5 Best Practices for Securing Modern Web Applications and APIs

paloalto NETWORKS | PRISMA BY PALO ALTO NETWORKS

SITC SECUREIT CONSULT

Table of Contents

Introduction

Web applications are nothing new. Neither is web application security. Many businesses have been building and securing web-based applications for more than a decade.

Yet, over the past several years, the nature of web applications has changed fundamentally. Monolithic applications hosted by individual web servers have been replaced by containerized, cloud native applications that are distributed across a cluster of host servers. According to O'Reilly, more than three-quarters of businesses have now pivoted to microservices as the go-to means of designing applications. Microservice architectures not only introduce additional security complexities but also, since they require orchestrators like Kubernetes®, lead to larger tech stacks, which increases the attack surface.

At the same time, APIs—which have also existed for decades but have never been as central to applications as they are today—have become increasingly critical for connecting web applications to external resources, as well as for managing internal communication. The average application now depends on between 10 and 15 individual APIs, TechCrunch reports. APIs also expand the attack surface of web applications and increase security challenges surrounding authentication, authorization, and data privacy.

Widespread adoption of public clouds for hosting web applications has also introduced significant new security challenges. In an age when 94% of enterprises are using public cloud, neatly segmenting applications behind firewalls simply doesn't work any longer. Today's developers and security teams must contend with "perimeterless" networks, where applications are continuously exposed to internet-borne threats.

True, some parts of some web applications can still be protected by firewalls, for example, microservices that don't communicate externally can typically be placed behind a firewall. However, in cases where microservices need to upload or download data from an object storage bucket hosted in a public cloud or interact with a third-party authorization API to log in users, traffic can't be kept within the boundaries of an internal network.

Traditional Web Application Security Is Broken

Traditional approaches to web application security like legacy web application firewalls (WAFs) simply don't work—at least not on their own. Although principles defined in the OWASP Top Ten Web App Security Risks still provide useful guidance, today's organizations must take additional measures to stay ahead of the security risks that threaten web applications and the APIs that power those applications.

Modern teams need a web application and API security strategy that enables:

- Holistic protection regardless of where and how an application is deployed—whether in a public, private, or hybrid cloud environment.
- Security at scale because security needs to scale as the application scales.
- Mitigation of security risks within API calls, no matter which API protocols—REST, GraphQL, gRPC, and so on—an application depends on.
- Application workload protection such as hosts, VMs, containers, and serverless from vulnerabilities and at runtime.

This is not to say that conventional security practices for web applications and APIs—like using encryption to help secure data or enforcing strong authentication and authorization requirements—should be thrown out the window. These practices still have a place in modern web application security. Tools like traditional WAFs and security monitoring and auditing solutions still help to mitigate certain types of risks.

Modernizing Web Application and API Security

Gaining full control over modern web application security requires going beyond conventional strategies and tools. Securing the modern web requires the embrace of additional practices tailored to meet the security challenges that arise in complex, distributed, large-scale cloud native applications and the environments that host them.

This e-book describes five key best practices that organizations should embrace to secure modern web applications and APIs. These practices enable a more holistic, agile, and modern approach to security than a conventional WAF or API gateway. By extension, they enhance not just the overall security posture of an organization, but also provide flexibility to application and cloud security teams for their unique architectures.

No two companies' security requirements are identical, so the most important web application and API security requirements for your organization will depend on exactly which types of workloads you run and how you manage them. The guidelines described below will help businesses modernize their approach to securing web applications and APIs on any cloud native architecture as well as ensure they are keeping up with the new complexities and threat categories they will face in a cloud-first, API-centric world.

Best Practice 1: Leverage Cloud-Agnostic Security Solutions

The vast majority (92%) of businesses now use multiple public clouds. Another 80% have embraced hybrid cloud architectures that pair private cloud infrastructure with public cloud resources.

For web application and API security, this means that security tools that only work within a single cloud environment no longer suffice. Today's teams need a cloud-agnostic approach to security. They must be able to identify and remediate security risks across any and all cloud environments and architectures.

This can be difficult, given that each public cloud uses different names for its workloads, different virtual network configurations, and so on. Nonetheless, being able to determine malicious activity on any type of cloud is the only way to ensure that web applications and APIs are secure, no matter where you choose to host them.

