

# Maintaining Your Authority to Operate (ATO) While Being Agile: Achieving Continuous Reauthorization with DevOps

June 2018

Timothy A. Chick

Security Automation Systems Technical Manager

Software Engineering Institute  
Carnegie Mellon University  
Pittsburgh, PA 15213

# Distribution Statements

Copyright 2018 Carnegie Mellon University. All Rights Reserved.

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

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

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

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

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at [permission@sei.cmu.edu](mailto:permission@sei.cmu.edu).

Carnegie Mellon® and CERT® are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM18-0704

# Topics



**What is DevOps**

**What is RMF**

**Security in an Agile World**

**Achieving Ongoing Authorization Decisions**

# DevOps Strategies

---

What are  
the core  
strategies  
of the  
DevOps  
paradigm?

Design flexible software architecture encompassing simple, independent components

---

Implement frequent, incremental changes

---

Integrate innovative, customizable tools that can automate maintenance processes to include communications, testing, deployment, cyber security . . .

---

# DevOps is an Extension of Agile Thinking

Agile

Embrace Constant Change

Embed Customer in team to internalize expertise on domain and requirements

DevOps

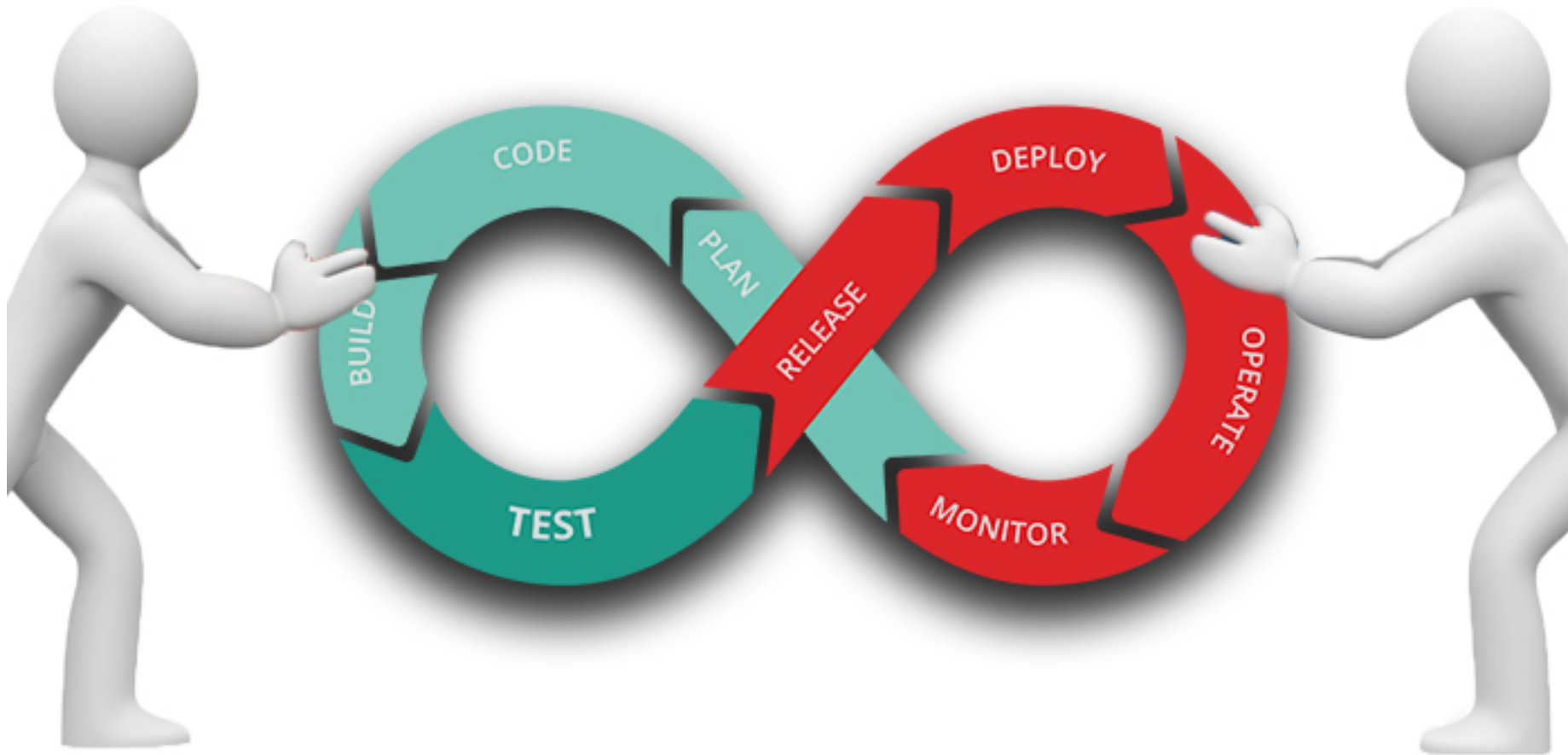
Embrace Continuous Integration, Testing, Delivery

Embed Operations in team to internalize expertise on delivery and maintenance

# DevOps Phases

## DEVELOPMENT

## OPERATIONS



# Topics



**What is DevOps**

**What is RMF**

**Security in an Agile World**

**Achieving Ongoing Authorization Decisions**



# What is the Risk Management Framework (RMF)?

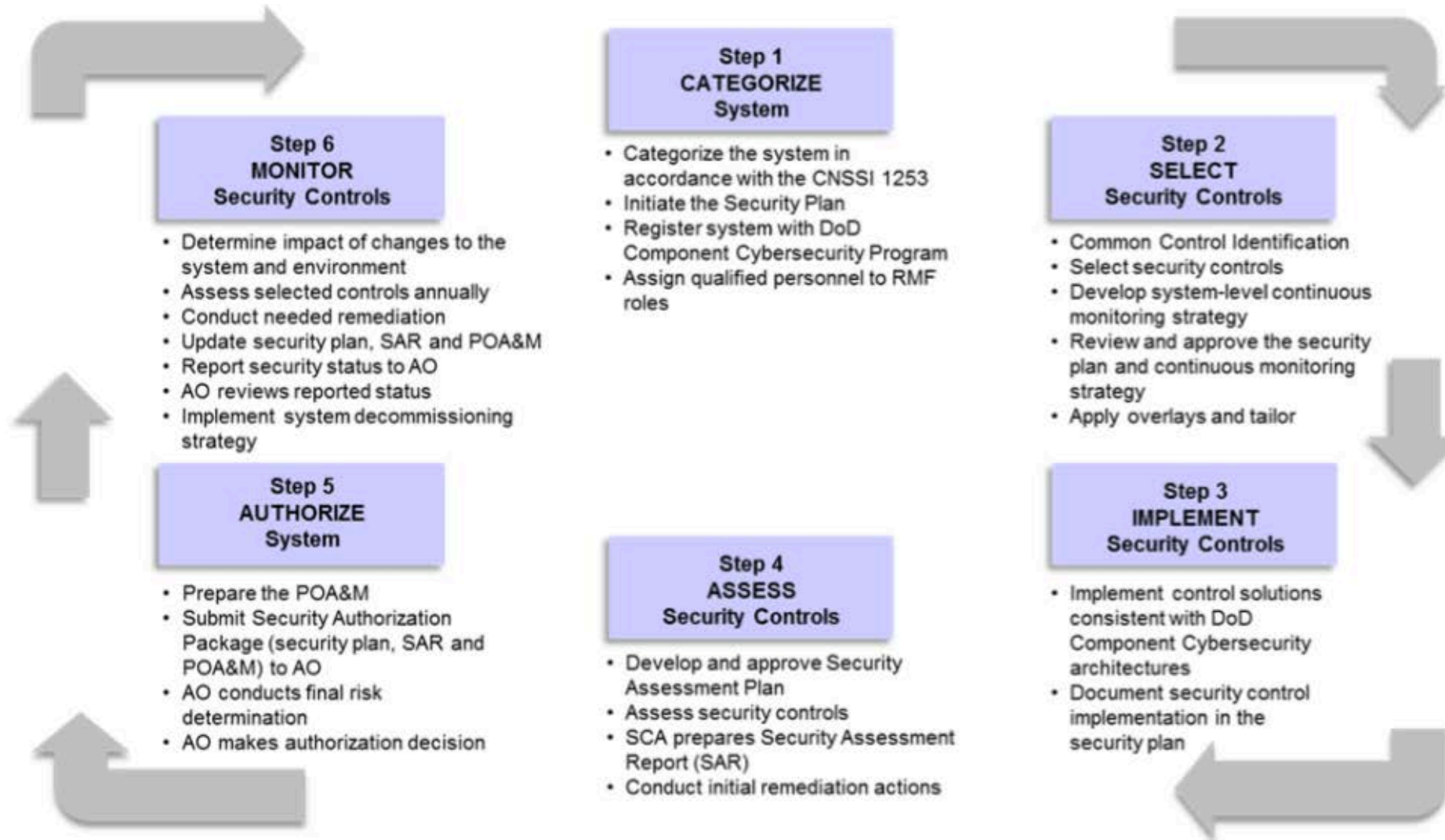
In 2014, the DoD started transitioning from the DoD Information Assurance Certification and Accreditation Process (DIACAP) to the Risk Management Framework for the DoD IT (RMF).

NIST Special Publication 800-37, "Guide for Applying the Risk Management Framework to Federal Information Systems", transforms the traditional Certification and Accreditation (C&A) process into the six-step Risk Management Framework (RMF).

The Risk Management Framework (RMF) provides a disciplined and structured process that integrates information security and risk management activities into the system development lifecycle.



# What is the RMF?



# The RMF/ATO Problem

Every system has **inherent risks** associated with it.

Program Manager (PM) is **graded** against the system's **KPP** and their compliance with all **regulations**, along with **cost** and **schedule** parameters.

PM makes **trades** between cost, schedule, quality, and functionality. With each trade **residual risks** occur.

Someone must **accept ALL residual risk** associated with the system before placing it into operations.

The Authorizing Official (AO) is responsible to **accepting information security risks**, which is done through the RMF process.

