FEDERAL DEVOPS SUMMIT

MARCH 12, 2019 | MARRIOTT METRO CENTER | WASHINGTON, D.C.

On behalf of the Advanced Technology Academic Research Center, I am proud to announce the release of a White Paper documenting the MITRE-ATARC DevOps Collaboration Symposium held on March 12, 2019 in Washington, D.C. in conjunction with the ATARC Federal DevOps Summit.

I would like to take this opportunity to recognize the following session leads for their contributions:

MITRE Chairs: Melissa Heeren and Michelle Casagni

Challenge Area 1: Overcoming Organizational and Technical Barriers to DevOps
Government Chairs
Kevin Bierschenk, IRS Director of Investment Portfolio Control and Oversight, Strategy and Planning
Debbie Trumbull, IRS Director, Demand Management & Project Governance Division
Mike Waite, DOJ Chief, Application Services Branch, US Trustee Program

Industry Chair
Andrew Baumann, Atlassian DevOps Team Lead

MITRE Chair
Mark Wahnish,MITRE Senior Software Engineer

Challenge Area 2: DevSecOps Automation and Software Factories
Government Chairs
Robin Moses, IRS Deputy Director, Enterprise Computing Center
John Soscia, IRS Continuous Integration / Continuous Delivery Initiative Manager
Amin Qazi, IRS DevOps Technical Advisor

Industry Chair
Rajiv Kadayam, Pyramid Systems CTO

MITRE Chair
Rick Cagle, MITRE Principal Information Systems Engineer, DevOps Market and Platform Lead

Challenge Area 3: Achieving Ongoing Authorization via DevSecOps
Industry Chairs
Rick Slade, Compuware Executive Solutions Architect, DevOps
Chuck Svoboda, RedHat OpenShift Practice Lead, Public Sector

MITRE Chair
Richard Eng, MITRE Principal Software Systems Engineer

Challenge Area 4: Implementing DevOps in Legacy Systems
Government Chairs
Kevin O'Connor, IRS Program & Project Mgmt Office, Enterprise Ops, Server Support & Services Division
Cheryl Quade, IRS IT Program Manager Enterprise Ops Server Support & Services Division

Industry Chair
Rizwan Tanoli, Acuity Director, Software Engineering: DevOps, Big Data, Agile Delivery

MITRE Chairs
Jennifer Flamm, MITRE Lead Agile Software Engineer
Vibha Dhawan, MITRE Lead Agile Software Engineer

Challenge Area 5: Cultural Barriers to DevOps Adoption
Government Chairs
Lisa Starr, IRS Acting Director, Enterprise Program Controls
Donna Reindorf, IRS Enterprise Program Controls & Organizational Readiness

Industry Chairs
Mihir Pathak, Stack Overflow Executive VP of Strategy
Laird Williams, Macro Solutions Lean/Agile/DevOps Practice Lead

MITRE Chairs
Noreen Gilsinn, MITRE Lead Systems Engineer
Deanna Stanley, MITRE Lead Software Engineer, Configuration Management

Below is a list of government, academic and industry members who participated in these dialogue sessions:

Challenge Area 1: Overcoming Organizational and Technical Barriers to DevOps: Nathan Baldwin, DISA; Jacqueline Barber, MITRE; Andrew Baumann, Atlassian; Yonas Berhe, IRS; Kevin Bierschenk, IRS; Carmie Carter, MITRE; Kristin Cooke, Acuity; Jonathan Lewis, USAID; David Rubens, IRS; Debbie Trumbull, IRS; Mike Waite, DOJ; Mark Wahnish, MITRE; Sahar Yamini, IRS; Keondra Whetstone, IRS; Audrey Winston, MITRE

Challenge Area 2: DevSecOps Automation and Software Factories: Jules Byron, MITRE; Rick Cagle, MITRE; Sooyon Cho, USCIS; Bob Classen, MITRE; Rajiv Kadayam, Pyramid Systems; Rupesh Kumar, IRS; Robin Moses, IRS; Shyam Parhi, FAA; John Soscia, IRS; Amin Qazi, IRS

Challenge Area 3: Achieving Ongoing Authorization via DevSecOps: Joe Barbano, USDA; Ed Boriso, GSA; Richard Eng, MITRE; David Kye, DISA; Paul Metzger, MIT; Vance Middleton, IRS; Thomas Reaves, EPA; Rick Slade, Compuware; Andrea Stenberg, IRS; Tom Volpe, SSA; Chuck Svoboda, RedHat

Challenge Area 4: Implementing DevOps in Legacy Systems: Eric Cohen, GAO; Scott Davis, VA; Vibha Dhawan, MITRE; Jack Dervin, USAF; Jennifer Flamm, MITRE; Dan Jasper, USAF; Chase Noel, Nuvitek; Kevin O'Connor, IRS; Sonia Smith, Peace Corps; Khalid Spotwood, SSA; Matt Steiner, MITRE; Rizwan Tanoli, Acuity; Cheryl Quade, IRS; Tony Wann, MITRE

Challenge Area 5: Cultural Barriers to DevOps Adoption: Noreen Gilsinn, MITRE; Mihir Pathak, Stack Overflow; Marcus Peduzzi, USAF; Donna Reindorf, IRS; Jillian Seabrook, MIT; Deanna Stanley, MITRE; Lisa Starr, IRS; Cheryl Tran, IRS; Sudha Venkateswaran, Pyramid Systems; Mark Webber, MITRE; Laird Williams, Macro Solutions

Thank you to everyone who contributed to the MITRE-ATARC DevOps Collaboration Symposium. Without your knowledge and insight, this White Paper would not be possible.

Sincerely,

Tom Suder
President,
Advanced Technology Academic Research Center (ATARC)

FEDERAL IT SUMMIT SERIES

MARCH 2019 FEDERAL DEVOPS SUMMIT REPORT*

May 8, 2019

Melissa Heeren, Michelle Casagni, Jules Burgo, Rick Cagle,
Vibha Dhawan, Richard Eng, Jennifer Flamm, Noreen Gilsinn, Diane Hanf,
Deanna Stanley, Justin F. Brunelle
The MITRE Corporation

Tom Suder
The Advanced Technology Academic Research Center

Contents

1 ABSTRACT

The fourth ATARC (Advanced Technology Academic Research Center) Federal DevOps Summit was held on March 12, 2019 in Washington, D.C. During this summit, five MITRE-ATARC Collaboration sessions provided representatives from industry, academia, government, and MITRE the opportunity to discuss challenges the government faces implementing DevOps and Agile processes for software development and delivery.

