

# Empowering Developers with EPIC Pipelines



*Presented By:*

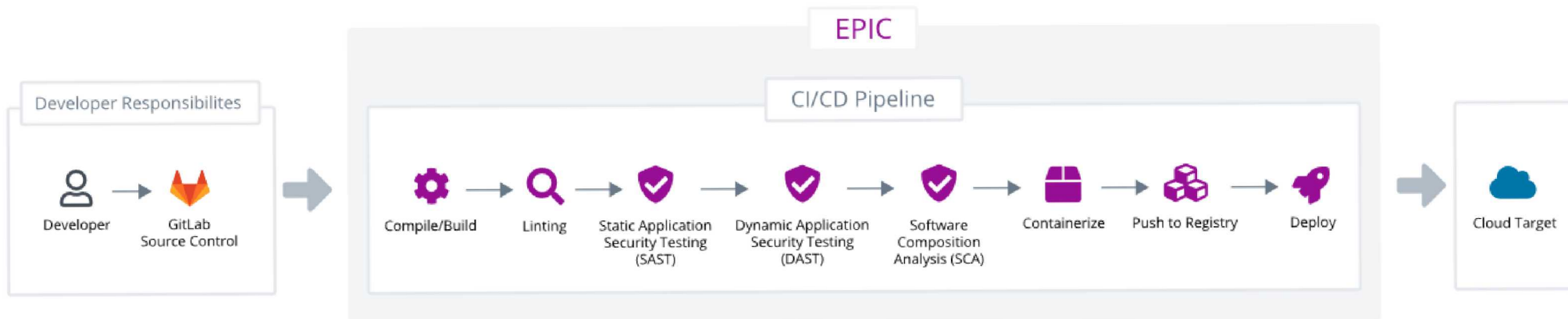
Joey Dickinson & Marc Sanchez



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.

# What Is EPIC?

- A set of standardized CI/CD Pipeline
- Developer tool that simplifies CI/CD at Sandia
- An opinionated view on certain CI/CD steps
- A tool that helps applications get to production faster, more secure, and with high quality.
- Learning tool



## The Problem EPIC solves

- Getting pipelines set up is hard
- Getting tools access is hard (and expensive)
- Need to be a sys-admin to do anything on underlying machines
- Choosing tooling is difficult

# What separates EPIC from other CI/CD platforms?

- Standardization of tools
- Built to be in line with SNL target architecture
- Low barrier to entry for customers
- Integrates with SNL container hosting platforms