To achieve this goal, the solution must run at the workload level instead of the edge. In other words, it must operate from within each workload, whether that is a host, VM, container, or serverless function. Only a workload-level solution can discover all the exposed web applications and APIs as well as which risks—such as access control misconfigurations or insecure API authentication techniques—affect them.

When you apply security controls at the workload level, you get a security strategy that works across any kind of environment—including multicloud and hybrid cloud. This ensures the same level of security across diverse infrastructures. You don't have to worry where the web application resides.

Best Practice 2: Use Priority-Based Risk Management

There are tens of thousands of known security vulnerabilities recorded in databases such as NIST NVD, and more are being discovered all the time. In 2021 alone, nearly 22,000 new vulnerabilities were published.

If those numbers sound intimidating, the good news is that not all security vulnerabilities are created equal. Only a small number are considered critical, and most vulnerabilities can only be exploited under certain conditions. Whether your web applications and APIs will be affected depends on factors like how your environments are configured, the versions of software libraries and the type of attack (e.g., Cross-site scripting or SQL injection).

Determining whether a security vulnerability actually poses a threat to your application—and the severity of the threat—is essential for responding efficiently and effectively to threats. By prioritizing vulnerabilities that pose the highest risk, you can optimize the relationship between the time your team spends managing risks and the security posture enhancements gained from that effort.

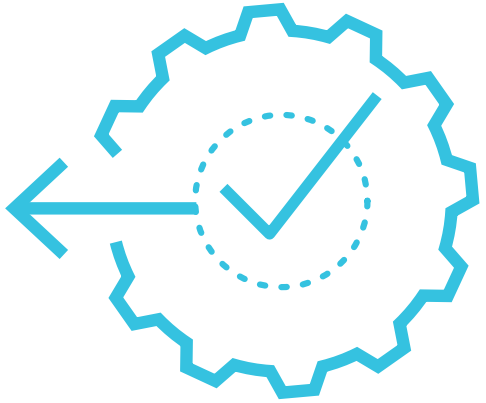
Modern teams should leverage security tools that don't just reveal vulnerabilities like risks in the OWASP Top 10 but also protect against vulnerabilities based on the level of threat they pose to a given environment. For example, vulnerabilities that enable remote execution of arbitrary code or that provide access to sensitive data without any kind of authentication are usually more severe than logging and monitoring failures. By understanding which vulnerabilities require urgent attention, application and cloud security teams can react intelligently to the dizzying array of security threats they face.

Best Practice 3: Shift Web Application Security "Left"

While you can—and should—monitor and inspect web application and API traffic within production environments to detect risks, you must also strive, wherever possible, to catch risks before applications are actually deployed into production.

This practice, which is known as shifting security "left" because it allows teams to discover risks earlier in the software development lifecycle, ensures that security issues can be remediated before they impact production environments. In addition, "shift left" facilitates faster and more efficient resolution of security problems because it's typically easier and less costly to remediate risks before software is in production.

Shifting security left requires application security teams to collaborate closely with development and DevOps teams to detect risks earlier in the software lifecycle. To achieve this level of collaboration, customers require a solution that is able to analyze code repositories and image libraries, and integrate with popular CI/CD tools like VS Code, Jenkins, and CircleCI, to name a few. Shifting security left helps to surface vulnerabilities early in the software delivery pipeline, allowing teams to catch vulnerabilities and misconfigurations before they are deployed to production.

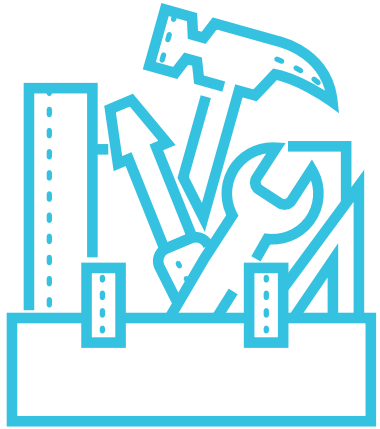


Best Practice 4: Embrace Comprehensive Security Tooling

You can attempt to manage each type of security threat facing your web applications separately. You could use one tool for OWASP Top 10 risks, another solution to protect against denial of service (DoS) attacks, and yet another for protecting against API or bot attacks. Given the complexity and scale of modern web application environments, however, relying on a collection of point solutions is often a mistake. Not only is it more difficult to manage multiple security tools, it's also harder to avoid oversights, gaps, and inconsistencies as you attempt to juggle disparate security solutions.

