InformationWeek
DARKReading
reports
reports.informationweek.com
December 2016

Secure Application Development: New Best Practices

The transition from DevOps to SecDevOps is combining with the move toward cloud computing to create new challenges – and new opportunities – for the information security team.

Sponsored by CloudPassage

ABOUT AUTHOR

Brad Causey is an active member of the security and forensics community worldwide. Brad tends to focus his time on web application security and penetration testing as it applies to global and enterprise arenas. He is the lead application security analyst with IBERIABANK and the president of the International Information Systems Forensics Association chapter in Alabama. Brad is also heavily involved in several projects with OWASP. He also holds many industry recognized certificates such as CISSP, MCSE, C|EH and CIFI.

EXECUTIVE SUMMARY

SecDevOps (Security + Development + Operations) is still a relatively new concept, invented in response to the nearly decade old idea of continuous development and deployment methodologies, such as Agile and virtualized infrastructure.

With the current drive for rapid development and the push to be first to market, many organizations have joined in on the "Agile" development movement. Many have had great success, but others have struggled to keep up with infrastructure demands, security concerns, user acceptance testing, and issue resolution. Because of this, the DevOps movement sought to join operations and development in a highly collaborative and joint effort to synchronize the two. SecDevOps seems like the logical next step.

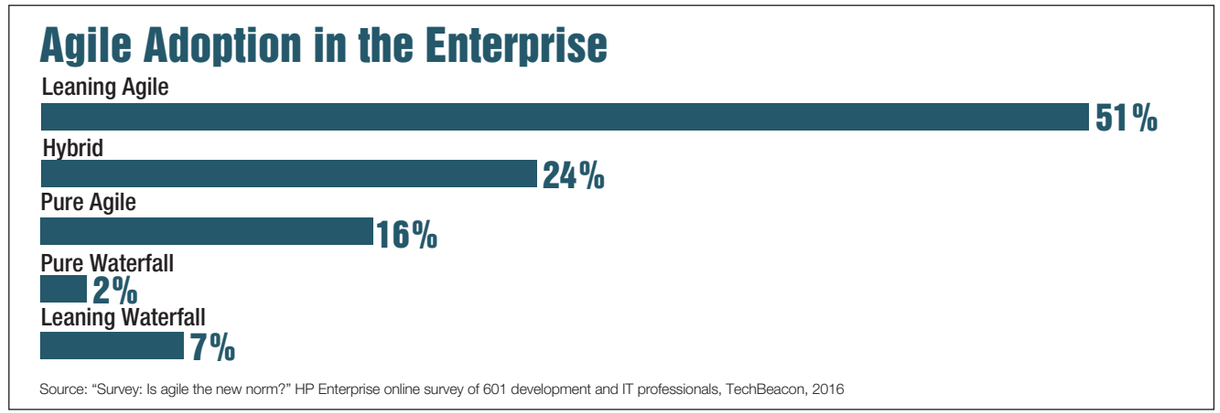
While these new trends have affected the development environment, the move toward cloud networks and services have affected the use of applications as well. Cloud-based applications became more and more common, and in many cases, outnumber traditional apps in consumption across enterprises world-wide. So, as if security departments everywhere weren't already struggling to keep up with Agile and virtualization, now we have a new challenge in cloud deployments that live mostly outside the network perimeter. These infrastructure-as-a-service, aka cloud, infrastructures have taken the world by storm, and their benefits have not been lost on the enterprise. With massive cost savings, huge uptimes and rapid provisioning, it's hard to ignore those benefits.

The combination of these technologies and processes have given organizations amazing power to rapidly push out new applications at a dizzying pace. This report will help security personnel and developers alike sort through and identify where those challenges live, and how they might be mitigated.