An ATO is usually good for 3 years, but **assumes no major changes** to the system's cybersecurity posture will be made during that time.

When **changes** do occur the AO may require a **reassessment** and **reauthorization**, which impacts the PM's cost and schedule and is **contrary to being Agile**.

# RMF's Solution to Problem

RMF encourages an alternative approach to the traditional 3 year ATO process through ongoing authorization decisions or continuous reauthorization.

RMF assumes these systems have “been evaluated as having sufficiently robust system-level continuous monitoring programs”

# Topics



**What is DevOps**

**What is RMF**

**Security in an Agile World**

**Achieving Ongoing Authorization Decisions**

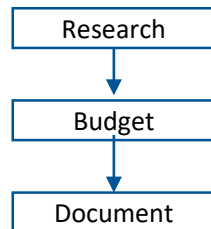
# Security in an Agile World - 1

*Security is often focused on testing, and security activities are often conducted outside and apart from the software development process.*

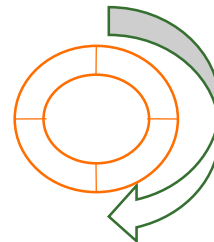
*As a result, the outcomes of security activities are presented in documents and outputs that do not naturally fit any of the software development activities.*

## Water - Scrum - Fall

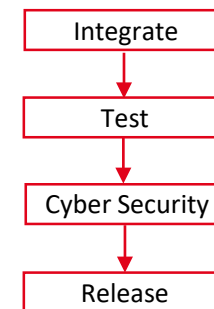
Business



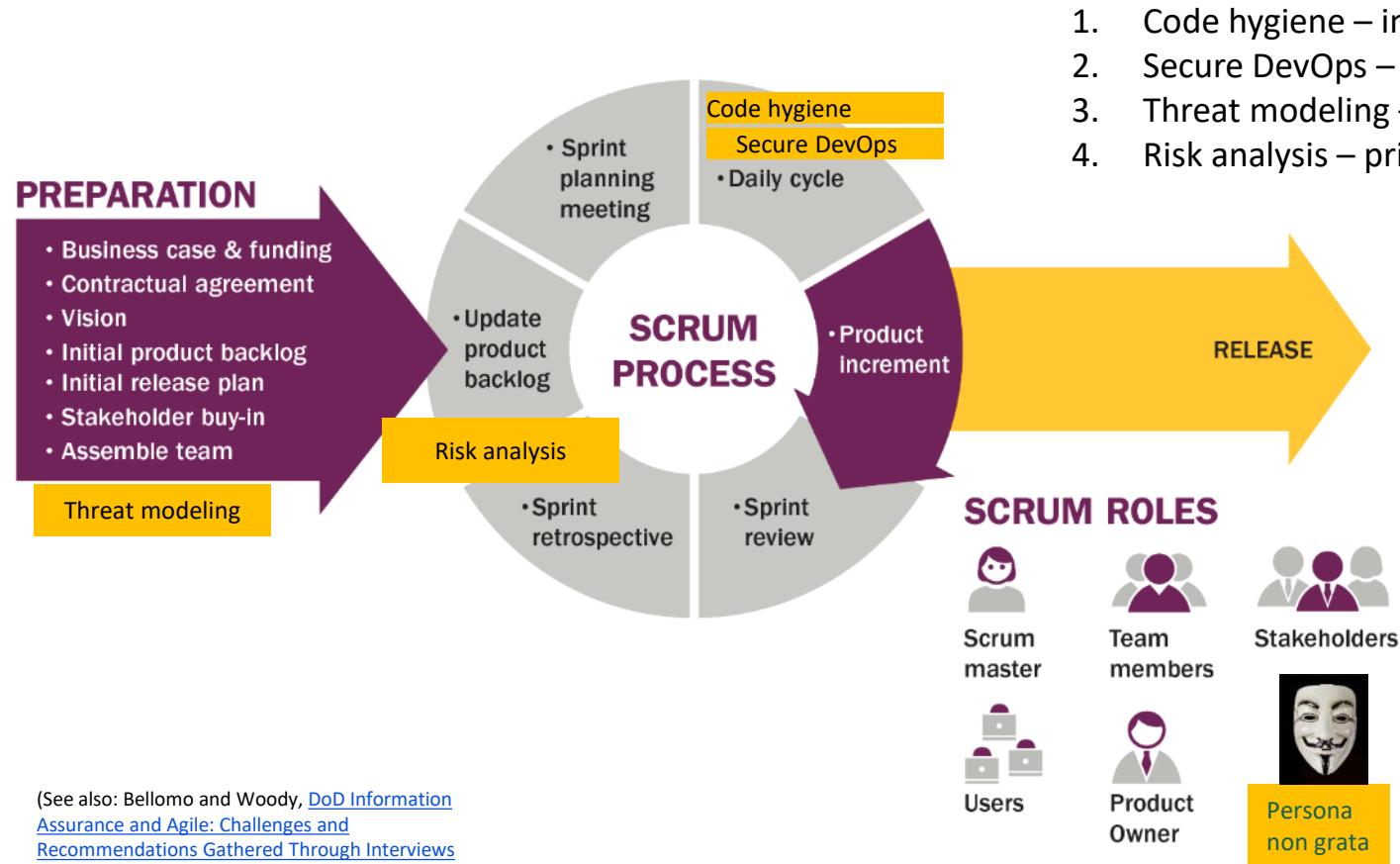
Development



QA  
Operations



# Integrating security into Agile (Scrum) development



(See also: Bellomo and Woody, [DoD Information Assurance and Agile: Challenges and Recommendations Gathered Through Interviews with Agile Program Managers and DoD Accreditation Reviewers](http://repository.cmu.edu/cgi/viewcontent.cgi?article=1674&context=sei) (<http://repository.cmu.edu/cgi/viewcontent.cgi?article=1674&context=sei>))

# Security in an Agile World - 2

*The goal is to guide the development of new activities and make adjustments to existing activities to make it natural and efficient to build security into an agile process.*

**DevSecOps Manifesto** (<http://www.devsecops.org>)

**Leaning in** over Always Saying “No”  
**Data & Security Science** over Fear, Uncertainty and Doubt  
**Open Contribution & Collaboration** over Security-Only Requirements  
**Consumable Security Services with APIs** over Mandated Security Controls & Paperwork  
**Business Driven Security Scores** over Rubber Stamp Security  
**Red & Blue Team Exploit Testing** over Relying on Scans & Theoretical Vulnerabilities  
**24x7 Proactive Security Monitoring** over Reacting after being Informed of an Incident  
**Shared Threat Intelligence** over Keeping Info to Ourselves  
**Compliance Operations** over Clipboards & Checklists

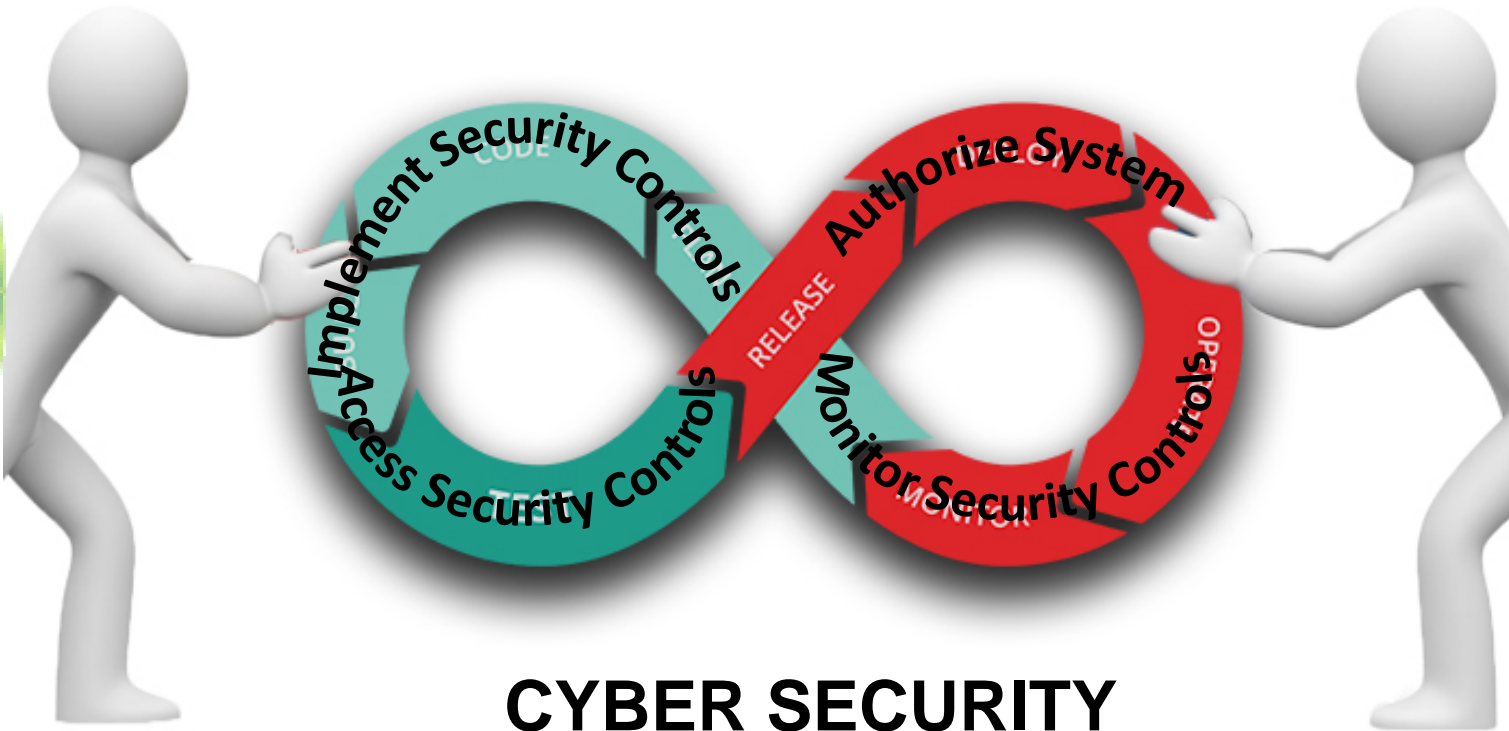
“By developing security as code, we will strive to create awesome products and services, provide insights directly to developers, and generally favor iteration over trying to always come up with the best answer before a deployment.”



