Azure Annual DevOps Report

Microsoft

sogeti
Part of Capgemini

Enterprise DevOps Report

Security 2020-21

Security

Enterprise DevOps and, with it, widescale cloud adoption, represent a seismic shift for information security teams.

Information security experts have access to capabilities they could only previously dream about. For example, we now have the ability to trace any piece of code from production deployment backward through the software development lifecycle, understand the validations that have taken place at each stage, all the way back to the initial code check-in. However, we also now live in a world where many teams are pushing multiple code deployments per day, each of which has the potential to introduce an exploitable vulnerability if security principles are not embedded across the entire DevOps lifecycle and organization.

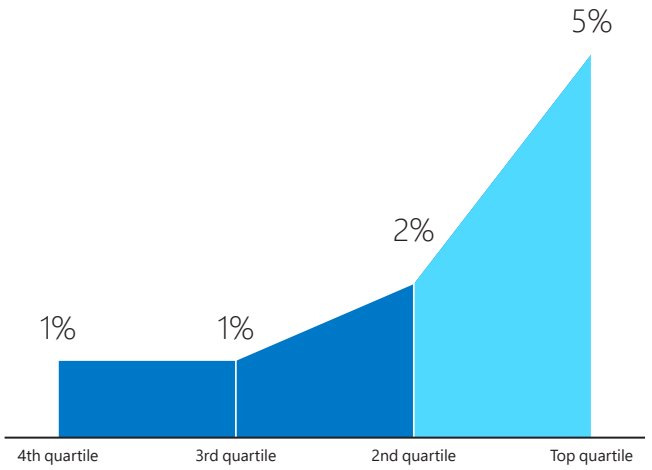
Of course, information security is a wider discipline than just the application code-focused example cited above. As the ISO 27002 (code of practice for information security controls) standard states; *"... information security management requires, at a minimum, participation by all employees in the organization"*. Security also has a real impact on the business—as the DVI research clearly showed that security and compliance was the second highest weighted driver (23%) on the difference between top quartile and second quartile enterprise performance. Security isn't just a cost, it's also a driver of performance.

DevOps helps organizations release software faster, but it can also help address the security needs when both developing software solutions and then running it in production. When security is embedded throughout an entire organization, it helps drive faster and more secure software delivery and assists in achieving consistent governance and control. Similarly to governance, when teams feel responsible for their products, and security is a communal goal, products are much more likely to be secure, and DevOps contributors will embed it into their systems and code. Security must be embedded in the full lifecycle of systems and teams, and top performing companies are turning to automation more frequently to achieve this consistency in a cost-effective way.

Unfortunately, there is no "one size fits all" approach to security. Rather, each application has its own unique requirements and constraints, so practices need to be tailored to its processes, security requirements, and operational procedures. What enterprises need are a set of principles and regulations that guide their security implementation across the complete software development lifecycle (SDLC).

What are the challenges?

REVENUE CAGR, 2014-2018 *FIG. 11*



Lack of a comprehensive approach to security

17% of organizations surveyed for the DVI report said they only test security vulnerabilities "for a major release" or "only when releasing to production", and only 46% of enterprises are incorporating security tooling into their DevOps pipelines. While 68% of respondents said that "security was everyone's responsibility" only around 40% were incorporating security requirements in the design phase, collaborating on threat models and prioritizing security requirements as part of their development backlogs.

Taken collectively, it's clear that organizations are not taking the holistic, systems thinking approach to security needed to implement a high-speed DevOps model securely. This is reinforced by the survey results regarding response time to resolve major security breaches. 6% of respondents said it would take "weeks", with 17% saying "between 3 days to a week", to resolve. Only 7% said they had the ability to resolve a major breach in less than 1 hour.

Securing open source

One of the other fascinating insights to emerge from the DVI research was the gap between the adoption of open source and the ability for companies to use their open source code securely. Despite 65% of organizations saying they "often or almost always leverage open source for product development", less than 20% are automatically scanning those components for vulnerabilities and remediating those vulnerabilities before deployment.

Deploying code with known vulnerabilities leaves organizations vulnerable to attack. Recent research in the State of Open Source Security Vulnerabilities report showed the number of OSS vulnerabilities jumped from 4,100 in 2018 to 6,100 in 2019, a 50% year-over-year increase.

