and Security Service Architectural Maturity (SSAM) serve as inputs.

The scores for each Security Control Family in SSAM form the column headers of the table. The rows of the table list the change types. Each change type is associated with a score in SAMM. [provide more details here?] The first change type matching any of those identified during Submission Review [update] uniquely selects the applicable SAMM score (i.e. when more than one change type applies, the most impactful one takes precedence over the others). The SAMM score is then summed to the SSAM score for each security control family, deriving scores between 0.0 and 6.0.

9.3.2 Determine test methods

Each numeric score is converted to a test method based on a predefined set of thresholds associated with the Change Type. These thresholds determine how high a score must be to receive a certain level of testing. For example, a product with an *Operating system patch* change type and a combined Process + Architecture Score of 2.5 or greater will receive `Streamlined` testing. However, a change of *Security patch of security service component(s)* with the same score would receive `Full` testing. The testing matrix is given below:

Type	Change Description	Process Assessment Score Type	> 5	5 - 4.5	4.49 - 4.0	3.99 - 3.5	3.49 - 3.0	2.99 - 2.5	2.49 - 2.0	< 2.0
1	Other major or multiple change(s) to security service component(s)	Total	Full	Full	Full	Full	Full	Full	Full	Full
2	Source code change to security service component(s)	InternalDev	Basic	Full	Full	Full	Full	Full	Full	Full
3	Major configuration change to security service component(s)	EnvMgmt	Basic	Basic	Basic	Full	Full	Full	Full	Full
4	Security patch of security service component(s)	SupplyChain	Basic	Basic	Basic	Basic	Full	Full	Full	Full
5	Dependency updates for security service component(s)	SupplyChain	Basic	Basic	Basic	Basic	Basic	Full	Full	Full
6	Minor configuration change to security service component(s)	EnvMgmt	Basic	Basic	Basic	Basic	Basic	Basic	Basic	Full
7	Source code change interfacing with security service component(s)	InternalDev	Streamlined	Basic	Basic	Basic	Basic	Basic	Basic	Full
8	Source code change unrelated to security service component(s)	InternalDev	Streamlined	Streamlined	Streamlined	Basic	Basic	Basic	Basic	Full
9	3rd party software patch to a non-security service component(s)	SupplyChain	Streamlined	Streamlined	Streamlined	Streamlined	Streamlined	Basic	Basic	Full
10	Operating system patch	EnvMgmt	Streamlined	Streamlined	Streamlined	Streamlined	Streamlined	Streamlined	Basic	Full
11	Other software or configuration change	Total	Streamlined	Streamlined	Streamlined	Streamlined	Streamlined	Streamlined	Basic	Full

CHAPTER

TEN

PRODUCT VERIFICATION ACTIVITY

The purpose of the product verification activity is to establish the Security Services Capability Maturity (SSCM) scores for this product revision. For some product changes, this activity will be streamlined because the changes were determined to pose a low risk to the current security capability scores. For other changes, this activity will be extensive in order to determine, or redetermine, the proper maturity scores. The risk is determined in the Testing Rules which produces a Test Plan commensurate with the risk.

10.1 Inputs

- Testing Rules
- Product Revision Submission (Product Revision deployed to test environment, Product development artifacts)
- Component definitions from Architecture Review (used to set scope of testing)

10.2 Outputs

- Results of verification test methods

10.3 Workflow

10.3.1 Test Plan Generation

The Test Plan for the Product Verification activity is generated from the product's Testing Rules. The Testing Rules are built in the Testing Rules Determination activity and may be recently created or be existing rules from prior RABET-V iterations.

The Testing Rules are a decision table where each change is processed by the table and the end result is a verification method(s) to use. This must be done for all changes and the Test Plan is the aggregation of all verification methods.

For initial submissions, a full system test is performed. A full system test will review automated test results, perform a system wide functional test and penetration test.

10.3.2 Execute Test Plan for Security Requirements

The Test Plan will identify how to test the product revision using one or more of the verification methods. Each verification method has its own workflow.

10.3.3 Sanity Testing for Product Type Requirements

RABET-V is primarily a security verification process. However, it is critical that each product revision processed by RABET-V meet basic product requirements for its stated purpose. These basic product requirements will vary by product type and are managed separately from the RABET-V Program Description.

During the initial RABET-V iteration, partial testing (testing that is done to ensure that all the major and vital functionalities are working correctly) will be performed against this basic set of product requirements based on the product type. For subsequent RABET-V iterations, the testing rules will indicate whether sanity testing is necessary and whether it is limited or full. Limited sanity testing is focused on the changed component. Full testing will perform testing on all requirements.

10.4 Verification Methods

As indicated in the Test Plan, the RABET-V administrator, or its designee, will use one of more of the following techniques. The scope of the testing (i.e., which components to test) will also be indicated by the Test Plan.

10.4.1 Artifact Review

This method will review an artifact provided by the RTP. The review will look for gaps or concerns in relevant security controls based on the information provided. Each type of artifact will have various indicators of acceptability. Types of RTP artifacts include:

- Automated source code unit test results
- Automated vulnerability test results
- Automated configuration verification results
- Security event audit logs
- 3rd party security analysis results (automated or manual)

The artifacts must be evaluated as “reliable” during the Process Assessment activity in order to be used for Product Verification.

10.4.2 Automated Testing

Automated testing is a broad type of testing relying on software to perform test routines against the product or product component. Automated testing will execute the testing software against its target and produce results which will be evaluated by the RABET-V Administrator or its agent. The type of automated test will depend on the target. The types of automated testing currently conceived for RABET-V include:

- Configuration Testing
- Vulnerability Analysis
- Source Code Analysis
- Accessibility Testing

- Browser Compatibility Testing

10.4.3 Functional Testing

Functional testing is a broad type of testing that focuses on the system output (i.e., the functionality users can interact with). It is geared toward testing the functional requirements of the product and is a manual testing method. The scope and intensity of functional testing can vary, and there are sub-types of functional testing to indicate the scope and intensity. The following sub-types are used in RABET-V:

- Component - testing which evaluates a singular component and the requirements associated with that component
- Sanity - testing that is done to ensure that all the major and vital functionalities are working correctly
- Regression - testing performed to ensure that adding new code, enhancements, fixing of bugs is not breaking the existing functionality or causing any instability and still works according to the specifications.
- Integration - validation that multiple components work coherently when operating together.
- System/Sub-system - testing that is performed on a complete system or sub-system to verify it works as expected once all the modules or components are integrated.
- End to End - testing performed to verify the functionality of the product.
- Exploratory/Ad-hoc - informal testing to explore the application and looks for defects which exist in the application.

10.4.4 Penetration Testing

Penetration tests evaluate the product to find security vulnerabilities that an attacker could exploit. The scope of a penetration test may be the product's network, computer systems, or software application(s). In RABET-V, the penetration testing is limited to web application penetration testing. Web applications expose the greatest surface area for which automated testing is incapable of fully evaluating. Automated tools are fairly effective for network and computer systems where the major issues are vulnerable versions and lack of patching. Web applications, however, are custom and may have a variety of issues not easily captured by automated tools. Automated tools help with web application pen tests but must be used by skilled and experienced testers.

RABET-V relies on the OWASP Web Application Security Testing Guide to segment up the penetration testing options.

In addition to a full penetration testing option, the following web application penetration testing subtypes are supported:

- Configuration and Deployment
- Identity Management
- Authentication
- Authorization
- Session Management
- Input Validation
- Error Handling
- Cryptography

Limited penetration testing may be used if the changes do not warrant full penetration testing.

10.4.5 List of Master Requirements Workbook

Test Method Description

- Fuzzing - Test of the application's ability to accept a wide variety of inputs without causing it to enter an unexpected or undefined state.
- Penetration Testing - Testing that verifies the extent to which a system, device or process resists active attempts to compromise its security. [NIST SP 800-152]
- Functional - Test that evaluates the functionality of a component against a design specification. Can be automated, but because the function will be implemented differently by each product, a custom test script may be required for each.
- Web Testing - A functional test that exercises one or more parts of the web stack and verifies the expected output.
- Failover and restore testing - Test that evaluates the resiliency of a system by making components of the system inoperable and evaluating the result.
- Code analysis - A white box test involving the use of code artifacts, such as source code or unobfuscated binaries in order to verify certain properties.
- BOM Analysis - Analysis of the bill of materials, such as software and their versions.
- Configuration Audit - Test to verify that the configuration of a component is configured as required.
- Data Audit - Test to verify the presence or absence of certain records, such as the inappropriate collecting of PII or the lack of authentication logs, can be combined with Functional Testing to provide a higher level of confidence
- Artifact Review - Review of RTP-supplied artifacts from their development, testing, integration, and deployment process or artifacts provided by the RTP'S hosting environment.
- Documentation Audit - Review of the RTP-supplied documentation for presence of required content or presence of poor guidance (i.e. direction to use insecure password).
- Vendor Attestation - A statement made by the vendor indicating the existence of one or more security controls.

If the sanity or streamlined test calls for documentation, artifact, attestation, etc that doesn't exist, the full test is used. If the full test requires documentation, artifact, attestation, etc and none exist, the full test fails.

Full test is required for any new claim.

Verifiers may opt to do a full test at any time to validate documentation or artifacts.

10.5 Out-of-scope Testing

There is other testing which is out of scope for RABET-V. RABET-V is chiefly concerned with verifying the security and reliability of the product revision in a rapid way. Since rapid change cycles are possible with RABET-V, other user-centered types of testing can be performed by the current or potential end users and the changes reprocessed through RABET-V without significant lag. RABET-V reports can be used by state authorities or state and local users to determine the level of this testing necessary. These other testing types include acceptance, beta, and usability testing.