# DoD SDLC, RMF, and DevOps

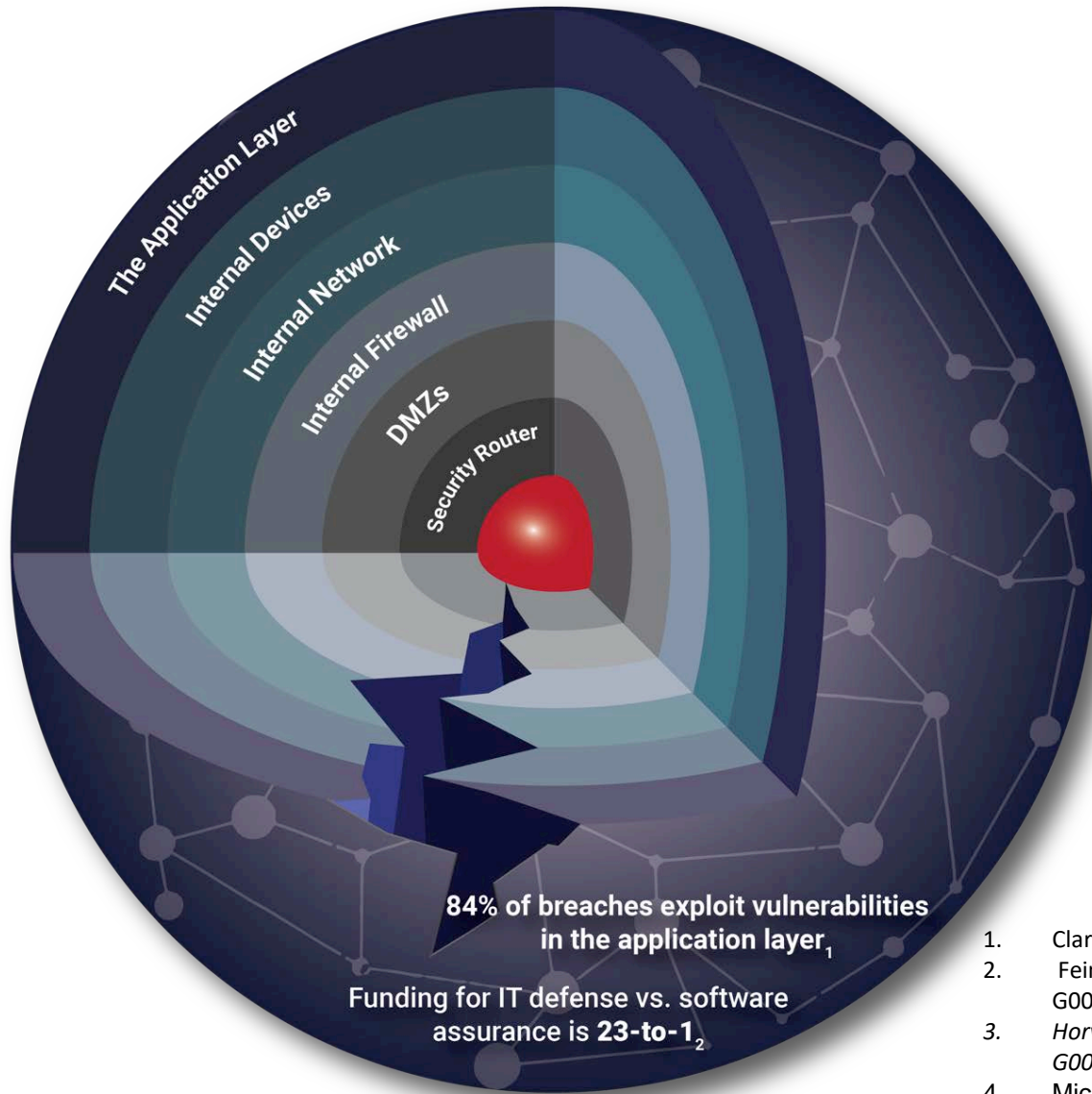
RMF's Continuous Reauthorization concept directly aligns with DevOps

DEVELOPMENT                      OPERATIONS



<http://www.truestonefed.com/wp-content/uploads/2016/09/risk-management-wheel.gif>

# Effective Security Must be Integrated



The Application Layer is the new perimeter exploited by 84% of breaches

Security must be Engineered into the Lifecycle of Applications changing the way we build and buy technology

- “76 percent of U.S. developers use no secure application program process”<sup>4</sup>
- “More than 40 percent of software developers globally say that security isn't a top priority for them”<sup>4</sup>
- 2017 less than 5% of DevOps initiatives have achieved the level of security automation required to be considered fully DevSecOps.<sup>3</sup>

1. Clark, Tim, *Most cyber Attacks Occur from this Common Vulnerability*, Forbes. 03-10-2015
2. Feiman, Joseph, *Maverick Research: Stop Protecting Your Apps; It's Time for Apps to Protect Themselves*, Gartner. 09-25-2014. G00269825
3. Horvath, Mark, Neil MacDonald, Ayal Tirosh: *Integrating Security Into the DevSecOps Toolchain*, Gartner. 11-16-2017. G00334264
4. Microsoft<sup>1</sup>— <http://visualstudiomagazine.com/articles/2013/07/16/majority-of-us-devs-dont-practice-secure-coding.aspx>

# Topics



**What is DevOps**

**What is RMF**

**Security in an Agile World**

**Achieving Ongoing Authorization Decisions**

# DevOps Has Four Primary Focus Areas

Collaboration between project team roles

Infrastructure as Code: Scripted Infrastructure Configuration

Automation of Tasks / Processes / Workflows

Monitoring Applications and Infrastructure

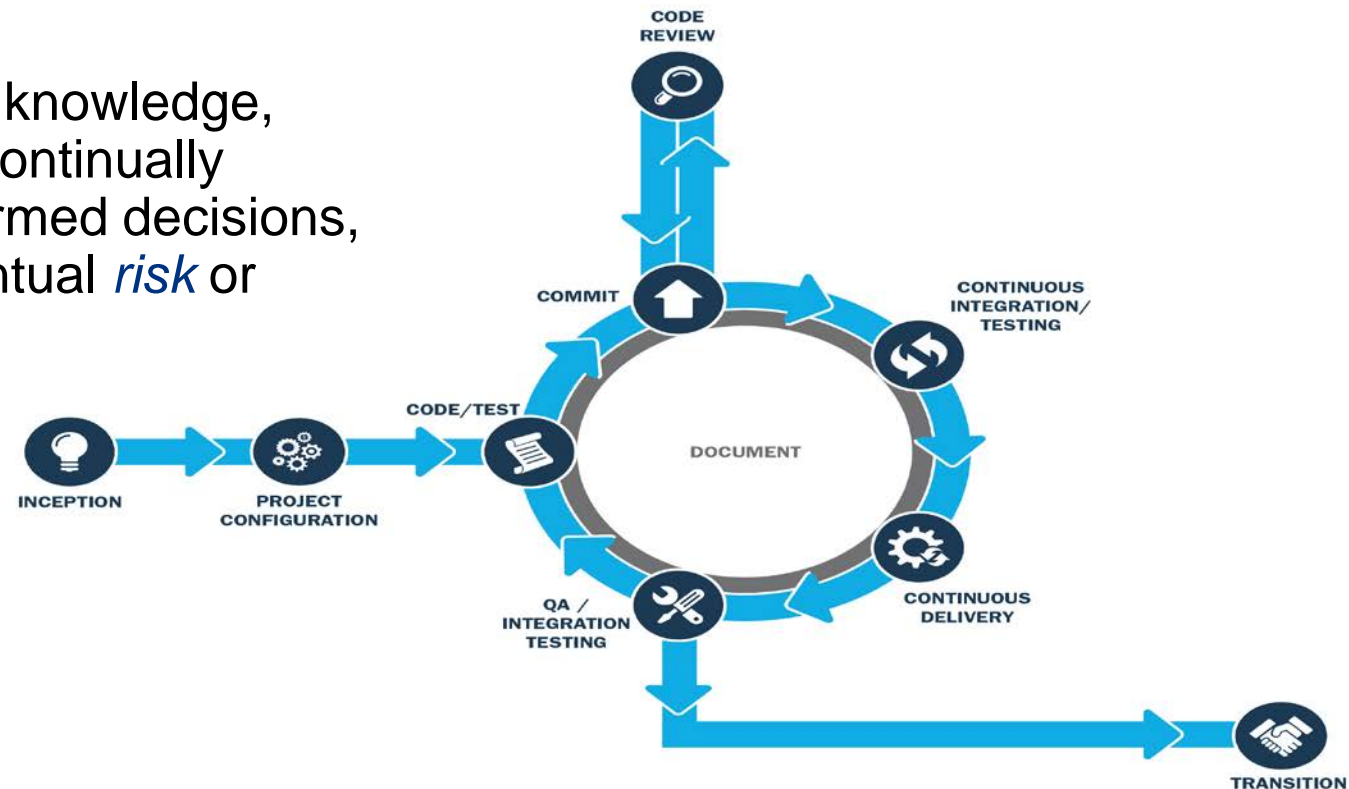
# DevOps Uses Automation to facilitate Communication



- Define Problem Domain
- Capture, Analyze, Track, and Communicate changes
- Collaborate Across Cross-Functional Team
- Used in support controls such as:
  - CA-7(1) Continuous Monitoring
  - CM-2 Baseline Configuration
  - CM-3 Configuration Change Control
  - CM-4 Security Impact Analysis
  - IR-4 Incident Handling
  - MA-6 Timely Maintenance