The goal of these sessions is to create an interactive forum for participants to exchange ideas on best practices, recommendations, success stories, barriers, and requirements to advance the adoption of DevOps and Agile within the government.

Participants ranged from Directors, CTOs, and other executive-level leaders from government and industry to active DevOps practitioners from government, industry, and MITRE. Each collaboration session had a MITRE, government, and industry lead to facilitate the participants' discussions about challenge areas in DevOps and Agile and create recommendations to address these challenges.

This white paper summarizes the discussions in the collaboration sessions and presents recommendations for government, academia, and industry while identifying intersecting points among challenge areas.

This year's collaboration session participants revealed that there is overall acceptance of the need to move to a DevOps framework and embrace automation. Several key challenges and recommendations emerged from the discussions.

Challenge Areas

The March 2019 Federal DevOps Summit highlighted several challenges facing the Federal Government's adoption of DevOps and Agile practices and presented recommendations to address those challenges.

Across the collaboration sessions there was agreement that government software programs need automation of development, test, deployment, and operational processes. Several challenges were identified by the session participants.

1. It is difficult to get the needed buy-in for automation from all organizations, team members, and leaders who are involved in the software development and operations process.

2. Stakeholders and Software Development Lifecyle (SDLC[1]) organizations do not recognize the value of DevOps.

[1] SDLC includes planning, requirements, design, development, test, security, deployment, and maintenance.

3. Existing staff (government and contractor) often lacks the deep technical knowledge and expertise to implement DevOps.

4. The lack of a high-level champion along with siloed organization structures prevents the enterprise adoption of full SDLC DevOps.

5. There is a perception that the purchase and deployment of tools is the same thing as DevOps adoption.

Recommended Solutions

1. **Identify a high-level champion**: It is necessary to have a committed champion at a high enough level in the organization to cut through organizational and cultural barriers.

2. **Communicate value appropriately to the recipient**: Realize each stakeholder (individual and organization) has its own value proposition; change agents and DevOps champions should communicate the value being delivered to each stakeholder (e.g., executives might need to demonstrate governance and security needs to trust that the solution is not vulnerable).

3. **Make customers, security, and testing part of your teams**: Using matrixed teams consisting of development, test, cybersecurity, and operations personnel can start to break down organizational barriers.

4. **Focus on feedback**: Define metrics that are meaningful to stakeholders. Continuously improve through feedback and failing fast. Look for big pain points that can be quickly resolved and celebrate those wins first.

5. **Modify the process before choosing the tools**: The new process workflow should take input from the security teams to ensure security is embedded throughout. The new processes should focus on thorough trusted testing earlier in the process. This testing can later be automated.

6. **Encourage adoption of automation**: Leadership needs to understand and communicate that automation takes time initially but leads to performance and efficiency gains in the long run.

7. **Provide training**: Train the project teams on the tools and processes to encourage effective use of the tools. Consider outside expertise to provide guidance and best

practices on how to efficiently and effectively configure and use the tools. Training is an ongoing investment.

2 INTRODUCTION

The fourth ATARC (Advanced Technology Academic Research Center) Federal DevOps Summit was held on March 12, 2019 in Washington, D.C. During this summit, five MITRE-ATARC collaboration sessions provided representatives of industry, academia, government, and MITRE the opportunity to discuss challenges the government faces using DevOps and Agile processes for software development and delivery. The goal of these sessions is to create an interactive forum for participants to exchange ideas on best practices, recommendations, success stories, barriers, and requirements to advance the adoption of DevOps and Agile within the government.

Given the criticality of security in government software systems, it is not surprising that there was a great focus on security and DevSecOps (building in secure code practices and appropriate security scanning throughout the DevOps pipeline to ensure that software delivered is secure).

Participants ranged from Director, CTO, and other executive levels from government and industry to practitioners from government, industry, and MITRE. Each collaboration session had a MITRE, government, and industry lead to facilitate the participants' discussions about challenge areas in DevOps and Agile and create recommendations to address these challenges.

The MITRE Corporation is a not-for-profit company that operates multiple Federally Funded Research and Development Centers (FFRDCs) [4]. ATARC is a non-profit organization that leverages academia to bridge between government and corporate participation in technology[2]. MITRE works in partnership with ATARC to host these collaborative sessions as part of the Federal DevOps Summit.

This white paper is a summary of the results of the collaboration sessions and identifies suggestions and recommendations for government, industry, and academia while identifying cross-cutting issues among the challenge areas.

3 COLLABORATION SESSION OVERVIEW

Each of the five MITRE-ATARC collaboration sessions consisted of a focused and moderated discussion of current problems, gaps in work programs, potential solutions, and ways forward

[2] http://www.atarc.org/about/

regarding a specific challenge area. The sessions for this summit addressed the following topics:

- Overcoming Organizational and Technical Barriers to DevOps – How can Federal organizations develop better ways to fully embrace the benefits of DevOps processes?
- DevSecOps Automation and Software Factories – What are the limits on automation of DevSecOps pipelines?
- Achieving Ongoing Authorization via DevSecOps – What are ongoing authorization strategies and approaches to alleviate the security and compliance bottlenecks?
- Implementing DevOps in Legacy Systems – What are the benefits of DevOps that can be realized by legacy systems, and what strategies and tactics may enable a DevOps pipeline for a legacy system?
- Cultural Barriers to DevOps Adoption – How does DevOps impact stakeholders beyond Development and Operations, and how can different organizations foster a sense of community?

This section outlines the goals, themes, and findings of each of the collaboration sessions.

3.1 Overcoming Organizational and Technical Barriers to DevOps

Many organizational structures exist in government spaces that do not align with the DevOps focus on rapid delivery of valuable software to the user. Similarly, there are real technical barriers to seamless DevOps adoption across these organizations. Participants in this session deep-dived these issues, identifying challenges and developing recommendations to address these challenges.

