2016 Federal DevOps Computing Summit Report

Michelle Casagni, Melissa Heeren, Diane Hanf, Michael Kristan, Susan Kuwana
The MITRE Corporation[1]

Tom Suder and Tim Harvey
The Advanced Technology Academic Research Center

December 9, 2016

[1] The authors' affiliation with The MITRE Corporation is provided for identification purposes only, and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions or viewpoints expressed by the authors.

Table of Contents

1 Executive Summary

An inaugural ATARC (Advanced Technology Academic Research Center) Federal DevOps Summit was held on August 18, 2016 in Washington, D.C. During this summit, four 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.

Participants ranged from Director, CIO, 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 drive the discussions with session participants towards addressing challenge areas in DevOps and Agile, as well as identifying courses of action to be taken to enable government and industry collaboration with academic institutions.

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. The sessions identified key overarching themes which are summarized below:

Culture changes are needed to implement DevOps/Agile practices in an organization.

Predominantly, organizations need to get buy in to change from an "us vs. them" culture, and dissolve the silo mentality. Support from executive levels of the organization are necessary to show advocacy for change and be a strong change agent to show the way. It is recommended that agencies ensure there is an organizational "Champion" willing to support and mandate DevOps/Agile methodologies and tools. It is also recommended that an organization-wide set of polices, guidelines, and language specific to DevOps/Agile be standardized and implemented in a consistent and repeatable manner.

Appropriate roles and training are necessary for successful DevOps/Agile adoption.

The iterative and rapid pace of DevOps and Agile processes requires multi-disciplinary teams working collaboratively together. Often times testers and developers are working side-by-side and some situations the different roles have competing objectives even though end goal is a shared responsibility. Several recommendations include ensure the correct roles are identified and supported with training, offer a rotation of roles in a "safe" environment to allow teams to experience the perspectives of different job functions, and pair developers with testers, operations, or security members.

The Product Owner is a critical role in a DevOps/Agile development environment. It is evident that in many organizations, Product Owner is assigned as an “on the side” role without an understanding of the level of effort and commitment necessary for success. The lack of a trained Product Owner lowers the likelihood of project success. A recommendation is to create a specific billet for the role of Product Owner, or that product owner responsibilities be mapped clearly to an existing billet. Furthermore, a Product Owner in a Federal Agile/DevOps program needs to be a government employee empowered to make decisions regarding the prioritization of user stories and the determination that the solution demonstrated meets the definition of done.

It was also recognized that training approaches, both formal and informal, assist in reinforcing the culture to support DevOps/Agile success. Upper management should receive appropriate training or briefing to understand impacts of moving to an Agile/DevOps culture. Another suggestion is that DevOps/Agile training become part of standard Product Manager training and certification (e.g., Defense Acquisition University (DAU), Federal Acquisition Institute Training Application System (FAITAS), Department of Homeland Security (DHS) Performance and Learning Management System (PALMS), etc.).

Security requirements are critical, yet onerous, and often not incorporated at the beginning of the development lifecycle.

This is concurrence that security requirements, while critical, are onerous in their current form. Stringent security and privacy related policies, requirements, paperwork and scanning requirements add significant burden to the DevOps process. Recommendations to incorporate into the DevOps process to ease the challenges of security compliance include:

- Build strong feedback loops into the software lifecycle that has shared visibility across developers, operators, and security professionals
- Elevate the cost of security vulnerabilities in planning and estimation
- Increase auditing of all software and configuration changes
- Treat security as another form of quality testing and assurance
- Adopt a test driven software development strategy with security acceptance criteria
- Define security as a functional requirement in software development instead of treating it as a non-functional requirement

Many existing policies and procedures are outdated in the sense that they better align with waterfall processes instead of Agile methodologies. Security needs to be talked about at the architecture level (and at every stage) in the software development lifecycle and that money and time should be devoted to secure development.

2 Introduction

An inaugural ATARC (Advanced Technology Academic Research Center) Federal DevOps Summit was held on August 18, 2016 in Washington, D.C. During this summit, four 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.

Participants ranged from Director, CIO, 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 drive the discussions with session participants towards addressing challenge areas in DevOps and Agile, as well as identifying courses of action to be taken to enable government and industry collaboration with academic institutions.

