NIST Cybersecurity White Paper
NIST CSWP 37B ipd

Automation of the NIST Cryptographic Module Validation Program:

April 2025 Status Report

Initial Public Draft

Christopher Celi Alex Calis Murugiah Souppaya *Computer Security Division* *Information Technology Laboratory* William Barker *Strativia LLC* Karen Scarfone *Scarfone Cybersecurity* Shawn Geddis *Katalyst*	Raoul Gabiam *The MITRE Corporation* Stephan Mueller Yi Mao *atsec information security* Barry Fussell Andrew Karcher *Cisco* Douglas Boldt *Amazon Web Services*

This publication is available free of charge from:
https://doi.org/10.6028/NIST.CSWP.37B.ipd

September 10, 2025

How to Cite this NIST Technical Series Publication:
Celi C, Souppaya M, Barker W, Scarfone K, Geddis S, Gabiam R, Mueller S, Mao Y, Fussell B, Karcher A, Boldt D (2025) Automation of the NIST Cryptographic Module Validation Program: April 2025 Status Report. (National Institute of Standards and Technology, Gaithersburg, MD), NIST Cybersecurity White Paper (CSWP) NIST CSWP 37B.ipd. https://doi.org/10.6028/NIST.CSWP.37b.ipd.

Author ORCID iDs
Chris Celi: 0000-0001-9979-6819
Alex Calis: 0000-0003-1937-8129
Murugiah Souppaya: 0000-0002-8055-8527
William Barker: 0000-0002-4113-8861
Karen Scarfone: 0000-0001-6334-9486
Raoul Gabiam: 0009-0000-7458-8028

Public Comment Period
September 10, 2025 - October 10, 2025

Submit Comments
applied-crypto-testing@nist.gov

National Institute of Standards and Technology
Attn: Computer Security Division, Information Technology Laboratory
100 Bureau Drive (Mail Stop 8930) Gaithersburg, MD 20899-8930

Additional Information
Additional information about this publication is available at https://csrc.nist.gov/publications/cswp, including related content, potential updates, and document history.

All comments are subject to release under the Freedom of Information Act (FOIA).

Abstract

The Cryptographic Module Validation Program (CMVP) validates third-party assertions that cryptographic module implementations satisfy the requirements of Federal Information Processing Standards (FIPS) Publication 140-3, Security Requirements for Cryptographic Modules. The current cryptographic module validation process is heavily manual, out of sync with the speed of technology industry development and deployment. Thus, the NIST National Cybersecurity Center of Excellence (NCCoE) has undertaken the Automated Cryptographic Module Validation Project (ACMVP) to support improvement in the efficiency and timeliness of CMVP operations and processes. The goal is to demonstrate a suite of automated tools that have the potential to make the FIPS 140-3 validation process more efficient and provide higher assurances that test findings reported for modules meet FIPS 140-3 requirements.

This report is the second status report for the project, which describes progress made between September 2024 and April 2025 and planned next steps. A prior update of work accomplished can be found in the September 2024 status report. This document outlines progress across each of the three workstreams: the Test Evidence (TE) Workstream, the Protocol Workstream, and the Research Infrastructure Workstream, each a focused effort in its own right. The combined impact of these workstreams intends to result in improvements to the overall automation of the CMVP.

Audience

The primary audience for this report is technology, security, and privacy program managers, architects, software developers, engineers, and IT professionals involved with the CMVP, and accredited cryptography and security testing labs, and conformance offices at companies that produce security software and hardware.

Keywords

Automated Cryptographic Module Validation Project (ACMVP); Cryptographic Module Validation Program (CMVP); cryptography; cryptographic module; cryptographic module testing; cryptographic module validation.

Collaborators

Collaborators participating in this project submitted their capabilities in response to an open call in the Federal Register for all sources of relevant security capabilities from academia and industry (vendors and integrators). The following respondents with relevant capabilities or product components signed a Cooperative Research and Development Agreement (CRADA) to collaborate with NIST in a consortium to build this example solution.

- Acumen Security
- AEGISOLVE

- Apple
- atsec
- AWS
- Cisco
- Katalyst
- Lightship Security
- Microsoft
- NXP Semiconductors
- SUSE

Certain commercial entities, equipment, products, or materials may be identified by name or company logo or other insignia in order to acknowledge their participation in this collaboration or to describe an experimental procedure or concept adequately. Such identification is not intended to imply special status or relationship with NIST or recommendation or endorsement by NIST or NCCoE and neither is it intended to imply that the entities, equipment, products, or materials are necessarily the best available for the purpose.

Acknowledgements

Contributors to each workstream are listed in the corresponding sections below. Additionally, the following people and organizations contributed to the project outside of a workstream: Courtney Maatta, Rochelle Casey, Alicia Squires, Margaret Salter, Tim Ness, Damian Zell, Derrick Williams III, Jeff Wright, Mickey Iqbal, and David Browning of Amazon; Dave Hawes, Gavin O'Brien, Tim Hall, Matt Scholl, Cherilyn Pascoe, Jim St. Pierre, Kevin Stine, Ann Rickerds, Shawn Winhoven, Jeffrey J McIntyre, Anil Das, Edgar Garay, Jim Simmons, Robert Staples, Rob Densock, and Blair Heiserman of NIST; Jason Arnold of HII; Heather Flanagan of Spherical Cow Consulting.

The project team recognizes and appreciates Apostol Vassilev of NIST for leading the project at the inception and kicking off the project and formulating the three workstreams and associated activities.

Table of Contents

List of Tables

List of Figures

1. Overview

This section summarizes some of the challenges faced by the Cryptographic Module Validation Program (CMVP) and describes the efforts at the NCCoE to address those challenges. It highlights the status thus far across three workstreams' activities and associated achievements to streamline the processes to increase efficiency.

1.1. Challenge

The CMVP validates third-party assertions that cryptographic module implementations satisfy the requirements of Federal Information Processing Standards (FIPS) Publication 140-3, Security Requirements for Cryptographic Modules [1]. Under the CMVP, cryptographic modules undergo third-party testing by National Voluntary Laboratory Accreditation Program (NVLAP) accredited laboratories, and the processes and results are validated under a program run by the National Institute of Standards and Technology (NIST) and the Canadian Centre for Cyber Security (CCCS). Current industry cryptographic product development, production, and maintenance processes place significant emphasis on time-to-market efficiency. A number of elements of the validation process are manual in nature, and the period required for third-party testing and government validation of cryptographic modules is often incompatible with industry requirements.

1.2. Solution

The NIST National Cybersecurity Center of Excellence (NCCoE) in collaboration with the CMVP has undertaken a project to demonstrate the value and practicality of automation support to improve the responsiveness of CMVP. The intent of the Automated Cryptographic Module Validation Project (ACMVP) is to support improvement in the efficiency and timeliness of CMVP [2] operations and processes. This NCCoE effort is one of many focused on the automation of module validation and report review flow and follows the successful completion of NIST efforts such as the automation of the Cryptographic Algorithm Validation Program (CAVP); the rollout of Web CRYPTIK, an application for submitting test results to the CMVP; and the automation of entropy data testing evidence processing for the Entropy Source Validation (ESV) program. The initiative will provide mechanisms for structural presentation of testing evidence by NVLAP-accredited parties to facilitate the automation of evidence validation by the CMVP.

The ACMVP's goal is to enable automated test report review where feasible for each of the test requirements found in FIPS 140-3 and International Organization for Standardization (ISO)/International Electrotechnical Commission (IEC) 24759 [3], which FIPS 140-3 incorporates by reference. Because of the wide range of the technologies and corresponding security requirements that the CMVP covers, this effort is being executed in phases. The initial phase of software module validation, such as an OpenSSL module, is foundational and will determine future phases.

The module testing and reporting aspects of module validation, according to ISO/IEC 24759, combine functional and nonfunctional security requirements. This project attempts to streamline the test methods for the functional tests of specific classes of technologies (e.g., software modules) and corresponding reporting of functional and non-functional security requirements. The team is working to demonstrate a suite of tools to modernize and automate manual review processes in support of existing policy and efforts to include technical testing under the CMVP, which employs an NVLAP-accredited testing concept that permits organizations to test their cryptographic products according to the FIPS 140-3 requirements and then directly report the results to NIST using appropriate protocols.

The accredited parties will have to identify the corresponding personnel and organizational structures needed to perform this testing while complying with the laboratory requirements for testing programs established by NVLAP under NIST Handbook (HB) 150-17. The accreditation requirements in HB 150-17 are both hierarchical and compositional in nature so that organizations can tailor the scope of accreditation according to their specific product/service portfolio.

The project is divided into three workstreams: the Test Evidence (TE) Workstream, the Protocol Workstream, and the Research Infrastructure Workstream, each a focused effort in its own right. The combined impact of these workstreams will result in improvements to the overall automation of the CMVP.

1.3. Progress to Date

This update covers progress in the project from September 2024 to April 2025. Due to the shift in the International Cryptographic Module Conference (ICMC) schedule, only six months passed between ICMC 2024 and ICMC 2025.

To date, the ACMVP project has:

1. Identified and classified categories of test evidence required for CMVP validation that can readily be automated in a reporting format that is consistent with current Web CRYPTIK and CMVP and identified the test evidence classes where manual processes are still needed;
2. Identified necessary schemas and protocols for evidence submission and validation for a scalable application programming interface (API) based architecture;
3. Designed and developed a cloud native infrastructure required to support validation program automation.

The ACMVP project team accomplished the following across the three workstreams:

Test Evidence Workstream

- Defined test methods for functional testing TEs to allow for more specific information and automation to be applied to the evidence collected

- Improved TE filtering coverage via thorough review of all sections of FIPS 140-3

Protocol Workstream

- Added an automated rule processing on submissions with instant feedback intended to catch inconsistencies and inaccuracies a CMVP reviewer would otherwise need to catch during their review of a submission and instantly provide feedback to the submitter, which needs to be corrected before the submission is accepted
- Added the source code evidence payloads to capture how source code TEs are evaluated by the lab
- Fleshed out the protocol to provide a more complete API for labs to interact with their submissions

Research Infrastructure Workstream

- **Tools Researched:**
    - Amazon API Gateway, Amazon Elastic Container Registry (ECR), Amazon Relational Database Service (RDS) for Structured Query Language (SQL) Server, AWS Application Load Balancer (ALB), AWS Database Migration Service (DMS), AWS CodeBuild, AWS CodeDeploy, AWS CodePipeline, Amazon ECS, Amazon EC2, Elastic Container Service (ECS) Fargate, Elastic Kubernetes Service (EKS) Auto Mode, Amazon Simple Storage Service (S3), GitHub, Linux Containers, Microsoft Windows Containers, Nginx Reverse Proxy
- **Outcomes**
    - Migrated legacy databases to a managed and scalable cloud platform
    - Automated builds, testing, and deployments through a CI/CD pipeline
    - Containerized core applications for faster deployments and improved maintainability
    - Replaced legacy web servers with scalable, cloud-based routing and authentication
    - Enabled secure, flexible authentication using mutual TLS and API keys
    - Reduced deployment downtime and improved system resilience
    - Streamlined developer workflows and accelerated update cycles
    - Lowered operational complexity and infrastructure overhead
    - Deployed a demo ACMVP server, enabling the community to explore and get acquainted with the newly developed application

2. Test Evidence Workstream

The structured application of test evidence filtering proposed by the Test Evidence (TE) Workstream plays a crucial role in streamlining the validation process for cryptographic modules under FIPS 140-3. By leveraging both basic and supplemental filters, the evaluation process ensures that only relevant test evidence is considered, reducing redundancy while maintaining rigorous security standards. This approach enhances efficiency, supports automation, and enables a more scalable validation framework. As the TE Workstream continues refining these methodologies, integrating well-defined filtering criteria will further strengthen the CMVP, improving consistency and accuracy in compliance assessments.

The September 2024 Status Report summarizes the NCCoE ACMVP project, including the deliverables from the TE Workstream. Since the publication of that report, the TE Workstream has been working to complete:

- Test methods for functional testing TEs
- Improvement of TE filtering coverage

The ACMVP TE Workstream is led by Yi Mao of atsec and Shawn Geddis of Katalyst under the NCCoE ACMVP leadership of Murugiah Souppaya and Chris Celi of NIST. The workstream is in debt to the invaluable contributions from Alex Calis of NIST CMVP. The workstream benefited from contributions from the atsec team members including Stephan Mueller, Walker Riley, and Swapneela Unkule; the Intertek Acumen Security team led by James Reardon with Chris Bell, Sowndar Gillan Gopi, and Rutwij Kulkarni; the AEGISOLVE team members including Travis Spann, Javier Martel, Mike McCarl, and Debbie Harrington; Ryan Thomas of Lightship Security; Barry Fussell and Andrew Karcher of Cisco; Alicia Squires and Courtney Maatta of Amazon; Marc Ireland of NXP; Mike Grimm of Microsoft; Ivan Teblin and Blaine Stone of SUSE; and Michael Dimond of the MITRE Corporation.

2.1. Test Methods for Functional Testing TEs

The diverse set of cryptographic modules and their varying restrictive operating environments can create challenges in choosing the right approach and selecting an appropriate toolset to capture the evaluation TE. The CMVP provides some limited guidance, but it is necessary to identify which test methods are relevant to the granularity of individual TEs.

2.1.1. Testing Access

Accessing the operational environment for effective testing of a cryptographic module is a persistent challenge and allowances for various methodologies to follow for accommodating these challenges exist. For any given evaluation, it is assumed by default that the Testing Access used for all TEs is the same; however, any given TE might in fact require an alternate allowed Testing Access method to be used.

The Testing Access methods are as follows:

- **Physical:** Testing a module directly by lab personnel within a controlled lab environment

- **Remote:** Testing a module remotely while obtaining the equivalent assurance as if the test were performed at the vendor's facility
- **Observed:** Testing a module by vendor personnel within a controlled lab environment while lab personnel observe the triggering and responses of the module

2.1.2. Selection Criteria

The current challenge is to assign only the appropriate test methods to each of the identified TEs. Drawing from CMVP, lab, and original vendor expertise, the criteria can be used to refine the test methods to be used for each TE.

Test methods are the defined techniques that can be utilized while ensuring confidence of capturing actual module operation under real-world conditions and enabling efficient evidence gathering workflow. Only a limited set of test method categories exist for the team to focus on in their pursuit, which can best be described as:

- **Debugger:** The ability to run or halt the target program using breakpoints, step through code line by line, and display or modify the contents of memory, CPU registers, and stack frames
- **Simulation:** Imitations of the functioning of one system or process by means of the functioning of another
- **Emulation:** Hardware or software that permits programs written for one environment to be run unaltered on another environment
- **Harness:** Hardware or software that manipulates an operating environment with the purpose of triggering events and capturing the corresponding responses or results.
- **Manual:** Action(s) by a user to perform a set of designated steps for the purpose of triggering events and capturing the corresponding responses or results.
- **Other:** Due to the diversity and complexity of operating environments, the toolset needed to perform the gathering of relevant TE may not fit precisely within the above five test methods, which warrants the need for a catch-all method that enables the tester to comprehensively describe the methodology used to capture the TE.

