



# DEVSECOPS: AUTOMATING SECURITY

## What is DevSecOps?

DevSecOps is a methodology in which security is integrated into the entire life cycle of application development. Rather than being an afterthought, security present throughout development. This benefits NASA and the NCCS by decreasing the instance of security issues in later stages of the development process.

## What is CI/CD?

CI – Continuous Integration: Rapid commitments of created code to repositories. Keeps developers up to date and allows for bugs/errors to be detected faster.

CD – Continuous Delivery: Focusing on frequent and automated deployments with confidence in code quality and security.

## What is a Pipeline?

A CI/CD “Pipeline” is a series of automated checks that are performed on code before deployment. As the code moves through the pipeline, a series of progressive tests/events are executed on the code.

Why is this useful? Manual code review can take days, weeks, or months depending on the size of the application. Use of tools to check code and security can take hours or days, and the process is still manual. A pipeline can complete these tasks without human intervention, and stores the results in a centralized location.

## Terminology

Container – A small, barebones virtual computer. Useful for running and testing apps without the need for a virtual machine, which requires large amounts of space and processing power

Docker – An open source tool which creates and runs containers

Enumeration – The process of gathering information about a system. This is typically performed early in an evaluation process in order to determine basic information about a system and its services

Static Analysis – Using tools to perform tests on source code while not running. Tests for potential injections, backdoors, code flaws

Dynamic Analysis – Using tools to perform tests on a program while “live” in order to determine vulnerabilities, observe behavior, test error handling, etc.

## Threat Modeling

Each pipeline includes a questionnaire. Answers to certain questions/combinations of questions will trigger pipeline actions based on the input using python to interact with the Gitlab API. For example, “Does this project interactive with SQL databases?”, if yes, will cause SQL scans to be performed. This increases the degree of precision in pipeline scans.

## Expansion

There are an infinite number of ways that this project could expand. Pipelines can be adapted to cater to the testing needed for various applications, systems, and tools.

Each of the pipelines developed in this project are well documented and are designed to be easily transferrable to other projects with minimal modification to the code.

## Conclusion & Results

The primary purpose of a pipeline is to save time by automating tasks that would otherwise be done manually. To gauge the effectiveness of the pipeline’s automation, the time needed to complete the exact same evaluation of a system/container is measured for both a pipeline, and manual execution of the same tools implemented by the pipeline.

Times for each use case were calculated from the average of 3 successful runs on the same use case.

For each use case, the automated pipeline saw significantly faster and more consistent results when compared to manual execution.

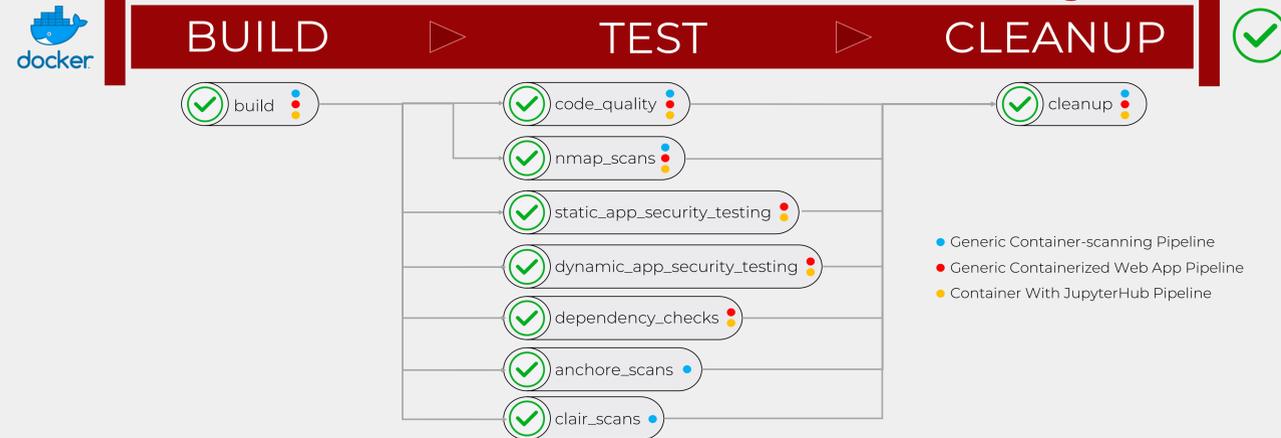
In addition to faster times, each pipeline produces a formatted report after execution. This is useful for creating issues in a tracker, as well as providing guidance for code changes.

