WHITE PAPER

DevSecOps Should Include Continuous Threat Modeling

By Jon Oltsik, Senior Principal Analyst and Fellow

January 2023

This Enterprise Strategy Group White Paper was commissioned by ThreatModeler and is distributed under license from TechTarget, Inc.

Contents

Executive Summary

Traditional waterfall software development may be "tried and true," but it also introduces cumbersome development cycles, often fraught with quality and security issues. Agile development can help accelerate development cycles with rapid "sprints" and feedback loops, but agile development remains misaligned with IT operations processes like production testing.

Many organizations have embraced DevOps models and continuous integration/continuous development (CI/CD) pipelines to address many historical limitations around software development. The result? DevOps can help accelerate and coordinate software development cycles, but there are still shortcomings. This white paper concludes:

- DevOps typically incorporates security only as fully automated defect discovery. Developers, IT staff, and security professionals report that DevOps teams often struggle to account for security requirements, visibility into security posture, or continuous security testing. When DevOps teams include these activities, they're often not consistently applied. This may be due to a perception that security activity will slow delivery and, therefore, can't be afforded. Alarmingly, this means that accelerated development cycles likely come with incrementally increasing levels of cyber-risk.

- Organizations are moving toward DevSecOps. Recognizing the security deficiency, many organizations are including security in their DevOps model. This new methodology, called DevSecOps, aims to marry a wider range of security touchpoints with the DevOps delivery culture and cadence. Successful DevSecOps includes:

    1. Keeping up with the pace of the CI/CD pipeline.

    2. Fostering cooperation between security, development, and IT operations teams.

    Cooperation typically entails security taking a "mentorship"-style role rather than that of an auditor, with software developers increasingly adopting more security practitioner responsibilities. While these are positive developments, some DevSecOps efforts tend to be myopically focused on tools rather than processes, limiting their effectiveness.

- Threat modeling can help add end-to-end security coverage while enhancing collaboration. Threat modeling is confined to a small percentage of the highest risk applications in some organizations. In others, it is minimized as a cursory white boarding or tabletop exercise at the start of a project. Either approach takes a point-in-time and high-level view of risk for a finite group of applications, while failing to dig into threats from an adversary or technical perspective. In other words, typical threat modeling fails to uncover how adversary tactics, techniques, and procedures (TTPs) are and how security controls would respond to real attacks. What's needed? Continuous threat modeling of all applications that starts during the planning stage of software's inception and is continually updated as part of the development lifecycle's phases and steps. In this way, threat modeling provides early insights to all threats (business, IT, and security) and a roadmap for integrating security into application structure, behavior, and data flows. Continuous threat modeling can help prevent the cadence-killing latency of a 'big bang' threat model, while keeping threat models in sync with the fluid reprioritization typical of sprint execution. Thus, threat modeling can add security to DevOps collaboration, improve the efficacy of those security activities, and even lower the costs of software maintenance.

DevOps Remains Fraught with Security Challenges

TechTarget's Enterprise Strategy Group (ESG) defines DevOps as a set of practices that coordinate activities across software development and operations teams. DevOps objectives include accelerating software development while improving software quality.