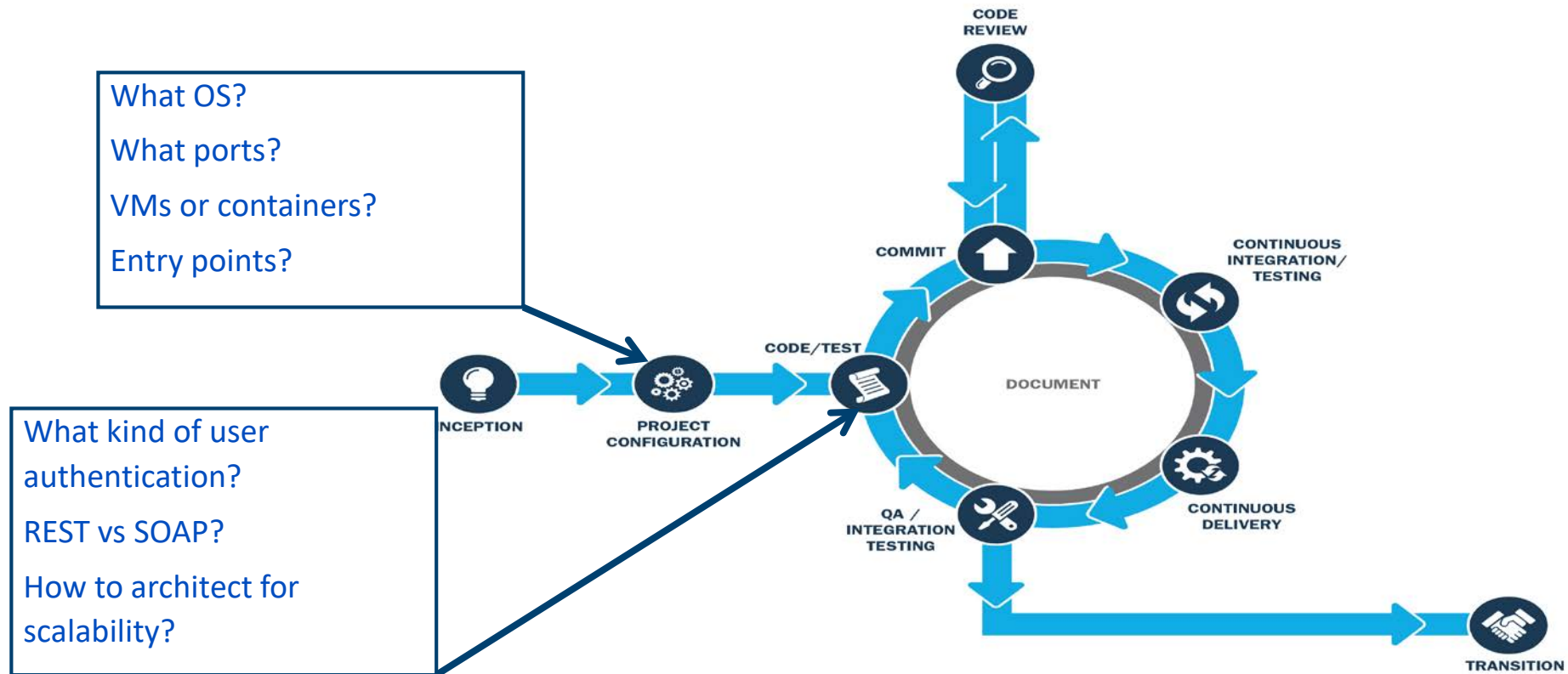
# The SDLC is Full of Decision Points

Without Ops knowledge, developers continually make uninformed decisions, causing eventual *risk* or *inefficiency*





# The SDLC contains many Decision Points with Security Implications



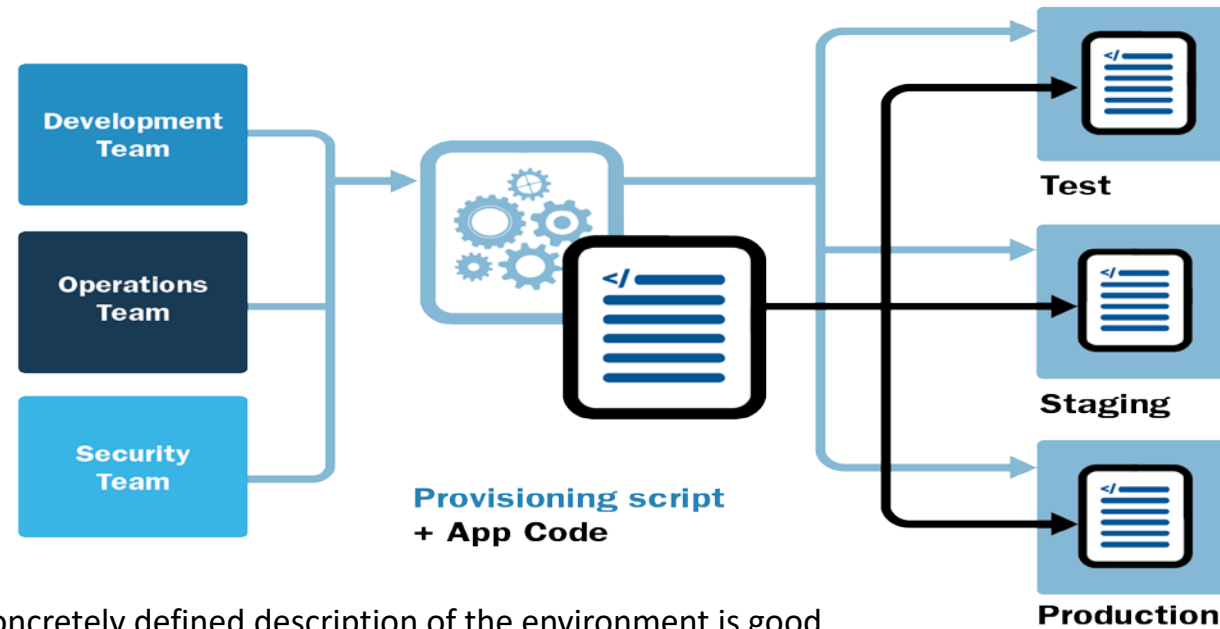




# Infrastructure as Code

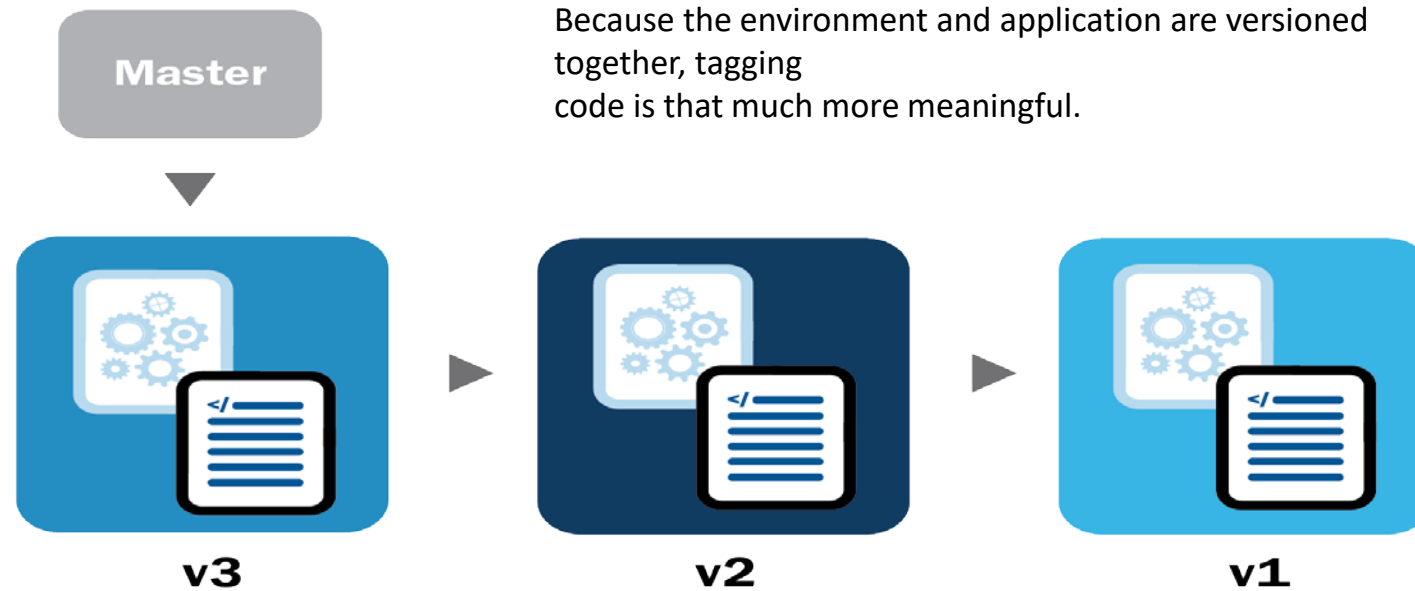
# What is IaC?

A program that creates infrastructure



\* A concretely defined description of the environment is good material for conversation between team members.

# IaC Apparent Benefits



# IaC Provides a Solution – Scenario #1

## Scenario #1

A vulnerability is being exploited in production that cannot be reproduced in development. Even rolling back development code to the production version doesn't allow it to manifest. It may be an issue with updated packages or OS in development.

Examples of related RMF Controls:

- IR-3 Incident Response Testing
- IR-10 Integrated Information Security Analysis Team
- SA-10 Developer Configuration Management

**Versioned Environment**

# IaC Provides a Solution – Scenario #2

## Scenario #2

The operations team is following recovery instructions for the production environment based on documentation. It turns out there is a dependency problem because an incorrect version of a package was cited.

Examples of related RMF Controls:

- CP-2 Contingency Plan
- CP-6 Alternative Storage Site
- CP-7 Alternative Processing Site
- CP-10 Information System Recovery and Reconstitution

**Scripted Environment**

# laC Provides a Solution – Scenario #3

## Scenario #3

Security features that worked perfectly during testing fails when deployed to the production infrastructure.

