Carnegie Mellon University
Software Engineering Institute

INTEGRATING ZERO TRUST AND DEVSECOPS

Geoffrey Sanders
Timothy Morrow
Nathaniel Richmond
Carol Woody, Ph.D.

White Paper

July 2021

Executive Summary

Zero Trust (ZT) and DevSecOps are popular strategies that leverage automation to execute organizational processes and workflows. ZT is a security strategy that uses policy to enforce explicit trust between subjects and resources. DevSecOps is a development strategy that combines tools and agility to continuously develop and operate software. Both strategies are interdependent and require balancing concerns of how services, data, and infrastructure must be shared to achieve efficiency, cost effectiveness, and risk mitigation for continuous authority to operate (cATO). A mission thread which focuses on the lifecycle of an application being developed within a DevSecOps environment is used to provide the context for this discussion.

Abstract

The DevSecOps development strategy is increasingly being used by organizations and programs to improve the quality and timeliness of the product being produced. The DevSecOps pipeline is built with a continuous integration and continuous delivery focus where automation is envisioned to help speed up the development and provide testing of the product. The use of hybrid computing environments to implement and support the DevSecOps strategy adds complexity to this situation.

Over the past year, we are experiencing a higher level of intrusions in these organizations and programs. A security strategy which has come into prominence to combat these intrusions is Zero Trust. This strategy cannot be implemented through the procurement of one or two commercial products. Zero Trust is a journey that can mature over time. The question we are addressing in this paper is, “How can you reason about Zero Trust considerations for a DevSecOps pipeline and associated ecosystem in a hybrid computing environment?”

We have created an application development mission thread which provides context for an application’s lifecycle as it would go through a DevSecOps pipeline. The DevSecOps phases are used to break the mission thread steps up to understand the actors and their actions in each step. Using this information,

Zero Trust considerations are identified for each phase. These can be used by organizations and programs which are transitioning to a Zero Trust implementation within their hybrid computing environment to be able to better make tradeoff decisions concerning their implementations.

1 Introduction

Digital modernization and transformation initiatives are driving organizational culture, infrastructure, and operations change. Executing these initiatives requires a strategy that communicates its vision, goals, objectives, priorities, and elements. Priorities common across strategies today include cybersecurity, artificial intelligence (AI), and cloud computing. They also share common goals of innovation, efficiency, agility, and evolution. Defense and industry are leveraging two security models to achieve these strategic goals and priorities: DevSecOps and Zero Trust (DoD 2019).

Organizations should understand both models, how they work together to improve cybersecurity, and how they must adapt to incorporate them into their processes and workflows. For this paper, we provide a high-level overview of a DevSecOps pipeline and its phases. A mission thread is used to provide the context for developing an application in this environment. Our focus is to identify Zero Trust considerations that organizations will need to address as they go through the seven DevSecOps phases (Plan, Develop, Build, Test, Release and Deliver, Deploy, and Operate). [DoD 2021e]

2 Zero Trust

Zero Trust is a cybersecurity strategy focused on protecting networked resources. The concept was originally developed by John Kindervag at Forrester Research in 2009 to mitigate targeted and malicious insider attacks. Network-centric solutions didn’t mitigate these attacks when perimeter security was breached and a different approach was required [Kindervag 2016].

Within a Zero Trust strategy, networked resources are any system, data, or application accessed by subjects. Subjects are humans or non-person entities (NPEs), such as IoT (Internet of Things) devices or application containers. Policy decision and enforcement points control subject access to resources, while dividing access areas into untrusted and implicit trust zones. Figure 1 provides an overview of these concepts [NIST 2020].

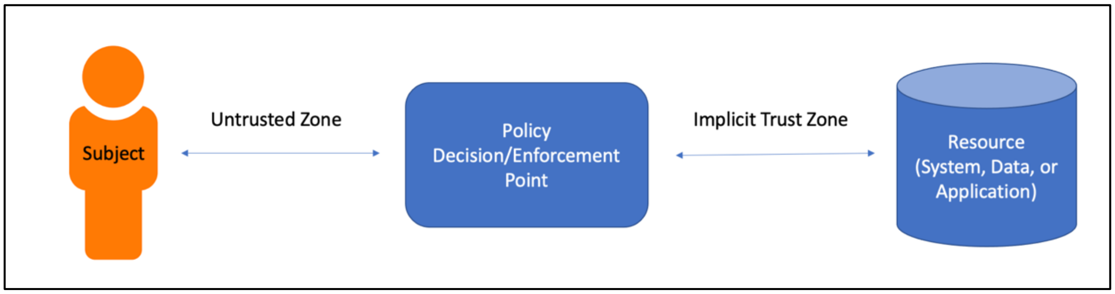

Subject
Untrusted Zone
Policy Decision/Enforcement Point
Implicit Trust Zone
Resource (System, Data, or Application)

Figure 1: Zero Trust Networked Resources

Two perspectives are commonly used to interpret and implement a Zero Trust strategy: principles and platform. Principles are abstract characteristics that form the strategy. Platform is the abstract system that implements the strategy. Both are important, inform each perspective, and must remain consistent.

2.1 Principles

Three core and expanded principles comprise Zero Trust [Garbis 2021]. Core principles required for every Zero Trust implementation are:

- Ensure all resources are accessed securely, regardless of location.
- Inspect and log all traffic.
- Adopt a least privilege strategy and strictly enforce access control.

Core principles assume mature organizations that inventory, manage, and monitor subjects and resources. Subjects are provisioned access to resources, access policy is dynamic, and all traffic is encrypted, logged, and monitored. Trust between subjects and resources is never assumed or implied.

Expanded principles, equally important and required for any enterprise-class Zero Trust implementation are:

- Ensure all components support application programming interfaces (APIs) for event and data exchange.
- Automate actions across environments and systems, driven by context and events.
- Deliver tactical and strategic value.

Expanded principles assume a high-performing, data-driven organization that develops systems, architecture, applications, and APIs. Systems are integrated, dynamically monitoring the environment, and updating access policy as factors change. The organization must be agile, define performance, and measure outcomes for tactical and strategic decision making.

2.2 Platform

A Zero Trust platform is an abstract system that implements organizational security strategy with components and capabilities. Generally viewed as an enterprise solution, Zero Trust comprises seven components: User, Device, Network/Environment, Application and Workload, Data, Visibility and Analytics, and Automation and Orchestration. Each component is a key system area required to implement the platform and its controls. The Visibility and Analytics component provides the information necessary to develop a situational awareness of the Zero Trust implementation collected from the User, Device, Network/Environment, Application & Workload, and Data components. This information is used to identify potential threats which will result in dynamic changes to the security policy and access decisions. (see Figure 2).

To make the dynamic and real-time aspects of the Zero Trust implementation a reality, the Automation and Orchestration component is responsible for automating, consistent security responses across the enterprise/system through the use of security orchestration, automation and response (SOAR), security information and event management (SIEM) and other automated security tools. All components implement encryption and telemetry-based access controls [DoD 2021a].

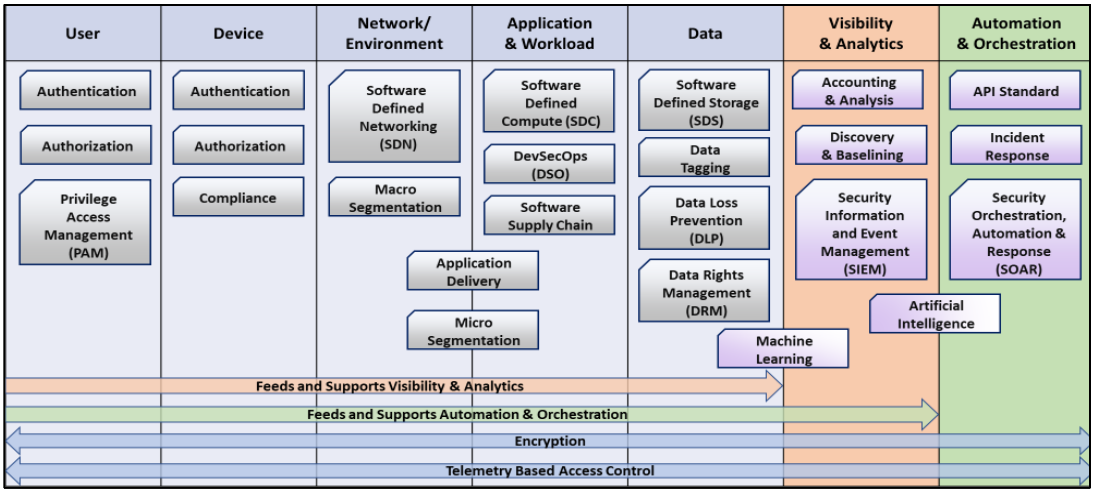

User
Device
Network/ Environment
Application & Workload
Data
Visibility & Analytics
Automation & Orchestration
Authentication
Authorization
Privilege Access Management (PAM)
Authentication
Authorization
Compliance
Software Defined Networking (SDN)
Macro Segmentation
Application Delivery
Micro Segmentation
Software Defined Compute (SDC)
DevSecOps (DSO)
Software Supply Chain
Software Defined Storage (SDS)
Data Tagging
Data Loss Prevention (DLP)
Data Rights Management (DRM)
Machine Learning
Accounting & Analysis
Discovery & Baselining
Security Information and Event Management (SIEM)
Artificial Intelligence
API Standard
Incident Response
Security Orchestration, Automation & Response (SOAR)
Feeds and Supports Visibility & Analytics
Feeds and Supports Automation & Orchestration
Encryption
Telemetry Based Access Control

Figure 2: Zero Trust Components and Capabilities

Zero Trust also comprises more than 20 capabilities. Capabilities are platform activities and resources required to perform a specified course of action, such as user and device authentication and authorization. Some capabilities, such as Machine Learning and Artificial Intelligence, span two components.