3.1.1 Session Goals

This session set goals to discuss challenges and solutions around the following topical areas:

- Switching development processes to better support DevOps workflows
- Addressing the cost of switching to DevOps for legacy software
- Obtaining buy-in for changed team workflows
- Incorporating user feedback in the DevOps process

- Modernizing legacy security certification techniques
- Maintaining security and quality while deploying rapidly and often

3.1.2 Challenges

Session participants stated that DevOps technical challenges are usually addressed first and then the organization begins to work on organizational challenges. The technical and organizational challenges discussed include the following:

- Siloed organizations are steeped in decades of investments
- Seamless cybersecurity is not included in the DevOps pipeline
- Teams encounter many organizational governance entities which slow them down
- Middle and upper management are not aware of the benefits of implementing proper DevOps
- Stakeholders often only see or experience the immediate disruption and are not able to see the overall benefit
- Software is not architected into smaller components for frequent delivery
- Cross-enterprise stakeholders fail to adjust and level-set practices to take full advantage of the DevOps capability
- Large-scale organizational change is difficult
- There is a lack of deep DevOps technical knowledge and expertise across the organization
- Change is hindered by potentially unwarranted trust and reliance on existing processes

3.1.3 Discussion Summary

Active discussion by the participants led to a number of valuable observations. Participants stated that DevOps technical barriers are usually tackled first, especially if adoption is driven by forward-leaning software development teams. This often means that other aspects of the organization – such as legal and policy adherence, test and evaluation, certification, and software quality assurance – are addressed later.

Longstanding organizations are often structured to support waterfall software development. Processes have matured to support product movement in and out of waterfall silos. This product movement across an organization is slow because applications are single, large products, and tightly coupled architecturally with hard-to-understand dependencies. To drive down risk, each of the siloes attempts to exhaustively validate the product from its perspective. Too often, management trusts these processes rather than its people.

Cybersecurity has long been regarded as a separate entity, usually found in later stages of the Software Development Lifecyle (SDLC). For many organizations it is especially difficult to include cybersecurity as an equal partner in the DevOps conversation.

There are usually separate financial cost centers for development vs. operations. To have a true DevOps pipeline, ownership and operations are best executed using a single cost center. This simplifies the question of "who owns it?" To address these barriers, session participants stated that it was important to focus on the following concepts:

- Have a champion
- Have metrics to measure DevOps success
- Educate all DevOps participants on the same principles
- Use an advocate from areas of the organization that are more advanced in DevOps to support those areas just learning about it
- Use an incremental approach to adopting DevOps
- Conduct value stream mapping to understand where the organization needs to focus
- If parts of the pipeline are outsourced to fill technical gaps, use performance-based contracting to ensure DevOps benefits are delivered
- Create matrixed teams to increase teams' productivity and lead to the teams' ownership of product outcomes

3.1.4 Recommendations

The session participants collaboratively identified several recommendations to address the identified organizational and technical challenges.

Have a champion: Management and leadership must engage and lead the charge on DevOps

adoption. There should not only be a champion for change in executive management, but there should be a committed change agent at all levels of the DevOps adoption process.

Have metrics to measure DevOps success: Establish metrics that measure progress towards meeting organizational goals. These newly established metrics are an indicator that the organization is committed to making change.

Educate all DevOps participants in the same principles: Expose everyone in the organization equally to the underlying principles and expected rules of engagement. Establish a regularly-convening forum to educate about DevOps, communicate progress, and build trust across all the stakeholders.

Use an advocate from areas of the organization that are more advanced in DevOps to support those areas just learning about it: Have knowledgeable team members on the development side of the organization reach out and help fill knowledge gaps that might exist on the operations and cybersecurity sides of the house, and vice versa.

Use an incremental approach to adopting DevOps: Evaluate the needed changes, and group them to create increments that can be more easily managed. This division helps to make progress towards adoption more recognizable, easier to quantify, easier to manage, and less overwhelming for stakeholders.

Conduct value stream mapping to understand where the organization needs to focus: Be highly sensitive to value to the user to increase user trust and increase satisfaction with the product. Identify specific pain points and show how DevOps addresses them; show how culture change can alleviate those pain points.

Use performance-based contracting if you outsource parts of the pipeline to ensure you get DevOps benefits: Establish specific metrics to start with that can be measured easily so that progress can be easily tracked and communicated; keep the stakeholders' eye on having a better outcome rather than laying blame.

Create matrixed teams to increase teams' productivity and lead to the teams' ownership of product outcomes: Matrixed teams consisting of development, test, cybersecurity and operations personnel can break down organizational barriers. Streamlining governance can

also help these matrixed teams be more efficient and empower them to follow best practices.

3.2 DevSecOps Automation and Software Factories

Much of the technical evolution of DevSecOps is driven by expanding on "Continuous" practices beyond Continuous Integration and Continuous Deployment. The trends follow the increasing ability of vendors to supply automation to testing, security, monitoring, and 'Authority-to-Operate'. This session focused on standardizing practices, tools and environments for DevOps and the constraints of automation.

3.2.1 Session Goals

The goals for this session included the following:

- How do we begin to standardize practices, tools and environments?
- What can and cannot be automated?
- What is a minimum viable product (MVP) for DevOps?
- What does a 'Software Factory' mean?

3.2.2 Challenges

Challenges in implementing as much automation as possible in a DevSecOps pipeline include the following:

- What are the limits on automation of DevSecOps pipelines?
- How do we begin to standardize practices, including tooling, accreditation, and deployment environments, across the federal government?
- How do we effectively close the feedback loop given classification or other sensitive deployment environments?
- If we suffer from inadequate test data and poor development/production environment parity, how do we test effectively?
- If we want to avoid constructing custom DevSecOps pipelines for each product architecture we want to build, what are the constraints to consider when building a DevSecOps Continuous Integration/Continuous Delivery (CI/CD) pipeline for maximum reuse?

3.2.3 Discussion Summary

The session began with introductions of the leads and their roles within their respective organizations. Voting on the stated goals focused the conversation towards DevOps standard practices, tools and environments. With the understanding that critical components of an automated pipeline are instantiated through a development process, the participants worked to get concurrence of a typical CI/CD practice. The pipeline stages of Build, Test, and Deploy were further broken down to include: Code Check-into a Repository; Build; Unit Testing; Code Coverage and Quality Checks; Static Security Scan; Check into an Artifact Repository; Prepare to Deploy; Deploy; Smoke Test. The output of each stage should be meaningful to the specific environment, and positive results allow the process to move to successive stages.