Examples of related RMF Controls:

- CA-8 Penetration Testing
- CM-2 Baseline Configuration
- CM-3 Configuration Change Control
- CM-4(2) Security Impact Analysis | Verification of Security Functionality
- CM-6 Configuration Settings

**Environment Parity**



# Common Tools

**Shell scripts** – scripting platform-specific commands

**Vagrant** - assists in managing virtual machines and provisioning

**Chef** or **Puppet** - wrappers around your shell

- provide hooks and convenience methods

- layer of indirection between script and OS-specifics

- provides portability

**Docker** – deployable Linux containers

- runs on Linux only (for now)

- whole environments can be easily shared



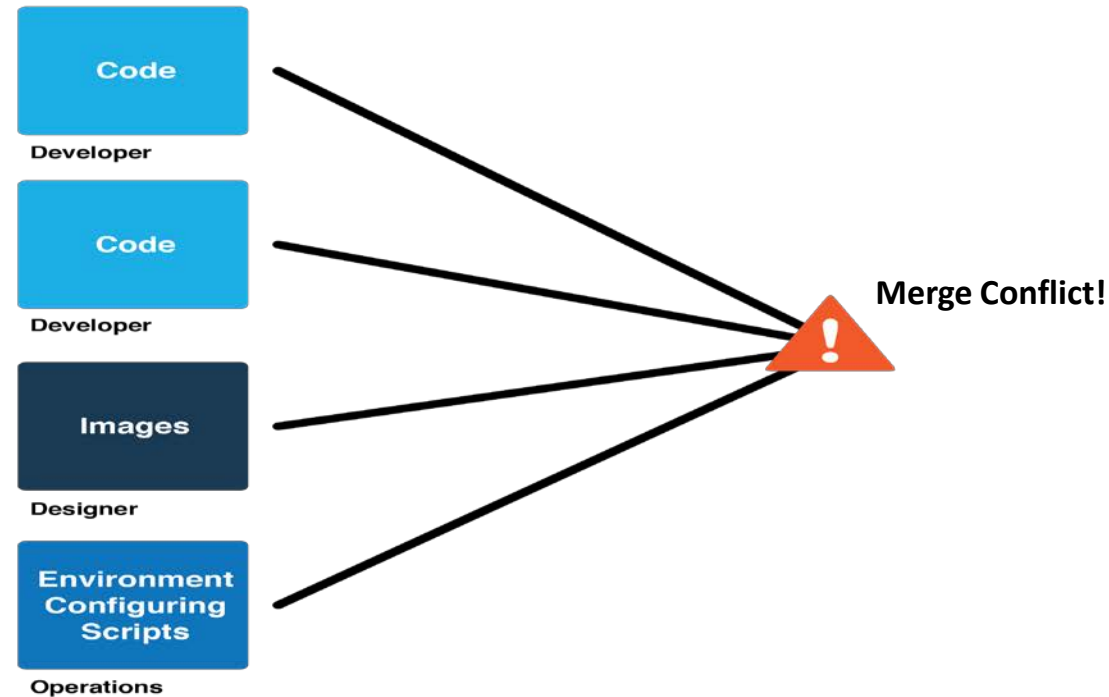


# Automation of Tasks, Processes, and Workflows

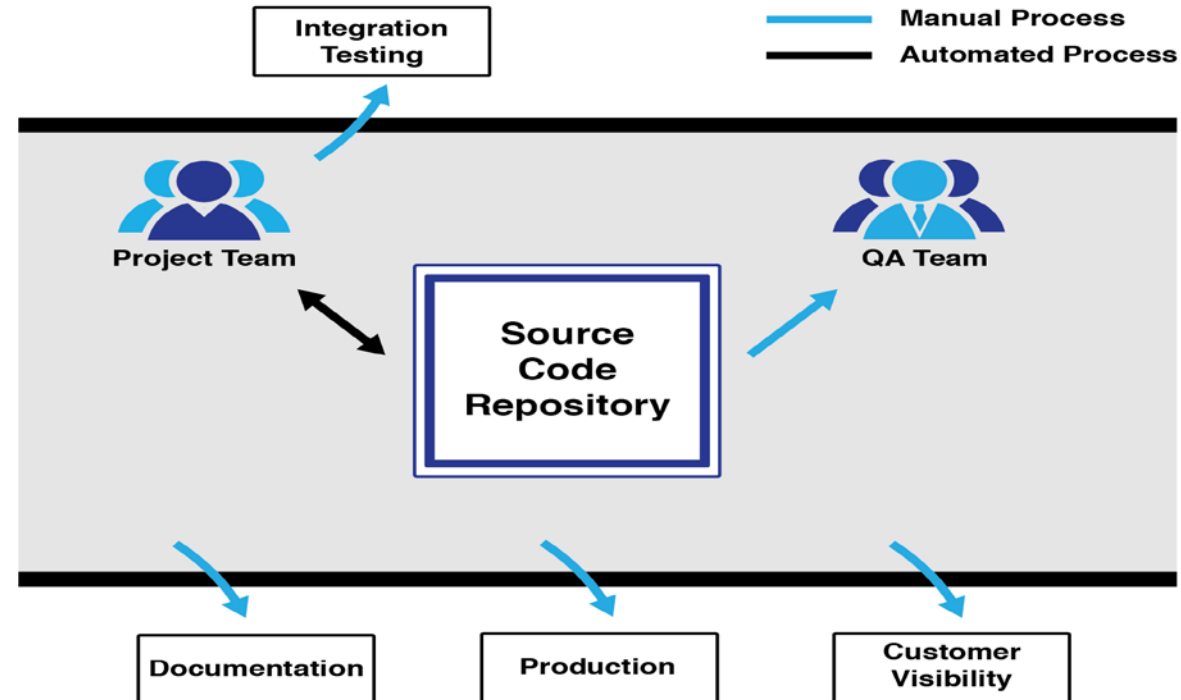
(SA-15 Development Process, Standards, and Tools)

# Software projects consist of many artifacts

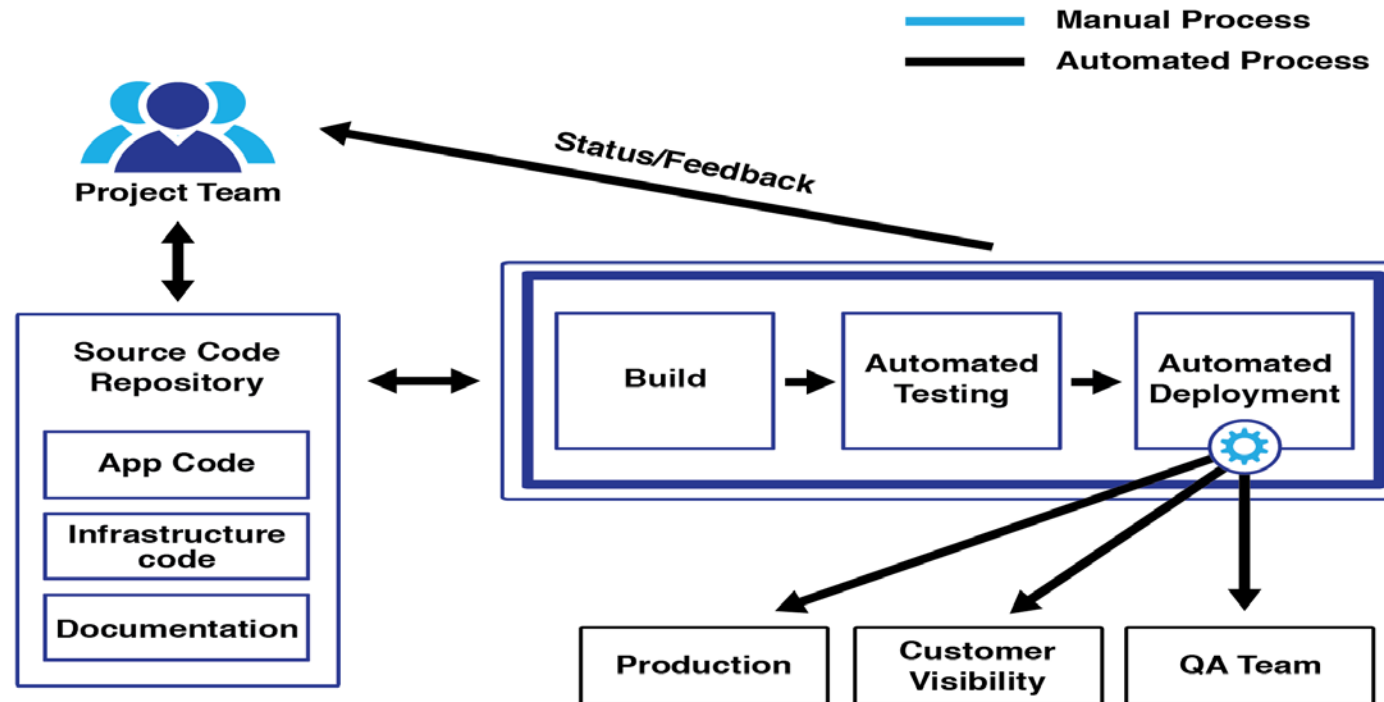
## Integration can be challenging



# This is often a manual process



# Continuous Integration (CI) Model



# Fail the Build When Software isn't Good Enough

Don't just configure failure for compile/build errors!

- Does the changes include a know weak coding practices (CWE)?
  - Automatically run changes against a static code analysis tool and fail the build if a new CWE is found
- Do any of the current or new libraries have known vulnerabilities (CVE)?
- Did any Functional Security Tests Fail?
- Example Security Controls:
  - SA-11 Developer Security Testing and Evaluation
  - SA-12 Supply Chain Protection
  - CM-4 Security Impact Analysis
  - RA-3 Risk Assessment

CI is your best tool to enforce security standards



# Functional Security Tests

Automated unit, integration and acceptance testing tools can be used to verify security controls.

A large proportion of security tests are essentially checks that known weaknesses have not been introduced.