The MITRE Corporation is a not-for-profit company that operates multiple Federally Funded Research and Development Centers (FFRDCs)[2]. ATARC is a non-profit organization that leverages academia to bridge between government and corporate participation in technology[3]. 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 regarding a specific challenge area. The sessions for this summit addressed:

- Nuts and Bolts of Culture Change in the Government Space
- Product Owner Expectations
- DevOps Reality
- Secure Development Operations (SecDevOps) – The Intersection of Security, Development and Deployment

[2] https://www.mitre.org/about/corporate-overview
[3] http://www.atarc.org/about/

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

3.1 Nuts and Bolts of DevOps Culture Change in the Government Space

This session focused on sharing current challenges in changing longstanding cultures to embrace Agile and DevOps and discussed the Do's and Don'ts from agencies working to enact this culture change.

3.1.1 Session Goals

- Share current challenges in changing longstanding cultures to embrace Agile and DevOps
- Identify behaviors to embrace and avoid based on experiences of Agency/Department that are working to enact this culture change

3.1.2 Session Summary

Session participants started out by identifying what they felt needed to change. Predominantly, organizations needed to get buy in to change from an "us vs. them" culture, dissolving the silo mentality and reaching out across Development, Operations and User communities to embrace an environment that supports a DevOps culture. To do this, participants identified they needed support from the executive levels of the organization, showing advocacy for change, and a strong change agent to show the way. Participants felt they needed cultural messaging: definitions, process of DevOps, meaning (outcomes) of DevOps, showing by action what it means to be part of a DevOps environment, and how it aligns with organizational goals. The group also felt they needed focus on getting the supporting information technology (IT) resources to foster and nurture the environment: knowledgeable DevOps practitioners, tools, lifecycle framework, and education plan to show program managers and other stakeholders how to collaborate and plan differently. For example, events as hackathons expose people to how DevOps can work, helping them to ease their fears of migrating.

3.1.3 Recommendations

Session participants then focused on more details for getting people to change and maintaining that change especially when they have been "raised" in a culture that is optimized for silo thinking. The group identified a predominant set of "Ps" that must be addressed and aligned to initiate the culture change:

Principles—understand they are working together using the same principles. Several agencies have DevOps projects under way with full support across the spectrum. They are being successful by keeping focus on the goal and delivering quality code in a user-responsive manner.

Practices—identify those key practices that are markers of the DevOps culture such as coming to the table to positively address issues as they arise, leaving out the finger-pointing and focusing on solving the problem, and "doing" vs. "being" are positive. More importantly, embrace a scientific mindset by holding technical exchange meetings, Lightning talks, or coffee talks for sharing, cross-pollinating and co-mingling across stakeholder disciplines to promote a shared understanding about DevOps.

Products—automate the DevOps process to meet your delivery goals ensuring that it not only reflects the code and user interfaces developed, but it also provides history for retrospectives and supports the desired habits of the culture.

Policy—set policies that propel people towards acting in the way you want them to behave.

People—ensure that the participants are able to participate. In the case of using DevOps that is supporting Agile development ensure the correct roles are identified and supported with training.

Pain Points—address pain points as soon as they arise and do it in a positive, nurturing manner.

Promotion—promote the benefits of DevOps and continue to reinforce them positively.

Perspective—understand that each participant in the DevOps comes from a different environment with differing perspectives. A great practice to keep forward momentum is to keep the perspective on meeting a delivery goal and contributing to the success of the mission, i.e., measuring value vs. velocity.

Perception—change perception of leadership, help them see that in a DevOps context, IT is a customer that is there to ensure that mission/organizational goals are met. A technique for altering perception is to rapidly move teams through a three phase process to achieving success: alpha, beta, live. The Alpha phase is designed to promote experimentation and exploration giving DevOps team members a place to learn. Beta allows them to do a meaningful prototype that will have value and promotes discussion on the users' true needs. Then as the requirement is better understood, go live using the DevOps environment to assure quick delivery with low quality issues. Doing this often and providing transparency to quick wins to as many stakeholders as possible also helps to educate the community on the shift in the DevOps delivery paradigm.

Lastly, though not a "P", the group identified that there is a need for **nucleation;** need a critical mass of expertise, tools, technologies and processes in-house to properly support a DevOps culture. The group concluded that the pillars of culture change success required an organization to show commitment by insisting on alignment to the mission, ensuring adequate budgets for the DevOps environment, supporting continuity in projects that are executing using DevOps and establishing a healthy, highly involved DevOps community.