STATE OF SECURITY

The current state of application security is not reassuring. A recent WhiteHat Security study found at least one critical security vulnerability in 86% of all websites, and in the majority of cases, it found multiple vulnerabilities. This means that these applications are either not undergoing proper security testing, or development teams are struggling to resolve security issues prior to being pushed to production. Often, when an application security program is in place, the timeline required to complete testing and resolve security issues is incompatible with the development timeline. Setting aside weeks for the security assessment when the development effort only takes days is not realistic.

Figure 1



Primary development method used in organization across projects.

Of course, this assumes that organizations actually employ application security talent and can afford the tools to perform testing on a constant basis. This has become increasingly difficult as of late, given the lack of security talent pool. According to Michael Brown, CEO at Symantec, the demand for cyber security experts is expected to rise to 6 million (globally) by 2019, with a projected shortfall of 1.5 million. This means that even enterprises that can afford to implement application security programs may not be able to do so. The main alternative is Security as a Service (SECaaS), which can become expensive in a continuous development environment.

As if these challenges weren't enough, many applications are now being rolled out in an environment of cloud infrastructure and rapid provisioning. The relatively new ability to build out dozens of servers distributed across the globe is creating even more challenges for security teams. Servers and virtualized applications that live outside the traditional network perimeter and often share hardware and even virtualized software instances create unique challenges for traditional security teams.

SecDevOps

DevOps has created a unique application development environment whose speed has left security in the dust. With most Agile sprints lasting 1-2 weeks and the average manual application security testing period pushing between 1 and 3 weeks, it is becoming increasingly difficult to efficiently build security into applications. Further, looping the discovery

of vulnerabilities back into the development process will impact the developers' ability to stay on task with current demands. Because of this, SecDevOps was born. There are a couple of key points to address when breaking down what SecDevOps strives to achieve.

First, collaboration. Integrating the security team with the development and operations team will ensure that designs and infrastructure which may be flawed from the start can be addressed. The worst-case scenario is that components of a story (user requirement) introduces a significant security flaw (sometimes by design) and the details aren't understood by the security team until a significant amount of effort has been undertaken. By including the security team in "scrum" meetings, and working out an acceptance process for proposed changes, these security issues can be resolved before they occur. One additional benefit of including the security team is that any questions about outstanding security issues can be addressed, as well as concerns about impact and priority.

Second, the code. Traditionally, development teams have stood up test environments dedicated to user acceptance and security testing. This process is fraught with challenges and generally creates gaps between what lands in production and what comes out of the development groups. By utilizing new, more effective static and dynamic analysis security tools, security teams can often integrate their testing to coincide with other tasks, such as load and user testing. Gone should be the days of doing a code drop and hammering down on manual security testing while the developers await a report.

Cloud Security

Now, with SecDevOps attacking security challenges head on, there remains another challenge that has yet to be addressed. Cloud and DevOps go together like… well, like something that goes together really well. Traditionally, DevOps utilized virtualization, and despite its excellent provisioning capabilities and scalability, still introduced challenges. With shared hardware, maintenance windows, licensing costs and a multitude of other limitations, it's no wonder operations teams jumped on the chance to do point-and-click deployment of entire server farms. The ability to scale up and out fits perfectly within the continuous development model, but introduces some challenges for the security team.

First, many strict cloud implementations exist outside the "border" of the traditional enterprise network, where the vast majority of security controls and resources exist. Cloud operating systems, supporting software infrastructure and even cloud network traffic are all valid and real concerns for security teams.

Second, ongoing security operations such as patching, anti-virus, user provisioning, access control and a host of other challenges present themselves on an ongoing basis. These challenges are possibly some of the largest and most difficult to solve, partially because of the lack of control over the underlying cloud components.

And there are many other challenges, such as monitoring, security alerting, incident response and forensics. What happens if a cloud system gets compromised? Taking a step back, it's easy to see how more traditional security mechanisms might get left behind. Typical security configurations on high value systems usually include turning off remote management ports and disabling data-sharing protocols, which can be difficult in a cloud environment. And security logging and monitoring traffic might drive up bandwidth costs with your infrastructure-as-a-service (IaaS) provider, or may overwhelm channels normally required for normal operations.