2.1.2.1. Debugger

No clearly articulated interpretation of when and how a debugger can and should be used is available as much of what is known comes from lab empirical evidence.

2.1.2.2. Simulation and/or Emulation

Drawing from guidance currently provided by CMVP in the Management Manual, Version 2.3 [4], labs may apply emulators or simulators, depending on the type of testing results to be achieved. The three broad areas of focus during the testing of a cryptographic module are operational testing of the module at the module's defined boundary, operational fault induction testing, and algorithm testing.

1. **Operational Testing** – Emulation or simulation is prohibited for the operational testing of a cryptographic module. Actual testing of the cryptographic module must be performed utilizing the defined ports and interfaces and services that a module provides. A test harness or a modified version to induce an error may be utilized; however, no changes to code or circuitry responsible for the tested response may be made.
2. **Operational Fault Induction Testing** – An emulator or simulator may be utilized for fault induction to test a cryptographic module's transition to error states as a complement to the source code review. Rationale must be provided for the applicable TE as to why a method does not exist to induce the actual module into the error state for testing.
3. **Algorithm Testing** – Algorithm testing utilizing the defined ports and interfaces and services that a module provides is the preferred method, as it most clearly meets the requirements of FIPS 140-3 Implementation Guidance (IG) 2.3.A. If this preferred method is not possible where the module's defined set of ports, interfaces, and services do not allow access to internal algorithmic engines, two alternative methods may be utilized:
    a. A module may be modified under the supervision of the Cryptographic and Security Testing Laboratory (CSTL) for testing purposes to allow access to the algorithmic engines (e.g., test jig, test API), or
    b. A module simulator may be utilized.

2.1.2.3. Harness

No clearly articulated interpretation of when and how a test harness can and should be used is available as much of what is known comes from experienced vendors that developed specialized test harnesses around their respective modules and within the restricted operating environments.

2.1.2.4. Manual

No clearly articulated interpretation of when and how a manual process can and should be used is available as much of what is known comes from the need for human interaction to trigger events or an inability to trigger the steps in an automated approach.

2.1.2.5. Other

As noted earlier, due to the diversity and complexity of operating environments, the toolset needed to perform the gathering of relevant TE may not fit precisely within the above five test methods. Therefore, a need for a catch-all method that enables the tester to comprehensively describe the methodology used to capture the TE exists.

2.1.3. Test Methods Allowed

Table 1 maps the allowed test methods to the grouping of associated TEs for purpose of condensing the resulting table.

Table 1. Allowed Test Methods

TE (TE##.##.##)	Debugger	Simulator	Emulator	Harness	Manual	Other
02.12.01	X	X	X	X	√	√
02.13.03	X	X	X	√	X	√
02.15.03	X	X	X	X	√	√
02.15.05, 02.16.04, 02.17.04	√	X	X	X	√	√
02.16.02, 02.17.02	X	X	X	√	X	√
02.19.02	√	X	X	√	√	√
02.22.02	√	X	X	√	X	√
02.24.02	√	X	X	√	√	√
02.26.03, 02.26.04, 02.26.05, 02.28.01, 02.28.02, 02.30.02	√	X	X	√	X	√
03.01.04, 03.02.01, 03.14.03, 03.15.03, 03.15.04, 03.15.06	√	X	X	√	√	√
03.05.01, 03.05.02	√	X	X	√	√	√
03.06.01, 03.06.02, 03.07.01, 03.07.02, 03.07.04, 03.07.08	√	X	X	√	√	√
03.08.01, 03.08.02	√	√	X	√	√	√
03.09.02, 03.10.02, 03.10.04	√	√	X	√	√	√
03.11.01, 03.11.03	√	X	X	√	√	√
03.13.02	X	X	X	X	√	√

TE (TE##.##.##)	Debugger	Simulator	Emulator	Harness	Manual	Other
03.18.02, 03.19.02, 03.19.04, 03.20.01, 03.21.01	√	X	X	√	√	√
03.22.01	√	X	X	√	√	√
04.02.02, 04.02.03	√	X	X	√	√	√
04.07.03	√	X	X	√	√	√
04.11.02	√	X	X	√	√	√
04.13.01, 04.13.02, 04.13.03	√	√	√	√	√	√
04.14.02	√	X	X	√	√	√
04.15.01	√	X	X	√	√	√
04.18.01, 04.19.02, 04.19.03, 04.20.01, 04.20.03, 04.21.02, 04.22.02	√	X	X	√	√	√
04.23.01, 04.25.01, 04.25.02, 04.25.03	√	X	X	√	√	√
04.28.01, 04.29.01, 04.32.01, 04.33.01, 04.34.01, 04.35.02, 05.13.08	√	√	√	√	√	√
04.37.02, 04.38.02	√	X	X	√	√	√
04.39.02, 04.39.03, 04.39.04, 04.42.03, 04.42.04	√	X	X	√	√	√
04.43.02, 04.44.02	√	X	X	√	√	√

TE (TE##.##.##)	Debugger	Simulator	Emulator	Harness	Manual	Other
04.45.02, 04.45.03, 04.47.01, 04.48.01, 04.52.01, 04.54.02, 04.54.03, 04.55.02	√	X	X	√	√	√
04.53.01	√	√	√	√	√	√
04.56.02	√	X	X	√	√	√
04.59.01	√	X	X	√	√	√
05.05.05	√	√	√	√	√	√
05.05.07, 05.06.06, 05.08.01, 05.08.02, 05.11.01, 05.11.02, 05.12.02, 05.13.03, 05.13.04, 05.13.05	√	X	X	√	√	√
05.06.02	√	√	√	√	√	√
05.06.03	√	X	X	√	√	√
05.06.04	√	X	X	√	√	√
05.13.01, 05.13.02	√	X	X	√	√	√
05.13.06	√	X	X	√	√	√
05.15.01, 05.15.02, 05.16.03, 05.17.02	√	X	X	√	√	√
05.20.01	√	√	√	√	√	√
05.23.01	√	√	√	√	√	√
06.05.01, 06.05.02, 06.05.03, 06.06.01, 06.06.02, 06.08.01, 06.08.03	√	√	√	√	√	√
06.06.02, 06.08.03	√	√	√	√	√	√

TE (TE##.##.##)	Debugger	Simulator	Emulator	Harness	Manual	Other
09.01.02, 09.01.03, 09.02.02, 09.03.02, 09.03.03, 09.14.02, 09.16.03, 09.25.02, 09.27.02	√	X	X	√	√	√
09.21.02, 09.21.03, 09.21.04, 09.22.01	√	X	X	√	√	√
09.24.02	√	X	X	√	√	√
09.28.02, 09.28.03, 09.28.04	√	X	X	√	√	√
09.33.02	√	X	X	√	√	√
09.36.02, 09.37.02	√	X	X	√	√	√
10.07.03, 10.08.03, 10.09.03, 10.10.01, 10.10.02, 10.28.02	√	X	X	√	√	√
10.07.04	√	X	X	√	√	√
10.25.02, 10.27.01	√	X	X	√	√	√
10.35.04	√	√	X	√	√	√
10.53.02, 10.53.03	√	X	X	√	√	√
11.08.06, 11.08.09, 11.11.01	√	X	X	√	√	√
11.13.02	√	X	X	√	√	√
11.28.02, 11.28.03, 11.28.04	√	√	√	√	√	√
11.32.02	√	X	X	√	√	√

2.2. Improvement of TE Filtering Coverage

TE filters serve as a pivotal mechanism to streamline the classification and evaluation of TE, ensuring that only relevant and applicable tests are conducted based on specific module characteristics. A proper set of applicable TEs tailored by a given module specification refines the required assessments and optimizes the validation process.

With the growing complexity of cryptographic modules and the need for efficient validation, TE filters are designed to:

- Target specific needs through focusing on applicable tests by narrowing down evidence requirements based on module attributes such as type, security level, and operational environment
- Reduce redundancy through minimizing repetitive validation steps by filtering out TEs that are not relevant to a given module's configuration or features
- Enhance automation through supporting automated workflows by integrating filters into structured JSON schemas, aligning with automation tools like Web-Cryptik

This document delves into the methodologies and criteria for applying TE filters, the implementation of filtering mechanisms, and their role in achieving a more efficient and scalable CMVP. By leveraging these filters, vendors and validators can focus on precise compliance requirements, reducing manual overhead while maintaining robust security standards.

Table 2 is excerpted from ISO/IEC 19790:2012 (2014) [5], which is the base of FIPS 140-3 and provides a structured summary of the FIPS 140-3 security requirements across various requirement areas. It outlines the security levels applicable to each category, specifying the testing expectations and security assurances needed to meet compliance. The table serves as a reference for understanding how different cryptographic module components must align with FIPS 140-3 standards, ensuring consistent evaluation and validation. Each requirement area focuses on distinct security aspects, such as module specifications, authentication mechanisms, physical security, and lifecycle assurance, enabling a comprehensive approach to cryptographic module validation.

Table 2. Summary of FIPS 140-3 Security Requirements

Requirement Area		**FIPS 140-3 Security Level**			
		1	**2**	**3**	**4**
1	**General**	No security testing requirements (i.e. no TEs)			
2	**Cryptographic Module Specification**	Specification of cryptographic module, cryptographic boundary, approved security functions, and normal and degraded modes of operation, and description of cryptographic module including all hardware, software and firmware components, which provide status information to indicate when the service utilizes an approved cryptographic algorithm, security function, or process in an approved manner			
3	**Cryptographic Module Interfaces**	Required and optional interfaces and specification of all interfaces and of all input and output data paths		Trusted channel	

<table>
<tr><th colspan="2" rowspan="2">Requirement Area</th><th colspan="4">FIPS 140-3 Security Level</th></tr>
<tr><th>1</th><th>2</th><th>3</th><th>4</th></tr>
<tr><td>4</td><td>Roles, Services and Authentication</td><td>Logical separation of required and optional roles and services</td><td>Role-based or identity-based operator authentication</td><td>Identity-based operator authentication</td><td>Multi-factor authentication</td></tr>
<tr><td>5</td><td>Software / Firmware Security</td><td>Approved integrity technique, defined SFMI, HFMI and HSMI, and executable code</td><td>Approved digital signature or keyed message authentication code-based integrity test</td><td colspan="2">Approved digital signature-based integrity test</td></tr>
<tr><td>6</td><td>Operational Environment</td><td>Non-modifiable, limited, or modifiable control of SSPs</td><td>Modifiable, role-based, or discretionary access control, and audit mechanism</td><td colspan="2"></td></tr>
<tr><td>7</td><td>Physical Security</td><td>Production-grade components</td><td>Tamper evidence and opaque covering or enclosure</td><td>Tamper detection and response for covers and doors, strong enclosure or coating, and protection from direct probing EFP or EFT</td><td>Tamper detection and response envelope, EFP, and fault injection mitigation</td></tr>
<tr><td rowspan="2">8</td><td rowspan="2">Non-Invasive Security</td><td colspan="4">Module is designed to mitigate against non-invasive attacks specified in Annex “F”.</td></tr>
<tr><td colspan="2">Documentation and effectiveness of mitigation techniques specified in Annex “F”</td><td>Mitigation testing</td><td>Mitigation testing</td></tr>
<tr><td rowspan="3">9</td><td rowspan="3">Security Parameter Management</td><td colspan="4">Random bit generators, SSP generation, establishment, entry & output, storage & zeroization</td></tr>
<tr><td colspan="4">Automated SSP transport or SSP agreement using approved methods</td></tr>
<tr><td colspan="2">Manually established SSPs may be entered or output in plaintext form.</td><td colspan="2">Manually established SSPs may be entered or output in either encrypted form, via a trusted channel or using split knowledge procedures.</td></tr>
<tr><td rowspan="2">10</td><td rowspan="2">Self-Tests</td><td colspan="4">Pre-operational: software/firmware integrity, bypass, and critical functions test</td></tr>
<tr><td colspan="4">Conditional: cryptographic algorithm, pair-wise consistency, SW/FW loading, manual entry, conditional bypass & critical functions test</td></tr>
<tr><td rowspan="2">11</td><td colspan="5">Life-Cycle Assurance</td></tr>
<tr><td>Configuration Management</td><td colspan="2">Configuration management system for cryptographic module, components, and documentation, each uniquely identified and tracked throughout lifecycle</td><td colspan="2">Automated configuration management system</td></tr>
</table>

	Requirement Area	FIPS 140-3 Security Level			
		1	2	3	4
	Design	Module designed to allow testing of all provided security related services			
	FSM	Finite State Model			
	Development	Annotated source code, schematics or HDL	Software high-level language, and hardware high-level descriptive language		Documentation annotated with pre-conditions upon entry into module components and postconditions expected to be true when components is completed.
	Testing	Functional testing		Low-level testing	
	Delivery & Operation	Initialization procedures	Delivery procedures		Operator authentication using vendor provided authentication information
	Guidance	Administrator and non-administrator guidance			
12	**Mitigation of Other Attacks**	Specification of mitigation of attacks for which no testable requirements are currently available			Specification of mitigation of attacks with testable requirements

Building on the summary of FIPS 140-3 security requirements in Table 2, Table 3 provides a more granular analysis of the number of security requirements per ISO/IEC 24759:2014(2015), which is a companion document to ISO/IEC 19790 specifying the derived test requirements, across different implementation areas. This table categorizes security requirements based on the module's type being Software (SW), Firmware (FW), Hardware (HW), SW-HW hybrid (SW-H), or FW-HW hybrid (FW-H), and further differentiates them by security levels. The breakdown facilitates a clearer understanding of the distribution of TE requirements, highlighting how various module implementations align with compliance expectations at each level.

The number of total TEs and percentage of applicable TEs will indicate how many TEs are not applicable. By filtering out these non-applicable TEs with public consensus, the CSTL can more directly perform the required testing.

409

Table 3. An overview of the number of Security Requirements

Area	Total TEs	Security Level 1					Security Level 2					Security Level 3					Security Level 4				
		SW	FW	HW	SW-H	FW-H	SW	FW	HW	SW-H	FW-H	SW	FW	HW	SW-H	FW-H	SW	FW	HW	SW-H	FW-H
2	65	40	45	49	55	60	40	45	49	55	60	40	45	49	55	60	40	45	49	55	60
3	53	41	43	43	43	43	41	43	43	43	43	46	48	52	52	52	47	49	53	53	53
4	74	45	45	45	45	45	63	63	63	63	63	70	70	70	70	70	71	71	71	71	71
5	39	23	23	23	30	30	30	30	29	37	37	32	32	30	39	39	32	32	30	39	39
6	50	10	10	10	10	10	50	50	50	50	50	0	0	0	0	0	0	0	0	0	0
7	82	0	14	14	14	14	0	27	27	27	27	0	69	69	69	69	0	78	78	78	78
8	5	3	3	3	3	3	3	3	3	3	3	4	4	4	4	4	4	4	4	4	4
9	63	44	43	43	44	43	48	47	47	48	47	56	56	56	56	56	57	57	57	57	57
10	74	68	68	68	68	68	68	68	68	68	68	74	74	74	74	74	74	74	74	74	74
11	52	36	36	35	38	38	41	41	41	44	44	44	44	44	47	47	49	49	49	52	52
12	5	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	5	5	5	5	5
A	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
B	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4	4
Total TEs	567	317	337	340	357	361	391	424	427	445	449	373	449	455	473	478	384	469	475	493	498
% Appli-cable	100	56	59	60	63	64	69	75	75	78	79	66	79	80	83	84	68	83	84	87	88

We recognize that software implementations only support levels 1 and 2. However:

- The Area 2 TEs include requirements from security level 1 through level 4, which are listed in Table 4. This area's requirements are about Cryptographic Module Specification and are the same for all four security levels. The unified area 2 requirements are reflected by the numbers of TEs in the red rectangle boxes on Table 3.
- The Area 7 TEs include requirements from security level 1 through level 4, which are listed in Table 5. The Physical Security requirements in Area 7 are incremental for cryptographic modules from a low security level to a higher level. The numbers of TEs in the green rectangle boxes on Table 3 illustrate this trend.

