Continuous ATO Playbook

Constructing a Secure Software Factory to Achieve
Ongoing Authority to Operate

Authorizing Official for Cyberspace Innovation
SAF/CIO A6

Disclaimer

Acknowledgements

The Continuous ATO Playbook was researched, developed and written by Wolf Den Associates, LLC in support of the Air Force's Cyberspace Innovation team led by Ms. Lauren Knausenberger. For questions or inquiries, please contact Rick, Nathan, or Tyler.

WOLF DEN
ASSOCIATES, LLC
www.wolfdenassociates.com
1751 Pinnacle Drive
McLean, VA 22102

Rick Tossavainen, Vice President
Phone Number: 703-638-5924
Email: Rick@wolfdenassociates.com

Nathan Hintz, Senior Associate
Phone Number: 434-409-8082
Email: Nathan@wolfdenassociates.com

Tyler Fordham, Senior Associate
Phone Number: 509-844-6977
Email: Tyler.Fordham@wolfdenassociates.com

Table of Contents

Table of Exhibits

1.0 Introduction to the Continuous ATO Playbook

The purpose of this Playbook is to provide a process and recommendations to government entities, both in the Department of Defense (DoD) and Civilian agencies, about how to create a Software Factory to obtain an ATO for all applications developed within the factory. This work was inspired by the move of software development organizations to a DevOps model, which requires a reshaping of current security processes to meet the objectives of continuous integration and deployment. The goal is to enable agencies to achieve better results by implementing an outcome-based performance framework in place of a compliance-based documentation framework as is common today. The Playbook was developed as a result of the Continuous ATO that was signed for the Kessel Run program as part of the Air Force Air Operations Center (AOC) Pathfinder.

The contents of this Playbook are not targeted at any specific role within the organization but provides recommendations and considerations for every level of the organization, ranging from the Chief Information Officer (CIO) to individual developers.

2.0 Introduction to the Concept and Implementation of Continuous ATO

A Continuous ATO was authorized at the Air Force for the Kessel Run program based on NIST 800-37 rev 2 and the NIST *Supplemental Guidance on Ongoing Authorization*, which allow ongoing risk determinations upon completion of an initial risk assessment. On 18 April 2018, Lauren Knausenberger, Director, Cyberspace Innovation, signed a memo to implement ongoing authorization for agile software development for an Air Force team. The scope of the ATO is for the platform "and all software products produced within their Software Factory pipeline." The Software Factory is made up of all tools, technologies, and processes that encompass the software development lifecycle (SDLC). By authorizing the factory, all applications that are developed within the factory will be granted ATOs upon release to production. Essentially, the ATO is shifting from the final product (as is commonly done currently) to the process by which the product is developed. The focus is shifting from the outputs to the inputs.

As the Kessel Run Continuous ATO Brief – 11 APR 2018, referenced in the authorization memo for the Software Factory states: "The core concept of #continuousATO is to build software security into the agile software development methodology so that the ATO process (as with the testing process) is done alongside development. If done correctly, an ATO is nearly guaranteed once the software is release ready." For clarity's sake, Continuous ATO, as it's described in that project, is essentially a named rebranding of the concept of Ongoing Authorization prescribed in the NIST *Supplemental Guidance on Ongoing Authorization* document.

The Continuous ATO is based on the hypothesis that building security into the entire development lifecycle, using best-in-breed tools and technologies and following industry-leading processes, ensures that an application has a level of security that is equal to, if not greater than, the security assured by the traditional Risk Management Framework (RMF) process.

The Continuous ATO relies on several key assumptions:

- Developers and Administrators want to deploy secure applications
- A process that bakes in security from the beginning will be more secure than one that tacks it on at the end
- A DevOps model provides continuous code updates which must be tested at a pace which cannot be met using traditional security review methods

- The Government must adopt more progressive practices such as Agile and DevOps for many of their applications to meet the accelerating pace of mission

2.1 Paradigm Shift in Software Development and Focus of Security Authorizations

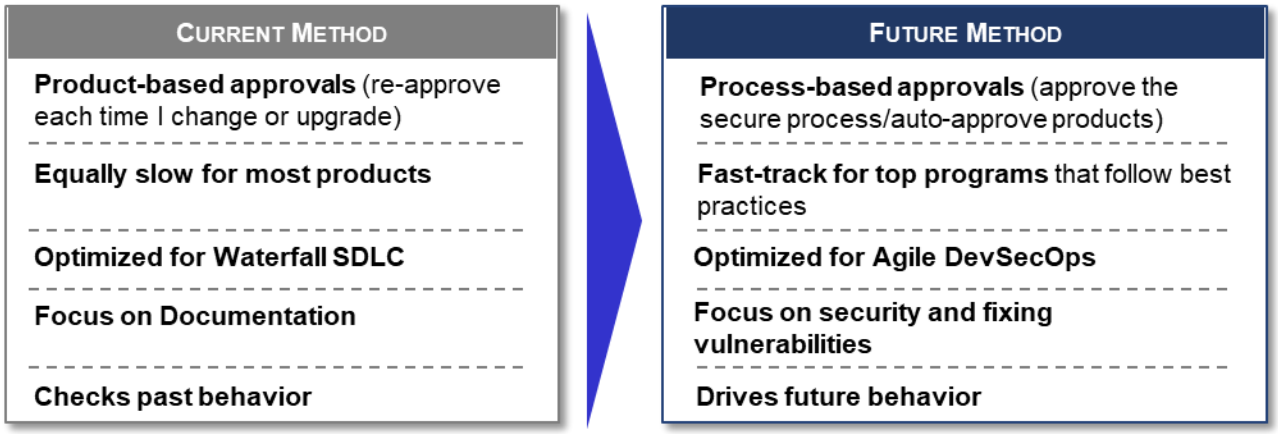
In most cases across the government, stakeholders consistently characterize the authorization process as slow, despite the lack of firm baselines on the length of time it takes for a system to receive an ATO. Stakeholder focus is commonly on documentation and "counting controls" rather than implementing the risk-based approach to security that the RMF prescribes. The time required to authorize differs based on the size, complexity, type of information or data, the impact to business or mission operations, and level of security required of the information system. Some processes can be accelerated by automation, but there is no way to automate the entire RMF process. The two main barriers to rapid ATOs for cloud systems are that the RMF process is inconsistently administered and that it is highly reliant on human involvement.

The current RMF process is optimized for waterfall software development and approves a product at a point in time – which is not conducive to an iterative approach to software development. Software is often developed over an extended period, then "put on hold" to undergo a lengthy ATO process before it is deployed to the stakeholder. The time between the approval of a requirement and the product's deployment often means that the technology is obsolete before it can be released into production. The government is missing out on functionality, performance, and speed to market by using traditional, restrictive software development processes and security authorizations.

The government is rarely able to implement real innovations at the current slow speed of authorizations. This prevents adoption and implementation of cutting-edge technologies such as cloud computing, and dilutes the benefits of a DevOps pipeline. Conversely, the commercial world has been modifying and enhancing the SDLC beginning with the publication of the Agile Manifesto in 2001. The Manifesto identifies four values:

- Individuals and interactions over processes and tools
- Working software over comprehensive documentation
- Customer collaboration over contract negotiation
- Responding to change over following a plan

As a result, a paradigm shift is taking place and pockets of the government are leading the way into the future. Programs such as Kessel Run are leading the transformation of the existing method to the future method, as shown in ***Exhibit 1.***



Current Method	Future Method
Product-based approvals (re-approve each time I change or upgrade)	**Process-based approvals** (approve the secure process/auto-approve products)
Equally slow for most products	**Fast-track for top programs** that follow best practices
Optimized for Waterfall SDLC	**Optimized for Agile DevSecOps**
Focus on Documentation	**Focus on security and fixing vulnerabilities**
Checks past behavior	**Drives future behavior**

Exhibit 1:The paradigm shift to DevOps development methodologies requires a change in security authorizations.

The traditional RMF process, and its predecessor DIACAP, have had a stranglehold on software development in the DoD, but modernizing the approach is not impossible. A combination of the clear and measurable benefits to implementing DevOps within the DoD, widespread stakeholder-holder driven desires for faster timelines and better code, and clear steps and processes to enable implementation mean that realizing DevOps in DoD software factories is entirely achievable, as explained in ***Exhibit 2***.