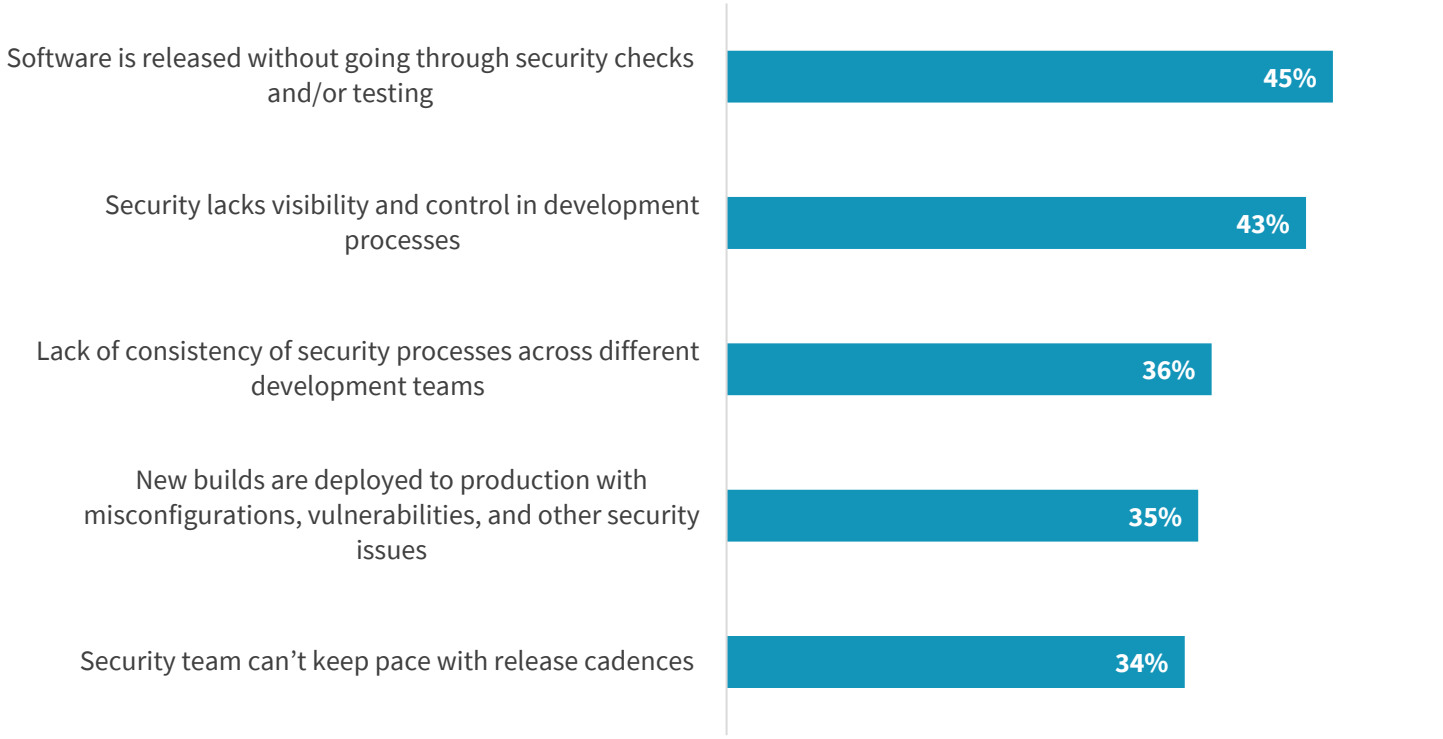
Driven by agile development, as well as platform-as-a-service (PaaS) and infrastructure-as-a-service (IaaS) innovation, DevOps is becoming increasingly popular. According to ESG research, 37% of organizations employ a DevOps model extensively, 28% employ a DevOps model on a limited basis, 7% plan on employing a DevOps model within the next 12 to 18 months, and 12% are interested in employing a DevOps model sometime in the future.[1] A typical DevOps process involves a variety of tools and technologies, including source code repositories, container image registries, build tools, automation and orchestration, project management, monitoring, and alerting.

Organizations are achieving benefits with DevOps but also admit to process and technology gaps between DevOps and security. According to ESG research, DevOps teams have a variety of concerns with faster release cycles of CI/CD as the pressure for rapid deployment means new builds get pushed with security issues. In a recent ESG research survey of 250 software development, operations, and security professionals, respondents pointed to several security challenges associated with their CI/CD pipelines including (see Figure 1):[2]

- 45% report software releases that didn't go through security checks or testing, leading to vulnerable code and an expensive repair cycle.
- 43% report that the security team lacks the right level of visibility and control in the development cycle. In this situation, security teams can't secure what they can't monitor and measure.
- 36% report a lack of consistency in security processes across different development teams. Different development teams likely have different skills, security models, and oversight. This can result in a wide range of security issues.
- 35% report that new builds are deployed to production with misconfigurations, vulnerabilities, and other security issues. Misconfiguration used to represent a tactical vulnerability or exposure. With infrastructure-as-code and Kubernetes configurations, however, misconfigurations may expose services, applications, infrastructure, or connections, as well as the controls employed for protection. This is an alarming situation that greatly increases the risk of a security incident.
- 34% report that security can't keep pace with the cadence of software releases. When organizations perceive a tradeoff between software release speed and security, security tends to lose.

[1] Source: Enterprise Strategy Group Complete Survey Results, *ESG/ISSA Cybersecurity Process and Technology Survey,* June 2022. All Enterprise Strategy Group research references and charts in this white paper have been taken from this complete survey results set unless otherwise noted.
[2] Source: Enterprise Strategy Group Research Report, *Walking the Line: GitOps and Shift Left Security,* November 2022.

Figure 1. Top 5 CI/CD Pipeline Security Challenges



Source: Enterprise Strategy Group, a division of TechTarget, Inc.