The conversation then turned to a question, "What does it mean for each stage to be considered 'done'?" For the Code Build stage, done is defined by a successful build with no human intervention. 'Done' in testing stages (Unit, Quality, and Security testing) means that given the constructed testing, the code passes all checks. Deploy 'done' means the release occurred to the environment and performs as expected. One of the outcomes from this discussion was a recognition by the participants of the need for meaningful feedback/metrics to inform future iterations.

With the standard pipeline practice settled, a discussion around security ensued. Several participants stated that security should occur at each stage, and therefore is not a stage unto itself. In the end, the attendees agreed with this concept and recognized that security should be incorporated much earlier in the process. There was also an understanding that with some activities, such as dynamic testing, it may not be possible to perform the operations very early in the process.

The participants then moved their attention to standard environments. They defined an environment as where the application lives and performs its function. Not surprisingly, the group settled on the following standard environments: Development, Staging (Integration), and Production. Fidelity, parity, and consistency are necessary between the standard environments to ensure that the application and its components respond in the same way. This led the participants to emphasize the need for consistent configuration management tools and practices to avoid environmental configuration differences. The need for Infrastructure-as-Code (IaC) to enforce configuration management was recognized. Along with good configuration management practices, it was noted that the workload should be the same for all environments; only the scale of the workload in Production is different.

The group concluded with a cautionary note. Transitioning legacy environments to new 'DevOps' enabled environments – whether in cloud or on-premise – can be challenging

[1]. The transition process is non-trivial and often frustrating. It's important to celebrate successes during a legacy transition.

With standard practices and environments, the team spent little time identifying the actual tools by name, but rather recognized there are tools that correspond with each layer/stage of the CI/CD pipeline. It was agreed that the actual tool was not as important as having a tool to fit the function of the pipeline. The conversation did drift into tool selection. The the participants were asked "Why did you pick the tools you did?" The answers varied: surveys, enterprise or industry standard, coupling with other tools, crowd sourcing, tools on hand, or staff experienced with tool set.

With time running out, the group was asked the other goal question, "What can and cannot be automated?" Respondents listed the following limitations to automation: user acceptance testing (continuous delivery versus continuous deployment), budget, culture, regulatory standards, classification levels, human feedback, subjective quality metrics, and air-gapped environments (e.g., shipboard environments, aircraft environments).

3.2.4 Recommendations

The participants identified several recommendations focused on standard processes, tools, and environments as well as constraints to automation.

- Push security to the left and include security in each stage. This is especially important for enterprises in the Federal space.
- Make customers and security part of your teams. These groups play an important role throughout the process.
- Define 'Done' criteria for each stage and publicize.
- Focus on Feedback: Define metrics that are meaningful to you. Continuously improve through feedback and failing fast.
- Look for big pain points that can be quickly resolved and celebrate those wins first.
- Display dashboards with workflow and blockage that are visible to all team members.
- Perform standard pipeline stages in parallel. Examine your processes to determine which may benefit by performing in parallel.
- Maintain consistent configuration management; it is necessary for environment parity and is required for automation.

- Understand the costs and benefits when examining automation. Identify your automation constraints first.

3.3 Achieving Ongoing Authorization via DevSecOps

Over 40 participants attended the Achieving Ongoing Authorization via DevSecOps session. Three quarters of the participants were from the government with the remainder from industry and academia. Ninety percent of the participants were from the development, test, or operations community. Three individuals self-identified as security engineers. Three individuals indicated that their organizations were in the process of getting an ongoing authorization (OA). All participants were interested in best practices and strategies to achieving ongoing authorization via DevSecOps.

3.3.1 Session Goals

The goals for this session were as follows:

- Learning how to remove obstacles to realizing the benefits of DevSecOps
- Obtaining DevSecOps best practices and lessons learned
- Identifying ongoing authorization strategies
- Sharing current approaches to eliminating the security compliance-checking bottleneck

3.3.2 Challenges

The challenges to achieve ongoing authorization discussed were as follows:

- Adopting the DevOps culture
- Incorporating security into the architecture and design
- Obtaining an Authority to Operate (ATO) – a checklist does not exist and the ATO process is not a “one size fits all”
- Maintaining the paperwork to obtain the initial and ongoing ATO
- Addressing the perception that scanning with automated security tools covers only 20% of the security controls
- Securing and controlling open source code

How Organizations Process Information

Pathological	Bureaucratic	Generative
Power oriented	Rule oriented	Performance oriented
Low cooperation	Modest cooperation	High cooperation
Messengers shot	Messengers neglected	Messengers trained
Responsibilities shirked	Narrow responsibilities	Risks are shared
Bridging discouraged	Bridging tolerated	Bridging encouraged
Failure ->scapegoating	Failure ->justice	Failure ->inquiry
Novelty crushed	Novelty ->problems	Novelty implemented

Table 1: This summary of the Westrum Model is derived from Westrum's 2004 publication [5]. It provides a mapping between methods of organizational information processing and the challenges facing the processing of information.

3.3.3 Discussion Summary

Over 40 participants attended the Achieving Ongoing Authorization via DevSecOps session, creating a lively standing room only discussion covering a range of issues, challenges, and suggestions on how to achieve ongoing authorization via DevSecOps. Three quarters of the participants were from the government with the remainder from industry and academia. Ninety percent of the participants were from the development, test or operations community. Three individuals identified themselves as security engineers. Three individuals indicated that their organizations were in the process of getting an ongoing authorization (OA). All participants were interested in best practices and strategies to achieving ongoing authorization via DevSecOps.

DevOps requires a cultural shift from siloed teams to a unified team delivering a trusted solution to the end user. By a show of hands, all the participants still had separate development, operations, quality assurance (QA), test, security and privacy, and DevOps teams. No one had adopted a unified DevSecOps organizational structure. Per Conway's Law[3], "Any organization that designs a system (defined broadly) will produce a design whose structure is a copy of the organization's communication structure." Until organizations are structured and function as a single DevSecOps team they will have bottlenecks and inefficiencies in deploying functioning solutions to the end user. Westrum's organizational model (presented in Table 1 which is a compilation of information from the model organized as a table by the authors of this paper) is an indicator of whether an organization is capable and committed to adopting a DevSecOps culture.