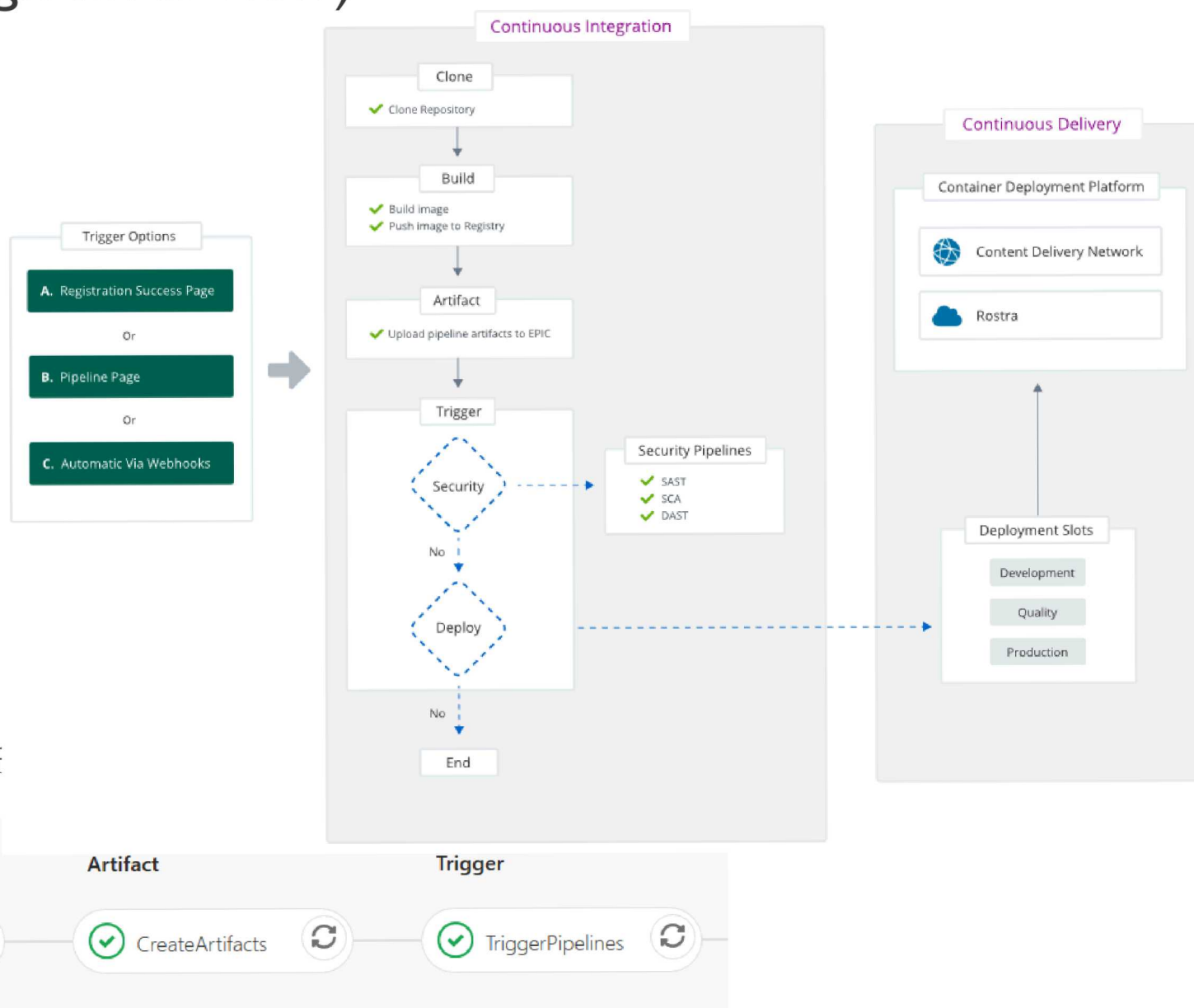
The screenshot displays the EPIC web interface. The top navigation bar is purple with links for 'EPIC', 'Dev', 'My Dashboard', 'Register', and 'Documentation'. On the right, a user profile for 'Hello, Joseph Dickinson' is shown with a 'Help' link. The breadcrumb trail reads 'EPIC / Stack Dashboard / tui / Overview'. The main content area is divided into several sections:

- Stack: tui**: Includes a 'Git URL containing Docker Compose:' field (None) and a 'Repository Checklist' showing '2 Pass', '0 Fail', and '1 Warn'.
- Projects: 1 project**: A link to 'Edit or Add New'.
- Stack Secret Token**: A masked token '\*\*\*\*\*\_\*\*\*\*\_\*\*\*\*\_\*\*\*\*\*' with an eye icon to toggle visibility.
- Owners**: Lists 'Marc Sanchez, Dickinson' and '. Joseph'.
- CI - Build, Containerize & Scan**: A table with columns 'Project', 'Artifacts', 'Pipeline Id', and 'Pipeline Created On'. It shows a pipeline for 'Thunderbird UI' (commit: Pipeline -) created on 'Oct 1, 2020, 7:51:19 AM'. A 'Trigger CI' button is present. A dropdown menu for artifacts is open, listing 'DAST-results.html', 'jshint-results.html', 'eslint-results.html', 'sca-report.pdf', 'dep-check.html', and 'SAST'.
- CD - Deploy**: A section with a 'Trigger CD Pipeline or Rollback' button. It contains three panels: 'Development' (Thunderbird UI, Currently Deployed), 'Quality' (Thunderbird UI, view commits), and 'Production' (Thunderbird UI, Currently Deployed).