10.5.1 Acceptance Testing

Acceptance testing, or user acceptance testing (UAT), is performed by the client and verifies whether the end to end flow of the system meets their business requirements or not. The client accepts the system only when all the features and functionalities work as expected.

10.5.2 Beta Testing

Beta testing is carried out by the customer or potential customer. It is performed in the real environment before releasing the product to the market for the actual end-users. Beta testing is often used to ensure that there are no major feature gaps or bugs in the product, and it satisfies the business requirements. This testing is typically done on a beta version of the software or product that is releasd to a subset of end-users. It is the final testing done before releasing an application for commercial purpose.

10.5.3 Usability Testing

Under usability testing, the user-friendliness is verified. The application flow is tested to know if a new user can understand the application easily or not and if proper help documentation is provided. RABET-V measures the RTP's usability and accessibility testing process maturity, but the ultimate usability testing should be performed by end-users.

CHAPTER

ELEVEN

REPORTING PROCESS

11.1 Inputs

- Results from Product Verification activity

11.2 Outputs

- Decision (see Decision Types)
- RABET-V Product Report
- RABET-V Product Public Report

11.3 Workflow

11.3.1 Review of Product Verification Results and Determination

An internal review of the Product Verification Results will examine whether the product's verification met its claims.

The internal review will result in a Verification Status. The possible Verification Statuses are Verified, Conditional Verified, and Returned. These determinations are published in the Public Portal and may be updated if a Verification Status changes, most commonly when a Conditional Verified product has made adjustments that move it to a Verified status.

Verified

A Verified status means that the product is likely to perform as described in its Product Goals, and Security Claims in the Expected Usage operating environment.

Conditional Verified

A Conditional Verified status means that while the product is likely to perform as described in its Product Goals and Security Claims in the Expected Usage operating environment, the RABET-V process identified at least one non-critical issue or deviation.

With a Conditional Verification, the RTP is expected to remediate the issue and submit for a re-verification. If no other changes are made to the product, the process for re-verifying is considered part of the same submission and, upon review, can result in the Verification Status being changed to Verified.

Issues and deviations are detailed in the Product Report.

Returned

A Returned status means that the product does not perform as described in in its Product Goals and Security Claims. It has critical issues or deviations that are unlikely to be addressed through minor fixes. The RABET-V process identified at least one critical issue or deviation, severe enough that additional review will require a new submission.

Issues and deviations are detailed in the Product Report.

11.3.2 Product Report Generation

Report Template

The RABET-V Results Summary provides scored outcomes for product security capabilities and security architecture maturity and for organizational software development process maturity. For Revision Submissions, it will include any change from the previous submission.

Product Security Capability Maturity: the quality of the product's capabilities of the system at providing safeguards under each of these security control families:

- Authentication
- Authorization
- Injection Prevention
- Key/Secret/Credentials Management
- User Session Management
- Logging/Alerting
- Data confidentiality and integrity protection

Product Security Architecture Maturity: the quality and reliability of the product's architecture to support security services and the likelihood that product changes will impact the Product Security Capability Maturity levels:

- Authentication
- Authorization
- Injection Prevention
- Key/Secret/Credentials Management
- User Session Management
- Logging/Alerting
- Data confidentiality and integrity protection

Software Development Maturity: the quality of the RTP's processes in each of these areas:

- Governance
- Design
- Implementation
- Verification
- Operations
- Usability

Product (Revision) Summary

- Details about the product that were submitted including its description, expected usage (i.e. use cases), version number(s), etc. This includes the Change List for Revision Submissions.

Verification Methods

- Description of how the system was tested to include verification methods used in the testing.

Maturity Trends

- A description of what caused a change for any product or process maturity level that changed.

Appendices

- Requirements Scores: a list of all individual requirements and whether the RTP is meeting them

11.3.3 Product Public Report Generation

Each completed Verification will have a public report that provides basic information on the verification. This information will include:

- A reference number for the review
- The product's name and version number
- The RTP's name
- The initial Verification Status and date
- The current Verification Status and date
- Contact information for the RTP
- Summary scores

CHAPTER

TWELVE

RABET-V SECURITY CONTROL FAMILIES

RABET-V defines ten *Security Control Families* that are used throughout the RABET-V process to help evaluate the products.

1. **Authentication:** Verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in an information system. NIST FIPS 200
2. **Authorization:** The right or a permission that is granted to a system entity to access a system resource. NIST SP 800-82 Rev. 2
3. **Injection Prevention:** The sanitization of data input and output (possibly by rejecting unacceptable inputs or outputs) to ensure malicious executable code is not executed.
4. **Key/Secret/Credentials Management:** The activities involving the handling of cryptographic keys and other related security parameters (e.g. passwords) during the entire life cycle of the keys, including their generation, storage, establishment, entry and output, and destruction. NIST CNSSI 4009-2015
5. **User Session Management:** The act of establishing, protecting, and, when necessary, demolishing the persistent interaction between a subscriber and an end point. Adapted from NIST SP 1800-17b
6. **Logging/Alerting:** The systemic management and monitoring of the events—the discrete interactions that happen within and between systems, applications, and users—occurring within an organization's systems and networks. Adapted from NIST SP 800-92
7. **Data confidentiality and integrity protection:** Assurance that the data has not been altered in an unauthorized manner. Data integrity covers data in storage, during processing, and while in transit. Adapted from NIST SP 800-33 Data Confidentiality deals with protecting against the disclosure of information by ensuring that the data is limited to those authorized or by representing the data in such a way that its semantics remain accessible only to those who possess some critical information (e.g., a key for decrypting the enciphered data). NIST SP 800-13
8. **Boundary protection:** Monitoring and control of communications at the external boundary of an information system to prevent and detect malicious and other unauthorized communications, through the use of boundary protection devices (e.g. gateways, routers, firewalls, guards, encrypted tunnels). NIST SP 800-53 Rev. 5
9. **System availability protection**: The property that data or information is accessible and usable upon demand by an authorized person. NIST SP 800-66 Rev. 1
10. **System integrity protection:** The activities based around protecting the quality that a system has when it performs its intended function in an unimpaired manner, free from unauthorized manipulation of the system, whether intentional or accidental. NIST SP 800-27 Rev. A

CHAPTER

THIRTEEN

RABET-V MATURITY INDEXES

RABET-V evaluates the security, reliability, and usability of products and provides a set of maturity scores for each product version. RABET-V provides three maturity indexes:

- *Software Development Maturity (SDM) Index* - identifies the maturity of the RTP's software development processes measured across 15 security practices and 2 usability/accessibility practices.
- *Security Services Architectural Maturity (SSAM) Index* - identifies the maturity and reliability of the product architecture to support each of the ten security control families.
- *Security Service Capability Maturity (SSCM) Index* - identifies the product's current capability level measured across the ten security control families.

CHAPTER FOURTEEN

SECURITY SERVICES CAPABILITY MATURITY INDEX

The SSCM Index provides a maturity score for each of ten security control families. The scores range from 0 to 3, where 3 is the best.

The scores are based on how well the product revision meets the security requirements set forth for each security control family. The requirements are pass/fail. Any assumptions made about the configuration or setup of the product must be documented with the result.

The scores are calculating for each security control family by taken the percentage of applicable requirements met at each maturity level and dividing that percentage by 100 resulting in a value between 0 and 1. Then, those values are summed to achieve a result between 0 and 3. This means that meeting 100% of maturity level 1 will result in 1.0 value which is added to the result for maturity levels 2 and 3.

CHAPTER

FIFTEEN

TESTING METHODS

Below is a non-exhaustive list of different types of tests that could be leveraged in the process of verifying the security controls

- **Fuzzing** Test of the application's ability to accept a wide variety of inputs without causing it to enter an unexpected or undefined state.
- **Penetration Testing** Test the applications resistance to various forms of common attacks
- **Functional Testing** Test that evaluates the functionality of a component against a design specification. Can be automated, but because the function will be implemented differently by each product, a custom test script may be required for each.
- **Web Testing** A functional test that exercises one or more parts of the web stack and verifies the expected output.
- **Failover and restore testing** Test that evaluates the resiliency of a system by making components of the system inoperable and evaluating the result.
- **Code analysis** A white box test involving the use of code artifacts, such as source code or unobfuscated binaries in order to verify certain properties.
- **BOM Analysis** Analysis of the bill of materials, such as software and their versions.
- **Configuration Audit** Test to verify that the configuration of a component is configured as required.
- **Data Audit** Test to verify the presence or absence of certain records, such as the inappropriate collecting of PII or the lack of authentication logs, can be combined with Functional Testing to provide a higher level of confidence
- **Attestation** A guided interview is conducted with vendor to confirm existence of security control

CHAPTER SIXTEEN

AUTHENTICATION REQUIREMENTS

16.1 Maturity Level 1

16.1.1 Default passwords are not used or are automatically changed as part of set up

Before deploying any new asset or instances, change all default passwords to have strong values consistent with policy.

Applies to: All components

Method: Copy

Reference: CIS Security Best Practices for Non-Voting Election Technology 2.4.2

16.1.2 Authentication is applied consistently through the application

Users are authenticated consistently through the application using an authentication service, with variations for different user types being permitted.

Applies to: All components

Method: Copy

Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.1

16.1.3 Encrypt or Hash All Authentication Credentials

Ensure that local accounts and accounts with third parties use this approach to store your credentials. This will limit the impact of a third-party provider breach from impacting the election technology. The encryption or hashing algorithm should be one approved for use by NIST.

Applies to: All components

Method: Copy

Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.4

16.1.4 Customer admins have access to an inventory of their user accounts

Maintain an inventory of all accounts organized by authentication system. Maintain an up-to-date list of accounts for each system and tie each account to an individual person wherever possible. Having this ability in the platform helps organizations manage their users.