The following security controls are examples of controls that can be monitored/tested using existing acceptance testing browser automation tools like Selenium:

- AC-2(4) Account Management | Automated Audit Actions
- AC-2(5) Account Management | Inactivity Logout
- AC-2(11) Account Management | Usage Conditions
- AC-6 Least Privilege
- AC-12(1) Session Termination | User-Initiated Logouts/Message Displays
- AC-7 Unsuccessful Login Attempts
- CM-2 Least Functionality
- IA-2 Identification and Authentication (multi-factor)

# Continuous Integration Systems



**Jenkins**







# Monitoring Applications and Infrastructure

# Clarifying Terms

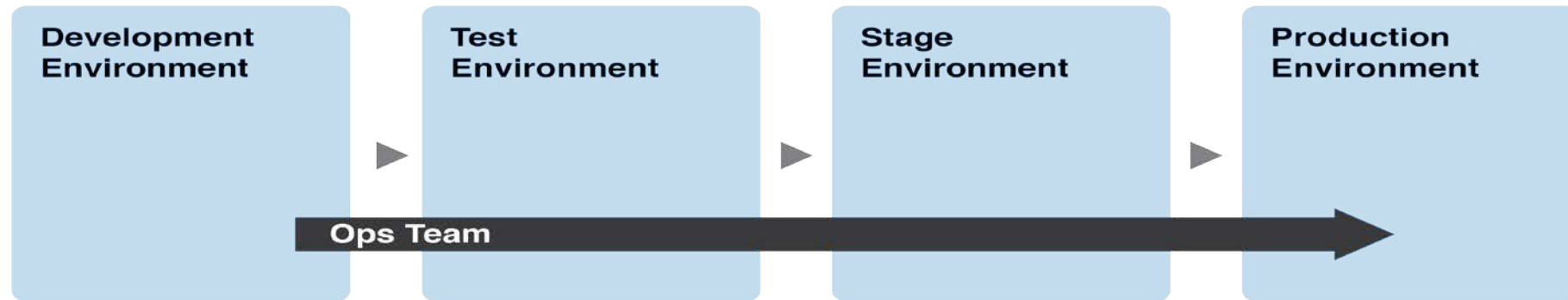
## Continuous Deployment

Changes are deployed ASAP into production

## Continuous Delivery

Changes are deployed immediately into a *production-like environment*, to ensure that they *could* be deployed into production

## Shift Left Operational Concerns



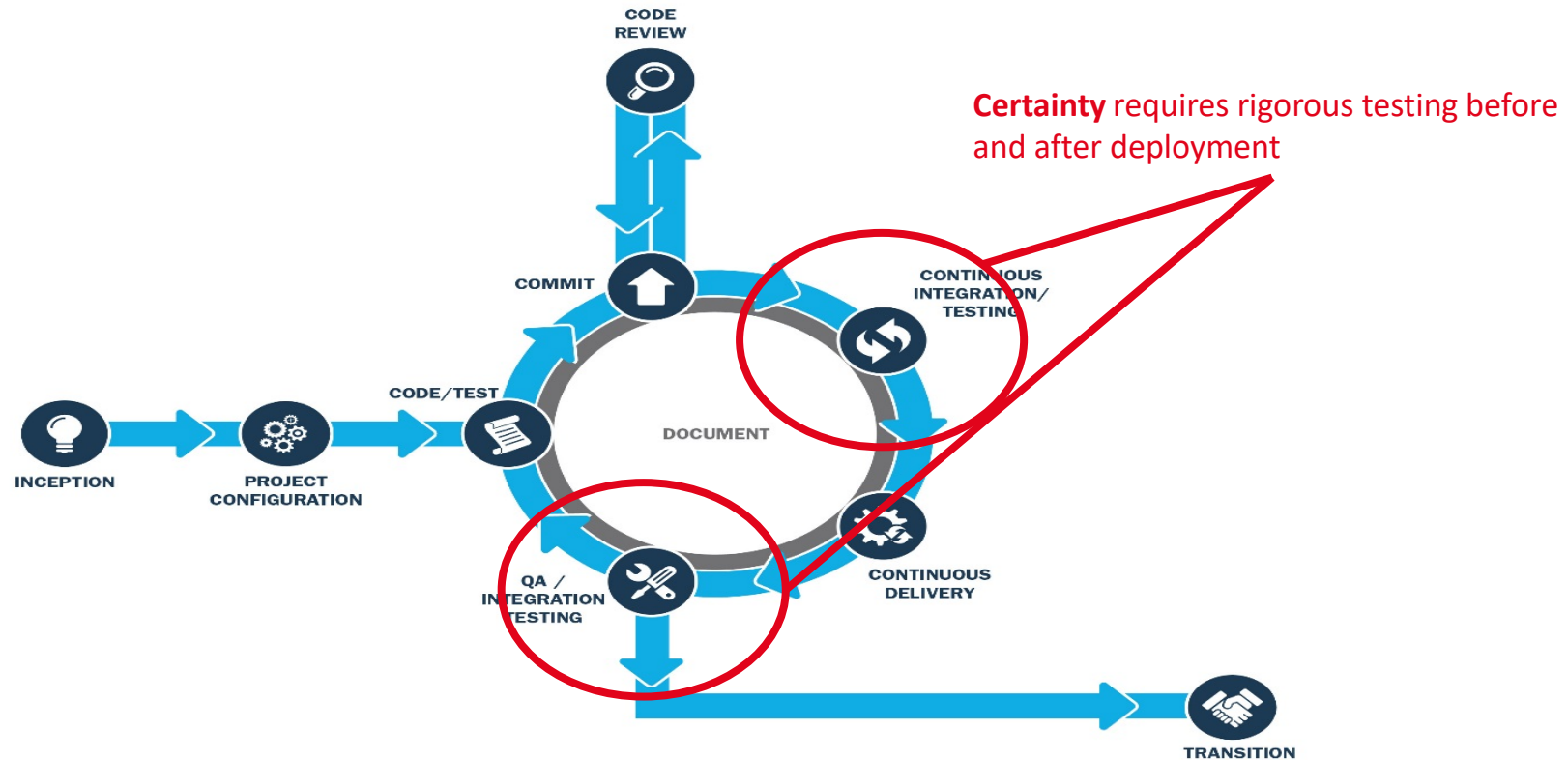
### Benefits of Continuous Delivery

- Speed time to market
- Reduce cost
- Reduce risk
- Scale

# Gold-standard deployment

- Environment Parity
- Process Parity
- Automate
- Perform incremental changes
- Appropriate Testing

# Continuous Delivery Is REALLY About Rigorous Testing

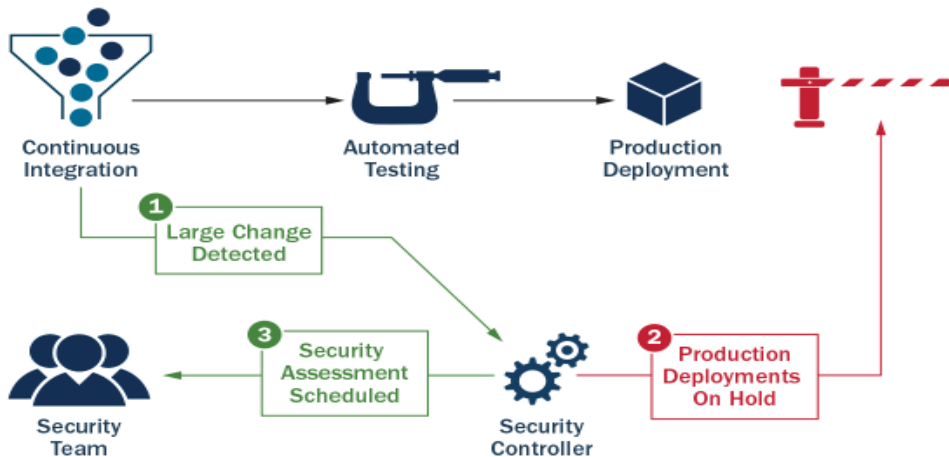


# Test Enough That You Are Sure You Could Deploy Successfully

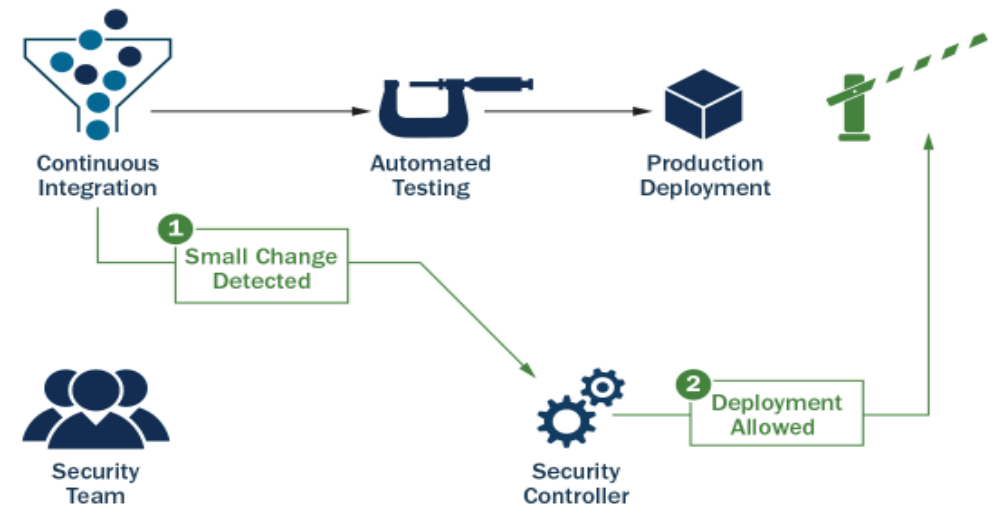
How much is enough?

What factors are important to you and your organization?

