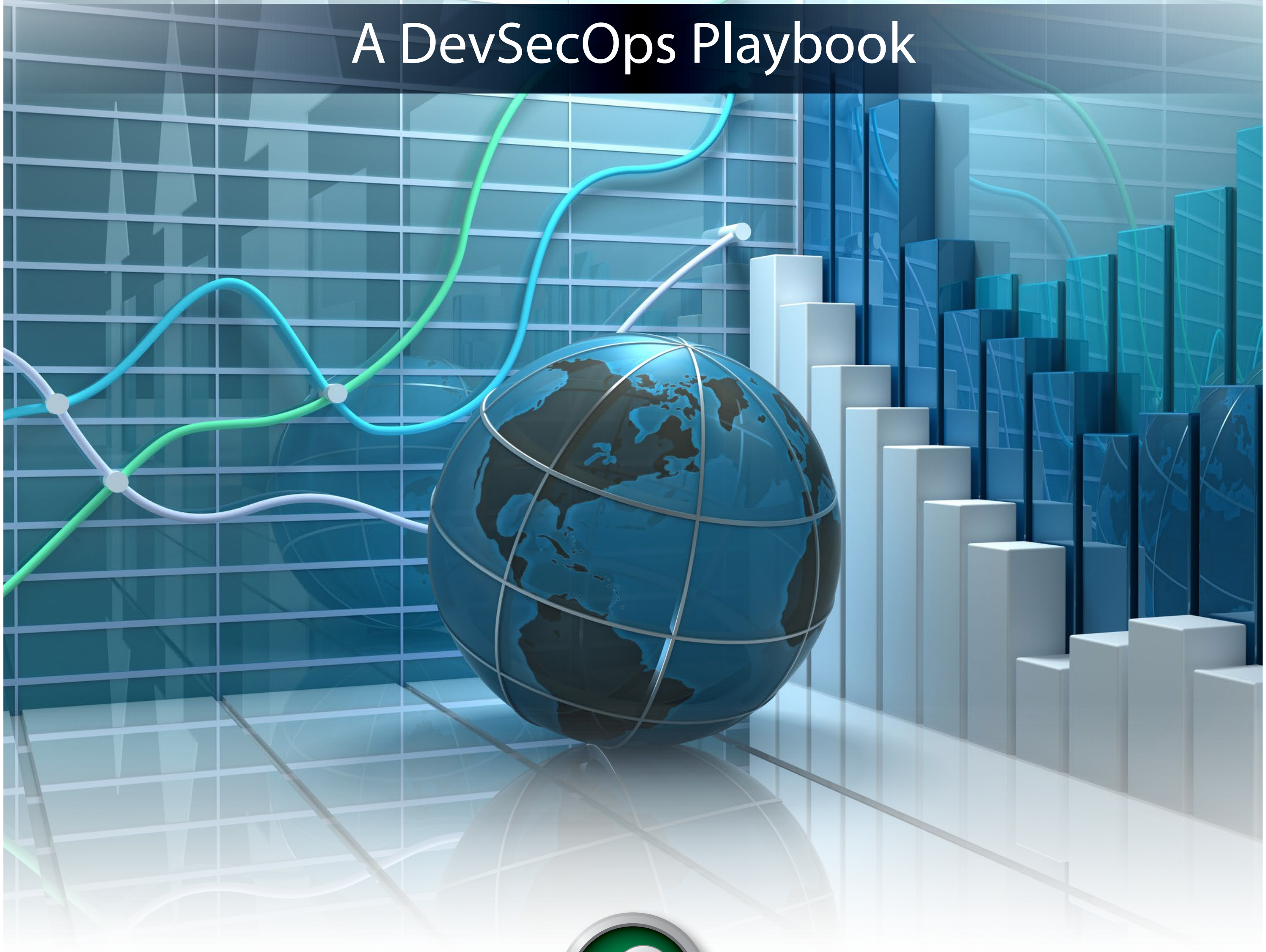

# A DevSecOps Playbook

**A SANS Whitepaper**

*Written by Dave Shackleford*

March 2016

*Sponsored by*
*CloudPassage*

# Executive Summary

Enterprise computing is going through a major transformation of infrastructure and IT delivery models, one that is at least as disruptive as the move from mainframe computing to client/server (Internet) architectures. With client/server architectures, the change in hardware was the most obvious difference, but the more meaningful transformation was IT organizations' new ability to build custom systems and software much more quickly, with far greater flexibility and at lower cost than had been possible during the mainframe era.

