SANDIA REPORT
SAND2023-08255
Printed August 2023

Sandia National Laboratories

CI/CD Pipeline and DevSecOps Integration for Security and Load Testing

Dominic, S, Donofrio. Melissa, L, Fusco. Hongrong, Zhong.

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico
87185 and Livermore,
California 94550

Issued by Sandia National Laboratories, operated for the United States Department of Energy by National Technology & Engineering Solutions of Sandia, LLC.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@osti.gov
Online ordering: http://www.osti.gov/scitech

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5301 Shawnee Rd
Alexandria, VA 22312

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.gov
Online order: https://classic.ntis.gov/help/order-methods/

Table of Contents

ACRONYMS AND TERMS

Acronym/Term	Definition
AKS	Azure Kubernetes Service
API	Application Programming Interface
ASA	The Attack Surface Analyzer
ASVS	Application Security Verification Standard
AWS	Amazon Web Service
CI/CD pipeline	Continuous Integration and Continuous Deployment Pipeline
CIS	Center for Internet Security
CLI	Command Line Interface
CPU	Central Processing Unit
CSF	Cybersecurity Framework
DAST	Dynamic Application Security Testing
DevOps	Development and Operations
DevSecOps	Development, Security, and Operations
EC2	Elastic Compute Cloud
EKS	Elastic Kubernetes Service
EMTS	Enterprise & Mission Testing Services
GKE	Google Kubernetes Engine
GUI	Graphical User Interface
HTTP	Hyper Text Transfer Protocol
IaC	Infrastructure as Code
IAM	Identity and Access Management
MiTM	Man-in-The-Middle
MSA	Microsoft Security Assessment
MSAT	Microsoft Security Assessment Tool
MSCA	Microsoft Security Code Analysis
NIST	National Institute of Standards and Technology
NMCCoE	New Mexico Cybersecurity Center of Excellence
OCI	OpenShift Container Platform
OS	Operating System
OWASP	Open Web Application Security Project
RBAC	Role-Based Access Control
SAST	Static Application Security Testing
SDL	Secure Development Lifecycle
SDLC	Software Development Lifecycle

Acronym/Term	Definition
SNL	Sandia National Laboratories
SOAP	Simple Object Access Protocol
SSDF	Secure Software Development Framework
Tracer FIRE	Forensic and Incident Response Exercise
UI	User Interface
VM	Virtual Machine
VPC	Virtual Private Cloud
YAML	Yet Another Markup Language
ZAP	Zed Attack Proxy

1. PROJECT MANAGEMENT INFORMATION

1.1. Introduction

Dominic D'Onofrio is currently a Junior studying Information and Technology at New Mexico Institute of Mining and Technology. He recently secured an Internship with NMCCoE, where he is involved with the TracerFIRE 12 project. Additionally, he is contributing to the load and security testing team by researching ways to implement pipelining and DevSecOps; this is his main project while he is at part time capacity for TracerFIRE 12. He is doing these projects to enhance his knowledge as a system administrator and gain a deeper understating of cybersecurity practices within national labs.

Hongrong Zhong is currently in his third year of studying Computer Science working towards a master's degree in Cybersecurity, with a minor in Psychology at New Mexico Institute of Mining and Technology. He's in the Network/Forensic team of TracerFIRE 12 and a part time contributor to Dominic's DevSecOps integration project.

Melissa Fusco is currently in her third year at New Mexico Institute of Mining and Technology studying Computer Science. She is working towards her master's in cybersecurity with a minor in Computer Engineering. Melissa has been working with the NMCCoE since her second semester at Tech as a tutor and a cyber outreach assistant. This summer, through a grant from Sandia National Laboratories, her work through the NMCCoE has consisted of a primary focus on contributing to TracerFIRE 12 and secondary focus on CI/CD pipeline integration research. Melissa's main goal of this summer is to learn more about red teaming and cloud networks while also networking within Sandia.

1.2. Problem Statement

How can we integrate security and load testing into a CI/CD pipeline?

By incorporating a CI/CD pipeline into load and security testing, teams can automate multiple processes that would otherwise require manual scripting or tasks. This is crucial as it will significantly reduce the time spent on each project and enable team members to concentrate on creating innovative methods for process automation and pipeline development. Ultimately, this approach will enhance efficiency and will provide opportunities for exploring new avenues of automation and pipeline implementation.

1.3. Governance Board

Lorie Liebrock	IT and TracerFIRE Internship Director
Andrew Brungard	Lead, Integrated Solutions Architect
Tanya DeLara	Manager, Information Systems Engineering
Zane Parker	Technical POC
Dominic D'Onofrio	Head Researcher
Melissa Fusco	Part Time Researcher
Hongrong Zhong	Part Time Researcher

1.4. Project Milestones

Milestone	Date	Completed
Research how to deploy NeoLoad web (and other recommendations)	6/19/23 – 6/23/23	Completed
Talk to other sandias about their pipelining systems and DevOps	7/3/23 – 7/28/23	Completed
Complete all notes and start writing research paper and recommendations	7/17/23 – 7/21/23	Completed
Finish research and recommendations	7/24/23 – 7/28/23	Completed
Create poster and finish revisions of Final draft	8/1/23 – 8/4/23	Completed
Present findings to group	8/7/23 – 8/11/23	Completed

1.5. Research Methods and Procedures

The research methods employed involve examining resources from Gartner, as well as utilizing online platforms and knowledge bases, to explore topics aimed at enhancing load and security testing, as well as obtaining valuable insights on pipeline information. This also included talking to other sections of Sandia who already implement pipeline and DevOps at Sandia National Labs.

1.6. Team Members and Responsibilities

Dominic D'Onofrio – Team lead and researcher for CI/CD pipeline deployment options and DevSecOps recommendations.

Melissa Fusco – Research security testing standards and provide recommendations. Help proofread and maintain the final document.

Hongrong Zhong (Ronny) – Research DevOps standards and provide recommendations. Help proofread and maintain the final document.

1.7. Project Team Meeting Schedule

Meeting	Time	People
Manager meeting	Biweekly, (Thursdays at 11am)	Tanya, Zane, and Dominic
Team meeting	Weekly, (Monday at 10 am)	Dominic, Ronny, Melissa
All hands meeting	Weekly, (Friday at 2pm)	All interns and Lorie Liebrock
Weekly report meeting	Weekly (Thursday at 1pm)	Zane and Dominic

1.8. Project Management Methodology

In this project, we utilized a presentation template that was presented every week and served as a viable tool for summarizing our weekly progress, outlining our upcoming task for the following week, addressing any questions or concerns, requesting assistance when needed, and identifying areas for personal improvement. This template allowed us to effectively communicate and track our progress while promoting collaboration and continuing growth. During the weekly team meetings we provided updates on how we plan to implement the recommendations and how they can be integrated within a pipeline.

1.9. Project Assumptions

In this project we assume that

- SNL will provide research tools and access to subject matters experts required to complete this project.
- EMTS cannot test a system without permission from the customer and notifying relevant stakeholders.
- We can use the given contacts to collect information about other SNL sections.
- Gartner will be provided for this research project.

1.10. Team Objectives

- Research how to implement DevSecOps and pipelines into the EMTS team.
- Create recommendations based on research and note how to implement them.
- Create a final report with final recommendations.

1.11. Team Deliverables

- A final research paper that includes recommendations about integrating DevSecOps and pipelining into security and load testing.

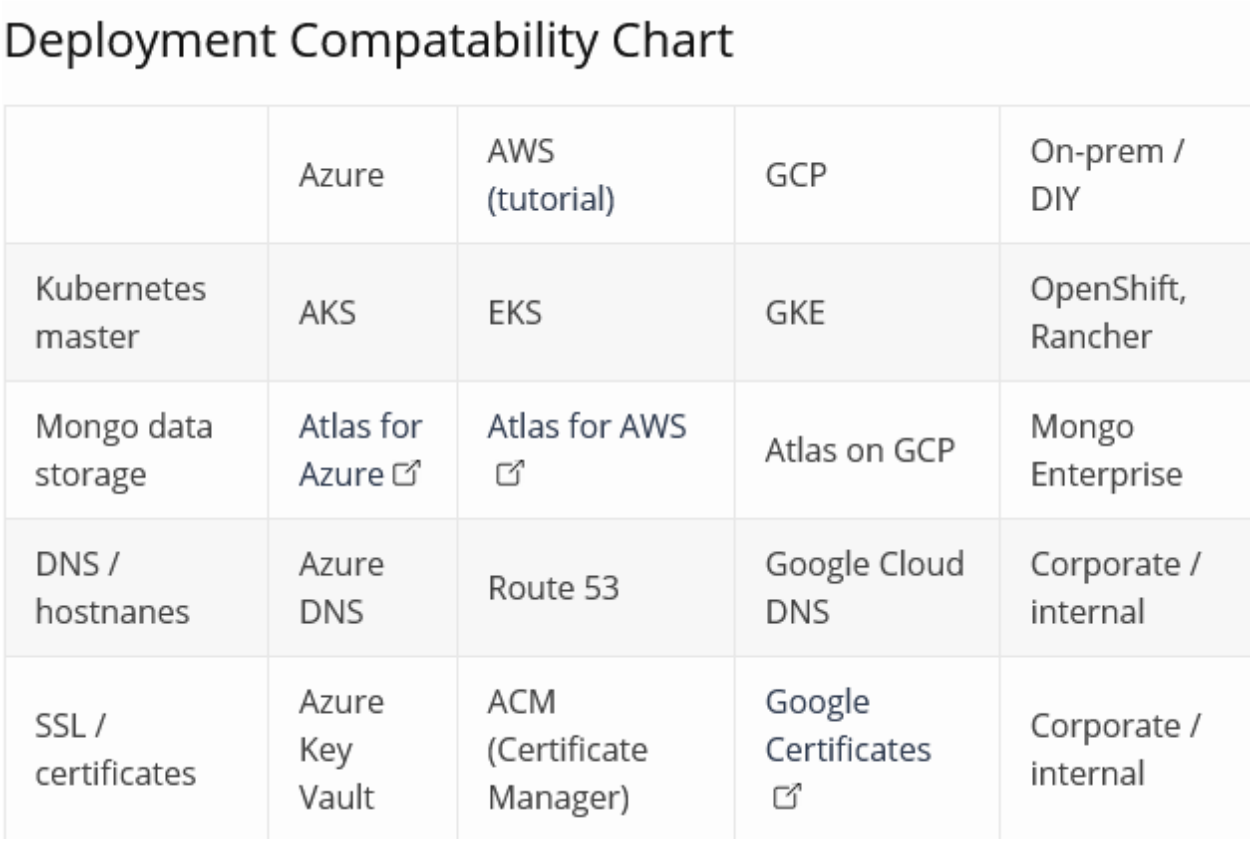
2. RESEARCH

2.1. Neoload Web

Neoload Web is a centralized performance testing platform. With this software, users are able to launch performance tests, load generation infrastructure, analyze results, export Neoload data, and launch Neoload Web from independent users. Neoload Web can be deployed in different ways through the use of Kubernetes and cloud infrastructure. Below are the deployment options that can be used and the ones that are allowed by SNL.

2.2. Deployment Options

Deployment option	On premise option	Allowed by SNL
Azure (AKS)	Yes	Yes
AWS (EKS)	Yes	Pending
GCP (GKE)	No	No
Helm Chart (OpenShift)	Yes	Yes



Deployment Compatability Chart

	Azure	AWS (tutorial)	GCP	On-prem / DIY
Kubernetes master	AKS	EKS	GKE	OpenShift, Rancher
Mongo data storage	Atlas for Azure	Atlas for AWS	Atlas on GCP	Mongo Enterprise
DNS / hostnanes	Azure DNS	Route 53	Google Cloud DNS	Corporate / internal
SSL / certificates	Azure Key Vault	ACM (Certificate Manager)	Google Certificates	Corporate / internal

IMAGE BY NEOTYS: WHICH SHOWS THE INFRASTRUCTURE THAT NEOLOAD WEB CAN RUN ON WITH THE SELECTED KUBERNETES MASTER. [32]

2.2.1. Azure Kubernetes Service

Azure AKS is a managed orchestration service provided by Microsoft Azure. Azure allows users to focus on deployment, management, and scaled containerization without worrying about infrastructure. With AKS users can focus on developing and running applications while azure manages Kubernetes clusters. Azure does this is by automating various tasks such as provisioning, upgrading, and monitoring Kubernetes clusters. Currently there is no installation guide for Neoload Web by Neotys but one is being developed.

Features include:

- Provides an easy way to create and configure a Kubernetes cluster with just a few clicks or through command-line tools.
- Enables users to scale applications seamlessly to meet demand. It can automatically adjust the cluster size based on resource utilization and workload requirements. This ensures applications can handle increased traffic or workload without downtime.
- Helps ensure that applications are highly available and resilient. By distributing clusters across multiple nodes in the cluster it can automatically handle node failures, reschedules containers, and maintains the desired state of applications.
- AKS integrates well with other Azure services, such as Azure Container Registry for container image storage, Azure Monitor for monitoring and diagnostics, Azure Active Directory for authentication and access control, and more.
- AKS integrates with popular DevOps tools like Azure DevOps and Azure Pipelines, which automates the deployment, testing, and continuous delivery of containerized applications.

Pros of AKS:

1. Ease of Use: AKS simplifies the deployment and management of Kubernetes clusters. It abstracts away much of the underlying infrastructure complexity, making it easier for developers to focus on application development rather than cluster administration.

2. Scalability: AKS enables seamless scaling of applications. It provides built-in support for scaling containers horizontally or vertically, allowing applications to handle increased workloads efficiently.

3. High Availability: AKS ensures high availability of applications by distributing them across multiple nodes in a cluster. It automatically monitors and replaces failed nodes, ensuring that applications remain accessible and responsive.

4. Integration with Azure Services: AKS integrates seamlessly with other Azure services, such as Azure Container Registry, Azure Monitor, Azure Active Directory, and Azure DevOps.