Objectives – Desired outcomes	Drivers – Necessity for change	Process - How it's done	Barriers – Obstacles to success
▪ **Create Intrinsic Software Security.** Build security into everything that is done. ▪ **Enable Continuous Updates and Delivery.** Provide code updates and deliveries as they are available. ▪ **Reduce Time-to-Authorize.** No need to perform an entire ATO assessment after every change if core components of the system have not changed. ▪ **Improve Security Posture.** Improve the overall security posture through the use of best-in-breed tools, continuous monitoring, and objective testing. ▪ **Improve Speed-to-Delivery.** New code updates should be deployed to production as soon as possible to improve software systems for our users. ▪ **Embrace Secure DevOps.** Fully realize the benefits of DevOps. ▪ **Minimize Opportunities for Human Risks.** Automating processes and activities to reduce the potential errors due to non-compliance or human error.	▪ **Legacy System Risk.** No way to ensure that legacy ATOs are being maintained – creating technical debt and potentially insecure applications. ▪ **Software Speed-to-Delivery.** Software deliveries lag mission speed due to security reviews. ▪ **Current RMF Implementation.** Current RMF process is checkbox-driven and doesn't test the entire system (application, middleware, platform, infrastructure). ▪ **Security Across SDLC.** Need to bake security into the SDLC by including developers and operations personnel more heavily in securing the system as they know it best.	▪ **Continuous Security Scanning.** Scanning will find many bugs as they arise and enable quicker vulnerability remediation. ▪ **Paired Programming.** One developer continually monitors the code input by the other. ▪ **NIST Control Traceability.** NIST control traceability to automated tests provides assurances to AO that system is meeting the security requirements of the organization. ▪ **Version Control System.** Version control system maintains all changes and releases, easing long-term maintenance tasks. ▪ **Strict Quality Checks.** Code quality checks drive accountability down to the development team and create more maintainable software. ▪ **Penetration Testing.** Independent Pen Testing validates appropriate application of automated testing tools and security control selection.	▪ **Workstreams with Varied Maturity Levels.** Every organization has different levels of maturity when it comes to tools, processes, and personnel. ▪ **One-size-fits-all Mentality.** The security controls applied must be identified early in the design process and in conjunction with the AO – they should be tailored appropriately to the system's security requirements. ▪ **Bureaucratic Inertia.** The "way we've always done it" mentality can be pervasive in hindering progression and change. ▪ **Resource Availability.** Resources in the form of budget, talent, and access to tools or technologies can be constrained across organizations. ▪ **Ineffective Organizational Culture Change Management.** Bringing major changes to a risk-averse community is commonly difficult. ▪ **Non-compliance.** The likelihood that individuals will not follow policy or standard operating procedures within a large organization.

Exhibit 2: Objectives, drivers, process, and barriers to adopting DevOps as part of the Software Factory.

2.2 Changes to Stakeholder Collaboration and Responsibilities

The Continuous ATO does not reduce or exclude the fundamental roles and responsibilities defined in NIST and DoD security publications. It also does not change the stakeholders involved in the ATO process. Rather, the Continuous ATO modifies the way responsibilities are implemented and the interactions between stakeholders while streamlining security best practices. If the Continuous-ATO-driven SDLC integrates all of the security best practices and controls as outlined in this document, most stakeholders' security responsibilities are simplified to a verification that the guidance is being followed.

2.3 Relevant Polices, Instructions, Regulations and Legislation

For years, cybersecurity has been a focal point for Congress and Department heads, who have led extensive efforts to lay out the activities required to secure the nation's information systems. The NIST, in coordination with DoD and other agencies, defined security controls and the RMF process. The NIST 800-53, *Security and Privacy Controls for Federal Information Systems and Organizations*, is applicable for all RMF activities laid out in DoDI 8510.01. The additional applicable policies for the Continuous ATO are:

- NIST Special Publication 800-37, rev 2, *Risk Management Framework for Information Systems and Organizations*, September 2017
- NIST *Supplemental Guidance on Ongoing Authorization*, June 2014

3.0 Constructing a Software Factory to Realize Continuous ATO

A Software Factory must be properly equipped and tuned to bake security into the entire SDLC and to automate testing and scanning. This eliminates the need for a separate and distinct security phase after development. The Software Factory certification model is optimized for DevOps software development and focused on certifying the process (within the Factory) by which applications are created. The Software Factory includes all processes, frameworks, and tools necessary to ensure security and speed to market are maximized. The five building blocks required in creating a Software Factory are industry-leading personnel, best-in-breed technology, DevOps best practices, secure DevOps, and continuous improvement and validation. Each of those components has key features required to build the factory, which are highlighted in ***Exhibit 3*** and explained in the following section.

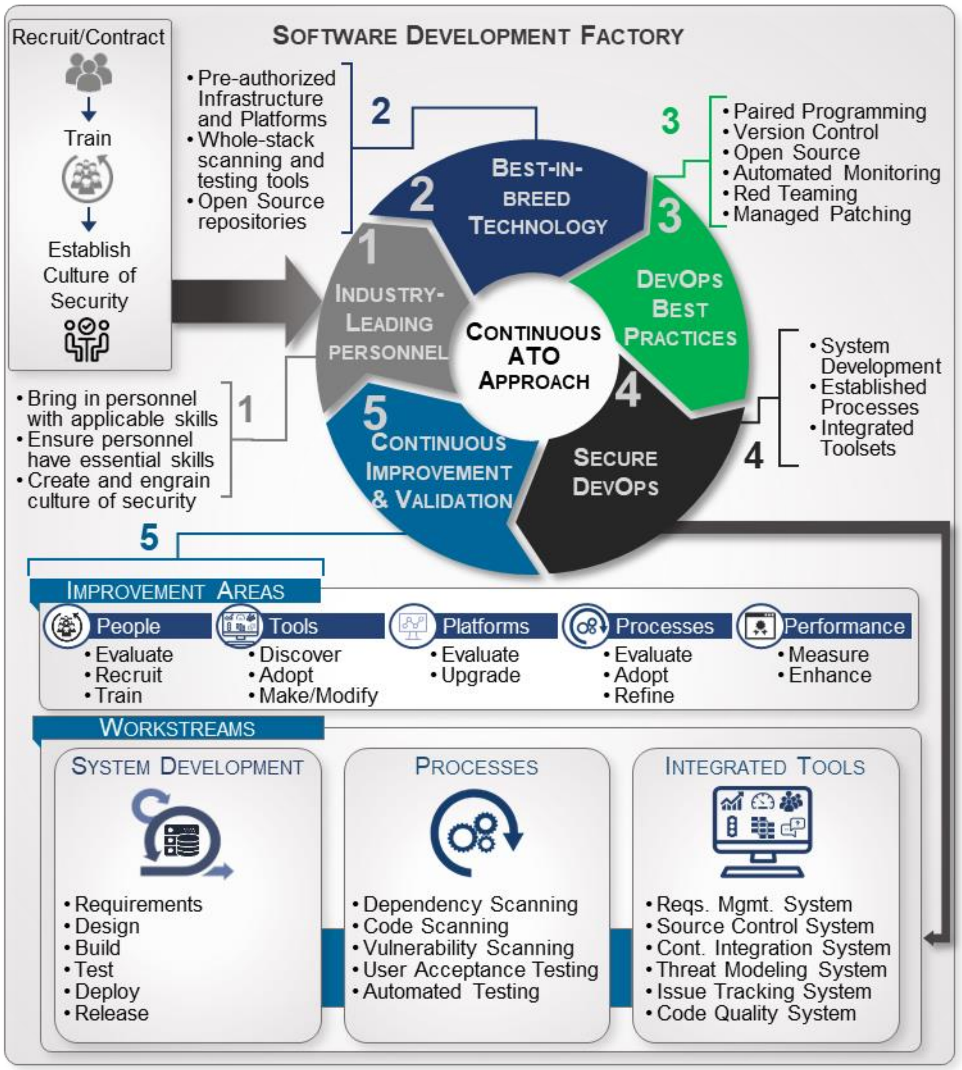


Exhibit 3: Building a Software Factory requires five components to work together for intrinsic security.

Creating the Software Factory requires more than implementing better processes: it requires a new mindset to change the way that government develops and secures software. The Software Factory shifts thinking from a traditional waterfall approach to a DevOps approach that trusts the people who develop the software to use the best tools and practices to create quality software. The Software Factory shifts the authorization process from a retrospective exercise in documentation to incentivize using inherited controls, building in security, and fixing vulnerabilities.

3.1 Setting the Foundation of Security with Industry-Leading Personnel

3.1.1 Hire or Acquire Talented Personnel to Develop and Secure Applications

Meeting the government's development, quality, and security needs requires both recruiting and grooming internal personnel and bringing in outside talent to fill immediate gaps. Gathering and training existing personnel will help to grow the organization's talent and build continuity, while recruiting and partnering with industry leaders and subject matter experts will enable the organization to rapidly deploy proven talent who can serve as project leads and technical guides.

This document does not recommend any particular choice in industry partner, but organizations should seek out the following qualifications, and other comparable alternatives, in their search:

- Applicable Degrees
  - Computer Science
  - Computer Engineering
  - Software Engineering
  - Cybersecurity
- Preferred Certifications
  - Certified Information Systems Security Professional (CISSP)
  - CompTIA Security+
  - Certified Secure Software Lifecycle Professional (CSSLP)
  - Offensive Security Certified Professional (OSCP)
  - Offensive Security Certified Expert (OSCE)
  - Certified Ethical Hacker (CEH)

3.1.2 Training Personnel to Empower Maximum Capabilities

Any new initiative that significantly changes a workflow requires a commensurate increase in training, especially when existing personnel will remain as the backbone of the project team. Traditional education reimbursement and an internal classroom training are necessary but not sufficient for success – less traditional training opportunities, focused specifically on DevOps practices, are equally important. One example is training personnel on paired programming, a technique in which two-programmer teams work side-by-side as a code-writing “driver” and a code reviewing “navigator.” This programming style may be foreign or uncomfortable for some personnel at first, which can be minimized by targeted training.

While new recruits and contractors are hired frequently because of the depth of their training in relevant disciplines and technologies, it is important to include them in these efforts where feasible. Hands-on collaboration in the training environment gives existing personnel another source of experience to draw from while promoting synergy among the team.