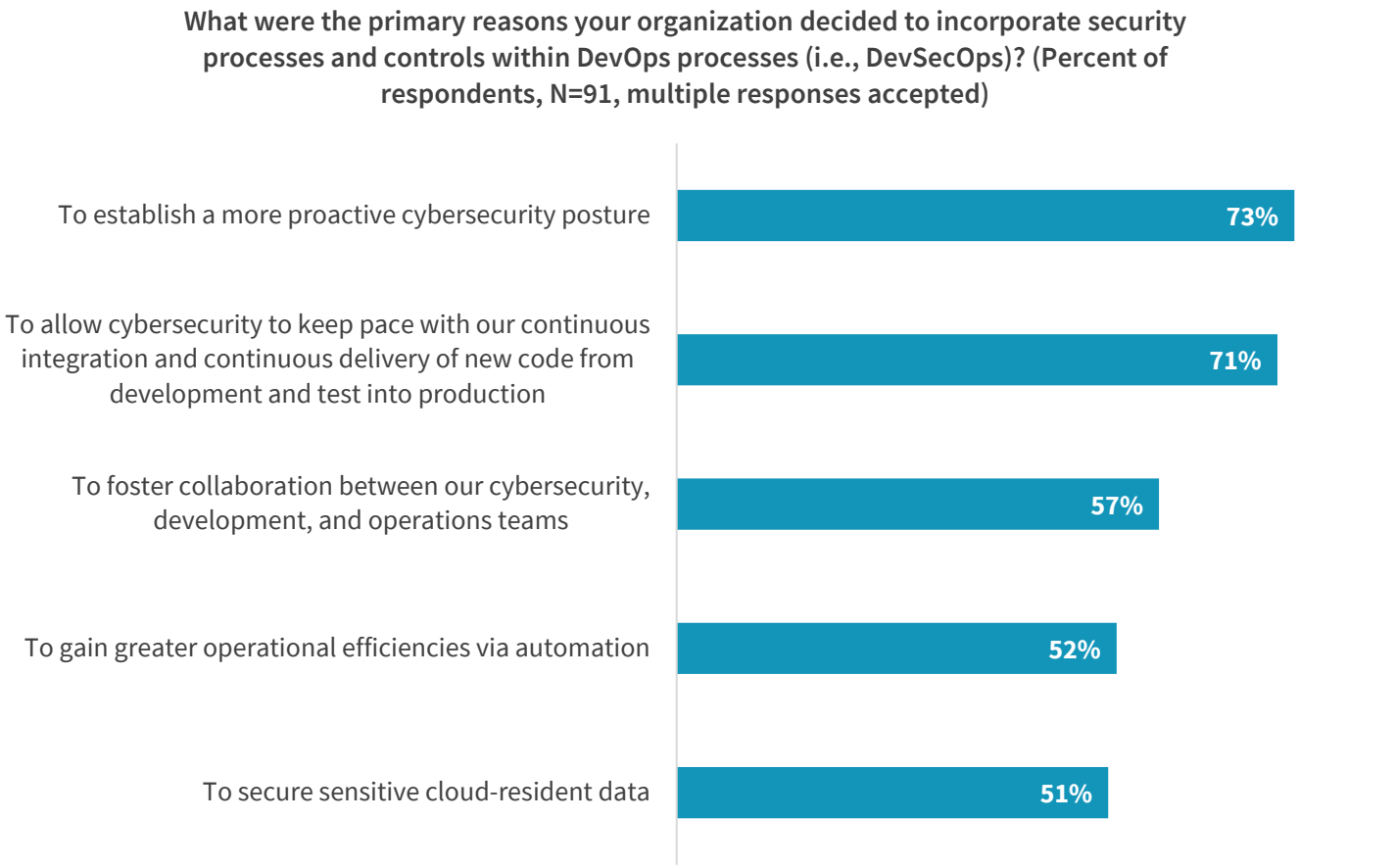
DevSecOps to the Rescue?

Recognizing the security shortcomings described above, many organizations have incorporated security into their DevOps processes, with tools for software composition analysis (SCA), static and dynamic application security testing (SAST and DAST), code reviews, penetration testing, security hygiene and posture management, logging, etc.

This intersection between DevOps and security creates a new set of processes, typically referred to as DevSecOps. Enterprise Strategy Group (ESG) defines DevSecOps as an emerging practice focused on automation, monitoring, and applying security to all phases of the software development life cycle (SDLC). It also expands the collaboration between development and operations teams to integrate security teams in the software delivery cycle.