**Security?** Automate a large number of security-focused unit/integration tests



Design your CI/CD success criteria to enforce your goals





# DevOps Operate Phase



- Continuous Monitoring
- Performance Measurement
- Incident Management
- System Security
- Auditing

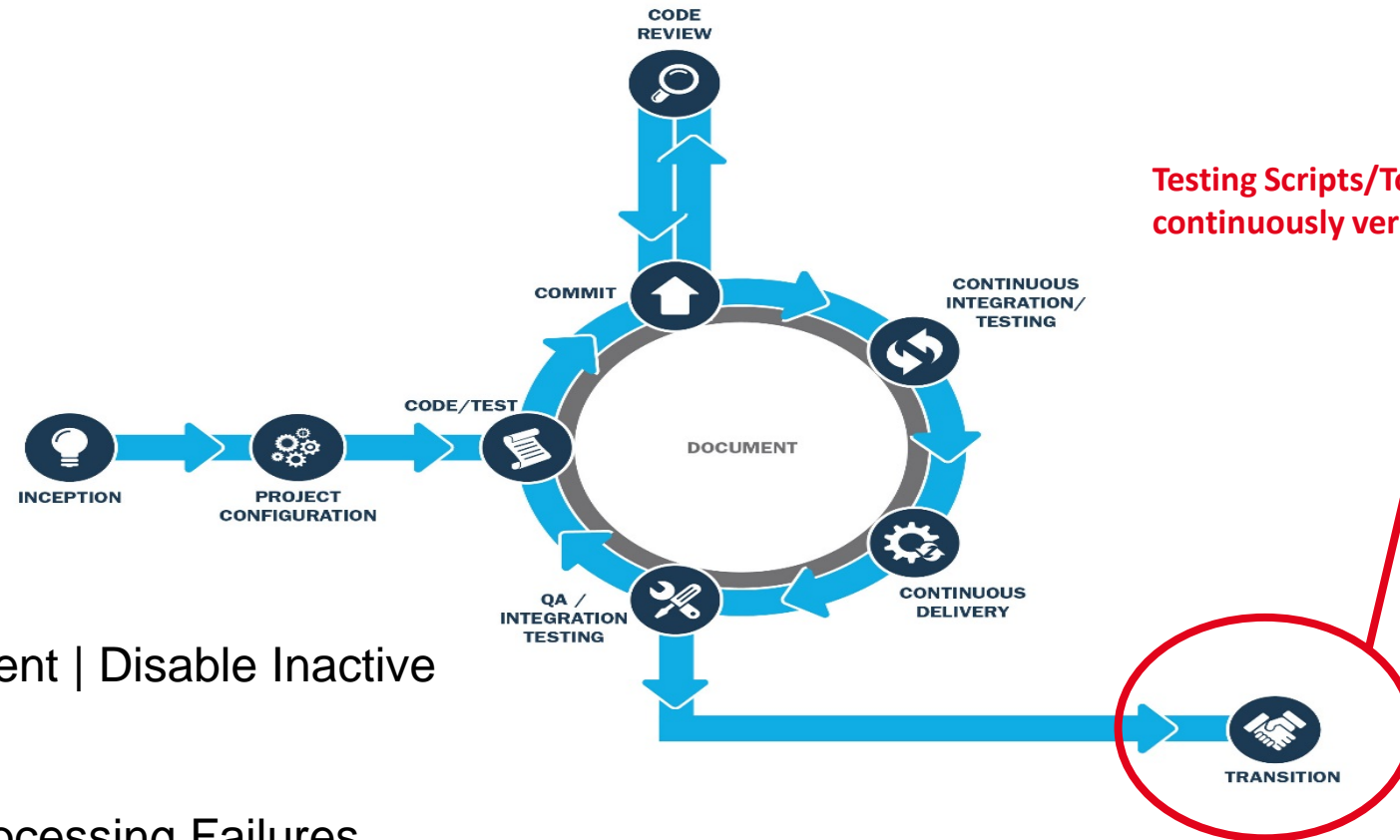


# DevOps and RMF Integration

There are several RMF process steps that can be executed throughout the DevOps deployment pipeline including:

- **Develop Security Assessment Plan**
  - DevOps team provides details/scripts regarding the implementation and configuration of automated testing tools
- **Assess security controls**
  - DevOps practices improve this process by automatically providing outputs from team code reviews, application scanning and regression tests
- **Prepare Security Assessment Report**
  - Automated test tool output can be generated into a suitable format for inclusion as input
  - These reports are generated as part of the Continuous Testing cycle (not just annually), so assessment data is fresh and reports can be compared for differences
- **Conduct initial remediation actions**

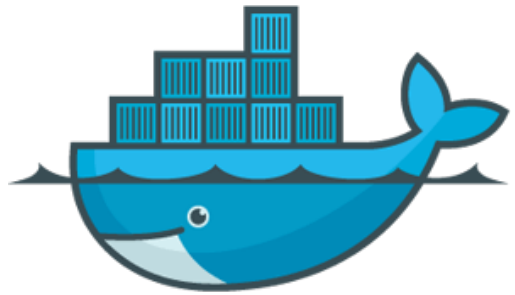
# Continue Automated Testing in Production



## Example Security Controls:

- AC-2(3) Account Management | Disable Inactive Accounts
- AU-2 Audit Events
- AU-5 Response to Audit Processing Failures
- AU-9 Protection of Audit Information
- AU-11 Audit Record Retention
- SI-7 Software Firmware, and Information Integrity
- SI-11 Error Handling

# DevOps Operate Phase: Integrate the Necessary Tools



docker



Prometheus

An open-source service monitoring system and time series database.

Nessus



MONIT



elasticsearch.



logstash



Kibana



SNORT™

pagerduty



ANSIBLE

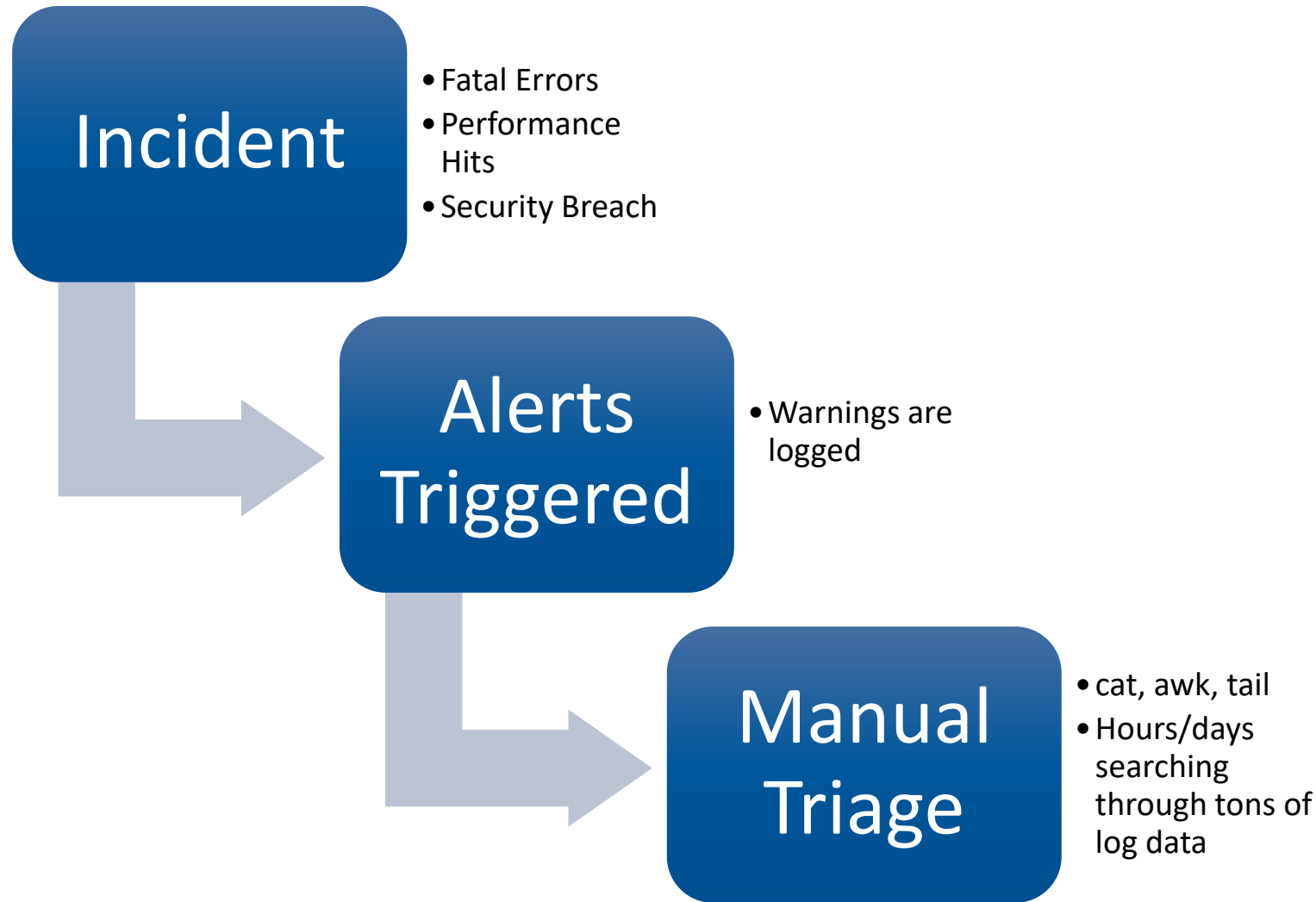
splunk>



# DevOps Operate Phase: Performance Measurement



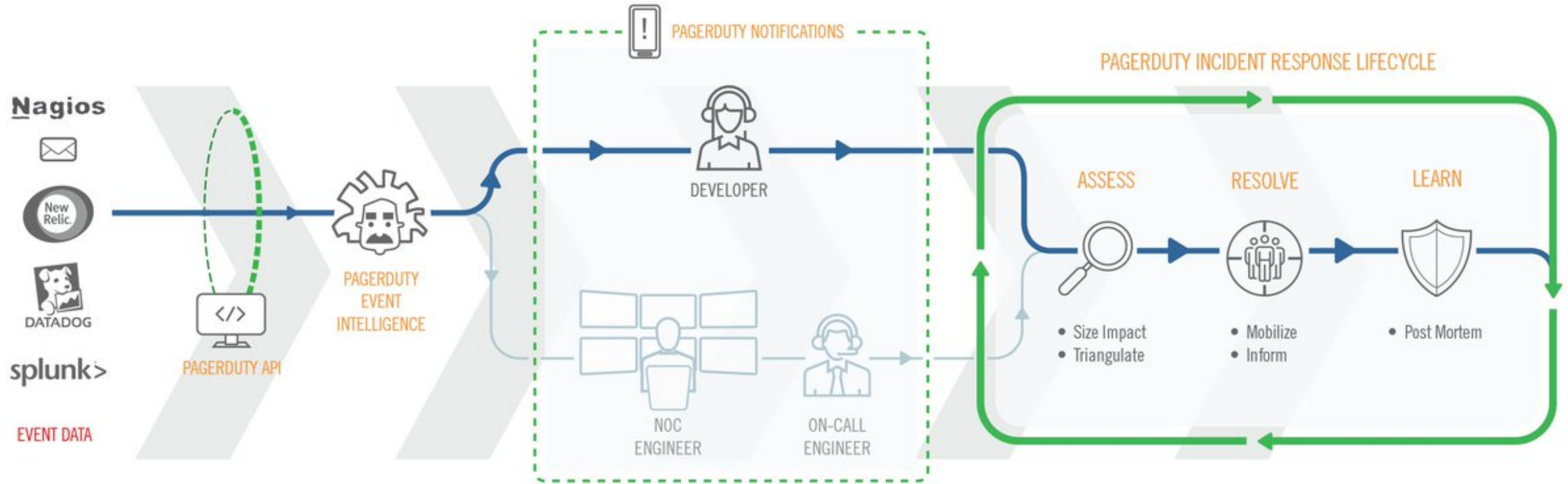
# DevOps Operate Phase: Incident Management



Before the advent of DevOps strategies and associated tools, handling production incidents (application errors, performance hiccups, security breaches) was a tedious and time-consuming process.