3.1.3 Establishing A Culture of Cybersecurity Throughout the Organization

The most common security risk in any organization is the human element. No amount of code scanning, vulnerability testing, or adherence to prescribed security frameworks will prevent an incident, such as spear phishing, that is caused by a lack of security awareness. Whether Software Factory personnel are recruited or trained, it is up to the organization to mitigate this risk by establishing a culture of security through awareness training, the implementation of strict policies, and strict accountability for adherence to them.

There is no one-size-fits-all solution to establish an organizational security culture, but it must be grounded in key principles. Common practices in security-minded organizations include leading from the top, hosting security awareness training, and gamification to improve engagement. Success requires a conscious, structured effort to involve discussions around security considerations in every public forum possible. Meetings that regularly happen throughout the SDLC should work to explicitly tie security into every applicable topic.

3.2 Using Best-in-Breed Technology to Enable Continuous ATO

The most basic element in building a successful Software Factory is arming developers with the infrastructure, platform, and tools that best enable them to accomplish the program's development goals. Traditionally, this would involve purchasing and managing physical components ranging from datacenters to software licenses, as well as hiring or training personnel to deploy and maintain them, but this resource intensive process is no longer the only option. With the advent of cloud managed infrastructure and platform services, organizations can simplify their management and personnel footprints by outsourcing these components to third-party providers. Multiple government programs have established security authorization frameworks for these Cloud Service Providers (CSP), enabling cloud-based options to reduce a significant portion of security-centered considerations and overhead while increasing control inheritance.

3.2.1 Categorize System and Select Security Controls to Maximize Inheritance

Setting parameters and security requirements for the Software Factory is critical to establishing a secure environment for future development work. Planning ahead and determining exactly which security specifications are required determines the types of technologies and tools required for an authorization. These security requirements are established by categorizing the type of data that will be transmitted through the applications that are being developed. The data categorization influences the type of controls required in order to obtain an authorization, and should set the foundation for what is needed from the Infrastructure as a Service (IaaS) and Platform as a Service (PaaS) to house the Software Factory. Ideally, the majority of the controls are inherited from preapproved IaaS and PaaS. The AO should be involved in these decisions as they will ultimately sign off on the ATO when the Software Factory is operational. These activities are discussed in greater detail in ***4.1.1.1 Categorize System and Select Security Controls for Planning and Development*** as part of the phased approach to the Continuous ATO.

3.2.2 Secure Infrastructure as the Foundation of Security

Infrastructure refers to all physical equipment and networking technology connecting the platform and software to the end-user. As the backbone of any software development project, it is critical that the infrastructure meets all relevant capability, availability, and security requirements. Any failure at the networking or storage level of the technology stack can neutralize all components on subsequent levels. These components can either be built in-house or outsourced to a cloud environment, which is usually more favorable for government projects that plan to achieve Continuous ATO.

3.2.2.1 Issues With a Do-It-Yourself Approach

Building a dedicated data center is an option in the right circumstances – for example, some organizations own existing enclaves that meet specifications and have existing personnel to manage them. That said, these are almost the only circumstances in which building out dedicated infrastructure makes sense as part of Software Factory dedicated to Continuous ATO. Without extensive existing hardware and personnel assets, the time required to prepare and deploy a self-built solution could introduce enough slowdown into the project's lifecycle to make it incongruent with a goal of Continuous ATO.

3.2.2.2 Outsourcing IaaS to a CSP with a Pre-authorized Solution

While outsourcing infrastructure was not historically viable for sensitive government or DoD projects, Federal Risk and Authorization Management Program (FedRAMP) and Defense Information Systems Agency's (DISA) efforts to provide a standardized approach to cloud-based security have enabled commercial CSPs to provide IaaS options that meet NIST and RMF guidelines.

FedRAMP's Joint Authorization Board (JAB) has designated approved CSPs with different DoD Impact Level (IL) ratings based on the sensitivity of information they are approved to contain. A full list of CSPs can found at the *DISA DoD Cloud Service Catalog*. This program enables the commercial-off-the-shelf (COTS) availability of infrastructure that ideally suits a Continuous ATO approach, as using IL4 and above rated providers automatically builds in a majority of RMF controls required at the infrastructure level. This process is detailed at *https://www.fedramp.gov/jab-authorization/.*

Some DoD projects employing Continuous ATO require a solution rated at a minimum of IL4, which is designated for mission data or other information that requires protection from unauthorized disclosure as established by Executive Order 13556. At the time of this publication, only the following IaaS solutions meet IL4 and IL5 criteria:

- Amazon Web Service (AWS) GovCloud
- Microsoft Azure
- MilCloud 2.0

3.2.3 Implementing a Secure Development Platform

Platform defines the environment in which users operate and develop, to include the operating system, programming languages, tools, and libraries. Depending on the scope of the project, the experience of the developers, and to some extent, personal preference, this highly variable element of the process can be tailored to best suit the project. However, the platform level of the stack is a growing target for malicious actors as security holes here can often enable a data breach. Similar to infrastructure selection, mission planners have a variety of choices in how to implement this layer of the stack:

- Develop or use prebuilt platform on top of existing infrastructure
- Integrate prebuilt platform into an IaaS solution
- Purchase a PaaS solution to implement on top of existing infrastructure
- Purchase an all-inclusive solution that provides both IaaS and PaaS

As with selecting an infrastructure solution, the cloud-based solution likely gets us closest to our Continuous ATO goal.

3.2.3.1 Issues With a "Do It Yourself" Approach

Depending on the project or circumstance, developing and operating on a self-built platform could have some advantages, like architectural flexibility and uptime. The risk with this approach is, much like with building your own infrastructure, it can significantly delay Continuous ATO between the development time and necessity for additional RMF involvement, as the AO must accredit the entire system and cannot rely on a previous agencies' accreditations of a pre-existing system as this one is unique. While a homemade platform could in theory be implemented into Continuous ATO, it is not the recommended approach.

3.2.3.2 Outsource to a Pre-authorized Secure Platform

Using a PaaS allows for a third party to provide operating systems, software tools, and essentially all other backbone elements of the development environment. This method aligns with a Continuous ATO approach in multiple ways:

- Many intrinsic security measures are handled autonomously at the vendor level
- We can rapidly clone new, identical development and production environments for ease of authorization
- Prebuilt authorization and accreditation (A&A) documents usually exist from previous A&A efforts and these can be provided by the vendor
- At IL4 and above, a majority of required RMF controls are built-in and inherited, as detailed at *https://www.fedramp.gov/jab-authorization/*

Based on both organizational and project needs, there are a few broad categories of PaaS offerings to consider:

- **Structured Platforms** simplify the platform model by providing the majority of tools, often open source; this model is primarily targeted at enterprise customers. Pivotal Cloud Foundry and IBM Bluemix are vendor examples of this type of offering
- **Composable Platforms** carry a flexible set of open source tool offerings while being packaged more tightly as a set of services. RedHat OpenShift and Docker Datacenter are vendor examples of this type of offering
- **Unstructured Platforms** are a more basic cloud service offering, ideal for an organization that plans to integrate and use more custom tools. Mesosphere and Atlas are vendor examples of this type of offering

These platforms also offer a significant range of capability depending on the size of the project, the experience of the team, and the budget of the organization. Pivotal's CloudFoundry is a more full-service platform and was used by Kessel Run, whereas options like OpenShift and Kubernetes offer more bare-bones solutions. This gives a great deal of flexibility to organizations looking to more finely tailor their platform solution to their needs.

With PaaS offerings as well, the FedRAMP program's JAB has designated approved CSPs with different DoD IL ratings based on the sensitivity of information they are approved to contain. A full list of CSPs can found at the *DISA DoD Cloud Service Catalog*.

Exhibit 4 visualizes the significant reduction in scope and complexity that a fully outsourced infrastructure and platform solution can offer a Software Factory.