DevSecOps is gaining popularity: ESG research indicates that 37% of organizations are incorporating security in the DevOps process extensively, while 33% are incorporating security into the DevOps process in a limited fashion. Development, operations, and security teams are embracing DevSecOps for multiple reasons, like keeping up with the CI/CD pipeline, fostering greater cooperations amongst different teams, gaining greater process efficiencies through automation, and securing cloud-resident data (see Figure 2).

Figure 2. Top 5 Primary Reasons for Adopting DevSecOps



Source: Enterprise Strategy Group, a division of TechTarget, Inc.

DevSecOps Should Start with Threat Modeling

DevSecOps momentum is a positive development, but efforts are still somewhat reactive. Since security teams are often late to the DevOps party, they join haphazardly by attempting to integrate tools (typically defect discovery tools) into different phases of the CI/CD cycle. This improves on the previous model by placing security tools in the hands of developers and executing them earlier in the development process, but it doesn't change the typical "build, test, fix" methodology. This 'atomized reactive security' doesn't go far enough to drive process efficiency, staff productivity, and security efficacy into software development. In short, it fails to produce a truly proactive and preventative result.

For this reason alone, DevSecOps benefits could be greatly improved if a foundation of threat modeling is included. The National Institute of Standards and Technology (NIST) defines threat modeling as "a form of risk assessment that models aspects of the attack and defense sides of a logical entity, such as a piece of data, an application, a host, a system, or an environment."[3] While there are many different threat methodologies, all models share a few common elements, including:

1. Diagramming the structure and behaviors. This involves mapping out the application structure, behavior, data flows, etc.

2. Discovering attack surfaces. From a DevSecOps perspective, an attack surface is a set of exposed applications (customer facing, back office, etc.), API, infrastructure, or even human interactions (i.e., social networks,

[3] Source: National Institute of Standards and Technology, *threat modeling.*

business partners, UI/UX, etc.) where an attacker can try to enter, cause an effect, or extract data. Threat modeling discovers and analyzes the attack surface, looking for potential vulnerabilities.

3. Imagining attack objectives. In this case, threat modeling helps an organization determine all the reasons why an attacker would want to exploit or compromise an application. For example, an application containing financial data or personally identifiable information (PII) might be an attractive cyber-crime target, while an application with lots of third-party connections could be a perfect launching point for disrupting a critical supply chain. Some methodologies refer to this is enumerating and valuating assets.

4. Enumerating adversaries. Beyond understanding what an adversary might want, threat modeling looks more granularly at specific adversary groups targeting particular industries and regions. It also digs into the tactics, techniques, and procedures (TTPs) these adversaries use most commonly as part of a cyber-kill chain.

5. Designing built-in security and compensating controls. Finally, threat modeling can help define threat prevention throughout the DevOps lifecycle, including directing software design and security control placement, mitigating risk with compensating controls such as a web application firewall, and tuning continuous security testing exercises.

Adding Threat Modeling to the CI/CD Pipeline

How can organizations bake these 5 aspects of threat modeling into their DevOps models? At a high level, threat modeling can be (see Table 1):

- Applied during project planning. Threat modeling helps enterprise security teams determine organization-wide threats and risk-rate their application and infrastructure portfolios. This allows organizations to apply risk-based governance to CI/CD pipelines from the start. Specifically, maturing DevSecOps practices change the security tools they use to CI/CD pipelines *and* change the configuration of those tools based on threat modeling output. Threat modeling poses questions like, "What do stakeholders value most in the application? Who will use (or abuse) the application? What sensitive or regulatory-scoped data does it include? Answering these questions can initiate a subsequent discussion about security architecture and compliance requirements, as well as usage policies and policy enforcement. Threat modeling in the planning stage is especially helpful for translating threats into a risk language that business managers can understand and act upon. By shifting as left as possible (while still protecting runtime applications to the right), threat models can expose risks before organizations write a single line of code.

- A guide to what can go wrong. Aside from business process risks, threat modeling digs into the application architecture, data flows, and dependencies. Considering a design's structure, behavior, and data flows, threat models also expose 'design flaws,' a category of risks that defect discovery tools cannot find. Armed with this data, organizations must align the most applicable threats so they can assess the potential business and technical impact if a given threat is exploited. Threat modeling can follow up on this knowledge by producing reports and checklists that can be used to implement and validate that security controls are in place as countermeasures to cyber-attacks.