Today, the dramatic shift toward the dynamic provisioning, shared resources and open standards of cloud computing models has, as with client/server, been driven largely by the benefits to be gained, in this case in speed, agility and cost through the judicious use of cloud systems and services. As before, changes in hardware and computing architecture are the most outwardly visible. But what promises to deliver the most significant payback, in terms of higher-quality software that is delivered faster than was possible before, are the less obvious benefits—in speed, flexibility and deployment methods—that are possible using DevOps and other rapid- or continuous application-delivery models.

*Security teams are often seen as roadblocks to rapid development or operations implementations, slowing down production code pushes.*

For business leaders, moving to the cloud is all about speed of service delivery, flexibility, scalability and new capabilities and features that would take too long to develop in-house. The ability to deploy applications and services into cloud environments is pushing all facets of IT to move faster than ever before. Unfortunately, despite interdependence on one another, development teams, IT operations and information security personnel often operate in their own silos. The alignment of development and operations teams has made it possible to build customized software and business functions far more quickly than before, but security teams continue to be left out of the DevOps conversation.

Security teams are often seen as roadblocks to rapid development or operations implementations, slowing down production code pushes. As a result, security organizations will likely have to change so they can fully support and facilitate cloud operations.

This paper will argue that DevOps and information security can co-exist through the application of a new approach referred to as DevSecOps.

## DevOps Versus Security

Leaders of information security organizations too often see commercial cloud services as a defeat for corporate security. That viewpoint puts them at odds with application development groups that see the continuous release of new features and functions as a major advantage. For them, anything that slows the pace—including information security—inhibits the organization's ability to compete.

Organizations can find it hard to resist becoming more agile by accelerating the rollout of innovative new applications. But when an organization speeds up development by skipping steps meant to identify flaws in untested code or resisting efforts to add security controls to the code before it ships, it exposes itself in ways that would have been unthinkable under traditional development models.

*When an organization speeds up development by skipping steps meant to identify flaws in untested code or resisting efforts to add security controls to the code before it ships, it exposes itself in ways that would have been unthinkable under traditional development models.*

The cloud itself is rapidly scalable, but that does not mean the development of applications to run on it can match the pace. For example, even perfunctory testing might reveal that a poorly designed application cannot scale on its own, a problem that can drive up costs because the application will use the cloud's endless supply of compute cycles to keep itself running at an acceptable pace.

Most continuous development methods include steps designed to allow developers to fix coding errors or small bugs as they happen rather than as they are executed in the field. They are not, however, designed to catch major architectural or security flaws.

And even if rapidly developed code is perfect, data, applications and systems designed for closed, trusted data center environments can still be vulnerable to configuration errors, especially those resulting from uncertainty about the version of a product or technology being used in the cloud—a frequent problem in an environment designed to make such details invisible. The virtual servers that support Amazon Web Services' Elastic Compute Cloud (EC2), for example, run on the open-source Xen hypervisor. Not many software-based security products have been successfully ported to formats that allow them to run as virtual machines on Xen hypervisors, and those that have been are probably going to run only on plain-vanilla Xen hypervisors, not the customized versions that AWS uses for EC2.

When a flaw does appear, such as the one that forced a reboot of about 10 percent of AWS' Xen servers in September 2014, Amazon notifies customers that the servers are going down but cannot put other organizations at risk by announcing why, according to Network World.[1] Without that information, it is impossible to tune new code to run more smoothly or add controls that address weaknesses that developers cannot see.