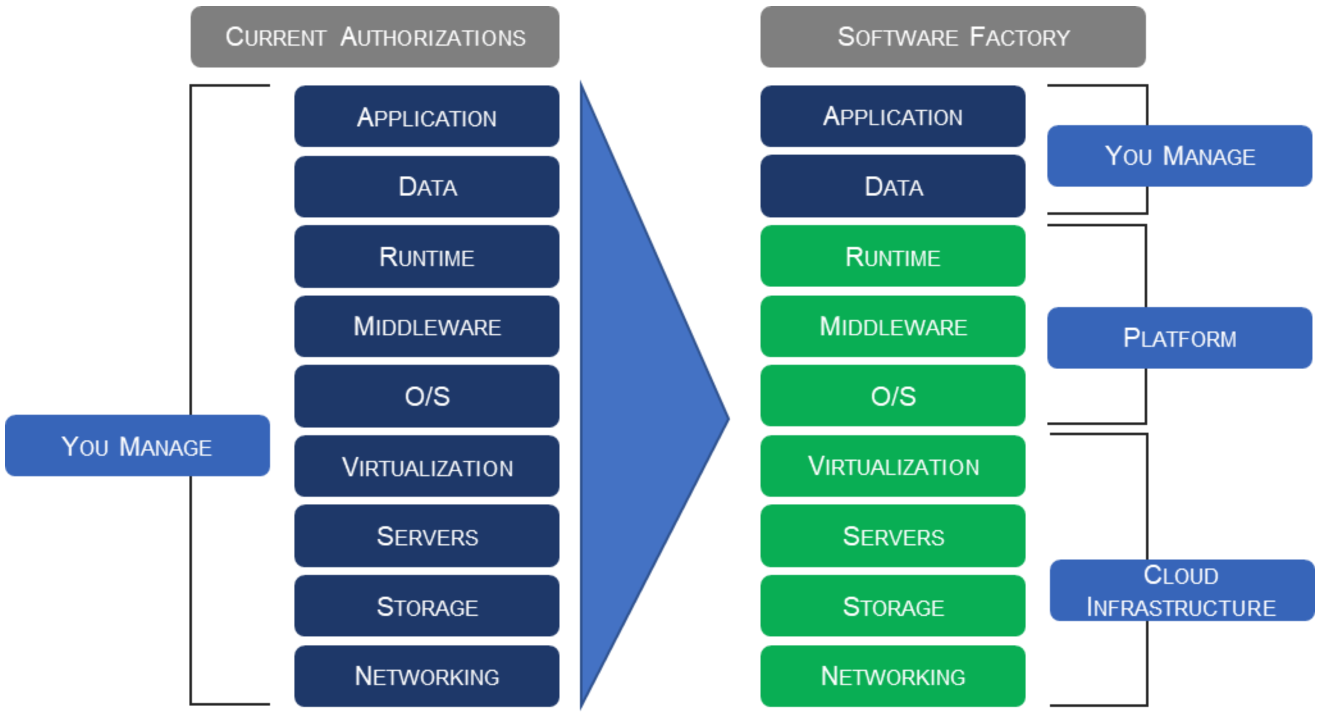


Exhibit 4: Using pre-authorized IaaS and PaaS enables maximum control inheritance for the ATO.

When the normal components that comprise a project's infrastructure and platform are no longer in scope, the authorization process is simplified to the applications and data layers.

3.2.4 Scanning and Monitoring Tools for Secure DevOps

After selecting the infrastructure and platform solutions, the last component to identify is the suite of tools the developers will use. While DevOps software development methodologies and the broad categories of tools they require are fairly well defined, the market includes a broad range of individual tools and ways to integrate and optimize them. This flexibility enables organizations to select the solutions that best fit both their project requirements and their personnel.

The Software Factory implements development best practices to proactively automate security checks and maintain continuous monitoring and fixes. This enables the development teams to fully embrace DevOps while understanding that security is an integral workstream from inception throughout the lifecycle. ***Exhibit 5*** includes examples of tools used for different aspects of the SDLC and ***Exhibit 8*** in ***Section 3.3.3*** shows how these tools can be implemented.

Task Area	Tool Description	Options
Version Control System	Helps software teams manage source code changes through modification tracking	GitLab, Mercurial SCM, Apache SVN, Bazaar, BitBucket
Continuous Integration System	Merges working copies of software on an automated and frequent basis to allow for early detection of integration errors	Concourse, Bamboo, Jenkins, Travis
Build Automation Tools	Automates software building process by compiling source code into binary, packing it, and running automated tests	Ant, Bundler, Maven, MSBuild, Waf
Source Code Analysis	Analyzes source code to flag programming errors, bugs, stylistic errors, and suspicious constructs	Lint, SonarQube, Veracode, PMD
Code Quality Inspection	Exposes additional areas of code that can be improved in terms of quality, readability and future usability	SonarQube, Codacy, Fortify, Coverity, CxSAST
Configuration Management and Deployment	Tracks and controls software changes and enables rapid application deployment	Ansible, Puppet, Chef, ControlTier, Run Deck, LiveRebel, CFEngine
Testing Automation Frameworks	Sets the rules of automation of a specific product, integrating function libraries, test data sources and object details	Selenium, Sauce Labs, JUnit, SureAssert, Serenity, Cucumber
Stress Testing	Pushes applications to their limits during the testing phase to find and fix errors that arise from heavy use	webLOAD, JMeter, LoadRunner
OWASP Dependency Check	Identifies known or publicly disclosed vulnerabilities in software; standalone or as a plugin for previously identified software	OWASP Dependency Check; plugins for *Ant*, *Maven*, *Jenkins*
Application Security Requirements and Threat Management (ASRTM) Platform	Documents the system's agreed upon security requirements, the security features' implementation details and schedule, and the resources required for assessment	SD Elements
Vulnerability Scans	Uncovers security exposures not covered under the scope of OWASP checks	Nessus, Nmap, Fortify, BDD-Security, Mittn, PortSwigger, AppDetective
Intermediate Code Repository	Manages and stores compiled code, allows for sharing with other developers	Nexus, PyPI, Docker Hub, Archiva, Pulp, Yarn

Exhibit 5: Automated tools consistently scan the Software Factory and all applications throughout the lifecycle.

3.3 DevOps Best Practices for Creating Secure Applications

DevOps is a workflow model focused on accelerating software delivery to enable a cycle of Continuous Delivery that encourages collaboration between the often-siloed roles of development, IT, security, and operations. The concept of Continuous ATO draws heavily on DevOps practices, and it is important to familiarize Software Factory teams with at least a few foundational DevOps practices. From there, a program can tailor these general practices to fit the process-flow of a Software Factory. At its core, DevOps is a means for developers to more efficiently and continuously deploy production code.

3.3.1 Overview of the DevOps Lifecycle

The DevOps SDLC enables incremental gain at every step of the development process. In an ideal implementation, the activities in each phase are conducted in parallel and the transition of one phase to the next is seamless. This rapid process transition minimizes loss of project inertia and encourages accountability through a consistent cycle of work.

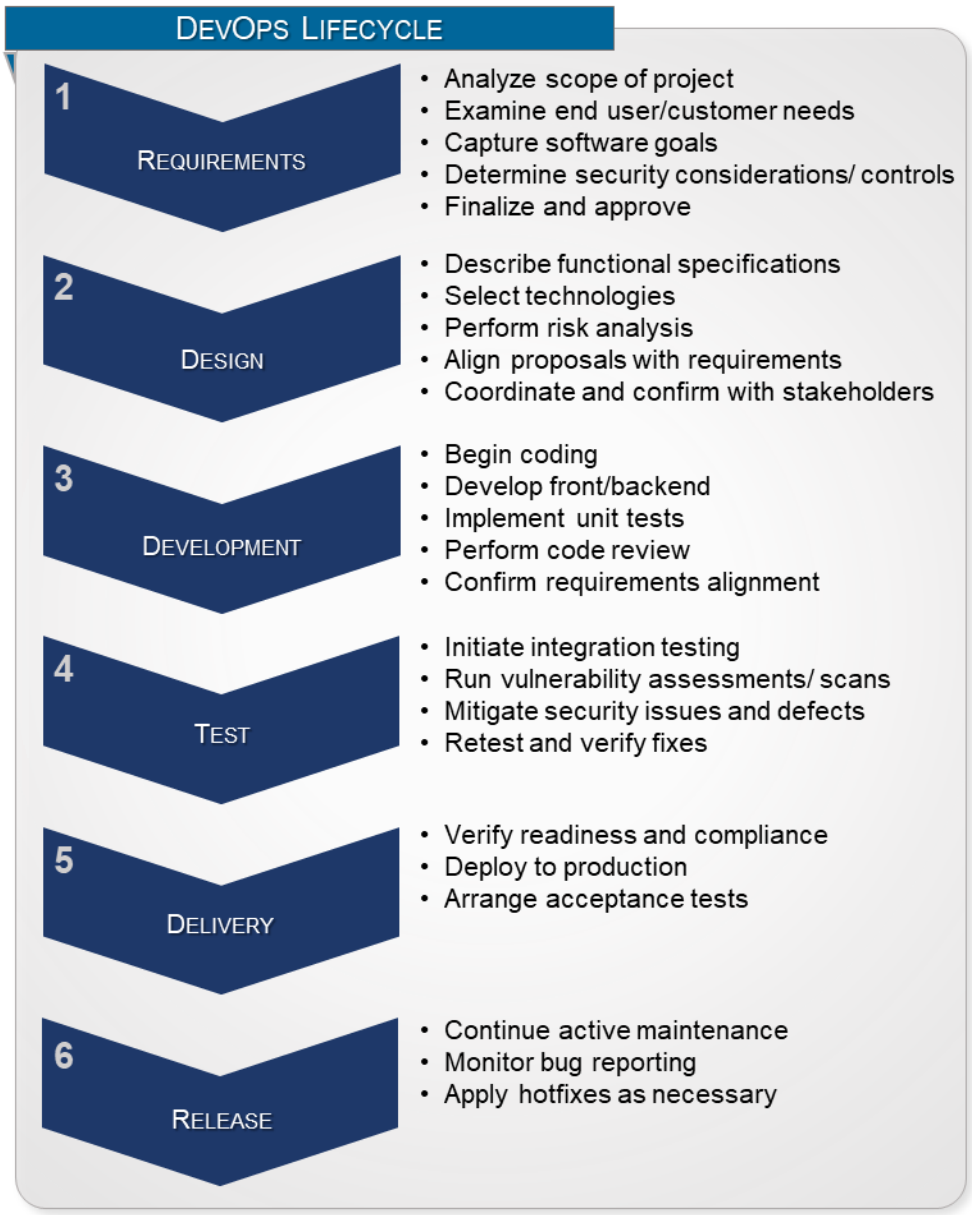

DEVOPS LIFECYCLE
1
REQUIREMENTS
• Analyze scope of project
• Examine end user/customer needs
• Capture software goals
• Determine security considerations/ controls
• Finalize and approve
2
DESIGN
• Describe functional specifications
• Select technologies
• Perform risk analysis
• Align proposals with requirements
• Coordinate and confirm with stakeholders
3
DEVELOPMENT
• Begin coding
• Develop front/backend
• Implement unit tests
• Perform code review
• Confirm requirements alignment
4
TEST
• Initiate integration testing
• Run vulnerability assessments/ scans
• Mitigate security issues and defects
• Retest and verify fixes
5
DELIVERY
• Verify readiness and compliance
• Deploy to production
• Arrange acceptance tests
6
RELEASE
• Continue active maintenance
• Monitor bug reporting
• Apply hotfixes as necessary

Exhibit 6: DevOps enables a phased approach to Continuous ATO.