The group transitioned to talking about the "Ts" of DevOps culture change. Not to be understated, is the need to have **training** to help with culture change. The group identified several training approaches, both formal and informal, to reinforcing the culture to support DevOps success. Amongst those approaches are initiating the organization by resetting them on the purpose of DevOps, standardizing organization-wide on DevOps language, understanding that the organization must migrate through a set of stages: listening-to-mindset-to-culture, and deliberately usher the organization through these stages to solidify the culture change.

Having a Resource Center in place where teams can get coaching, courses, or mentors is very useful for supporting culture change. Continuously communicate the approach;

ensure that that participants understand that they are focusing on quality deliveries at a pace faster than they may have experienced, and last but not least, make and take time to train across teams so that everyone can better understand how the other stakeholders play into the process.

As the discussion continued the group finally arrived at the nexus of Agile and DevOps. Some felt Agile should be kept out of the DevOps conversation, many more felt that DevOps was a critical contribution to building quality into Agile development. Discussions moved to a use case for DevOps in which one organization was trying to determine how to reduce a 187- day window to make a single line of code change. DevOps could certainly reduce that window and the group thought that showing that window reduction using DevOps would be a huge boost towards acceptance and garnering support for culture change.

The group also discussed an approach for demonstrating how DevOps could help improve delivery and quality by demonstrating improvement through <u>small</u>, <u>highly focused</u> demonstrations where performance is baselined and improvement is shown. This will allow an organization to easily **track** their progress as they embrace DevOps. Continuing along the thread of measuring success an organization should set **thresholds** which are immediately addressed if falling behind. Measurement and accountability are fundamental to DevOps culture.

From a **tools** perspective, group members with more experience felt tools were enablers to the culture, allowing organizations to empirically collect data on performance, improve delivery speeds, and get early feedback on defects with quick fixes when it is cheaper to fix them. Once the DevOps process is in place, a recommendations was tools should be chosen through experimentation to determine the best fit with the DevOps environment that has been established. Collaboration session participants also shared wisdom to stay technology agnostic when selecting tools, once again, selecting ones that make sense for their environment.

Techniques that were emphasized include removing focus from “us vs. them”, creating empathy beyond what each DevOps team member knows by deploying training techniques that allow for safe swapping of roles during team training or in real situations where the duties to be done by the learner are highly scripted, supporting high awareness of the day-to-day experience from which the process can recover if the task at hand does not go as planned. These techniques tend to enlighten team players as to how their contributions can be improved to help other team players do their jobs.

The other “Ts” discussed include:

> **Testing**—expect that there will be brownfield testing of capability. Rely on usage to provide empirical evidence of cosmetic preferences system-wide that can be rapidly incorporated back into future releases. This technique however, should not be used for critical functionality.
>
> **Teaming**—employ, select or nurture people who are highly collaborative, inquisitive, talented, and they with naturally sustain the culture that supports

DevOps success. Set their focus on collective code/product ownership and getting it through the DevOps process; this is a key enabler.

Tangle of Security—get rid of rework due to security; involve security early and often and team with security to determine how to attain mission needs securely.

Transparency—using Kanban and other tools to ensure that members across the team understand the team process, how they can adjust, and where improvements can be made.

Last but not least, shape your organizational structure the way you want your code to be developed and the agility you expect your business/mission to have (Conway's Law); at the heart of success should be to respond to the needs of the mission product users, remove burden from use of the product and remove redundancy in the process to allow for speedy delivery.

3.2 Product Owner Expectations

This session focused on the role of the Product Owner in a DevOps environment, and how leaders can transition from other traditional Program Management roles to become a Product Owner.