This integration provides a comprehensive ecosystem for building, deploying, and monitoring containerized applications.

5. Security: AKS offers robust security features. It supports role-based access control (RBAC) to manage user permissions, and it provides network isolation through Azure Virtual Networks. It also supports Azure Security Center for threat detection, monitoring and forensics.

6. Continuous Integration and Deployment: AKS can be integrated with CI/CD (Continuous Integration/Continuous Deployment) pipelines, allowing developers to automate application builds, testing, and deployment processes. This enables faster development cycles and reduces the time to market.

Cons of AKS:

1. Learning Curve: While AKS abstracts much of the underlying Kubernetes infrastructure, there is still a learning curve associated with understanding Kubernetes concepts and managing cluster configurations. Developers and operators need to invest time in learning Kubernetes to leverage AKS effectively.

2. Cost: While AKS itself is a free service, you need to consider the cost of Azure resources such as virtual machines, storage, and networking that are required to run the AKS cluster. The cost can vary based on cluster size, number of nodes, and resource utilization.

3. Dependency on Azure: AKS is tightly integrated with Azure services. While this provides a seamless experience for Azure users, it also means that AKS might not be the best choice if you have a multi-cloud or hybrid cloud strategy that involves other cloud providers.

4. Limited Control: As a managed service, AKS abstracts away much of the infrastructure management. While this can be a benefit, it also means you have limited control over the underlying infrastructure compared to running your own self-managed Kubernetes cluster.

5. Version Lag: There might be a slight lag between the release of new Kubernetes versions and their availability on AKS. If you require the latest Kubernetes features immediately, self-managed clusters or other Kubernetes platforms may be more suitable.

2.2.2. Amazon Web Services (AWS) EKS

Amazon Web Services EKS is a managed service provided by Amazon. AWS simplifies how Kubernetes manages, deploys, and scales applications. This allows users to focus on development of applications without worrying about infrastructure.

Features include:

- Scalability and Elasticity: AWS provides on-demand and scalable resources. This allows businesses to scale up or down based on their requirements, using services such as Elastic Compute Cloud (EC2) and Auto Scaling.
- Variety of Services: AWS offers a broad set of products and services such as computing power, storage, databases, analytics, networking, mobile, developer tools, management tools,

IoT, security, and enterprise applications. This makes it a one-stop solution for many businesses' digital infrastructure needs.

- Security: AWS prioritizes security and compliance, offering end-to-end security for all services. Features include encryption methods, security groups and networks ACLs, dedicated connections, and AWS Identity and Access Management (IAM).
- Global Presence: AWS has a worldwide network of regions and availability zones (AZs), which allows you to host your applications in multiple locations worldwide for redundancy and lower latency.
- Cost-Efficiency: AWS follows a pay-as-you-go model, meaning customers only pay for the services they use. It also provides several tools to help users manage their costs, such as Cost Explorer and Budgets.

Pros of EKS:

1. Easy Deployment: EKS simplifies the deployment of Kubernetes clusters by handling the underlying infrastructure management. It provides an easy-to-use interface and CLI tools that abstract away the complexities of cluster setup and configuration.

2. Scalability and Elasticity: EKS allows seamless scaling of applications. It provides built-in integration with AWS Auto Scaling, enabling automatic scaling of the underlying infrastructure based on application demand. This ensures that applications can handle increased workloads efficiently.

3. High Availability: EKS ensures high availability of applications by distributing them across multiple availability zones within a region. It automatically detects and replaces failed nodes, ensuring that applications remain accessible and resilient.

4. Integration with AWS Services: EKS integrates tightly with the AWS ecosystem. It provides seamless integration with various AWS services like Elastic Load Balancer (ELB), Amazon RDS, Amazon S3, AWS IAM, and more. This enables users to leverage a wide range of complementary services for networking, storage, security, monitoring, and database management.

5. Security: EKS offers robust security features. It integrates with AWS Identity and Access Management (IAM) for fine-grained access control, and it supports Amazon Virtual Private Cloud (VPC) networking, enabling network isolation and security group rules. EKS also supports AWS CloudTrail for auditing and AWS Identity and Access Management Roles for secure authentication and authorization.

6. Managed Upgrades: EKS manages the control plane and automates Kubernetes upgrades, ensuring that users have access to the latest stable versions without the burden of manual upgrades. This helps keep the cluster secure and up to date.

Cons of EKS:

1. Learning Curve: While EKS simplifies Kubernetes cluster management, it still requires users to understand Kubernetes concepts and best practices. Users need to have knowledge of Kubernetes and associated tools to effectively deploy and manage applications on EKS.

2. Cost: EKS has associated costs, including charges for EC2 instances, load balancers, storage, and data transfer. The cost can vary based on cluster size, number of nodes, and resource utilization. Users need to consider the cost implications of running EKS clusters and associated AWS resources.

3. AWS Lock-In: EKS is tightly integrated with the AWS ecosystem. While this provides a seamless experience for AWS users, it may result in vendor lock-in. Moving an EKS workload to another cloud provider or managing it outside of AWS may require significant effort and adjustments.

4. Support Limitations: While EKS offers support options, users may find limitations in terms of response time and available support channels. Depending on the support plan chosen, users may experience delays in issue resolution or have limited access to technical assistance.

5. Limited Regional Availability: EKS may not be available in all AWS regions. This can restrict deployment options for users in certain geographical areas, potentially requiring them to choose an alternative Kubernetes solution.

6. Version Lag: There might be a slight lag between the release of new Kubernetes versions and their availability on EKS. If immediate access to the latest Kubernetes features is critical, self-managed Kubernetes clusters or other Kubernetes platforms may provide more flexibility.

2.2.3. Google Cloud Platform (GCP) GKE

Google Kubernetes service provides operational power of Kubernetes while managing the underlying components such as control panel and nodes. GKE environment consists of nodes, which are Compute Engine virtual machines (VMs), that are grouped together to form a cluster. You package your apps (also called workloads) into containers. You deploy sets of containers as Pods to your nodes.

Features in GKE:

- Can fully manage nodes on auto pilot modes which also has built in hardening.
- Has flexible maintenance windows and exclusions allowing for configuring and upgrading.
- Node upgrades are available to optimize availability and manage disruptions.
- Can automatically scale nodes based on the number of Pods with auto mode or standard mode.
- Has an option for node auto repair to maintain health and availability.
- Can integrate CI/CD pipelines with cloud deploy and cloud build.
- Hardened node operating systems for security.

Pros of GKE:

1. Seamless Integration with Google Cloud: GKE is tightly integrated with the Google Cloud ecosystem. It provides seamless integration with other Google Cloud services, such as

Google Cloud Storage, Google Cloud Load Balancing, Stackdriver Monitoring, and Cloud IAM. This integration simplifies the development and management of applications on Google Cloud.

2. Ease of Use: GKE offers a user-friendly interface and command-line tools that simplify cluster creation, deployment, and management. It provides an intuitive user experience, allowing users to focus on application development rather than infrastructure management.

3. Scalability and Elasticity: GKE enables easy scaling of applications. It leverages Google Cloud's infrastructure capabilities, such as Auto Scaling, to automatically adjust the cluster size based on application demand. This ensures that applications can handle increased workloads efficiently.

4. High Availability and Reliability: GKE ensures high availability and reliability by distributing applications across multiple Google Cloud zones within a region. It automatically monitors and replaces failed nodes, ensuring that applications remain accessible and resilient.

5. Security: GKE offers robust security features. It integrates with Google Cloud Identity and Access Management (IAM), providing fine-grained access control. GKE also supports features like node auto-upgrades, automatic SSL/TLS certificate management, and VPC-native networking, enhancing the security posture of applications running on the platform.

6. Managed Upgrades: GKE manages the control plane and automates Kubernetes version upgrades. It ensures that users have access to the latest stable versions without the need for manual upgrades. This helps keep the cluster secure and up to date.

Cons of GKE:

1. Learning Curve: While GKE simplifies Kubernetes cluster management, it still requires users to understand Kubernetes concepts and best practices. Users need to have knowledge of Kubernetes and associated tools to effectively deploy and manage applications on GKE.

2. Cost: GKE has associated costs, including charges for compute resources, networking, storage, and data transfer. The cost can vary based on cluster size, node types, and resource utilization. Users need to consider the cost implications of running GKE clusters and associated Google Cloud resources.

3. Vendor Lock-In: GKE is a Google Cloud-specific service, which can result in vendor lock-in. Moving a GKE workload to another cloud provider or managing it outside of Google Cloud may require significant effort and adjustments.

4. Regional Availability: GKE may not be available in all Google Cloud regions. This can restrict deployment options for users in certain geographical areas, potentially requiring them to choose an alternative Kubernetes solution.

5. Support Limitations: While GKE offers support options, users may find limitations in terms of response time and available support channels. Depending on the support plan chosen, users may experience delays in issue resolution or have limited access to technical assistance.

6. Version Lag: There might be a slight lag between the release of new Kubernetes versions and their availability on GKE. If immediate access to the latest Kubernetes features is critical, self-managed Kubernetes clusters or other Kubernetes platforms may provide more flexibility.

2.2.4. Red Hat OpenShift

OpenShift is a cloud-based Kubernetes platform. OpenShift Container Platform (OCI) is a private platform as a service for organizations that deploy and manage OpenShift on their own on-premises hardware or on the infrastructure of a certified cloud provider.

Features in OpenShift:

- OpenShift can scale to thousands of instances across hundreds of nodes.
- Its high flexibility and hybrid infrastructure allows for a self-managed service for on premise environments.
- Uses the OCI industry standard for portability between developers/workstations.
- Has an automated installation process (over-the-air platform) and is supported in a cloud with multiple cloud platforms.
- Offers advance security with features like access controls, networking, enterprise registry, and a built-in security scanner.
- Has persistent storage with a broad spectrum of enterprises storage solutions.

Pros of OpenShift:

1. Easy Deployment: OpenShift simplifies the deployment process by providing an intuitive web interface and a command-line interface (CLI) for managing applications and resources. It abstracts away much of the Kubernetes complexity, making it easier for developers to focus on application development.

2. Scalability: OpenShift allows seamless horizontal and vertical scaling of applications. It provides automated load balancing and intelligent scaling mechanisms that enable applications to handle increased workloads efficiently.

3. Built-in DevOps Features: OpenShift integrates CI/CD workflows seamlessly. It offers built-in support for source code management, continuous integration, and deployment pipelines. This allows for automated application builds, testing, and deployments, reducing manual effort and enabling faster development cycles.

4. Multi-tenancy and Isolation: OpenShift supports multi-tenancy, allowing different teams or projects to share the same cluster while maintaining resource isolation and security. Each project can have its own set of resources, policies, and access controls, ensuring separation and control over application environments.

5. Application Lifecycle Management: OpenShift provides features for managing the entire application lifecycle. It supports deployment strategies like rolling updates, blue-green deployments, and canary releases. It also offers application monitoring, logging, and auto-recovery mechanisms to ensure application availability and performance.

6. Integration with Red Hat Ecosystem: OpenShift is tightly integrated with the Red Hat ecosystem, which includes a wide range of tools, services, and technologies. This integration provides access to Red Hat's enterprise-grade support, security patches, and a vast library of certified container images and middleware components.

Cons of OpenShift:

1. Learning Curve: OpenShift adds an additional layer of complexity on top of Kubernetes. It requires users to learn OpenShift-specific concepts and configuration options. The learning curve can be steep, especially for developers and administrators who are new to containerization and Kubernetes.

2. Resource Requirements: OpenShift requires a significant amount of compute resources to run efficiently. The infrastructure needs to be adequately provisioned to handle the demands of running OpenShift clusters and containerized applications.

3. Cost: OpenShift is not a free open-source platform. Red Hat provides both community and enterprise editions, and the enterprise edition comes with licensing costs. Additionally, there may be costs associated with the infrastructure and resources required to run OpenShift clusters.

4. Limited Flexibility: OpenShift has its own options and configurations, which may limit flexibility compared to running a bare Kubernetes cluster. Customizing certain aspects or integrating with non-OpenShift tools or services may require additional effort or workarounds.

5. Version Compatibility: OpenShift releases are typically slightly behind the upstream Kubernetes releases. This lag may delay the availability of new Kubernetes features and updates on the OpenShift platform.

6. Vendor Lock-In: While OpenShift is built on Kubernetes and adheres to the Kubernetes API and standards, it has additional proprietary components and integrations. Choosing OpenShift and using these proprietary components and integrations may result in some level of vendor lock-in to Red Hat's ecosystem and tools.

2.3. Burp Suite Professional vs Burp Suite Enterprise

Burp Suite is an integrated platform for security testing that is developed by PortSwigger. This tool uses several different attacks to test a web application for variabilities and exploits. Burp Suite does this by mapping and analysis of an application's attack surface. Burp Suite has certain tools like Burp Intruder that automates attacks on web applications. Other tools include Burp Repeater and Burp Proxy. Burp can also be expanded by using BApps which enhance how people can run and apply

test. Some of the most popular BApps include Autorize, Turbo Intruder, Hackvertor, Burp Bounty, and Param Miner.

Attacks in Burp Suite:

- Brute force attacks
- Dictionary attacks
- Vector attacks
    - Sniper attack
    - Battering ram attack
    - Pitchfork attack
    - Cluster bomb

Most Popular BApps:

- Autorize
    - Used to help detect authorization vulnerabilities. This extension works without any configuration but, if need be, can be highly configurable. The way this works is by using low privileged users' session cookies in requests to test for proper access and authorization protection.
- Turbo Intruder
    - This tool is used for sending many HTTP request in a short amount of time. This allows the BApp to complement Burp Intruder by being able to customize the python snippet.
- Hackventor
    - Is a tag-based conversation BApp that supports multiple escapes and encodings.
- Burp Bounty
    - used to find advance flaws and variabilities.