[3] `http://www.melconway.com/Home/Conways_Law.html`

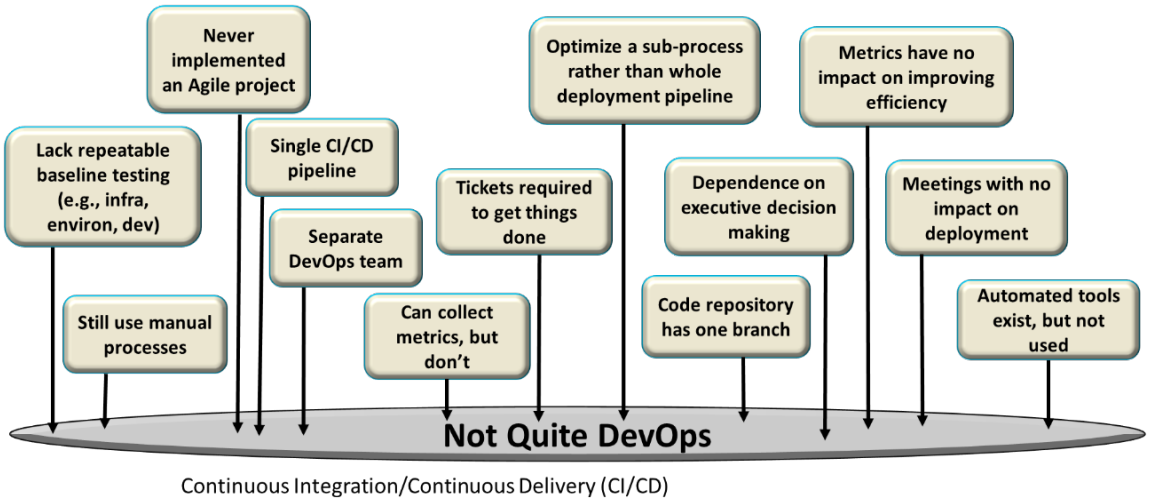

Figure 1: The indicators of "Not Quite DevOps" in an organization. If any of the items exist, the organization is not really DevOps and is unlikely to achieve ongoing authorizations via DevSecOps.

The Westrum organizational model [5] can be used to determine whether an organization is committed and capable of adopting a DevSecOps culture. Most of the participants agreed that their organization had some work to do to achieve the Generative model needed to realize a DevSecOps culture.

There is a need for security engineering professionals to work with development teams to architect and design trusted solutions at the beginning of the DevSecOps process. Relying on security control compliance assessments near the end of the delivery cycle will result in sub-optimization of the DevSecOps process and delay obtaining an ATO. Development and operations teams have, to various extents, adopted DevOps workflows. However, many of the security and privacy teams are still following a waterfall mentality when it comes to security and compliance. Part of the problem stems from the paucity of qualified security engineers versus security professionals who are compliance checkers. Security compliance officers need to be able to trust automated and inherited security control assessments, potentially by helping to design those assessments. It was mentioned that security is not the same as security compliance.

The group began to list indicators that would indicate that an organization is not really achieving DevOps. This list is summarized in Figure 1[4] and may not be complete since there was insufficient time to continue adding items to the list.

Many participants agreed that their organizations had collected automated tools (e.g.,

[4]Figure created based on participants' discussion.

testing, scripting, security scanning). However, none was able to pass the screening criteria in Figure 1.

The consensus was that no ATO checklist exists! Participant are all looking for a "Unicorn": an ATO checklist. The checklist needs to be created and implemented.

The need for a utility or tool to collect and track security control compliance for DevSecOps projects was identified. These include the NIST 800-53 [3] security controls plus the additional Department/Agency security controls that need to be applied to each system being implemented.

Having an automated tool would allow the DevSecOps team to shift the security control assessment and ATO processes to the left and reduce the amount of time it takes to receive their initial and ongoing authorizations. Some participants indicated that Kessel Run[5] and Cyber Unified Platform[6] could provide potential ATO checklists. Open Control Compliance Masonry[7] by 18F was another example of a security control assessment checklist. Masonry Compliance as Architecture (MCA) Framework[8] is available to facilitate the government's mobile mission-related solutions. In addition, Centers for Medicare and Medicaid Services (CMS) is currently prototyping a tool.

Concepts for achieving ongoing authorization include the following:

- Adaptive capability testing (ACT)
- Penetration testing
- Risk thresholds that are defined to map security capabilities to security controls

Several people believe that DevSecOps is a single pipeline; however DevSecOps should be thought of as a layer cake. DevSecOps is composed of multiple pipelines (e.g., Application, Environment, and Infrastructure) and each layer needs to be secured and verified for the system to receive an ATO.

The concept of a continuous ATO was discussed. Some thought that authorizing the process rather than the system was the solution. This works only if the process can respond to changes in security policy and vulnerabilities over the life of the project. Systems that already have their initial OA are still challenged to get an OA when new features are added because the risks may have changed.

[5] https://kesselrun.af.mil/
[6] https://www.afcea.org/content/unified-platform-unifies-cyber-warfighting
[7] https://github.com/opencontrol/compliance-masonry
[8] https://monkton.io/masonry

Participants discussed the need for system design elements tools. They were looking for tools that would examine the technology used in a solution and generate the security controls required. The participants were looking for tools to automatically generate the Security Impact Assessment (SIA), System Security Plan (SSP), and other artifacts that need to be generated to obtain an ATO.

DevOps metrics were discussed, including DORA[9] metrics, mean time to resolve, number of user stories, and test metrics. A simple baseline test is to measure how long it takes for a DevSecOps team to implement and deploy a "Hello World" application into production. Several participants indicated that the ATO process alone could take several months. The test process could take two months to schedule and execute. The recommendation was to document the bottlenecks and revise the processes to eliminate the bottlenecks rather than wasting time and money collecting DevSecOps tools. Another suggestion was to not measure things that are not going to be fixed; this simply wastes time collecting data that never gets used.

pen source security and verification is a DevSecOps issue: How much open source software should be used or allowed in a solution to accelerate the deployment of a trusted solution in a DevSecOps project? Challenges include verifying the security of the open source code. The source code and binaries need to be reviewed, verified, and placed under version control before they can be used. For example, the open source contributions to Kubernetes are from Huawei contributors[10]. Can the open source Kubernetes be trusted for sensitive government software solutions? The consensus was don't use Kubernetes! Another idea was that open source software can be provided and backed by a commercial vendor.