3.2.1 Session Goals

- Identify Product Owner responsibilities
- Compare Program/Project manager tasks with Product Owner tasks
- Identify the biggest challenges that Product Owners in the Federal Space face
- Determine what is necessary to make a Product Owner successful

3.2.2 Session Summary

This session began with introductions of participants, where participants identified their current role and their goals for the session. There was a great diversity of experience with DevOps, and with the Product Owner role in an Agile/DevOps environment. As the discussion progressed, several takeaways emerged:

3.2.2.1 Training is needed for successful Agile Product owner

Several participants had been placed in the role of "Product Owner" in an "Agile" development environment with no prior experience in Agile. The lack of understanding of the basics of the Product Owner role clearly lowered the likelihood of success for these individuals in that role. Because the role of the Product Owner is so critical in an Agile development environment, this lack of a trained Product Owner similarly lowers the likelihood of project success. The group recommended that Agile training become part of standard Product Manager training and certification (e.g., Defense Acquisition University (DAU), Federal Acquisition Institute Training Application System (FAITAS), Department of Homeland Security (DHS) Performance and Learning Management System (PALMS), etc.).

3.2.2.2 Clarity of roles

Some of the participants self-identified as "project manager/product owner/scrum master." As the discussions progressed, it became clear that the lack of clarity of what a role entails and clear assignment of those responsibilities to a team member lead to confusion and overload. This leads to the next takeaway.

3.2.2.3 Appreciation by upper management of role of product owner as a real job

In Agile development and DevOps culture, the role of Product Owner is critical. Yet there was evidence that in many organizations, upper management assigns someone, or multiple people to that role, without an understanding of the level of effort and commitment necessary for success. In several organizations, the Product Owner has a different primary job and is performing the Product Owner role "on the side." The group recommended that a specific billet be created for the role of Product Owner, or that product owner responsibilities be mapped clearly to an existing billet.

3.2.2.4 Product Owner needs to be government/full-time participation/single point of contact /Empowered

The Product Owner in a Federal Agile/DevOps program needs to be a government employee empowered to make decisions regarding the prioritization of user stories and the determination that the solution demonstrated meets the definition of done. The Product Owner validates the business needs. He or she is a subject matter expert who works closely with users, that the customer (who understands the product owner role) respects. In the past, the customer has not had a role in IT development. This change and level of involvement required is not well understood. A strong Product Owner can clarify these roles and set expectations.

3.2.2.5 Role doesn't change significantly with DevOps

DevOps can be defined as a deliberate collaboration between an operational team and the engineers developing capabilities to meet a customers' needs. While many processes are automated in a DevOps environment, the role of the Product Owner remains the same: prioritize the backlog, and validate the solutions demonstrated by the teams. The difference is that with DevOps, the tangible outcomes are visible sooner.

3.2.2.6 Focus on roles, not the terminology

In some organizations, there is a focus on getting positions staffed; an Agile project must have a Product Owner, so a Product Owner is named. It is critical that the role itself and the value it brings to the Agile team is understood and appreciated. Without that, focusing only on the terminology, an organization can end up with someone who is Product Owner "in name only".

There is a lot of confusion about the difference between product owner and project manager role. These roles are very different, but often a project manager will be reassigned as Product Owner. In this situation, it can be helpful for an organization that is committed to transitioning to Agile and DevOps to bring in an Agile Coach to work with product owners.

3.2.3 Recommendations

- Upper management receive appropriate training/briefing to understand impacts of moving to an Agile/DevOps culture

- Careful consideration to be made when selecting Product Owner; consider end-user relationships, interpersonal skills, and long-term availability and ability to commit
- Train Product Owner in Agile, DevOps
- Consider small project to pilot DevOps, with full support of upper management.

3.3 DevOps Reality

This session focused on the policy, cultural and technical challenges and solutions for implementing DevOps methodology and tools in the federal government.

3.3.1 Session Goals

- What can "continuous" integration and deployment look like in your Agency?
- You don't have to be Google to embrace DevOps culture

3.3.2 Session Summary

This session began with the agency representatives discussing a wide range of DevOps experiences within their organizations. While each organization goes through its own journey, there are common challenges and obstacles that the organizations encounter. The common challenges with implementing DevOps can be categorized into Culture, Testing and Security, and DevOps Policy.

The **culture changes** needed to implement Agile/DevOps practices in an organization cannot be overstated. The group identified lack of knowledge, fear of change and organizational stovepipes as challenges. One organization stated that a clear lesson learned was that the first step with DevOps should be to establish an overall goal that addresses cultural change. The group recognized the difficulty of changing the way people think about their processes and systems.