FAST FACT

86%

of all websites have at least one critical security vulnerability.

— WhiteHat Security 2015 Website Security Statistics Report

Making DevOps and the Cloud Work for Security

Since day one, IT security has generally frowned upon Agile, DevOps and cloud solutions because they are notoriously difficult to secure. How can security keep up with server lockdowns, code reviews, dynamic assessments, all while trying to main a secure architecture? It's certainly not easy — but it can be done effectively.

Right off the bat, DevOps in a cloud environment offers some key advantages that we haven't traditionally had when dealing with enterprise applications. First, the DevOps team is already meeting on a regular basis and collaborating heavily. Security teams should get right in the middle of that, and have a say in the provisioning process for new application servers. Security can help the developers adopt secure coding practices, and architect new libraries that can be reused in future builds.

The same could be said for cloud-based solutions. Let's attack security by capitalizing on the benefits of rapid provisioning and high availability. Intra-cloud bandwidth is much less costly than routing monitoring traffic over the WAN, so does it make sense to stand up a low-cost instance of your monitoring solution next to the resources it protects? What about building out code and security analysis instances alongside development and QA environments? Security departments can turn these perceived negatives about continuous development into positives using a simple concept: security orchestration.

Security Orchestration

In a nutshell, security orchestration is a fancy term for security's effort to match what DevOps has done for IT operations in a continuous development environment. It's a drive to streamline and automate security tasks and technology. Security Orchestration in and of itself is a giant concept. It includes everything from SIEM to dynamic firewall configurations and all the way around to automated code analysis. For the purposes of our effort here, we will focus on the orchestration of cloud computing, Agile development, and security.

Cloud Security

Cloud security challenges come in all shapes and sizes, and are usually dependent heavily on a few variables.

What type of cloud implementation does your enterprise need: public, private or hybrid? While a private cloud does alleviate some risk, many of the DevOps issues still remain. Private cloud instances still must be kept up to date, libraries need to be upgraded, and monitoring needs to be present. One of the advantages to a private cloud and a hybrid implementation is that at least some of the infrastructure remains on the local enterprise network, making access easier.

All of this may seem like a case for keeping the cloud local, but that's not the case. In most cases, larger IaaS vendors have a well-stocked pool of IT Security talent, as well as big budget for network security resources like intrusion detection, security information and event management, and distributed denial of service protection. Additional services are often available for increased cost, but are almost always cheaper than procuring the same technology for your local enterprise.

Of course, public clouds aren't without some increased risk. Keep in close contact with the cloud security teams with regards to hypervisor and underlying virtualization console vulnerabilities. Some providers may be slow to update or patch due to the sheer volume of instances in production. The private clouds and internal sides of hybrid clouds

might be more straightforward in this regard.

Fortunately, since most IaaS providers have standardized operating system clones, any customization can be done initially by the security teams so that they become integrated automatically from the start. This is where the largest win for cloud and security comes into play. Historically, server build teams would install an operating system or push an image to a virtualized platform, run what security automation they could (such as GPOs and lockdown scripts), then hand off the servers to the security teams for further review and configuration. Today, there are many vendors that provide automation and APIs for this purpose that are implemented specifically on platforms like Azure and AWS.

One ever-increasing concern about securing cloud resources is the amount of resources that they consume. In days past, servers had ample hardware to handle the security overhead of AV, Syslog, HIDS, etc. Cloud systems operate on a razor thin resource margin for a number of reasons, not the least of which is cost savings. Security tasks should be offloaded or frontloaded where possible.

Offloading security in the cloud isn't always as simple as it sounds. In effect, we want to be able to move as much of the overhead of ongoing security tasks away from the production resource where we can. This involves tasks like configuration management, access control, update management, etc. In old-school server builds, much of this workload would live locally, but with the rise of dedicated cloud security management platforms, this doesn't have to be the case.