Burp Suite Professional:

Burp Suite Professional is a manual web application testing tool that cannot be fully automated, but there are ways to automate processes. Using a CI/CD pipeline, users can integrate scripts written by the testing team and then they can be ran in an environment using a mockup of the testing webapp to run multiple scripts or one depending on the circumstance. Burp Suite professional is mostly used for manual testing and used to find advance flaws and variabilities.

Burp Suite Enterprise:

Burp Suite Enterprise can be fully automated and run through a CI/CD pipeline. It provides scaled security testing that can find vulnerabilities using dynamic security scanning. By integrating development and security Burp Suite Enterprise can reduce false positives, removes bottle necks, and avoids alerting fatigue.

List of all Vulnerabilities Tested:

For full list of vulnerabilities look at [62].

2.4. Invicti

Invicti is an automated and fully configurable web application security scanner that scans web applications, web services, websites to find security flaws. It can automatically exploit identified vulnerabilities in a read-only and safe way to confirm identified issues. It is designed to help securing web applications easily so team can focus on fixing the reported vulnerabilities.

Features in Invicti:

- Produces highly accurate web application security scans, whose vulnerabilities are verified [25].
- Proof-Based scanning technology actively and automatically verifies detected vulnerabilities.
- Identify vulnerability and safely exploits during the web vulnerability scan as Proof of Concept that it exists.
- Proof of Exploit reports the data that can be extracted from the vulnerable target once the vulnerability is exploited, demonstrating the impact of the exploited vulnerability.
- Able to integrate with wide range of software of tools that enable to connect with existing SDLC.

Pros of Invicti:

1. Discover + Crawl: Invicti crawls through advanced websites with heavy scripting and dynamically generated content. Allows complete visibility into all of the applications.
2. Detect: Invicti utilizes its unique combined DAST + IAST scanning to detect more vulnerabilities. Furthermore, combined signature-based and behavior-based scanning gives faster and accurate results.
3. Resolve: Invicti utilizes automation and workflow features manage security tasks. It also reduced false positive with Proof-Based Scanning and assigned confirmed vulnerabilities to developers. The detailed documentation helps developer fix issues faster.
4. Integrate: Invicti builds security into development by automatically giving the developer rapid feedback on secure code and catching vulnerability early in SDLC. This supports shift left principle in DevSecOps by helping developers tackle the security problem on their own.

5. Continuous Secure: Invicti's continuous security feature helps prevent delay and ensure fewer risks are introduced throughout the SDLC. It also gives automatic notification when deployed technology becomes outdated.

Cons of Invicti:

1. Cost: Invicti offers a free trail, pro plan and enterprise plan. The cost is available through contact of vendor, but it is said to cost more in compared to other options.
2. Complexity: While powerful and comprehensive, these tools have a steep learning curve and may require a significant amount of time and resources to fully understand and utilize effectively.
3. Overwhelming Information: The extensive detail provided by these tools might be overwhelming for less experienced users. Some teams might require additional training to interpret and act on the results effectively.
4. Limited to Web Applications: Invicti's tools are specifically designed for web application security, making them less suitable for organizations seeking a broader range of cybersecurity solutions.

2.5. OWASP ZAP

OWASP ZAP (Zed Attack Proxy) is an open-source web application security scanner developed by the OWASP. It's a powerful tool used for finding vulnerabilities in web applications during the development and testing phases. ZAP provides functionalities for a variety of security tests, including automated scanning, fuzzing, scripting, and proxying.

One of the primary reasons OWASP ZAP should be considered being used is because it supports a wide range of customizations and is suitable for different users, from developers to security experts. As an open-source tool, it's free to use, making it accessible for small to medium-sized businesses with tighter budgets. Its active community also ensures regular updates, new features, and quick fixes. Furthermore, ZAP can be effectively integrated into a CI/CD pipeline to automatically find vulnerabilities before deploying to production. The goal is to catch security issues as early as possible in the development process, minimizing the potential impact.

ZAP's API, command-line interface, or Docker image can all be used to integrate it into a CI/CD pipeline. Once integrated, it may be used to automatically run security checks and generate reports on any vulnerabilities that might be discovered in your apps. For instance, in a Jenkins pipeline, users can employ ZAP's official Docker image to run a container in a pipeline job. Then, as part of your build process, developers can start and control ZAP scans using command-line scripts or the ZAP API. Following a scan, ZAP offers a report outlining any vulnerabilities found, which may be examined and addressed. By including ZAP into your CI/CD pipeline, you're encouraging the use of DevSecOps by integrating application security into your development process.

2.6. Industry Standards for Pipelines

In developing a CI/CD pipeline, standards are essential, especially when several team members are working on creating and managing the CI/CD pipelines. These guidelines serve several reasons, such as keeping team members on the same branch of work and preventing confusion when developing code. They also set standards for executing automated tests that confirm the code's security and usability. Establishing standards is crucial when building a CI/CD pipeline and working together as a team. By following these principles, teams can keep their code development processes clear and consistent while eliminating potential misunderstandings and disputes. Additionally, using standardized methods, developers may optimize their operations and make sure they are working on the right code branches, reducing errors, and promoting productive cooperation.

It is essential to make sure building a CI/CD pipeline is not a "fire-and-forget exercise [20]." To do this properly, many companies and teams analyze how productive their pipeline is by listening to feedback and keeping the pipeline in a state of evolution, which allows you to constantly "refine your CI/CD process [20]." Some simple standards include:

- When using a software like GitLab, it is important to commit any changes early and do it as often as practical. By doing this, it allows the team to effectively work on the same code and scripts in the pipeline. It is very important to update the Main Brach as often as practical even if it feels uncomfortable.
- It is important to make a collaborative team environment in order to "keep the builds green [20]" by always keeping the code and structure releasable. This makes the teams main priority fixing the pipeline if any issues arise when pushing to the main branch group. Team members should not accuse the last person who pushed to the main branch. The main focus should be fixing the pipeline.
- When testing your pipeline for bugs and other issues it's important to run automated tests in order. The tests should go fastest to slowest because feedback is essential when changing or adding a process into a pipeline.
- "When environments are kept running for a long time it becomes harder to keep track of all the configuration changes and updates that have been applied to each one." [20] In order to prevent a situation like this groups can run environments in containers that can easily be deployed for testing.

2.6.1. List of Tools for CI/CD Pipelines

Tools	Description
Jenkins	One of the first and most well-known CI/CD tools is Jenkins. It is quite flexible and can be included with practically any product on the market, thanks to the large number of plugins it supports.
GitLab CI/CD	GitLab is a web-based DevOps lifecycle platform that includes GitLab CI. It is a strong CI/CD tool that is widely employed in the sector. Through the use of the GitLab CI/CD configuration file, it enables you to define pipelines for code.
GitHub Actions	GitHub Actions simplify automating all software workflows with CI/CD. You can build, test, and deploy your code right from GitHub.
Travis CI	A hosted continuous integration service called Travis CI is used to develop and test software projects that are stored on GitHub and Bitbucket.
Circle CI	Circle CI allows for faster development cycles and quickly pinpointing where a problem occurred.
Azure DevOps	Microsoft's approach to DevOps offers a cloud service for code sharing, project management, and software shipping.
Docker	Docker containers are quickly evolving into industry standards for packaging, deploying, operating applications and containerizing.
Kubernetes	Kubernetes is a portable, adaptable, open-source platform that supports declarative configuration and automation for managing containerized workloads and services. Frequently used with Docker.
Terraform	An Infrastructure as Code tool called Terraform is used to construct, modify, and version infrastructure in a secure and effective manner.

2.6.2. Microsoft Azure Standards

When looking at these considerations Microsoft uses the Pillars of Azure framework.

Pillar	Description
Reliability	"The ability of a system to recover from failures and continue to function [22]"
Security	"Protecting applications and data from threats [22]"
Cost optimization	"Managing costs to maximize the value delivered [22]"
Operational excellence	"Operations processes that keep a system running in production [22]"
Performance efficiency	"The ability of a system to adapt to changes in load. [22]"

Operational Excellence:

Infrastructure as Code:

"Infrastructure as code (IaC) uses DevOps methodology and versioning with a descriptive model to define and deploy infrastructure, such as networks, virtual machines, load balancers, and connection topologies. Just as the same source code always generates the same binary, an IaC model generates the same environment every time it deploys." [24]
IaC has many advantages, making it a crucial tool for modern DevOps procedures. First, by enabling teams to automate the process of setting up and modifying IT infrastructures, which traditionally might be time-consuming and prone to human mistakes, IaC helps with the quick and simple deployment of servers. IaC has the advantage of version control, easier rollbacks, and reduced inconsistencies caused by human setups because the infrastructure setup is defined as code, which can be versioned and reviewed like any other software code. Additionally, it helps preserve consistency by facilitating disaster recovery and replicating the infrastructure across several environments.

The advantages of IaC also include raising team productivity and effectiveness overall. It encourages the practice of treating "infrastructure as software" and permits adopting software development approaches like continuous integration/continuous deployment (CI/CD), which aid in reducing delivery times and improving product quality. IaC also encourages team communication, bridging the gap between development and operations and advancing the DevOps culture. Lastly, capacity planning and monitoring infrastructure usage over time help firms manage risk and cost more effectively. The operational efficiency, dependability, and robustness of IT environments can be considerably improved using Infrastructure as Code.

Self-Hosted Agents:

"With Microsoft-hosted agents, maintenance and upgrades are taken care of for you. Each time you run a pipeline, you get a fresh virtual machine for each job in the pipeline. The virtual machine is discarded after one job (which means any change that a job makes to the

virtual machine file system, such as checking out code, will be unavailable to the next job)." [25] There is also an option call "Self-hosted agents give you more control to install dependent software needed for your builds and deployments. Also, machine-level caches and configuration persist from run to run, which can boost speed." [25]

When Sandia is looking to streamline their development, operations benefit by utilizing Microsoft-hosted agents in Azure DevOps. These hosted agents' main benefits are their simplicity of use and seamless connection with the Microsoft ecosystem. These agents save time and work because they are pre-configured, ready-to-use environments that do not require human installation or upkeep. Each sprint deployment automatically updates, ensuring that your projects always use the most recent and secure versions. Furthermore, various application kinds, languages, and frameworks are supported by hosted agents, giving project development flexibility.

Another important advantage of using Microsoft-hosted agents is scalability. Azure gives you the freedom to scale up or down to the requirements of your pipeline, enabling optimal resource use. Costs associated with infrastructure are decreased because there is less demand for specialized hardware resources. Additionally, since Microsoft promises 99.9% agent availability, there is no need to worry about it. Multiple jobs can run simultaneously with parallel processing provided by Microsoft-hosted agents, accelerating the build and deployment process. Last, Azure's Pay-As-You-Go approach guarantees that you only pay for what you use, making it an economical choice for companies of all sizes.

Cost Optimization:

Managing and reducing operational expenses in the Microsoft Azure cloud ecosystem requires a fundamental approach called Azure cost optimization. Azure resources, such as VM instances, storage, databases, and bandwidth, may all be used more effectively to reduce costs over time. The ineffective use of resources, bloated budgets, and skyrocketing costs can result from a lack of cost optimization. This is where Azure cost optimization excels; it makes use of automation, right-sizing, and other best practices for cloud management to assist organizations in making sure they are only paying for what they require. The advantages include higher ROI, more simplified and effective cloud infrastructure, and costs that are more predictable and controlled. Organizations can better match their cloud resources with their business demands by utilizing Azure cost optimization, which can help free up funds for strategic initiatives, and fostering business growth.

Security:

Azure Security integration into a pipeline provides a wide range of advantages that are essential for protecting data and guaranteeing compliance with security laws. Data encryption, advanced threat protection, threat intelligence, and identity and access management are just a few of the powerful, enterprise-grade security capabilities offered by Azure Security. Organizations can discover and address security issues early by including these features in the development and deployment pipeline, which will stop vulnerabilities from entering the production environment. With such a proactive approach, data breaches and other security issues are much less likely to occur, safeguarding a company's reputation as well as its financial health. As it has built-in compliance tools, using Azure

Security also makes it easier to comply with different security standards. The pipeline is thus shielded from any threats by Azure Security, assuring security.

- "Ensure all changes to environments are done through pipelines. Implement role-based access controls (RBAC) on the principle of least privilege, preventing users from accessing environments" [21]
- Consider integrating steps in Azure Pipelines to track dependencies, manage licensing, scan for vulnerabilities, and keep dependencies to date. [21]

2.6.3. Jenkins

Jenkins is a widely used tool for implementing continuous integration and continuous development for pipelines. It is a server base system that runes serval containers like Apache Tom. Jenkins also supports version control tools like AccuRev, CVS, Subversion, Git, Mercurial, Perforce, ClearCase and RTC. Jenkins can also execute Apache Ant, Apache Maven and SBT based projects as well as shell scripts and Windows batch commands.

Practice	Description
Scripted Pipelines	Use scripted pipelines for straightforward projects, where the pipeline script resides within the Jenkinsfile in the root directory of your repository. This approach allows the same Jenkinsfile to be used across all branches of your project.
Declarative Pipelines	Declarative pipelines provide more structure and flexibility than scripted pipelines, especially for complex projects. Declarative pipelines also make the pipeline code easier to read and write.
Pipeline as Code	Store Jenkins pipeline scripts (Jenkinsfile) within your version control system (like Git) and follow the 'Pipeline as Code' principle. This allows version control, code review, full audit trails, and a single source of truth.
Use Shared Libraries	Shared Libraries allow common steps to be shared across multiple pipelines. This promotes code reuse and reduces redundancy in Jenkinsfiles.
Parameterized Pipelines	Use parameters in pipelines when you want to execute the same pipeline with different configurations. Parameters allow pipelines to be more dynamic and flexible.
Parallel Execution	Jenkins pipelines allow you to run steps in parallel, which can significantly reduce the total time taken for a pipeline run. Use this feature wisely to speed up your CI/CD process.

