eBook

4 Strategic Principles of Modern Web App and API Security

Four principles to maximize your modern Web App and API Security Strategy

fastly

Introduction

Driven by the shift to public cloud services, microservice-based and multi-cloud architectures, and API-centric applications, the way businesses build applications has drastically changed.

Today, applications are globally distributed. Developers and security practitioners work together to ship secure software using integrated workflows. And engineers expect to be able to make instant, global updates.

The problem for security teams: Most web app and API security tools were designed for that sometimes-in-the-cloud era, not the pervasive cloud-native business era we are in today. While the way in which we build apps has seen exponential innovation, security tools and processes have not kept pace.

Creating and securing applications is difficult when the tools were built for a different time. A time before developers and security practitioners worked together to ship secure software using integrated workflows. A time before applications were globally distributed and API-based. A time before engineers expected to be able to enter a command and instantly make a global update.

Attackers are developers, too. And attackers aren't bogged down by the limitations of legacy solutions. They are as nimble as ever, using modern tools and workflows to build advanced pipelines of attack. They use automation to quickly create variations of malware and exploits to bypass systems that only look for known threats. It's never been more clear that it's time for a change.

It is time to evaluate your approach to web application and API security, respecting the way modern applications are built and managed. Using traditional tools and processes will leave you perpetually behind the adversaries and their attacks. You can use these four principles/ imperatives to analyze your existing security program/efforts, ensuring you have what you need to get ahead of the attackers and take a proactive stance on your state of security.

Principle 1: Tools must fight intent, not specific threats

> Today, antivirus experts have classified over 1.2 billion malware variants(2).

Despite this number, security teams continue to focus on fighting specific threats. When an organization evaluates new tools, the most common question is, “Can this protect me against X?” These concerns often come straight from the boardroom, where executives and board members hear about news-making threats, such as Stuxnet and the SolarWinds hack, and immediately want to know if the business is vulnerable.

This style of evaluation leads practitioners to tools that look for signatures or “indicators of compromise” (IoCs) for a particular threat. Yet, indicators of compromise typically focus on details specific only to the current instance of the threat, like the IP addresses of known attackers and regular expressions that match a particular attacker-controlled URL uncovered during the investigation into the attack.

Signature-based tools do not work well when attackers have automated their infrastructure, because current adversaries typically build a development pipeline that varies the IP addresses, domains, and code signatures of their attack in a very short time — usually on the order of days, but often within hours.

One example of these efforts is attackers’ adoption of automated techniques, such as domain-name generation, to quickly vary the source of an attack or spread out malicious resources among many IP addresses to defeat blocklists.

Signature-based tools also do not work well when the indicators of compromise are not definitive. The tools can have trouble differentiating between legitimate and malicious traffic and often fail to keep up with the unyielding increase in threats. And how could they? Recent reports indicate over 200,000 new malware variants are created every day.4

The signature-based model was sufficient two decades ago, but not today. We’ve seen it fail with anti-virus vendors that couldn’t protect against compromises (5), legacy WAF providers that look only for SQL injections or cross-site scripting (6), and bot mitigation tools that look only at the user agent of requesting browsers. If your tools are focused on specific threats defined by indicators of compromise, the results can be predictable: you might be compromised and not know, or you may notice that you are compromised, but only after damage has been done.

The result is that your security will be porous: bad attackers may get blocked right away, but pretty good attackers are likely to get through. And if the attacker is really good, you may never know.

Principle 1: In Action

GitGuardian Case Study

With over 3 billion commits scanned for GitHub since 2017, GitGuardian’s code security platform needed a WAF solution that didn’t require 100s of rule exceptions to prevent legitimate traffic - like code snippets in in-app messages - from being blocked. SmartParse contextually filters allowed code from attack payloads, allowing GitGuardian’s small team to remain focused on their tasks with robust security visibility and monitoring.

Intelligent tools signal on intent

The new rules of web application and API security demand a shift toward a more intelligent model, one that infuses enough confidence into the security toolchain that a practitioner can run the system in front of valuable traffic without fear that it will block legitimate attempts or allow malicious ones through.