A final discussion point: Who accepts the risk in the organization? Is it the CIO, CISO, or some other risk executive? There is a need for authorizing officials and security assessors that understand DevSecOps and are willing to accept NIST Special Publication (SP) 800-37 Revision 2 [2], Risk Management Framework for Information Systems and Organizations.

3.3.4 Recommendations

Several recommendations were made as a result of the collaborative session.

- Measure how long it currently takes to deploy a "Hello World" application and remove inefficiencies before trying to implement DevSecOps.

[9] ttps://devops-research.com/

[10] https://www.huawei.com/en/press-events/news/2017/11/Huawei-Kubernetes-member

- Establish a central source of trusted DevSecOps best practices that is available to the community.
- Provide regular training by the security engineers to the development community to influence the architecture and design of the trusted solutions rather than compliance checking security into the system.
- Shift away from compliance-based testing to Adaptive Capabilities Testing (ACT).
- Take risks by changing the culture and processes to adopt a DevSecOps philosophy.
- Review, verify, and place open source software under version control before it is used to create a functioning solution.

3.4 Implementing DevOps in Legacy Systems

The Implementing DevOps in Legacy Systems session delved into the challenges inherent in transitioning legacy systems to DevOps. The session participants discussed definitions of success for DevOps adoption and the view that DevOps is a path to solve problems rather than a goal in itself. The group considered challenges to the success of DevOps adoption for legacy systems, and techniques and tactics to overcome those challenges. Attendees ranged from practitioners of DevOps who are at various stages of transitioning legacy systems to DevOps to staff who are interested in beginning the migration to DevOps.

3.4.1 Session Goals

The goals for this session were as follows:

- Identify characteristics of a legacy system
- Identify challenges to implementing DevOps in legacy systems (e.g., technical challenges and architectural constraints)
- Identify the benefits legacy systems can realize by implementing DevOps (e.g., shortening the release cycle and time to operations)
- Determine to what degree testing can be automated and how to prioritize that automation

3.4.2 Challenges

The key challenges faced in implementing DevOps in Legacy systems can be categorized into the following four major themes:

- Lack of Executive Champion
- Organizational Inertia
- Technical Challenges
- Program Management Barriers

3.4.3 Discussion Summary

Goals for legacy system owners included developing repeatable processes, improving product quality by identifying defects as early as possible, and delivering releases more frequently through shorter release cycles. Session attendees were curious as to whether the challenges they encountered in adopting DevOps in Legacy Systems were unique to their organization or applicable across other organizations as well. The discussion in the session built on these goals and the key challenges.

3.4.3.1 Lack of Executive Champion Several of the barriers to DevOps adoption in legacy systems could be addressed by an executive champion, preferably with direct influence over different groups within the organization. Session attendees agreed that without an influential executive champion there would be limited success in corralling different SDLC groups across different organizations (e.g., test, security, operations and maintenance (O&M)) together. For instance, a manager in development would have little sway in the test or O&M organization. To bring management on board with this philosophy, the champion for the transition effort needs to create a strong business case to highlight pain points including issues with current development, testing, and deployment; difficulties in achieving a speedy release cycle; and steep costs incurred in the current way of doing business.

3.4.3.2 Organizational Inertia In many organizations, project teams (e.g., development, test, systems engineering, O&M) supporting legacy systems are independent and not tightly integrated. This is not conducive to coordinated change. Teams working on legacy systems are accustomed to a certain way of operating which results in resistance to change. A misperception among development staff in these project teams is that security is a barrier to

getting work done. Processes for the legacy system are not always conducive to DevOps since efficiencies from automation of tasks or procedures are not recognized. Compounding this, multiple teams – such as test and O&M – can have negative attitudes towards DevOps as it is viewed as "taking jobs away" instead of an opportunity to redefine roles to add organizational value. Manual test teams sometimes view test automation as taking over their responsibilities. Similarly, legacy O&M teams can view automated deployments as a threat to their responsibilities.

3.4.3.3 Technical Challenges Project teams working on legacy systems are familiar with the tools they currently use to accomplish their mission but have less familiarity or experience with the DevOps toolset. This problem is compounded by employee turnover within the teams. This turnover results in knowledge loss which is not easily replaced. Once the DevOps toolset is identified, staff will need training on the tools to adequately and appropriately use the tools. Outside expertise may be needed to provide guidance and best practices on how to efficiently and effectively configure tools and use the tools to their maximum potential. Project teams need to be trained to implement security consistently and not as an afterthought. Many free online resources are available including training courses, documentation, manuals, and message boards. DevOps lacks defacto standardized training; it is not clear what resources are best for training the team.

3.4.3.4 Program Management Barriers Several barriers fall under the category of program management. From the outset, if the contract does not have specific verbiage that supports DevOps, it will not be included by government contractors without contract modifications. Similarly, O&M contracts may only include provisions for fixing issues, not adding enhancements or automation. Another constraint for government organizations is that fiscal year barriers limit when changes can occur. Program management may have unrealistic expectations about how long a transition to DevOps will take.

Management is then disappointed when the transition takes longer than anticipated. This can result in efforts that start and stop. Migrating legacy systems to DevOps requires long-term planning which needs to be adequately communicated, both within the team and to upper management. Program management is required to communicate that automation takes time initially but will yield increasing efficiencies over time.

3.4.4 Recommendations

The goal of transitioning legacy systems to use DevOps is achievable and the challenges may generally be overcome by breaking down large challenges into more manageable parts. The collaboration session offered several specific recommendations to address the challenges.

Designate an executive champion: The executive champion should be a leader throughout the process of transitioning to DevOps. The support of the executive champion is essential, especially in ensuring adequate resources (such as funding) on the migration project.

Modify the process before choosing the tools: Modify and update the processes involved in developing and maintaining a legacy system. The new process workflow should take input from the security teams to ensure security is embedded throughout. The new processes should focus on release methodology and automation. Develop processes for onboarding employees to ensure knowledge transfer.