Practice	Description
Automated Testing	Incorporate automated testing into the Jenkins pipeline. This includes unit tests, integration tests, and any other tests relevant to your project. The earlier you catch issues in your code, the easier they are to fix.
Build Artifact Management	Use Jenkins' Archive Artifacts step to store important files that are produced as part of your build process. This can include compiled binaries, test results, and more.
Security	Implement proper security measures in Jenkins. This includes limiting job configuration permissions, using credentials-binding-plugin to handle sensitive data, and regularly updating Jenkins and its plugins to their latest stable version.
Notifications	Use notifications to alert the relevant people when a pipeline fails. Jenkins can send notifications to various channels including email, Slack, and more.
Docker Usage	Use Docker containers for build environments. This ensures that the build environment is consistent and reproducible.
Regularly Cleanup	Regularly clean up old build records and unused plugins to ensure that your Jenkins instance is not using unnecessary resources.

2.6.4. GitLab

GitLab is a well know tool for implementing pipelines and it has a set of standards to help develop a CI/CD pipeline. GitLab provides seven different standards that can be implemented in order to achieve the desired outcome that is needed. These pipelines designs and practices include:

- Basic pipelines run everything in each stage concurrently, followed by the next stage. [31]
- Directed Acyclic Graph Pipeline (DAG) pipelines are based on relationships between jobs and can run more quickly than basic pipelines. [31]
- Merge request pipelines run for merge requests only (rather than for every commit). [31]
- Merged results pipelines are merge request pipelines that act as though the changes from the source branch have already been merged into the target branch. [31]
- Merge trains use merged results pipelines to queue merges one after the other. [31]

- Parent-child pipelines break down complex pipelines into one parent pipeline that can trigger multiple child sub-pipelines, which all run in the same project. This pipeline architecture is commonly used for mono-repos. [31]
- Multi-project pipelines combine pipelines for different projects together. [31]

Practice	Description
Pipeline Configuration	GitLab uses a YAML file (.gitlab-ci.yml) located at the root of the repository to manage the CI/CD pipeline. This file contains all the configurations and defines stages, jobs, and actions that should be executed during pipeline runs.
Pipeline Stages	A common standard is to divide the pipeline into distinct stages, such as build, test, and deploy. Each stage comprises different jobs that run in parallel.
Automated Testing	Automating your testing in the test stage is an essential practice in CI/CD. This could involve unit tests, integration tests, and even UI/functional tests depending on your application.
Docker Usage	Docker and containerization have become a standard for building, testing, and deploying applications. GitLab CI/CD integrates seamlessly with Docker.
Artifact Management	GitLab allows you to define and manage artifacts, which are the output files from each job. These can be passed between stages or stored for later use.
Environment Management	GitLab provides support for handling different environments like staging, production, and others. You can configure specific jobs to run only on specific branches and in specific environments.
Review Apps	GitLab feature automatically creates a live preview of changes made in a branch. Review apps are extremely useful for reviewing changes without affecting the main application.
Security Scanning	GitLab feature automatically creates a live preview of changes made in a branch. Review apps are extremely useful for reviewing changes without affecting the main application.
Monitoring and Analytic	GitLab offers detailed monitoring and analytics tools, such as CI/CD Pipeline charts, to help track the performance and status of your pipelines.
Infrastructure as Code	GitLab is compatible with various IaC tools like Terraform, Ansible, etc. IaC is a key practice in the DevOps industry.

2.7. Industry Standards for Load and Stress Testing

Load and stress testing are critical components of a comprehensive testing strategy, especially for web applications or other network-based applications. These tests are conducted to determine how an application operates under heavy load and how it responds when the load is greater than the application's claimed capacity. The following are some key recommendations for load and stress testing:

Practice	Descriptions
Plan for the Test	Before beginning testing, develop a detailed plan outlining objectives, target system, test scenarios, and performance acceptance criteria.
Define Test Environment	Identify the physical test environment, including other software, hardware, and network configurations. Ensure you have a similar setup to the production environment.
Design Test Scenarios	Identify key scenarios that will be stress and load tested. These are often the most common or resource-intensive operations.
Identify Performance Metrics	Metrics might include response time, throughput, hits/sec, concurrent users, error rates, resource utilization like CPU, memory, network I/O, etc.
Create Load Profile	Identify the different types of users and their patterns, simulate real-world user load diversity in your test scenarios.
Automate	Use automated load testing tools such as Apache JMeter, LoadRunner, Gatling, or Locust. These tools can simulate a large number of users and record key metrics.
Incremental Testing	Start with small load tests and progressively increase the load. This will help identify performance bottlenecks.
Run Stress Test	Push the system beyond its designed capacity to identify its breakpoint or recoverability after failure.
Analyze, Report, and Retest	After testing, analyze the data to find bottlenecks and problems. Create a detailed report and share it with all concerned stakeholders. Make necessary code or system changes and retest to ensure problems are fixed.

Practice	Descriptions
Continuous Testing	Perform these tests regularly, especially after major code updates or infrastructure changes. This will help ensure that system performance remains high as the system evolves. Integrate these tests into your CI/CD pipeline if possible.
Monitor stress and load test	Monitor application during testing to identify any performance issue. Use application performance management tools to get detailed insights into the system's behavior under load.
Test under Realistic Conditions	Emulate real-world conditions as much as possible. This includes simulating various network speeds, user geographic locations, and different types of user interactions.

2.7.1. Tools for Load and Stress Testing

The main tool that is used for load and stress testing is Neoload. There are numerous other tools that can be used with Neoload Web or standalone which include:

Tool	Description	Industry/Industries
Apache JMeter	"Apache JMeter may be used to test performance both on static and dynamic resources, Web dynamic applications. It can be used to simulate a heavy load on a server, group of servers, networks or objects to test its strength or to analyze overall performance under different load types." [46]	• Information Technology and Services • Computer Software
SOAPUI/ReadyAPI TestEngine	"TestEngine allows testing, development, and operations groups to seamlessly run ReadyAPI and SoapUI OS functional & security tests as part of their Azure DevOps process. Deploying TestEngine as part of a larger collection of cloud services allows quality checks to be executed alongside deployed applications,	• Information Technology and Services • Computer Software

Tool	Description	Industry/Industries
	eliminating the need to expose individual components to external testing solutions." [69]	
Gatling	"Gatling is a load test tool. It officially supports HTTP, WebSocket, Server-Sent-Events and JMS." [70]	• Information Technology and Services • Computer Software

2.8. Industry Standards for Security Testing

2.8.1. Introduction

Security testing is the process of determining whether software is vulnerable to cyber-attacks. Testing software can come in many forms, including penetration testing, crawling through a web application, and automatic testing. Ultimately, secure software starts with secure coding practices, but security testing provides evidence that systems and information are safe, reliable, and that they do not accept unauthorized inputs [60]. However, security testing is not functional testing, meaning that it will not test whether a specific application works as expected or produces the correct output.

The main goals of security testing include identifying assets, threats and vulnerabilities, and risks, and then displaying results and performing remediation [60].

Goals	Descriptions
Identify Assets	Assets are things that need to be protected, such as software applications and computing infrastructure.
Identify Threats and Vulnerabilities	Threats and vulnerabilities are activities that can cause damage to an asset, or weaknesses in one or more assets that can be exploited by attackers.
Identify Risk	Security testing aims to evaluate the risk that specific threats or vulnerabilities will cause a system. Risk is evaluated by identifying the severity of a threat or vulnerability, and the likelihood and impact of exploitation.
Perform Remediation	Security testing is not just a passive evaluation of assets. It provides actionable guidance for remediating vulnerabilities discovered and can

Goals	Descriptions
	verify that vulnerabilities were successfully fixed.

2.8.2. General Security Testing

Below is a list of general types of security testing that should be implemented into the security testing process:

- Review source code and end-to-end architecture
- Generate a product architecture and dataflow model
- Retest findings as they are fixed
- Application Security Testing [49, Figure 1]
- Automated configuration scanning
- Conduct security audits and risk assessments
- Vulnerability management
- Black box (dynamic) and white box (static) testing

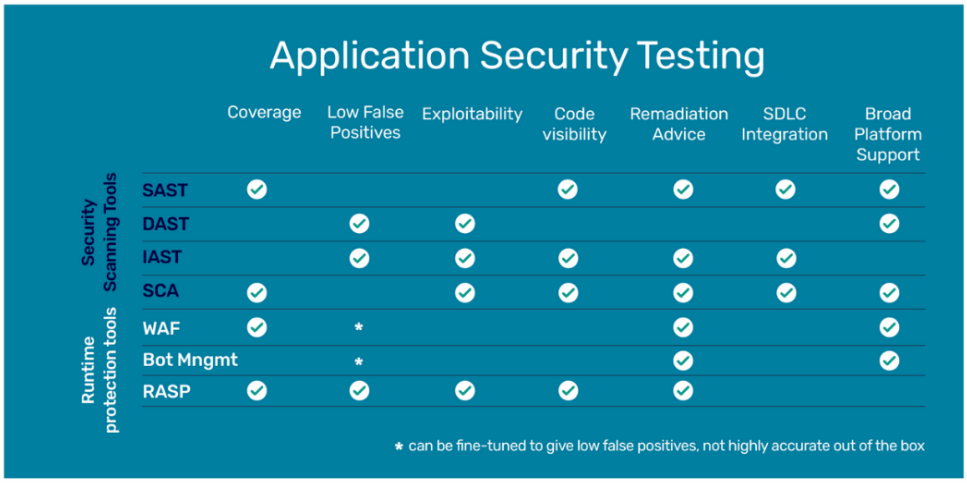


Figure 1 - Application Security Testing [49]

Reference Documents:

The following are documents that contain security testing standards. Documents with an asterisk will be discussed in more depth later in the paper:

- NIST 800-53: Security and Privacy Controls for Information Systems and Organizations
- NIST 800-63b: Authentication and Lifecycle Management
- NIST 800-115: Technical Guide to Information Security Testing and Assessment

- NIST Cybersecurity Framework
- * Microsoft Security Assessment (Section 2.6.4)
- * Microsoft CIS (Section 2.6.4)
- * OWASP ASV (Section 2.6.6)

2.8.3. Typical Findings

Some discovered vulnerabilities are more common than others. Depending on the type of application and the data being protected, an application may be more prone to certain vulnerabilities. Understanding how your assets or information are viewed by adversaries will give you a better idea of the types of vulnerabilities your system is prone to. Below is a list of common vulnerabilities found when conducting security tests:

- Insufficient authorization controls
- Improper input validation
- Insecure coding practices
- Compromised root cloud account credentials
- Insecure methods of storing cryptographic secrets

Some common results of vulnerable web applications include application logs leaking sensitive information, adversaries reverse engineering the application, and applications that are vulnerable to MiTM attacks.

The OWASP also provides organizations with the top 10 vulnerabilities in web applications, mobile security, and API's [61]. Below are the top 10 vulnerabilities in web applications (a link to the OWASP Top 10 can be found at [61]):

- Broken Access Controls
- Cryptography Failures
- Injection Vulnerabilities
- Design Flaws
- Security Misconfigurations
- Outdated and Vulnerable Components
- Authentication Issues
- Integrity Vulnerabilities
- Security Monitoring and Logging Failures
- SSRF Vulnerabilities (server-side request forgery)

2.8.4. Microsoft Security Testing Standards

Microsoft defines security testing as testing to make sure "that at no point can somebody gain unauthorized access to data" [59]. Microsoft security testing standards are used to make sure that an authorized or unauthorized user can't impersonate another user, gain access to somebody else's data, or cause a system in the architecture to do something that the developers or engineers never expected to happen.

Common Mistakes and Fixes:

Microsoft highlights the importance of conducting proper and reliable security tests by explaining the three most common mistakes that organizations make when creating software applications.

1. The industry is still relying on final application testing at the end of the software development life cycle.
    a. Instead, organizations should constantly be looking, during the entire lifecycle of the application, for bugs, misconfigurations, and the incorrect usage of encryption.
2. Today's applications are complex, which results in multiple teams that are responsible for individual parts of the process. Unfortunately, these teams don't all work together to understand the data flow of the applications, which can lead to essential parts of secure coding development being overlooked.
    a. There should be a central person who has full visibility of the entire application. There needs to be someone who understands what all components are doing and how they interact with each other.
    b. Teams should overlap people and parts of the project to promote cohesion.
    c. There should be weekly meetings, at least once a week if not more, to update of teams, update the person who has full visibility, and to make sure that teams are not in standstills with one another.
3. Unfortunately, there is still a large gap between software developers and security professionals. Developers will create software without having security in mind, then security professionals will have to send back entire applications because they don't follow industry standards.
    a. Going back to point 1.1, applications should be designed and created with security in mind. Developers and security should be working together throughout the entire lifecycle of application development.

Microsoft Security Standards Resources:

Microsoft provides three Microsoft specific security testing resources to conduct security assessments and develop more secure software. Additionally, Microsoft uses the OWASP ASVS standards and NIST CSF, which consists of standards, guidelines, and best practices to manage cybersecurity-related risks.

Microsoft Security Assessment:

"The Cloud Collective Microsoft Security Assessment (MSA) is a security review performed in accordance with industry best practices that includes recommendations to improve your security posture and to address security issues that are discovered as part of this review" [57]. As the cloud provider, Microsoft provides a service for conducting a security assessment, prepared and performed by a certified senior consultant with a deep understanding of cloud and on-premises security technologies, aimed at identifying potential security gaps.

Figure 2 - MSA Process [57]

Center for Internet Security Benchmarks:

Microsoft's CIS provides benchmarks that act as "configuration baselines and best practices for securely configuring a system" [53]. CIS benchmarks are internationally recognized as security standards for defending IT systems and data against cyberattacks. Each of the guidance recommendations references one or more CIS controls that were developed to help organizations improve their cyber defense capabilities. The CIS controls are also mapped to a variety of "established standards and regulatory frameworks" including the NIST CSF, NIST SP 800-53, and the ISO 27000 series of standards.