[1] "What happens inside Amazon when there's a Xen vulnerability," Network World, March 3, 2015, www.networkworld.com/article/2892313/cloud-computing/what-happens-inside-amazon-when-there-s-a-xen-vulnerability.html

## Delivering Speed—and Control

If organizations are not going to let information security slow down the business, then information security needs to find a way to embed security controls and monitoring into the deployment cycle. The answer to this may come from automation, which many enterprises are focusing on more heavily than in the past in their data centers.

The rise in automation has come as development and operations teams have started to collaborate more often and with much more rigor. This trend has led to a movement of sorts known as DevOps, which strives to foster open dialogue and intense collaboration between development and operations teams, leading to the possibility of what is referred to as the continuous delivery of code, or much more frequent code promotion than has been traditional. This collaboration can have a major effect in condensed data centers and cloud environments because it allows for many new features to be rolled out much more quickly. What are continuous delivery and DevOps exactly? The following basic definitions may help to clarify things:

- **Continuous delivery.** Small, incremental and frequent code pushes to production. Continuous delivery eschews large production code releases separated by weeks or months.
- **DevOps.** A new mode of intense collaboration between development and operations for the same goals.

DevOps strives for a number of goals and focal areas:

- **Automated provisioning.** The more automated the provisioning of resources and assets, the more rapidly the software development life cycle and operations model can operate.
- **No-downtime deployments.** Because cloud services are based on service-oriented costing models, downtime is less acceptable.
- **Monitoring.** Constant monitoring and vigilance of code and operations helps streamline and improve quality immensely. This is one of the foundations of DevOps.
- **"Fail fast and often."** The sooner code flaws can be detected, the less effect they will have in a working production environment. Rapid and almost constant testing is needed for this to happen.
- **Automated builds and testing.** More automation in the testing and quality assurance (QA) processes helps speed things up and improve delivery times.

Large cloud environments such as Amazon and Netflix push hundreds, even thousands, of code changes per day—a practice that would be considered insane under traditional IT operational standards, which required thorough testing of every patch before deployment. But that pace is required by the working model of commercial cloud services.

Information security is building on the push within DevOps teams for collaboration and automation by introducing another concept, DevSecOps. DevSecOps strives to automate core security tasks by embedding security controls and processes into the DevOps workflow. DevSecOps originally focused primarily on automating code security and testing, but now it also encompasses more operations-centric controls. Security can benefit from automation by incorporating logging and event monitoring, configuration and patch management, user and privilege management, and vulnerability assessment into DevOps processes. In addition, DevSecOps provides security practitioners with the ability to script and monitor security controls at a much larger and more dynamic scale than traditional in-house data centers.

*DevSecOps strives to automate core security tasks by embedding security controls and processes into the DevOps workflow.*

## Step 1: Assess Your Current Security Controls for Cloud

When planning for DevSecOps, security teams first need to perform threat modeling and risk assessment for the deployment types that they envision. By performing a threat modeling exercise, security teams can better understand the types and sensitivity levels of the assets they are protecting, how they will be managed and monitored in the cloud, and what the most likely threat vectors are for those assets. Once the threat landscape is understood, security and compliance teams should perform a risk assessment to better clarify what the real risks are in cloud deployments, what risks should be considered priorities, and what the potential impact and likelihood of those risks are.

The following is a short list of things to consider during threat modeling and risk assessment:

- **Most likely threats.** A comprehensive threat modeling exercise should take both insider threats (within the organization and the cloud provider environment) and external adversaries into account.
- **Data types and sensitivity.** The type of data that is stored, transmitted and processed will make a difference when assessing the risk of systems and applications in the cloud. Some data types will dictate specific security controls, as well as provisioning into compliant cloud provider environments. There may also be more (and more serious) threats in play with certain types of data (credit card, health care and personal information, for example), and the risk severity may be greater as well.