While a universal set of standards to implement a Zero Trust platform doesn't currently exist, multiple organizations are working towards this goal. IEEE established a Zero Trust Security Working Group (P2887) mid-2020 to develop recommended security guidance [IEEE 2020]. NIST is actively working on developing Zero Trust standards dealing with architecture, microservices, access control, and DevSecOps. This information can be located with a "zero trust" topic search [NIST 2021]. NIST also established an NCCOE (National Cybersecurity Center of Excellence) Zero Trust Architecture Community of Interest late in 2018 to actively build out an example architecture [NCCOE 2018].

The most comprehensive set of published standards, the US DoD (United States Department of Defense) *Zero Trust Reference Architecture* contains Profile (StdV-1) and Forecast (StdV-2) standards for DoD systems [DoD 2021a]. Profile standards list technology, laws, regulations, policies, and tactics, techniques and procedures for Zero Trust components. Forecast standards list technology, operations, and business standards and conventions for Zero Trust capabilities.

3 DevSecOps

DevSecOps is a collaborative software development strategy that integrates development (Dev), security (Sec), and operations (Ops) practices into one lifecycle. It improves software quality, efficiency, and security by combining teams, culture, infrastructure, and Agile development practices. Previous development strategies employed siloed teams and longer, disconnected lifecycles. When security problems occurred, the actions to correct them involved long time delays and significant expense to fix. DevSecOps addresses those challenges by introducing features, patches, and fixes early and frequently in the lifecycle.

DevSecOps is commonly viewed from three different perspectives: lifecycle, platform, and product. The lifecycle perspective views the strategy from its eight progressing software development stages. The platform perspective views DevSecOps from the tools used to implement the lifecycle, while the product perspective is the application developed with the platform. Each perspective provides a unique lifecycle view to consistently implement DevSecOps and integrate it with zero trust.

Zero Trust provides DevSecOps platform security infrastructure such as ICAM (identity, credential, and access management), SIEM (security information and event management), and SOAR (security orchestration, automation, and response). [DoD 2021a] This enables faster time to implementation, integrated security, and broader situational awareness. DevSecOps is also a Zero Trust capability that supports 11 operational activities and five services. These listings are available in DoD ZTRA (Zero Trust Reference Architecture) CV-6 and CV-7 capability views [DoD 2021a].

3.1 Lifecycle

The DevSecOps lifecycle perspective views the development strategy from its eight stages and is commonly represented by *infinity* and *unfolded* diagrams. The infinity diagram (Figure 3) depicts DEVSECOPS as an infinite lifecycle with integrated security practices. The left side of the diagram contains the lifecycle development stages and their integrated security practices, while the right side of the diagram contains the operations stages and their respective security practices.

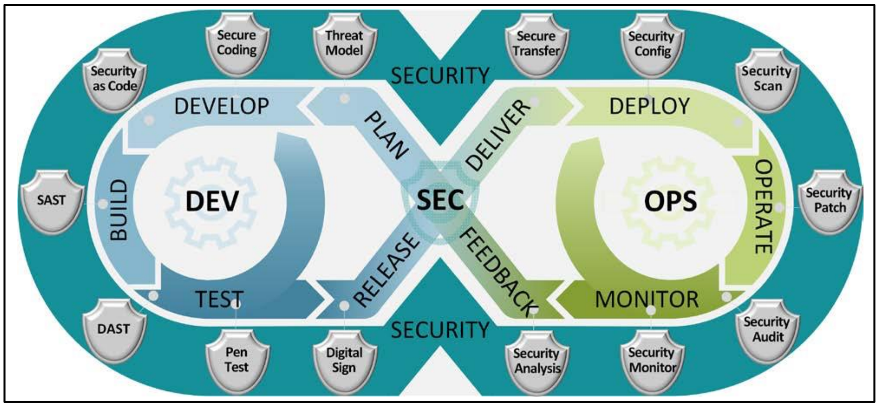


Figure 3: DevSecOps Infinity Software Lifecycle [DoD 2021b]

The unfolded diagram (Figure 4) depicts the infinity diagram as it would be in an unfolded state. This view emphasizes the occurrence of monitoring, cybersecurity automation, control gates, risk determination, and feedback loops in the lifecycle. Cybersecurity automation, monitoring, and risk determination occur at each phase of the lifecycle. Build, integration, delivery, and deployment feedback loops occur at specific phases of the lifecycle. In Figure 4, the feedback loops do not show an actual looping back in the phases. It just shows a flat bar of the stages included in each feedback loop. Continuous build feedback occurs at the develop and build phases, while continuous integration feedback occurs at the develop, build, and test phases. Continuous delivery feedback occurs at the plan, develop, build, test, and release and deliver phases, while continuous deployment feedback includes the deploy phase. Control gates occur between the develop, build, test, release and deliver, and deploy phases. A control gate is a checkpoint in the process where a checklist of stage-specific goals are reviewed to see if the stage has accomplished this and can move to the next stage or not.

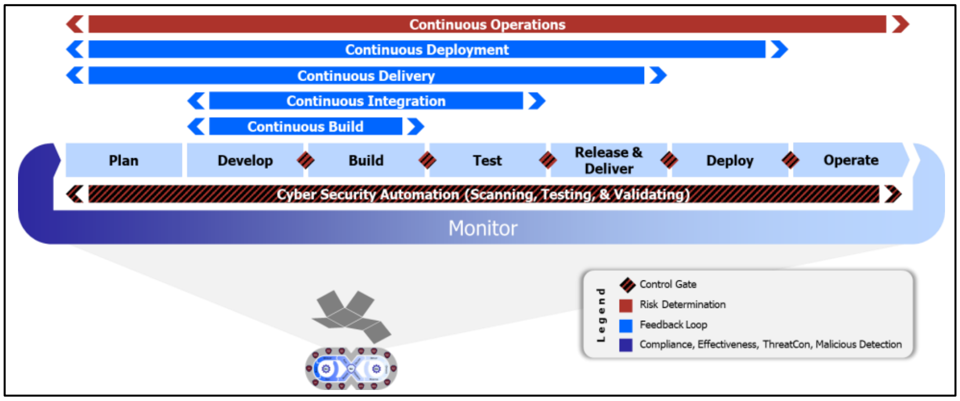


Figure 4 DevSecOps Unfolded Software Lifecycle [DoD 2021d]

3.2 Platform

The DevSecOps platform perspective views the development strategy from the components used to implement the lifecycle. Components include applications, workflows, tools, software factory, services, and infrastructure. Figure 5 shows how platform tools integrate with people, infrastructure and a CI/CD (continuous integration/continuous delivery) orchestrator to develop, test, and release applications with through workflows. Some of the workflows provide CM (configuration management) verification and audit check capabilities, while a few tools provide CM control capabilities. Figure 6 depicts the interaction of software factory tools, workflows, and infrastructure services to produce applications. In this diagram, code repository and artifact services provide CM control capabilities, while development testing and release and deliver workflows provide CM verification and audit capabilities.

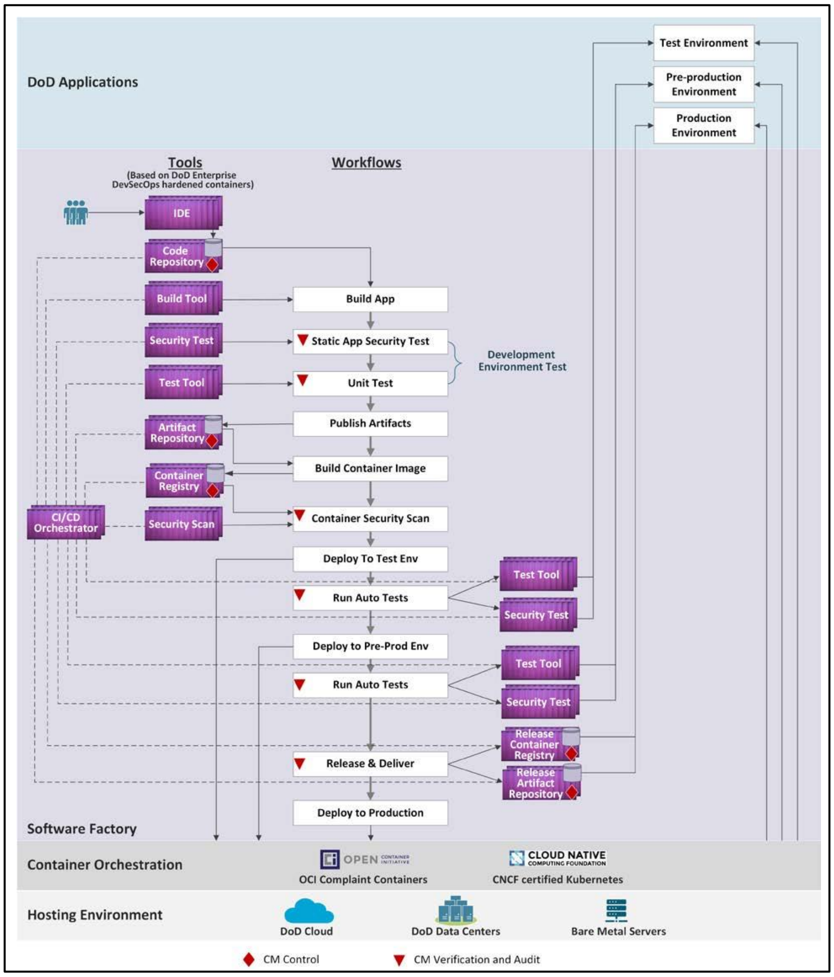


Figure 5 Notional Expansion of a DevSecOps Pipeline [DoD 2021b]