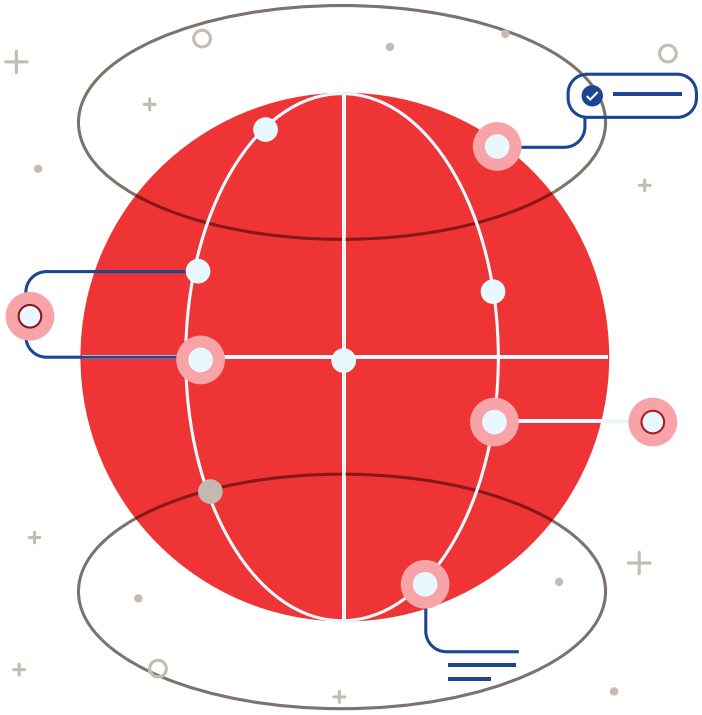
Getting there requires resetting the bar on security technology. First, practitioners must demand tools that examine not just the signature of the traffic but also its intent or behavior. Signatures can be a good first line of defense, culling the low-hanging fruit from consideration. More modern security tools can then take into account factors like the speed of the request, time of day, and user log-in status. Today's security tools must have knowledge of the user, device, and application, and know what constitutes normal behavior.

The depth of the analysis is important. Assuming a bot sends automated requests through software, such as CURL, and conversely, that a human uses a browser may be true in general, but it rarely is that easy. An attacker using an automated web application testing tool can make a request look like any browser, so security tools need to be able to know what an actual user's behavior will be.

In addition, security that does not focus on intent will mean that legitimate users who do something unexpected may get blocked. Remitly, a company that helps people all over the world transfer money back home, encountered occasional spikes in traffic on the Pacific coastline. Only with visibility into the traffic could they determine intent — the traffic represented legitimate transfers occurring from ships coming into port and customers using the service.[7]

Confident enough to block

Detecting threats based on behavior will give your security team confidence that the system can discern



actual threats from new yet unseen user behaviors. This means that organizations can have enough confidence to put the tool in blocking mode, not just monitoring mode.

Tools that can only run in monitoring mode for fear of false positives reinforce a broken system. Not only does it represent a lack of faith — perhaps, justifiably so — in the capabilities of the tools, but the damage is done by the time the team can respond. Imagine a superhero that stands on a street corner yelling out crimes — "Jaywalking!" or "Burglary!" — and then waits for a law enforcement response rather than rolling up her sleeves and taking action in the moment.

Not only does a monitor-only approach fail to take action, but it also contributes to another problem: Security and operations teams are drowning in alerts. Teams need a foundation of tooling that can confidently block threats as they happen, and remove them from the daily deluge of security events that they have to triage.

Tooling needs to keep up with modern threats without placing a burden on the security and operations teams.

Investigating every threat is a near impossible feat. If you're blocking the insignificant, your team can spend time focusing on the most significant threats.

Cloud-native companies need security that's also in the cloud. With modern cloud and SaaS solutions, you get the full weight of a security team staying ahead of threats and proactively delivering updates. The cloud model is not just about improving scalability and agility, but about consolidating knowledge — a security service in the cloud delivering the collected knowledge of the vendor to your security team.

> So, when evaluating a new web application and API security tool, demand that it assesses intent — and validate that it does. Speak with peers who have experience using the tool and test it for yourself.