- **System builds and controls.** Configuration management and vulnerability management tools and practices play a big role in the implementation of DevSecOps, but the first step is to evaluate the current system builds in use and determine whether they need to be locked down more (or differently) when deployed in the cloud. Depending on the overall exposure of the systems, as well as the threat models developed, configuration items and hardening policies may need to be revisited.

- **Cloud environment security posture.** A major part of the risk inherent in any cloud scenario is the security posture of the cloud provider. Most reputable cloud providers offer a variety of controls attestation documents, such as the SSAE 16 SOC 2, ISO 27001 and ISO 27002 reports, or a report on the Cloud Security Alliance Cloud Controls Matrix (CCM). Security teams should review this documentation carefully when choosing a cloud provider, if at all possible.

- **Existing controls in place.** The types of controls currently in use will make a difference in how cloud operations will function. Some controls may be absolutely required, while others may not be. Making a list of controls (network access controls, host monitoring, etc.) will help security teams evaluate the cloud providers' control offerings as well as compatible third-party options.

- **Controls we lose in the cloud.** Not all controls will be available in the cloud; others will need to be adapted to function correctly in a new environment. Some controls will be available from the cloud providers themselves—AWS and other large providers offer many native encryption and key management options, for example. Other controls may have to come from new vendors or service providers or be provided and managed entirely by the cloud provider staff.

After performing a risk assessment, security teams should have a better understanding of what controls they currently have available, which ones they will have to modify when moving to the cloud and which of their concerns are the most pressing. It is almost a guarantee that some security controls will not operate the way they did in-house or will not be available in the environment of the cloud service provider. For example, many network security controls are not available to cloud tenants, and hypervisor configuration and patching is entirely up to the service providers as well.

Table 1 provides a brief breakdown (not a complete list by any means) of some of the major things to consider for risk assessment purposes in various cloud deployment scenarios.

**Table 1. Security Considerations for the Cloud**

| | Cloud Model* | | |
|---|---|---|---|
| **Security Considerations** | **SaaS** | **PaaS** | **IaaS** |
| Virtual network security | | | |
| Virtual machine instance template management | | | |
| System build configuration | | X | X |
| Anti-malware | | X | X |
| Data security, at rest and in transit | X | X | X |
| Administrative console security | X | X | X |
| Roles and privileges | X | X | X |
| Logs and monitoring for activity | X | X | X |
| Sensitive-data and policy compliance (data loss prevention) | X | X | X |

* SaaS, PaaS and IaaS are software as a service, platform as a service, infrastructure as a service, respectively.

In a nutshell, different cloud models afford tenants access to and control over different layers of the stack and the components therein. Tenants should thoroughly assess the risks of every layer that they can configure and consider controls that align with the design and operational model they have chosen for that cloud environment.

## Step 2: Inserting "Sec" into DevOps

With the rise of DevOps, most security teams tried to minimize risk by limiting the speed of change. Though minimizing risk is a valid goal, the method failed to address the requirements of extremely fast-moving, technology-dependent businesses. And it seemed likely to only become less relevant as the need for rapid adaptation spread to more organizations.

The notion of DevSecOps first appeared in a January 2012 blog entry by Neil MacDonald of Gartner. He argued that DevOps efforts should not be curtailed in favor of security, while acknowledging that security was needed. His answer was that security should be integrated into the DevOps process.[2] (MacDonald called this DevOpsSec, but most people now refer to it as DevSecOps.)

[2] "DevOps Needs to Become DevOpsSec," Gartner Blog Network, Jan. 17, 2012, blogs.gartner.com/neil_macdonald/2012/01/17/devops-needs-to-become-devopssec

MacDonald's idea was interesting, though contrary to most security organizations' long-held practices. Most promising, it seemed to hold out hope for circumventing an ever-starker choice between speed and security.

Security practitioners should focus on certain areas when looking to adapt in support of a DevSecOps practice.

## Development

Internal security teams remain the primary testers of application security (see Table 2).

