The rise of secure DevOps in modern financial organizations

Kevin Rollison
Product Marketing, Fastly
The rise of secure DevOps in modern financial organizations

Today, consumers expect continuous innovation from every service or product they use. They want faster and simpler ways to save, invest, and interact with their financial institutions. Beyond bringing digital banking to every consumer, financial institutions are also tasked with developing new applications and solutions, such as automated investing platforms, advanced budgeting applications, secure and fast mobile payments, and cryptocurrency solutions.

Building new financial products and delivering innovative features in a rapid fashion requires a shift in an organization's approach to application development and delivery. The term that has defined this is DevOps — a set of practices and a shift in culture that aims to shorten the development life cycle while providing continuous delivery of applications. But despite the wide adoption of DevOps, financial services organizations may be missing a key element of this practice: security. In this paper, we'll discuss the path to secure DevOps and what to look for in a delivery partner when implementing secure DevOps.

The emergence of DevOps in financial services

Before we dive into secure DevOps, it's important to understand the rise of DevOps. In the past, the core focus of application development was functionality and availability. Apps were developed using a single-tenant approach with semi-annual or annual releases. These long release cycles meant financial companies couldn't react to changing business conditions, and customers weren't receiving the innovations they craved quickly enough. These apps were delivered using traditional CDNs, which lacked the visibility and control needed for continuous development. This new need — focusing on the acceleration of app delivery — led to the agile development processes and modern edge cloud platforms.

Agile development allowed organizations to shorten their development life cycles so they could deliver new features, fixes, and updates more frequently and keep up with growing business needs. This meant placing greater emphasis on automation, visibility, and control. Agile processes and continuous integration/continuous delivery (CI/CD), the process of completely automating code commit and test practices, led to the growing popularity of DevOps. The Phoenix Project defines DevOps as “a collaborative working relationship between Development and IT operations, resulting in the fast flow of planned work, while increasing the reliability and stability of the production environment.”

DevOps is a culture shift — a symbiotic relationship in which developers care more about how their code operates in production and operations teams purposefully familiarize themselves with how developers build their apps and services.

In addition to DevOps, developers have started using edge cloud platforms to locate logic and deliver functionality at the edge, improving application and web performance at a reduced cost. Edge cloud platforms give DevOps teams greater control and real-time decision making.

---

Why financial institutions need to go beyond DevOps

DevOps is a great start, but focusing on DevOps alone is not enough. In the past, security teams were pulled in at the end of development cycles. This approach was sufficient when development cycles lasted months. But, with DevOps making cycles much shorter — from months to weeks, days, or even hours — outdated security practices and traditional delivery approaches were getting in the way.

When security is an afterthought, it also leads to issues and vulnerabilities being identified later on in the development process, resulting in higher mitigation costs and risk exposure. For FinTech and FinServ organizations, the bar for security is set even higher. Customers provide their most sensitive data and trust that it will remain secure. In order to achieve this, security needs to be a shared responsibility between development, operations, and security teams — this is facilitated by baking security into the DevOps process earlier.

By combining DevOps and security, security gets included in the application delivery cycle from the start through the automated, early detection of known vulnerabilities. Some organizations may refer to this approach as DevSecOps. According to Gartner, “DevSecOps is the integration of security into emerging agile IT and DevOps development as seamlessly and as transparently as possible. Ideally, this is done without reducing the agility or speed of developers or requiring them to leave their development toolchain environment.” Secure DevOps is a natural progression from DevOps and CI/CD. Gartner estimates that “by 2021, DevSecOps practices will be embedded in 60% of rapid development teams, as opposed to 20% in 2019.”