Table 4 and Table 5 in Section 2.2.2 serve as examples of how the basic TE filters work by listing all applicable TEs and non-applicable TEs for a given type of module at any possible security level. A complete set of TE tables elaborating on Table 3 is provided in Appendix B of this status report.

2.2.1. TE Filtering Criteria

The TE Filtering criteria consists of the Module Information and Supplemental Information from the Web-Cryptik as the base. The CMVP provided Module Supplemental Information (V3.0.0 as of 2024-09-04) but is not currently used to tailor the set of TEs to fit the module under test.

In the CMVP's Module Supplemental Information (MSI) document, most Supplemental Information questions map to the security assertions (AS), test requirement (TE), implementation guidance (IG), and security policy (SP), but a few questions are not mapped to any of these and are left blank. The list below reflects the CMVP's current MSI document. The TE Workstream provides a complete mapping of MSI questions to relevant TEs in Table 6.

By reviewing all TEs contained in the WebCryptik Br1 v1.0.6, the TE Workstream completed the list of criteria, including the basic filters and supplemental filters, as the following:

- **Basic Filters**
  - Module Embodiment: Single Chip, Multi-Chip Embedded, Multi-Chip Standalone
  - Module Type: Software, Hardware, Firmware, Software-hybrid, Firmware-hybrid
  - Operational Environment: modifiable, limited, non-modifiable
  - Section Level: Per Table 2, area 6 is not applicable to Level 3 and Level 4
- **Supplemental Filters**
  - **Cryptographic module specification**
    - Does the module implement OTAR? – IG D.C
    - Does the module have a non-approved mode? – IG 2.4.A
    - Does the module require initialization steps to operate in the approved mode? – Certificate Caveat and SP

- Does the module have excluded components? – AS02.13, AS02.14
- Does the module allow a degraded mode of operation? – AS02.25
- Does the module have an implementation of PPA or PAI? – IG 2.3.C
- Does the module contain an embedded or have a bound cryptographic module? – IG 2.3.A
- Does the module have any critical functions? – AS10.16, AS10.23, AS10.24, AS10.52
- Is the module a sub-chip implementation? – IG 2.3.B
- Does the module's approved mode make use of any non-approved algorithm? – IG 2.4.A
- Does the module have a non-compliant state?

- **Cryptographic module interfaces**
  - Does the module receive any of its input from an external input device? – TE03.05.02, TE03.06.02, TE03.08.02, TE03.11.02
  - Does the module provide any of its output through an external output device? – TE03.05.02, TE03.06.02, TE03.08.02, TE03.11.02
  - Does the module implement a Trusted Channel? – IG 3.4.A
  - Is there a control output interface? – AS03.09, AS03.10
- **Roles, services, and authentication**
  - Does the module support concurrent operators? – AS04.02
  - Does the module support any authentication mechanism? – AS04.43-AS04.55
  - Does the module use identity-based authentication?
  - Does the module support role-based authentication?
  - Does the module support multi-factor-based authentication? – AS03.22
  - Does the module have a bypass capability? – AS04.22, AS10.21-AS10.22, AS10.47-AS10.51
  - Is there a maintenance role? – AS04.07
  - Is there a user role? – AS04.06
  - Can operators change roles? – AS04.38, AS04.42
  - Does the module support self-initiated cryptographic output? – AS04.23-AS04.26
  - Is default information used for first-time authentication? – AS04.46

  - Does the module support software/firmware loading? – AS04.28-AS04.33, AS05.13
  - Is a complete image replacement supported within software/firmware loading? – AS04.33-AS04.35
- **Software/firmware security**
  - Does the module use a hash or MAC to verify the integrity of its software/firmware? – TE05.05.03
  - Does the module use a digital signature to verify the integrity of its software/firmware? – TE05.05.04
  - Does the module use an EDC for the software/firmware components of a hardware module? – AS05.06
  - Does the module contain any non-reconfigurable memory? – IG 5.A
  - Does the module utilize open-source software? – Annex B
- **Operational environment**
  - None
- **Physical security**
  - Is there a maintenance access interface? – AS07.11-AS07.13, TE11.08.07
  - Are there any ventilation holes or slits? – AS07.20, AS07.25
  - Are there any removable covers/doors? – AS07.22, TE07.39.02, TE07.39.05, AS07.47, TE07.51.02, TE07.51.07, TE07.51.08, AS07.62, TE07.65.02, TE07.65.07, TE07.65.08
  - Are there tamper seals? – IG 7.3.A
  - Are there tamper seals applied by the module user?
  - Does the module implement EFP or EFT mechanisms?
- **Non-invasive security**
  - None
- **Sensitive security parameters management**
  - Does the module support input and/or output of SSPs or other sensitive data? – AS09.13, AS09.18, AS09.19
    - Are there plaintext keys, CSPs, or sensitive data output? – AS09.16-AS09.17
    - Does the module support manual/direct entry of SSPs? AS09.15, AS10.42-AS10.46, TE10.46.04
  - Is split knowledge utilized? – AS09.21, AS09.22, AS09.23

   - Is one-time programmable (OTP) memory used in the module? – IG 9.7.A
- **Self-tests**
   - None
- **Life-cycle assurance**
   - Are there any CVEs related to this module? – IG 11.A
- **Mitigation of other attacks**
   - Is the module designed to mitigate other attacks?
- **Approved security functions**
   - Are any non-NIST curves used? – IG C.A

2.2.2. TEs Impacted by Basic TE Filters

To ensure a structured approach to TE filtering, it is necessary to categorize TEs based on the security level and module type. Table 4 presents a detailed breakdown of the TEs applicable to different security levels for software modules, illustrating how filtering criteria refine the validation scope. By segmenting TEs according to security requirements, this table helps streamline the testing process, ensuring that only the relevant test evidence is considered for a given module configuration, which enhances efficiency while maintaining rigorous security standards.

The team recognizes that software implementations only support levels 1 and 2. However, Table 4 lists the Area 2 Cryptographic Module Specification TEs required from security level 1 through level 4, and Table 5 lists the Area 7 Physical Security TEs for all four security levels.

Table 4. Area 2 TEs Filtered by Security Level for Software Modules

Sec Lvl	Applicable TEs	Non-Applicable TEs	TEs N/A due to Module Type
1	TE02.03.01, TE02.03.02, TE02.07.01, TE02.07.02, TE02.09.01, TE02.10.01, TE02.10.02, TE02.11.01, TE02.11.02, TE02.12.01, TE02.13.01, TE02.13.02, TE02.13.03, TE02.14.01, TE02.16.01, TE02.16.02, TE02.16.03, TE02.16.04, TE02.16.05, TE02.19.01, TE02.19.02, TE02.20.01, TE02.20.02, TE02.20.03, TE02.20.04, TE02.21.01, TE02.21.02, TE02.22.01, TE02.22.02, TE02.24.01, TE02.24.02, TE02.26.01, TE02.26.02, TE02.26.03, TE02.26.04, TE02.26.05, TE02.28.01, TE02.28.02, TE02.30.01, TE02.30.02	TE02.15.01, TE02.15.02, TE02.15.03, TE02.15.04, TE02.15.05, TE02.15.06, TE02.15.07, TE02.15.08, TE02.15.09, TE02.15.10, TE02.15.11, TE02.15.12, TE02.15.13, TE02.15.14, TE02.17.01, TE02.17.02, TE02.17.03, TE02.17.04, TE02.17.05, TE02.17.06, TE02.17.07, TE02.17.08, TE02.17.09, TE02.17.10, TE02.18.01	TE02.15.01, TE02.15.02, TE02.15.03, TE02.15.04, TE02.15.05, TE02.15.06, TE02.15.07, TE02.15.08, TE02.15.09, TE02.15.10, TE02.15.11, TE02.15.12, TE02.15.13, TE02.15.14, TE02.17.01, TE02.17.02, TE02.17.03, TE02.17.04, TE02.17.05, TE02.17.06, TE02.17.07, TE02.17.08, TE02.17.09, TE02.17.10, TE02.18.01

Sec Lvl	Applicable TEs	Non-Applicable TEs	TEs N/A due to Module Type
2	TE02.03.01, TE02.03.02, TE02.07.01, TE02.07.02, TE02.09.01, TE02.10.01, TE02.10.02, TE02.11.01, TE02.11.02, TE02.12.01, TE02.13.01, TE02.13.02, TE02.13.03, TE02.14.01, TE02.16.01, TE02.16.02, TE02.16.03, TE02.16.04, TE02.16.05, TE02.19.01, TE02.19.02, TE02.20.01, TE02.20.02, TE02.20.03, TE02.20.04, TE02.21.01, TE02.21.02, TE02.22.01, TE02.22.02, TE02.24.01, TE02.24.02, TE02.26.01, TE02.26.02, TE02.26.03, TE02.26.04, TE02.26.05, TE02.28.01, TE02.28.02, TE02.30.01, TE02.30.02	TE02.15.01, TE02.15.02, TE02.15.03, TE02.15.04, TE02.15.05, TE02.15.06, TE02.15.07, TE02.15.08, TE02.15.09, TE02.15.10, TE02.15.11, TE02.15.12, TE02.15.13, TE02.15.14, TE02.17.01, TE02.17.02, TE02.17.03, TE02.17.04, TE02.17.05, TE02.17.06, TE02.17.07, TE02.17.08, TE02.17.09, TE02.17.10, TE02.18.01	TE02.15.01, TE02.15.02, TE02.15.03, TE02.15.04, TE02.15.05, TE02.15.06, TE02.15.07, TE02.15.08, TE02.15.09, TE02.15.10, TE02.15.11, TE02.15.12, TE02.15.13, TE02.15.14, TE02.17.01, TE02.17.02, TE02.17.03, TE02.17.04, TE02.17.05, TE02.17.06, TE02.17.07, TE02.17.08, TE02.17.09, TE02.17.10, TE02.18.01
3	TE02.03.01, TE02.03.02, TE02.07.01, TE02.07.02, TE02.09.01, TE02.10.01, TE02.10.02, TE02.11.01, TE02.11.02, TE02.12.01, TE02.13.01, TE02.13.02, TE02.13.03, TE02.14.01, TE02.16.01, TE02.16.02, TE02.16.03, TE02.16.04, TE02.16.05, TE02.19.01, TE02.19.02, TE02.20.01, TE02.20.02, TE02.20.03, TE02.20.04, TE02.21.01, TE02.21.02, TE02.22.01, TE02.22.02, TE02.24.01, TE02.24.02, TE02.26.01, TE02.26.02, TE02.26.03, TE02.26.04, TE02.26.05, TE02.28.01, TE02.28.02, TE02.30.01, TE02.30.02	TE02.15.01, TE02.15.02, TE02.15.03, TE02.15.04, TE02.15.05, TE02.15.06, TE02.15.07, TE02.15.08, TE02.15.09, TE02.15.10, TE02.15.11, TE02.15.12, TE02.15.13, TE02.15.14, TE02.17.01, TE02.17.02, TE02.17.03, TE02.17.04, TE02.17.05, TE02.17.06, TE02.17.07, TE02.17.08, TE02.17.09, TE02.17.10, TE02.18.01	TE02.15.01, TE02.15.02, TE02.15.03, TE02.15.04, TE02.15.05, TE02.15.06, TE02.15.07, TE02.15.08, TE02.15.09, TE02.15.10, TE02.15.11, TE02.15.12, TE02.15.13, TE02.15.14, TE02.17.01, TE02.17.02, TE02.17.03, TE02.17.04, TE02.17.05, TE02.17.06, TE02.17.07, TE02.17.08, TE02.17.09, TE02.17.10, TE02.18.01
4	TE02.03.01, TE02.03.02, TE02.07.01, TE02.07.02, TE02.09.01, TE02.10.01, TE02.10.02, TE02.11.01, TE02.11.02, TE02.12.01, TE02.13.01, TE02.13.02, TE02.13.03, TE02.14.01, TE02.16.01, TE02.16.02, TE02.16.03, TE02.16.04, TE02.16.05, TE02.19.01, TE02.19.02, TE02.20.01, TE02.20.02, TE02.20.03, TE02.20.04, TE02.21.01, TE02.21.02, TE02.22.01, TE02.22.02, TE02.24.01, TE02.24.02, TE02.26.01, TE02.26.02, TE02.26.03, TE02.26.04, TE02.26.05, TE02.28.01, TE02.28.02, TE02.30.01, TE02.30.02	TE02.15.01, TE02.15.02, TE02.15.03, TE02.15.04, TE02.15.05, TE02.15.06, TE02.15.07, TE02.15.08, TE02.15.09, TE02.15.10, TE02.15.11, TE02.15.12, TE02.15.13, TE02.15.14, TE02.17.01, TE02.17.02, TE02.17.03, TE02.17.04, TE02.17.05, TE02.17.06, TE02.17.07, TE02.17.08, TE02.17.09, TE02.17.10, TE02.18.01	TE02.15.01, TE02.15.02, TE02.15.03, TE02.15.04, TE02.15.05, TE02.15.06, TE02.15.07, TE02.15.08, TE02.15.09, TE02.15.10, TE02.15.11, TE02.15.12, TE02.15.13, TE02.15.14, TE02.17.01, TE02.17.02, TE02.17.03, TE02.17.04, TE02.17.05, TE02.17.06, TE02.17.07, TE02.17.08, TE02.17.09, TE02.17.10, TE02.18.01

While Table 4 focuses on the impact of TE filtering for software modules, the filtering criteria must also be applied to hardware-based implementations. Table 5 extends this analysis by examining TEs specific to single-chip hardware modules, mapping the applicable security requirements to different security levels. This comparison highlights the distinctions in validation approaches between software and hardware modules, ensuring that the filtering process remains consistent and comprehensive across various module types.