CIS benchmarks provide two levels of security settings [53]:

- **Level 1** recommends essential basic security requirements that can be configured on any system and should cause little or no interruption of service or reduced functionality.
- **Level 2** recommends security settings for environments requiring greater security that could result in some reduced functionality.

In addition to the benchmarks for Microsoft products and services, CIS has published CIS Hardened Images on Azure configured to meet CIS Benchmarks. "They've been pre-tested for readiness and compatibility with the Microsoft Azure public cloud, Microsoft Cloud Platform hosted by service providers through the Cloud OS Network, and on-premises private cloud Windows Server Hyper-V deployments managed by customers" [53]. CIS Hardened Images are securely configured virtual machine images based on CIS Benchmarks hardened to either a Level 1 or Level 2 CIS Benchmark profile.

Azure Security Test Practices:

Microsoft's Azure recommends conducting penetration tests and simulated attacks. Penetration testing is good for simulating one-time attacks in order to detect vulnerabilities while simulated

attacks are good for indicating security gaps within an application by conducting long-term persistent attacks.

These kinds of testing can result in data that allows for the planning of more efficient risk mitigation. This data can help organizations identify and manage the lowest cost methods for preventing and detecting attacks. Penetration testing and simulated attacks helps organizations make sure that their "security assurances are effective and maintained as per the security standards set by the organization" [51].

2.8.5. Microsoft Security Testing Tools

The Microsoft Security Assessment Tool:

The MSAT is for tracking down network security problems [58]. The MSAT is a free tool which is composed of an electronic questionnaire in which you describe your security environment. The questions are broken up into different categories, and once completed, it provides an analysis of your situation and recommendations on how to improve it. The MSAT offers three tools: the Summary Report, the Complete Report, and the Comparison report. The Summary Report displays two bar graphs as a result of the questions answered. There is a Business Risk Profile and Defense-in-Depth Index graph which represent risk and security level respectively. The Complete Report indicates whether the system meets best practices and if it needs improvement. Finally, the Comparison Report gives you the option to upload the results to the MSAT Web site where they can be compared with those of other organizations.

Microsoft Security Code Analysis:

This tool is aimed at implementing the SDL into software development. MSCA is a toolset used to "integrate security controls into [the] development process" [55]. MSCA will enable organizations to integrate Shifting Left, Secure and Complaint Pipeline, Automation, and Secure Credentials principles into a CI/CD pipeline when using Azure DevOps. Both static and dynamic application testing are included within the MSCA toolset. Integration with Azure DevOps makes it easy to automate tests and therefore catch and remediate security issues early and often within the SDL. MSCA also provides scanning tools to help ensure that the pipeline and DevOps team are following best practices. Risks can be reduced because MSCA can be used to address issues with the code by notifying developers and blocking Pull Requests when issues are found.

Microsoft Attack Surface Analyzer:

The ASA is a security tool that analyzes the attack surface of Windows and Linux systems and reports potential security implications. It can be used to help identify potential security risks exposed through system changes and ensure that the software development process is following standard best practices. ASA provides evidence that the code is doing only what it claims to do and that it follows practices for least privilege and reduction of attack surface.

Typical users of ASA [56]:

- DevOps Engineers - View changes to the system attack surface introduced when your software is installed.

- IT Security Auditors - Evaluate risk presented when third-party software is installed.

ASA comes with both CLI and GUI options.

2.8.6. PortSwigger Setup Recommendations

Burp Suite is an integrated platform and visualization tool for performing security testing of web applications in CI/CD pipelines. Burp Suite is a dynamic testing tool that allows for automatic and manual testing of a variety of security vulnerabilities, these vulnerabilities can be found at [64].

Below are some recommendations for effectively setting up and using Burp Suite [53], [63].

- For larger, more dynamic scan targets with complex website interactions, these are the recommend specifications:

Concurrent scans	CPU cores	Ram (GB)	Free disk space (GB)	Swap space (Linux only)
1	4	8	30	10
2	8	16	50	18
3	12	24	70	26
4	16	32	90	34
5	20	40	110	42
10	40	80	210	82

- Make use of browser profiles so that setting, cookies, etc. are cleared for your web testing profile.
- Use dedicated server-class machines and separate enterprise server and scanning machines.
- Use an external database that follows Burb Suite system requirements.

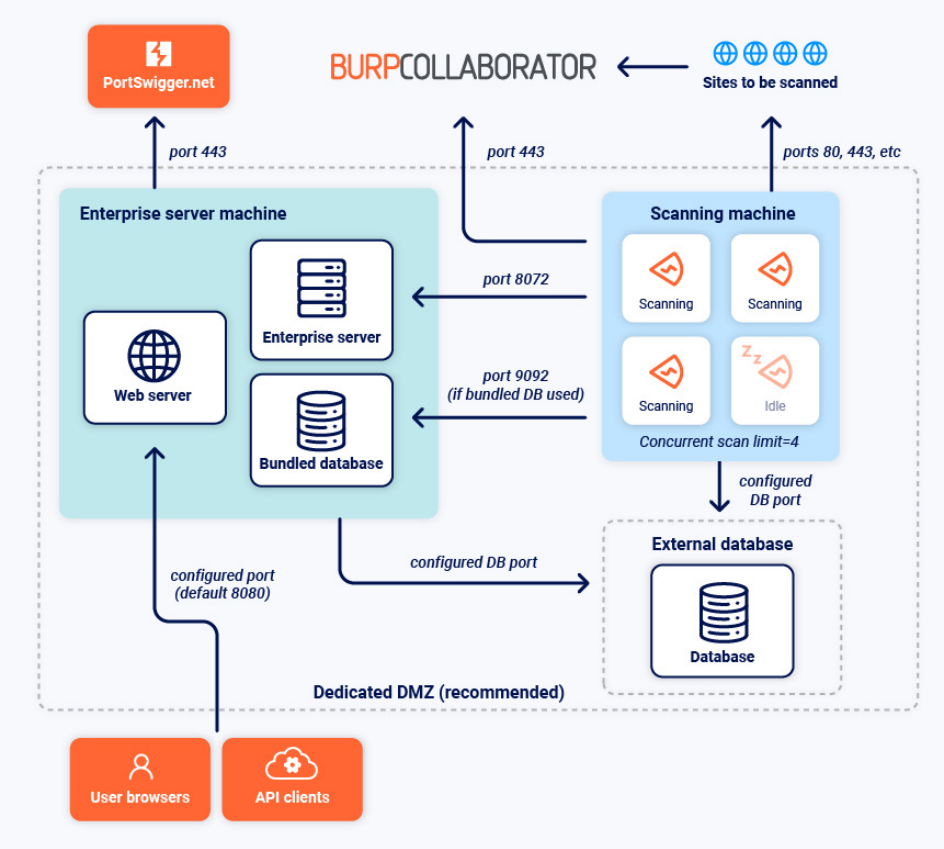


Figure 3 - Recommended Burp Suite Setup [63]

2.8.7. OWASP ASVS Review

The Open Web Application Security Project (OWASP) Application Security Verification Standard (ASVS) Project provides a basis for testing web application technical security controls and provides developers with a list of requirements for secure development [62]. The OWASP ASVS is a comprehensive list of requirements and test for software developers and security professionals to build, test and verify secure applications. The OWASP ASVS can be found at [50].

The OWASP ASVS is organized into 14 chapters, each dealing with a different aspect of web application security. Each chapter contains several sections. All requirements within a section and chapter can be referenced using the identifier format <chapter>.<section>.<requirement>. Each requirement has a check mark indicating which OWASP ASVS level it is suitable for. Organizations are strongly encouraged to look deeply at their unique risk characteristics based on the nature of their business, and based upon that risk and business requirements, determine the appropriate ASVS level.

These levels allow for the tradeoff between security and resources.

"Tailoring the ASVS to your use cases will increase the focus on the security requirements that are most important to your projects and environments" [50].

- **Level 1** - the bare minimum for an application

- Can be automatic or manual.
- Adequately defends against application security vulnerabilities that are easy to discover and included in the OWASP Top 10.
- Protects against attackers who are using simple and low effort techniques to identify easy-to-find and easy-to-exploit vulnerabilities.
- Can be used as a first step to protect applications that don't handle sensitive information.

- **Level 2** - the standard for most applications
  - Adequately defends against most of the risks associated with software today.
  - Ensures that security controls are in place, effective, and used within the application.
  - Protects against attackers that are typically skilled and motivated, focusing on specific targets, using tools and techniques that are highly practiced, and effective at discovering and exploiting weaknesses within applications.
- **Level 3** - high value applications that perform critical functions and CUI information
  - Requires more in-depth analysis of architecture, coding, and testing.
  - A secure application is modularized in a meaningful way (layers of security), and each module takes care of its own security responsibilities (defense in depth), that need to be properly documented.
  - Responsibilities include controls for ensuring confidentiality, integrity, availability, authentication, authorization, and auditing.

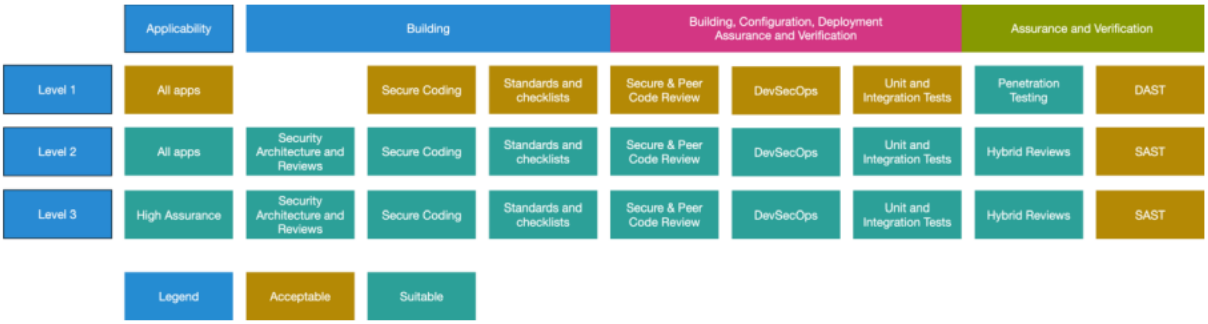


Figure 1 - OWASP Application Security Verification Standard 4.0 Levels

Figure 4 - OWASP ASVS Levels [50]

2.9. *Industry Standards for DevSecOps*

2.9.1. *Introduction*

DevSecOps helps ensure security is addressed to all part of DevOps practices by integrating security practices, automatically generating security and compliance artifacts throughout the processes and environments, including software development, builds, packaging, distribution, and deployment [45]. It is important for reasons of:

- "Reduces vulnerabilities, malicious code, and other security issues." [45]
- "Mitigates the potential impact of vulnerability exploitation throughout the application lifecycle." [45]
- "Addresses the root causes of vulnerabilities to prevent recurrences such as strengthening test tools and methodologies in the toolchain and improving practices for developing code and operating hosting platforms." [45]
- "Reduces friction between the development, operation, and security teams." [45]

The most important principle of DevSecOps is shifting left, which minimizes any technical problem that would require remediating later in development or after software is in production [45]. This helps not only reduce the cost but also result in software with stronger security.

2.9.2. Microsoft Secure DevOps Standards

When looking at these considerations Microsoft uses following practices:

Practice	Description	Useful Link
Provide Training	"Ensuring that everyone understands the attacker's perspective... help capture the attention of everyone." [23]	N/A
Define Security Requirements	"Establish a minimum-security baseline that takes account of both security and compliance controls." [23]	Azure DevOps / Azure Boards [72]
Define Metrics and Compliance Reporting	"Defining specific metrics which drive action and support compliance objectives." [23]	SDL Security Bug Bar Sample [73]
Use Software Composition Analysis (SCA) and Governance	"The impact that a vulnerability in the component could have on the overall security of the system." [23]	Secure Supply Chain Consumption Framework [74]
Define and use Cryptography Standard	"Only use industry-vetted encryption libraries and ensure easily replacement if needed." [69]	Microsoft SDL Cryptographic Recommendations [75]

Practice	Description	Useful Link
Perform Threat Modeling	"A critical component of any secure development process." [23]	Threat Modeling [76]
Use Tools and Automation	"Integrating Static Application Security Testing (SAST) into your IDE … can provide deep analytical insight." [23]	Microsoft Security Code Analysis [77]
Keep Credentials Safe	"Scanning for credentials and other sensitive content in source files is necessary." [23]	Credential Scanner [78]
Use Continuous Learning and Monitoring	"Gain better visibility into your application health and proactively identify and mitigate risks to reduce exposure to attacks." [23]	Advanced Threat Detection [79]
Manage Security Risk of using Third-Party Components	"Having an accurate inventory of third-party components and a plan to respond for new vulnerabilities are discovered." [69]	Managing Security Risks Inherent in the Use of Third-Party Components [80]
Use Approved Tools	"Define and publish a list of approved tools and their associated security checks." [69]	Managing Security Risks Inherent in the Use of Third-Party Components [80]
Perform Penetration Testing	"Penetration tests performed in conjunction with automated and manual code reviews provide greater level of analysis." [69]	SDL Tools [81]
Establish a Standard Incident Response Process	"It should be created in coordination with your organization's dedicated Product Security Incident Response Team (PSIRT)." [69]	Attack Surface Analyzer [82] SDL Security Bug Bar Sample [73]

Conduct Threat Modeling:
Performing threat modeling allows for the identification of potential threats and vulnerabilities, and the enumeration of possible mitigating controls. This helps to secure applications and services during the production run-time stage [27]. Additionally, it is important to secure the artifacts, CI/CD pipeline, and other tooling environments used for building, testing, and deployment [27].