3.3.2 Implementing Secure DevOps within the Software Factory

While DevOps inherently brings security into greater focus, the subdiscipline of Secure DevOps, usually abbreviated to SecDevOps, integrates an additional set of best practices into the DevOps SDLC. While not mandatory nor all-inclusive, the following are examples of additional security practices that a SecDevOps environment will implement.

Best Practice	Practice Description	Benefits
Requirements		
Assess-only security assessments	Leveraging existing security assessments and authorizations from other organizations	Reduces redundant scanning and authorization efforts in cases where software or components are identical or inherited
Security controls align with policy	Using security control software (currently only SD Elements) to generate security controls aligning with organizational (NIST) policies	Conjoins the processes of checking off security and policy controls, enabling a significantly more rapid potential authorization
Simplify problem resolution	Creating a clear process for resolving issues that are identified	A defined process for reaching problem resolution helps to combat bureaucratic slowdown
Two-factor authentication	Requiring an additional form of verification during account logins	Increases the difficulty of credential or account theft
Design		
Data protection strategy	Implementing automated or manual data backups	Protects data and development progress in the event of availability failures
Extensive Open Source use	Making use of Open Source libraries for multiple components of the Software Factory process	Adds additional security, as widely used, Open Source libraries receive security updates frequently
User Access Management	Managing user and developer access to only allow usage of needed tools or features	Minimizes vulnerable space to manage on a per-user basis; simplifies development landscape
Development		
Source Code Control System implementation	Using software GitLab to securely version control and share code among development team	Simplifies branch management, easing bug fixes while developers to continue on the main trunk
Continual access to documentation	Ensuring storage of and perpetual access to pertinent documentation (such as NIST 800.53) in repositories	Increases likelihood of strict adherence to organizational policy; increased policy clarity; reduces time spent seeking documentation
Test		
Automated scanning and reporting	Using tools to automate both the code and vulnerability scanning processes	Increases project agility by removing the time spent on more rudimentary, surface level checks
Continuous monitoring plan	Maintaining ongoing awareness of current security vulnerabilities and threats	Supports organizational risk management decisions; helps maintain risk tolerance accuracy; ensures effectiveness of controls

Delivery		
Red teaming/"Hack the X"	Creating an environment for experts to attempt red team style penetration testing methods	Helps to uncover vulnerabilities and security holes that might be missed by normal penetration testing practices; more closely emulates nation state threats
Release		
Bug bounty programs	Advertising an incentive for private disclosures of software vulnerabilities found	Helps to identify security gaps in currently deployed software versions
Random penetration testing	Conducting off-schedule penetration tests through an outside vendor	Brings a new set of eyes to the problem, helps detect dormant issues in between scheduled testing
Rapid vulnerability response	Setting timelines for a fix after vulnerabilities are detected; i.e., 10 days for critical, next sprint for moderate	Reduces risk of forgetting to patch any vulnerabilities found
Managed patching services	Implementing patch management software to automate software updates	Automates the pushing of new patches to remove or lessen risk of end-user negligence

Exhibit 7: DevOps best practices are implemented throughout the SDLC to ensure security.

3.3.3 Implementing Scanning and Monitoring Tools into the Secure DevOps Lifecycle

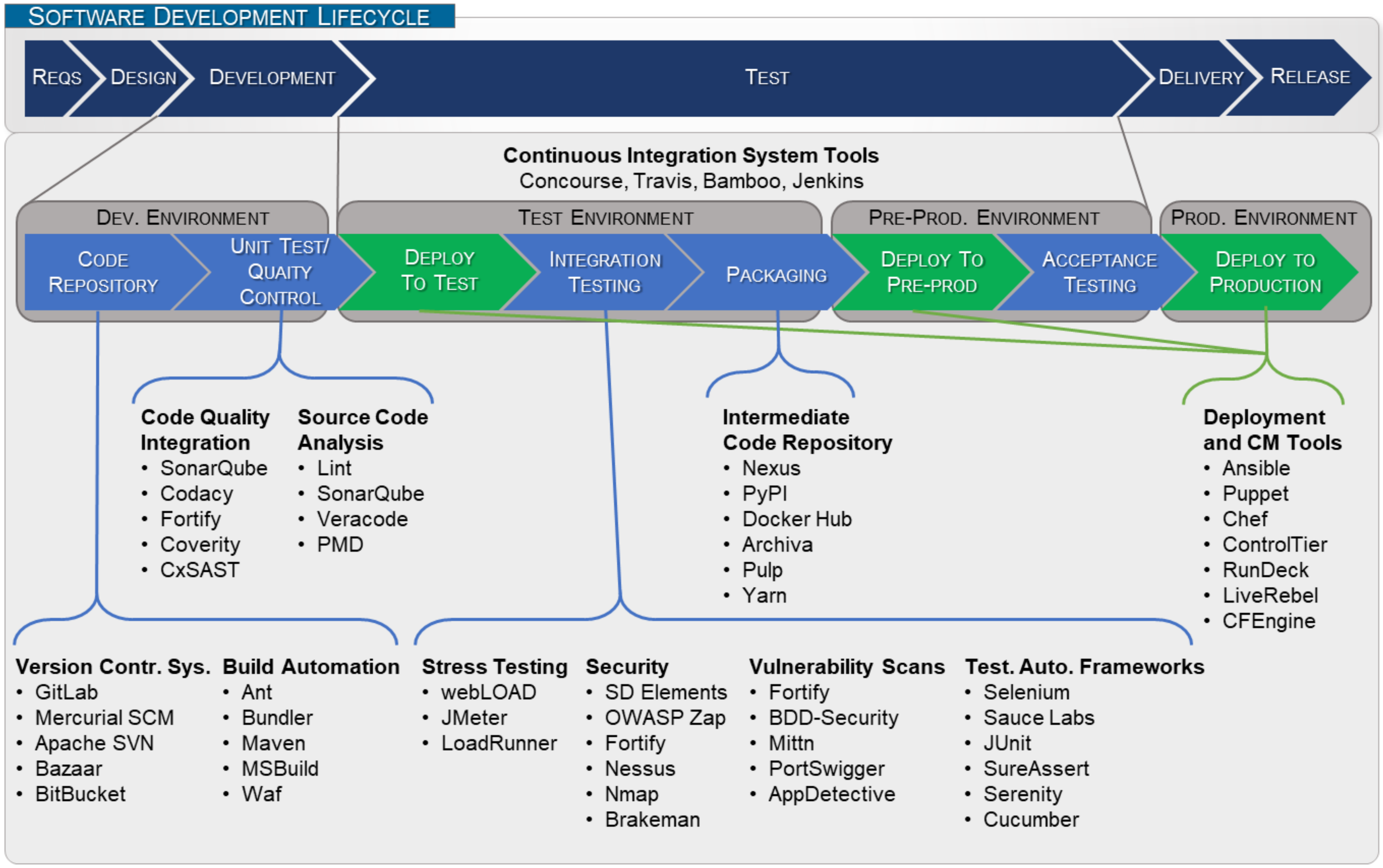
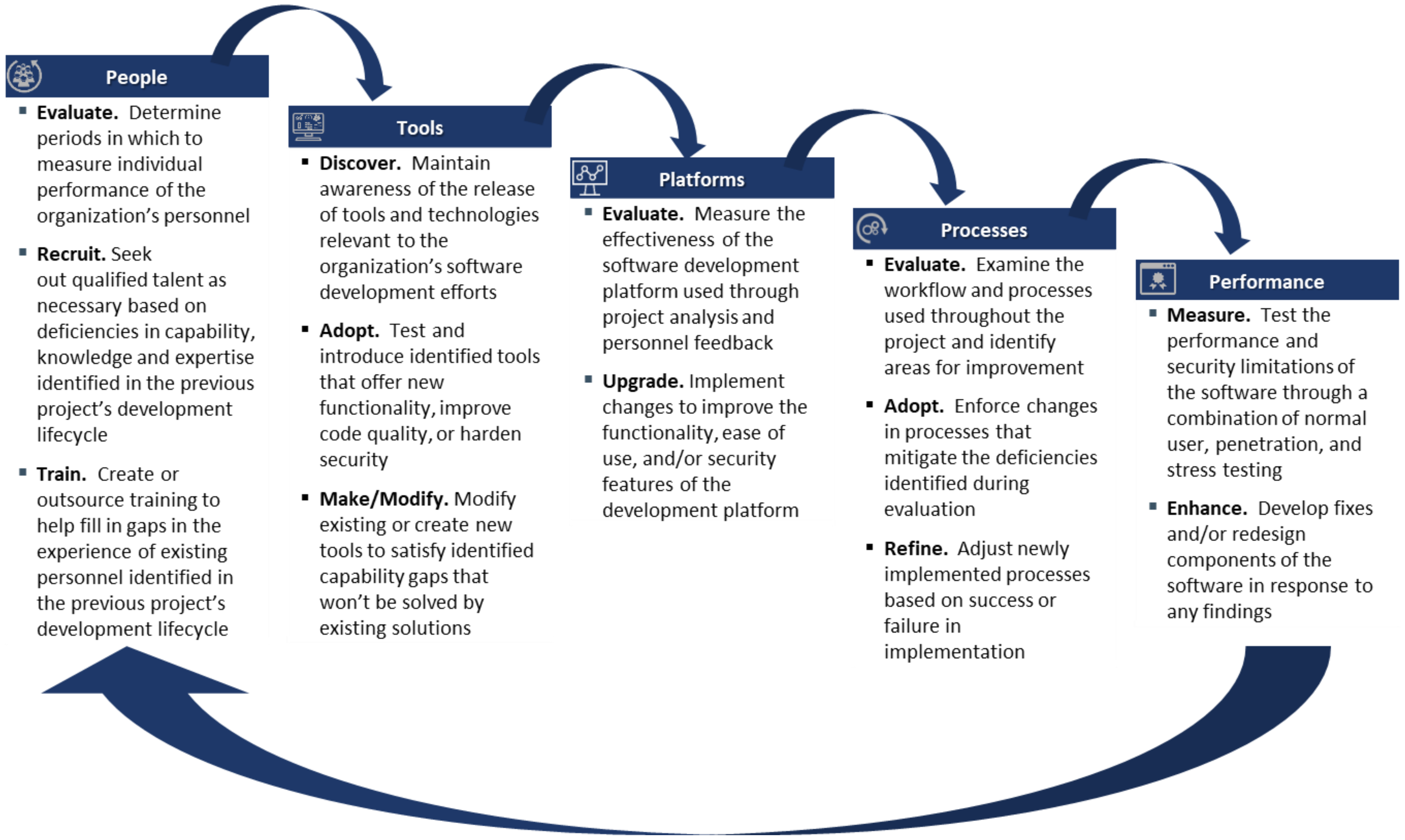