Table 5. Area 7 TEs Filtered by Security Level for Single Chip Hardware Modules

Sec Lvl	Applicable TEs	Non-Applicable TEs	TEs N/A due to Module Type/Embodiment
1	TE07.01.01, TE07.01.02, TE07.09.01, TE07.09.02, TE07.10.01, TE07.10.02, TE07.11.01, TE07.11.02, TE07.12.01, TE07.13.01, TE07.15.01, TE07.15.02	TE07.19.01, TE07.20.01, TE07.25.01, TE07.26.01, TE07.26.02, TE07.27.01, TE07.32.01, TE07.33.01, TE07.35.01, TE07.37.01, TE07.37.02, TE07.37.03, TE07.39.01, TE07.39.02, TE07.39.03, TE07.39.04, TE07.39.05, TE07.39.06, TE07.41.01, TE07.41.02, TE07.42.01, TE07.42.02, TE07.43.01, TE07.44.01, TE07.45.01, TE07.45.02, TE07.46.01, TE07.47.01, TE07.47.02, TE07.48.01, TE07.48.02, TE07.50.01, TE07.50.02, TE07.50.03, TE07.51.01, TE07.51.02, TE07.51.03, TE07.51.04, TE07.51.05, TE07.51.06, TE07.51.07, TE07.51.08, TE07.51.09, TE07.53.01, TE07.55.01, TE07.57.01, TE07.58.01, TE07.60.01, TE07.62.01, TE07.63.01, TE07.65.01, TE07.65.02, TE07.65.03, TE07.65.04, TE07.65.05, TE07.65.06, TE07.65.07, TE07.65.08, TE07.65.09, TE07.67.01, TE07.71.01, TE07.71.02, TE07.73.01, TE07.77.01, TE07.77.02, TE07.77.03, TE07.77.04, TE07.81.01, TE07.81.02, TE07.81.03	TE07.43.01, TE07.60.01
2	TE07.01.01, TE07.01.02, TE07.09.01, TE07.09.02, TE07.10.01, TE07.10.02, TE07.11.01, TE07.11.02, TE07.12.01, TE07.13.01, TE07.15.01, TE07.15.02, TE07.19.01, TE07.20.01, TE07.35.01	TE07.25.01, TE07.26.01, TE07.26.02, TE07.27.01, TE07.32.01, TE07.33.01, TE07.37.01, TE07.37.02, TE07.37.03, TE07.39.01, TE07.39.02, TE07.39.03, TE07.39.04, TE07.39.05, TE07.39.06, TE07.41.01, TE07.41.02, TE07.42.01, TE07.42.02, TE07.43.01, TE07.44.01, TE07.45.01, TE07.45.02, TE07.46.01, TE07.47.01, TE07.47.02, TE07.48.01, TE07.48.02, TE07.50.01, TE07.50.02, TE07.50.03, TE07.51.01, TE07.51.02, TE07.51.03, TE07.51.04, TE07.51.05, TE07.51.06, TE07.51.07, TE07.51.08, TE07.51.09, TE07.53.01, TE07.55.01, TE07.57.01, TE07.58.01, TE07.60.01, TE07.62.01, TE07.63.01, TE07.65.01, TE07.65.02, TE07.65.03, TE07.65.04, TE07.65.05, TE07.65.06, TE07.65.07, TE07.65.08, TE07.65.09, TE07.67.01, TE07.71.01, TE07.71.02, TE07.73.01, TE07.77.01, TE07.77.02, TE07.77.03,	TE07.43.01, TE07.44.01, TE07.45.01, TE07.45.02, TE07.46.01, TE07.47.01, TE07.47.02, TE07.48.01, TE07.48.02, TE07.60.01, TE07.62.01, TE07.63.01

Sec Lvl	Applicable TEs	Non-Applicable TEs	TEs N/A due to Module Type/Embodiment
		TE07.77.04, TE07.81.01, TE07.81.02, TE07.81.03	
3	TE07.01.01, TE07.01.02, TE07.09.01, TE07.09.02, TE07.10.01, TE07.10.02, TE07.11.01, TE07.11.02, TE07.12.01, TE07.13.01, TE07.15.01, TE07.15.02, TE07.19.01, TE07.20.01, TE07.25.01, TE07.26.01, TE07.26.02, TE07.27.01, TE07.35.01, TE07.37.01, TE07.37.02, TE07.37.03, TE07.39.01, TE07.39.02, TE07.39.03, TE07.39.04, TE07.39.05, TE07.39.06, TE07.73.01, TE07.77.01, TE07.77.02, TE07.77.03, TE07.77.04, TE07.81.01, TE07.81.02, TE07.81.03	TE07.32.01, TE07.33.01, TE07.41.01, TE07.41.02, TE07.42.01, TE07.42.02, TE07.43.01, TE07.44.01, TE07.45.01, TE07.45.02, TE07.46.01, TE07.47.01, TE07.47.02, TE07.48.01, TE07.48.02, TE07.50.01, TE07.50.02, TE07.50.03, TE07.51.01, TE07.51.02, TE07.51.03, TE07.51.04, TE07.51.05, TE07.51.06, TE07.51.07, TE07.51.08, TE07.51.09, TE07.53.01, TE07.55.01, TE07.57.01, TE07.58.01, TE07.60.01, TE07.62.01, TE07.63.01, TE07.65.01, TE07.65.02, TE07.65.03, TE07.65.04, TE07.65.05, TE07.65.06, TE07.65.07, TE07.65.08, TE07.65.09, TE07.67.01, TE07.71.01, TE07.71.02	TE07.43.01, TE07.44.01, TE07.45.01, TE07.45.02, TE07.46.01, TE07.47.01, TE07.47.02, TE07.48.01, TE07.48.02, TE07.50.01, TE07.50.02, TE07.50.03, TE07.51.01, TE07.51.02, TE07.51.03, TE07.51.04, TE07.51.05, TE07.51.06, TE07.51.07, TE07.51.08, TE07.51.09, TE07.60.01, TE07.62.01, TE07.63.01, TE07.65.01, TE07.65.02, TE07.65.03, TE07.65.04, TE07.65.05, TE07.65.06, TE07.65.07, TE07.65.08, TE07.65.09
4	TE07.01.01, TE07.01.02, TE07.09.01, TE07.09.02, TE07.10.01, TE07.10.02, TE07.11.01, TE07.11.02, TE07.12.01, TE07.13.01, TE07.15.01, TE07.15.02, TE07.19.01, TE07.20.01, TE07.25.01, TE07.26.01, TE07.26.02, TE07.27.01, TE07.32.01, TE07.33.01, TE07.35.01, TE07.37.01, TE07.37.02, TE07.37.03, TE07.39.01, TE07.39.02, TE07.39.03, TE07.39.04, TE07.39.05, TE07.39.06, TE07.41.01, TE07.41.02, TE07.42.01, TE07.42.02, TE07.77.01, TE07.77.02, TE07.77.03, TE07.77.04	TE07.43.01, TE07.44.01, TE07.45.01, TE07.45.02, TE07.46.01, TE07.47.01, TE07.47.02, TE07.48.01, TE07.48.02, TE07.50.01, TE07.50.02, TE07.50.03, TE07.51.01, TE07.51.02, TE07.51.03, TE07.51.04, TE07.51.05, TE07.51.06, TE07.51.07, TE07.51.08, TE07.51.09, TE07.53.01, TE07.55.01, TE07.57.01, TE07.58.01, TE07.60.01, TE07.62.01, TE07.63.01, TE07.65.01, TE07.65.02, TE07.65.03, TE07.65.04, TE07.65.05, TE07.65.06, TE07.65.07, TE07.65.08, TE07.65.09, TE07.67.01, TE07.71.01, TE07.71.02, TE07.73.01, TE07.81.01, TE07.81.02, TE07.81.03	TE07.43.01, TE07.44.01, TE07.45.01, TE07.45.02, TE07.46.01, TE07.47.01, TE07.47.02, TE07.48.01, TE07.48.02, TE07.50.01, TE07.50.02, TE07.50.03, TE07.51.01, TE07.51.02, TE07.51.03, TE07.51.04, TE07.51.05, TE07.51.06, TE07.51.07, TE07.51.08, TE07.51.09, TE07.53.01, TE07.55.01, TE07.57.01, TE07.58.01, TE07.60.01, TE07.62.01, TE07.63.01, TE07.65.01, TE07.65.02, TE07.65.03, TE07.65.04, TE07.65.05, TE07.65.06, TE07.65.07, TE07.65.08, TE07.65.09, TE07.67.01, TE07.71.01, TE07.71.02
N/A		TE07.01.01, TE07.01.02, TE07.09.01, TE07.09.02, TE07.10.01, TE07.10.02, TE07.11.01, TE07.11.02, TE07.12.01, TE07.13.01, TE07.15.01, TE07.15.02, TE07.19.01, TE07.20.01, TE07.25.01, TE07.26.01, TE07.26.02, TE07.27.01, TE07.32.01, TE07.33.01, TE07.35.01, TE07.37.01, TE07.37.02, TE07.37.03, TE07.39.01, TE07.39.02, TE07.39.03, TE07.39.04, TE07.39.05, TE07.39.06,	

Sec Lvl	Applicable TEs	Non-Applicable TEs	TEs N/A due to Module Type/Embodiment
		TE07.41.01, TE07.41.02, TE07.42.01, TE07.42.02, TE07.43.01, TE07.44.01, TE07.45.01, TE07.45.02, TE07.46.01, TE07.47.01, TE07.47.02, TE07.48.01, TE07.48.02, TE07.50.01, TE07.50.02, TE07.50.03, TE07.51.01, TE07.51.02, TE07.51.03, TE07.51.04, TE07.51.05, TE07.51.06, TE07.51.07, TE07.51.08, TE07.51.09, TE07.53.01, TE07.55.01, TE07.57.01, TE07.58.01, TE07.60.01, TE07.62.01, TE07.63.01, TE07.65.01, TE07.65.02, TE07.65.03, TE07.65.04, TE07.65.05, TE07.65.06, TE07.65.07, TE07.65.08, TE07.65.09, TE07.67.01, TE07.71.01, TE07.71.02, TE07.73.01, TE07.77.01, TE07.77.02, TE07.77.03, TE07.77.04, TE07.81.01, TE07.81.02, TE07.81.03	

2.2.3. TE Impacted by Supplemental TE Filters

In addition to the basic TE filtering criteria, supplemental filters further refine the selection of applicable test evidence based on specific module properties and security features. Table 6 highlights the TEs affected by these supplemental filtering properties, which include factors such as authentication mechanisms, cryptographic output capabilities, tamper response measures, and other specialized security attributes. By applying these filters, the validation process can be optimized to focus on the most relevant security assurances while reducing redundant or inapplicable tests, which enhances the efficiency and accuracy of the TE selection process.

Table 6. TEs Affected by the Supplemental Filtering Properties

Filter Property	Include If True	Exclude If False	Number of Affected TEs
Has Excluded Components		TE02.13.01, TE02.13.02, TE02.13.03, TE02.14.01, TE02.15.05, TE02.16.04, TE02.17.04	7
Has EFP		TE07.77.01, TE07.77.02, TE07.77.03, TE07.77.04	4
Uses Split Knowledge		TE09.21.01, TE09.21.02, TE09.21.03, TE09.21.04, TE09.22.01, TE09.23.01, TE09.23.02, TE09.23.04, TE09.24.01	9
Allows Self-Initiated Cryptographic Output		TE04.23.01, TE04.25.01, TE04.25.02, TE04.25.03	4

Filter Property	Include If True	Exclude If False	Number of Affected TEs
Supports Bypass Capability		TE04.18.01, TE04.19.01, TE04.19.02, TE04.19.03, TE04.20.01, TE04.20.02, TE04.20.03, TE04.21.01, TE04.21.02, TE04.22.01, TE04.22.02, TE10.21.01, TE10.21.02, TE10.21.03, TE10.21.04, TE10.22.01, TE10.22.02, TE10.22.03, TE10.22.04, TE10.22.05, TE10.48.01, TE10.48.02, TE10.48.03, TE10.49.01, TE10.49.02, TE10.49.03, TE10.51.01, TE10.51.02, TE10.51.03	29
Has Identity-Based Authentication		TE03.20.01, TE04.39.01, TE04.39.02, TE04.39.03, TE04.39.04, TE04.42.01, TE04.42.02, TE04.42.03, TE04.42.04, TE09.22.01	10
Provides Maintenance Access Interface	TE07.50.03	TE07.11.01, TE07.11.02, TE07.12.01, TE07.13.01, TE07.51.07, TE07.51.08, TE07.65.02, TE07.65.07, TE07.65.08, TE11.08.07	11
Uses EDC		TE05.06.02, TE05.07.01	2
Supports Manual SSP Entry		TE09.14.01, TE09.14.02, TE10.46.01, TE10.46.02, TE10.46.03, TE10.46.04	6
Supports Concurrent Operators		TE04.02.01, TE04.02.02, TE04.02.03	3
Supports Software Firmware Loading		TE04.28.01, TE04.29.01, TE04.32.01, TE04.34.01, TE05.13.01, TE05.13.02, TE05.13.03, TE05.13.04, TE05.13.05, TE05.13.06, TE05.13.07, TE05.13.08	12
Supports Complete Image Replacement		TE04.33.01, TE04.35.01, TE04.35.02	3
Uses Hash MAC Integrity		TE05.05.03	1
Has Control Output		TE03.09.01, TE03.09.02, TE03.10.01, TE03.10.02, TE03.10.03, TE03.10.04, TE03.10.05	7
Has Ventilation or Slits		TE07.20.01, TE07.25.01	2
Has EDC		TE10.46.02, TE10.46.03	2
Has External Input Device		TE03.05.02, TE03.08.02	2
Has User Role		TE04.06.01	1
Has External Output Device		TE03.06.02, TE03.11.02	2
Has Removable Cover	TE07.50.03	TE07.13.01, TE07.20.01, TE07.25.01, TE07.39.02, TE07.39.05, TE07.47.01, TE07.47.02, TE07.48.01, TE07.48.02, TE07.51.02, TE07.51.07, TE07.51.08,	18

Filter Property	Include If True	Exclude If False	Number of Affected TEs
		TE07.62.01, TE07.63.01, TE07.65.02, TE07.65.07, TE07.65.08	
Outputs Sensitive Data as Plaintext		TE09.16.01, TE09.16.02, TE09.16.03	3
Has Critical Functions		TE10.24.01, TE10.24.02	2
Uses Authentication		TE04.43.01, TE04.43.02, TE04.44.01, TE04.44.02, TE04.45.01, TE04.45.02, TE04.45.03, TE04.47.01, TE04.48.01, TE04.50.01, TE04.50.02, TE04.51.01, TE04.51.02, TE04.52.01, TE04.53.01, TE04.54.01, TE04.54.02, TE04.54.03, TE04.55.01, TE04.55.02	20
Uses Role-Based Authentication		TE04.37.01, TE04.37.02, TE04.38.01, TE04.38.02	4
Has Default Authentication Data		TE04.45.03	1
Has Degraded Mode		TE02.26.01, TE02.26.02, TE02.26.03, TE02.26.04, TE02.26.05, TE02.28.01, TE02.28.02, TE02.30.01, TE02.30.02	9
Has EFT		TE07.81.01, TE07.81.02, TE07.81.03	3
Has Trusted Channel		TE03.16.01, TE03.18.01, TE03.18.02, TE03.19.01, TE03.19.02, TE03.19.03, TE03.19.04, TE03.20.01, TE03.21.01, TE03.22.01, TE09.21.01, TE09.21.04	12
Uses Multi-Factor Authentication		TE04.59.01, TE09.24.01, TE09.24.02	3
Allows Operator to Change Roles		TE04.38.01, TE04.38.02, TE04.42.01, TE04.42.02, TE04.42.03, TE04.42.04	6
Uses Digital Signature Integrity		TE05.05.04	1
Has Maintenance Role		TE04.07.01, TE04.07.02, TE04.07.03	3
Has Additional Mitigations		TE12.01.01, TE12.02.01, TE12.04.01, TE12.04.02, TE12.04.03	5
Supports Sensitive Data I/O		TE09.13.01, TE09.13.02, TE09.13.03, TE09.18.01, TE09.18.02, TE09.19.01	6
Has Tamper Seals		TE07.27.01, TE07.48.01, TE07.48.02, TE07.63.01	4
Has CVE		TE11.38.03	1
Total number of TEs affected by the supplemental filter properties			**192**

Note: The total number of the TEs affected by the supplemental filter properties is not the sum of the numbers in the column of "Number of Affected TEs" (i.e., 218) because some TEs are affected by multiple filter properties and so appear multiple times in Table 6.