Unless you trust a tool's ability to accurately differentiate malicious behavior, you will not trust its ability to automatically take immediate action.

Principle 2: There is no security without usability

The rise of intuitive, consumer-like web experiences in SaaS-based tooling has made usability near table-stakes for most technology tooling.

And yet, security solutions lag woefully behind. Security teams are least effective when they have no ready way to understand the current state of their organization's security. Teams that sift through log files, look for signals in false-positive-prone lists of alerts, or need a specialist to understand the output of a security tool are suffering because of poor usability.

The user interface is the first line of defense for an operator. Unfortunately, it's also the first place usability is neglected. Legacy systems often force operators to log into multiple user interfaces to manage the system, even when using solutions from the same provider. Older UIs can be slow and clunky. Sometimes, they force security teams to work directly with log files.

Legacy tools were designed to enforce and control — not to actually be operated. But modern teams expect a relationship with their tools. They need the ability to integrate, observe, and take action.

A poor UI creates a multitude of risks: gaps in policy and enforcement across tooling, slow and uncoordinated response to urgent threats, and inconsistent — or worse, absent — visibility into the holistic security ecosystem. A good user interface should deliver meaningful signals and surface trends and insights that invite users to dive deeper and quickly discover targeted information. A dashboard that surfaces high-level metrics and charts — so spikes can be quickly identified, zoomed in, and eventually lead the user to the specific threat in the logs — can inform decision making and create a learning feedback loop that can increase engagement and confidence in the business's state of security.

A single window into your security

Modern security solutions should have a single, intuitive, easy-to-use interface that allows control and visibility of the entire solution. Observability should be all-encompassing and integrated to provide full visibility into the state of the system at a glance.

Even if your system simply forces you to switch between tabs to navigate your solutions, you're still losing valuable seconds and integrated visibility while you're under attack. With DevOps and agile processes, developers and engineers can deploy 100 times a day, and security needs to keep up, or fall behind and fail to keep watch on what is going on from a security standpoint. If your security engineers are digging through logs to find signs of a threat, then your detection will be slow and response hindered.

> If you don't trust the tool that you are using, then you will probably ignore the information it provides.

Without a single view into security, your overall security posture will be weakened because of technical and visibility gaps.

Having a single, unified tool that can be used by a number of different audiences gives you the flexibility to create the dashboards that each of those customers need to view the state of security relevant to them. Both security and non-security teams need integrated security dashboards, because more viewpoints mean more people across ops and engineering can be empowered in the fight against threats.

The key is to unlock insights. Security teams should feel that the tool that they are using has their back. The tool needs to be able to provide easy-to-use functionality and visibility. If you don't trust the tool that you are using, then you will probably ignore the information it provides.

Integration is necessary for useable security

Modern tools must match modern application design. Too often, toolsets are simply packaged and sold together by a provider, but not actually integrated on a technical and data-analysis level. This "integration by invoice" is a sales tactic, not a well-thought-out solution for security.

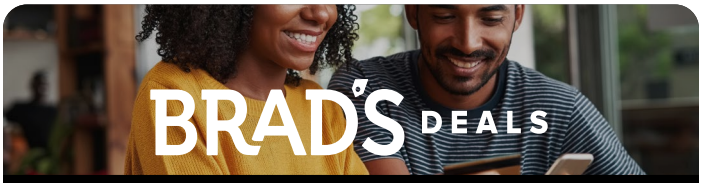
Providers should offer automation and integration by default, which starts with full API control. All security solutions should have easy-to-use APIs that expose all of the functionality of the system. Solutions should easily integrate not just with each other, but with the entire response toolchain, including tools like Jira, PagerDuty, Slack, and Splunk. And they should offer real-time logs and stats that expose the data to the company's preferred security monitoring or observability system.

The security tool should serve your needs. You should not have to serve poorly integrated software by spending inordinate effort integrating it with your security workflow.

Good integration makes for better security. Integrating all of your solutions makes it far easier to determine the true intent of the attacker. Static code analysis, for example, needs to deliver understandable results to the developers and directly into the development pipeline.
Web application firewalls need to alert, not only the security team, but also operations.