Applies to: Web Components

Method: Derived

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.6

16.1.5 Allow Password Policy Customization

Allow customers to configure and enforce a strong password policy according to best practices - A password policy should be created and implemented so that passwords meet specific strength criteria.

Applies to: All Components

Method: Derived

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.2.3

16.1.6 Implement Protections Against Brute Force Attacks

Account lockout needs to be implemented to guard against brute forcing attacks against both the authentication and password reset functionality. After several tries on a specific user account, the account should be locked for a period of time or until unlocked by an administrative action or use of a separate authenticator controlled by the user. Additionally, it is best to continue the same failure message indicating that the credentials are incorrect or the account is locked to prevent an attacker from harvesting usernames.

Applies to: Web Components

Method: Copy

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.2.4

16.1.7 Provide Options for Multifactor Authentication

Allow users to protect their accounts with multifactor authentication. Allow users to choose the authenticator that works best for them, subject to meeting security requirements. Where possible, allow the issuance of multiple authenticators so that multiple combinations can still meet an MFA requirement and be used in the reissuance of lost or stolen authenticators.

Applies to: Web Components

Method: Copy

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.2.8

16.2 Maturity Level 2

16.2.1 Develop a Strong Password Reset System

The Password Reset systems will leverage access to email or other known authenticators, such as confirming possession of a hardware token or a mobile device. Email alone should be augmented by security questions. When you do ask questions for password resetting, base them on questions that are both hard to guess, hard to brute force, and are not available through social media or previous data breaches. Additionally, any password reset option must not reveal whether an account is valid, preventing username harvesting.

Applies to: Web Components

Method: Copy

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.2.2

16.2.2 Block commonly used passwords

When credentials are set up for a new account, those credentials are run against a list of commonly used password and password patterns to ensure that users are not using passwords that are easily guessable.

Applies to: Web components

Method: Re-interpretation

Reference: CIS Security Best Practices for Non-Voting Election Technology 2.4.4

16.2.3 Require Multifactor Authentication for All Administrative Access

Use MFA via encrypted channels for all administrative account access.

Election technology administrative accounts have tremendous capabilities to do harm if taken over through a social engineering or other attack. Protecting them with MFA is extremely important.

Reference: CIS Security Best Practices for Non-Voting Election Technology 2.4.5

16.2.4 Ensure Authentication is centrally managed

Configure access for all accounts through as few centralized points of authentication as possible, including network, security, and cloud systems.

This makes it easier to ensure all users are being properly authenticated with the appropriate level of scrutiny and can centralize authentication logging as well.

Applies to: Web components

Method: Copy

Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.2

16.2.5 Authentication visibility

Provide customers with visibility on user logins including the time, IP address of the login and user agents of the browser.

Applies to: All components

Method: New

16.3 Maturity Level 3

16.3.1 Enable the integration with organization authentication systems

By enabling customers to integrate their authentication system, such as Oauth and SAML, with the platform it makes it easier for them manage their users and ensure that users are maintained throughout the user life cycle.

Applies to: Web components

Method: Derived

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.2

16.3.2 Provide capability to identify unassociated accounts

Provide the ability for customer admins to identify and disable any account that cannot be associated with a business process or business owner.

> Try to document relevant business processes and owners to make auditing and maintaining accounts easier.

Applies to: Web components

Method: Derived

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.8

16.3.3 Automatically disable dormant accounts

Automatically disable dormant accounts after a set period of inactivity.

> This is especially helpful for critical components of the election technology and assist with the manual accounts audits that should be done on a periodic basis.

Applies to: Web components

Method: Copy

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.9

16.3.4 Ensure Temporary Accounts Have An Expiration Date

Ensure that all temporary accounts have an expiration date that is monitored and enforced.

> This best practice should be applied to contractor accounts and accounts that are meant to be temporary, such as election-specific accounts. It is OK for service accounts and employee accounts to not have an expiration date. Treat users as temporary whenever there is uncertainty

Applies to: Web components

Method: Copy

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.10

16.3.5 Require Multi-Factor Authentication

Require MFA for all user accounts, on all systems, whether managed on-site or by a third-party provider.

> This is one of the best protections against social engineering attacks.

Applies to: Web components

Method: Copy

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.3

16.3.6 Provide the ability for customer admins to revoke access

Establish and follow an automated process for revoking system access by disabling accounts immediately upon termination or change of responsibilities of an employee or contractor.

> Employee new hire, termination, promotion, and demotion checklists should include the steps to setting user permissions commensurate with the employee's job responsibilities, or lack thereof. This should apply to employees and contractors.

Applies to: Web components

Method: Copy

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.7

CHAPTER

SEVENTEEN

AUTHORIZATION REQUIREMENTS

17.1 Maturity Level 1

17.1.1 Platform provides an authorization system

Platform provides an authorization system, such as RBAC, that restricts access to sensitive data and functions - Protect all information stored on systems with file system, network share, claims, application, or database-specific access control lists. These controls will enforce the principle that only authorized individuals should have access to the information based on their need to access the information as a part of their responsibilities.

Applies to: All components

Reference: CIS Security Best Practices for Non-Voting Election Technology 4.2.1

17.1.2 Applications and Middleware Should Run With Minimal Privileges

If an application becomes compromised, it is important that the application itself and any middleware services be configured to run with minimal privileges. For instance, while the application layer or business layer needs the ability to read and write data to the underlying database, administrative credentials that grant access to other databases or tables should not be provided.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.2.7

17.1.3 Apply the Principle of Least Privilege

Provide the customer with the ability to make all access decisions based on the principle of least privilege. Based on permission settings, access should be denied when not explicitly allowed. Additionally, after an account is created, rights must be specifically added to that account to grant access to resources. Where defaults are used, the defaults should be the minimal level of permissions.

Applies to: All components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.4.1

17.1.4 Use Tokens to Prevent Forged Requests

In order to prevent Cross-Site Request Forgery (CSRF) attacks, you must embed a random value that is not known to third parties into the HTML form. This CSRF protection token must be unique to each request. This prevents a forged CSRF request from being submitted because the attacker does not know the value of the token.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.8

17.1.5 Verify object requests

The product must verify during each request for data that the user has authorization to the data object. This prevents authenticated users from accessing data above or outside of their permission set.

Applies to: All components

17.2 Maturity Level 2

17.2.1 Apply Access Controls Checks Consistently

Always apply the principle of complete mediation, forcing all requests through a common security gatekeeper. This ensures that access control checks are triggered whether or not the user is authenticated.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.4.2

17.2.2 Set the Cookie Domain and Path Correctly

The cookie domain and path scope should be set to the most restrictive settings for your application. Any wildcard domain scoped cookie must have a good justification for its existence.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.5.1

17.3 Maturity Level 3

17.3.1 Don’t Use Direct Object References for Access Control Checks

Do not allow direct references to files or parameters that can be manipulated to grant excessive access. Access control decisions must be based on the authenticated user identity and trusted server-side information.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.4.4

17.3.2 Enforce Access Control to Data through Automated Tools

Use an automated tool, such as host-based Data Loss Prevention, to enforce access controls to data even when the data is copied off a system.

This will help ensure sensitive data that is not properly labeled is still protected from leaving its host system.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology 4.2.9

CHAPTER

EIGHTEEN

BOUNDARY PROTECTIONS REQUIREMENTS

18.1 Maturity Level 1

18.1.1 Deny Communications with Known Malicious IP Addresses

Deny communications with known malicious or unused Internet IP addresses. Limit access to trusted and necessary IP address ranges at each of the organization's application and network boundaries.

> This can be done using a network firewall at the perimeter of your election network. Preventing access from known malicious IP addresses can be done for all election applications, even public facing ones. The Election Infrastructure Information Sharing and Analysis Center (EI-ISAC) provides list of known malicious IP addresses.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.1.3

18.1.2 Deny Communication over Unauthorized Ports

Deny communication over unauthorized transportation control protocol (TCP) or user datagram protocol (UDP) ports or application traffic to ensure that only authorized protocols are allowed to cross each of the organization's network boundaries.

> Election system boundaries should be configured to deny traffic on all ports except ports explicitly needed for legitimate traffic.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.1.4

18.1.3 Deploy Network-Based IDS Sensors

Deploy network-based Intrusion Detection Systems (IDS) sensors to look for unusual attack mechanisms and detect compromise of these systems at each of the organization's network boundaries.

> The EI-ISAC and the Albert sensors together capture and monitor networks traffic of election jurisdictions. Election technology deployed outside of the jurisdictions' network should have a similar technology deployed and monitored.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.1.6

18.1.4 Document Traffic Configuration Rules

All configuration rules that allow traffic to flow through network devices should be documented in a configuration management system with a specific business reason for each rule, a specific individual's name responsible for that business need, and an expected duration of the need.

> This is important for production networks that host election solutions. Exceptions are normal but should be few and must be removed when no longer necessary. This is one good reason to keep general purpose workstations in a separate network segment.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.3.2

18.1.5 Use MFA for managing Network Infrastructure

Manage Network Infrastructure Using Multifactor Authentication and encrypted sessions

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.3.5

18.1.6 Configure Perimeter Devices to Prevent Common Types of Attacks

Define strict "TCP keepalive" and "maximum connection" on all perimeter devices, such as firewalls and proxy servers. This assists with preventing the success of SYN Flood attacks. Another approach is leveraging SYN cookies to prevent TCP SYN floods.

> A SYN Flood is one of the most common forms of DDoS attacks observed by the MS-ISAC.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.5.4

18.1.7 Disable Wireless Access on Devices if it is Not Required

Disable wireless access on devices that do not have a business purpose for wireless access.

> Disable all wireless options on election technology devices that are not authorized to use wireless. Periodically review device settings to ensure wireless options (Wi-Fi, Bluetooth, etc.) remain off.