2.3. Removing ASes Not Separately Tested

Some assertions (ASes) are not separately tested, and they do not have associated TEs.

These ASes depend on the completion of other ASes and their TEs. For example: **AS05.22** is not separately tested but is instead tested as part of **AS05.05**. Table 7 highlights ASes that are not separately tested. Since these ASes are conditional in nature, a solution to the problem they pose could be to use these assertions to further automate the report writing process. In this instance, the AS that is not separately tested could be omitted from the report template provided by the NCCoE ACMVP server if the server will include ASes in addition to TEs.

The TE Workstream does not address the dependency at the TE level (e.g., TE10.28.02 and TE10.34.03) as opposed to the AS level.

Table 7. Assertions (ASs) not separately tested

FIPS 140-3 Section Title	ASes not separately tested
General	N/A
Cryptographic Module Specification	AS02.01, AS02.02, AS02.04, AS02.05, AS02.06, AS02.08, AS02.25, AS02.26, AS02.29, AS02.31, AS02.32
Cryptographic Module Interfaces	AS03.12, AS03.17
Roles, Services, and Authentication	AS04.01, AS04.05, AS04.08, AS04.09, AS04.10, AS04.12, AS04.16, AS04.17, AS04.24, AS04.26, AS04.27, AS04.30, AS04.31, AS04.36, AS04.40, AS04.41, AS04.46, AS04.49, AS04.57, AS04.58
Software/Firmware Security	AS05.01, AS05.03, AS05.09, AS05.10, AS05.14, AS05.18, AS05.19, AS05.21, AS05.22
Operational Environment	AS06.01, AS06.02, AS06.04, AS06.09, AS06.16, AS06.21, AS06.22, AS06.23, AS06.29
Physical Security	AS07.02, AS07.03, AS07.04, AS07.05, AS07.06, AS07.07, AS07.08, AS07.14, AS07.16, AS07.17, AS07.18, AS07.21, AS07.22, AS07.23, AS07.24, AS07.28, AS07.29, AS07.30, AS07.31, AS07.34, AS07.36, AS07.38, AS07.40, AS07.49, AS07.52, AS07.54, AS07.56, AS07.59, AS07.61, AS07.64, AS07.66, AS07.68, AS07.69, AS07.70, AS07.72, AS07.74, AS07.75, AS07.76, AS07.78, AS07.79, AS07.80, AS07.81, AS07.82, AS07.83, AS07.84, AS07.85, AS07.86
Non-Invasive Security	N/A
Sensitive Security Parameter Management	AS09.11, AS09.12, AS09.15, AS09.17, AS09.20, AS09.26, AS09.30, AS09.34, AS09.35

FIPS 140-3 Section Title	ASes not separately tested
Self-Tests	AS10.01, AS10.02, AS10.03, AS10.04, AS10.05, AS10.06, AS10.13, AS10.14, AS10.16, AS10.17, AS10.18, AS10.19, AS10.23, AS10.26, AS10.30, AS10.31, AS10.32, AS10.32, AS10.36, AS10.38, AS10.39, AS10.40, AS10.41, AS10.42, AS10.43, AS10.44, AS10.45, AS10.47, AS10.50, AS10.52, AS10.55
Life-Cycle Assurance	AS11.02, AS11.07, AS11.09, AS11.10, AS11.12, AS11.14, AS11.20, AS11.22, AS11.27
Mitigation of Other Attacks	None

3. Protocol Workstream

The Protocol Workstream defines the interactions between automated CMVP server assets and the NCCoE ACMVP clients supporting a proof-of-concept of automation capabilities. This section captures the progress made since the last report in September 2024.

The ACMVP Protocol Workstream is led by Barry Fussell and Andrew Karcher of Cisco and Chris Celi of NIST with contributions from Panos Kampanakis of Amazon, Michael McCarl and Deborah Harrington of AEGISOLVE, Alex Thurston of Lightship, Stephan Mueller and Walker Riley of atsec, Mike Grimm of Microsoft, Robert Staples of NIST, and Raoul Gabiam, Michael Dimond, Kyle Vitale, Doris Rui, and Matthew Fortes of the MITRE Corporation.

3.1. Proof-of-Concept Server Features

The proof-of-concept server currently implements the following features:

- Two-factor authentication using TOTP and mTLS, which improves the TOTP from ACVP by allowing a user to maintain multiple seeds for simultaneous connections
- Module registration that defines the security levels, embodiment, and other properties of the cryptographic module and automatically determines which TEs are applicable to the cryptographic module
- Module evidence submission that prompts a client to provide evidence addressing TEs that are applicable to the cryptographic module and will show which TEs have not yet been addressed by the submission to ensure completeness
- Module security policy submission defined entirely in JSON, which will generate the security policy automatically, allowing the client to retrieve the completed PDF, and ensures that all sections are present and completed.
- The awarding of a validation certificate once all evidence and security policy information are completed
- Automatic processing of functional test evidence (FE-TEs) based on the test type selected by the lab
- Acceptance of source code test evidence based on the test procedure selected by the lab

3.2. Server Implementation

The server uses much of the same infrastructure as ACVP and ESV, which is intended to keep the same team available to maintain the systems once they are integrated by the CMVP. The system is comprised of C# and Python applications along with SQL Server databases.

The server development team is also using this opportunity to re-evaluate the required security assurances within NIST to see if any improvements can be implemented into the rest of the CMVP applications, which includes the requirement for Two-Factor authentication, separation between internal and external systems, International Traffic and Arms Restrictions (ITAR), and other elements of the ACVP and ESV systems.

3.3. Client Implementations

This section describes the two open source clients, Libamvp and ACVP Proxy, that provide foundational code for developers to build upon when interfacing with the server.

3.3.1. Libamvp - Cisco

Libamvp is an example client for the AMVP protocol developed by Cisco engineers. It is C based and interacts with the server by parsing user-generated JSON and is intended to be a simple tool to showcase the protocol and assist developers as they create workflows for the generation and submission of AMVP data. Libamvp can create modules and certification requests, submit all required evidence and security policy information, retrieve security policy PDFs, check for the status of a certification request, and other actions, as development continues.

Libamvp can be found here: https://github.com/cisco/libamvp.

3.3.2. ACVP Proxy – atsec

The client is called the ACVP Proxy and is supported by atsec information security corp. It provides the interface to access the NIST ACVP, ESVP, and AMVP services using an open sourced code that is available at the public repository: https://github.com/smuellerDD/acvpproxy.

The ACVP Proxy has many options, allows a flexible deployment, and is extendable to cover an arbitrary number of IUT definitions. The AVP Proxy implements the entire interaction with the NIST servers to obtain the data from the server and upload all required data to the server.

3.4. Accessing the ACMVP Demo Server

Here are the instructions and steps to request access to the upcoming demo environment:

Send a CSR (Certificate Signing Request) file to the CMVP via the Secure File Communication service found at the URL https://sfc.doc.gov. Due to policy, a CSR cannot be accepted via email or email attachment and must be sent through the SFC system. To establish an account on SFC, send an email to amvp-demo@nist.gov.

Please send the CSR file in PEM format following these requirements:

1. Use this naming convention for the CSR:
    - **OrganizationName_FirstName_LastName_AMVPDemo.csr**
        - No spaces in the filename

        - No more than three underscore "_" characters in the filename
    - Do not zip the file. Send it exactly as specified above. Any file submitted beyond a reasonable CSR size (maximum 10KB) will be automatically rejected
    - Use a minimum 2048-bit RSA key pair
    - Sign using at least a SHA-256 hash
    - Include the EMAILADDRESS attribute in the certificate subject, which can either be the user's email address OR a group alias email address if applicable (If a single user email address is used, the generated certificate is non-transferable)
    - Include the CN attribute in the certificate subject, which can either be the user's first and last name OR the name of the organization
    - No URLs in the CN attribute
    - If submitting multiple CSRs using the same organization name and group email alias, the CN attribute *must* be unique for each submission (e.g. CN=Orgname 1, CN=Orgname 2, CN=Orgname 3, etc.) because the submission will be rejected with feedback to fix the error if it does not meet this requirement
    - Ensure the C (country) attribute is only two letters

   For example:

```
EMAILADDRESS=email.address@domain.com, CN=firstname lastname,
OU=organization.unit, O=organization.name, L=city, ST=state,
C=country.abbreviation
```

   Here are the openssl commands to generate a csr:

```
openssl genrsa -out private-key-name.key 4096
openssl req -new -key private-key-name.key -out
   OrganizationName_FirstName_LastName_AMVPDemo.csr -sha256
```

2. Upon receipt of the CSR file, the CMVP will validate that it meets the above stated requirements and will point out via email response what needs to be corrected if there are any issues
3. Once the certificate is generated, a notification will be sent with the certificate and TOTP seed via an SFC message and the credentials will be valid immediately upon receipt

Users are expected to protect the keypair from unauthorized use and notify NIST in the event the keypair becomes compromised in any way.

Note that per policy, SFC accounts and attachments are only valid for two calendar weeks from when the invitation email is sent. Existing SFC accounts may be used to send the CSR file but it is advised to begin the process by sending the initial request to amvp-demo@nist.gov.

Note that external SFC accounts will go dormant after two weeks by NIST policy, which is normal behavior. After the certificate is exchanged, there is no further need for SFC.

Additionally, the account can be reinstated at any point in time by going through the same process.

3.5. Planned Work

This work is still in progress. Here are some features that will be addressed by Fall 2025:

- Continue developing automated checklist rules to ensure submissions are as correct as possible before entering the hands of a reviewer
- Add reviewer comment rounds to the protocol and implementations rather than handle them out of band over encrypted email
- Begin integrating ACMVP research products into the production CMVP workflows

4. Research Infrastructure Workstream

Over the past few months, the infrastructure workstream team adopted an iterative approach to modernize the CMVP supporting infrastructure. Each iteration introduced progressively advanced architectures, leveraging cloud-native services to improve scalability, portability, deployment speed, and security, all while ensuring cost efficiency. The modernization efforts have resulted in a containerized application compatible with both Windows and Linux platforms using Amazon Elastic Container Service. Furthermore, it integrates a managed database service to enhance operational efficiency and features a fully automated CI/CD pipeline to simplify and streamline deployments on a Linux platform. Authentication mechanisms have been modernized to incorporate cloud-native solutions, including the AWS ALB. The remaining tasks include completing a final iteration that employs AWS Elastic Kubernetes Service as an alternative container deployment service and implementing Amazon API Gateway to modernize the authentication process for server API requests.

The Research Infrastructure Workstream is led by Raoul Gabiam of The MITRE Corporation and Douglas Boldt of Amazon, with contributions from Courtney Maatta, Annie Cimack, Diana Brooks, Charlotte Fondren, Zhuo-Wei Lee, Keonna Parrish, Abhishek Isireddy, Abi Adenuga, Bradley Wyman, Brittany Robinson, Gina McFarland, Damian Zell, Cavan Slaughter, Rayette Toles-Abdullah, Keith Hodo, John Dwyer, Ahmed Virani, Daftari Mrunal, Kasireddi Srikar Reddy, Srujana Alajangi, and Natti Swaminathan of Amazon; Robert Staples and Murugiah Souppaya of NIST; Jason Arnold of HII; Michael Dimond, Kyle Vitale, Phillip Millwee, and Josh Klosterman of the MITRE Corporation; and John Booton, Aaron Cook, and Jeffrey LaClair of ITC Federal.

4.1. Modernization Approach

The existing CMVP production environment was initially deployed in a data center internal to NIST. A subset of the environment that was providing services to the test labs was virtualized and migrated to AWS GovCloud to take advantage of the high availability and resiliency offered by cloud infrastructure. The CMVP system administrators have maintained the AWS infrastructure for several years.

The modernization journey started with a complete inventory and understanding of the existing production environment in AWS, including all the virtualized assets, the network, data flows, functionalities, and dependencies. Once the existing architecture was fully documented, it was replicated in a research environment managed by the NCCoE team to establish an initial baseline that could be analyzed, and opportunities were identified to incrementally modernize the application and supporting infrastructure throughout the lifecycle of this project. The NCCoE research is performed in AWS to ensure the findings can be easily replicated in the production environment. The objective is to deliver the new capabilities required at the application level to support the Protocol Workstream while maintaining some compatibilities with the existing production environment.

4.2. Replication of the Legacy Production CMVP Environment

This section gives historical context to the ACMVP application. The production CMVP AWS environment was replicated to the current ACMVP research environment, which set a baseline from which modernization opportunities were identified.

Figure 1 represents the baseline architecture present in the research environment before modernization efforts.

