ALERT LOGIC®

PRACTICAL SECURITY TIPS FOR DEVOPS

More organizations are embracing DevOps and automation to realize compelling business benefits, such as more frequent feature releases, increased application stability, and more productive resource utilization. However, many security and compliance monitoring solutions have not kept up. In fact, they often represent the largest single remaining barrier to continuous delivery.

By working with the DevOps team, you can ensure that the production environment is more predictable, auditable and more secure than before. The key is to integrate your security requirements into the DevOps pipeline; however, as part of that integration you will need to change the way you work. A normal approach of checklists, templates, and manual processes will not scale. With the speed of cloud deployments, you will need to automate security as part of the process. This will allow you to move as fast as the DevOps team needs you to.

PRACTICAL SECURITY TIPS FOR DEVOPS

For security professionals, it is key to understand that instead of validating the end solution, you need to validate the pipeline. If you are happy that the pipeline is building the solution in a way that meets the security goals, you can be confident that this will be repeated every time a developer needs to get new web applications into production.

• ARCHITECTURE AND DESIGN

During architecture and design the development teams will be attempting to rapidly iterate against the requirements whilst building out the cloud infrastructure. It is at this point that security teams need to get involved to understand the scope of what teams are looking to accomplish because different elements of the infrastructure need protection in different ways. Amazon S3 is very different to protecting EBS storage on an EC2 instance, the barriers between IaaS and PaaS are rapidly breaking down, and each has a different security paradigm. This is why it's important to learn and understand the shared security model so you know how to implement security controls at different levels of the cloud stack.

ACTION: Work with the architecture to understand the cloud components being used, and the security controls required for each.

• APPLICATION SECURITY TESTING

Automated unit tests running after check-in are a core part of DevOps. This is where security teams can add-in testing procedures to automate the validation of the web application (https://www.owasp.org/index.php/Appendix_A:_Testing_Tools). The reason why automated web application and testing is so key in DevOps is that the shorter the time between a developer checking in code and a test failing - the less time it will take for

the developer to fix the issue. The same holds true for security vulnerabilities, running testing at the end of the project can inject significant delays as developers struggle to identify the issues and fix the bug. Identifying the issue within minutes of a developer checking the code in, reduces the time taken to identify and fix the issue.

ACTION: Automate testing procedures and integrate into the development process.

• SECURE AND HARDEN THE OPERATING SYSTEM

If you are orchestrating the creation of compute resources or general infrastructure it's important to harden the OS at the beginning of the project. The risk of applying OS hardening at the end of the project is that the web application stops working. If it is applied at the beginning of the project it allows issues to be identified up front. If the hardening needs to be relaxed it can be identified early allowing security teams to work with the developer to potentially find another way of performing the function.

ACTION: Include the hardened OS as part of the base image. Use resources like CIS Benchmark.

• HARDEN YOUR CLOUD DEPLOYMENT (STANDARD AMIS, SECURITY GROUPS, IAM ROLES, MFA TOKENS)

Cloud services can deliver incredibly secure infrastructures 'if' done correctly. However it is also very quick and easy to open up significant security holes. You need to review how your company is using the cloud. This includes the segregation of roles - do developers have the rights to change the production environment. If so why? I am sure you do not let your server administrators walk around using Domain Admin accounts; so why should people have root access in the AWS Console. You need to review everything from the development environment through to production

ACTION: Review how teams are accessing the console and what permissions that they have. People should only have the permission they need to do their job, and if they have significant permissions they should be using two factor authentication.

• DEPLOYMENT OF SECURITY APPLICATIONS

Once you get to deploying applications to production you must keep up with multiple teams deploying multiple applications into production. In the same way to can use automation to ensure that security is as you require it during development, you can ensure that your security controls are deployed at the same time. You should be looking at deploying network detection for threats on the network, monitoring of HTTP for attacks as well as monitoring log files.

ACTION: Script the deployment of your security controls so that all environments have a baseline coverage.

• VULNERABILITY SCANNING OF OS AND APPLICATIONS

One of the most common attack vectors is for people to exploit the vulnerabilities in the OS or applications that are running on the servers. As part of a DevOps pipeline servers can be checked for vulnerabilities. This ensures that you know what state your servers are in at any point. You will need to implement a scalable method to correlate security events and identify security incidents needing attention while keeping false positives to a minimum.

ACTION: Run regular vulnerability scans against the environments and remediate any vulnerabilities.

• PHOENIX UPGRADES

Instead of deploying patches to production, you should be burning and redeploying servers as required. This not only increases your agility to roll out new versions, but also increases your ability to rapidly respond to security issues. You can deploy a new patched version across your entire cloud environment rapidly and safely; and with the phoenix upgrade strategy you also reduce the risk of technical debt and configuration drift.

ACTION: Work with the DevOps team to support them using Phoenix Upgrades and ensure this gives you the ability to patch security issues and roll them out.

• ON-GOING AND REAL TIME AUDIT OF ASSET ACCESS

Visibility post deployment is often down to the level of auditing that has been put in place. You should have standard auditing levels across different server roles and applications. Once all of these elements are in place, it will allow you to audit production to ensure that at any point in time you understand what state production is in, and if it has drifted from its defined security profile. The cloud is often referred to as a programmable datacenter. Developers can use this to create huge IT systems in very short timeframes - you can use this same power to audit these systems multiple times a day.

ACTION: Work with the development team to set logging levels and use a tool like Chef to ensure that your configuration and security profile does not drift.

The evolution of DevOps should integrate security across the development cycle – providing speed and agility to securing critical applications, assets and services in a more predictable, auditable and secure way.

ABOUT ALERT LOGIC®

Alert Logic is the industry's first SaaS-enabled managed detection and response (MDR) provider, delivering unrivaled security value. Our purpose-built technology and team of MDR security experts protect your organization and empower you to resolve whatever threats may come. Founded in 2002, we are headquartered in Houston, Texas.

ALERT LOGIC®