Applies to: On-prem components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.6.4

18.1.8 Documentation Clearly Identifies Wireless Capabilities

Product documentation clearly defines any required wireless capability associated with the product along with information regarding the security and management of those wireless capabilities.

> Identify election technology that uses a wireless connection, and document each access point. For Wi-Fi, this will be a Wi-Fi router and any endpoint devices. For Bluetooth and NFC, this may be multiple devices. The decision to enable wireless technology should be made by the election administrator using a risk-based decision-making process.

Applies to: On-prem components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.6.1

18.1.9 Provide Dedicated Wireless Networks

Create a separate wireless network for each separate use. Access from the wireless network should be treated as untrusted and filtered and audited accordingly.

> Use of any wireless technology in election technology should be isolated for a very specific purpose, and incoming connections from the wireless network should be handled with care.

Applies to: On-prem components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.6.10

18.1.10 Disable Wireless Peripheral Access to Devices

Disable wireless peripheral access of devices (such as Bluetooth and NFC), unless such access is required for a business purpose.

> Printers and other peripherals often have Bluetooth capabilities.

Applies to: On-prem components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.6.9

18.2 Maturity Level 2

18.2.1 Enable Firewall Logging

Enable firewall logging of accepted and denied traffic to determine where a DDoS may be originating from.

> Most election technology must be careful not to block based on IP address unless there is evidence of malicious behavior.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.5.3

18.2.2 Configure Devices to Detect and Alarm on Traffic Anomalies

Configure firewalls and intrusion detection/prevention devices to alarm on traffic anomalies. Establish and regularly validate baseline traffic patterns (volume and type) for public-facing websites.

> Active and automated monitoring during peak election periods is critical to early detection and mitigation of DDoS attacks.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.5.5

18.2.3 Limit Wireless Access on Client Devices to only Authorized Wireless Networks

Configure wireless access only on client machines that have an essential wireless business purpose. Allow access only to authorized wireless networks, and restrict access to other wireless networks.

> All Wi-Fi connected election technology devices must only connect to the authorized wireless access point and no other.

Applies to: On-prem components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.6.5

18.2.4 Disable Peer-to-Peer Wireless Network Capabilities on Wireless Clients

Disable peer-to-peer (ad hoc) wireless network capabilities on wireless clients.

Applies to: On-prem components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.6.6

18.2.5 Segment the Network Based on Sensitivity

Segment the network based on the label or classification level of the information stored on the servers, and locate all sensitive information on separated Virtual Local Area Networks (VLANs).

> Consider establishing unique networks for each election technology and service offering.

Applies to: On-prem components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.2.5

18.2.6 Apply Upstream Port and Packet Size Filtering

Have upstream network service provider or network appliance apply port and packet size filtering to limit unnecessary traffic to the product's network infrastructure.

> Work with upstream providers to filter out as much as possible that is not related to the election service being provided.

Applies to: Hosted Components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.5.2

18.3 Maturity Level 3

18.3.1 Deploy Network-Based Intrusion Prevention Systems

Deploy network-based Intrusion Prevention Systems (IPS) to block malicious network traffic at each of the organization's network boundaries.

> This should be applied to all network-connected election technology. It must be monitored and configured to ensure it does not prevent legitimate traffic.

Applies to: Hosted components

Reference: CIS Security Best Practices for Non-Voting Election Technology 1.1.7

18.3.2 Manage All Vendor-issued Devices Remotely Accessing sensitive networks

Scan all vendor issued devices remotely logging into the organization's network prior to accessing the network to ensure that each of the organization's security policies has been enforced.

Applies to: Hosted components

Reference: CIS Security Best Practices for Non-Voting Election Technology 1.1.12

18.3.3 Manage System's External Removable Media's Read/Write Configurations

Configure systems not to write data to external removable media, if there is no business need for supporting such devices.

This prevents someone with physical access to a system storing sensitive information from extracting that information onto a USB drive.

Applies to: On-prem components

Reference: CIS Security Best Practices for Non-Voting Election Technology 4.1.7

18.3.4 Limit Workstation-to-Workstation Communication

When not in use, limit workstation-to-workstation communication using technologies such as private VLANs or micro-segmentation.

Whenever possible, workstations should be limited to talking only to servers thereby limiting lateral movement between workstations.

Applies to: On-prem components

Reference: CIS Security Best Practices for Non-Voting Election Technology 4.2.7

18.3.5 Use Wireless Authentication Protocols That Require Mutual, Multifactor Authentication

Ensure that wireless networks use authentication protocols such as Extensible Authentication Protocol-Transport Layer Security (EAP/TLS) that requires mutual, multifactor authentication.

Use of wireless technology in election technology demands that all parties be properly and fully authenticated.

Applies to: On-prem components

Reference: CIS Security Best Practices for Non-Voting Election Technology 1.6.8

18.3.6 Limit Access to Trusted IP Address Ranges

By applying an allowlist of known trusted IP addresses this allows organizations to greatly reduce their attack surface.

> This can be done using a network firewall at the perimeter of your election network. Preventing access from known malicious IP addresses can be done for all election applications, even public facing ones. The Election Infrastructure Information Sharing and Analysis Center (EI-ISAC) provides list of known malicious IP addresses.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.1.3

CHAPTER

NINETEEN

DATA CONFIDENTIALITY AND INTEGRITY REQUIREMENTS

19.1 Maturity Level 1

19.1.1 Leverage the Advanced Encryption Standard (AES) to Encrypt Wireless Data

Leverage the Advanced Encryption Standard (AES) to encrypt wireless data in transit.

> Wi-Fi, Bluetooth, and NFC all support encrypted communication. Ensure Wi-Fi uses Wi-Fi Protected Access 2 (WPA2) or better.

Applies to: On-premise

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.6.7

19.1.2 Use Only Standardized and Extensively Reviewed Encryption Algorithms

Use only standardized and extensively reviewed encryption algorithms that are validated by trusted third parties, such as NIST.

> Use standard libraries available from reputable sources instead of developing your own cryptographic solutions.

Applies to: All components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 3.2.15

19.1.3 Use Valid HTTPS Certificates From a Reputable Certificate Authority

HTTPS certificates should be signed by a reputable certificate authority (CA). The name on the certificate should match the fully qualified domain name (FQDN) of the website. The certificate itself should be valid and not expired.

Applies to: Web Components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.2

19.1.4 Follow Secure Configuration Guidance for Cloud Storage

Follow guidance from CIS Foundations Benchmarks or other secure configuration guidance to ensure all cloud storage containers with sensitive election data are properly secured.

> CIS Foundations Benchmarks are available for Amazon Web Services, Microsoft Azure, Google Cloud, and Microsoft Office 365.

Applies to: Hosted component

> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.3.1

19.1.5 Encrypt Transmittal of Username and Authentication Credentials

Ensure that all account usernames and authentication credentials are transmitted across networks using encrypted channels.

> This includes network traffic and data moved using removable media.

Applies to: All components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.5

19.1.6 Limit the Use and Storage of Sensitive Data

Product ensures that sensitive data is not being unnecessarily transported or stored. Where possible, use tokenization to reduce data exposure risks.

Applies to: All components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.1

19.1.7 Use the Strict-Transport-Security Header

The Strict-Transport-Security header ensures that the browser does not talk to the server over non-TLS. This helps reduce the risk of TLS stripping attacks as implemented by the TLSsniff tool.

Applies to: Web components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.10

19.1.8 Disable Data Caching Using Cache Control Headers and Autocomplete

Browser data caching should be disabled using the cache control HTTP headers or meta tags within the hypertext markup language (HTML) page. Additionally, sensitive input fields, such as the login form, should have the autocomplete=off setting in the HTML form to instruct the browser not to cache the credentials.

Applies to: Web components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.3

19.1.9 Updated TLS Configuration on Servers

Weak ciphers must be disabled on all servers. For example, SSL v2, SSL v3, and TLS protocols prior to v1.2 have known weaknesses and are not considered secure. Additionally, disable the NULL, RC4, DES, and MD5 cipher suites. Ensure all key lengths are greater than 128 bits, use secure renegotiation, and disable compression.

Applies to: Web components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.5

19.1.10 Use TLS Everywhere

TLS should be used whenever data is transferred over a network. TLS must be applied to any authentication pages as well as all pages after the user is authenticated. If sensitive information (e.g., personal information) can be submitted before authentication, those features must also be sent over TLS.

Applies to: Web components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.6

19.1.11 Disable HTTP access for All TLS Enabled Resources

For all pages requiring protection by TLS, the same URL should not be accessible via the non-TLS channel.

Applies to: Web components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.9

19.1.12 Don't Disclose Too Much Information in Error Messages

Messages for authentication errors must be clear and, at the same time, must be written so that sensitive information about the system is not disclosed. For example, error messages that reveal that the userid is valid but that the corresponding password is incorrect confirms to an attacker that the account does exist on the system. Instead, provide only a message that indicates that the login failed.

Applies to: All components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.2.5

19.1.13 Display Generic Error Messages

Error messages should not reveal details about the internal state of the application. For example, file system path and stack information should not be exposed to the user through error messages.

Applies to: All components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.6.1

19.1.14 Store User Passwords Using a Strong, Iterative, Salted Hash

User passwords must be stored using secure hashing techniques with strong algorithms like PBKDF2, bcrypt, or SHA-512. Simply hashing the password a single time does not sufficiently protect the password. Use adaptive hashing (a work factor) combined with a randomly generated salt for each user to make the hash strong.

Applies to: All components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.8

19.2 Maturity Level 2

19.2.1 Encrypt the Hard Drive of All Vendor Issued Devices

Utilize approved whole disk encryption software to encrypt the hard drive of all devices issued by the vendor.