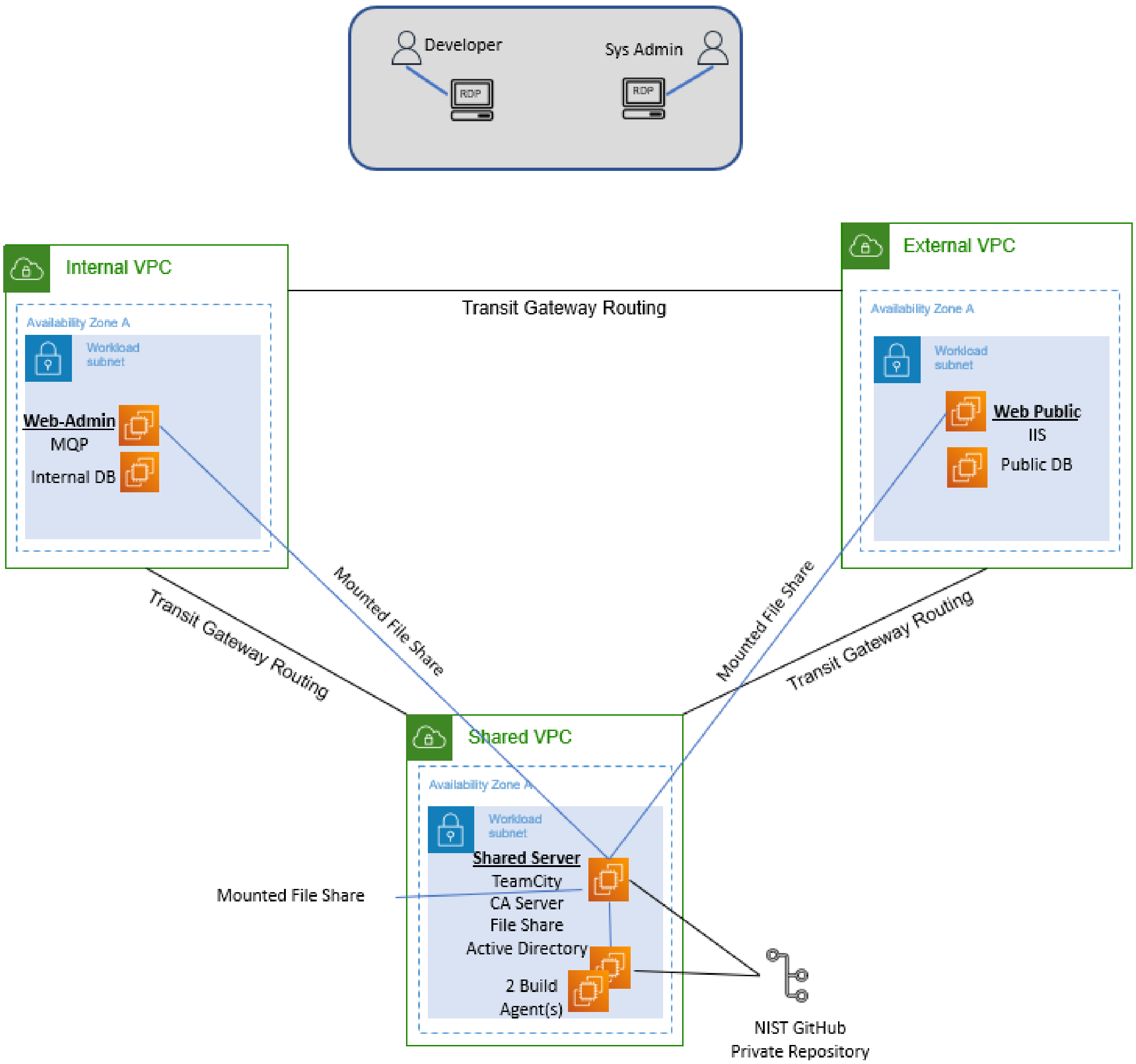


Fig. 1. Legacy System Architecture Diagram

The External Amazon Virtual Private Cloud (VPC) handles any public-facing applications and utilities, including the WebPublic application (sitting underneath Microsoft IIS) and the public database. These services are split into two separate Amazon EC2 instances.

The Internal Amazon VPC hosts private applications and utilities, including the MessageQueueProcessor (MQP) application and the internal database. These services are split into two separate Amazon EC2 instances.

The Shared Amazon VPC hosts shared applications and utilities, including JetBrains TeamCity for CI/CD, the Certificate Authority (CA) server, the file share service for backups and logs, and the Microsoft Active Directory service, which is hosted on one Amazon EC2 instance in the research environment for the sake of simplicity.

Figure 2 details the steps in the workflow that occur when the user submits a request, which are listed in this document to describe the necessary tools and their use cases in the critical workflow.

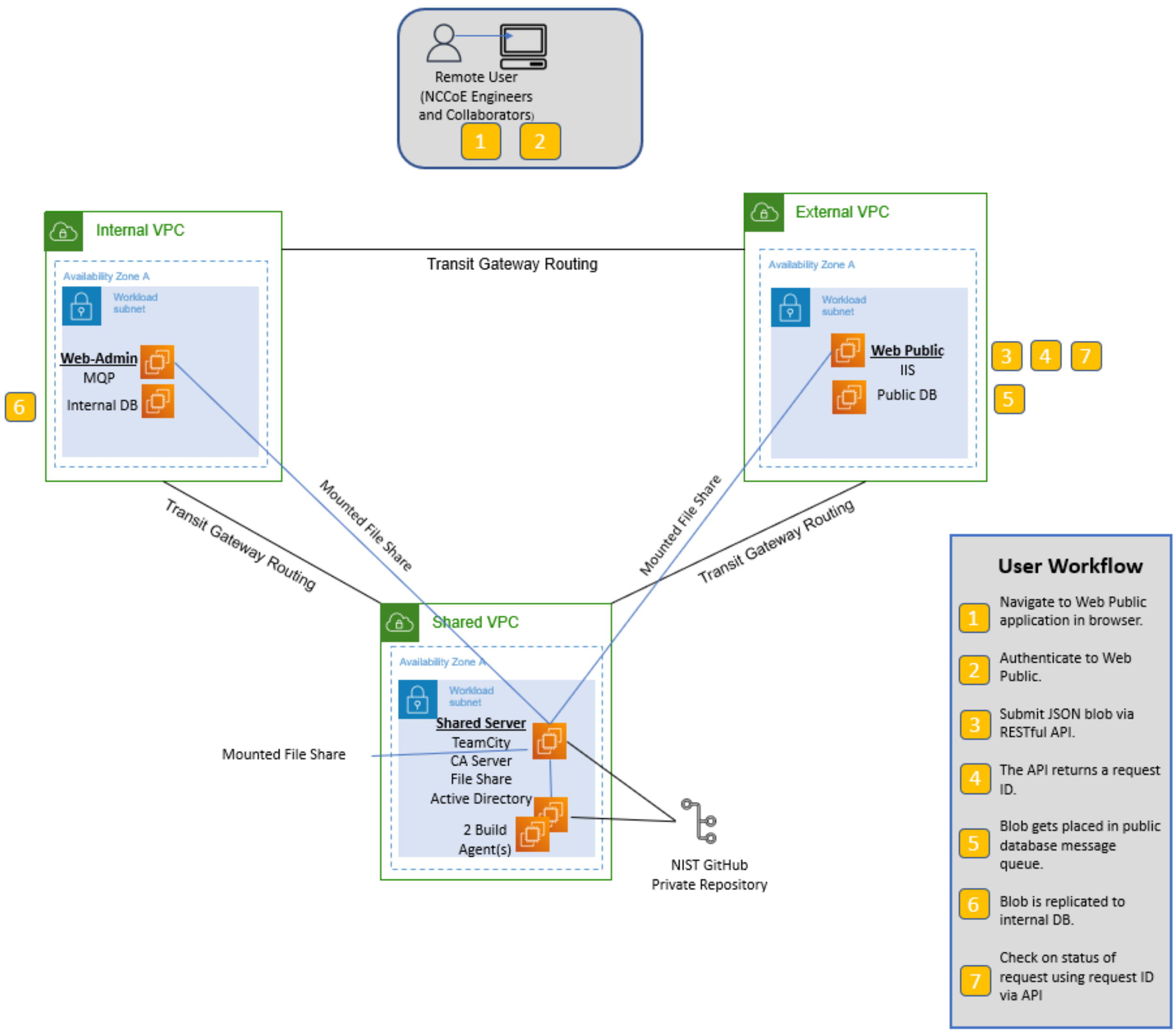


Fig. 2. Legacy System End User Workflow

WebPublic is publicly available for registered NVLAP users to submit their requests, which includes authentication requests that are partially handled by Microsoft IIS for Windows Server through mutual TLS (mTLS). Microsoft IIS receives its server-hosting certificate through the CA

Server. The application stores and retrieves data from the Public DB as needed by the requests it receives. Any stored data is replicated to the Internal DB through the encrypted message queue (MQ). The MQP processes the request and stores necessary changes to the Internal DB, which is replicated to the Public DB for user retrieval. Logging occurs throughout the process, tracking the request and where the processing is in the WebPublic or MQP application. These logs are stored to a file share for access by a system administrator, along with database backups.

4.3. AWS Target Architectures by Service

This section maps services in the baseline legacy infrastructure to equivalent services provided by AWS. Due to the CMVP system administrators' familiarity with hosting environments in AWS, the research was focused on AWS-based solutions. While this document only addresses AWS services, equivalent services could be found in other cloud providers.

Table 8 provides the mapping between services used in the legacy ACMVP research environment and equivalent services offered by AWS. A more detailed explanation between the mappings can be found below. Explanations are provided for selected mapped services. Services in bold were modernized to equivalent versions, and services in italics were not selected for modernization.

Table 8. Modernized Service Mapping

Service In Legacy ACMVP	AWS Equivalent Service(s) Considered	AWS Selected Service(s)
Microsoft SQL Server Database	**Amazon Relational Database Service (RDS) for SQL Server, Amazon Aurora, PostgreSQL**	**Amazon RDS for SQL Server**
Microsoft SQL Server Replication	**AWS Database Migration Service (DMS)**	**AWS DMS**
JetBrains TeamCity	**AWS CodePipeline, AWS CodeBuild**	**AWS CodeBuild**
WebPublic	**Containerized Application, Amazon Elastic Container Service (ECS), Amazon Elastic Kubernetes Service (EKS), Amazon Lambda**	**Amazon ECS and Amazon EKS**
MessageQueueProcessor	**Containerized Application, Amazon ECS, Amazon EKS, Amazon Lambda, Amazon SQS, Amazon MQ**	**Amazon ECS and Amazon EKS**
Microsoft IIS	**AWS Application Load Balancer (ALB), AWS Network Load Balancer (NLB), Amazon API Gateway, Nginx Reverse Proxy**	**AWS ALB**
Microsoft Active Directory	*AWS Managed Microsoft AD*	*No changes made*
Microsoft Windows AD DS	*AWS Route 53 with AWS Managed Microsoft AD*	*No changes made*
File Share	*Amazon FXs for Windows, Amazon S3, AWS Storage Gateway*	*No changes made*
Git Repository	*AWS Code Commit*	*No changes made*

Equivalent AWS services for the Microsoft SQL Server Database are Amazon RDS for SQL Server, Amazon Aurora, and PostgreSQL. Amazon Aurora only supports MySQL and PostgreSQL, requiring a change from the ACMVP's use of Microsoft SQL Server. Amazon RDS supports a managed version of Microsoft SQL Server. Amazon RDS was selected as the modernization approach due to the existing CMVP code that relies on Microsoft SQL Server.

AWS DMS was selected following the decision to use Amazon RDS to meet the need for data replication. Data replication in Amazon RDS requires AWS DMS, as the instances hosting the databases are managed by AWS and may change IP addresses over time. AWS manages this by providing DNS names to resolve the IP addresses for the databases.

JetBrains TeamCity's equivalent service is mapped to AWS CodeBuild, which was used to provide insight to the CMVP on alternative technologies.

WebPublic had the potential to be containerized or moved to an Amazon Lambda function. The containerized option was selected as it enables local testing, integrates with GitHub and allows for portability of the codebase. Note that streamlining the deployment process and improving code portability were desired outcomes of the production CMVP infrastructure support team. WebPublic was deployed via a Docker daemon on a NIST Secure Amazon EC2 instance to meet security requirements for a demo server, but Amazon ECS and Amazon EKS were selected as the modernization approaches in the research environment.

The MQP was mapped to other MQ services. However, the developed MQP performs functions unique to the ACMVP application, resulting in a decision to containerize the application.

Microsoft IIS was mapped to AWS ALB, AWS NLB, Amazon API Gateway, and Nginx Reverse Proxy. The AWS NLB handles layer 3 request routing to the application, requiring Microsoft IIS or Nginx to process mTLS authentication, or Amazon API Gateway to process API keys as an alternative mode of authentication. The AWS ALB was selected as it processes both mTLS authentication and the routing to the containerized WebPublic application. The other tools may still meet the requirements but were not explored further.

While equivalent services were identified for GitHub, Microsoft Active Directory, Microsoft Windows AD DS, and File Share, these services were left unchanged as they were already well established within the environment.

4.4. Key Modernization Components

This section describes the specific modernization research items completed or planned in the scope of the ACMVP application. As the application is a REST API with a backend database and MQP, similarly structured applications can utilize this research in making informed decisions to update, improve, or otherwise modernize their infrastructure.

Figures 3, 4, and 5 depict a timeline of the key modernization components that have been implemented before ICMC '25 and are planned to be implemented following ICMC '25. A pentagon flag in dark blue represents a timeline event, a green rectangle represents a Windows OS container development, a cyan hexagonal represents a general modernization development, and an orange elliptical represents a Linux OS container development. Note that AWS CodePipeline CI/CD is in orange, as it only applies to Linux OS containers, as explained within the Application Deployment Modernization section.