Engagement is a critical way to measure usability. When you have a usable security tool, the security team is able to engage business executives, developers, and analysts alike. By engaging more parties, more pervasive security knowledge embeds deeper and wider within your company. If the tool is not usable and not delivering security value, then the level of engagement with other teams will be small.

Security is a team sport; the more people involved, the better.



Principle 2: In Action

Brad's Deals Case Study
Using the Next-Gen WAF has been a game changer for online retailer Brad's Deals. Keith Mazanec, Director of Software Engineering, says "Rather than having our analytics team come to me a week later and ask 'what's this spike in clicks?' I can see right in Slack when a bot is calling the site, and most of the time, it just gets blocked automatically.

Principle 3: Real-time attacks require real-time reactions

Attackers are developers. And just like any other developers, attackers are pushing to speed up their development. Adversaries do not just use their own tools, but frequently use cybercrime-as-a-service capabilities to quickly adopt sophisticated techniques, whether using compromised Internet of Things (IoT) devices to target applications with distributed denial-of-service (DDoS) attacks[8] or using botnets to conduct password spraying against web applications.

Developers of these services have plenty of material with which to build attack tools. In 2022, more than 25,000 vulnerabilities were publicly disclosed, according to the National Vulnerability Database and CVE.org[9], maintained by the U.S. National Institute of Standards and Technology. Exposed web services and internet-facing applications are frequently targeted by credential stuffing attacks and exploit attempts. The number of daily attacks on Remote Desktop Protocol (RDP), for example, tripled over six months in 2020.[10]

Little wonder, then, why threats are outpacing legacy security models. Agile attackers are employing advanced DevOps workflows to quickly attempt, adjust, and deploy new methods — sometimes hundreds of times during a single attack. The Kashmir Black botnet, for example, used DevOps techniques to create a modular and easily-updatable software platform for infection[11], while other malware developers use a bespoke service to detect vulnerabilities in the attack tools.[12]

> Reaction time isn't limited by how quickly your brain works. It's dependent on the speed of your security solutions, which can be broken down into speed of visibility and speed of control.

Unless defenders can react at a similar speed, they have no effective way to detect, block, or respond to such agility. To be clear, the reaction time isn't limited by how quickly your brain works. It's dependent on the speed of your security solutions, which can be broken down into speed of visibility and speed of control.

Speed of visibility

If it takes minutes or hours to spot an attack, it's already too late. Attackers attempt one way to compromise your network and, if it fails, they will quickly pivot to a different approach. In 2022, the average time it took for an attacker to "breakout" — pivot from an initial attack to lateral movement across your network — dropped to eighty-four minutes, down 14% from 2021.[13]

The initial breach, however, happens even faster: often, in minutes. If your security pipeline consists of creating an entry for each security event, inserting that event into a long queue of alerts, and waiting for an analyst to have time to review the information, you will never catch the attackers in time. By the time your security solution is able to relay the data to an analyst, a security information and event management (SIEM) tool, or a monitoring system, the damage has been done.

Real-time visibility — measured in seconds — serves both the automated and manual workflows. It allows the system to apply logic in real time for threat examination, and it provides security operators the ability to react to alerts that require human intervention. Security solutions that push intelligence decisioning to the endpoints and network edge improves your ability to react quickly. Distributing your security telemetry everywhere allows the edge devices to filter and process raw data, and the main system to tackle harder correlations, detections, and responses.

Speed of control

Once you or your system can see the threat, speed of response or remediation is critical. Visibility must go hand-in-hand with control.

The most intelligent, intent-based approach to mitigation requires multiple streams of data to make a decision. Intent-based systems need to be able to analyze data — and make decisions based on that analysis — in real time. The systems should also be constantly analyzing patterns and behaviors to predict new or evolving threats, so it's imperative they can both see and interpret traffic in real time, and have the power to deploy new rules in response to changing threats.

Speed of visibility and control cannot be constrained to a single deployment or geography. The prevalence of distributed systems — either multi-region and multi-cloud deployments, or a distributed workforce that needs to be protected — requires the ability to take action across physical locations or boundaries. Some security solutions are fast only when protecting a single location. Such systems are not adapting to the modern cloud-native world — software and people are deployed across the globe and security must keep up.