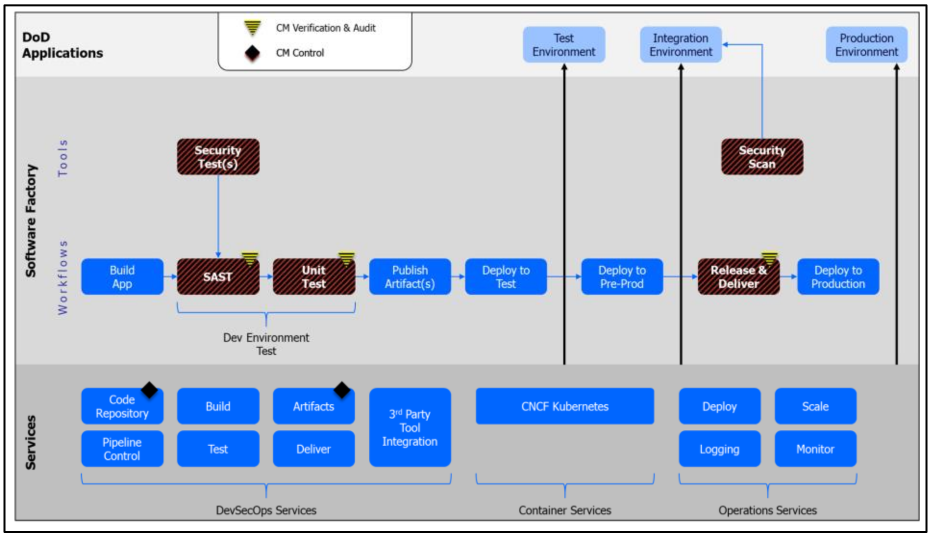


Figure 6 Software Factory - DevSecOps Services [DoD 2021c]

3.3 Product

With a basic understanding of Zero Trust and DevSecOps strategies, we have developed a mission thread which we use to set context for developing an application in DevSecOps pipeline and identify some assumptions which will help use identify some Zero Trust considerations in this process. A mission thread is a sequence of end-to-end activities and events, given as a series of steps, that accomplish the execution of one or more capabilities that a system-of-systems (SoS) supports. A SoS is integrated to accomplish a number of missions that involves cooperation among individual systems. The mission thread helps tell a story of creating an application and gaining an understanding of the actions, and the people who have to perform those actions. The mission thread involved a DevSecOps pipeline provides the opportunity to add some "interesting twists" like having a third party develop the application only using the environment and supporting development artifacts provided to it.

In this paper, small business provider MTECH is contracted to develop specialized situational awareness (SA) dashboards for Bank1's cybersecurity operations center (CSOC). MTECH must use artifacts from BANK1's repository to create and update dashboards in a software factory using a DevSecOps pipeline. The repository contains reference software that is hardened, version controlled, and patched by BANK1. MTECH combines these artifacts with the DevSecOps lifecycle and platform to develop SA dashboards.

4 Mission Thread

Mission threads are sequences of end-to-end activities and events that take place to accomplish the execution of SoS (system of systems) capabilities [Gagliardi 2013]. This paper uses an application development mission thread for a fictional bank to understand DevSecOps and Zero Trust interaction.

BANK1 has developed a DevSecOps ecosystem to improve the cybersecurity of applications developed for use on its CSOC. The ecosystem will be used by BANK1 staff, as well as third-party application developers. The ecosystem contains an artifact repository that holds hardened virtual machine (VM) images and hardened Open Container Initiative (OCI) compliant container images of: DevSecOps tools, container security tools, common program platform components, custom code, build process, configuration files, and applications. VM is a computer resource that uses software instead of a physical computer to run programs and deploy applications. OCI is working to develop open industry standards around container formats and runtime. A container is a standard unit of software that packages up code and all its dependencies so that the application runs quickly and reliably from one computing environment to another. BANK1 is providing the cloud infrastructure that developers work in and is implementing Zero Trust principles into the DevSecOps ecosystem to improve cybersecurity.

MTECH, a small business that specializes in situational awareness (SA) dashboards, is contracted to create applications for BANK1's CSOC. MTECH is required to use artifacts from BANK1's repository to create and update SA dashboards in a software factory process. This thread will be used to explore the impacts of implementing zero trust tenets to the DevSecOps ecosystem.

The following sections break the application development mission thread into DevSecOps lifecycle phases. At the end of each section, a table of Zero Trust considerations and the action responses which are involved is provided. The goal of this information is to spawn the readers thoughts concerning additional possible Zero Trust considerations.

5 Mission Thread Plan

The DevSecOps plan phase sets a foundation for the remaining phases with activities that help manage time, cost, quality and risk. Activities include requirements gathering and analysis, gap and risk analysis, and process definition. Multiple assumptions also accompany this phase that impact successful BANK1 and MTECH engagement.

One assumption is that BANK1 understands data and how it flows in their environment. Data, assets, applications, and services (DAAS) are categorized and managed by mission criticality. Technical and policy controls implement data management, while data at rest and in-transit is monitored to identify and mitigate theft.

Another assumption is BANK1 has an operational zero trust infrastructure in place. Identity, credential, and access management (ICAM) is used to provision, manage, authenticate, and authorize subject access to resources throughout the environment. ICAM, combined with public key infrastructure (PKI), micro- and macro-segmentation, and other technical controls enforce least-privilege policy and prevent lateral movement. Subject/resource communications are visible, baselined, and monitored by security event and information management (SIEM) to identify events of interest. Machine learning (ML), artificial intelligence (AI) and security orchestration automation and response (SOAR) platforms work together to integrate information and respond to incidents with standardized application programming interfaces (APIs).

Governance, risk, and compliance (GRC) is also assumed. Processes and procedures are established, documented, and practiced. The software factory provides reference software and templates to standardize development, configuration, and maintenance [DoD 2021c]. This includes containers and other reusable components that standardize implementing ZT services during application development such as authentication, authorization, and logging. The software factory is also monitored, measured and remediated to mitigate risk and enable achieving the goal of a continuous authority to operate (cATO). Authority to Operate (ATO) is a process that certifies a system to operate for a certain period of time by evaluating the risk of the system’s security controls. cATO is an on-going authorization which deals with ongoing understanding and acceptance of security and privacy risk. cATO is focused on transparently defined and well understood continuous monitoring program.

BANK1 and MTECH enter the plan phase after completion of contractual agreements. BANK1 provides MTECH onboarding instructions that include user and device provisioning and token requirements. Roles and policies are defined in the mission thread for the BANK1 environment in order to properly provision subjects to resources needed to complete work, including PKI certificate generation and use.

Onboarding also provides all documentation necessary for MTECH to begin application development with the platform and ZT infrastructure. Documentation includes the current state of DevSecOps pipeline and ZT infrastructure, transition roadmaps, and stakeholders that can answer MTECH infrastructure questions.

MTECH then begins engagement planning. ZT and application requirements are gathered from stakeholders and used to develop a roadmap. The roadmap is developed using an agile process.

Table 1: Mission Thread DevSecOpsS Plan Phase ZT Considerations

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Plan	Data, assets, applications, and services (DAAS) are categorized and managed by mission criticality.	BANK1 infrastructure team, MTECH developer team.
Plan	Technical and policy controls implement data management.	BANK1 infrastructure team, MTECH developer team.
Plan	Data at rest and in-transit is monitored to identify and mitigate theft.	BANK1 infrastructure team, MTECH developer team
Plan	Identity, credential, and access management (ICAM) is used to provision, manage, authenticate, and authorize subject access to resources throughout the environment.	BANK1 infrastructure team, MTECH developer team
Plan	ICAM, combined with public key infrastructure (PKI), micro- and macro-segmentation, and other technical controls enforce least-privilege policy and prevent lateral movement.	BANK1 infrastructure team, MTECH developer team
Plan	Subject/resource communications are visible, baselined, and monitored by security event and information management (SIEM) to identify events of interest.	BANK1 infrastructure team, MTECH developer team
Plan	Machine learning (ML), artificial intelligence (AI) and security orchestration automation and response (SOAR) platforms work together to integrate information and respond to incidents with standardized application programming interfaces (APIs).	BANK1 infrastructure team, MTECH developer team

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Plan	The software factory provides reference software and templates to standardize ZT concepts, configuration, and maintenance. This includes containers and other reusable components that standardize implementing ZT services during application development such as authentication, authorization, and logging.	BANK1 infrastructure team, BANK1 repository administrator, MTECH developer.
Plan	The software factory is also monitored, measured and remediated to mitigate risk and enable continuous authority to operate (cATO).	BANK1 administrator, MTECH developer.
Plan	Roles and policies are defined in the BANK1 environment in order to properly provision subjects to resources needed to complete work, including PKI certificate generation and use.	BANK1 infrastructure team.
Plan	Onboarding provides all documentation necessary for MTECH to begin application development with the platform and ZT infrastructure. Documentation includes the current state of ZT infrastructure, transition roadmaps, and stakeholders that can answer MTECH infrastructure questions.	BANK1 infrastructure team.
Plan	ZT and application requirements are gathered by MTECH from stakeholders and used to develop a roadmap.	BANK1 stakeholders, MTECH developer.