The group switched to ideas about breaking down organization stovepipes and discussed ideas about rotating developers around the operations jobs and skills and rotating on-call responsibilities. It is important to provide opportunities for developers to see and experience operational problems first-hand to improve their understanding. Hackathons have proven to be valuable ways to educate developers and support cultural changes while concurrently getting some valuable development done.

The discussion shifted to the **testing and security** challenges for organizations. DevOps tools can provide transparency on what exactly the change(s) are at a much great level of fidelity and automation than previously possible. However, existing configuration management, change control boards, testing practices, security assessment and release management processes pose challenges to the goal of rapid deployments. These processes and policies need to be more agile and adapted to the opportunities that the cloud and DevOps practices provide.

The group had good discussion about the impacts of testing and Independent Verification and Validation (IV&V) on DevOps practices. Questions about where and how often IV&V fits into the DevOps pipeline process were raised. The importance of configuration reports on the build package details and automated test result reports were essential. The goal of DevOps is that everything is code (software, infrastructure, test, security).

The group discussed that daily releases are possible and even multiple releases in a day could be achieved. The keys to these rapid releases are that each release is small with phased gate reviews, automated reviews, and quality standards. The idea of reducing and condensing these various processes into System Level Agreements (SLAs) was discussed. SLAs could specify that deployments do not wait for weekends or nights and they should include rollback strategies to clearly identify what happens if something goes wrong on a deployment. One organization described a daily change management process that they proved out as a pilot. The discussion included the benefits of containerization that allows software packaging, version control, and deployment via containers that can be version controlled in a repository. The whole infrastructure can be rolled back via the container, if needed.

There was concurrence that security requirements, while critical, are onerous in their current form. Stringent security and privacy related policies, requirements, paperwork and scanning requirements add significant burden to the process. The group discussed the opportunities that DevOps methods and tools offer to ease the challenges of security compliance. Possible solutions include automating security controls, shifting static security methods to dynamic security concepts and tools.

The importance of establishing effective **DevOps policy** was another topic. One organization described how a project started off as mobile, morphed into DevOps with automation and extended to continuous integration. This organization successfully established a policy to identify these processes and require that applications comply with this new policy. Another organization described their mission to build and deploy new applications within a portfolio. Without well-defined policies and standards, the result was a huge mix of approaches, tools and inconsistent deployment outcomes.

The discussion turned to design guidance and the current need to decouple the application lifecycle from infrastructure lifecycle. End users do not care about application versus infrastructure; they only care about the service. Audit trails of user actions can help. Auditing and logging are part of systems architecture and should be some of the policies defined at the beginning.

3.3.3 Recommendations

The recommendations from this session are categorized as follows:

3.3.3.1 Changing Organizational Culture - "Integrate" Don't "Separate" - Essential for DevOps

- Ensure there is an organizational "Champion" willing to support and mandate DevOps methodology and tools
- Organize the teams in different ways, e.g., a matrix team approach (development, testing, security, system/database administrators, and business) or team rotations can extend existing expertise and expand the general skill levels

- Pair operations staff with the developers. Increase skill levels and round out knowledge for both operations and development staff through rotations
- Pair developers together as they tend to be driven by efficiency, but will bring different strengths and focus areas. Pairing can help fill in the gaps and smooth out the skill variances
- Add a security person to DevOps team to ensure the upfront security focus and compliance is established early in the lifecycle.

3.3.3.2 Test & Security – "Automate, Automate, Automate" - Mandatory for DevOps

- Ensure code standards are established and conduct code reviews to validate compliance as important steps before automated testing can be effective
- Establish a test pyramid. Testers need to be knowledgeable about unit tests. On the functional side, testers should be included early in the development process
- Test Driven Development (TDD) is essential. Know what is being built so that tests are developed prior to the first line of code
- Developers need to start asking how they are testing the code, not just how they are developing it. Developers may feel the unit tests will slow them down, so more education on the entire process and positive impacts of upfront testing
- Behavioral driven development is another effective approach that facilitates early collaboration and better understanding of test criteria. Provide and utilize business scenarios that can be tested
- Be attentive to the ratio of developers to testers. Begin with a 1:1 ratio of developers to testers. The ratio can lessen for testers as more test automation and the functional and regression suite grows
- Property based testing (generative) is another effective approach, e.g., typically test a range, test inner and outer boundaries; property based testing addresses tests that provides all the values
- Implement continuous monitoring and alerts for non-compliance with security requirements
- Implement a "Shift Left" in the software development lifecycle for security compliance testing to accelerate security compliance and improve software quality upfront in the lifecycle.