WHICH OF THE FOLLOWING SECURITY PRACTICES CURRENTLY APPLY TO YOUR ORGANIZATION? *FIG. 10*

68%
Security is everyone's responsibility (e.g., there is a centralized security team; skilled security experton almost every team;
every developer is responsible for securing his/her code

47%
We have a centralized team to manage the security of open source code and package used by our organization

46%
Security tools are integrated in the development integration pipeline

44%
Security personnel review and approve code change before deployment

34%
We have a centralized team of security researchers

42%
Security and development teams collaborate on threat models

41%
Security requirements are treated as design constraints and product owners engage with security experts as early as the requirements definition and design stage

40%
Security requirements are prioritized as part of the product backlog

30%
Developers can provision a security of open source code and package used by our organization

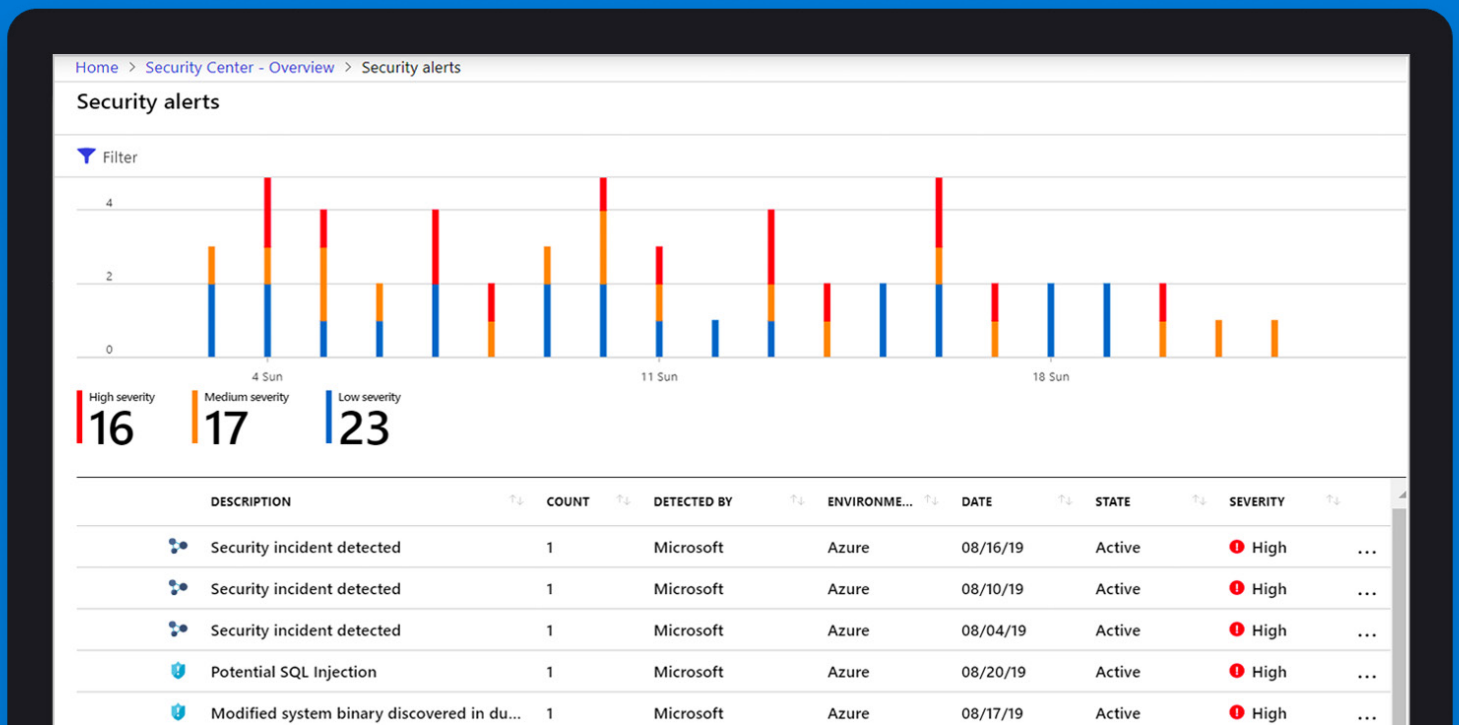
While open-source adoption offers huge benefits to enterprise organizations, it also raises major security challenges. Organizations need to understand where open source is being used within their applications and consistently adhere strictly to the open source security best practices across the software development lifecycle.

The cloud security challenge

Despite cloud vendors' best efforts, many enterprises seem to be struggling with securing their cloud environments. There have been numerous high profile and well-publicized cases of security breaches in cloud environments, through human error resulting in misconfigured systems, lack of key skills, negligence, or company insiders with malicious intent. Many companies are struggling with the transition from the traditional data center "perimeter (network) security" model to the multi-layered "defense-in-depth" approach recommended in cloud environments. Security professionals are having to learn new skills and new approaches in order to deliver effective cloud security, and for many organizations this transformation is lagging behind their cloud adoption, creating a security gap that leaves them vulnerable.