Attackers often operate from a different region than where your data or applications reside, because investigating attacks across jurisdictions is more difficult and gains them some measure of immunity. A security system operating in real time also brings another benefit: blocking attacks becomes a fast and effective way to stop an attacker from compromising your systems. For companies that have mastered Rule #1 (recognizing intent) and Rule #3 (real-time speed), accurate blocking in real time becomes not only a possibility, but a preference.

Principle 3: In Action

Le Monde. Case Study

The ease of implementation and pre-packaged rules with Fastly's Next-Gen WAF proved critical during a time-sensitive situation. "We were under attack when we put it into production," said Paul Paul Laleu, CTO of Le Monde, "so we had to find a response very fast. Fastly helped us do that." The Next-Gen WAF was deployed in production in a matter of days, and the default rules proved highly effective without the weeks or months-long tuning period of legacy WAFs.

Principle 4: Dev, sec, or ops — everyone must think like an engineer

The trio of application teams — developers, security, and operations — have historically been siloed, working independently and inefficiently to ship software. The lack of alignment between teams results in conflicting priorities, incompatible toolsets, redundant workflows, and stunted collaboration. The result? The ops team doesn't communicate necessary best practices to developers, the developers fail to provide deployment-ready applications to operations, and the security team blocks deployments that aren't compliant with security policies.

These silos put businesses at a disadvantage. Attackers often use the work of other malware developers and adopt services that encapsulate sophisticated attack techniques.

To have a chance against these adversaries, security, engineering, and operations must work together through the secure DevOps model. When developers talk with security people — or have security champions within their group — then they can get feedback in real time on design decisions. Developers and operations together will incorporate more efficient deployment processes and faster patch pipelines into the development lifecycle.

When developers, security, and operations work together, it can be a corporate superpower.

Yet, just adopting a commitment to work together is not enough. While many security and engineering leaders believe bridging the technical and cultural divide between teams will result in faster, more secure application development cycles, antiquated practices and tooling still hold teams back.

Often, initiatives to pursue secure DevOps become more performance art than authentic integration. Bolting security operators and their preferred tooling onto the end of your deployment pipeline does not mean you're doing secure DevOps — and it won't make your software ship faster.

True secure DevOps builds security verification and vulnerability scanning directly into the automated testing and deployment framework. It provides a path for security teams to show up as an integrated part of the development team — not a gate brought in at the last minute to submit a list of vulnerabilities and hope they get fixed before the system goes live.

The engineering mindset

In effect, good secure DevOps practices require that everyone think like an engineer. Thinking like an engineer means that developers need to write code using secure development practices and automated CI/CD pipelines that test — not just for functionality, but for common security holes. Thinking like an engineer means that operations can quickly roll back changes to the last known good version, and the security team has the skills and authority to employ in-depth security audits in case the automated system misses anything.

In order for secure DevOps to live up to its promise, security practitioners, operations professionals, and developers must all adopt an engineering mindset with a focus on shipping secure software. Eliminate the toil of managing security products and capabilities so that security professionals can shift from operators to engineers. It not only makes the overall system more reliable and secure, it gives employees a more fulfilling career path.

The Paved Road approach

When the development pipeline has been well-engineered, then each member of engineering teams practicing secure DevOps knows their role and knows what services and capabilities are available to them. This "Paved Road" concept, made popular by the DevOps groups at Netflix, makes each part of the pipeline a service that can be offered to developers, operations, or security. Software components need to be reusable.

Development services need to be available to everyone who may need them, just as security services should be part of the entire pipeline.

For instance, say a security team needs to solve a vulnerability management problem, but to do that they need to improve asset discovery. By widening the project's scope and talking to the operations and engineering teams, the security team may find that they, too, have an interest in that service, and can collaborate to solve a universal issue. The lesson: If security has a challenge, there are likely engineering teams that also have that same challenge.

A holistic approach to secure DevOps is not reflected in your organizational chart, but in your culture — knowing that you — dev, sec, or ops — are all on the same team. Everyone should consider how security-focused changes may slow development and hurt the business. Everyone should adopt adversarial thinking and threat modeling. And everyone should worry about the efficiency of applications and the security in front of those applications.