A better approach is to deploy a comprehensive, integrated set of security tools that enable a wide range of web application and API protections—from application firewalls to vulnerability detection to API security and beyond—in a single solution. You'll save time and energy and enjoy higher confidence that you're not overlooking critical types of protection.

The ability to manage all security risks through a single solution is of critical importance because you never know which threats you will actually face. You need to be sure you are prepared to handle whatever arises. You could face a botnet attack today, an access control exploit tomorrow, and a security misconfiguration issue the day after that. A comprehensive web application and API security solution ensures that you're ready to address multiple threat vectors, from Layer 7 attacks on the application down to the API communications.



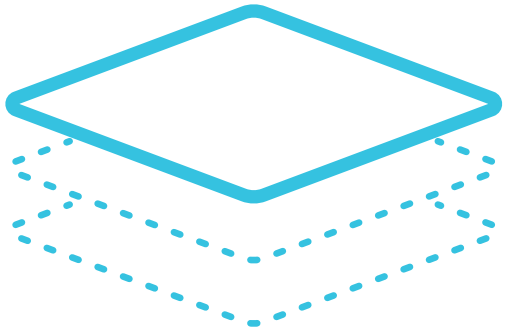
Best Practice 5: Use Multiple Layers of Defense

While it may be tempting to believe that you can prevent all web-based attacks, the fact is that no single tool or practice can guarantee 100% protection. If this were possible, we wouldn't be living in a world where there are more than 2,200 attacks per day, on average.

What you can do, though, is deploy multiple layers of defense. Multiple layers help ensure that even if attackers manage to circumvent one protection, they won't actually compromise your entire environment. If a breach does happen, multiple layers of defense can prevent escalation from affecting additional resources.

Another way to look at the benefits of a layered security approach is the "Swiss cheese paradigm," in which security layers are compared to slices of Swiss cheese stacked one after the other. In this analogy, the cybersecurity risk of a threat becoming a reality is mitigated by the differing layers and types of defenses, which are "layered" on top of each other. Therefore, weaknesses in one defense do not allow a risk to materialize since other defenses also exist to prevent a single point of failure.

Your defense layers should start with visibility and monitoring of your attack surfaces, which can auto-discover all the web applications and API endpoints within your environment. Defense layers should also include policies for both north-south and east-west traffic to block malicious threats, whether they originate from the internet or from within your applications. Another layer of vulnerability and compliance scanning, along with strong authentication, adds further protection to your applications. In addition, you need to secure your workloads or the infrastructure layer, such as the hosts, VMs, containers, and serverless functions that help host your applications. Together, multiple layers of protection provide defense in depth, thus decreasing the chances of a successful attack.



Conclusion

Security teams' jobs would be easier if we still lived in the world of simple monolithic web applications, but that world is gone. The complexity of today's cloud native, API-centric web applications, as well as the microservices they leverage, is a world full of new security challenges.

This new world requires new security strategies and solutions to complement conventional approaches to security. These new practices and tools must enable scalable, flexible, multilayered security that works for any type of workload in any type of environment or cloud architecture.

Prisma® Cloud Web Application and API Security is integrated into the Palo Alto Networks Cloud Native Application Protection Platform (CNAPP) and provides a modern approach to web application and API security. This module is the industry's only integrated solution to provide comprehensive detection and protection of web applications and APIs for any cloud native architecture. Security, IT, and DevOps teams can confidently leverage best-in-class protection for all their web applications and APIs, seamlessly integrated into their CI/CD pipeline.

You can learn more about Prisma Cloud Web Application and API Security on our website and in our documentation, and see Prisma Cloud in action by requesting a free trial.

paloalto® NETWORKS

3000 Tannery Way
Santa Clara, CA 95054

Main: +1.408.753.4000
Sales: +1.866.320.4788
Support: +1.866.898.9087

www.paloaltonetworks.com