Exhibit 8: Scanning and monitoring tools are implemented in each phase of the SDLC.

3.4 Ensuring Security through Continuous Improvement and Validation



People

- **Evaluate.** Determine periods in which to measure individual performance of the organization's personnel
- **Recruit.** Seek out qualified talent as necessary based on deficiencies in capability, knowledge and expertise identified in the previous project's development lifecycle
- **Train.** Create or outsource training to help fill in gaps in the experience of existing personnel identified in the previous project's development lifecycle

Tools

- **Discover.** Maintain awareness of the release of tools and technologies relevant to the organization's software development efforts
- **Adopt.** Test and introduce identified tools that offer new functionality, improve code quality, or harden security
- **Make/Modify.** Modify existing or create new tools to satisfy identified capability gaps that won't be solved by existing solutions

Platforms

- **Evaluate.** Measure the effectiveness of the software development platform used through project analysis and personnel feedback
- **Upgrade.** Implement changes to improve the functionality, ease of use, and/or security features of the development platform

Processes

- **Evaluate.** Examine the workflow and processes used throughout the project and identify areas for improvement
- **Adopt.** Enforce changes in processes that mitigate the deficiencies identified during evaluation
- **Refine.** Adjust newly implemented processes based on success or failure in implementation

Performance

- **Measure.** Test the performance and security limitations of the software through a combination of normal user, penetration, and stress testing
- **Enhance.** Develop fixes and/or redesign components of the software in response to any findings

Exhibit 9: Continuous improvement is critical for every component of the Software Factory.

4.0 Phased Approach to Achieving a Continuous ATO

The Continuous ATO process meets all traditional RMF requirements but differs in implementation. The authorization of controls, updates, and new releases would traditionally be conducted as a separate process, but Continuous ATO makes these checks an ongoing process throughout all phases of the SDLC and all components of the Software Factory. While not a guarantee of a faster authorization, it streamlines what was traditionally a disjointed process. Continuous ATO still adheres to all general RMF guidelines, but enables the Software Factory to tailor and integrate those processes. As discussed earlier in the Playbook, the Continuous ATO shifts the paradigm from assessing only the final product to assessing the process used to develop the product. Based on the NIST *Supplemental Guidance on Ongoing Authorization*, the Ongoing ATO is as follows:

> *Ongoing authorization is part of RMF Step 5, the Authorize step, and is dependent on the organization's Information Security Continuous Monitoring (ISCM) strategy and program (summarized in Section 2.3) which is implemented as part of RMF Step 6, the Monitor step. OA is fundamentally related to the ongoing understanding and ongoing acceptance of information security risk. It is also closely related to the dynamic, organization-wide risk management process that develops a more refined and articulated situational awareness of an organization's security and risk posture based on the ongoing assessment, response to, and monitoring of information security risk.*

The NIST *Supplemental Guidance on Ongoing Authorization* establishes the Ongoing Authorization as having three steps: Initial Authorization, Ongoing Authorization, and Reauthorization. The Continuous ATO process adds a phase prior to the Initial Authorization for building the Software Factory and a phase at the end for continuous improvement and validation of the Software Factory and all of its products. ***Exhibit 10*** details the activities required for each of the five phases of the Continuous ATO. Each phase will be defined in greater detail in subsequent sections along with a crosswalk of the phases to the traditional RMF process.

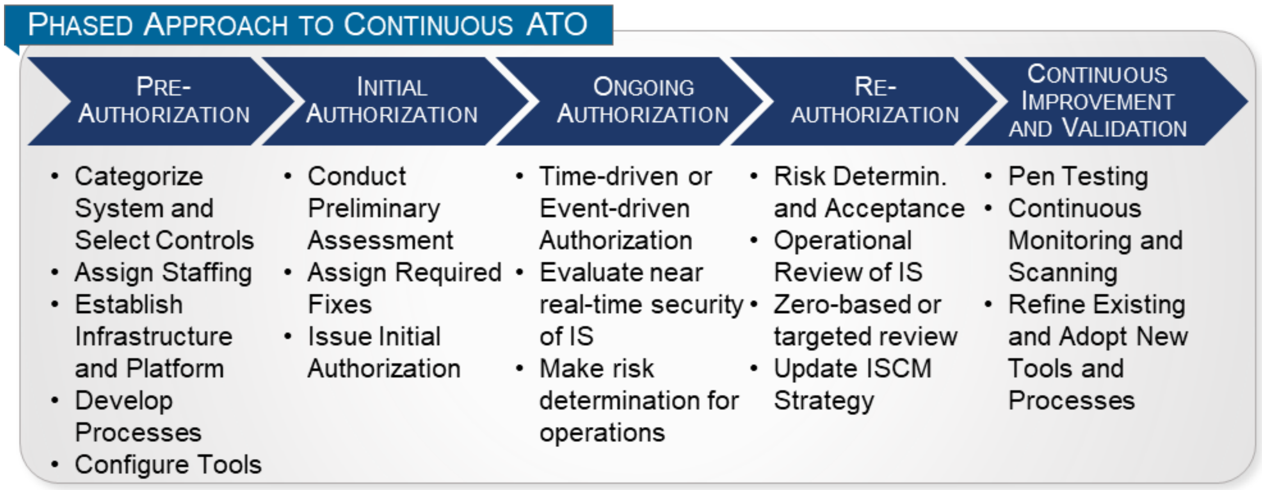

PHASED APPROACH TO CONTINUOUS ATO
PRE-AUTHORIZATION
• Categorize System and Select Controls
• Assign Staffing
• Establish Infrastructure and Platform
• Develop Processes
• Configure Tools
INITIAL AUTHORIZATION
• Conduct Preliminary Assessment
• Assign Required Fixes
• Issue Initial Authorization
ONGOING AUTHORIZATION
• Time-driven or Event-driven Authorization
• Evaluate near real-time security of IS
• Make risk determination for operations
RE-AUTHORIZATION
• Risk Determin. and Acceptance
• Operational Review of IS
• Zero-based or targeted review
• Update ISCM Strategy
CONTINUOUS IMPROVEMENT AND VALIDATION
• Pen Testing
• Continuous Monitoring and Scanning
• Refine Existing and Adopt New Tools and Processes

Exhibit 10: The Continuous ATO follows the five-phased set forth in the NIST Supplemental Guidance on Ongoing Authorization.