> Determine what sensitive information you will permit on employees' laptops and mobile devices. Ensure the hard drives of laptops and mobile devices are fully encrypted to prevent information from being stolen.

Applies to: On-prem components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.1.5

19.2.2 Encrypt Data on USB Storage Devices

If USB storage devices are required, all data stored on such devices must be encrypted while at rest.

Applies to: On-prem components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.1.8

19.2.3 Encrypt All Sensitive Information in Transit

Encrypt all sensitive information in transit.

> Consider whether the election data's confidentiality is sensitive. If you are unsure, consider it sensitive.

Applies to: All components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.2.3

19.2.4 Encrypt Sensitive Information at Rest

Encrypt all sensitive information at rest.

> Election databases and their backups, for example, should be encrypted to ensure they are protected from manipulation.

Applies to: All components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.2.4

19.2.5 Use Separate Storage Containers for Unique Data Classifications

Don't overload one container with data at various classification levels. Create separate containers with appropriate names and configuration settings for each data classification level.

> Follow your data classification scheme and establish containers based on sensitivity. Also, don't mix production and test data.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.3.4

19.2.6 Remove or Isolate Sensitive Data or Systems Not Regularly Accessed by the Organization

Remove sensitive data or systems not regularly accessed by the organization from the network.

> These systems should only be used as stand-alone systems (disconnected from the network) by the business unit needing to occasionally use the system or completely virtualized and powered off until needed. In addition, disconnect systems that store or process election data that do not absolutely have to be online. Do not leave USB devices with sensitive information plugged into machines when they are not in use.

Applies to: All components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.1.2

19.2.7 Don't Use Unvalidated Forwards or Redirects

An unvalidated forward can allow an attacker to access private content without authentication. Unvalidated redirects allow an attacker to lure victims into visiting malicious sites. Prevent these from occurring by conducting the appropriate access control checks before sending the user to the given location.

Applies to: Web Components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.4.3

19.3 Maturity Level 3

19.3.1 Monitor and Block Unauthorized Movement of Sensitive Data

Deploy an automated tool on network perimeters that monitors for unauthorized transfer of sensitive information and blocks such transfers while alerting information security personnel.

> Deploy and configure Data Loss Prevention (DLP) solutions to look for election and voter-related information that should not be leaving your network boundaries.
>
> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.1.3

19.3.2 Utilize an Active Discovery Tool to Identify Sensitive Data

Utilize an active discovery tool to identify all sensitive information stored, processed, or transmitted by the organization's technology systems, including those located on-site or at a remote service provider, and update the organization's sensitive information inventory.

This helps an organization find and secure all instances of sensitive election information.

Reference: CIS Security Best Practices for Non-Voting Election Technology 4.2.8

19.3.3 Digitally Sign Sensitive Information in Transit

Sensitive data should be digitally signed by its originator and verified by all components which read, store, or process the data.

The integrity of election data must be maintained throughout its lifecycle.

Applies to: All components

Reference: CIS Security Best Practices for Non-Voting Election Technology 4.2.2

19.3.4 Encrypt Data Stored in Cloud Storage Containers

Use application encryption with secret keys only known to the data owner(s) to protect confidential data stored in a cloud storage container.

This protects the data even in the event of a data breach of the cloud hosting provider or a misconfiguration of the cloud storage container's permissions.

Applies to: Hosted components

Reference: CIS Security Best Practices for Non-Voting Election Technology 4.3.2

CHAPTER

TWENTY

INJECTION PREVENTION REQUIREMENTS

In these requirements, *interpreted* is defined as: Input that may be treated as data or as code depending on its content.

20.1 Maturity Level 1

20.1.1 Use Secure HTTP Response Headers

[Public key pins is deprecated. Unclear if replacement is well supported]

To protect against cross-site scripting (XSS) and man-in-the-middle (MITM) attacks, use the Content Security Policy (CSP) and Public-Key-Pins headers.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.2

20.1.2 Validate Uploaded Files

When accepting file uploads from the user, make sure to validate the size of the file, the file type, and the file contents as well as ensure that it is not possible to override the destination path for the file.

Applies to: Components that accept file input

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.6

20.1.3 Set the Encoding for Your Application

For every page in your application, set the encoding using HTTP headers or meta tags within HTML. This ensures that the encoding of the page is always defined and that the browser will not have to determine the encoding on its own. Setting a consistent encoding, like Unicode transformation format 8 bit (UTF-8), for your application reduces the overall risk of issues like XSS.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.7

20.1.4 Use Allowlist On Interpreted Input

For input that will be interpreted, allowlist acceptable inputs. Only inputs that appear on the whitelist will be accepted.

Applies to: Interpreted inputs (including SQL)

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.10

20.1.5 Validate all input

For each user input field, there should be validation on the input content.

> Examples of validation include data type validation, length validation, pattern validation, among others.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.10

20.2 Maturity Level 2

20.2.1 Use Parameterized Inputs

Input to an interpreter (e.g. an SQL Engine) should be passed using parameterized input, such as a bind variable. If Dynamic SQL is constructed within stored procedures, the procedural database code must also use bind variables. For example `dbms_sql` (Oracle), `EXECUTE IMMEDIATE` (Oracle) and `execute sp_executesql` (SQL Server) allow dynamic SQL to be constructed from within stored procedures or triggers.

Applies to: Interpreted inputs

Satisfies: Prefer Whitelists Over Blacklists for Input Validation

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.9

20.2.2 Use the X-Frame-Options Header

Use the X-Frame-Options header to prevent content from being loaded by a foreign site in a frame. This mitigates Clickjacking attacks. For older browsers that do not support this header, add frame busting JavaScript code to mitigate Clickjacking (although this method is not foolproof and can be circumvented).

> The use of frame busting is only required for products that support browsers that do not support X-Frame-Options.

Applies to: Web components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.1

20.2.3 Use the Nosniff Header for Uploaded Content

When hosting user uploaded content that can be viewed by other users, use the X-Content-Type-Options: nosniff header so that browsers do not try to guess the data type. Sometimes the browser can be tricked into displaying the data type incorrectly (e.g., showing a GIF file as HTML). Always let the server or application determine the data type.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.3

20.2.4 Validate the Source of Input

The HTTP method used to make a request must be validated. For example, if input is expected from a POST request, do not accept the input variable from a GET request.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.4

20.2.5 Conduct Contextual Output Encoding

All output functions must contextually encode data before sending it to the user. Depending on where the output will end up in the HTML page, the output must be encoded differently. For example, data placed in the URL context must be encoded different than data placed in JavaScript context within the HTML page.

Applies to: Web components

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.3.5

20.3 Maturity Level 3

20.3.1 Deploy Web Application Firewalls (WAFs)

Protect web applications by deploying WAFs that inspect all traffic flowing to the web application for common web application attacks. For applications that are not web-based, specific application firewalls should be deployed if such tools are available for the given application type. If the traffic is encrypted, the device should either sit behind the encryption or be capable of decrypting the traffic prior to analysis. If neither option is appropriate, a host-based web application firewall should be deployed.

Applies to: All

Reference: CIS Security Best Practices for Non-Voting Election Technology 3.2.14

CHAPTER

TWENTYONE

LOGGING/ALERTING REQUIREMENTS

21.1 Maturity Level 1

21.1.1 Activate Audit Logging

Ensure that logging has been enabled on all systems and networking devices.

> Components of election technology solutions must utilize available logging capabilities to store system activity.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.3.1

21.1.2 Ensure Adequate Storage for Logs

The product must provide a mechanism to maintain the storage of logs over a certain period of time.

> Election technology components should be designed to store audit logs for multiple significant election events without losing any data. Logs should be retained for a minimum of 180 days with the option to archive logs for longer periods of time.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.3.2

21.1.3 Log All Authentication Activities

Log all authentication activities, whether successful or not.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.6.4

21.1.4 Log All Privilege Changes

Log all activities or occasions where the user's privilege level escalates.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.6.5

21.1.5 Do Not Log Inappropriate Data

While logging errors and auditing access is important, sensitive data must never be logged in an unencrypted form.

> For example, under HIPAA and PCI, it would be a violation to log sensitive data into the log itself unless the log is encrypted on the disk. Additionally, it can create a serious exposure point should the application itself become compromised.

Applies to: Products that handle sensitive data

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.6.8

21.1.6 Store Logs Securely

Logs must be stored and maintained appropriately to avoid information loss or tampering by an intruder. Log retention should also follow the retention policy set forth by the organization to meet regulatory requirements and provide enough information for forensic and incident response activities.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.6.9

21.2 Maturity Level 2

21.2.1 Alerting

Provide a mechanism to alert responsible parties to the occurrence of certain logged events.

> The method of alerting can vary, but must take the form of a "push" notification.

Applies to: All

21.2.2 Centralize Anti-Malware Logging

The product must allow all malware detection events to be sent to enterprise anti-malware administration tools and event log servers for analysis and alerting.

> This assist in the early detection of an incident and ensures the proper security personnel are alerted to malware on the network.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.3.4

21.2.3 Enable DNS Query Logging

Enable Domain Name System (DNS) query logging to detect hostname lookups for known malicious domains.

> This is used to detect attempts to reach known malicious sites from within your network. This will help detect malware and prevent it from communicating with its command and control infrastructure.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.3.5

21.2.4 Enable Command-Line Audit Logging

Enable command-line audit logging for command shells, such as Microsoft Powershell and Bash.

> A large percentage of malware uses Powershell and Bash. This logging will assist in the detection of malware and a better understanding of its impact.

Applies to: Appliance-based products, or other standard configurations that include a command shell.

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.3.6

21.2.5 Log and Alert on Changes to Administrative Group Membership

Configure systems to issue a log entry and alert when an account is added to or removed from any group assigned administrative privileges.