- A prescription for building security into applications and infrastructure. Once cross-functional teams understand what they may be up against, DevSecOps teams can then analyze and prioritize risks and work with the development and operations team on a mutually acceptable plan for risk mitigation. Threat modeling at release inception allows DevSecOps teams to target security design spikes and design out flaws proactively and preventatively. For existing functionality and defect discovery data, threat modeling can help drive prioritization. The DevSecOps team can determine critical necessary fixes that need immediate attention and those that can be addressed in subsequent

releases. Of course, the goal here is to mitigate risk efficiently during the DevOps lifecycle, rather than react to problems once applications are in production. In this proactive way, threat modeling can improve security *and* lower costs.

- Used for continuous improvement. In a DevSecOps practice, threat modeling activities occur incrementally and continually, shadowing developer activities. User story updates trigger attack objective updates. Design spikes trigger diagramming, attack surface identification, and security control design. Coding triggers an update to structures and behaviors, as well as attack modeling, and so forth. DevSecOps teams can follow up on their threat models to continually check whether critical vulnerabilities have been addressed adequately. If so, they can close tickets in Jira or ServiceNow. If not, they can track progress with DevOps teams. Additionally, DevSecOps teams can strive for continuous process improvement by assessing the efficacy of their threat models and make necessary changes for the next iteration. Continuous threat modeling can also help teams assess and measure their security performance across code revisions. Most importantly, threat modeling activities conducted at release inception allow DevSecOps teams to target security design spikes and help design out flaws as a preventive measure before implementation begins.

Table 1. Threat Modeling Use Cases

Application of Threat Modeling	Reason	Benefit
Apply threat modeling during the planning stage of the DevOps lifecycle	Rate application risk, conduct risk-based security governance in the SDLC, align security with business objectives	Identify risks and provide a common risk register for all project participants, 'right-size' security activity to application risk tolerance
Define what can go wrong	Determine adversary objectives and how they might achieve them	DevSecOps team can work backward to identify the attack surface and attack paths for remediation prioritization
Build security into applications and infrastructure	Improve security while "shifting left," preventative security design, reasoned triage of defect discovery tool output	Improved efficiency and lower cost, architecture flaws alleviated via security control design
Continuously improve	Measure and improve security and process efficiency across a CI/SC pipeline	Security efficacy at the speed of agile development

Source: Enterprise Strategy Group, a division of TechTarget, Inc.

The Bigger Truth

The DevOps train has left the station but, based on Enterprise Strategy Group (ESG) research, security may not be onboard. DevSecOps models can help address this hole but not if the focus is security tools and technology.

ESG believes DevSecOps must feature a holistic end-to-end model, supporting all processes and phases of a CI/CD pipeline. This requires a commitment to threat modeling because:

- Threat modeling can act as a risk register for DevOps—a common set of models, reports, and metrics for evaluation by business managers, developers, IT operations, and security. As a result, threat modeling can act as a great equalizer and as a source of truth for all application-centric security information. This can help all parties improve collaboration and communications.

- Threat modeling can help organizations optimize security spending by focusing on defect discovery activities and prioritizing their output. This not only lowers development costs but also improves overall security efficacy and reduces the "alert fatigue" defect discovery has created.

- Threat modeling can continually identify design flaws that discrete defect discovery tools can't find. It also forces organizations into building security into software design. This helps align security with business processes.

It is also worth noting that, as of May 2021, threat modeling became a US federal agency requirement based on the Biden administration's issuing of executive order (EO) 14028, *Improving the Nation's Cybersecurity*. The executive order states that "the federal government must take action to rapidly improve the security and integrity of the software supply chain." In response to the executive order, the National Institute of Standards and Technology (NIST) published an interagency/internal report (NISTIR 8397), titled *Guidelines on Minimum Standards for Developer Verification of Software*. The report provides 11 recommendations for software verification techniques, including threat modeling ("to look for design-level security issues").

In summary, threat modeling could mean the difference between winning (i.e., a strong secure DevOps process) and losing (i.e., applications fraught with misconfigurations and security vulnerabilities). Threat modeling is a way out of the "hamster wheel of pain": build, detect defects, remediate. It focuses teams on what *could* go wrong proactively and helps formulate a secure design that prevents it. Based on this and all other benefits, it's safe to conclude that DevSecOps works best when it is anchored by strong threat modeling.

Enterprise Strategy Group is an integrated technology analysis, research, and strategy firm that provides market intelligence, actionable insight, and go-to-market content services to the global IT community.

www.esg-global.com


contact@esg-global.com


508.482.0188