The threat modeling should have aspects of:

- Defined security requirement of the applications [27].
- Analyze application components, data connectors and their relationship [27].
- Potential threats and attack vectors that application components might be exposed to [27].
- Identify applicable security controls to mitigate the threats [27].
- Identify potential controls gaps that require additional treatment [27].
- Enumerate and design the controls that can mitigate the vulnerabilities identified [27].

The use of the STRIDE model can help to enumerate threats from both internal and external sources and identify applicable controls [27]. It is important to ensure that the threat modeling process includes threat scenarios in the DevOps process. This can help to improve the security of applications and services.

Software Supply Chain Security:

"Ensure enterprise's SDLC include a set of security controls to govern in-house and third-party software component is important [27]." Having a defined gating criteria to prevent vulnerable or malicious components being integrated and deployed into the environment is important. It should have at least the following aspect:

- "Inventory and track the in-house and third-party software components for known vulnerability when there is a fix available in the upstream [27]."
- "Assess the vulnerabilities and malware in the software components using static and dynamic application testing for unknown vulnerabilities [27]."
- "Ensure the vulnerabilities and malware are mitigated using the appropriate approach such as include source code local or upstream fix, feature exclusion and/or applying compensating controls if the direct mitigation is not available [27]."

If using closed source third-party component, it's important to consider additional controls such as "include source code local or upstream fix, feature exclusion and/or applying compensating controls if the direct mitigation is not available [27]."

For Azure DevOps, "third-party extensions can be implemented to analyze and remediate third-party software components and vulnerabilities [27]."

Secure DevOps infrastructure:
Ensure the DevOps infrastructure and pipeline follow the best practices mentioned above along with security control for following scopes:

- CI/CD pipeline configuration [27].
- Servers, services, and tooling that hosting CI/CD pipelines [27].
- Artifact repositories that store source code, built packages and images, project artifacts and business data [27].

The following controls should be prioritized:

- "Protect artifacts and underlying environment to ensure the CI/CD pipelines don't become avenues to insert malicious code [27]."
- "Avoid providing permanent "standing" privileged access to the human accounts such as developers or testers [27]."
- "Remove keys, credentials, and secrets from code and scripts used in CI/CD workflow jobs and keep them in key store or Azure Key Vault [27]."

Integrate static/dynamic application security testing into DevOps pipeline:
Ensure SAST and DAST are a part of gating controls in the CI/CD flow. "The gating can be set based on the testing results to prevent vulnerable packages from committing into the repository, building into the packages, or deploying into the production [27]."

Integrate SAST into the pipeline for source code to be scanned automatically and DAST for runtime application. "The automated penetration testing (with manual assisted validation) should also be part of the DAST [27]."

Enforce security of workload throughout DevOps lifecycle:
Ensure the workload is secured throughout the entire lifecycle in development, testing, and deployment stage. Use Azure Security Benchmark to evaluate the control and ensure the following controls are in place:

- "Ensure VMs, container images and other artifacts are secure from malicious manipulation [27]."
- "Automate the deployment by using Azure in CI/CD workflow, infrastructure management, and testing to reduce human error and attack surface [27]."
- "Scan the workload artifacts prior to deployment [27]."
- "Deploy vulnerability assessment and threat detection capability into the production environment and continuously use these capabilities in the run-time [27]."

2.9.3. DoD DevSecOps standard

DevSecOps is recognized as the central theme of software modernization across the DoD that enables the delivery of resilient software capability [31]. In the paper they suggested following guiding principles which create guardrails for Sandia's DevSecOps teams.

Relentless pursuit of Agile:

Core competencies which define functional relationships that Sandia's EMTS team should value:

- "Individuals and Interactions over Processes and Tools [31]."

- "Working Software over Comprehensive Documentation [31]."
- "Customer Collaboration over Contract Negotiations [31]."
- "Responding to Change over Following a Plan [31]."

These competencies emphasize a certain value over the other, but it shouldn't be interpreted as of the latter are irrelevant. This should be seen as the latter are also important but not at the cost of former.

Software factories mandate baked-in security:

In DoD, a key differentiator in DevSecOps is "functional and security capabilities are tested and built simultaneously, with a series of recognized control gates that aim to prevent defect escapes and enhance the cyber survivability of the software artifact before release into the next environment [31]." Along with the use of Azure's cyber defender tools such as Microsoft Defender for Cloud, Sandia will be able to better monitor and have better feedback for future updates and patches.

Integrated, automated & continuous end-to-end testing and monitoring:

The shift towards "Continuous Authorization to Operate (CATO) stipulates continuous testing and monitoring that shifts the risk assessment further left to evaluate the people, platform, and processes using real-time data-driven metrics [31]." Each phase of the DevSecOps in Sandia should contribute in their own unique way to the real-time performance metrics along with continuous monitoring.

Immutability of infrastructure achieved via "x as Code" design patterns:

Sandia should use "Infrastructure as Code, Policy as Code, and Everything as Code techniques provides security and value in a number of ways [31]" for the shift to immutability of infrastructure. It helps avert error-prone step-by-step manual deployments and configuration. It also confirms commands are executed as expected. This principle "establishes a clear mandate for automated infrastructure configuration driven by code. Code can be version controlled, tested, peer reviewed, and its execution (logs) tracked [31]."

2.9.4. NIST for Secure DevSecOps

NIST is a "collaborative hub where industry organizations, government agencies, and academic institutions work together to address businesses' most pressing cybersecurity challenges [45]." They utilize existing guidance, practice, and recommendations that's applicable to develop their standards for DevSecOps.

NIST's Value Proposition:

NIST's risk-based approach for secure software development have several positive impacts.

More security with little delay:

This approach is "designed to implement better security while minimizing the delay to software production [68]." As the security responsibilities shift left to development, it helps shorten the time for security precautions like source code review and threat modeling.

Mitigate vulnerabilities through the software development lifecycle (SDLC):

This approach aimed to improve security through all phase of the SDLC, not just the development. It is important to test the application throughout the cycle as some vulnerability may show up in one set of testing but not in the other [68].

Reduce friction between Development, Operations and Security Teams:

It is essential to make sure the company has the speed and agility to meet the organization's goal utilizing the automation and latest security technology. Along with security testing built in the automation, it's also important that the security team have visibilities into development teams' operations and workflow so they can provide advice on mitigating insecure practices [68].

Software Supply Chain and DevOps Security Practices:

Most software today relies on third-party components, which is a security vulnerability as organizations have "little to no visibility into or understand how it is developed, integrated, deployed, and maintained [45]." The security concerns tied into third-party components need to be addressed to improve the security of DevOps. To manage cybersecurity risk from third-party software components, "identifying, assessing, selecting, and implementing processes and mitigating controls [45]" should be involved. This risk management can be integrated into DevSecOps through automation.

The secure software development practice should be implemented throughout SDLC for the following reasons:

- "To reduce the number of vulnerabilities in released software [45]."
- "To reduce the potential impact of the exploitation of undetected or unaddressed vulnerabilities by both external attacks and insider threat [45]."
- "To address the root causes of vulnerabilities to prevent recurrences [45]."

Security Control Map:

In cited document 45, Table 1 on page 9-16 maps the characteristics of the closed source and open-source products to applicable standards and recommended practices in the forms of SSDF practices and tasks associated to improve the Critical Infrastructure Cybersecurity [45].

This table contains important practices described in detail along with SSDF tasks to provide guidelines and ideas to support the practices.

3. FINAL RECOMMENDATIONS

3.1. Picking the Right Infrastructure

Infrastructure is a critical tool that should be thought about when trying to implement DevOps and CI/CD pipelines. Azure or Gitlab would be a great choice when trying to implement a DevOps cycle and pipeline. It is important to note that every pipeline and cycle will be very dependent on the goal and expected outcome. The three main pipelines that are implemented at Sandia National Labs are Jenkins, GitLab, and Azure. Most teams at Sandia use Gitlab because of the premium Gitlab license and the instances that are used at Sandia. Azure has public cloud DevOps which can be rather limiting because the Sandia tenet and the third-party extensions that will have to get approved. Azure stack is also available on-premise which will allow for source code management, issue tracking, agile tracking and code reviewing.

Integrating both Azure Stack and Gitlab would be the best way to develop a CI/CD pipeline and DevOps cycle. To implement this idea, Gitlab and Azure accounts are needed. With the help of GitLab's extensive DevOps platform and integrated CI/CD system, your team can manage code more effectively, automate testing, and release new features and updates more quickly. From a single platform, it enables developers to write, test, deploy code, store scripts, and store client data. Additionally, GitLab offers a common platform for all participants, fostering teamwork and transparency—two fundamental tenets of the DevOps mindset. On the other hand, Microsoft offers Azure, a highly scalable, dependable, and accessible cloud platform. Azure integration with your DevOps and CI/CD pipeline enables you to manage and deploy your applications quickly in a stable cloud environment. Azure is adaptable for various project requirements since it supports a range of programming languages, tools, and frameworks, including both Microsoft-specific and third-party applications.

GitLab and Azure stacks work together to streamline every step of the software delivery process. The Azure stack offers a versatile and scalable platform for deploying and administering the application, and GitLab enables effective code integration and testing. This combination speeds up the process of delivering applications to end customers while also increasing the productivity of your team. Your team may concentrate more on development work because you don't have to operate your own servers when you use Azure's cloud platform, which further lowers overhead.

3.2. Implementing Neoload Web with Azure and GitLab

Implementing NeoLoad Web with Azure and GitLab in a CI/CD pipeline can be a crucial part of setting up an efficient DevOps cycle for load and stress testing. The process begins with defining the application's codebase in GitLab. The codebase in GitLab is then linked to Azure, or another cloud service platform.

The CI phase is initiated every time a new code is committed and pushed to the GitLab repository. Azure Pipelines – an Azure DevOps service – can be utilized for this. A build process is initiated which could include compilation, unit testing, and code quality checks. The results of these processes are tracked and can serve as a gateway to whether the changes can be merged into the

main branch or not. For the CD phase, the pipeline can be configured to automatically deploy the application to the appropriate environment in Azure. This could be a testing environment or a production environment, depending on the branching strategy and pipeline configuration.

The starting point for doing load and stress testing at various levels of the pipeline is the integration of NeoLoad Web. During load-testing, NeoLoad Web APIs enable the extraction of important performance indicators. To simulate how the application would be used in the real world, these performance tests could be carried out during the build process, before deployment to an environment, or even after deployment. The teams can track performance trends over time by sending test results back to GitLab and Azure. NeoLoad Web dashboards can also be utilized for in-the-moment monitoring and for finer-grained data analysis. The pipeline can be set up to stop further distribution and alert the team if the performance test fails.

GitLab, Azure, and NeoLoad Web can be integrated to provide a solid and efficient DevOps cycle. Organizations may assure quicker, more reliable releases while maintaining a high level of performance and code quality by putting up this CI/CD pipeline. It is important to note that the EMTS team will still have to write scripts for clients but other processes like emails, removing information, scheduling, and running scripts can be automated.

3.3. Integrating Azure and Gitlab with Burp Suite

Burp Suite integration into an Azure and GitLab CI/CD pipeline adds a useful tool for automating security testing as part of the DevOps cycle. Establishing the codebase within GitLab is the initial step. This platform makes it simple to collaborate, version code, and initiate CI every time new code is committed and published to the GitLab repository. This stage can be handled by Azure Pipelines, a component of Azure DevOps services. The build phase, which could involve operations like compilation, unit testing, and code quality checks, starts after the CI phase is launched. Once these procedures are complete, the results are watched to see if the changes can be merged into the main branch. For CD, the pipeline is set up to deploy the application automatically to the appropriate environment in Azure, based on the pipeline configuration and branching strategy. It could be a testing or production environment.

For automated security testing, Burp Suite can now be added to the pipeline. Burp Scanner, a scanning component of Burp Suite, can automatically explore and examine web applications for a variety of security flaws. Automated security scans can be performed before deployment to an environment or during the build process by integrating Burp Suite into the CI/CD pipeline. These scans' outcomes can be communicated to GitLab and Azure, informing those teams of any security vulnerabilities. The pipeline can be set up to stop deployment in the case that the security scan fails and alert the team. This makes it possible to quickly mitigate security issues before they become a production issue. Additionally, by integrating Burp Suite in this way, it is possible to track changing security trends and continuously enhance application security posture.

When integrating Burp Suite with a CI/CD pipeline it is important to note that the recording of the testing script cannot be automated unless clients submit their own recordings. It is not advisable to

have clients do their own recordings. However, scheduling, emails, error notification, and variabilities reports can be automated but should get human approval before being sent out. Burp Suite should be used for major web application changes that involve adding new features or changing old features.

3.4. Deploying Zap for Automated Security Testing

Deploying ZAP with a CI/CD pipeline would be beneficial for automated security testing. Teams could fully integrate Zap into a security testing pipeline. This will allow the team to create a One Service page or other webpage that can handle CUI information that will be sent to a pipeline to begin testing. The major application of this pipeline would be when a customer makes a minor change that needs to be tested. After the test has finished the security testing team can look at the generated report and then determine if there are any false positives.

By automating the process of identifying vulnerabilities in web applications, the integration of OWASP ZAP into a CI/CD pipeline with Azure and Gitlab can greatly improve the security of software development projects. For businesses aiming to identify and reduce security threats before they enter the production environment and so strengthen overall security posture, this procedure is crucial. OWASP ZAP is available as a Docker image. Setting up an OWASP ZAP Docker image is the first step in incorporating ZAP into a CI/CD process. Docker is a containerization platform that can package an application and its dependencies into a standardized unit for software development. The Docker image can be adjusted and customized in accordance with the needs of a particular web application that it needs to scan after being downloaded from the Docker Hub.