Frontloading security efforts isn't a new concept, but has drastically expanded over the years. Application security is a perfect example, as evidenced by the push from manual and dynamic testing to static code scanners and cloud-based code review products.

Application Security

Only a few years ago, IT security analysts would get some input into how the application is designed at the front-end, but wouldn't see the code until it was stood up in a test environment for security assessments. This means there were untold numbers of missed opportunities to guide the development and design as the product was being written. Looking at how SecDevOps works, you could argue that this is a huge positive, both for the business and the security teams. The key will be getting input from the security team on stories and epics, as early and frequently as possible. Secure coding practices from organizations such as OWASP should be provided, reviewed, and frequently checked.

You should also consider moving application security as closely to the developer as possible. As an example, injection vulnerabilities are arguably the most common and severe of all application vulnerabilities, and yet are notoriously easy to detect using automation. They are also easy to resolve, using standard security libraries for input fields. Wouldn't it make sense to make injection checking and library reference validation an automated process, initiated by the developers as part of their standard build process? Now, let's take that mentality and apply it across as much of the security spectrum as we can. Think about tasks that we are already automating that run parallel with security testing, such as automated user testing. Identify possible business logic vulnerabilities and roll them into existing automated testing platforms.

Other application security tasks, such as library version checking, SSL validation and authorization checks can be treated much the same way.

FAST FACT

20%

of security pros said vulnerabilities introduced by their app development team are a top concern.