**Table 2. Who Tests Application Security?[3]**

| Answer Options | Response % |
|---|---|
| Internal security team | 83.2% |
| External security consultants | 29.6% |
| Quality assurance | 22.4% |
| Development team | 21.6% |
| Security-as-a-service providers | 15.2% |
| Business unit owner | 11.2% |
| Our commercial application vendors | 5.6% |
| Other | 3.2% |

If security teams are going to be a core component of DevSecOps, they must impress upon development and operations that they can bring a series of tests and quality conditions to bear on production code pushes *without slowing the process*. If security parameters and metrics are incorporated into development and test qualifications, then the chance for security to be involved in the processes for DevOps is much higher. Security teams should work with QA and development to define certain parameters and key qualifiers that need to be met before any code can be promoted.

In addition, security teams should look to integrate automated dynamic and static code testing throughout the development and promotion life cycle to help development and security teams detect and fix any major code flaws discovered as rapidly as possible.

[3] "2015 State of Application Security: Closing the Gap," SANS Institute InfoSec Reading Room, May 2015, www.sans.org/reading-room/whitepapers/analyst/2015-state-application-security-closing-gap-35942

### Inventory Management

The first stumbling block to a mature DevSecOps implementation is the difficulty of having visibility into what is running in the environment and those assets' current state. To overcome that problem, a proper discovery process is indispensable. An effective, dynamic inventory must quickly and continuously discover and validate new assets, or changes in existing assets, as soon as they appear online. The discovery of assets depends on network scanners, system-level scanners and specialized scanning tools that can peruse files and storage infrastructure. (A thoroughly reliable inventory of assets located on cloud platforms or services can be difficult, however, because of the limitations of many tools and access methods.) Once an inventory has been created and validated, a process needs to be implemented that will continuously discover new assets (or changes in assets) as soon as they are online or shortly thereafter. The top two controls in the CIS Critical Security Controls[4]—inventorying hardware and software and maintaining a continuously updated list of assets in the cloud—are critical to ongoing security success.

*The first stumbling block to a mature DevSecOps implementation is the difficulty of having visibility into what is running in the environment and those assets' current state.*

### Configuration and Patch Posture

With a sound inventory system in place, organizations need to determine a set of configuration items that they want or need to develop and maintain. Federal agencies or organizations in heavily regulated industries may need to adhere to standards such as the DISA STIGs or CIS secure configuration guides, but most organizations are free to develop internal standards to meet their own requirements. Monitoring and policy enforcement systems are available—both agentless and agent-based—that can help organizations apply configuration standards to new systems and begin assessing the configuration of workloads or systems that deviate from policy.

Defining configuration baselines and solidifying those into practical policies that can be applied to all systems are vital steps in the proper implementation and monitoring of secure configurations. One challenge that many large organizations have had with this over the years is the sheer diversity of system types, as well as the disparity in tools that will not work on all platforms. As organizations move to the cloud, finding some homogeneity across tools and consolidating systems will prove invaluable in maintaining system security and patch levels over time, especially in a dynamic, DevOps-driven architecture.

[4] Center for Internet Security, CIS Controls for Effective Cyber Defense Version 6.0, www.cisecurity.org/critical-controls.cfm

In addition, it is vital that organizations using cloud-based instances in IaaS and PaaS environments agree on a program for template creation and maintenance that incorporates security controls into the templates and updates them frequently. In rapid deployment and scaling scenarios, security should be built in at all times and not added later, if possible.

### Vulnerability Scanning and Assessment

Related to configuration and patch assessment is the larger scheme of vulnerability scanning and assessment. In many cloud environments, this is best accomplished in one of two ways. First, some traditional vulnerability scanning vendors have adapted their products to work within cloud provider environments, often relying on APIs to avoid manual requests to perform more intrusive scans on a scheduled or ad hoc basis. The second option is to rely on host-based agents that can scan their respective virtual machines continually or as needed. Ideally, systems will be scanned on a continuous basis, with reporting of any vulnerabilities noted in real time or near real time.