6 Mission Thread Develop

The DevSecOps develop phase uses software development tools to convert requirements from planning into source code. Source code includes application code, test scripts, infrastructure as code (IaC), security as code (SaC), workflow scripts, and other code artifacts. Integrated development environments (IDEs) and plugins are development aids used to enhance productivity and improve quality during this phase. Code completion, templating, styling, build, and debugging automation increase efficiency and reduce defects. Security plugins analyze code during development to enforce security policy and compliance, while identifying suspicious and sensitive content [DoD 2021a].

For this phase, the following assumptions are identified. Development tools integrate natively with ZT infrastructure services to build a secure entry point for code artifacts. Policy decision points (PDPs) generate dynamic policy by combining ICAM, enterprise policy, and external data such as threat intelligence, compliance, SIEM, and SOAR. Policy enforcement points (PEPs) use policy to enable, monitor, and terminate connections between subjects and resources [NIST 2020]. The PEPs have been implemented by BANK1 and operates under PDP controls. A ticket management system (TMS) is part of the BANK1 ecosystem and automates communication between both organizations.

MTECH combines development tools with artifacts from the BANK1 central repository for its SA application development. Hardened OCI (Open Container Initiative) compliant containers, BANK1 security stacks, software libraries, templates, and other artifacts streamline development and reduce risk. Repository artifacts are version controlled, scanned, and patched to remediate vulnerabilities and centralize BANK1 ZT software and tenets.

MTECH also verifies its current understanding of the ZT infrastructure and DevSecOps platform during this phase. Source code incorporates the BANK1 ZT security stack, APIs, infrastructure strategy, and roadmaps. MTECH code adopted from outside engagements is authorized and configuration managed by BANK1 before incorporation into the central repository. Standards and metrics are developed and measured to verify the application meets or exceeds BANK1 requirements as it interoperates with the ZT infrastructure.

MTECH uses the TMS to record and report progress and issues to BANK1 stakeholders. BANK1 owns and maintains the ZT infrastructure, central repository, and DevSecOps environment and responsively reports and addresses MTECH issues. To accomplish this, BANK1 incorporates MTECH reporting into its ZT infrastructure and DevSecOps Epics for internal decision making. In turn, BANK1 reports and tracks MTECH related incidents with the TMS.

Table 2: Mission Thread DevSecOps Develop Phase ZT Considerations

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Develop	Development tools integrate natively with ZT infrastructure services to build a secure entry point for code artifacts.	Pipeline administrator, BANK1 infrastructure team, MTECH developer.

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Develop	Policy decision points (PDPs) generate dynamic policy by combining ICAM, enterprise policy, and external data such as threat intelligence, compliance, SIEM, and SOAR.	Pipeline administrator, BANK1 infrastructure team.
Develop	Policy enforcement points (PEPs) use policy to enable, monitor, and terminate connections between subjects and resources.	BANK1 infrastructure team.
Develop	MTECH combines development tools with artifacts from the BANK1 central repository in this phase for SA application development.	BANK1 repository administrator, pipeline administrator, MTECH developer.
Develop	Source code incorporates the BANK1 ZT security stack, APIs, infrastructure strategy, and roadmaps.	BANK1 repository administrator, pipeline administrator, MTECH developer, BANK1 infrastructure team.
Develop	MTECH code adopted from outside engagements is authorized and configuration managed by BANK1 before incorporation into the central repository.	BANK1 repository administrator, MTECH developer.
Develop	Standards and metrics are developed and measured to verify the application meets or exceeds BANK1 requirements as it interoperates with the ZT infrastructure.	BANK1 stakeholders, MTECH developer.
Develop	BANK1 owns and maintains the ZT infrastructure, central repository, and DEVSECOPS environment and responsively reports and addresses MTECH issues.	BANK1 infrastructure team, MTECH development team.
Develop	BANK1 incorporates MTECH reporting into its ZT infrastructure and DEVSECOPS Epics for internal decision making.	BANK1 infrastructure team, MTECH development team.
Develop	BANK1 reports and tracks MTECH related incidents with the TMS. The TMS is part of the BANK1 ecosystem and automates communication between both organizations.	BANK1 infrastructure team, MTECH development team.

7 Mission Thread Build

The DevSecOps build phase uses tools and automation to build and package applications, services, and microservices from source code and other artifacts. Activities in this phase include application build and compiling, static application security tests or scans, dependency vulnerability checks, containerization, release packaging, artifact storage, and build configuration control and auditing. Common tools include build automation, source code linters, container builders, artifact repositories, Static Application Security Test (SAST), and software bill of materials (SBOM) checking tools [DoD 2021b].

Build tools and automation integrate ZT security services and tenets into the application. Some tenets and security services originate from artifacts in BANK1's central repository, while others originate from MTECH's development and build processes. MTECH integrates BANK1's common security stack from the central repository to incorporate ZT tenets, services, and APIs such as ICAM, logging, and behavior protection. MTECH develops security templates for their application that meet BANK1 requirements and configuration management guidelines.

BANK1's CI/CD (continuous integration/continuous deployment) orchestrator uses NPEs (non-person entities) and PKI (public key infrastructure) to restrict human access to the build process. SAST and DAST (dynamic application security testing) tools are integrated with the CI/CD pipeline and perform security testing on the SA dashboard application source code. MTECH iterates the build process by developing code, performing SAST/DAST, dependency checking, and other testing. The build phase also integrates control gates such as ZT compliance checks to identify and mitigate risks before moving to the test phase. BANK1 reports risks to MTECH for review and mitigation.

BANK1's ZT infrastructure monitors build activities for anomalous behavior and enforces policy by restricting subject and resource interaction. NPEs authenticate and authorize automated build activities with PKI, while ZT infrastructure monitors data at rest and in transit and dynamically updates policy to mitigate risk. BANK1 SOAR playbooks automate incident response activities and report build incidents to MTECH.

Table 3: Mission Thread DevSecOps Build Phase ZT Considerations

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Build	Build tools and automation integrate ZT security services, tenets, and APIs into the SA dashboard application.	BANK1 repository administrator, MTECH developer.
Build	CI/CD orchestrator uses NPEs and PKI for access control.	Pipeline administrator, ICAM administrator, PKI administrator, BANK1 repository administrator, MTECH developer.

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Build	Security templates enforce compliance and configuration management.	Pipeline administrator, BANK1 repository administrator, MTECH developer.
Build	Build logs are monitored to identify events, alerts, and anomalies.	SIEM administrator, MTECH developer, pipeline administrator.
Build	Data at rest and in transit is monitored and policy is dynamically updated to mitigate risk.	SIEM administrator, SOAR administrator, ZT policy administrator.
Build	Control gates such as ZT compliance checks are used to mitigate risks before they propagate to the DEVSECOPS test phase.	Pipeline administrator, BANK1 repository administrator, MTECH developer.
Build	BANK1 SOAR playbooks automate incident response activities and report build incidents to MTECH.	MTECH developer, pipeline administrator, SOAR administrator.

8 Mission Thread Test

The DevSecOps test phase supports continuous testing across the software development lifecycle and requires a significant portion of tests to be automated. Automated tests commonly used in this phase include unit, integration, system, functional, regression, acceptance, and performance tests. Multiple tools and activities are combined to execute each test type in this phase and they must integrate with BANK1's zero trust architecture (ZTA) [DoD 2021b].

Successful builds are automatically deployed to the test environment for manual and automated testing. The test environment may provide operational devices and services, or others that emulate BANK1's environment. A staging environment may also be required for testing devices and services not available in BANK1 infrastructure. Staging environments hosted outside of BANK1 must interoperate with BANK1's ZTA.

User acceptance testing (UAT) is typically performed via manual testing, where people use the application to identify issues or improvements. Automated testing includes security scans, test, and compliance checks on the SA dashboard application. Manual and automated test cases should define ZT subjects, resources, and policies while verifying components and capabilities. For example, human and NPE subject personas should be created and provisioned for resources. These personas are then incorporated into tests of malicious and non-malicious behavior that validate ZT components and capabilities. It is assumed that MTECH will not perform administrative functions on BANK1 infrastructure, requiring MTECH and BANK1 to collaborate and identify appropriate personas, tests, and configuration management necessary for implementation.

The testing phase comprises three stages: development, system, and pre-production. The development test stage is comprised of unit and SAST testing. The system test stage includes DAST, integration tests, and system tests. The pre-production test stage incorporates manual security, performance, regression, and acceptance tests, while enforcing container policy and conducting compliance scans.

Data and infrastructure during this phase are labeled for ZT to determine context for dynamic policy and decision making. The iterative nature of this phase also generates volatile data that can impact ZT Visibility and Analytics and Automation and Orchestration components. BANK1 and MTECH must balance risks and threats unique to this phase with detection and mitigation capabilities.

MTECH iterates back to the build process to eliminate or mitigate defects. Integration testing is performed on the SA dashboard until it is deemed ready for release into production. Once the SA Dashboard is ready for production, NBD moves to the Release and Deliver phase.

Table 4: Mission Thread DevSecOps Test Phase ZT Considerations

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Test	The test environment may provide operational devices and services, or others that emulate BANK1's environment.	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Test	Staging environments (internal and externally hosted) must interoperate with BANK1's ZTA.	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Test	Manual and automated test cases should define ZT subjects, resources, and policies while verifying components and capabilities.	MTECH developer.
Test	Data and infrastructure during this phase are labeled for ZT to determine context for dynamic policy and decision making.	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Test	The iterative nature of this phase also generates volatile data that can impact ZT Visibility and Analytics and Automation and Orchestration components.	BANK1 infrastructure team, pipeline administrator, MTECH developer.