Embed Security from the outset: Security should be embedded throughout the transition process, from the beginning. Security team members should be viewed as partners and – more importantly – the project team should understand that security is part of the developer's responsibility.

Implement Automated Security Scanning: Involving security in repeated scanning helps to mitigate risk and gives security teams confidence that a secure, workable solution can be delivered faster.

Identify a standard DevOps tool stack: Develop a standard stack of tools to integrate with the legacy system environment to achieve DevOps. The tools need to be actively managed for updates and fixes on a regular schedule.

Provide training: Train the project teams on the tools to encourage effective use of the tools. Consider outside expertise to provide guidance and best practices on how to efficiently and effectively configure and use the tools.

Increase automation of manual processes: Automate the software development process as much as possible, through unit tests, static code analysis, functional tests, build, and deployment. This will reduce the time spent on repeated manual processes.

Update procurement strategies to include DevOps verbiage: Involve procurement in government agencies from the outset to update procurement strategies to include DevOps in the contract verbiage (e.g., language regarding test automation, secure coding practices, speed of release).

Develop long-term plans for DevOps migration: Identify candidate legacy applications for migration. Prioritize applications to modernize based on value that modernization will provide to the organization. Plan to perform migrations incrementally to show measurable progress. Start small and succeed; demonstrate quick wins.

3.5 Cultural Barriers to DevOps Adoption

The Cultural Roadblocks to DevOps Adoption session identified common cultural obstacles that organizations face and successful mitigation measures for migrating to the DevOps methodology. Culture was defined as how the organization reflexively reacts to events when everyone is not trying to follow the rules.

3.5.1 Session Goals

The goals of this session were to discuss the following topics:

- Determine how DevOps impacts organizations beyond development and operations Understand when to plan for and implement cultural changes
- Determine how to foster a sense of community between different organizations
- Identify unique team building activities and traditions that can be implemented to encourage cohesiveness and communication between team members of various disciplines
- Identify incentives that have been used to encourage a DevOps culture
- Learn how to manage the unique challenges presented by contractors, unions, and tenured workers

3.5.2 Challenges

Over the course of the collaboration session, several roadblocks to DevOps adoption were identified.

- Siloed organizations
- Fear of change
- Loss of power and prestige
- Job loss
- Fear of failure
- Dev and Ops have different goals
- DevOps is seen as a suite of tools, not organizational change
- Lack of management champions
- Quantifying value of DevOps
- Lack of trust in automation

3.5.3 Discussion Summary

The Cultural Roadblocks to DevOps Adoption session identified common cultural obstacles that organizations face and successful mitigation measures for migrating to the DevOps methodology. Culture was defined as how the organization reflexively reacts to events when everyone is not trying to follow the rules.

While many of the strategies discussed in the collaboration session can be used individually, using the suite of recommendations in the order listed below may provide the framework for a DevOps adoption roadmap.

3.5.3.1 Identify a high-level and influential champion – Successful DevOps efforts cross organizational boundaries, which makes having a high-level, influential champion vital. The champion needs to have the authority to break down the barriers between organizations. High-level champions need to focus on the long-term goals for the DevOps initiative and how the DevOps program will improve the overall mission. High-level champions cannot delegate their role; instead they must assume a leadership role in breaking down barriers and bringing success to the DevOps initiative. In organizations with frequent management turnover, the high-level champion needs to define the goals of the DevOps initiative over the long term, so that momentum can be maintained even after his or her departure.

3.5.3.2 Need to define and communicate value – The value of DevOps needs to be conveyed to all constituents of an organization in order to get buy in for migrating to DevOps. For many commercial ventures, value is easily mapped to cost, either through savings on development expenditures or through increased revenue by being first to market with new product features.

Government agencies often are not revenue centers, which means that the value of adopting DevOps needs to be expressed in terms of the advantages that DevOps will bring to the organization. These advantages could be providing better service, mitigating or reducing cyber risks for government websites and computing infrastructure, or making the development and deployment team's processes more streamlined.

After the value of moving to DevOps has been defined, steps should be taken to assess if the value is actually being realized. Metrics that measure various aspects of the value statement should be identified and collected; not only will they prove the value of the initial DevOps migration, but they will also demonstrate the value of ongoing process improvement efforts. These metrics should be standard across all DevOps projects in order to provide a common language and so that the value of DevOps across the entire organization can be assessed.

3.5.3.3 Plan for organization readiness – DevOps is not merely tools; it also encompasses people and processes. Before focusing on tools, create a plan to change the existing processes and communicate with the people who support them. DevOps will force organizations to work closely together in ways that may be uncomfortable and new. Some people may find the combined DevOps organization threatens their personal power or prestige. This must be addressed up front. Others may fear that the drive for automation will make their jobs obsolete; this creates natural resistance.

A plan for implementing DevOps needs to address how the change will impact the people. One primary method to address these impacts is regular, open communication. Hosting open question and answer sessions or manning an information table during lunch will help familiarize organizations with the upcoming changes and address individual concerns.

It is important to tie the DevOps initiative into the organization's overarching mission, and include the customer – whether internal or external – in the planning. In addition, by incorporating the DevOps initiative into staff development plans so that individuals have a stake in the initiative's success. Training plans should be designed to address all the changes, both in processes and tools. Providing people with new, valuable job skills will help alleviate organizational concerns.

Communicating the DevOps initiative to various stakeholders is vital. It is important to define the value of the DevOps initiative in ways that the stakeholders appreciate. Since most government projects are not cost centers, focusing on money is not the primary way to communicate value and may lead to cost-cutting attempts that demoralize the organization. Instead, organizations should focus on what the stakeholder cares about, whether it is reducing errors and bad press or in improving the organization's effectiveness. Not all stakeholders will value the same things. What high-level champions value may be different from what lower level organizations value. DevOps initiatives need to determine how to communicate value at all organizational levels.

3.5.3.4 Create an environment for success – Management's key responsibility is to create an environment in which their people can succeed – leaders succeed when their people succeed. Creating an environment for successful DevOps adoption requires more than removing fears and communicating value. It also requires allowing the organizations involved to find their own "best path." DevOps and Agile emphasize autonomy of both people and teams. A successful DevOps initiative addresses people's fears, gives them the skills to succeed, and empowers individuals and teams to make changes to their processes and procedures to better accommodate DevOps.