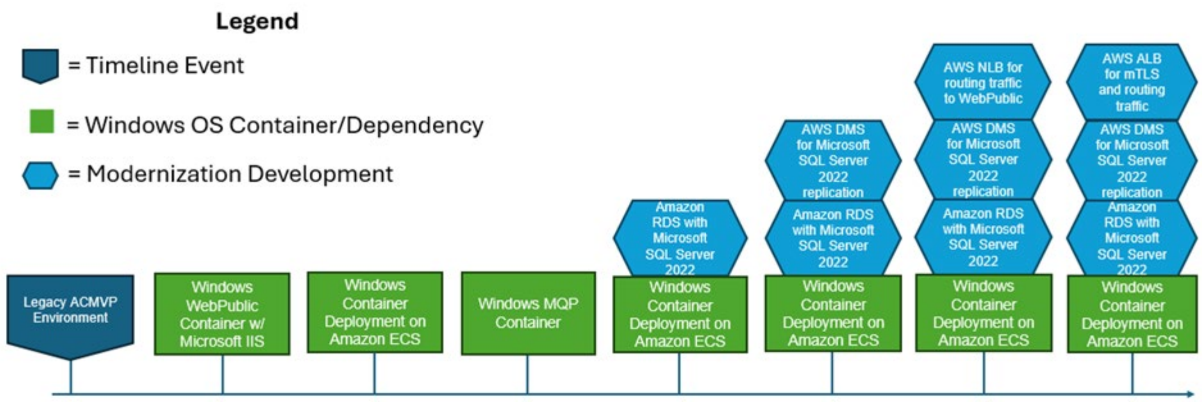


Fig. 3. Windows Container OS Modernization Progression

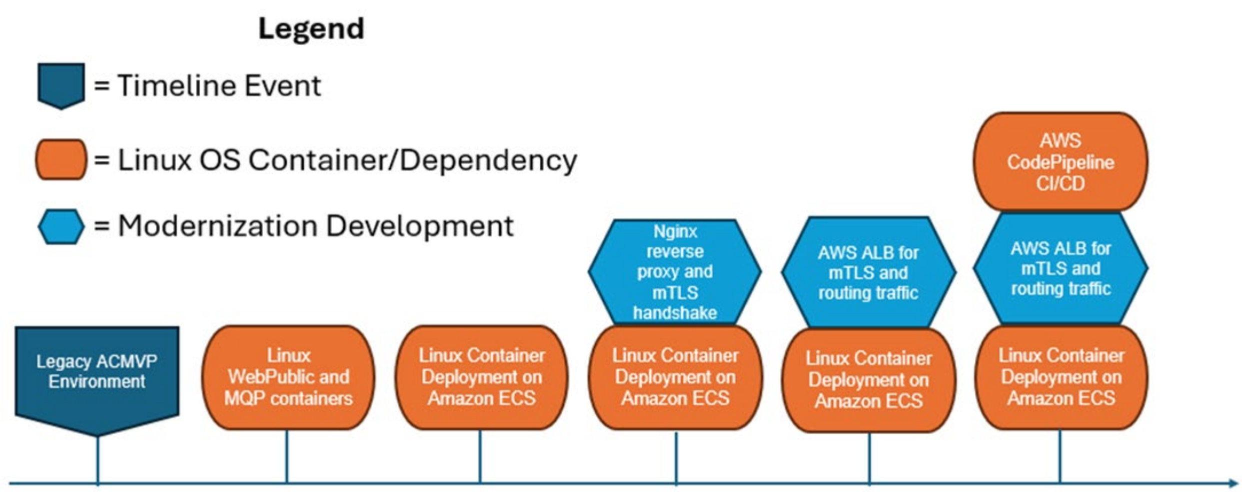


Fig. 4. Linux Container OS Modernization Progression

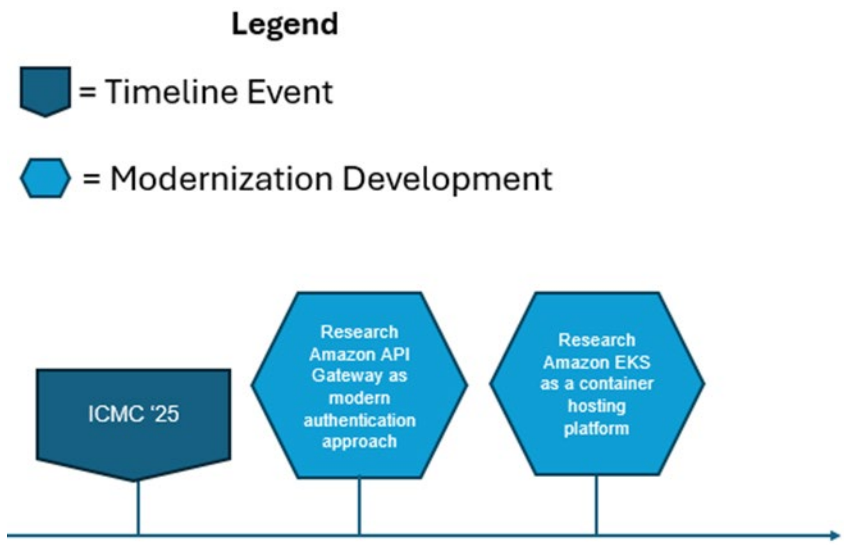


Fig. 5. Future Research Progression

Figure 6 shows the services and tools used in the modernized system architecture.

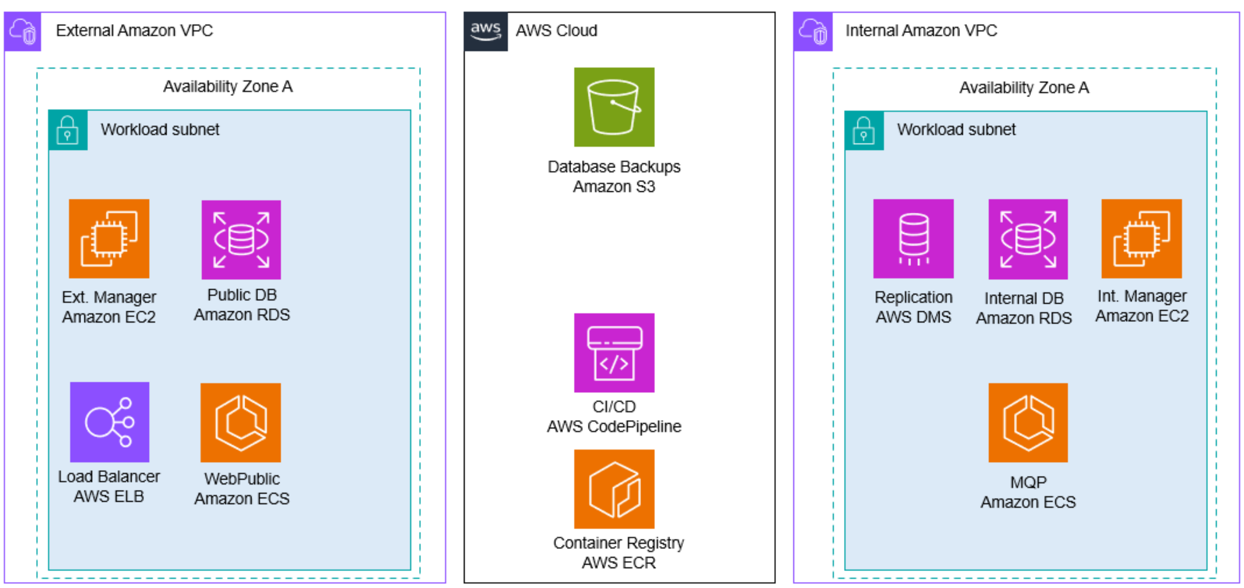


Fig. 6. Modernized System Architecture

Figure 7 depicts the desired client workflow through the modernized resources. The client connects to an AWS NLB or ALB, whose destination is open to the public. The load balancer forwards the traffic to the WebPublic application, running through one of the launch types identified in the Application Deployment Modernization section. This application uses its connection to the Public Database to store the data passed through by the client. AWS DMS, lying in the Internal Amazon VPC, replicates that information to the Internal Database through the MessageQueue table. The MQP recognizes the new items in the queue and processes them, finishing its processing by storing updates back into the Internal Database. These updates are replicated back into the External Database through the AWS DMS instance. Once updates are populated into the External Database, clients can view those changes through their original connection workflow.

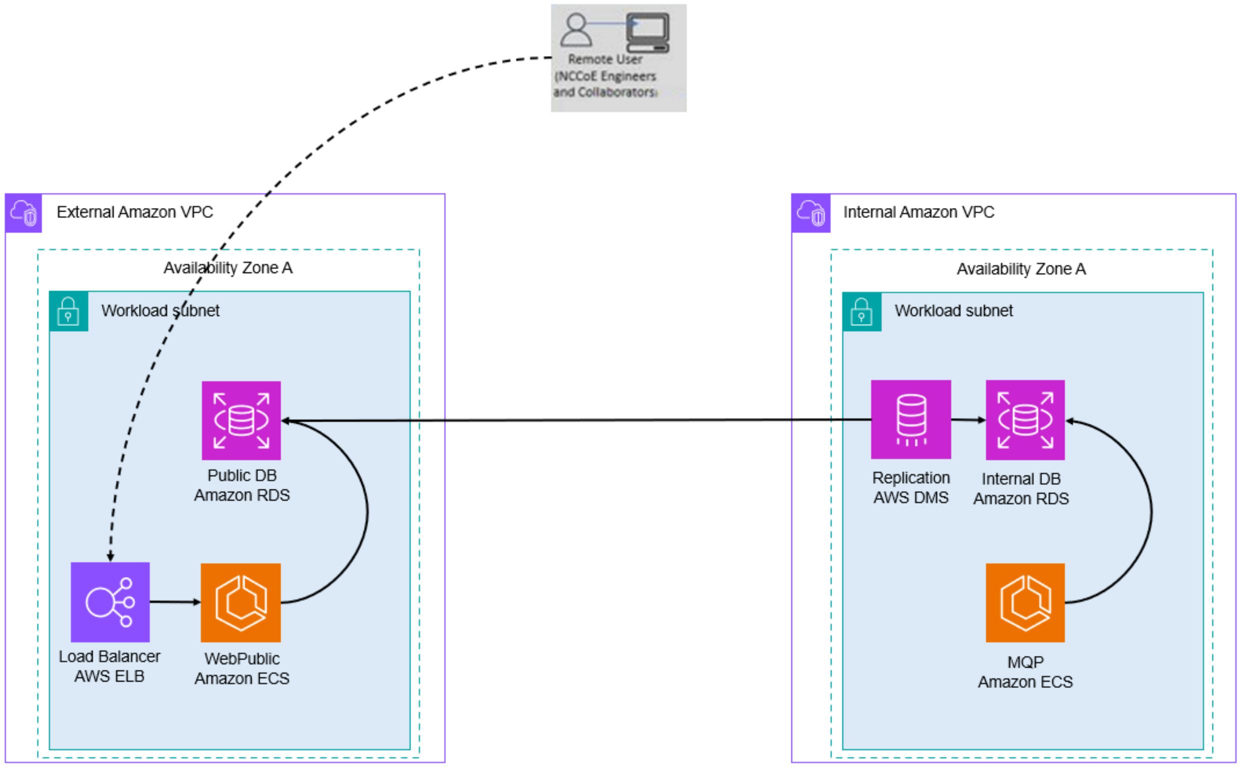
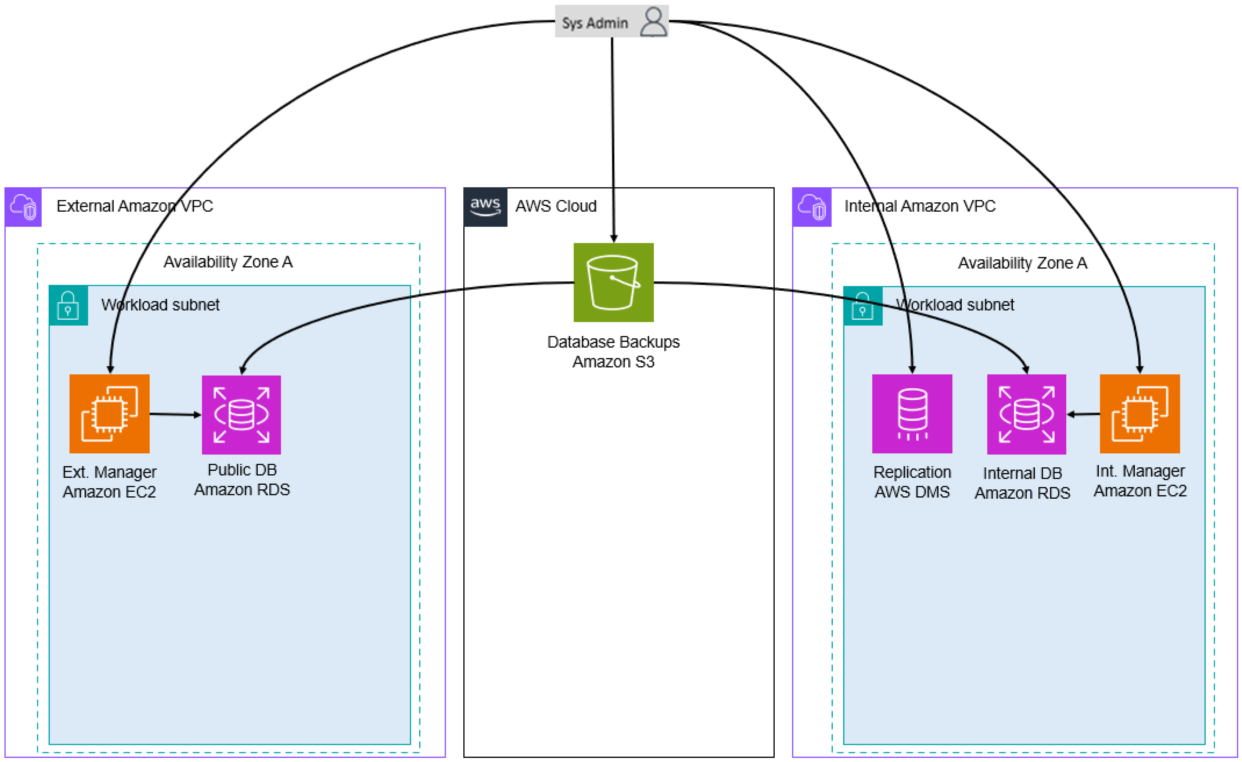


Fig. 7. Modernized Client Workflow

Figures 8 and 9 depict the different workflows the system administrator and the developer take to implement updates to the application code or database.



818 **Fig. 8. Modernized System Administrator Workflow**

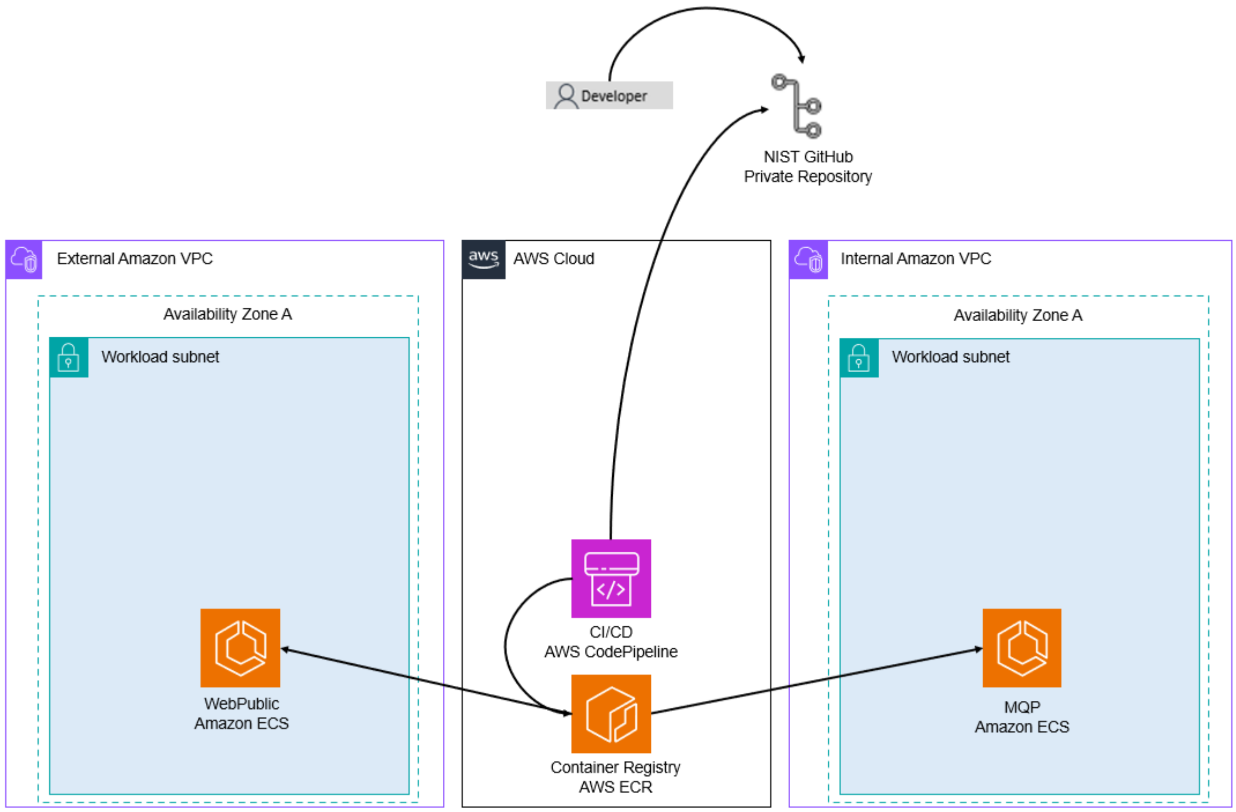


Fig. 9. Modernized Developer Workflow

To make code changes, a developer would push their changes to a code repository, like GitHub. From there, a container build is completed either locally by a system administrator or through the AWS CodePipeline, where a container image is created and stored in the Amazon Elastic Container Registry (ECR). Once those changes are pushed, new tasks can be added (manually or automatically) with the updated application code.