### Account and Privilege Management

A critical aspect of managing security in a cloud environment is to carefully limit and control the accounts and privileges assigned to resources. All users, groups, roles and privileges should be carefully discussed and designated to resources on a need-to-know basis. The best practice of assigning the least-privilege model of access should also be applied whenever possible. Any privileged accounts (such as root and the local administrator accounts) should be monitored very closely (or ideally be disabled completely).

### Logging and Event Management

Logs and events generated by services, applications and operating systems within cloud instances should be automatically collected and sent to a central platform. Automated and remote logging is something many security teams are already comfortable with, so organizations implementing DevSecOps really just need to ensure that they are collecting the appropriate logs, sending them to secure central logging services or cloud-based event management platforms, and monitoring them closely using security information and event management (SIEM) and/or analytics tools.

## Change Detection and Automated Rollback

One major goal of both DevOps and DevSecOps is to facilitate much more rapid and scalable IT, with security controls built in and rapid execution of security processes. To scale in a large hybrid or public cloud, security will need to embrace automation, a concept that many security practitioners have been loath to embrace. For true DevSecOps to take hold, security teams will need to embed automated tests and validation of controls into the deployment cycle and monitor applications continuously in production with triggered responses that can roll controls back to a known good state, among other outcomes.

## Microsegmentation

While most network services and controls are unavailable in cloud provider environments, one technique for network isolation and policy control that is gaining significant traction is microsegmentation.

With this technique, each cloud instance adopts a "zero trust" policy model that allows for very granular network interaction controlled at the virtual machine network interface controller. This allows each cloud system to essentially take its network access control and interaction policy with it as it migrates through virtual and cloud environments, minimizing disruption and reliance on physical and hypervisor-based network technology, although software-defined networking and automation techniques can definitely play a role in microsegmentation operations.

## Step 3: Integrate DevSecOps into Security Operations

There are many application security standards and methods to choose from (see Figure 1) and several tools for helping to automate and speed up DevOps processes.