4.1 Pre-Authorization – Building a Secure Software Factory

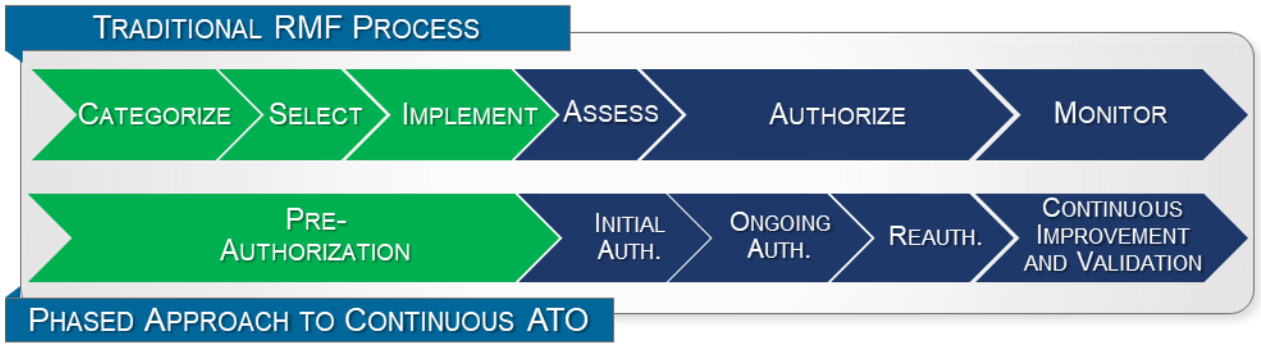


Exhibit 11: The Pre-authorization phase incorporates the first three steps of the RMF process.

4.1.1 Required Tasks for Pre-authorization Phase

Pre-authorization within Continuous ATO consists largely of setting the stage prior to the start of building the Software Factory. This includes determining the system's classification and applicable security controls, and selecting and gathering all Software Factory components to prepare the project for a transition to Initial Authorization.

4.1.1.1 Categorize System and Select Security Controls for Planning and Development

These activities align with the first two phases of the RMF process and are critical to planning and designing a secure Software Factory. Selecting the proper data categorization for the systems that will be developed within the Software Factory determines the level of security required for the IaaS and PaaS layers and will determine the controls that will be inherited from other authorizations. The categorization is based on the Federal Information Processing Standard (FIPS) 199 classifications of confidentiality, integrity, and availability on a low, moderate, or high level. The categorization also sets the precedent for all applications developed in the factory. The controls defined in NIST SP 800-53 will be implemented as a minimum baseline and the controls that are selected in this activity will ultimately need to be met to obtain an ongoing ATO. These controls will determine what is evaluated as part of the initial assessment and should be traceable back to the RMF database throughout the SDLC.

4.1.1.2 Implement Security Controls Through Secure Tools and Technologies

Building the Software Factory aligns to phase three of the RMF process. The goal is to maximize control inheritance through preapproved IaaS and PaaS, and to use tools to automate control implementation. Traceability to controls is required for assessment in the next phase. As discussed in ***Section 3.0***, The activities involved in building the Software Factory include:

- Assign Staffing
- Establish Infrastructure and Platform
- Develop and Establish Processes
- Configure Tools

4.2 Initial Authorization – Evaluating and Authorizing the Software Factory

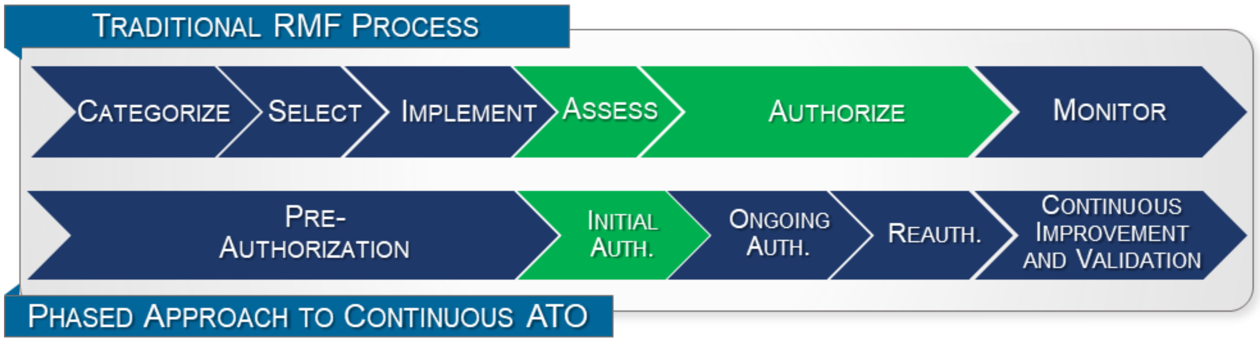


Exhibit 12: The Initial Authorization phase of Continuous ATO incorporates Step 4 Assess and Step 5 Authorize of the traditional RMF process .

The process of Initial Authorization can be tied to the wording in the NIST *Supplemental Guidance on Ongoing Authorization*. "Initial authorization is defined as the initial (start-up) risk determination and risk acceptance decision based on a zero-base review of the information system conducted prior to its entering the operations/maintenance phase of the system development life cycle. The zero-base review includes an assessment of all security controls (i.e., system-specific, hybrid, and common controls) contained in a security plan and implemented within an information system or the environment in which the system operates."

4.2.1 Required Tasks for the Initial Authorization Phase

With the pre-authorization groundwork complete, the initial authorization of the Software Factory is now performed. At roughly the same point in the lifecycle as the RMF Assess phase, it involves an AO assessment and a subsequent remediation by the Software Factory team of any discovered issues. When these tasks are complete, a transition to ongoing authorization begins.

4.2.1.1 Conduct Assessment of Software Factory

The AO assesses the operations of the Software Factory to determine that there is sufficient focus on security in every stage of the SDLC process. The assessment is based on the data categorization and control selection completed during pre-authorization. The Software Factory should demonstrate that software security is intrinsic to the DevOps software development methodology and the security accreditation process (as with the testing process) is done alongside development. This includes reducing human error by using best in class commercial tools and ongoing security and functionality tests. Most importantly, the AO is assessing whether all applicable NIST controls are included and checked prior to release. This includes creating a Trackable Identifier for any task in the process from control to task to code. Each assessment will be different due to the varying categorizations of data systems and selection of controls. The evaluation criteria should trace directly to the RMF database to ensure that controls are satisfied prior to authorization. The choice of evaluation criteria depends on the data categorization and controls selected earlier in the process. Ultimately, the AO is responsible for determining their evaluation criteria for the Software Factory's inherent security. Some options for the criteria are included in ***Exhibit 13***.

Assessment Area	Evaluation Criteria
Software Development Processes	▪ Clearly defined security-related roles and responsibilities ▪ Feedback loop between security and development teams ▪ Clear process for resolving issues as they are identified ▪ Continuous Monitoring Plan ▪ Data back-up/protection strategy ▪ Use of version control system ▪ Real-time access to documentation built to the full NIST 800.53 template ▪ Accountability for the Fix: Critical vulnerabilities addressed within 10 days; moderate the next sprint
Platform	▪ Implementation of pre-authorized IaaS and PaaS ▪ Maximized control inheritance through technologies
Software Factory Pipeline	▪ Use of well-defined and automated continuous monitoring ▪ Automated scanning and reporting where possible ▪ User Access Management and two-factor authentication ▪ Managed services for patching/update process ▪ NIST control traceability for ongoing compliance with C&A and individual code check-ins ▪ Extensive use of open source libraries which are routinely updated with security bug fixes ▪ Automated Build and Test: Daily build to check code against unit tests ▪ Dynamic Test (Weekly): Random bulk data/break test ▪ Fuzz Test (Monthly): Random bulk data/break test ▪ Scans (each sprint): Check platform, web interface, DB scan, and network traffic
Production System	▪ Automated deployment process – tools to ensure every deployment is conducted the same ▪ Processes to only deploy code which was thoroughly tested as part of the pipeline ▪ Annual penetration test with random testing as required by AO ▪ Development environment is mirror of production environment

Exhibit 13: The AO determines the evaluation criteria by which he/she will evaluate the security of the Software Factory for the initial assessment based upon the controls selected in the Pre-authorization phase.

4.2.1.2 Make Recommendations for Remediation

Following the initial assessment, the AO defines any changes to the people, technology, or processes that must be made prior to authorization of the Software Factory. The Software Factory team will have a time-bound set of milestones to receive an Initial Authorization signature from the AO. Similar to the current RMF process, Continuous ATO may require several discussions or reviews before all parties are comfortable with the overall security of the applications that will be developed within the Software Factory.

4.3 Ongoing Authorization – Achieving a Continuous ATO

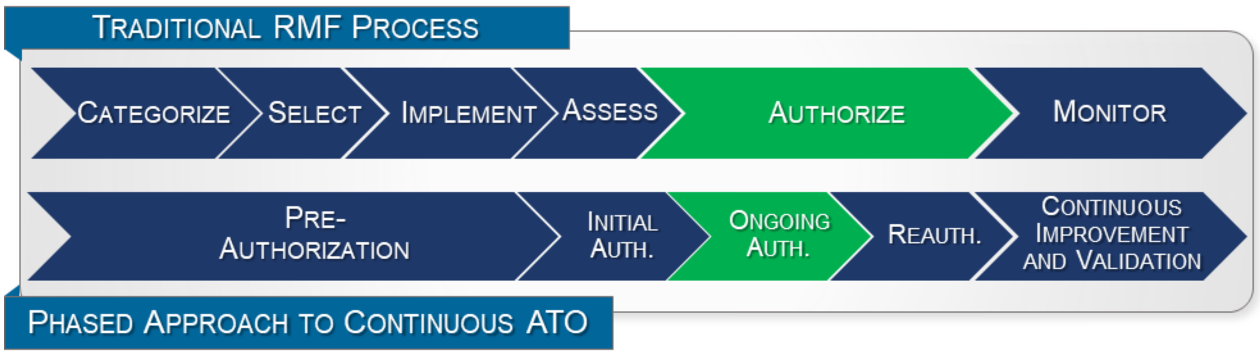


Exhibit 14: The Ongoing Authorization aligns with the Authorize step of the traditional RMF process with similar activities and outcomes.