3.3.3.3 DevOps Policy

- Establish and mandate organization-wide policies and guidelines to effectively implement DevOps in a consistent and repeatable manner
- Use effective design principles for micro-services: simplify, loosely couple, remove dependencies, remove technical debt
- Make legacy applications small by keeping the data inside, have an API layer, and use micro-services to break down functional areas
- Change acquisition requirements from software requirements to delivery of development teams and skillsets.

3.4 Secure Development Operations (SecDevOps) – The Intersection of Security, Development and Deployment

This session focused on understanding where security fits into a DevOps model and how to continuously modernize applications in the federal government with security baked in.

3.4.1 Session Goals

- Where does Security fit in a DevOps model?
- How do you continuously modernize applications with security baked in?

3.4.2 Session Summary

This session began with a general discussion of DevOps and what it means to each organization. Given that this session has a focus on security, most of the participants came to the discussion with either a background in security or an interest in trying to make DevOps work in an environment that has strong requirements (compliance, accreditation, policy) that constrain the process. The discussion took a number of turns and tangents throughout the time period, but revolved around three major topics or themes: addressing the culture gaps, building trust in software products, and refining processes and standards.

Looking at the cultural gaps, the group agreed that security professionals and software developers don't always share the same goals even though security is a shared responsibility. A lot of developers lack backgrounds in security and systems administration. Because DevOps teams may not have security engineers assigned to a team, developers need to become more aware of security and to build security decisions in the process. The group discussed training strategies to make developers and operators more aware of a compliance mindset and the risks of having elevated permissions, but acknowledged that training is difficult when teams frequently rotate out. The other methodology discussed is to bring security engineers and auditors into teams early on in the process and to reconcile that different organizations have different risk factors.

On the theme of building trust and continually monitoring for threats, the discussion started with the observation that many agencies and organizations are relying more on open source and more complicated code bases that make up lots of micro services and libraries, possibly from third parties. The session participants posed the question of how government organizations are ensuring the integrity of this virtual supply chain of software. There is a constant reuse of untrusted code and the concern is that software engineers and software developers don't go the extra step to look up security of third party libraries. Security professionals, assessors, and auditors pointed out that if the platform or repository isn't integrated or trusted, then you might not be able to ensure security of the system. Discussion continued with hypothetical scenarios of potentially blocking the internet access to software developers or as a last drastic measure, creating a white list of application programming interfaces (APIs) and libraries as part of enterprise architectures.

Lastly, the group discussed aspects of refining their own organization's processes and standards to accommodate both DevOps and the increasing velocity of security activities. Many existing policies and procedures are outdated in the sense that they better align with waterfall processes instead of Agile methodologies that are attempting to rapidly deploy a product that meets stakeholder requirements. The concern is that the pace of innovation is so much faster that the security community cannot catch up. The group voiced the opinion that security needs to be talked about at the architecture level (and at every stage) in the software development lifecycle and that money and time should be devoted to secure development. Different strategies were discussed such as thread modeling, secure testing, and applying the Risk Management Framework (RMF) into the DevOps process. The concern was how these security activities will scale. Working against well-defined standards such as NIST Special Publication 800.53 and OSCAL may assist with those challenges.

3.5 Recommendations

Following the themes of culture, trust, and processes/standards, the recommendations are broken up into these categories:

3.5.1 Addressing the gaps in culture

- Build strong feedback loops into the software lifecycle that has shared visibility across developers, operators, and security professionals
- Elevate the cost of security vulnerabilities in planning and estimation
- Provide full stack training to all team members and give them strong ties to academia to understand emerging trends
- Increase transparency and integration of IT Service Management (ITSM) systems into DevOps
    - Give security professionals access to developer work logs and flows
    - Give developers access to IT Information Library (ITIL) portals
    - Tie all products and tools together to include source code management (SCM), build servers, and monitoring systems

3.5.2 Building trust in software products

- Digitally sign all code and verify checksums of all libraries that are being leveraged in software
- Use plugins in Integrated Development Environments (IDEs) that in real-time, flag code for security warnings
- Leverage stateless architectures coupled with immutable architectures that lower risk of compromise
- Increase auditing of all software and configuration changes
- Treat security as another form of quality testing and assurance