Integrating DevOps tools (e.g. PagerDuty) can automate the process of incident monitoring, handling and reporting (IR-4, IR-5, IR-6, SI-4 Security Controls)

# DevOps Operate Phase: Incident Management





# DevOps Operate Phase: System Security

DevOps strategies and tools greatly enhance the tedium of implementing and managing application security (SI-4, SI-5, SI-6 Security Controls).

IaC tools such as Puppet can be used to configure, deploy and manage tools like Nessus and Snort.

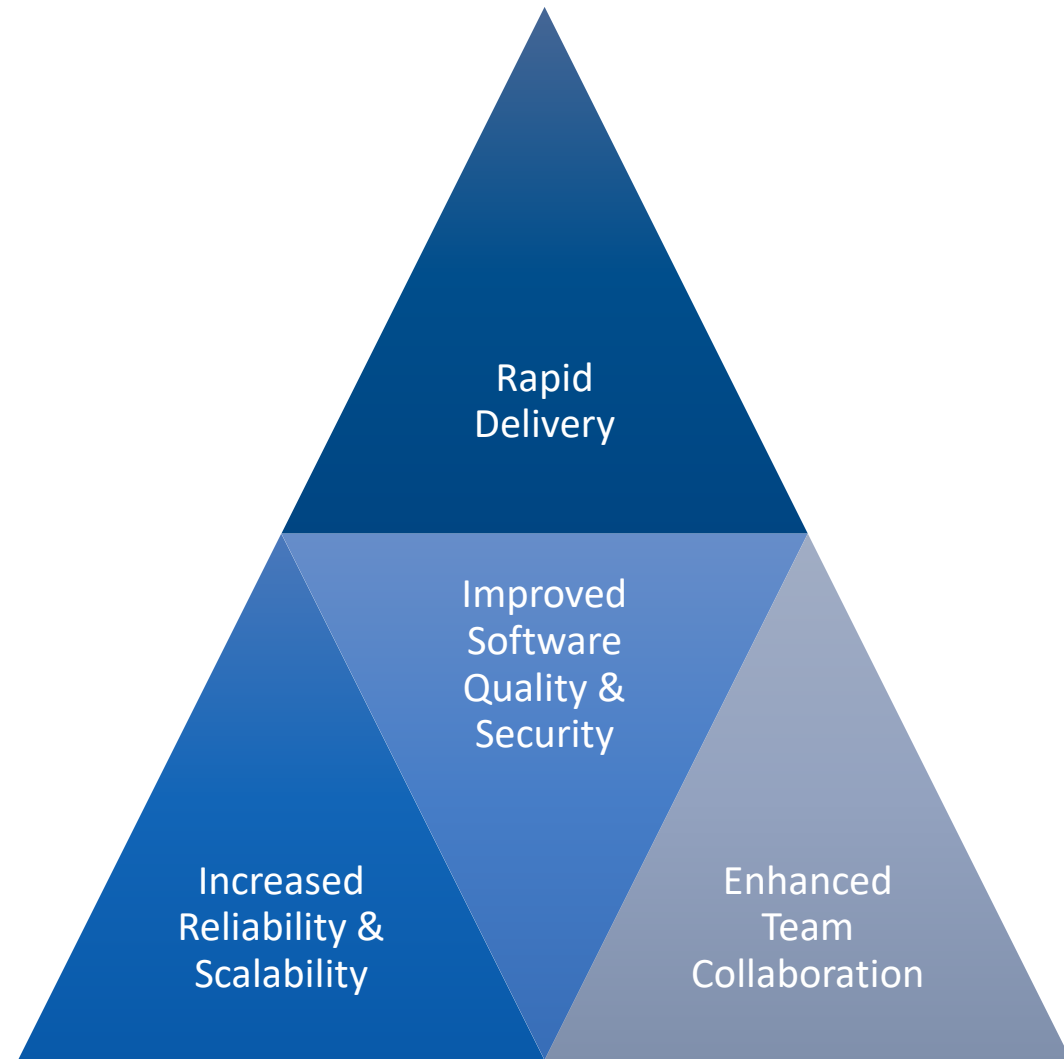
Nessus scans systems and applications for weaknesses and vulnerabilities (RA-5). Reports are automatically generated and sent to necessary members of cross-functional team.

Snort can be deployed to automate intrusion detection (SC-38).



# Benefits of DevOps

When security is fully integrated into the full Software Development, Maintenance, and Operational lifecycle a sufficiently robust system-level continuous monitoring program can be demonstrated in order to achieve a continuous reauthorization (ATO)



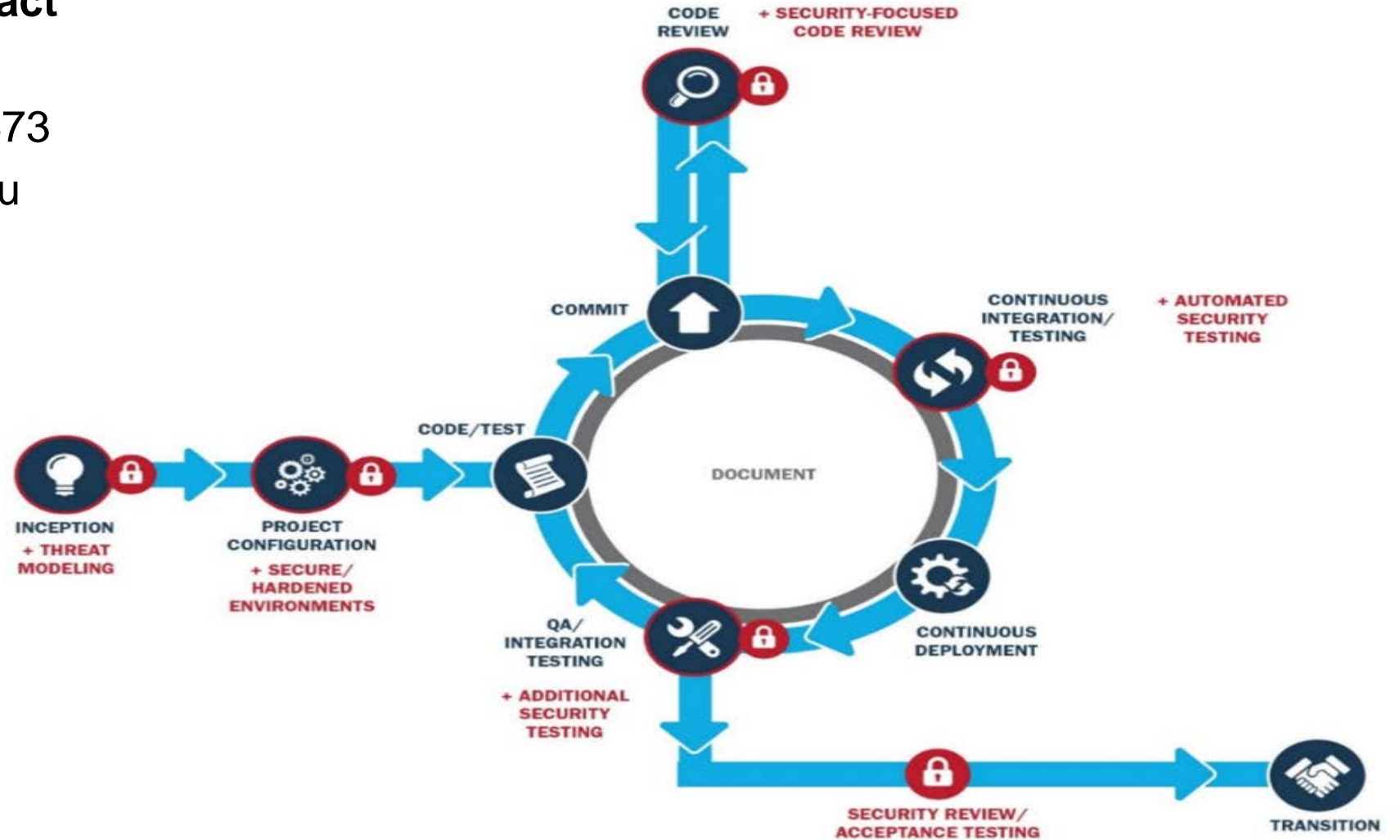
# Contact Information

## Presenter / Point of Contact

Tim Chick

Telephone: +1 412.268.1473

Email: tchick@sei.cmu.edu



# Security Controls

AC – Access Control

AT – Awareness and Training

AU – Audit and Accountability

CA – Security Assessment and Authorization

CM – Configuration Management

CP – Contingency Planning

IA – Identification and Authentication

IR – Incident Response

MA – Maintenance

MP – Media Protection

PE – Physical and Environmental Protection

PL – Planning

PS – Personnel Security

RA – Risk Assessment

SA – System and Services Acquisition

SC – System and Communications Protection

SI – System and Information Integrity