The process of Ongoing Authorization is defined by the NIST *Supplemental Guidance on Ongoing Authorization*. "Ongoing authorization is defined as the subsequent (i.e., follow-on) risk determinations and risk acceptance decisions taken at agreed upon and documented frequencies in accordance with the organization's mission/business requirements and organizational risk tolerance. OA is a time-driven or event-driven security authorization process whereby the AO is provided with the necessary and sufficient information regarding the near real-time security state of the information system (including the effectiveness of the security controls employed within and inherited by the system) to determine whether or not the mission/business risk of continued system operation is acceptable."

4.3.1 Required Tasks to Achieve Ongoing Authorization

With the initial authorization completed, maintaining the Software Factory's ongoing authorization becomes a streamlined process of maintaining self-directed reviews based on the AO's pre-defined risk determinations.

- Conduct the subsequent risk determinations and risk acceptance decisions taken at agreed upon and documented frequencies
- Review the near real-time security state of the IS to determine whether or not the mission/business risk of continued system operation is acceptable
- Sign Continuous ATO memo

4.4 Reauthorization – Maintaining a Continuous ATO

The concept of achieving reauthorization in a Continuous ATO development cycle is another significant departure from the traditional RMF process. Reauthorization has historically been another major source of delays for software updates, as it is required every time a vendor releases a major change in the production code. In a Continuous ATO environment, with constant monitoring and built-in adherence to security controls, reauthorization can take place in response to arbitrary guidelines and known incidents alone. In Kessel Run's implementation, this was accomplished in part through quarterly penetration testing, but may be different for future Software Factories. AOs have complete flexibility to set guidelines that meet reauthorization standards with which they are comfortable; timelines, security checks, and outside authorities can all be tailored.

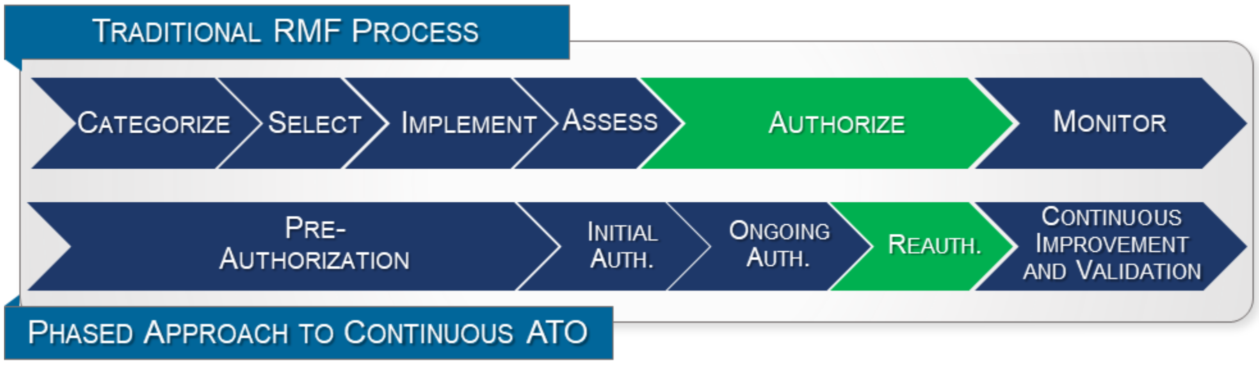


Exhibit 15: The Reauthorization phase can be either event-drive or time-driven.

The process of Reauthorization follows the same NIST *Supplemental Guidance on Ongoing Authorization.* "Reauthorization is defined as the static, single point-in-time risk determination and risk acceptance decision that occurs after initial authorization. In general, reauthorization actions may be time-driven or event-driven; however, under OA, reauthorization is typically an event-driven action initiated by the AO or directed by the Risk Executive (function) in response to an event that drives information security risk above the previously agreed upon organizational risk tolerance. Reauthorization consists of a review of the information system similar to the review carried out during the initial authorization but conducted during the operations/maintenance phase of the system development life cycle rather than prior to that phase. The reauthorization process differs from the initial authorization inasmuch as the AO can initiate: (i) a complete zero-base review of the information system or common controls; or (ii) a targeted review based on the type of event that triggered the reauthorization, the assessment of risk related to the event, the risk response of the organization, and the organizational risk tolerance. Reauthorization is a separate activity from the ongoing authorization process, though security-related information from the organization's ISCM program may still be leveraged to support reauthorization. Note also that reauthorization actions may necessitate a review of and changes to the ISCM strategy which may in turn, affect ongoing authorization."

4.4.1 Required Tasks for Achieving Reauthorization

Based on the guidelines determined by the AO during earlier phases, Reauthorization is a flexible event that can consist of different types of review based on the time or incident-driven need. The Reauthorization process will likely be different for all Software Factories and may involve some or all of the follow activities:

- As required by event-driven action or direction by the Risk Executive, conduct a static, single point-in-time risk determination and risk acceptance decision after the initial authorization
- Review IS during operations/maintenance phase of SDLC
- Complete a zero-based review of IS or common controls; or a targeted review based on type of event that triggered the reauthorization, the assessment of risk related to the event, the risk response of the organization, and the organization risk tolerance
- If necessary, review and change the ISCM strategy

4.5 Continuous Improvement and Validation

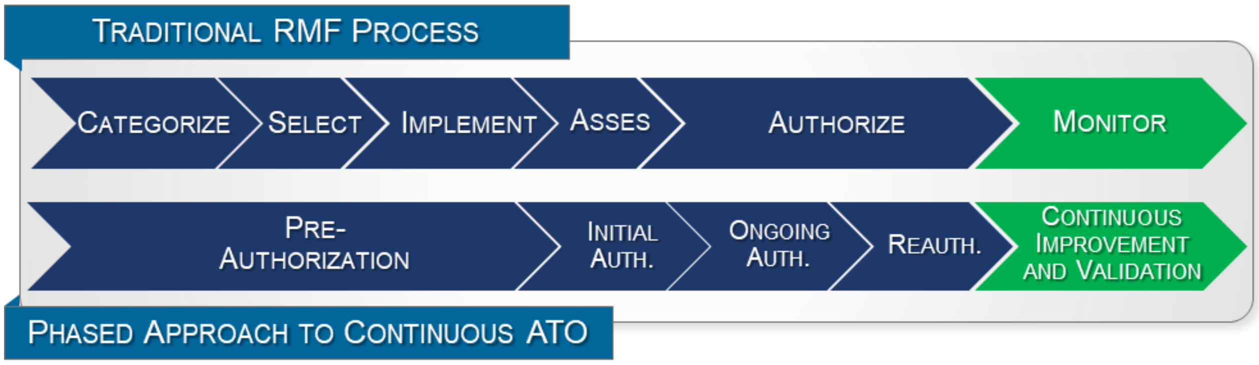


Exhibit 16: Continuous Improvement and Validation of the Software Factory requires ongoing efforts to improve the overall security of the Software Factory and the applications developed.

4.5.1 Required Tasks for the Continuous Improvement and Validation

When software moves into production and release phases, so begins the ongoing effort to maintain and improve its security and functionality as issues arise or new features are required.

- Conduct Penetration Testing as required by ATO or as determined by AO
- Ensure completion of all activities as required by ATO
- Continue including cybersecurity experts throughout the life of the program
- Confirm that AO and cyber personnel have real-time access to results of testing, scanning, monitoring, and performance metrics in a mutually agreeable format
- Verify that critical vulnerabilities are mitigated within 24 hours of discovery and moderate vulnerabilities are mitigated within a timeframe acceptable to the AO
- Maintain disciplined compliance with program processes and procedures, with a focus on spreading best practices across teams
- Ensure the security of the development environment, including physical security, information security, and operational security measures to prevent unauthorized access, misattribution of access, or unintentional release of information
- Keep team members up to speed on software development and cybersecurity best practices through continuous learning initiatives, including externally provided DevSecOps training to supplement the paired programming and enablement model
- Continue testing and scanning activities as designated intervals

5.0 Conclusion

The Air Force has seen tangible results from the Kessel Run program and their implementation of DevOps, which has saved the Air Force millions of dollars since inception. Project teams can take this playbook and tailor it to their needs by working closely with their AO to select the appropriate tools and infrastructure for building and operating the IS. We recommend organizations apply the model as an addendum to this guide and use this to serve as their initial documentation toward obtaining a Continuous ATO as part of the Pre-authorization stage.

The Continuous ATO Playbook was researched, developed and written by Wolf Den Associates, LLC in support of the Air Force's Cyberspace Innovation team led by Ms. Lauren Knausenberger. For questions or inquiries, please contact Rick, Nathan, or Tyler.

Wolf Den Associates
www.wolfdenassociates.com
1751 Pinnacle Drive
McLean, VA 22102

Rick Tossavainen, Vice President
Phone Number: 703-638-5924
Email: Rick@wolfdenassociates.com

Nathan Hintz, Senior Associate
Phone Number: 434-409-8082
Email: Nathan@wolfdenassociates.com

Tyler Fordham, Senior Associate
Phone Number: 509-844-6977
Email: Tyler.Fordham@wolfdenassociates.com