9 Mission Thread Release and Deliver

The DevSecOps release and deliver phase sets a production deployment milestone for releasing an application. Analysis tools and data are used to review the release, verify the implementation reflects the design, assess potential threats, and reason about interoperability and performance. Analysis data and results are stored. The application is ready for deployment and provides data for ZT infrastructure to generate events, alerts, and anomalies for dynamic policy development and enforcement.

It is assumed that new information, interfaces, and services are developed to support ZT Visibility and Analytics and Automation and Orchestration components are implemented in this phase. New security templates and controls are also deployed with the SA dashboard application. Supporting templates, controls, and other information are released as a separate repository artifact(s). Examples include log formats and descriptions, detection signatures, and algorithms needed for ZT components and capabilities. Based on BANK1's implementation, this information would be identified in the contract with MTECH. Identification of these artifacts is an area that will need continued effort to define as more experience is gained.

The completion of this phase acts as a control gate to release the SA dashboard to BANK1's repository. The control gate can be manual, automated, or a hybrid approach. To accomplish this, BANK1 CSOC staff work with quality assurance and MTECH technical support to review and mitigate DevSecOps and

ZT risk. Build and test phase data is reviewed to verify and accept the SA dashboard application for release. This process is documented and refined in order to automate the release of most applications. Automation of the release assumes a deep understanding of the data, environment, subjects, resources, and their interactions for release and deliver. MTECH must have access to the same information to address questions about or provide revisions for the SA dashboard application and this should be accounted for in the pipeline.

Table 5: Mission Thread DevSecOps Release and Deliver Phase ZT Considerations

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Release and Deliver	New information, interfaces, and services developed for ZT Visibility and Analytics and Automation and Orchestration components are implemented in this phase.	BANK1 infrastructure, BANK1 stakeholders, MTECH developer.
Release and Deliver	The SA dashboard application provides data for ZT infrastructure to generate events, alerts, and anomalies for dynamic policy development and enforcement.	BANK1 infrastructure, MTECH developer.
Release and Deliver	New security templates and controls are deployed with the SA dashboard application.	BANK1 infrastructure, BANK1 repository administrator, MTECH developer.
Release and Deliver	Supporting templates, controls, and other information are released as a separate repository artifact(s)	BANK1 infrastructure, BANK1 repository administrator, MTECH developer.
Release and Deliver	BANK1 CSOC staff work with quality assurance to review and mitigate DEVSECOPS and ZT risk.	BANK1 infrastructure, BANK1 stakeholders, MTECH developer.
Release and Deliver	Automation of the release assumes a deep understanding of the data, environment, subjects, resources, and their interactions for release and deliver.	BANK1 infrastructure, BANK1 stakeholders, pipeline administrator, MTECH developer.
Release and Deliver	Least privilege is applied to analysis data and results.	BANK1 infrastructure, BANK1 stakeholders, MTECH developer.

10 Mission Thread Deploy

The DevSecOps deploy phase releases the SA dashboard application into production. This phase uses load balancers, containers, Infrastructure as Code (IaC), and a blue/green deployment strategy to build the new application alongside the operational production environment. The environment then switches to the new application release without outage. The cATO process verifies that the SA dashboard application is production ready and authority to operate is approved.

Load balancers sit in front of blue (active) and green (staging) environments that run applications in containers (Figure 7). The blue environment represents the current application production release. All application requests are routed by the load balancer to the blue environment. The DevSecOps pipeline builds the new release version and deploys it to the green environment. BANK1 and MTECH monitor the green environment application for production stability. When the application is considered stable, load balancers are reconfigured to switch application requests from the blue to green environment. The blue environment remains active for a specified period during monitoring until a decision is made to retire the old application version. When the blue environment is destroyed, the green environment becomes the release version and is recategorized as blue.

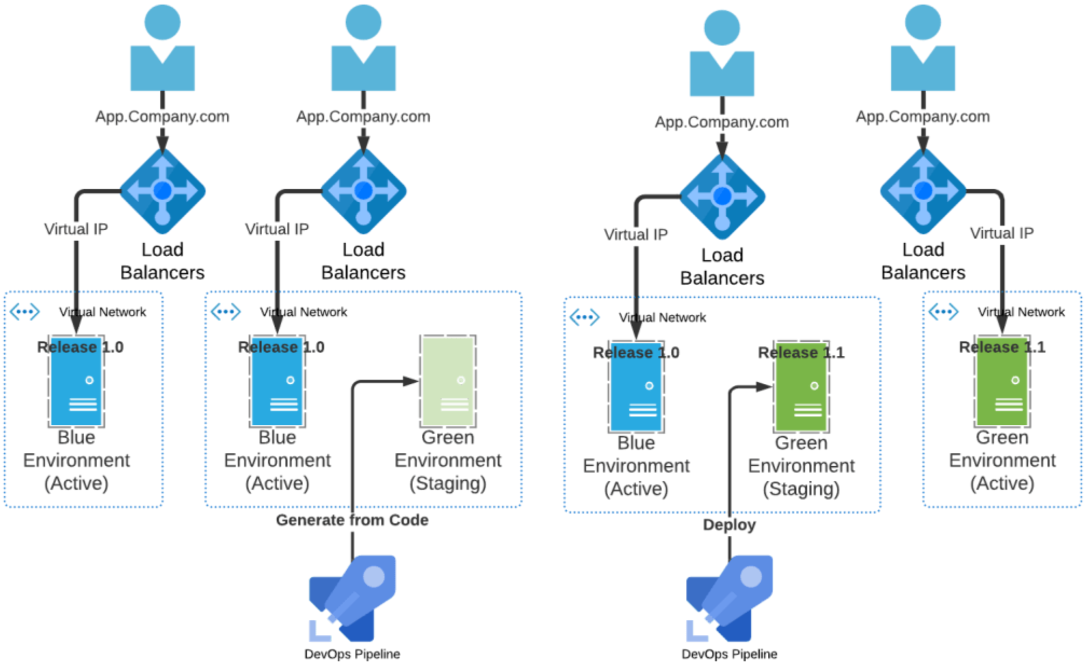


Figure 7: Blue/Green Deployment Pattern [Lieberman 2020]

Logs from blue and green application environments are used by ZT Visibility and Analytics and Automation and Orchestration components for dynamic policy. PEPs restrict subject access to blue and green environment resources. Application design incorporates the blue/green environment transition and load balancers to ensure user and device credentials are not expired during application deployment.

Table 6: Mission Thread DevSecOps Deploy Phase ZT Considerations

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Deploy	Logs from blue and green application environments are used by ZT Visibility and Analytics and Automation and Orchestration components for dynamic policy	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Deploy	PEPs restrict subject access to blue and green environment resources.	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Deploy	Application design incorporates the blue/green environment transition and load balancers to ensure user and device credentials are not expired during application deployment.	BANK1 infrastructure team, pipeline administrator, MTECH developer.

Mission Thread Operate

In the DevSecOps operate phase BANK1 stakeholders use the new release of the SA dashboard application. BANK1 operations monitors the application to verify function and elasticity as active users increase. Application users provide feedback to BANK1 with DevSecOps tools that track and triage the information. This feedback loop is an important source of information for BANK1 and MTECH to understand user needs and shape future application development.

Table 7: Mission Thread DevSecOps Operate Phase ZT Considerations

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Operate	The SA dashboard application enforces ZT tenets and restricts data at the element level.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.
Operate	The SA dashboard application sends log data to ZT APIs for event, alert, and anomaly detection.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.
Operate	The SA dashboard application consumes data from other ZT-enabled services.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.
Operate	The SA dashboard application consumes data from services outside of BANK1 that may not be ZT-enabled.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.
Operate	BANK1 CSOC analysts understand the SA dashboard and incidents that may occur.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.

11 Conclusion

ZT adoption can challenge organizations with new ways of thinking, operating, and collaborating. Enterprises must thoroughly understand their architecture, assets, data, subjects, and resources in order to adopt this new security strategy. Organizations won't be able to focus on technology alone when transitioning to ZT, but must combine it with strategy, vision, culture, and collaboration for success.

DevSecOps changes how organizations historically develop software. Agile methods, combined with people, process, and technology reduce development time and automate the software development lifecycle. While short development lifecycles combined with continuous software test, build, and release help deliver software faster, it also introduces complexity and risk that must be understood and mitigated.

Both initiatives continue to increase in popularity and adoption and likely exist in some form within an organization. Successfully integrating ZT and DevSecOps requires organizations to understand their current state, how they function together, and the impacts that can occur between them. Our mission thread assumes that BANK1 has an established ZT infrastructure and DevSecOps pipeline for MTECH to develop a SA dashboard application. We use this thread to highlight ZT considerations and resources for each phase of a DevSecOps pipeline (Table 8).

A common theme across all considerations and resources listed in Table 8 is that monitoring is a continuous process across all phases of DevSecOps that affects cATO (continuous authority to operate). The automated nature of ZT and DevSecOps require security and performance logging, monitoring, analysis, and response for all components. This provides insight into how the SA dashboard application performs, its security compliance and effectiveness at mitigating threats, and overall residual risk for authorizing officials.