Lack of board-level oversight and investment

While an increase in federal and multi-national data regulations has led to a big uptick in enterprise data security initiatives, major hurdles have included the lack of organizational focus on security and inadequate funding for security initiatives. Nearly half of organizations have cybersecurity on their board agenda at least quarterly, according to Deloitte's Future of Cyber Survey[15]. However, only 4% of respondents say cybersecurity is on the board agenda once a month, despite increasing regulatory pressures that make the company directors personally liable for data security.



Addressing the challenges

Adopting DevOps, with its focus on automation and shared responsibility, can result in secure systems that are compliant with security guidelines and able to withstand the security threats that exist, if implemented properly.

Addressing the challenges of security as part of an Enterprise DevOps transformation requires a combination of key principles, standardized architectures (aka "platforms"), organizational design and automation.

Adopt the Zero Trust model

The fundamental mindset shift, particularly for enterprise organizations that have moved from on-premise hosting to a cloud-first model as part of their DevOps adoption, is the Zero Trust model. Microsoft defines Zero Trust as: "Instead of assuming everything behind the corporate firewall is safe, the Zero Trust model assumes breach and verifies each request as though it originates from an open network… Zero Trust teaches us to "never trust, always verify."

Guiding principles of Zero Trust are:

- **Verify explicitly.** Always authenticate and authorize based on all available data points, including user identity, location, device health, service or workload, data classification, and anomalies.
- **Use "least privileged access."** Limit user access with Just-In-Time and Just-Enough Access (JIT/JEA), risk-based adaptive polices, and data protection to protect both data and productivity.
- **Assume breach.** Minimize the blast radius for breaches and prevent lateral movement by segmenting access by network, user, devices, and application awareness. Verify that all sessions are encrypted end-to-end. Use analytics to get visibility, drive threat detection, and improve defenses.

To read more about how your enterprise can adopt a Zero Trust model, click here.

Secure development lifecycle and shift left

Ever since Bill Gate's now famous Trustworthy Computing memo in 2002, the role of software developers in security, and the importance of embedding security within the software development lifecycle[16], has become part of accepted industry best practice.

Enterprises must build security best practices into the way they design, write, build and test applications to dramatically reduce the number of exploitable vulnerabilities that make it into production. Setting clear security standards and building them into both the day-to-day development and operational practices, while embedding them into the platforms used by teams, is the first step. This will remove the over-reliance on centralized security teams that are so often overwhelmed with demands caused by the rapid rate of DevOps change.

Sogeti research shows that top performing enterprises introduce security checks at all phases of the software lifecycle—plan, develop, build, test, release, deliver, deploy, operate and monitor. Application security must speed up to keep pace with operations by bolting security checks and control inside the CI/CD pipeline itself. Testing and security should be "shifted to the left" through the automated unit, functional, integration, and security testing. This is a key differentiator since security and functional capabilities are tested and built simultaneously.

Secure reusable code components

Creating secure reusable code components, and making them widely used via your InnerSource strategy, is also key in solving Enterprise DevOps security.

Top performers are consistently using reusable components and leveraging what is known as the "DRY principle"[17] (Don't Repeat Yourself). DRY has the added benefit that you only need to fix a security vulnerability identified in your reusable code module once to fix that vulnerability for everyone, compared to potentially hundreds of times if the code has been reused via 'copy & paste'.

Obviously, open source software is a key source of reusable components for many enterprise organizations and improving the security of these components is an important industry goal. Top performers are installing automated security remediation technologies, like GitHub's Dependabot[18], which ensures security and visibility across the open source software supply chain. Dependabot pulls down your third-party open source code and checks for any outdated or insecure requirements. It then automatically flags what sections are out of date and allows you to make required updates to ensure that all code deployed is secure and compliant.

Build security into reference architectures and platforms

Platforms & executable reference architectures are a great place to start when dealing with Enterprise DevOps security. Central security teams help to define a set of secure reference architectures, ideally defined as-code and these are then shared using the InnerSource model discussed earlier. Solid examples of this practice can be found in the Azure Architecture center. These secure reference architectures, ideally along with secure release pipelines, provide a path of least resistance, so that doing things the *right* way securely, also becomes the *easiest* way.

Successful organizations are adopting DevOps reference architectures, a template documenting a system's full set of activities, steps, practices, and technologies that are used when creating an application or technology. With all the ideal steps, from build to monitoring, laid out clearly, enterprises can be confident that things are being done the "right" way, whilst addressing all of the organization's security goals at the same time.