**What application security standards or models do you follow?**
***Select all that apply.***

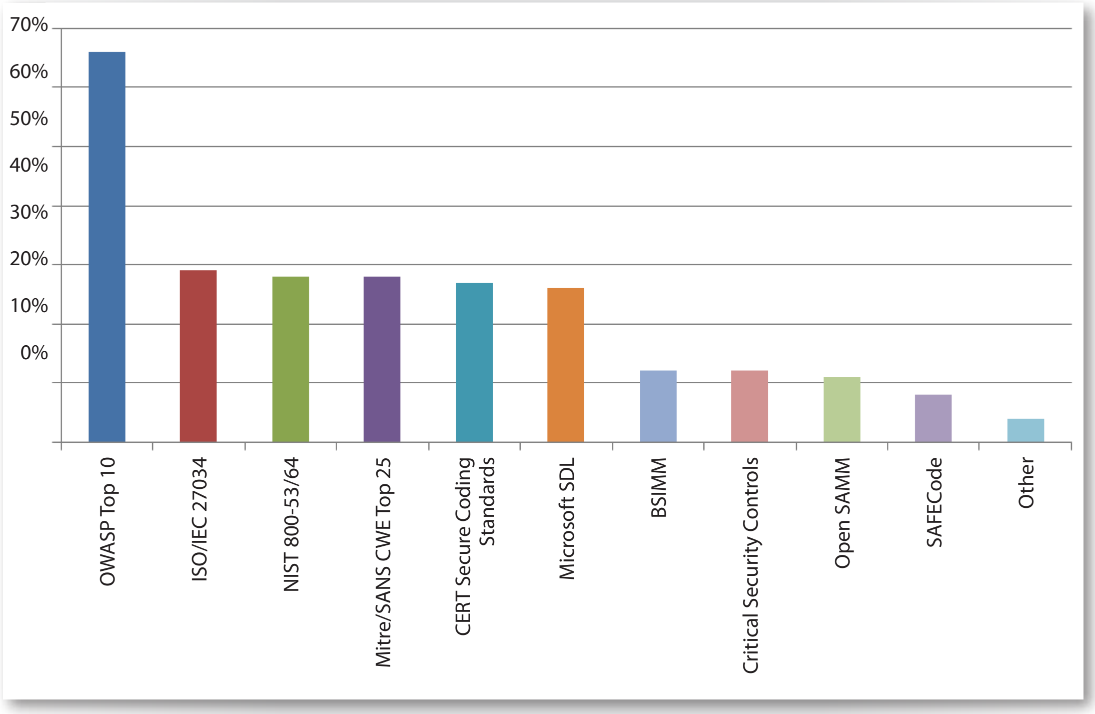

*Figure 1. Application Security Standards in Use*[5]

One of the better-known tools is Chef, a Ruby-driven framework that is used to create configuration "recipes" and "cookbooks." The goal is an increased number of automated configurations of controls across multiple servers in virtualized and cloud data centers. Chef is ideal for building automated "blueprints" of systems for rapid template-based builds, as well as recovery and cloning of environments in very short time frames.

Also popular is Puppet, a configuration management toolkit that stores "Puppet manifests," which are descriptions of systems and their state. Puppet automatically discovers and updates the system states with a tool called Facter and then creates and maintains dynamic system "catalogues" of configuration data. Many popular sites and companies use Puppet, including Disney, Google and Oracle.

[5] "2015 State of Application Security: Closing the Gap," SANS Institute InfoSec Reading Room, May 2015, www.sans.org/reading-room/whitepapers/analyst/2015-state-application-security-closing-gap-35942, Figure 7, p. 12.

One other toolkit that is becoming more popular is Salt, which strives to make the process and management of infrastructure configuration much simpler. It leverages the ZeroMQ messaging library and a central configuration library and uses Python-based modules for handling all aspects of configuration queries, responses, rapid system state gathering for immediate use, remote execution of commands and more.

There are also commercial products and third-party services that can help security teams implement automated DevSecOps by embedding simple agents into virtual machine instance templates, which then fetch policy updates upon instantiation. These systems are under the control of the operations team, but security can continuously monitor the state of the systems as well.

In 2012, researchers Josh Corman and Gene Kim,[6] both of whom are DevOps-focused analysts formerly with the 451 Group, and James Wickett of the Open Web Application Security Project (OWASP)[7] suggested an approach they called "rugged DevOps." Wickett laid out a series of steps to implement the concept in a presentation he called the "Survival Guide Pyramid." It was designed to illustrate a practical approach to secure development, at speed.

The levels of this pyramid are as follows:

- **Defensible infrastructure:** Better configuration and security controls in place, with more consistency overall.
- **Operational discipline:** Changes and code pushes managed collaboratively and with heightened awareness and interaction.
- **Situational awareness:** All changes and systems monitored proactively to determine any adverse impacts or potential attack surface created.
- **Countermeasures:** Quick response, with proper preventive controls enabled and sound communication strategy maintained.

As an example of a more automated security runbook for incident management, consider how you could build security into an incident response (IR) scenario. In a traditional IR scenario, a security team would likely want to identify an indicator of compromise (or any suspicious event or behavior that may indicate an incident), quarantine the system in some way and then perform forensics and response activities. In large, complex networks, this can be a convoluted procedure that often involves manual processes, numerous different tools and a disjointed workflow at best.