Table 8: Mission Thread DevSecOps/ZT Considerations

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Plan	Data, assets, applications, and services (DAAS) are categorized and managed by mission criticality.	BANK1 infrastructure team, MTECH developer team.
Plan	Technical and policy controls implement data management.	BANK1 infrastructure team, MTECH developer team.
Plan	Data at rest and in-transit is monitored to identify and mitigate theft.	BANK1 infrastructure team, MTECH developer team
Plan	Identity, credential, and access management (ICAM) is used to provision, manage, authenticate, and authorize subject access to resources throughout the environment.	BANK1 infrastructure team, MTECH developer team

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Plan	ICAM, combined with public key infrastructure (PKI), micro- and macro-segmentation, and other technical controls enforce least-privilege policy and prevent lateral movement.	BANK1 infrastructure team, MTECH developer team
Plan	Subject/resource communications are visible, baselined, and monitored by security event and information management (SIEM) to identify events of interest.	BANK1 infrastructure team, MTECH developer team
Plan	Machine learning (ML), artificial intelligence (AI) and security orchestration automation and response (SOAR) platforms work together to integrate information and respond to incidents with standardized application programming interfaces (APIs).	BANK1 infrastructure team, MTECH developer team
Plan	The software factory provides reference software and templates to standardize ZT concepts, configuration, and maintenance. This includes containers and other reusable components that standardize implementing ZT services during application development such as authentication, authorization, and logging.	BANK1 infrastructure team, BANK1 repository administrator, MTECH developer.
Plan	The software factory is also monitored, measured and remediated to mitigate risk and enable continuous authority to operate (cATO).	BANK1 administrator, MTECH developer.
Plan	Roles and policies are defined in the BANK1 environment in order to properly provision subjects to resources needed to complete work, including PKI certificate generation and use.	BANK1 infrastructure team.
Plan	Onboarding provides all documentation necessary for MTECH to begin application development with the platform and ZT infrastructure.	BANK1 infrastructure team.

DEVSECOPS Phase	ZT Consideration	Required Action Resources
	Documentation includes the current state of ZT infrastructure, transition roadmaps, and stakeholders that can answer MTECH infrastructure questions.	
Plan	ZT and application requirements are gathered by MTECH from stakeholders and used to develop a roadmap.	BANK1 stakeholders, MTECH developer.
Develop	Development tools integrate natively with ZT infrastructure services to build a secure entry point for code artifacts.	Pipeline administrator, BANK1 infrastructure team, MTECH developer.
Develop	Policy decision points (PDPs) generate dynamic policy by combining ICAM, enterprise policy, and external data such as threat intelligence, compliance, SIEM, and SOAR.	Pipeline administrator, BANK1 infrastructure team.
Develop	Policy enforcement points (PEPs) use policy to enable, monitor, and terminate connections between subjects and resources.	BANK1 infrastructure team.
Develop	MTECH combines development tools with artifacts from the BANK1 central repository in this phase for SA application development.	BANK1 repository administrator, pipeline administrator, MTECH developer.
Develop	Source code incorporates the BANK1 ZT security stack, APIs, infrastructure strategy, and roadmaps.	BANK1 repository administrator, pipeline administrator, MTECH developer, BANK1 infrastructure team.
Develop	MTECH code adopted from outside engagements is authorized and configuration managed by BANK1 before incorporation into the central repository.	BANK1 repository administrator, MTECH developer.
Develop	Standards and metrics are developed and measured to verify the application meets or exceeds BANK1 requirements as it interoperates with the ZT infrastructure.	BANK1 stakeholders, MTECH developer.

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Develop	BANK1 owns and maintains the ZT infrastructure, central repository, and DEVSECOPS environment and responsively reports and addresses MTECH issues.	BANK1 infrastructure team, MTECH development team.
Develop	BANK1 incorporates MTECH reporting into its ZT infrastructure and DEVSECOPS Epics for internal decision making.	BANK1 infrastructure team, MTECH development team.
Develop	BANK1 reports and tracks MTECH related incidents with the TMS. The TMS is part of the BANK1 ecosystem and automates communication between both organizations.	BANK1 infrastructure team, MTECH development team.
Build	Build tools and automation integrate ZT security services, tenets, and APIs into the SA dashboard application.	BANK1 repository administrator, MTECH developer.
Build	CI/CD orchestrator uses NPEs and PKI for access control.	Pipeline administrator, ICAM administrator, PKI administrator, BANK1 repository administrator, MTECH developer.
Build	Security templates enforce compliance and configuration management.	Pipeline administrator, BANK1 repository administrator, MTECH developer.
Build	Build logs are monitored to identify events, alerts, and anomalies.	SIEM administrator, MTECH developer, pipeline administrator.
Build	Data at rest and in transit is monitored and policy is dynamically updated to mitigate risk.	SIEM administrator, SOAR administrator, ZT policy administrator.
Build	Control gates such as ZT compliance checks are used to mitigate risks before they propagate to the DEVSECOPS test phase.	Pipeline administrator, BANK1 repository administrator, MTECH developer.
Build	BANK1 SOAR playbooks automate incident response activities and report build incidents to MTECH.	MTECH developer, pipeline administrator, SOAR administrator.
Test	The test environment may provide operational devices and services, or others that emulate BANK1's environment.	BANK1 infrastructure team, pipeline administrator, MTECH developer.

DEVSECOPS Phase	ZT Consideration	Required Action Resources
Test	Staging environments (internal and externally hosted) must interoperate with BANK1's ZTA.	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Test	Manual and automated test cases should define ZT subjects, resources, and policies while verifying components and capabilities.	MTECH developer.
Test	Data and infrastructure during this phase are labeled for ZT to determine context for dynamic policy and decision making.	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Test	The iterative nature of this phase also generates volatile data that can impact ZT Visibility and Analytics and Automation and Orchestration components.	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Release and Deliver	New information, interfaces, and services developed for ZT Visibility and Analytics and Automation and Orchestration components are implemented in this phase.	BANK1 infrastructure, BANK1 stakeholders, MTECH developer.
Release and Deliver	The SA dashboard application provides data for ZT infrastructure to generate events, alerts, and anomalies for dynamic policy development and enforcement.	BANK1 infrastructure, MTECH developer.
Release and Deliver	New security templates and controls are deployed with the SA dashboard application.	BANK1 infrastructure, BANK1 repository administrator, MTECH developer.
Release and Deliver	Supporting templates, controls, and other information are released as a separate repository artifact(s)	BANK1 infrastructure, BANK1 repository administrator, MTECH developer.
Release and Deliver	BANK1 CSOC staff work with quality assurance to review and mitigate DEVSECOPS and ZT risk.	BANK1 infrastructure, BANK1 stakeholders, MTECH developer.
Release and Deliver	Automation of the release assumes a deep understanding of the data, environment, subjects, resources, and	BANK1 infrastructure, BANK1 stakeholders, pipeline administrator, MTECH developer.

DEVSECOPS Phase	ZT Consideration	Required Action Resources
	their interactions for release and deliver.	
Release and Deliver	Least privilege is applied to analysis data and results.	BANK1 infrastructure, BANK1 stakeholders, MTECH developer.
Deploy	Logs from blue and green application environments are used by ZT Visibility and Analytics and Automation and Orchestration components for dynamic policy	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Deploy	PEPs restrict subject access to blue and green environment resources.	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Deploy	Application design incorporates the blue/green environment transition and load balancers to ensure user and device credentials are not expired during application deployment.	BANK1 infrastructure team, pipeline administrator, MTECH developer.
Operate	The SA dashboard application enforces ZT tenets and restricts data at the element level.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.
Operate	The SA dashboard application sends log data to ZT APIs for event, alert, and anomaly detection.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.
Operate	The SA dashboard application consumes data from other ZT-enabled services.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.
Operate	The SA dashboard application consumes data from services outside of BANK1 that may not be ZT-enabled.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.
Operate	BANK1 CSOC analysts understand the SA dashboard and incidents that may occur.	BANK1 infrastructure team, BANK1 CSOC team, MTECH developer.

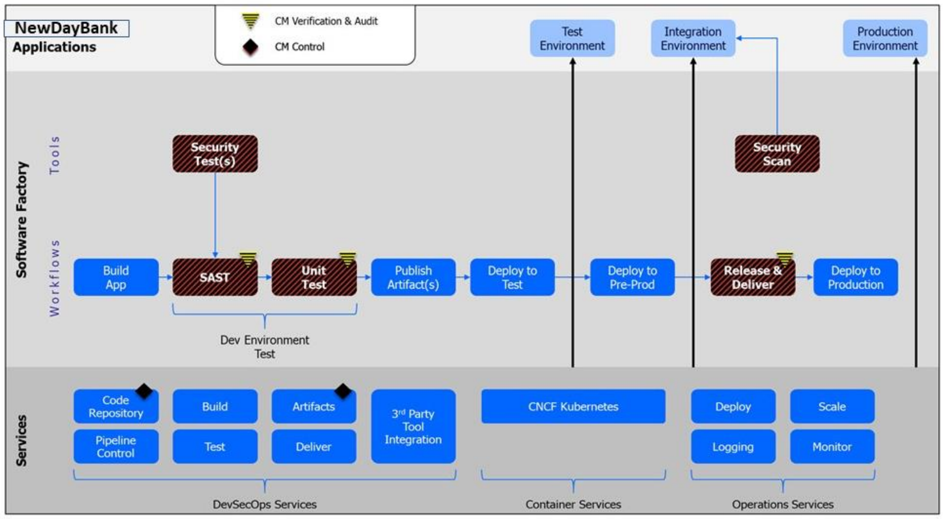
Appendix A Application Development Mission Thread