Next, Gitlab CI/CD is configured to automate the launching of the ZAP Docker image as part of the pipeline. The GitLab CI configuration file (.gitlab-ci.yml) can be modified to add a new stage to accomplish this. The steps in this stage should comprise the necessary commands to launch the ZAP Docker container, carry out the necessary scan, and gather the results. A push artifact can then be created with the results for further study. On the other side, Azure can host the application that has to be scanned and offer the resources required to carry out ZAP scans. Azure DevOps can make it easier to incorporate ZAP into the CI/CD workflow. Docker has built-in tasks available through Azure DevOps that may be used to execute ZAP scans as part of the pipelines for building and releasing software. Additionally, GitLab's CI/CD pipeline can be set up to be activated by Azure DevOps anytime a change is pushed to the repository, ensuring that security checks are performed continuously throughout the development process. The scan reports can be read, and relevant action can be taken in response to the results when the pipeline has finished, and they have been generated.

Zap can be used in a pipeline when a client needs to have minor changes looked at for variabilities. Because Zap can be fully automated into a pipeline it has the opportunity to take some of the load off of the EMTS team. This would allow the team to look at a report generated by Zap and verify with the client if there are any vulnerabilities allowing the team to focus on major testing.

3.5. Standardizations for Security Testing

3.5.1. Follow OWASP Standards using Threat Modeling and Risk Assessments

The OWASP ASVS [50] is a comprehensive list of requirements and tests for software developers and security professionals to build, test and verify secure web applications.

Organizations are strongly encouraged to look deeply at their unique risk characteristics based on the nature of their business, and based upon that risk and business requirements, determine the appropriate ASVS level. This level will determine the tradeoff between security and resources that is appropriate for your application. Conducting risk assessments and creating threat models can help you determine an appropriate ASVS level. Threat modeling will allow your organization to identify vulnerabilities for specific applications through penetration testing or application security testing scenarios. Some security testing tools have already been discussed in previous sections and will be discussed later in this section. Risk assessments can then be used to determine the likelihood that certain vulnerabilities will be exploited by attackers and their most likely points of attack. Both threat modeling and risk assessment will allow you to determine how to best allocate resources when it comes to security testing. Additionally, you will want to take into consideration what the application does, what information it houses, and how that information could be used by adversaries. Looking into these questions can help you understand the possible motivations of attackers and their corresponding threats. The applications corresponding ASVS level should be apparent after conducting a risk assessment and determining adversarial threats. Each security requirement within the OWASP ASVS document has one or more associated ASVS levels. This allows you to customize the standards report to better find the needs of your organization or application.

3.5.2. Integrate Early and Often Security Testing

An integral part of creating a DevSecOps pipeline is the concept of shift-left security. This means following secure coding practices throughout the entire software development life cycle and conducting security testing early and often. Automation should be used to conduct security tests as code within applications is being pushed to the central repository.

3.5.3. Increase Collaboration Between Software Developers and Security Professionals

Having software developers and security professionals work together when developing software not only saves time and resources in the future, by preventing the need for entire applications to be reworked because they are insecure, but it also helps promote the central idea of DevSecOps which is integrating security early in the software development lifecycle. Additionally, increasing collaboration between these groups will encourage the use of more secure coding practices and security testing. Collaboration between groups also applies to groups within the software development team. Increased collaboration between all the groups creates an environment where there is at least one person who has full visibility of the project and can keep tasks moving forward. The idea of close collaboration should also prevent groups from experiencing deadlock with other groups and decrease frustration when it comes to teamwork.

3.5.4. Use the Microsoft Security Code Analysis Toolset

The MSCA toolset is an extension for the Azure DevOps pipeline service. It is a collection of tools and tasks that contribute to the secure development of software. Static and dynamic testing tools are included within MSCA. Additionally, analysis and result logging tasks are already built in. This tool is aimed at implementing the SDL into software development. Integration with Azure DevOps makes it easy to automate tests and therefore catch and remediate security issues early and often within the development lifecycle. MSCA also provides scanning tools to help ensure that the pipeline and DevOps team are following best practices. Risks can be reduced because MSCA can be used to address issues with the code by notifying developers and blocking Pull Requests when issues are found.

3.6. Standardizations for Pipelines

3.6.1. Implementation of Immutable Infrastructure

When using CI/CD pipelines, immutable infrastructure must be implemented. This concept asserts that once a service is deployed, no alterations should occur to the live infrastructure. Rather than applying updates or patches to the existing infrastructure, a new infrastructure should be built from a standard image, containing all the required updates. This approach helps reduce inconsistencies in the environments, enhances the repeatability of the deployments, and adds to the system's resilience. In case of a failure, a new infrastructure can be spun up quickly from the standard image.

3.6.2. Compliance with Standard Practices and Procedures

Pipelines should strictly adhere to established standards. This includes following a standardized directory structure for projects, adhering to coding standards, implementing code reviews, and using version control. These practices help ensure high code quality, ease of maintenance, and effective collaboration. In addition, version control systems provide a reliable mechanism for backing up and restoring the codebase.

3.6.3. Rigorous Testing and Monitoring

Any changes to code or configuration should be accompanied by comprehensive testing and continuous monitoring. This would include unit testing, integration testing, performance testing, security testing, and any other testing relevant to the specific application. Monitoring should be performed not just at the infrastructure level but also at the application level to ensure that any issues are detected and addressed promptly. Testing and tracking critical for ensuring that the delivered software meets the required quality standards and performs as expected in the production environment.

3.6.4. Incorporation of Security Practices

The practice of shift-left security, where security considerations are moved earlier in the development cycle, should be adopted by developers. This entails integrating security practices into every CI/CD pipeline phase. Static code analysis should be performed to detect any potential security vulnerabilities in the code. Dynamic analysis should be conducted to detect runtime vulnerabilities. Security checks should also be incorporated into the pipeline to ensure compliance with security standards. In addition, appropriate measures should be taken to protect sensitive information in the development, testing, and production environments.

3.6.5. Development of a Disaster Recovery Plan

Pipeline designers should ensure that a comprehensive disaster recovery plan is in place. The CI/CD pipeline should be designed to enable quick and reliable recovery in case of any disasters. This includes ensuring that all critical data is backed up regularly so it can be restored quickly if there is a major issue. The recovery procedures should be documented and regularly updated. Regular drills should be conducted to ensure that the recovery procedures work as expected and that all team members are familiar with them. The plan should also include provisions for communication and coordination during a disaster.

3.7. Standardizations for DevSecOps

3.7.1. Adoption of Infrastructure as Code

When creating a DevOps cycle, developers should embrace the practice of Infrastructure as Code to manage and provision their technological frameworks. By utilizing tools such as Terraform, Ansible, or Chef, they can easily create, change, and improve their infrastructure with version control, automation, and consistency. Infrastructure becomes reproducible and scalable, mitigating the risk of human error associated with manual processes. Moreover, IaC can review and test changes in a staging environment before implementing them into production, promoting a more robust and resilient infrastructure.

3.7.2. Frequent Iterative Updates Over Massive Overhauls

Developers must adopt a CI/CD approach to application development and updates. This allows for small, regular application updates instead of massive, infrequent overhauls. This approach reduces the risk associated with changes, as smaller updates are easier to test and validate. It also allows for quicker responses to security vulnerabilities, as patches can be developed, tested, and deployed rapidly. Tools like Jenkins, GitLab, or Azure can aid in implementing a CI/CD pipeline.

3.7.3. Adopt Monitoring and Logging Practices

Effective monitoring and logging are essential for maintaining system health and troubleshooting issues. Implementing a comprehensive monitoring solution provides visibility into the performance of applications, servers, and networks. It should cover key performance indicators, error rates, and user experience metrics. In conjunction with logging, which records events and transactions within the system, the operations team can quickly identify and resolve problems, reducing downtime and enhancing the reliability of services.

3.7.4. Security-First Approach

Sensitive and crucial data will be available to developers when creating a DevOps cycle, so developers must adopt a security-first approach. This involves security practices being baked into the development lifecycle, not just addressed at the final stage or as an afterthought. Techniques such as automated security scans, regular code reviews, secure coding practices, and employing a least privilege model for access controls should be commonplace. Technologies like Security as Code and tools such as SonarQube can help automate security checks within the development process.

3.7.5. Promotion of a Collaborative Culture

A collaborative culture should be fostered within developers. Communication, knowledge-sharing, and cross-skilling are all vital components of an effective DevOps strategy. An environment where developers and operations teams understand each other's challenges and work together to resolve them encourages faster problem-solving, innovation, and a more efficient working process.

4. REFERENCES

[1]

"Burp Suite Enterprise Edition vs. Burp Suite Professional," *portswigger.net*. https://portswigger.net/burp/enterprise/resources/enterprise-edition-vs-professional

[2]

monya, "Burp Pro vs Burp Enterprise - Burp Suite User Forum," *forum.portswigger.net*. https://forum.portswigger.net/thread/burp-pro-vs-burp-enterprise-ada7e6db (accessed Jul. 06, 2023).

[3]

E-SPIN, "Burp Suite Pro vs Enterprise what the differences," *E-SPIN Group*, Nov. 19, 2021. https://www.e-spincorp.com/burp-suite-pro-vs-enterprise-what-the-differences/

[4]

iCorps Technologies, "The Pros and Cons of Microsoft Azure: Cloud Services for Businesses," *blog.icorps.com*, Jan. 16, 2022. https://blog.icorps.com/pros-and-cons-microsoft-azure#:~:text=Here%20are%20the%20Pros%20and%20Cons%20of%20Microsoft

[5]

M. Shivanandhan, "How to Use Burp Suite to Audit Web Applications – Pentesting and Bug Bounty Tool Overview," *freeCodeCamp.org*, Jan. 17, 2023. https://www.freecodecamp.org/news/how-to-audit-web-apps-with-burpsuite/ (accessed Jul. 06, 2023).

[6]

GeeksforGeeks, "What is Burp Suite?," *GeeksforGeeks*, Aug. 22, 2019. https://www.geeksforgeeks.org/what-is-burp-suite/

[7]

S. Shukla, "Understanding Burp Suite Intruder Attack Types," *www.linkedin.com*, Oct. 04, 2017. https://www.linkedin.com/pulse/basic-tutorial-security-testing-using-burp-force-qa-engineer- (accessed Jul. 06, 2023).

[8]

"Configuring Burp Intruder attacks," *portswigger.net*, Jul. 06, 2023. https://portswigger.net/burp/documentation/desktop/tools/intruder/configure-attack

[9]

"Burp Intruder attack types," *portswigger.net*, Jul. 06, 2023. https://portswigger.net/burp/documentation/desktop/tools/intruder/configure-attack/attack-types

[10]

"burpsuite | Kali Linux Tools," *Kali Linux*. https://www.kali.org/tools/burpsuite/

[11]

M. Atkinson, "Some of the best Burp extensions - as chosen by you," *PortSwigger Blog*, May 27, 2021. https://portswigger.net/blog/some-of-the-best-burp-extensions-as-chosen-by-you (accessed Jul. 06, 2023).

[12]

F. Cheng, "Burp Suite," *LO4D.com*. https://burp-suite.en.lo4d.com/windows (accessed Jul. 06, 2023).

[13]

"BApp Store," *portswigger.net*. https://portswigger.net/bappstore

[14]

A. Entry, "How to use Autorize," *Medium*, Jan. 12, 2021. https://authorizedentry.medium.com/how-to-use-autorize-fcd099366239 (accessed Jul. 06, 2023).

[15]

james kettle, "Turbo Intruder: Embracing the billion-request attack," *PortSwigger Research*, Jan. 25, 2019. https://portswigger.net/research/turbo-intruder-embracing-the-billion-request-attack (accessed Jul. 06, 2023).

[16]

"Turbo Intruder," *GitHub*, Jul. 06, 2023. https://github.com/PortSwigger/turbo-intruder (accessed Jul. 06, 2023).

[17]

G. Heyes, "Hackvertor," *portswigger.net*. https://portswigger.net/bappstore/65033cbd2c344fbabe57ac060b5dd100 (accessed Jul. 06, 2023).

[18]

"Burp Bounty – Website vulnerability scanner," *Burp Bounty*. https://burpbounty.net/ (accessed Jul. 06, 2023).

[19]

"param-miner," *GitHub*, Mar. 31, 2023. https://github.com/intruder-io/param-miner (accessed Jul. 06, 2023).

[20]

"Best Practices for Successful CI/CD | TeamCity CI/CD Guide," *JetBrains*. https://www.jetbrains.com/teamcity/ci-cd-guide/ci-cd-best-practices/ (accessed Jul. 06, 2023)

[21]

mijacobs, "Azure Pipelines baseline architecture - Azure Pipelines," *learn.microsoft.com*, May 08, 2023. https://learn.microsoft.com/en-us/azure/devops/pipelines/architectures/devops-pipelines-baseline-architecture?view=azure-devops (accessed Jul. 06, 2023).

[22]
mijacobs, "Azure Pipelines baseline architecture - Azure Pipelines," *learn.microsoft.com*, May 08, 2023. https://learn.microsoft.com/en-us/azure/devops/pipelines/architectures/devops-pipelines-baseline-architecture?view=azure-devops#considerations (accessed Jul. 06, 2023).

[23]
"Microsoft Security DevOps," *Microsoft.com*, 2019. https://www.microsoft.com/en-us/securityengineering/devsecops (accessed Jul. 06, 2023).

[24]
mijacobs, "What is infrastructure as code (IaC)? - Azure DevOps," *learn.microsoft.com*. https://learn.microsoft.com/en-us/devops/deliver/what-is-infrastructure-as-code

[25]
https://plus.google.com/100518058839384323139, "What is Invicti?," *www.invicti.com*, Mar. 09, 2022. https://www.invicti.com/support/what-is-invicti/

[26]
steved0x, "Azure Pipelines Agents - Azure Pipelines," *learn.microsoft.com*, Jun. 20, 2023. https://learn.microsoft.com/en-us/azure/devops/pipelines/agents/agents?view=azure-devops&tabs=browser#install (accessed Jul. 11, 2023).
[27]

baldwin, "Azure Security Benchmark v3 - DevOps Security," *learn.microsoft.com*, Nov. 14, 2022. https://learn.microsoft.com/en-us/security/benchmark/azure/security-controls-v3-devops-security (accessed Jul. 14, 2023).