3.5.3 Refining processes and standards

- Define security as a functional requirement in software development instead of treating it as a non-functional requirement
- Increase transparency in software development lifecycle so that all stakeholders have visibility
- Adopt a test driven software development strategy with security acceptance criteria
- Automate security processes to make assessments repeatable and scalable
    - Assess and accredit the process (early-on) of continuously delivering software instead of assessing the end product
- Deploy a robust logging infrastructure

    - Centralize logging so that ephemeral services don't need to persist state and logs
    - Make application logging a security requirement up front
- Build abuse cases in addition to use cases to tease out malicious users
- Leverage automation of documentation whenever possible

4 Conclusion & Summit Recommendations

The August 2016 Federal DevOps Summit highlighted several challenges facing the Federal Government's adoption of DevOps and Agile practices. A common set of themes emerged across the individual session: cultural barriers to adoption, lack of appropriate training, and security requirements. Success stories are coming from government adoption efforts and with continued collaboration and sharing best practices and recommendations for mitigation these challenge will evolve.

The following are the key overarching themes from the four sessions.

Culture changes are needed to implement DevOps/Agile practices in an organization.

Predominantly, organizations need to get buy in to change from an "us vs. them" culture, and dissolve the silo mentality. Support from executive levels of the organization are necessary to show advocacy for change and be a strong change agent to show the way. It is recommended that agencies ensure there is an organizational "Champion" willing to support and mandate DevOps/Agile methodologies and tools. It is also recommended that an organization-wide set of polices, guidelines, and language specific to DevOps/Agile be standardized and implemented in a consistent and repeatable manner.

***Appropriate roles and training are necessary for successful DevOps/Agile adoption*.**

The iterative and rapid pace of DevOps and Agile processes requires multi-disciplinary teams working collaboratively together. Often times testers and developers are working side-by-side and some situations the different roles have competing objectives even though end goal is a shared responsibility. Several recommendations include ensure the correct roles are identified and supported with training, offer a rotation of roles in a "safe" environment to allow teams to experience the perspectives of different job functions, and pair developers with testers, operations, or security members.

The Product Owner is a critical role in a DevOps/Agile development environment. It is evident that in many organizations, Product Owner is assigned as an "on the side" role without an understanding of the level of effort and commitment necessary for success. The lack of a trained Product Owner lowers the likelihood of project success. A recommendation is to create a specific billet for the role of Product Owner, or that product owner responsibilities be mapped clearly to an existing billet. Furthermore, a Product Owner in a Federal Agile/DevOps program needs to be a government employee empowered to make decisions regarding the prioritization of user stories and the determination that the solution demonstrated meets the definition of done.

It was also recognized that training approaches, both formal and informal, assist in reinforcing the culture to support DevOps/Agile success. Upper management should receive appropriate training or briefing to understand impacts of moving to an Agile/DevOps culture. Another suggestion is that DevOps/Agile training become part of standard Product Manager training and certification (e.g., Defense Acquisition University (DAU), Federal Acquisition Institute Training Application System (FAITAS), Department of Homeland Security (DHS) Performance and Learning Management System (PALMS), etc.).

Security requirements are critical, yet onerous, and often not incorporated at the beginning of the development lifecycle.

This is concurrence that security requirements, while critical, are onerous in their current form. Stringent security and privacy related policies, requirements, paperwork and scanning requirements add significant burden to the DevOps process. Recommendations to incorporate into the DevOps process to ease the challenges of security compliance include:

- Build strong feedback loops into the software lifecycle that has shared visibility across developers, operators, and security professionals
- Elevate the cost of security vulnerabilities in planning and estimation
- Increase auditing of all software and configuration changes
- Treat security as another form of quality testing and assurance
- Adopt a test driven software development strategy with security acceptance criteria
- Define security as a functional requirement in software development instead of treating it as a non-functional requirement

Many existing policies and procedures are outdated in the sense that they better align with waterfall processes instead of Agile methodologies. Security needs to be talked about at the architecture level (and at every stage) in the software development lifecycle and that money and time should be devoted to secure development.

5 Acknowledgements

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 FedSummits web site[4].

[4] http://fedsummits.com/devops/august-2016/agenda/