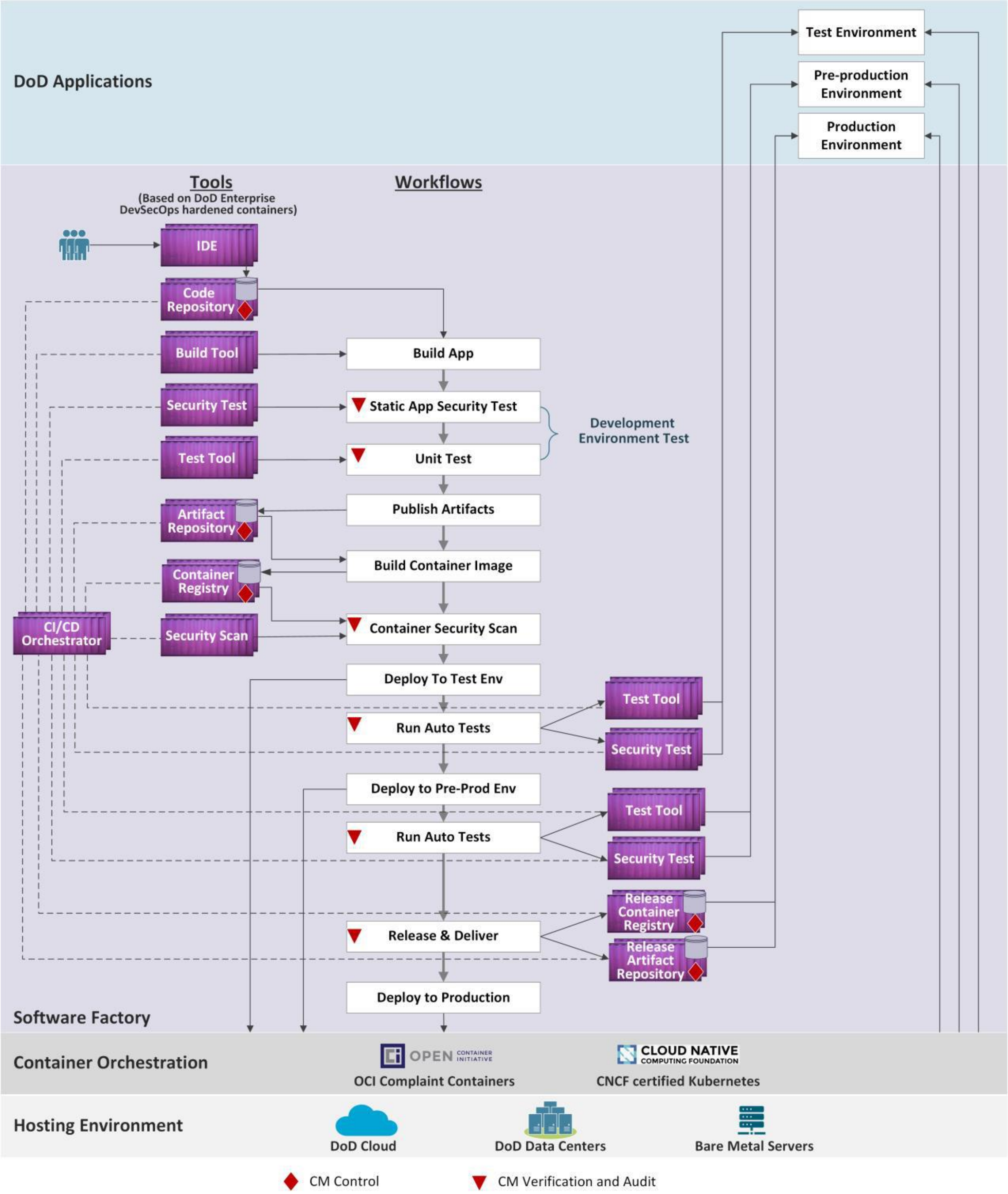
Introduction

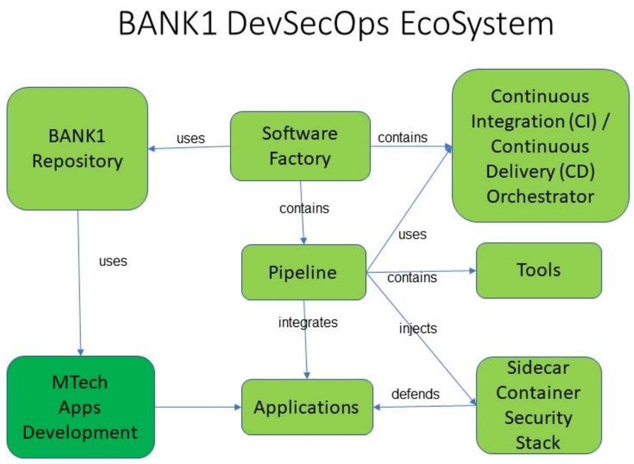
BANK1 has developed a DevSecOps ecosystem to improve the cybersecurity of applications developed for use on its cybersecurity operations center. The ecosystem will be used by BANK1 staff, as well as third-party application developers. The ecosystem contains an artifact repository which holds hardened VM images and hardened OCI compliant container images of: DevSecOps tools, container security tools, common program platform components, custom code, build process, configuration files, and applications. BANK1 is providing the cloud infrastructure that developers will work in. In addition, BANK1 is implementing Zero Trust principles into it DevSecOps ecosystem with the goal to improve cybersecurity.

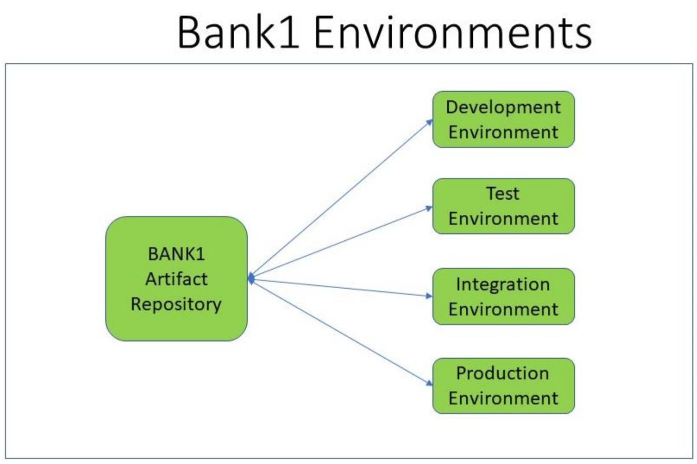
A small business (MTECH) which specialized in situational awareness (SA) dashboards has been contracted to create applications for a company's BANK1 cybersecurity operations center. MTECH is required to use artifacts from the ecosystem's repository to create and update SA dashboards in a software factory process. This thread will be used to explore the impacts of implementing zero trust tenets to the DevSecOps ecosystem.

Supporting Diagrams




DoD Applications
Test Environment
Pre-production Environment
Production Environment
Tools
(Based on DoD Enterprise DevSecOps hardened containers)
Workflows
IDE
Code Repository
Build Tool
Security Test
Test Tool
Artifact Repository
Container Registry
Security Scan
CI/CD Orchestrator
Build App
Static App Security Test
Unit Test
Development Environment Test
Publish Artifacts
Build Container Image
Container Security Scan
Deploy To Test Env
Run Auto Tests
Test Tool
Security Test
Deploy to Pre-Prod Env
Run Auto Tests
Test Tool
Security Test
Release Container Registry
Release Artifact Repository
Release & Deliver
Deploy to Production
Software Factory
Container Orchestration
OPEN CONTAINER INITIATIVE
OCI Complaint Containers
CLOUD NATIVE COMPUTING FOUNDATION
CNCF certified Kubernetes
Hosting Environment
DoD Cloud
DoD Data Centers
Bare Metal Servers
◆ CM Control
▼ CM Verification and Audit





NewDayBank Artifact Repository – holds hardened VM images and hardened OCI compliant container images of: DevSecOps container security tools, common program platform components, custom code, build process, configuration files, and applications.

Thread

Name	Application Development
Vignette (Summary Description)	A small business MTECH which specialized in situational awareness (SA) dashboards has been contracted to create applications for a company's BANK1 cybersecurity operations center. The BANK1 has in place a DevSecOps ecosystem which MTECH is required to use artifacts from the ecosystem's repository to create and update SA dashboards in a software factory process. BANK1 is implementing Zero Trust principles into it DevSecOps ecosystem with the goal to improve cybersecurity.

Nodes/Actors	1. BANK1 staff a. contract specialist b. development environment staff member c. IAM staff member d. COC technical point of contact (PoC) e. DevSecOps staff member f. COC staff members g. testing staff member h. quality assurance staff member i. system engineer/architect 2. BANK1 artifact repository assets a. IDE b. Code repository c. Build tool d. Security test tool e. Test tool f. Artifact repository g. Container registry h. CI/CD orchestrator i. Security scan tool j. Release container registry k. Release artifact repository 3. MTECH staff a. Project manager b. IT IAM c. Application development team i. 1 lead developer ii. 4 developers d. Technical PoC
Assumptions	Assump-1. BANK1 provides the cloud infrastructure to ensure the ability to monitor any activity within its DevSecOps ecosystem. Assump-2. Training is available for developers to be able to develop an understanding to securely use the ecosystem. Assump-3. There will be a separate mission thread which deals with situational awareness (SA) dashboard app maintenance. Assump-4. MTECH only uses BANK1's cloud infrastructure and the artifact repository for its applications' development and testing. (i.e., support for subcontractors) Assump-5. MTECH developers will be able to monitor the applications and develop metrics to assess performance and security via BANK1's artifact repository.
Quality Attributes	1. Securability 2. Resiliency 3. Agility
References	1.