[28]
B. Reselman, "A developer's guide to CI/CD and GitOps with Jenkins Pipelines," *Red Hat Developer*, Jan. 13, 2022. https://developers.redhat.com/articles/2022/01/13/developers-guide-cicd-and-gitops-jenkins-pipelines#how_jenkins_supports_ci_cd (accessed Jul. 17, 2023).

[29]
"What Is Jenkins in CI/CD - Everything You Need To Know," *Knowledge Base by phoenixNAP*, Jan. 20, 2022. https://phoenixnap.com/kb/what-is-jenkins (accessed Jul. 17, 2023).

[30]
JenkinsCI, "Pipeline," *Pipeline*. https://www.jenkins.io/doc/book/pipeline/

[31]
U. Unclassified, "DoD Enterprise DevSecOps Strategy Guide CLEARED For Open Publication Department of Defense OFFICE OF PREPUBLICATION AND SECURITY REVIEW," Sep. 2021. Accessed: Jul. 18, 2023. [Online]. Available: https://dodcio.defense.gov/Portals/0/Documents/Library/DoD%20Enterprise%20DevSecOps%20Strategy%20Guide_DoD-CIO_20211019.pdf

[31]
G. Lab, "CI/CD pipelines | GitLab," *docs.gitlab.com*. https://docs.gitlab.com/ee/ci/pipelines/

[32]
Neotys, "Deploying NeoLoad Web," *Neotys Connect*. https://connect.neotys.com/tutorials/deploying-neoload-web (accessed Jul. 18, 2023).

[33]
MongoDB, "MongoDB Atlas Database | Multi-Cloud Database Service," *MongoDB*, Oct. 19, 2021. https://www.mongodb.com/atlas/database

[34]
paulsbruce, "neoload_kube/aws/nlw_deploy_eks.md at master · Neotys-Connect/neoload_kube," *GitHub*, Dec. 01, 2020. https://github.com/Neotys-Connect/neoload_kube/blob/master/aws/nlw_deploy_eks.md (accessed Jul. 18, 2023).

[35]
paulsbruce , "neoload_kube/aws/dynamic_infra.md at master · Neotys-Connect/neoload_kube," *GitHub*, Dec. 01, 2020. https://github.com/Neotys-Connect/neoload_kube/blob/master/aws/dynamic_infra.md (accessed Jul. 18, 2023).

[36]
neotysbde, "neoload-web-test-launcher-docker/AzureDevops-usage.md at master · Neotys-Labs/neoload-web-test-launcher-docker," *GitHub*, Nov. 18, 2019. https://github.com/Neotys-

Labs/neoload-web-test-launcher-docker/blob/master/AzureDevops-usage.md (accessed Jul. 18, 2023).

[37]
FROD-Neotys , "NeoLoad Web OpenShift template," *GitHub*, Jan. 28, 2023. https://github.com/Neotys-Labs/neoload-web-openshift-template (accessed Jul. 18, 2023).

[38]
P. Kirvan, "What is Red Hat OpenShift? | Definition from TechTarget," *Cloud Computing*. https://www.techtarget.com/searchcloudcomputing/definition/Red-Hat-OpenShift (accessed Jul. 18, 2023).

[39]
A. Gillis, "What is MongoDB? A definition from WhatIs.com," *SearchDataManagement*. https://www.techtarget.com/searchdatamanagement/definition/MongoDB

[40]
Lambdatest, "Stress Testing Tutorial: Comprehensive Guide With Best Practices," *www.lambdatest.com*. https://www.lambdatest.com/learning-hub/stress-testing#best-practices (accessed Jul. 19, 2023).

[42]
atatus, "What is Load Testing? Processes, Types, Best Practices, Tools, and More," *Atatus Blog - For DevOps Engineers, Web App Developers and Server Admins.*, Aug. 25, 2021. https://www.atatus.com/blog/what-is-load-testing/#best-practices-for-lt (accessed Jul. 19, 2023).

[43]
Dr. C. Thun, S. Prioux, and M. Cañamero, "Stress Testing Best Practices: A Seven Steps Model | Moody's Analytics," *www.moodysanalytics.com*, Sep. 2013. https://www.moodysanalytics.com/risk-perspectives-magazine/stress-testing-europe/approaches-to-implementation/stress-testing-best-practices-a-seven-steps-model

[44]
Joe, "9 Load Testing Best Practices (Don't Make These Mistakes)," *Automation Testing Made Easy Tools Tips & Training*, Mar. 17, 2020. https://testguild.com/best-load-testing/ (accessed Jul. 19, 2023).

[45]
M. Souppaya, M. Ogata, P. Watrobski, and K. Scarfone, "Software Supply Chain and DevOps Security Practices: Implementing a Risk-Based Approach to DevSecOps," *csrc.nist.gov*, Nov. 2022, Accessed: Jul. 20, 2023. [Online]. Available: https://csrc.nist.gov/pubs/pd/2022/11/09/implementing-a-riskbased-approach-to-devsecops/final#:~:text=To%20help%20improve%20the%20security%20of%20DevOps%20practices%2C

[45]

G2, "Best Load Testing Tools," *G2*, May 2023. https://www.g2.com/categories/load-testing-tools (accessed Jul. 20, 2023).

[46]
Apache Software Foundation, "Apache JMeter - Apache JMeter™," *Apache.org*, 2019. https://jmeter.apache.org/

[47]
Microsoft, "Microsoft Azure Marketplace," *azuremarketplace.microsoft.com*. https://azuremarketplace.microsoft.com/en-us/marketplace/apps/micro-focus.ms-loadrunner?tab=overview (accessed Jul. 20, 2023).

[48]
A. Murray, "Vulnerability Management - Everything You Need To Know," Mend, Dec. 2, 2021. https://www.mend.io/blog/vulnerability-management/ (accessed Jul. 18, 2023).

[49]
A. Murray, "All About Application Security: Tools, Types, Trends in 2023," Mend, Dec. 29, 2022. https://www.mend.io/blog/application-security/ (accessed Jul. 18, 2023).

[50]
"Application Security Verification Standard 4.0 Final," 2019. Available: https://owasp.org/www-pdf-archive/OWASP_Application_Security_Verification_Standard_4.0-en.pdf

[51]
"Azure security test practices - Microsoft Azure Well-Architected Framework," learn.microsoft.com, Mar. 24, 2023. https://learn.microsoft.com/en-us/azure/well-architected/security/monitor-test (accessed Jul. 20, 2023).

[52]
"Burp Suite for Beginners Part 1: Setup and Target/Proxy Tools," jaimelightfoot.com, Jan. 12, 2019. https://jaimelightfoot.com/blog/burp-suite-for-beginners-setup-and-target-proxy-tools/ (accessed Jul. 18, 2023).

[53]
"Center for Internet Security (CIS) Benchmarks - Microsoft Compliance," learn.microsoft.com. June 5, 2021. https://learn.microsoft.com/en-us/compliance/regulatory/offering-cis-benchmark (accessed Jul. 18, 2023).

[54]
"Cyber security assessments," RedSquall, https://www.redsquall.com/ (accessed Jul. 18, 2023).

[55]
D. Support, "Microsoft Security Code Analysis – a tool that seamlessly empowers customers to enable security controls in your CI/CD pipeline," Developer Support, Dec. 02, 2019. https://devblogs.microsoft.com/premier-developer/microsoft-security-code-analysis/ (accessed Jul. 20, 2023).

[56]
"Home," GitHub. https://github.com/Microsoft/AttackSurfaceAnalyzer/wiki/ (accessed Jul. 20, 2023).

[57]
"Microsoft Security Assessment." Accessed: Jul. 20, 2023. [Online]. Available: https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RWOdEk#:~:text=Microsoft%20Security-Microsoft.com, 2023. https://download.microsoft.com/documents/uk/technet/downloads/technetmagazine/utilityukdesfin.pdf (accessed Jul. 20, 2023).

[58]
N. G. Cuthbert Daniel, "Practical tips on how to use application security testing and testing standards," Microsoft Security Blog, Oct. 05, 2021. https://www.microsoft.com/en-us/security/blog/2021/10/05/practical-tips-on-how-to-use-application-security-testing-and-testing-standards/ (accessed Jul. 20, 2023).

[59]
O. Moradov, "Security testing: Types, tools, and best practices," Bright Security, May 29, 2022. https://brightsec.com/blog/security-testing/ (accessed Jul. 18, 2023).

[60]
O. Moradov, "What Is OWASP and What Are OWASP Top 10 for Web/API/Mobile?," Bright Security, Jul. 20, 2022. https://brightsec.com/blog/owasp/ (accessed Jul. 18, 2023).

[61]
OWASP, "OWASP Application Security Verification Standard," owasp.org. https://owasp.org/www-project-application-security-verification-standard/
"System requirements for standard deployments," portswigger.net. July 3, 2023. https://portswigger.net/burp/documentation/enterprise/getting-started/system-requirements/standard-sys-req (accessed Jul. 18, 2023).

[62]
"Vulnerabilities detected by Burp Scanner," portswigger.net. June 8, 2023. https://portswigger.net/burp/documentation/scanner/vulnerabilities-list (accessed Jul. 18, 2023).

[63]
"Configuring your environment network and firewall settings," PortSwigger, Nov. 11, 2022. Accessed: Nov. 20, 2022. [Online]. Available: https://portswigger.net/burp/documentation/enterprise/getting-started/network-firewall-config

[64]
zaproxy, "The ZAP Homepage," *Zaproxy.org*, 2022. https://www.zaproxy.org (accessed Jul. 20, 2023).

[65]
D. Lukanov, "How to integrate OWASP ZAP in Gitlab CI/CD pipeline.," *Codific*, Dec. 29, 2022. https://codific.com/integrate-owasp-zap-and-gitlab/ (accessed Jul. 20, 2023).

[66]
G. Hegde , "Configure OWASP ZAP Security Tests in Azure DevOps - DZone," *dzone.com*, Jul. 10, 2019. https://dzone.com/articles/owasp-zap-security-tests-in-azure-devops-pipeline (accessed Jul. 20, 2023).

[67]
I. T. L. Computer Security Division, "DevSecOps | CSRC | CSRC," *CSRC | NIST*, Oct. 21, 2020. https://csrc.nist.gov/Projects/devsecops

[68]
"NIST's recommendations for secure DevSecOps," *GitGuardian Blog - Automated Secrets Detection*, Aug. 04, 2021. https://blog.gitguardian.com/nist-recommendations-for-secure-devsecops/

[69]
"Microsoft Azure Marketplace," *azuremarketplace.microsoft.com*. https://azuremarketplace.microsoft.com/en-us/marketplace/apps/smartbearsoftware.readyapi_testengine?tab=overview (accessed Jul. 24, 2023).

[70]
"Gatling," *GitHub*, May 28, 2023. https://github.com/gatling/gatling

[71]

Microsoft, "Microsoft Security Development Lifecycle Practices," *Microsoft.com*, 2019. https://www.microsoft.com/en-us/securityengineering/sdl/practices

[72]
"Azure Boards | Microsoft Azure," *azure.microsoft.com*. https://azure.microsoft.com/en-us/products/devops/boards/

[73]
"Appendix N: SDL Security Bug Bar (Sample)," *learn.microsoft.com*. https://learn.microsoft.com/en-us/previous-versions/windows/desktop/cc307404(v=msdn.10)?redirectedfrom=MSDN

[74]
"OSS Secure Supply Chain Framework," *www.microsoft.com*. https://www.microsoft.com/en-us/securityengineering/opensource

[75]
"Microsoft SDL Cryptographic Recommendations," *Microsoft.com*, 2022. http://download.microsoft.com/download/6/3/A/63AFA3DF-BB84-4B38-8704-B27605B99DA7/Microsoft%20SDL%20Cryptographic%20Recommendations.pdf

[76]

Microsoft, "Microsoft Security Development Lifecycle Threat Modelling," *Microsoft.com*, 2018. https://www.microsoft.com/en-us/securityengineering/sdl/threatmodeling

[77]
TerryLanfear, "Microsoft Security Code Analysis documentation overview," *learn.microsoft.com*, Jan. 09, 2023. https://learn.microsoft.com/en-us/previous-versions/azure/security/develop/security-code-analysis-overview (accessed Jul. 28, 2023).

[78]
TerryLanfear, "Azure threat protection," *learn.microsoft.com*. https://learn.microsoft.com/en-us/azure/security/fundamentals/threat-detection

[79]
"Managing Security Risks Inherent in the Use of Third- party Components." Available: https://safecode.org/wp-content/uploads/2017/05/SAFECode_TPC_Whitepaper.pdf

[80]
"Microsoft Security Development Lifecycle Resources," *www.microsoft.com*. https://www.microsoft.com/en-us/securityengineering/sdl/resources

[81]
"Attack Surface Analyzer," *GitHub*, Oct. 07, 2022. https://github.com/Microsoft/AttackSurfaceAnalyzer

[82]
dcurwin, "Alert response tutorial - Microsoft Defender for Cloud," *learn.microsoft.com*, Jun. 29, 2023. https://learn.microsoft.com/en-us/azure/defender-for-cloud/tutorial-security-incident (accessed Jul. 31, 2023).

DISTRIBUTION

Email—Internal

Name	Org.	Sandia Email Address
Tanya DeLara	09744	tdelara@sandia.gov
Andrew Brungard	09762	tabrung@sandia.gov
Zane Parker	09744	zparker@sandia.gov
Technical Library	1911	sanddocs@sandia.gov

Email—External

Name	Company Email Address	Company Name
Dominic D'Onofrio	Dominic.donofrio@student.nmt.edu	New Mexico Tech
Lorie Liebrock	Loire.liebrock@nmt.edu	NMCCoE

This page left blank

Sandia National Laboratories

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia LLC, a wholly owned subsidiary of Honeywell International Inc. for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.