[6] "Security is Dead. Long Live Rugged DevOps: IT at Ludicrous Speed," Gene Kim LinkedIn slideshow, March 9, 2012, www.slideshare.net/realgenekim/security-is-dead-long-live-rugged-devops-it-at-ludicrous-speed

[7] "Rugged DevOps: Bridging Security and DevOps," James Wickett LinkedIn slideshow, March 31, 2012, www.slideshare.net/wickett/rugged-devops-bridging-security-and-devops

In a DevSecOps setting, the workflow would look something like this:

- Security tools detect a suspicious behavior on an instance in the cloud provider environment and trigger an automated response workflow via APIs that communicate with a DevSecOps automation engine or product.
- The network allocation of the instance is changed via scripts and API calls to a dedicated "quarantine virtual switch" in the cloud environment that has no direct Internet connectivity.
- A local process begins disk and memory acquisition on the suspect instance, which is copied to a forensic storage node in the cloud controlled by the security team and automatically protected with dedicated encryption.
- The security and operations teams can then automatically perform a rollback of the instance to a known good state (or likely create a new one from the most recent template).

*Getting started with a DevSecOps program requires a commitment to work side by side with the development and operations teams, while getting buy-in from all parties to embed security controls and processes into the entire DevOps workflow.*

## A Final Checklist

Getting started with a DevSecOps program requires a commitment to work side by side with the development and operations teams, while getting buy-in from all parties to embed security controls and processes into the entire DevOps workflow. As you get started with the risk assessment, operations and automation aspects of a DevSecOps implementation, keep the following in mind to help build a positive and productive feedback loop:

- Ensure that periodic reviews of the overall risk posture within cloud environments are performed to guarantee continued alignment of security and the other DevOps teams involved.
- Keep system instances in the cloud as locked down as you can, commensurate with the exposure and data classification types involved.
- Pay careful attention to privilege allocation and user, group and role management. This can easily creep over time in a dynamic environment.
- Commit to a culture of continuous monitoring, helping to automate detection and scripted response activities that minimize manual intervention wherever possible.
- Discuss vulnerabilities detected in cloud deployments with all team members, and make sure DevOps teams are involved in vulnerability, patch and configuration management discussions and policy creation.
- Ensure that you are gathering adequate security and operations logs and event data, sending it to a remote monitoring and collection platform.
- Discuss the changing threat landscape with DevOps teams, and solicit their feedback on practical measures that can be taken to implement the most effective security without impeding progress or slowing down the pace of business activities.

## Conclusion

The DevOps movement has just begun. Because it is in its nascent stage, information security has a window of opportunity right now to better align with both development teams and IT operations groups, whether ultra-rapid code promotion is the immediate goal or not. To accommodate a shift to DevOps, or the mentality of DevSecOps, the following should be top of mind for security teams:

- Determine the current code promotion and QA processes in place at your organization and decide where security team members can best integrate into the code development and promotion life cycle for evaluating and analyzing current practices.
- Work with business unit leaders to understand their goals as they relate to rapid development, and learn how operations and security teams can better work with programmers throughout the software development life cycle to facilitate, not hinder, these goals.
- Evaluate operations collaboration with development currently, and see where the major gaps are related to communication and ongoing management and maintenance. Record these, have conversations with all relevant teams to understand where roadblocks are and determine what actions might help overcome obstacles.
- Learn more about DevOps and major automation frameworks like Puppet and Chef, and conduct informal "Lunch and Learn" or other sessions to educate security and other teams about this methodology if it is not already underway.

All in all, security teams can be valuable team players in the DevOps movement. However, they need to adapt to more rapid changes, more flexibility and the willingness to concede "perfect security" for business benefits, with advanced monitoring and awareness after the fact.

## About the Author

**Dave Shackleford**, a SANS analyst, instructor, course author, GIAC technical director and member of the board of directors for the SANS Technology Institute, is the founder and principal consultant with Voodoo Security. He has consulted with hundreds of organizations in the areas of security, regulatory compliance, and network architecture and engineering. A VMware vExpert, Dave has extensive experience designing and configuring secure virtualized infrastructures. He previously worked as chief security officer for Configuresoft and CTO for the Center for Internet Security. Dave currently helps lead the Atlanta chapter of the Cloud Security Alliance.

## Sponsor

*SANS would like to thank this paper's sponsor:*

CloudPassage