Why is secure DevOps so critical? Because FinTech organizations are under pressure to innovate faster than their competitors and new market entrants — and do so while keeping their product secure. When implemented correctly, secure DevOps can provide the balance between moving quickly while meaning, or improving, security.
Four considerations when implementing secure DevOps

When considering how to implement secure DevOps, it’s important to think about the tools and processes that your developers will use when building and delivering their applications. A modern edge cloud platform (rather than a traditional CDN) can process, serve, and secure applications as close to users as possible, at the edge of the network. When evaluating edge cloud platforms to deliver your financial applications, we recommend looking for the following characteristics:

1. Robust control
   The tighter security controls can be aligned to an app, the more effective they will be. You should look for a solution that is fully configurable and API-driven, so your developers can create and adjust their own security controls for things such as WAF (web application firewall), ACLs (access control lists), and TLS (Transport Layer Security). This allows your developers to better align controls to the software and applications they’re building. Ideally, those controls can be adjusted in real-time based on key insights from traffic, and developers can push out changes globally in seconds. Adding automation takes this idea one step further. For example, a developer could automatically block certain traffic based on an alert. When that alert is triggered again, a closed-loop resolution is already established, negating the need to watch for that alert. This approach can significantly reduce the security “noise” and allows your developers to focus on other key initiatives, like supporting the latest cryptocurrency standards.

2. High performance
   With application development, there’s a constant tug-of-war between performance and security. Security is something that’s expected to slow app performance and potentially have a direct impact on revenue. Your security controls should be built into the platform layer, so you can scale them rapidly without introducing bottlenecks or impacting performance. Some providers may maintain separate networks for things such as TLS, WAF, PCI DSS, and traffic. If one or more of those components are treated as add-ons or afterthoughts, they could slow everything down. However, if the components (such as the WAF) are tightly integrated, it will preserve application performance. This allows developers to keep applications highly performant, while delivering secure experiences. If a new vulnerability is found, or you’d like to roll out a feature in time for tax season — it’s important to get fixes and updates to your users as quickly as possible.

3. Comprehensive visibility
   Visibility is critical in agile environments. An organization needs to quickly identify and address issues at every stage. For example, if you can stream logs from the edge in near real-time, you’ll gain valuable insights for rapid detection and mitigation. All logs should be provided in a standard format and integrate with multiple tools such as analytics tools, visualization tools, or security information and event management (SIEM) platforms. Streaming real-time logs of events and incidents can help provide closed-loop resolution, allowing your teams to automatically spot incidents and remediate them quickly. If access to logs is not delivered in real-time, you may be forced to play catch up and constantly deal with blind spots. With real-time visibility, your organization is better equipped to tackle the vulnerabilities and common attacks like cross-site scripting (XSS) and SQL injection that often affect financial applications.

   Take caution with approaches that provide interpreted intelligence. While this information can be beneficial, it is not the whole picture: you can miss out on the forensic level of information needed to dig deeper into a problem and understand what actually happened. By clearly seeing what gets logged, teams can lead better investigations to identify and prevent future issues. Ensure that any provider is delivering the type and number of fields that you need along with customizability of those fields.
4. Easy integration into existing toolchains and compliance frameworks

Any tool leveraged by your organization should fit into your current DevOps toolchains (as in, don’t force developers to perform security testing outside of their preferred toolchain environment). These toolchains can include common CI/CD toolchains (Travis CI, Jenkins), config management tools (Chef, Puppet, Terraform), and code repo systems (GitHub). All security controls should be able to be applied in container-based development and deployment environments, such as VMWare and Docker. Controls must fit into the compliance and corporate governance and policy frameworks designed for FinServ and FinTech, such as PCI DSS and SOC 2. For example, to ensure secure coding practices are in place, a WAF could support the PCI DSS requirement for DSS30 requirement 66, or protect against the OWASP Top Ten. Additionally, a WAF can be leveraged for virtual patching and as a secondary set of controls and cloud-based security enforcement.

Conclusion

Modern financial organizations have started to incorporate agile development into their engineering and development teams. However, solely implementing DevOps leaves out a crucial element of the development life cycle. Instead, organizations should move towards combining DevOps and security — an approach that maintains the ability to quickly deliver high-quality applications with security woven throughout. Integrating security into DevOps allows FinTech and other financial services organizations to gain the performance, visibility, and control needed to build secure applications that users love.

Adopting secure DevOps requires buy-in from all sides, along with transparency, trust, continuous collaboration, and optimization. In short, it calls for a culture shift. To learn more about preparing teams for secure DevOps, read our blog post, where we outline 5 tips for creating a secure DevOps culture.