> Changes to election technology administrator accounts must be logged and alerted. Quick notification allows for timely remediation in the event of privilege escalation or other attack.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.4.8

21.2.6 Central Log Management

Logs must be aggregated to a central log management system for analysis and review.

> Networked election technology solutions must utilize central event logging. Central event logging is extremely beneficial for detecting events and ensuring event logs are properly protected.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.3.5

21.2.7 Enable Detailed Logging

Enable system logging to include detailed information such as an event source, date, user, timestamp, source addresses, destination addresses, and other useful elements.

> Election technology components particularly servers and those devices in publicly accessible network interfaces should capture detailed enough information to fully understand and reconstruct security incidents.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.3.6

21.2.8 Log User Activity

Log relevant use activity, at a minimum login times, pages/screens viewed. Take care to not log information that would violate voter or ballot privacy.

> This can greatly assist with understanding the impact of security incidents involving user accounts. This is especially important for administrators.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.6.10

21.2.9 Log Administrative Activities

Log all administrative activities on the application or any of its components.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.6.6

21.2.10 Log Access to Sensitive Data

Log all access to sensitive data. This is particularly important for corporations that have to meet regulatory requirements like Health Insurance Portability and Accountability Act (HIPAA), PCI, or Sarbanes-Oxley Act (SOX).

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.6.7

21.3 Maturity Level 3

21.3.1 Log and Alert on Unsuccessful Administrative Account Login

Configure systems to issue a log entry and alert on unsuccessful logins to an administrative account.

> This enables election technology administrators to detect attempts to brute force or socially engineer access to administrator accounts.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.4.9

21.3.2 Enforce Detail Logging for Access or Changes to Critical or Sensitive Data

Enforce detailed audit logging for access to sensitive data or changes to sensitive data using tools such as File Integrity Monitoring or Security Information and Event Monitoring.

> This can help detect a malicious attempt to alter the integrity of the data. Database level logging can be enabled to track all changes to the database.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 4.2.10

21.3.3 Monitor Attempts to Access Deactivated Accounts

Monitor attempts to access deactivated accounts through audit logging.

> This can alert election system administrators to likely malicious behavior.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.12

21.3.4 Alert on Account Login Behavior Deviation

Alert when users deviate from normal login behavior, such as time-of-day, workstation location, and duration.

> Major commercial systems have the capability to establish an activity baseline based on time of day, IP address, and other data. Where possible, set up alerts to anomalous behavior for early detection of a possible attack.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.13

21.3.5 Deploy SIEM or Log Analytic Tools

Support the use of Security Information and Event Management (SIEM) or log analytic tool for log correlation and analysis.

> Timely and accurate detection of potential security events is critical during peak election periods. A SIEM solution can greatly assist with this.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.3.4

CHAPTER

TWENTYTWO

SECRET MANAGEMENT REQUIREMENTS

22.1 Maturity Level 1

22.1.1 Don't Hardcode Credentials

Never allow credentials to be stored directly within the application code. While it can be convenient to test the application code with hardcoded credentials during development, this significantly increases risk and should be avoided.

Applies to: All

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.2.1

22.1.2 Store Credentials Securely

Modern web applications usually consist of multiple layers. The business logic tier often connects to the other tiers, such as a database. Connecting to a database, of course, requires authentication. The authentication credentials, if stored, must be stored in a centralized location that is under strict access control. Scattering credentials throughout the source code is not acceptable. Some development frameworks provide a centralized secure location for storing credentials. These encrypted stores should be leveraged when possible.

Applies to: All

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.2.6

22.1.3 Securely Exchange Encryption Keys

If encryption keys are exchanged or preset in your application, any key establishment or exchange must be performed over a secure channel.

Applies to: All

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.7

22.2 Maturity Level 2

22.2.1 Set Up Secure Key Generation Processes

When keys are generated and stored in your system, the product must use PKCS standards and provide a way for customers to securely generate those keys to provide mutual authentication and non-repudiation between components.

Applies to: All

Reference: CIS Security Best Practices for Non-Voting Election Technology A1.1.4

22.2.2 Use a FIPS 140-2 Validated Module

Use a cryptographic module that meets or exceeds FIPS 140-2 validation, operating in FIPS mode, for performing cryptographic operations.

It is only necessary that the cryptographic software is FIPS 140-2 certified, not the specific hardware.

Applies to: All

Reference: NIST Voluntary Voting System Guideline Requirements Version 2.0 (Draft) 13.3-A

22.3 Maturity Level 3

22.3.1 Use Hardware Security Modules or Key Management Service for keys

Use a Hardware Security Module (HSM) or Key Management Service (KMS) when using cryptographic keys. These products are tamper evident and provide a secure environment for the management and operation of keys.

Applies to: All

CHAPTER

TWENTYTHREE

SYSTEM AVAILABILITY REQUIREMENTS

23.1 Maturity Level 1

23.1.1 Ensure Regular Automated Backups

Ensure that all system data is automatically backed up on a regular basis.

> Backups of election data should be done on a nightly basis. There may be applications which need to back up data at even higher frequencies during critical election periods.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.4.1

23.1.2 Backup and Failover capabilities

Ensure application and data storage components have fail over options in the event of a service degradation for primary component.

23.2 Maturity Level 2

23.2.1 Backup data should be restorable

Verify backup data is restorable by performing a data restoration.

> This is important to do once per election or more frequently for some systems.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.4.3

23.3 Maturity Level 3

23.3.1 Establish DDoS Mitigation Services With a Third-Party DDoS Mitigation Provider

Obtain third-party DDoS mitigation services.

A number of DDoS protection services have made their offerings available to election jurisdictions. Whether free or at a cost, these services can be very helpful to protect the most critical internet-connected election functions.

Applies to: Hosted components

Reference: CIS Security Best Practices for Non-Voting Election Technology 1.5.6

CHAPTER

TWENTYFOUR

SYSTEM INTEGRITY REQUIREMENTS

24.1 Maturity Level 1

24.1.1 Install the Latest Stable Version of Any Security-Related Updates on All Network Devices

Install the latest stable version of any security-related updates on all network devices. Latest refers to all updates which were available prior to the internal product testing of the product.

> Ensure that you are monitoring for updates.
>
> The vendor must use the most recent security updates available at the beginning of the development cycle, or later.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.3.4

24.1.2 Perform Complete System Backups

Ensure that all of the organization's key systems are backed up as a complete system, through processes such as imaging, to enable the quick recovery of an entire system. On premises products must provide this capability.

> These types of backups should be done prior to each election for each type of election system used. This allows for quick recovery back to the known good version. Maintaining extra units created from these system backups is another good approach.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 1.4.2

24.1.3 Ensure Anti-Malware Software and Signatures Are Updated

For systems that support the use of anti-malware software, the product must allow an administrator to perform updates to its scanning engine and signature database.

> Ensure that all anti-malware instances are receiving signature updates. This requires periodic review of devices within the election technology system.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.3.2

24.1.4 Configure Devices to Not Auto-Run Content

Configure devices to not auto-run executable code from removable media.

> This helps ensure an attacker cannot insert a malicious device and execute it without having user credentials.

Applies to: Vendor supplied hardware

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.5.3

24.1.5 Use Port Protectors on Unused Ports

Cover all unused communication ports (e.g. USB, Thunderbolt, HDMI, etc.) on endpoint devices with locks or tamper-evident port protectors to ensure unauthorized devices are not inserted into the device. This must be done prior to delivery to the customer.

Applies to: Provider supplied hardware

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.5.6

24.2 Maturity Level 2

24.2.1 Establish Secure Configurations

Maintain documented, standard security configuration standards for all authorized operating systems and software such as the CIS Benchmarks.

> Using a vetted configuration standard, identify each component of the election technology and its secure configuration standard to use.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.1.1

24.2.2 Deploy Operating System Patches

Ensure operating systems are running the latest security updates provided by the software vendor. Latest refers to all updates which were available prior to the internal product testing of the product.

Applies to: Hosted components

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.2.4

24.2.3 Deploy Software Patches

Ensure that third-party software on all systems is running the latest security updates provided by the software vendor.

> Latest refers to all updates which were available prior to the internal product testing of the product.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.2.5

24.2.4 Utilize Centrally Managed Anti-Malware Software

Utilize centrally managed anti-malware software to continuously monitor and defend workstations and servers.

> All endpoints in an election technology solution must use properly installed and constantly running anti-malware software. Central management allows administrators to enforce this rule.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.3.1

24.2.5 Limit Access to Scripting Tools

Limit access to scripting tools (such as Microsoft PowerShell and Python) to only administrative or development users with the need to access those capabilities.

> Election technology may make use of these technologies, but access to them should be limited to only the most trusted and protected accounts.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.4.7

24.2.6 Configure Anti-Malware Scanning of Removable Devices

Configure devices so that they automatically conduct an anti-malware scan of removable media when inserted or connected.

> Use of USB devices is very common in election systems. Therefore, it is critical that all external devices be scanned for malware prior to use.

Applies to: Vendor supplied or controlled hardware

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.5.5

24.2.7 Use Standard Hardening Configuration Templates for Databases

For applications that rely on a database, use standard hardening configuration templates.

> CIS Benchmarks are available for various database offerings such as MySQL, SQL Server, and PostgreSQL. Guidance for cloud-based databases are also available.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 3.2.16

24.3 Maturity Level 3

24.3.1 Implement Automated Configuration Monitoring Systems

Utilize a Security Content Automation Protocol (SCAP) compliant or equivalent configuration monitoring system to verify all security configuration elements, catalog approved exceptions, and alert when unauthorized changes occur.

> This prevents accidental misconfiguration and allows RTPs the ability to prove the component has been properly and securely configured.