— 2016 Black Hat Attendee Survey

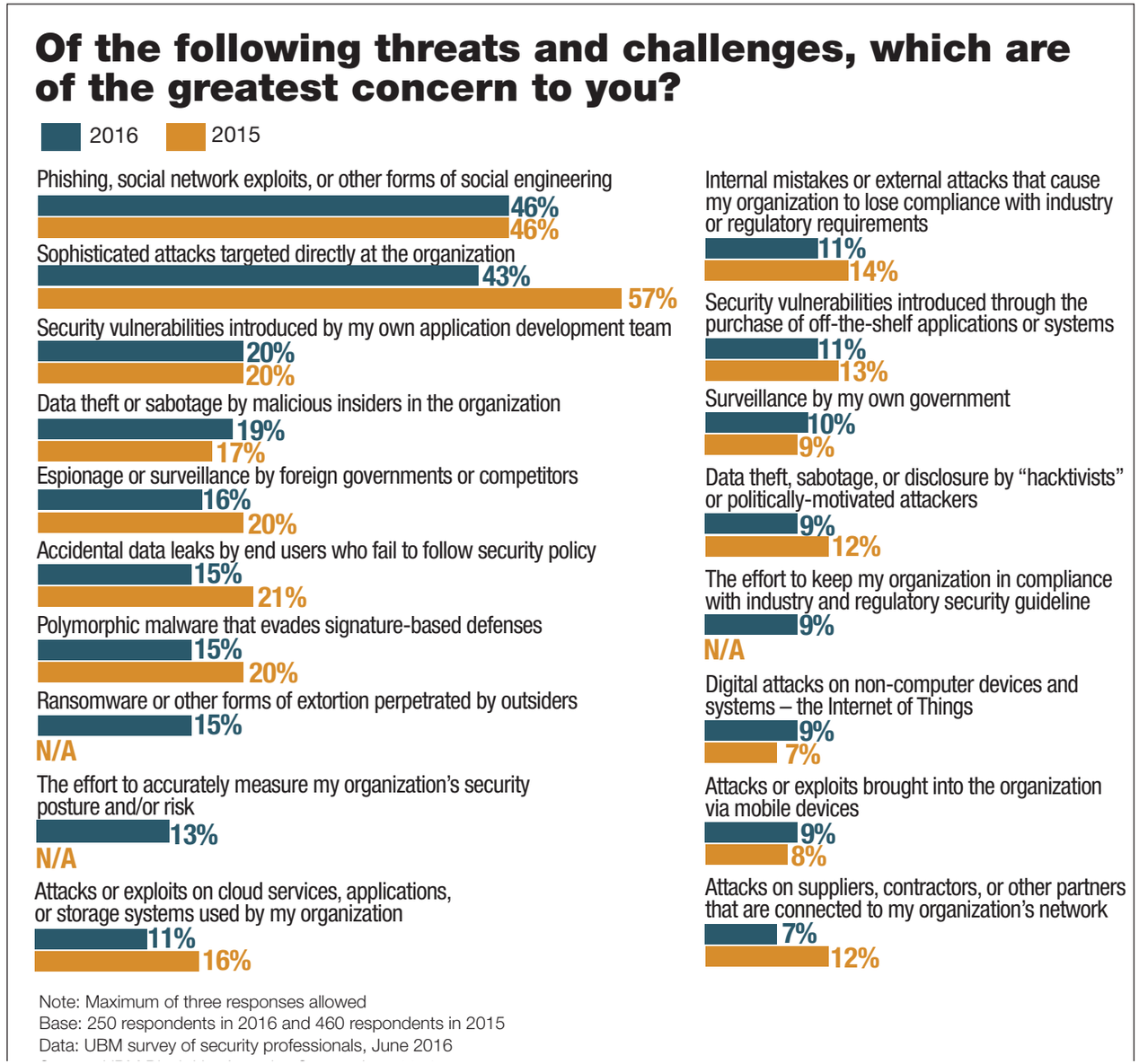
Development Security

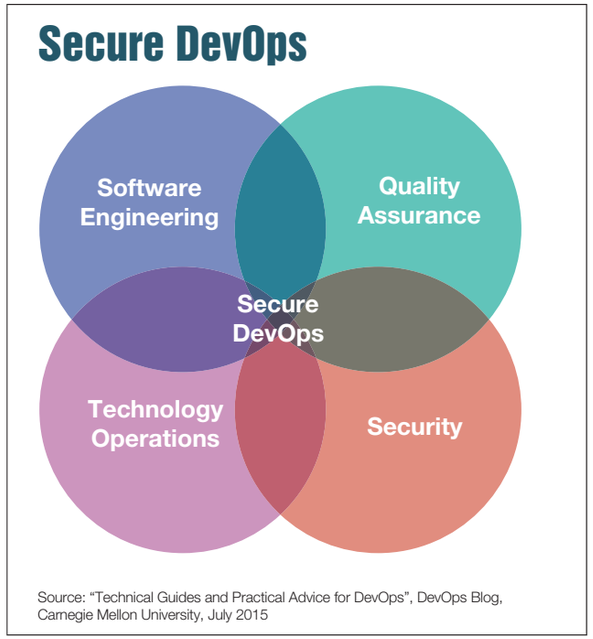
Development security is really just a process of migrating risks back in the hands of the developers. Most developers are perfectly happy implementing security mechanisms into their day-to-day tasks, especially if it saves them work in the long run.

There are two primary components to development security.

Code-specific risk introduced during development is often only discovered after the code has been released to beta or test. The problem is that code almost always has interdependencies, and the fix to a vulnerability might have far-reaching impacts to the overall application functionality or stability. By allowing the developers themselves to initiate automated code assessments as part of their normal bug checking process, you can alleviate some of that pain. Obviously, this shouldn't happen in a vacuum. By including the security team in the implementation of the automated processes that the developers use, they will be sure that the proper tests are run, and the tool and API configurations are setup correctly. In an ideal world, when code is delivered to security for review, it would be accompanied

Figure 1





By integrating with each discipline and capitalizing on their specialties, we can accomplish more with less, and with less impact than ever before.

by a report containing a clean static code assessment. This reduces a significant amount of work on both sides.