Creating an environment for success requires proving a clear vision for why DevOps is being adopted. It requires assessing the organization's skills and developing training plans to address experience gaps in the workforce. Allowing for process flexibility rather than forcing a one-size-fits-all solution empowers people and teams to develop solutions that work best for their unique circumstances. Regular communication of the DevOps initiative's goals, implementation status, and future plans allows the workforce to stay in sync and work toward the end goal.

3.5.3.5 Redefine failure as a learning opportunity – DevOps encourages people and organizations to "fail fast." This sort of organizational mindset requires redefining failure as a learning opportunity. Individuals should be encouraged to experiment and learn from each failure. The lessons need to be communicated outward so other organizations may learn, too. There should be a regular means of communicating lessons learned, whether via a newsletter or a periodic town hall. Recognizing people for learning rather than criticizing failures will empower individuals to stretch themselves and their work product.

3.5.3.6 Incentives for adopting DevOps do not have to be costly – Encouraging DevOps adoption takes more than communicating a clear vision and value. Providing incentives will help speed an organization's willingness to change. Using monetary incentives is expensive and sometimes counterproductive. Instead, management should look to alternative incentives that provide high-value reward to organizations.

Recognition is a valuable incentive for encouraging change. Recognition should come from both external sources (such as management publicly acknowledging individual contributions) and internally with teams recognizing their MVPs. People thrive on feeling valued, and acknowledging their contributions regularly will decrease the cultural roadblocks to DevOps adoption.

3.5.3.7 Change takes time – be patient – Change takes time. While this would be true for many new initiatives, this is especially accurate for DevOps. Since many organizations will be shifting to DevOps from older development methodologies such as waterfall, team members will have to go through a significant shift in the way that they think about and execute development and deployment activities. People's fear of change, especially on such a large scale, will take time to address and assuage. All previous processes will need to be evaluated to determine how they will need to be modified so that they better fit the DevOps paradigm. This shifting and changing will take time before an overall DevOps implementation is realized. One of the best strategies for making sure DevOps adoption is successful is to temper expectations that migration will happen quickly.

3.5.4 Recommendations

The strategies provided by session participants can be used individually or as a whole to minimize the culture roadblocks to DevOps adoption.

- Identify a high-level and influential champion who can clearly define the vision and value of migrating to DevOps
- Plan for organizational readiness
- Create an environment for success
- Redefine failure as a learning opportunity
- Incentives for adopting DevOps do not have to be costly
- Change takes time – be patient

4 SUMMIT RECOMMENDATIONS & CONCLUSIONS

The March 2019 Federal DevOps Summit highlighted several challenges facing the Federal Government's adoption of DevOps and Agile practices and presented recommendations to address those challenges.

This year's collaboration session participants revealed that there is overall acceptance of the need to move to a DevOps framework and embrace automation. Several key takeaways can be gleaned from the collaboration session discussions.

4.1 Challenge Areas

Across the collaboration sessions there was agreement that there is a need for automation of development, test, deployment, and operational processes. The challenges are as follows:

1. It is difficult to get the needed buy-in for automation across all organizations.
2. Stakeholders and SDLC organizations do not recognize the value of DevOps.
3. Existing staff lacks the deep technical knowledge and expertise to implement DevOps.
4. Lack of high-level champion and siloed organizations prevent the enterprise adoption of full SDLC DevOps.
5. There is a perception that the purchase and deployment of tools is the same thing as DevOps adoption.

4.2 Recommended Solutions

Likewise, there were several recommendations that arose from the common themes at the DevOps summit.

1. **Identify a high-level champion**: It is necessary to have a committed champion at a high enough level in the organization to cut through organizational and cultural barriers.
2. **Communicate value appropriately to the recipient**: Realize each stakeholder (individual and organization) has its own value proposition and communicate the value being delivered to each (e.g., executives might need to demonstrate governance and security needs to trust that the solution is not vulnerable).

3. **Make customers, security and testing part of your teams**: Using matrixed teams consisting of development, test, cybersecurity, and operations personnel can start to break down organizational barriers.

4. **Focus on feedback**: Define metrics that are meaningful to you. Continuously improve through feedback and failing fast. Look for big pain points that can be quickly resolved and celebrate those wins first.

5. **Modify the process before choosing the tools**: The new process workflow should take input from the security teams to ensure security is embedded throughout. The new processes should focus on thorough trusted testing earlier in the process. This testing can later be automated.

6. **Automation**: Leadership needs to understand and communicate that automation takes time initially but is faster in the long run.

7. **Provide training**: Train the project teams on the tools to encourage effective use of the tools. Consider outside expertise to provide guidance and best practices on how to efficiently and effectively configure and use the tools. Training is an ongoing investment.

ACKNOWLEDGMENTS

The authors of this paper would like to thank The Advanced Technology Academic Research Center and The MITRE Corporation for their support and organization of the summit. The authors would also like to thank the session leads and participants that helped make the collaborations and discussions possible. A full participant list is maintained and published by ATARC on the DevOps Summit web site[11].

REFERENCES

[1] J. F. Brunelle, A. Bognar, V. Dhawan, N. G. Parrish, A. King, V. Kuppusamy, M. Malayanur, T. Harvey, and T. Suder. June 2018 Federal Cloud & Data Center Summit Report. Technical

[11] https://atarc.org/event/devops-summit-agenda/

report, The MITRE Corporation; The Advanced Technology Academic Research Center, 2018.

[2] NIST. Guide for Applying the Risk Management Framework to Federal Information Systems: a Security Life Cycle Approach. Technical Report Special Publication 800-37, National Institute of Standards and Technology, 2019.

[3] NIST. Recommended Security Controls for Federal Information Systems and Organizations. Technical Report Special Publication 800-53, National Institute of Standards and Technology, 2019.

[4] The MITRE Corporation. FFRDCs – A Primer. `http://www.mitre.org/sites/default/files/publications/ffrdc-primer-april-2015.pdf`, 2015.

[5] R. Westrum. A typology of organisational cultures. *BMJ Quality & Safety*, 13(2):22–27, 2004.