Reference: CIS Security Best Practices for Non-Voting Election Technology 2.1.4

24.3.2 Deploy System Configuration Management Tools

Deploy system configuration management tools that will automatically enforce and redeploy configuration settings to systems at regularly scheduled intervals.

> Where possible, each component should be inspected and updated with the latest known good secure configuration prior to use in any election.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.1.5

24.3.3 Enable Operating System Anti-Exploitation Features and Deploy Anti-Exploit Technologies

Enable anti-exploitation features such as Data Execution Prevention (DEP) or Address Space Layout Randomization (ASLR) that are available in an operating system, or deploy appropriate toolkits that can be configured to apply protection to a broader set of applications and executables.

> This applies to servers and other sensitive endpoints.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.3.3

24.3.4 Disable Access to USB Devices Where Possible

Disable the use of USB devices (including Thunderbolt) on a system. This completely removes the risk of removable USB media based attacks.

> This may not be feasible for all components. It should be feasible for servers and other devices which do not use USB connected devices.

Applies to: Vendor provided hardware

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.5.7

24.3.5 Use USB Write Blocker to Transfer Data Into Sensitive Systems

Use USB Write Blockers to allow a high integrity system to read the content of a USB device. This mitigates the risk of transferring any malicious payload.

> These devices should be used when transferring data into the voting system or the voter registration system using removable USB media.

Applies to: Vendor supplied hardware

> Reference: CIS Security Best Practices for Non-Voting Election Technology 2.5.8

24.3.6 No Single Points of Failure

Protect product reliability against any one system component failing by providing redundancy of critical components.

24.3.7 Deny application execution by default

Implement default-deny technologies (such as AppLocker) to only permit applications on an allow-list to execute on the product.

> An allow-list of acceptable applications should be established by the vendor based on the use-cases of the application.

Applies to: Vendor provided hardware

CHAPTER

TWENTYFIVE

USER SESSION MANAGEMENT REQUIREMENTS

25.1 Maturity Level 1

25.1.1 Lock Endpoint Device Sessions After Inactivity

Product must provide capability to automatically lock endpoint device sessions after a standard period of inactivity.

> This is a basic security control that should be used universally. Employees should also be trained to lock their computers whenever they leave them.

Applies to: Vendor provided hardware

> Reference: CIS Security Best Practices for Non-Voting Election Technology 5.1.11

25.1.2 Set the Cookie Expiration Time

Set the session cookie expiration time to a reasonable value given the sensitivity of the data. Non-expiring session cookies should only be allowed for applications with no sensitive information, such as one providing basic public information that is customized for a user.

Applies to: Web components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.5.2

25.1.3 Place a Logout Button on Every Page

Place the logout button or logout link in an easily accessible place for every authenticated page.

Scope: Web components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.5.3

25.1.4 Use Secure Cookie Attributes (i.e., HttpOnly and Secure Flags)

Set the session cookie with both the HttpOnly and Secure flags. This ensures that the session ID will not be accessible to client-side scripts and it will only be transmitted over HTTPS.

Applies to: Web applications

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.5.4

25.1.5 Ensure That Session Identifiers Are Sufficiently Random

Session tokens must be generated by secure random functions and must be at least 128 bits or provide 64 bits of entropy.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.5.5

25.1.6 Invalidate the Session after Logout

When the user logs out of the application, the session on the server must be destroyed. This ensures that the session cannot be accidentally revived.

Applies to: Web applications

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.5.6

25.1.7 Implement an Idle Session Timeout

When a user is not active for a period of time, the application should automatically log the user out.

> Be aware that Ajax applications may make recurring calls to the application, effectively resetting the time-out counter automatically.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.5.9

25.2 Maturity Level 2

25.2.1 Regenerate Session Tokens

Regenerate session tokens when the user authenticates to the application. Additionally, should the encryption status change, the session token must be regenerated.

Applies to: All

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.5.10

25.3 Maturity Level 3

25.3.1 Destroy Sessions at Any Sign of Tampering

Unless the application requires multiple simultaneous sessions for a single user, implement features to detect session cloning attempts. Should any sign of session cloning be detected, the session must be destroyed, forcing the real user to reauthenticate.

Applies to: Web components

> Reference: CIS Security Best Practices for Non-Voting Election Technology A1.5.7

CHAPTER

TWENTYSIX

SECURITY SERVICES ARCHITECTURAL MATURITY INDEX

The SSAM index provides scores that indicate how well the product's architecture is built to support each security service. This is a measure of the reliability of the security service and how isolated the security service is from other system changes. These maturity scores are measured during the RABET-V Architecture Review.

The SSAM Index provides a maturity score for each of ten security control families. The scores range from 0 to 3, where 3 is the best.

The SSAM Index provides maturity scores across *four measures*.

26.1 Definitions

26.1.1 Composite Service

A service that is composed of two or more coupled security service components in order to provide functionality. Most composites will consist of a security service that surfaces at the system level (core service), and an adaptor that uses that service (dependent service).

26.1.2 Transparent Service

A security service that is not directly or indirectly invoked by vendor software.

CHAPTER

TWENTYSEVEN

RUBRIC

27.1 Reliability and Quality

The Component (or the substantial logic thereof) is provided by a reputable party and actively maintained.

- 0 – Written in-house with minimal documentation or third-party component that is uncommon and/or not actively supported
- 1 – Third-party component is used, but may not be a current version or actively supported
- 2 – Mature, third-party component with multiple active contributors; configured by secure best practices/guidelines
- 3 – Using a mature, third-party component, that is actively supported by a professional community/organization, and is enforced by technical or procedural controls

27.2 Manageability and Consistency

The Component is: Centrally managed by the provider, configurations are tuned with best practices, configurations are enforced, and the configuration is under full change management with attribution.

- 0 – Component does not exhibit any of the criteria
- 1 – Component exhibits one or two criteria
- 2 – Component exhibits three of the criteria
- 3 - Component exhibits all four criteria

27.3 Maintainability: Modularity

Component is segregated from other components at the system level and dedicated to providing its security service

- 0 – no segregation, not separated into own library
- 1 – separated into library (inclusive of namespace segregation)
- 2 – separated process, same execution environment as protected component
- 3 – separate unit of deployment (cloud service, or physically)

27.4 Maintainability: Isolation (Composite Services Only)

Access to the security service component is mediated through a central software component.

- 0 – No use of façade or proxy class
- 1 – Partial use of façade or proxy class
- 2 – Consistent use of façade or proxy class
- 3 – Invocation of security service is handled by global handler, framework or platform (i.e. it is written in such a way that its usage is guaranteed)

27.5 Depth

Component is segregated from other components and reusable inside other components. Components are complimentary to provide a consistent, layered defense for the overall system. There should not be multiple versions or flavors variations of the security service component unless absolutely necessary.

- 0 – Components coverage is lacking and/or haphazardly applied
- 1 – Component coverage has gaps, is managed inconsistentlyComponent coverage has gaps, is managed inconsistently, and is not segregated
- 2 – Components coverage has minimal gaps, some layeringComponent coverage has minimal gaps, some layering and segregation, and part of a repeatable process
- 3 – Components are intentional, built into layersComponents are intentional, built into layers, part of a repeatable/auditable process, and tested regularly

CHAPTER

TWENTYEIGHT

RUBRIC CONFIGURATION

Each use of a security service is scored separately (excepting depth). For example, if Log4Net and EnterpriseLibrary.Logging were used as Logging and Alerting Services, each would be scored separately across the measures below.

Scoring is based on three measures, with maintainability broken down into modularity (for system level services) and isolation (for software-only or composite services). Depth is scored once per security service type, at the aggregate level only.

Table 1 - Rubric configuration per use of security service provider

TYPE	RELIABILITY	CONSISTENCY	MODULARITY	ISOLATION	EXAMPLE
Transparent	x	x	x		Firewall
Composite	x	x	Service Only	Software Only	Azure AD integrated with App

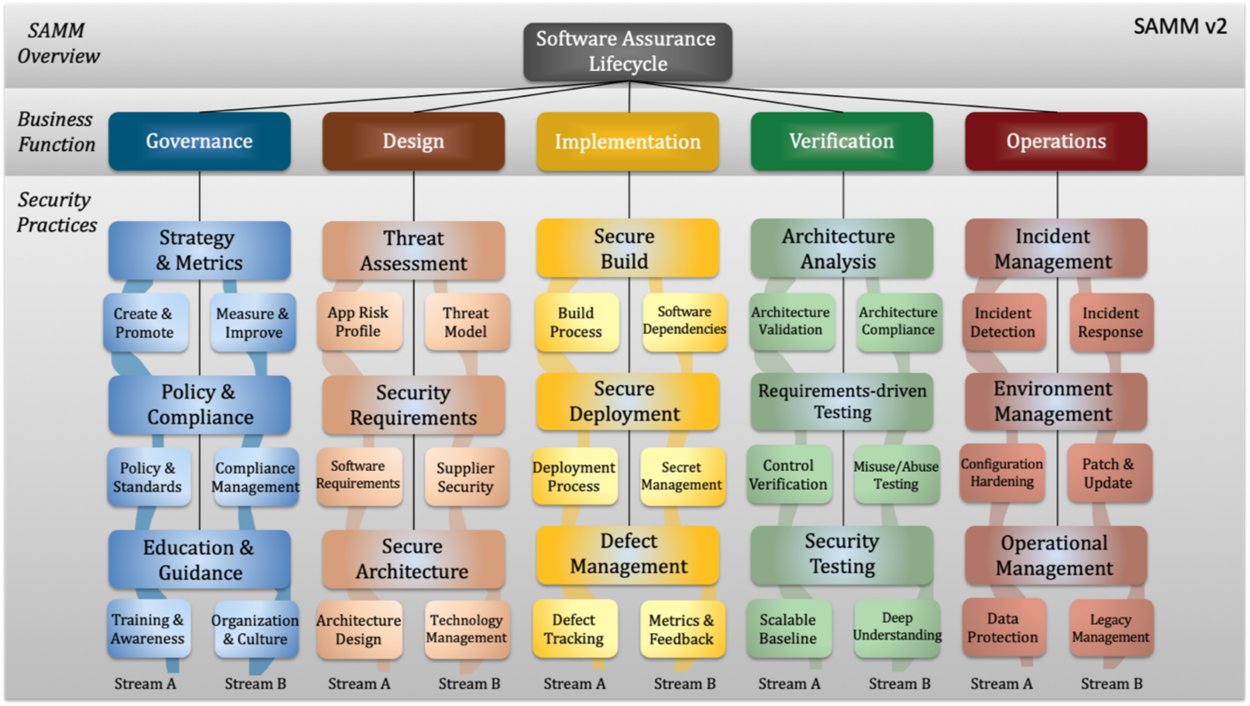
CHAPTER

TWENTYNINE

SOFTWARE DEVELOPMENT MATURITY INDEX

The SDM Index score is measured by the RABET-V Process Review activity and indicates the maturity of the RTP's software development processes for security and usability. The RABET-V SDM score is based on the OWASP Software Assurance Maturity Model (SAMM).