Step	**Time**	**Description**

1		BANK1 places a contract with MTECH to develop and update SA applications for their Cybersecurity Operations Center (COC). BANK1 provides a link to information concerning its SecDevOps ecosystem which describes the concept of operations, processes and procedures, and how to request accounts to be able to use the ecosystem.
notes	1. BANK1 contract specialist 2. MTECH project manager	
2		MTECH provides the information for its team to be able to access the ecosystem.
notes	1. MTECH IT IAM staff member (handles credentials aspects) 2. MTECH app development team (lead developer, 4 developers) 3.	
3		MTECH receives the account information and completes training on the ecosystem.
notes	1.	
4		MTECH creates its development environment and initiates building its SA dashboard application.
notes	1. One of the developers is responsible for setting up development environment (consists of team collaboration tool issue tracking tool, configuration management tool, project management tool, integrated development environment (IDE), source code repository, database tool, test tools) using tools from BANK1 development repository. 2. Please see Assump-4	
5		MTECH iterates on the Dev Environment Test process (application coding -> static application security testing (SAST) -> unit testing) until it's ready to publish its artifact(s).
notes	1. BANK1 development environment staff member (can address SAST tool questions) 2. BANK1 COC technical PoC (can address operations questions)	
6		MTECH publish their SA dashboard artifact(s) to BANK1's repository.
notes	1. MTECH lead engineer handles this	
7		Under BANK1's control, a test environment is created and artifacts (including MTECH's SA dashboard) are pulled from the repository and tested. Steps 5-7 are repeated on until artifacts are deemed ready to deploy to pre-production.

notes	1. BANK1 DevSecOps staff member configures the Test Environment pipeline (need to identify all components (build tool, code analysis tool, SAST tool, test development tools, Interactive Application Security Test (IAST) tool, database tool) 2. BANK1 testing staff member 3. BANK1 quality assurance staff member 4. BANK1 COC staff members (analyze design, perform threat analysis, identify monitoring requirements, analyze app's operation) 5. BANK1 system engineer/architect oversees work and addresses tradeoff decisions 6. MTECH technical PoC (support testing questions) 7. Defect information from testing is added to the associated SA dashboard artifacts in the repository. 8. This step is a "control gate"	
7a		BANK1 publishes the SA dashboard artifact(s) to integration repository.
	1. BANK1 DevSecOps engineer handles this	
8		Under BANK1's control, an integration environment which represents the cybersecurity operations center is created, artifacts are pulled, and SA dashboard is built.
notes	1. BANK1 DevSecOps staff member configures the Integration Environment pipeline (components similar to Test environment pipeline) 2. BANK1 COC staff members (analyze design, perform threat analysis, identify monitoring requirements, analyze app's operation) 3. MTECH technical PoC (support testing questions)	
9		Security scanning is performed in the integration environment under BANK1's control. If defects are found, return to step 5.
notes	1. BANK1 testing staff member (identify different types of testing performed) 2. BANK1 quality assurance staff member 3. MTECH technical PoC (support testing questions) 4. Defects information from the scanning are added to the associated SA dashboard artifacts in the repository.	
10	Day: Time:	Integration testing with the SA dashboard application is performed under BANK1 control until it's deemed ready to be released for production use.
notes	1. BANK1 COC staff members (analyze design, perform threat analysis, identify monitoring requirements, analyze app's operation) 2. BANK1 testing staff member 3. BANK1 quality assurance staff member 4. BANK1 system engineer/architect oversees work and addresses tradeoff decisions 5. MTECH technical PoC (support testing questions) 6. This step is a "control gate"	
11	Day: Time:	BANK1 cybersecurity operations team analyzes the SA dashboard to determine what monitoring and logging capabilities are needed to support operations. (move analysis description to planning; this step is a validation of the monitoring capabilities)
notes	1. BANK1 COC staff members (analyze design, perform threat analysis, identify monitoring requirements, analyze app's operation) 2. BANK1 system engineer/architect (call out in earlier steps to perform reviews; list role prior to step one in Assumptions sections)	

12	Day: Time:	Under BANK1 control, the SA dashboard app is deployed to the production environment and additional monitoring capabilities are put in place that support the SA dashboard, if needed.
notes	1. BANK1 IT staff member creates new production environment 2. BANK1 COC staff members 3. BANK1 testing staff member 4. BANK1 quality assurance staff member 5. BANK1 system engineer/architect oversees work and addresses tradeoff decisions 6. BANK1 uses a blue/green deployment approach 7. This step is a “control gate”	
13	Day: Time:	
notes	1.	

Bibliography

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

[Barnett 2019]

Barnett, J. *Zero Trust Guide Coming to DOD in 2021*. 2019.
https://www.fedscoop.com/zero-trust-reference-guide-disa-nsa-cybersecurity.

[Cherdantseva 2013]

Cherdantseva, Y. and Hilton, J., A Reference Model of Information Assurance & Security. 2013. *2013 International Conference on Availability, Reliability and Security*. Pages 546-555.
https://doi.org/10.1109/ARES.2013.72.

[Cloutier 2010]

Cloutier, R., Muller, G., Verma, D., Nilchiani, R., Hole, E. and Bone, M. The Concept of Reference Architectures. *Syst. Engin*. Volume 13. Pages 14-27. https://doi.org/10.1002/sys.20129. 2010.

[Cunningham 2018]

Cunningham, C. *The Zero Trust eXtended (ZTX) Ecosystem: Extending Zero Trust Security Across Your Digital Business*. https://www.cisco.com/c/dam/m/en_sg/solutions/security/pdfs/forrester-ztx.pdf. 2018.

[DoD 2019]

United States Department of Defense. *Department of Defense (DoD) Digital Modernization Strategy*. https://media.defense.gov/2019/Jul/12/2002156622/-1/-1/1/DOD-DIGITAL-MODERNIZATION-STRATEGY-2019.PDF. 2019.

[DoD 2021a]

United States Department of Defense (2021). *Department of Defense (DoD) Zero Trust Reference Architecture*. https://dodcio.defense.gov/Portals/0/Documents/Library/(U)ZT_RA_v1.1(U)_Mar21.pdf.

[DoD 2021b]

United States Department of Defense Chief Information Officer. *DoD Enterprise DevSecOps Reference Design Version 1.0*. 2019. https://dodcio.defense.gov/Portals/0/Documents/DoD%20Enterprise%20DevSecOps%20Reference%20Design%20v1.0_Public%20Release.pdf?ver=2019-09-26-115824-583.

[DoD 2021c]

United States Department of Defense. *DoD Enterprise DevSecOps Reference Design: CNCF Kubernetes*. 2021. https://software.af.mil/wp-content/uploads/2021/05/DoD-Enterprise-DevSecOps-Reference-Design-v2.0-CNCF-Kubernetes.pdf.

[DoD 2021d]
United States Department of Defense. *DoD Enterprise DevSecOps Fundamentals Version 2.0*. 2021. https://software.af.mil/wp-content/uploads/2021/05/DoD-Enterprise-DevSecOps-2.0-Fundamentals.pdf.

[DoD 2021e]
United States Department of Defense. *DoD DevSecOps Fundamentals Guidebook: DevSecOps Tools & Activities 2.0*. 2021. https://software.af.mil/wp-content/uploads/2021/05/DoD-Enterprise-DevSecOps-2.0-Tools-and-Activities-Guide.pdf.

[Gagliardi 2013]
Gagliardi, Michael; Wood, William; & Morrow, Timothy. *Introduction to the Mission Thread Workshop*. CMU/SEI-2013-TR-003. Software Engineering Institute, Carnegie Mellon University. 2013. http://resources.sei.cmu.edu/library/asset-view.cfm?AssetID=63148

[Garbis 2021]
Garbis, J. and Chapman, J. (2021). *Zero Trust Security: An Enterprise Guide*. Berkley, CA: Apress.

[IEEE 2020]
IEEE. *P2887-Recommended Practice for Zero Trust Security*. 2020. Retrieved from https://standards.ieee.org/project/2887.html.

[Kindervag 2016]
Kindervag, J. *No More Chewy Centers: The Zero Trust Model Of Information Security*. 2016. https://www.forrester.com/report/No+More+Chewy+Centers+The+Zero+Trust+Model+Of+Information+Security/-/E-RES56682.

[Lieberman 2020]
Lieberman, B. *DevSecOps–Blue/Green Deployment Pattern*. 2020. https://blogs.perficient.com/2020/05/14/devsecops-blue-green-deployment-pattern

[NCCOE 2018]
NIST National Center of Cybersecurity Excellence. *Zero Trust Architecture Community of Interest.* 2018. https://www.nccoe.nist.gov/projects/building-blocks/zero-trust-architecture.

[NIST 2020]
Rose, S., Borchert, O., Mitchell, S., and Connelly, S. NIST Special Publication 800-207*: Zero Trust Architecture*. 2020. https://csrc.nist.gov/publications/detail/sp/800-207/final.

[NIST 2021]
NIST. *NIST Zero Trust Topic*. 2021. https://csrc.nist.gov/Topics/Security-and-Privacy/zero-trust.

Contact Us

Software Engineering Institute
4500 Fifth Avenue, Pittsburgh, PA 15213-2612

Phone: 412/268.5800 | 888.201.4479
Web: www.sei.cmu.edu
Email: info@sei.cmu.edu

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

DM21-0679