Code support component risk basically outlines the risk introduced by all of the moving parts of the deployment process, from the bug tracking software, change management and content management to the test environments. All of these are infrastructure that has an attack surface — and in most cases, doesn't get the attention from the security team that it deserves. According to a 2015 study by PTSecurity, 50% of all test environments contained critical security vulnerabilities. Further, the average number of high-severity vulnerabilities detected in test systems was 12.8 — almost twice as high as production systems.

By involving the security team in the provisioning of all components in the development support infrastructure, and allowing them to tie in automation points for security checks, you can alleviate a huge percentage of what would otherwise be significant security issues.

Getting Started on Linking Cloud and DevOps Security Efforts

Getting off the ground with so many moving parts can be a daunting task. There are some initial basic steps you can take to get the ball rolling.

1 Executive Buy-in

Initiatives of this scale and impact must come from the top down. First, the initial workload is going to be higher than the ongoing task-based work that typical application and operational security teams are used to. This effort may take them away from the day to day work for some time. Second, there will be costs — although usually much less than other large-scale security initiatives. Fortunately, though, many of the tools and resources available for application security are SaaS or open source, so much of that cost can be mitigated. This is probably the most critical point of all, because without it, any grassroots security efforts are going to struggle to get traction.

2 Automate, Automate, Automate

The nature of DevOps almost completely eliminates the ability of security teams to inject themselves into the software development life cycle in a traditional way. Fortunately, many products have automation and API integration capabilities that allow security features to be used without someone at the helm of the ship. IaaS companies such as Azure and AWS have APIs that can be used to automate secure provisioning, while application security analysis — both dynamic and static — can be automated locally or in the cloud. Be sure to think outside of the box, too. Integrate business logic testing alongside your existing QA testing, for example. Look at your automated cloud provisioning as well, and find opportunities to automate security tasks there.

3 If You Can't Beat 'Em, Join 'Em

The traditional battle lines of security versus the rest of IT has created more than a few headaches over time. In the spirit of DevOps, though, consider integrating one or more security staff into the daily scrum meetings. Having a security resource involved in the day to day tasks will allow the security team to have more visibility into the efforts that are planned or underway. Using this intelligence, you can poise your security efforts to head off future issues, or guide the operations and development groups so that these issues are avoided entirely.

4 Have an Open Mind

Consider changing the typical IT security approach from a "no, you can't do that", to a "let's find a way to do that securely." There will obviously be a give and take on both sides, but this approach of honey rather than vinegar has seen some successes. Most companies aren't in the "perfect code" business — they're in the moneymaking business. With this in mind, try to look at "destination security" with a "how can we get there" attitude.

5 Change the Way You Evaluate Security Tools

The typical approach of security tool selection has been along the lines of "I need it do perform xyz." While this has worked for security teams that typically operate in silos, integration and automation now need to take front stage. By involving the development and operations teams, and polling them for what they're using now (and what features they'd like to see), security teams can win them over and streamline future workloads.

Next Steps for the Cloud

Having taken these steps to improve secure application development, what additional steps might you take to ensure a smooth transition to the cloud? First, reach out to your cloud vendor and find out what other customers are using from a security deployment and security configuration perspective. They'll likely have a short list of third party vendors with whom they have integrated. Using this information, find out what those capabilities are, and bump them up against your risk matrix or overarching security concerns. You may find that vendors have already solved many of your concerns, and they're probably doing it in an automated fashion. Make sure whatever solution you choose can be implemented and guided by the security team, but used with minimal effort by the operations team.

Look for solutions that solve more than one problem. The cloud offers a lot of advantages for the security team. The scalability alone might allow security to spin up more resources for automated code analysis or configuration deployment across the cloud footprint. This goes back to the open-minded, out of the box approach.

Summary

By combining the concept of DevOps and cloud platforms, you can capitalize on the benefits of both and quite effectively offset the obvious security drawbacks. For security, this means increased automation and empowerment. The ultimate challenge will be getting buy-in from leadership to do the hard work on the front end so that things will run smoothly going forward. In doing this, you can seamlessly integrate security into your Agile and cloud implementations while maintaining a strong security posture. ■