Maturity scores are provided for each of the 17 software development areas (15 SAMM plus Usability and Accessibility). The scores range from 0 to 3, where 3 is the best. The graphic below represents the SAMM process with its 15 areas.



CHAPTER

THIRTY

ACCESSIBILITY

Accessibility is often overlooked as a development priority. It may be hard for developers without a disability to conceptualize needing or using accessibility features, but it's easy to find examples that may be possible for anyone to imagine. For example, some software developers developed repetitive stress injuries and turned to speech-to-text aids to continue working in their profession. Beyond the general necessity, adhering to accessibility standards is often a hard requirement for software solutions in many state systems.

Accessibility Maturity Levels	Quality Criteria	Required Activity
Level 0		
Level 1: automated conformance to accessibility guidelines	Performs automated accessibility validation during development.	Use automated testing tools during development for: All major releases (partial credit)All significant changes to user interface functionality (full credit)Other (no credit)
Level 2: Testing with accessibility tools	Perform accessibility tests with commercial accessibility software and OS-specific features, including using personas and scenarios	Use commercial software, OS-specific features, and personas and scenarios for:All major releases (partial credit)All significant changes to user interface functionality (full credit)Other (no credit)
Level 3: formal accessibility testing and analysis program	Use of research methods and usability experts to test prototypes with users that have accessibility needs.	Conduct accessibility testing and integrate results for: All major releases (partial credit)All significant changes to user interface functionality (full credit)Other (no credit)

CHAPTER

THIRTYONE

USABILITY

Usability testing and analysis helps bridge the gap between a solution that meets a set of requirements and a solution that meets the needs of the organization, people, and processes. Meeting usability objectives is the distinction is between a solution that people want to use (i.e., meets a set of requirements and usability needs) versus one they don't (i.e., solely meets a set of requirements).

Users will attempt to reduce friction in completing their desired task. A poorly designed user experience will result in users finding workarounds, often circumventing well-intentioned security controls. For a product to achieve the risk mitigation intended by the security requirements, it must integrate usability principles with security controls and, thus, an organization's maturity in implementing usability is critical to its security outcomes.

Usability Maturity Levels	Quality Criteria	Required Activity
Level 0		
Level 1: formally established feedback loops with customers	Established processes for receiving feedback from customers and incorporating that feedback into the product	Incorporation of feedback into products for: All major releases (partial credit)All updates involving user-facing functionality (full credit)Other(no credit)
Level 2: deploy enhanced feedback capabilities	Interview users, accept feedback directly through the product, collect logs and analytics through the product, or other similar approaches; from these, product form reports on findings and plans for incorporating feedback	Use commercial software, OS-specific features, and personas and scenarios for: Most major releases (partial credit)All significant changes to user interface functionality (full credit)Other (no credit)
Level 3: formal usability testing and analysis program	Formal research on the business processes and users' behaviors, and conduct usability studies with users interacting with a prototype or version of the software solution.	Conduct formal usability testing and integrate results for: Most major releases (partial credit)All significant changes to user interface functionality (full credit)Other (no credit)

CHAPTER

THIRTYTWO

DOCUMENTATION SUMMARY

Document	Description	Produced By	Disclosure(s)
Registered Technology Provider Request (excluding Organizational Security Audit)	Information submitted by RTP to the Administrator	Registered Technology Provider	Public
Organizational Security Audit	Information submitted by RTP to the Administrator annually	Registered Technology Provider	Subscribers
Subscriber Agreement	The agreement completed by the Subscriber to be given access to sensitive information.	Subscriber	List of subscribers will be maintained on the RABET-V Portal
Product Goals	Overview of what the product is intended to do	Registered Technology Provider	Public
Expected Usage	Statements of how the product is intended to be used	Registered Technology Provider	Public
Security Claims	Claims of which requirements are met by the product	Registered Technology Provider	Subscribers
Process Descriptions	Descriptions of how the provider does product development	Registered Technology Provider	RABET-V Administrator only
Architecture Documentation and Diagrams	Documentation on how the product is constructed	Registered Technology Provider	RABET-V Administrator only
Third-Party Component Details	Listing of the 3rd party software packages used by or included in the product	Registered Technology Provider	RABET-V Administrator only
User Documentation	Documentation intended to help non-technical users use the product	Registered Technology Provider	Subscribers
Product Revision Submission Artifacts	Outputs of the providers internal product development process	Registered Technology Provider	RABET-V Administrator only
Submission Review Checklist	Checklist completed during the Submission Review activity	RABET-V Administrator, or agent	Registered Technology Provider
SAMM Toolkit and interview session notes	Interview and scoring toolkit used for the Process Assessment	RABET-V Administrator, or agent	Registered Technology Provider
Reliable Artifacts Evaluation	Produced by the Process Assessment and used for Testing Rules	RABET-V Administrator, or	Registered Technology Provider

CHAPTER

THIRTYTHREE

RABET-V GLOSSARY

1st Degree Component
A component that provides or configures one of the 10 *security service*s. Components are determined to be 1st or *2nd degree component*s in the Architecture Review.

2nd Degree Component
A component that uses one of the components which provide or configure a *security service*. Components are determined to be 1st or 2nd degree components in the Architecture Review.

Activity
A self-contained aspect of the RABET-V Program. Each activity has a process with inputs, outputs, and a workflow.

BPMN
Business Process Model and Notation
A "graphical notation that depicts the steps in a business process. BPMN depicts the end to end flow of a business process. The notation has been specifically designed to coordinate the sequence of processes and the messages that flow between different process participants in a related set of activities."[1]

Data Criticality Label
A label indicating the sensitivity of the data the component is handling. This may be thought of as a label of "integrity". This is measured by the impact of the data being manipulated to an unknown or incorrect value. Criticality can be determined by examining a component's exposed interfaces.

Data Sensitivity Label
A label indicating the sensitivity of the data the component is handling. This may be thought of as a label of "confidentiality". This is measured by the impact of the data being exposed to an unauthorized party. Sensitivity can be determined by examining a component's exposed interfaces.

Functions
A discrete piece of functionality provided by the *product*. Represented as a "*port*" in the UML Component diagram.

Port
A bundle of interfaces that provides system functionality.

Product
An election technology submitted to RABET-V.

Product Revision
A specific version of the *product* submitted to RABET-V.

RABET-V Administrator
The organization responsible for overseeing and executing the RABET-V Program. CIS is the administrator for the pilot program.

[1] https://www.bpmn.org/

RABET-V Iteration
A complete cycle through the RABET-V activities with a unique *product revision*. The first iteration is called the Initial Iteration.

RABET-V Subscriber
A state or local jurisdiction who has requested access to sensitive RABET-V reporting

Registered Technology Provider (RTP)
An organization that develops election technology and has met the minimum requirements to become a RABET-V Registered Technology Provider.

Required Security Services
Mechanisms used to provide confidentiality, integrity authentication, source authentication and/or support non-repudiation of information.

Security Control Family
A group of security services that supports the security goals. RABET-V defines ten security control families which are used to create the *Security Service Capability Maturity (SSCM)* scores and the *Security Services Architectural Maturity (SSAM)* scores.

Security Service
A capability that supports one, or many, of the security goals (NIST definition). Multiple security services (or controls) are collected in a *Security Control Family*.

Security Service Capability Maturity (SSCM)
A set of maturity scores for each of the ten *security service*s that is one of the primary metrics reported by RABET-V.

Security Service Catalog
A set of *security service*s identified by RABET-V to mitigate *threat*s.

Security Service Label
Mechanisms used to provide confidentiality, integrity authentication, source authentication and/or support non-repudiation of information.

Security Services Architectural Maturity (SSAM)
A maturity score created by the RABET-V Architecture Review activity to indicate how well the *product's* architecture is defined to provide the *security services*.

Security Services Architecture
An architectural view created in the Architecture Review which identifies components and maps them to the 10 *Security Control Families*.

Services
A system level component that provides data processing capabilities.

Software Development Maturity (SDM)
A maturity score measured by the *RABET-V Process Review* activity to indicate maturity of the provider's software assurance processes. The RABET-V SDM score is based on the OWASP Software Assurance Maturity Model (SAMM) with enhancements in the areas of usability and accessibility.

Testing rules
A set of rules specific to the technology provider and *product* which determine how changes to that product will be verified during *RABET-V iterations*.

Threat
A role of a situation that my lead to one ore more related incidents or failures.

INDEX