The ultimate measure of whether your developers, security team, and operations team have adopted a secure DevOps mindset is not whether each group focuses on their specific responsibilities, but whether each group understands the pipeline and their role in moving secure, functional software to deployment.

Better security is integral to building better software

Over fifteen years ago, Amazon launched Amazon Web Services and kickstarted the migration of the business world to the cloud. In that time span, businesses have seen and adopted hundreds of new frameworks, languages, services, and tools to build faster and more user-centric applications. But the friction between shipping software quickly and shipping software securely remains a sticking point.

The path to reducing that friction must include security solutions that meet the needs of modern teams — ones that include security as an integral part of the cultural and technical aspects of building software. **It's not enough to ship software quickly. We must ship high-quality software securely.** To do that, we need to not only embed security process controls into the software development lifecycle (SDLC), but also better integrate security into company culture. The more that security is ingrained in your culture, the faster you will respond to threats.

A secure DevOps practice only becomes worthy of the name when everyone knows they are on the same team and the culture not only embraces fast development, but secure software as well. To do that, security must be viewed as a partner, rather than a checkbox — one that lives up to these new rules.

Principle 4: In Action

Stuff Case Study

The breadth of Fastly's solutions allowed Stuff to completely replatform, using Fastly not only for content delivery but also for security and DevOps. "Replatforming with Fastly provides the basis for how we innovate, build new products and improve user experience... Stuff turned to Fastly to accelerate development, maintain a strong security posture, and ensure fast, high-quality delivery to every user on a tightly integrated platform.

Conclusion

Better security is integral to building better software

Over fifteen years ago, Amazon launched Amazon Web Services and kickstarted the migration of the business world to the cloud. In that time span, businesses have seen and adopted hundreds of new frameworks, languages, services, and tools to build faster and more user-centric applications. But the friction between shipping software quickly and shipping software securely remains a sticking point. The path to reducing that friction must include security solutions that meet the needs of modern teams — ones that include security as an integral part of the cultural and technical aspects of building software. It's not enough to ship software quickly. We must ship high-quality software securely. To do that, we need to not only embed security process controls into the software development lifecycle (SDLC), but also better integrate security into company culture. The more that security is ingrained in your culture, the faster you will respond to threats. A secure DevOps practice only becomes worthy of the name when everyone knows they are on the same team and the culture not only embraces fast development, but secure software as well. To do that, security must be viewed as a partner, rather than a checkbox — one that lives up to these new rules.

Endnotes

1 "Symantec Internet Security Threat Report, Trends for July 1, 2003 - December 31, 2003." Symantec Corp. PDF Report. p. 27. Published March 2004.

2 "Malware: Total Malware." AV-TEST. Chart. Updated: 05 Sep 2023.

3 "Wordfence Malware Signatures in Production." Wordfence blog. Chart. 21 Feb 2017

4 "Malware: New Malware." AV-TEST. Chart. Updated: 05 Sep 2023.

5 "Low VirusTotal detection rates for new malware, do they matter?" Virus Bulletin. Web blog. 3 Feb 2015.

6 Hurder, Liz. "Four Ways Legacy WAFs Fail." Fastly. Web blog. 17 March 2021.

7 "Case Studies: Remitly." Fastly. Web site. Retrieved 5 Sep 2023.

8 Chickowski, Ericka. "The Rising Tide of Crimeware-as-a-Service." DarkReading. Web. 13 Jun 2017.

9 CVE.org Published CVE Records. Database. Web page. Accessed: 5 Sep 2023.

10 Kubovič, Ondrej. "Remote access at risk: Pandemic pulls more cyber-crooks into the brute-forcing game." WeLiveSecurity. Web blog. 29 Jun 2020.

11 Muncaster, Phil. "KashmirBlack Botnet Uses DevOps to Stay Agile." Infosecurity Magazine. Web. 20 Oct 2020.

12 Krebs, Brian. "This Service Helps Malware Authors Fix Flaws in their Code." KrebsOnSecurity. Web. 18 May 2020.

13 Muncaster, Phil. "Attacker Breakout Time Drops to Just 84 Minutes" InfoSecurity Magazine. Web. 1 Mar 2023.