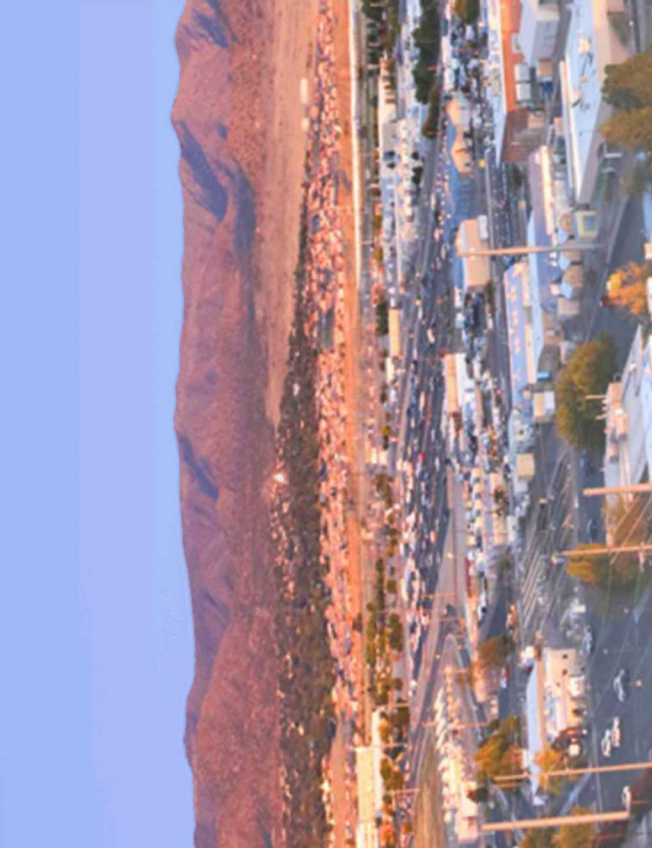
At the bottom of the CI section, there is a link 'View previous CI pipelines →'. At the bottom of the CD section, there is a link 'View previous CD pipelines →'.

# How does it work (high level view)

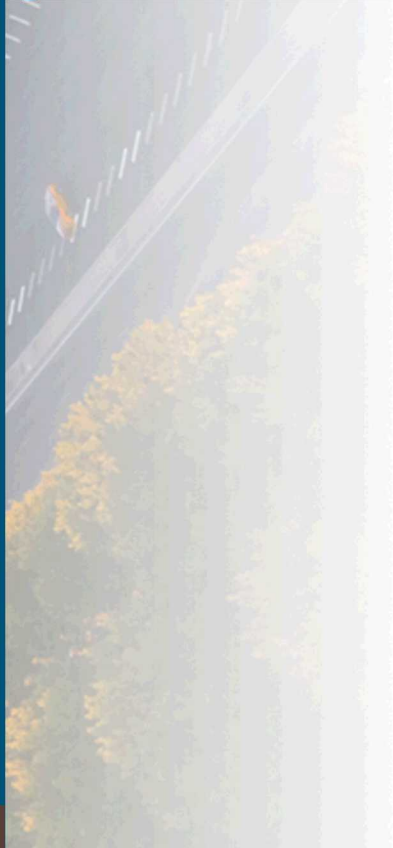
- Register
- Kickoff build pipeline
  - Clone repo's
  - Build with Dockerfile
  - Trigger other pipelines
    - SCA
    - Image Scanning
    - SAST
    - DAST
    - Deploy to container hosting platf







# Security Tools



# SCA – Software Composition Analysis

- OWASP Top 10 Web Application Security Risks
  - #9 – Using Components with Known Vulnerabilities
- Looks at the packages utilized in an application and scans for known vulnerabilities
- Utilize National Vulnerability Database and proprietary databases to match packages to vulnerabilities
  - <https://nvd.nist.gov/>
- Will usually offer guidance on how to resolve the vulnerability
  - Update package
  - Stay away from a known piece or function that is vulnerable
  - Change dependencies altogether

- Evolution of SCA that takes a containerized image and scans it for known vulnerabilities
- Can identify vulnerabilities in OS-level dependencies
  - SCA tools will only look at the deployable application, not how it's deployed
- Also identify common bad practices with running containers
  - Running as root
  - Running without health checks



# SAST – Static Application Security Testing

- Looks at the application's source code to identify security errors made by the developers
  - Using an insecure cryptographic algorithm
  - Using unsafe input validation
- Some of these tools will also call out places to be careful in the code
  - A small syntax error or incorrect function could cause a vulnerability
- Like SCA, will usually provide remediations
  - Use a similar, more secure function
  - Use input validation on user-controlled input

# DAST – Dynamic Application Security Testing

- Tests against the full application stack by supplying malicious inputs
  - Works like an automated penetration tester
- Can find vulnerabilities that may be impossible with SAST or SCA alone
  - Cross-site Scripting
  - SQL Injection
  - Data Exposure
- Generally requires some work by developers to help the tool navigate the application
  - Can be done fully automatically, but tools have a hard time with different authentication methods

# Key Principles

- Provide users with security relevant information early and often
  - Don't wait until the code is moving to production to run security tests
- Don't break builds on found vulnerabilities
  - Other processes in place to ensure quality + security in production
- Tools may evolve over time
  - Keep the same categories of security tests without relying on a specific vendor