When using these organizational reference architectures, teams often then create DevOps Blueprints, which extract a team's exact needs along each stage of the system's reference architecture. By figuring out exactly where along the software development lifecycle teams will need to use different technologies, expectations are clear. This ensures that DevOps teams know exactly what they must do to achieve business and security goals.

Observability, SIEM and SOAR

In a traditional cybersecurity world, Security, Information, and Event Management (SIEM) tools provide a unified security analytics platform to help the enterprise identify threats and malicious activity across the entire organization. Increasingly, top performing companies have started leveraging artificial intelligence (AI) to make sense of the terabytes of data that are collected by security monitoring tools. AI can help to provide deeper insight into what is happening within the Secure Development Lifecycle and to determine the best response to the security threats.

[16] Microsoft, SDL Practices, 2020
[17] Microsoft, Patterns and practices for Super DRY development for ASP.NET Core, 2019
[18] Dependabot

But, in order to aggregate this security and performance data, the applications and systems needs to be "observable". This means systems must be designed to expose important operational data via logging or other interfaces so that enterprises can "see inside" to understand how any given system is performing. The requirements for observability need to be part of the application design, and clear parameters must be set regarding what data is needed to understand the health of an application. This is vitally important because good observability enables us to baseline the behavior of an application under normal conditions. Anomaly detection can then be used to spot abnormal behavior that might be the result of malicious activity.

Having near real-time insight into the health of applications and infrastructures, and the security threats they face, enables DevOps automation techniques to be leveraged to automate some of the responses to these attacks. Some organizations are using the acronym SOAR (Security Orchestration, Automation and Response) to describe this process. Applications like Azure Sentinel can trigger an automated response when an alert threshold is met. These automated playbooks can deal with common attacks or issues without needing human intervention.

Make security a board-level responsibility

Security is too often seen as an IT problem, but building up a risk-aware culture should start at board level. Failing to invest in security, and not ensuring that critical security safeguards are in place, leaves companies open to liability. From our interviews, it is clear that enterprises must introduce security as a monthly checklist item at board meetings, and define a rubric of compliance that helps them adhere to the set of decided upon policies.

The National Cyber Security Center[19] offers a great list of questions that executive boards and CEOs should constantly be asking themselves. Most importantly, does your enterprise have a full and accurate understanding of:

- The impact on the company's reputation, share price or existence if sensitive internal or customer information held by the company were to be lost or stolen?
- The impact on the business if online services were disrupted for a short or sustained period?

Boards and executive-level leadership must always think large picture, and while largely "unseen" by customers, security concerns must always be a top priority to avoid costly security failures.

[19] National Cyber Security Centre, Introduction to Cyber Security- Board Level Responsibility, 2019

SECURITY CASE STUDY

Security is high on the agenda for Dutch electricity and gas provider Enexis. As a key player in the Netherlands' energy infrastructure, its public profile states: "Our 4,500 or so dedicated employees are working hard every day to ensure stable and reliable grids and to secure the future of our energy supply."

A scalable and agile IT-capacity in a public and cloud-native environment is key to this, and the company worked with Sogeti to bring about the transformations needed to succeed. In cooperation with partners, Sogeti created a roadmap and transitioned more than 100 existing applications to a cloud-native application platform, either on IaaS, PaaS or SaaS.

Security sat at the core of this process, with part of the success being the risk classification of data and systems that moved to the cloud. Within a reference architecture, Sogeti defined the risk levels for building capabilities in the cloud, with principles for both security and cloud.

With measures appropriate to acceptable levels of risk, the Enexis Platform DevOps teams delivered standard services (e.g. network, servers, etc.) with embedded security, and automatic patching. For both the Platform DevOps and the Apps DevOps teams using these standard services, the security policy based on this risk classification was translated into concrete requirements for the cloud platform. In this way, the teams could easily carry out a 'cloud readiness assessment review' before deploying to production.

This secure Enterprise DevOps model was further enabled by the simplification of applications implementation and integration within the IT operation using Continuous Integration and Continuous Deployment (CI/CD). The results were impressive. Application delivery went down from 6-12 weeks to less than 2 days, with the shift left approach enabling security issues to be identified and fixed early in the cycle to reduce overall lifecycle costs.

SECURITY NEXT STEP RESOURCES

Adopt the zero trust model
Learn more about the Zero Trust model
See how Microsoft assesses Zero Trust policies

Secure development lifecycle and Shift Left
Learn how to bake security into your development lifecycle

Secure reusable code components
Read more about the DRY principle
See how GitHub implements Security

Build Security into reference architectures and platforms
Learn more about the Azure Architecture center

Observability, SIEM, SOAR
See Microsoft's Azure Sentinel
Read the Intro to Observability
Automate security responses with playbooks