To make database changes, a developer would generate a backup of the database they would like to deploy in the modernized environment. This backup would be given to the system administrator, where the backup is placed into a private Amazon S3 bucket. The system administrator can then connect to a database connector, where the backup can be retrieved from Amazon S3 and deployed into the Amazon RDS instance. This process requires AWS DMS replication to be reinitiated for the new set of desired tables.

4.5. CI/CD Pipeline Modernization with AWS CodePipeline

AWS CodePipeline was used to automate the continuous integration and deployment (CI/CD) process. The pipeline used is structured into multiple stages that ensure code tracking, containerized builds, artifact storage, and automated deployment to AWS services. AWS CodePipeline was only tested while deploying to AWS services.

Source Control & Change Detection – GitHub + AWS CodePipeline: AWS CodePipeline is integrated with GitHub, allowing it to automatically detect new code changes in the repository.

When a developer pushes new code, AWS CodePipeline triggers the pipeline execution, ensuring an automated and streamlined development lifecycle.

Build & Containerization – AWS CodeBuild + Amazon ECR: AWS CodeBuild is used to build Docker containers based on the latest code changes. The build process includes compiling, testing, and packaging the application into containerized images. These images are then tagged and stored securely in Amazon ECR for deployment.

Deployment & Orchestration – AWS CodeDeploy + Amazon ECS: AWS CodeDeploy handles the deployment of containerized applications into Amazon ECS. Amazon ECS ensures that the latest container versions are automatically deployed and scaled across available compute resources.

4.6. Database Modernization

Database modernization focuses on modernizing the hosting environment for the database service. The application requires an internal and external database with replication of data between the two to communicate updated information.

Amazon Relational Database Service (Amazon RDS): The Microsoft SQL Server 2019 edition in the ACMVP demo environment has been replaced with Amazon RDS for SQL Server 2022 with a standard license.

AWS Database Migration Service (AWS DMS): Microsoft SQL Server allows for native data replication in the legacy ACMVP research environment. However, the migration to Amazon RDS necessitates a new data replication service because the underlying resource hosting the database is not owned by the customer but by AWS. AWS DMS maintains replication between the Amazon RDS databases.

4.7. Application Deployment Modernization

The application deployment modernization focuses on containerizing the WebPublic and MQP applications. Utilizing containers provides benefits and options such as blue/green deployments, vulnerability scanning images in a registry in advance of deployments, and less exposure times from routine deployments.

Figure 10 demonstrates the progression of the approaches taken to modernize the application into a container. The markers on the top represent the Microsoft Windows Container while the markers on the bottom represent the Linux Container.

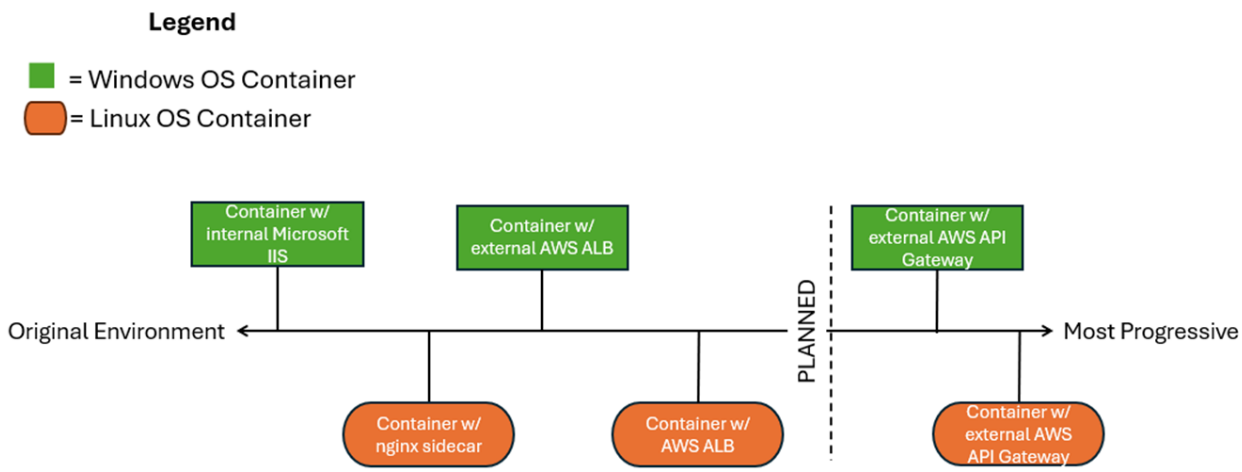


Fig. 10. Progression of Containerization Builds

The closest iteration to the original ACMVP environment is the Microsoft Windows container that encapsulates both the application and the Microsoft IIS proxy to authenticate and route traffic. This solution containerizes the precise environment that exists in the WebPublic Amazon EC2 instance.

The Linux container with an Nginx sidecar advances the environment by offering a smaller container image size and utilizing proxy. It allows for the container or Nginx to be modified without causing the other to be taken offline, decoupling the application.

The AWS ALB lifts the authentication and proxy services into cloud services, which allows AWS ALB to handle the mTLS handshake.

Further research is planned for the Amazon API Gateway, later referred to in the document.

4.7.1. Microsoft Windows Containers

Microsoft Windows containers were the starting point of the research since they run the same OS as the legacy ACMVP infrastructure. Additionally, they allow the use of Microsoft IIS in the container to handle the mTLS handshake for authentication. The applications were successfully containerized and enabled the modernization of the supporting infrastructure. However, there was a limitation with the AWS CodeBuild/CodePipeline integration, which requires docker-in-docker.

4.7.2. Linux Containers

Linux containers do not support Microsoft IIS (where mTLS authentication is handled), which resulted in research for alternative authentication mechanisms. Nginx was found as an open-source solution that can be hosted locally in a container. AWS ALB was found as a cloud solution.

Linux containers support docker-in-docker, required for AWS CodeBuild, which enables streamlined code deployment.

4.7.3. Amazon EC2 Launch

This container launch type utilizes a base Amazon Machine Image (AMI) to launch onto an Amazon EC2 instance. The container runs via docker daemon and is built locally. Network connections are routed through the Amazon EC2 instance to the underlying container.

4.7.4. Amazon ECS Fargate Launch

The serverless Amazon ECS Fargate service provides a hosted platform for containerized tasks and services. Managed components consist of automation around host provisioning and compute monitoring. The end user is responsible for managing Amazon ECS tasks or service definitions that interface with the AWS-provided host through a mixture of AWS Identity and Access Management (IAM) controls, Amazon VPC security groups, and Elastic Network Interface (ENI) allocations.

4.7.5. Amazon ECS with Amazon EC2 Instance Launch

This launch type was identified and will be researched. It allows more granular control of the underlying Amazon EC2 instance hosting the container by the system administrator.

4.7.6. Amazon EKS Fargate and Amazon EKS Auto Mode Launch

The Amazon EKS Auto Mode launch type was identified as part of this research. The team plans to explore this option in earnest following ICMC ‘25. As with the Amazon ECS Fargate launch type, the foundational pieces controlling container workloads are managed and maintained by AWS.

The NCCoE can leverage a majority of the underlying functionality provided by the Kubernetes service stack, such as workload management, security policy enforcement, service discovery, and many others.

As previously mentioned, the Amazon EKS Fargate service provides an AWS-managed solution for containerized workloads, which leverages the automated host provisioning and auto-scaling integration behind the scenes with Amazon EC2. Cluster owners will only manage how defined services and containerized workloads will interface with the underlying host through security groups and ENI mappings.

4.8. Layer 3 Authentication Modernization

4.8.1. Nginx Reverse Proxy

Nginx is a reverse proxy that routes requests to the ACMVP server, similar to the use of Microsoft IIS in the WebPublic application. Nginx supports mTLS authentication, allowing it to verify client certificates before forwarding requests. Nginx in a Linux container maintains robust load balancing, security, and authentication capabilities similar to Microsoft IIS in a Windows container.

4.8.2. AWS Application Load Balancer (ALB)

An AWS Network Load Balancer (AWS NLB) was initially used to route traffic to the containerized application with Microsoft IIS. This architecture was then transitioned to an AWS Application Load Balancer (AWS ALB) because the AWS ALB can handle both the routing to the containerized application and the application-level authentication previously handled by Microsoft IIS.

The AWS ALB completes the mTLS handshake, further decoupling that service from the WebPublic application. Certificate details may be passed on to the application for any further authentication or logging details required.

4.8.3. Amazon API Gateway

Amazon API Gateway is an AWS service for creating, publishing, maintaining, monitoring, and securing REST, HTTP, and WebSocket APIs at any scale. This service allows for a one-to-one layer of connection between the gateway and the ACMVP web app endpoints and enables the development team to provision, distribute, and revoke API keys as an alternative and modern form of authentication for each API request made to the server. In combination with other services like AWS Cognito, labs could manage their own credentials to further improve operational efficiency.

5. Conclusion

To date, the project has:

- Identified and sorted categories of test evidence required for CMVP validation that can readily be automated in a reporting format consistent with current Web CRYPTIK used by CMVP and identified those test evidence classes for which manual processes are still needed;
- Identified necessary schemas and protocols for report submission and validation for a scalable API-based architecture;
- Designed and developed a cloud-based infrastructure required to support validation program automation;
- Added automated rule processing on submissions with instant feedback intended to catch inconsistencies and inaccuracies a CMVP reviewer would otherwise need to catch during their review of a submission and provides instant feedback to the submitter to correct before the submission is;
- Added the source code evidence payloads to capture how source code TEs are evaluated by the lab;
- Added details to the protocol to provide a more complete API for labs to interact with their submissions;
- Defined test methods for functional testing TEs to allow for more specific information and automation to be applied to the evidence collected;
- Improved the TE filtering coverage via thorough review of all sections of FIPS 140-3;
- Modernized infrastructure by migrating legacy systems to a scalable cloud platform, implementing CI/CD pipelines for automation, and containerizing applications for faster, more maintainable deployments;
- Upgraded web servers with cloud-based solutions for routing and authentication, enhanced security with mutual TLS and API keys, and improved system resilience while reducing downtime;
- Streamlined developer workflows, accelerated updates, and minimized operational complexity and infrastructure costs;
- Deployed a demo ACMVP server, enabling the community to explore and get acquainted with the newly developed application;

Moving forward, the project staff plans in the second half of 2025 to:

- Finalize a coordinated JSON structure for test evidence catalogue;
- Refine the research infrastructure to support enabling automated acceptance of test evidence and processing of functional test evidence from NVLAP-accredited parties;

- Streamline test methods for functional testing;
- Improve test requirement filtering capabilities;
- Demonstrate an ability for the CMVP staff to use an API to handle “comment round” interactions with NVLAP-accredited parties;
- Begin integrating ACMVP research outputs into the production CMVP workflows;
- Perform security analysis for the proposed design.

References

[1] National Institute of Standards and Technology (2019) Federal Information Processing Standards Publications (FIPS PUBS) 140-3: Security Requirements for Cryptographic Modules. (National Institute of Standards and Technology, Gaithersburg, MD), NIST. https://doi.org/10.6028/NIST.FIPS.140-3

[2] National Institute of Standards and Technology and National Cybersecurity Center of Excellence, (2022) Automation of the Cryptographic Module Validation Program (National Institute of Standards and Technology, Gaithersburg, MD), NIST, NCCoE. https://www.nccoe.nist.gov/automation-nist-cryptographic-module-validation-program

[3] ISO, ISO/IEC 24759:2017: Information Technology - Security Techniques - Test Requirements for Cryptographic Modules, Geneva, Switzerland: International Organization for Standardization, 2017.

[4] National Institute of Standards and Technology and Canadian Centre for Cyber Security (2025) FIPS 140-3- Cryptographic Module Validation Program Management Manual, Version 2.5. (National Institute of Standards and Technology, Gaithersburg, MD), NIST. https://csrc.nist.gov/Projects/cryptographic-module-validation-program/cmvp-fips-140-3-management-manual

[5] ISO, ISO/IEC 19790:2012: Information Technology - Security Techniques - Security Requirements for Cryptographic Modules, Geneva, Switzerland: International Organization for Standardization, 2012.

Appendix A. List of Symbols, Abbreviations, and Acronyms

140A-TE
Vendor-documentation-dependent Test Evidence

ACMVP/ACVP
Automated Cryptographic Module Validation Project

AD DS
Active Directory Domain Services

ALB
Application Load Balancer

AMVP
Automated Module Validation Program

API
Applications Programming Interface

AS
Assertion

CAVP
Cryptographic Algorithm Validation Program

CCCS
Canadian Centre for Cyber Security

CL
Component List

CMVP
Cryptographic Module Validation Program

CRADA
Cooperative Research and Development Agreement

CSTL
Cryptographic and Security Testing Laboratory

CVE
Common Vulnerabilities and Exposures

DMS
Database Migration Service

ECR
Elastic Container Registry

ECS
Elastic Container Service

EDC
Error Detection Code

EFT
Electrical Fast Transients

EKS
Elastic Kubernetes Service

ESV
Entropy Source Validation

ESVP
Entropy Source Validation Program

FIPS
Federal Information Processing Standards

FSM
Finite State Model

FT
Functional Test

FW
Firmware

HW
Hardware

ICMC
International Cryptographic Module Conference

IEC
International Electrotechnical Commission

IG
Implementation Guidance

ISO
International Organization for Standardization

IUT
Implementation Under Test

MAC
Message Authentication Code

MIS
Module Information Structure

MQP
Message Queue Processor

NCCoE
National Cybersecurity Center of Excellence

NLB
Network Load Balancer

NVLAP
National Voluntary Laboratory Accreditation Program

OD
Other Documents

OTAR
Over the Air Rekeying

OTP
One-time Programmable

RDS
Relational Database Service

S3
Simple Storage Service

SC
Source Code

SP
Security Policy

SQL
Structured Query Language

SSP
Sensitive Security Parameter

SW
Software

TE
Test Evidence

VE
Vendor Evidence

WS
Workstream

Appendix B. CMVP TE Tables

Applicable TEs for each combination of the basic filtering criteria based on TETables_v2.3.03.json developed by the NCCoE ACMVP project team can be found on the ACVMP Documentation website.