## Acknowledgements

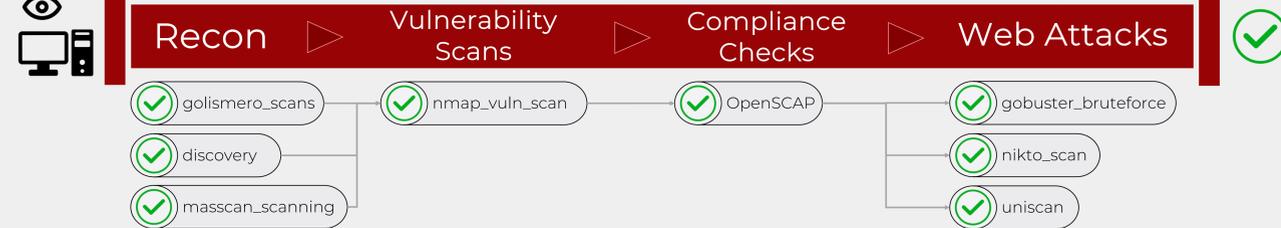
I would like to sincerely thank George Rumney, John Jasen, Jordan A. Caraballo-Vega, Jasaan Neff, and the rest of the NCCS SWG team for their guidance and expertise.

## Pipelines

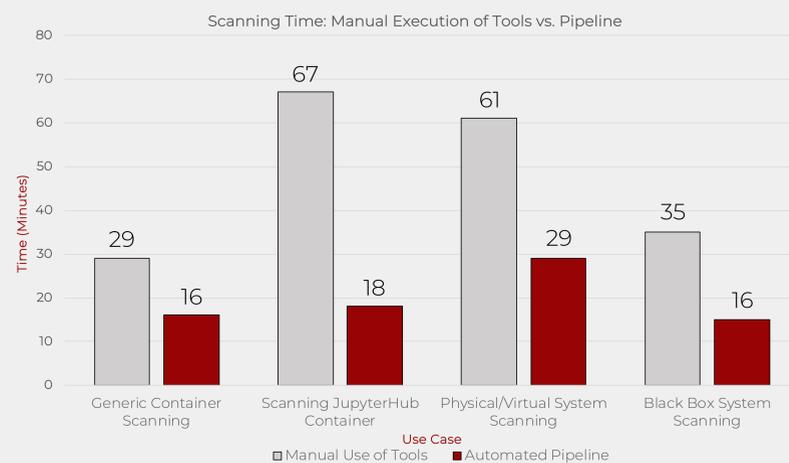
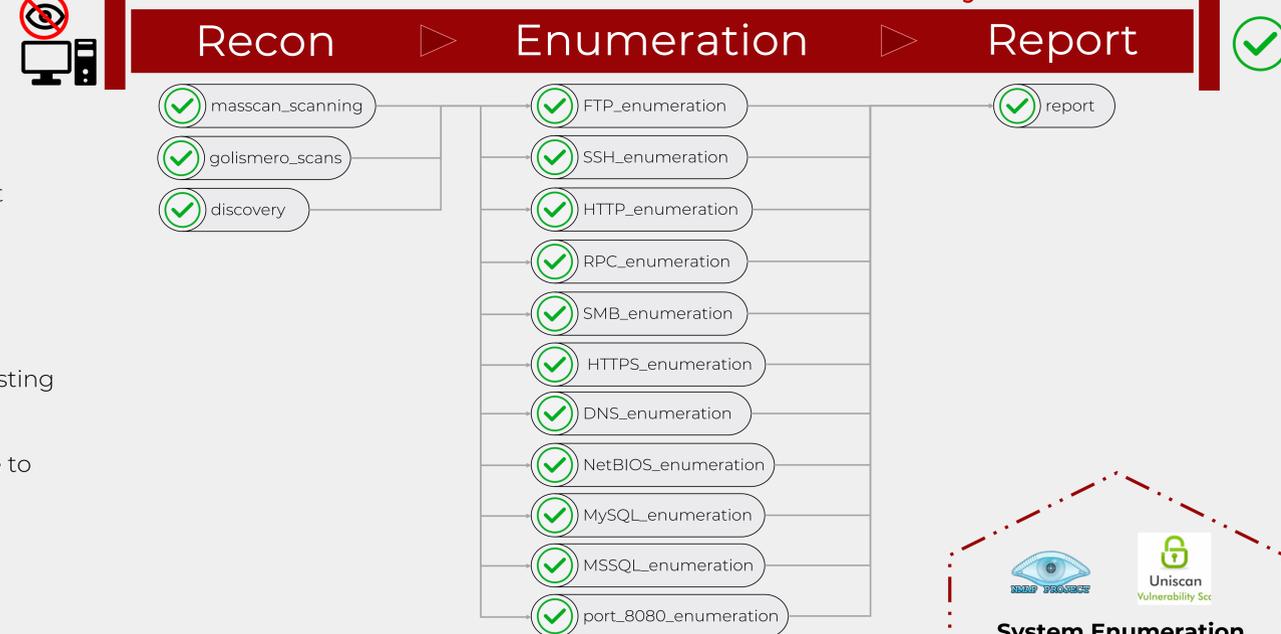
### Use Case: Docker Container Scanning



### Use Case: Evaluate Security of a Physical / Virtual System



### Use Case: Black Box Enumeration of System



## Technologies

