NGINX
Part of F5

EBOOK

Modern App and API Security

Incorporating Protection During Development

By the security experts at NGINX and F5, Inc.

NGINX is a part of F5

Table of Contents

Foreword

Application security is a key priority for us here at NGINX. We see it as vital to eliminating vulnerability to security threats such as unauthorized access and data modification. Increasingly, applications are accessible over networks rather than only within a secured perimeter like a data center. This hugely increases both the app's attack surface and the number of potentially malicious users trying to exploit it.

Sometimes treated as an afterthought by teams designing and building applications, security must instead be integrated into development processes and CI/CD pipelines. Imposing application security policies during development minimizes the likelihood that unauthorized users will be able to access, steal, modify, or delete sensitive data when apps go into production.

The most basic software countermeasure is a web application firewall (WAF). A WAF inspects and filters traffic between web applications and end users. A WAF operates through a set of rules often called policies, which filter out malicious traffic to protect against exploitation of vulnerabilities in the application. The value of a WAF comes in part from the speed and ease with which policies can be modified or rate limits imposed, enabling faster response to attacks such as DDoS.

In many organizations, DevOps teams view SecOps as a roadblock they need to get around to deliver apps at the velocity required to keep the company competitive. When security is integrated in the development process, however, we can adopt a model where SecOps provides guidelines and policies that DevOps can implement in their apps as they see fit. This "self-service" model minimizes friction between teams and increases development velocity.

We've implemented this model in NGINX App Protect (NAP) – a lightweight WAF for modern applications that's easily integrated into CI/CD pipelines. With NAP you can non-disruptively enforce SecOps-authorized security in the DevOps CI/CD process, and deploy and manage app security controls across distributed and modern app environments such as containers and microservices.

In this eBook we explore the security challenges facing enterprises today, and how NGINX software helps you deal with them.

Rajiv Kapoor
Senior Product Marketing Manager, NGINX (part of F5)

1. Can Application Security Be Pain Free?

By Rajiv Kapoor, Senior Product Marketing Manager, NGINX (part of F5)

Application security is hard, and it's even harder to do right. If it were easy and straightforward, executives wouldn't be held accountable for breaches with jail time! In today's digital-first world, securing modern applications against cybersecurity threats has become one of the most pressing challenges for enterprises to overcome.

Organizations are continuing their efforts to keep attackers at bay – cybersecurity spending is expected to reach $123 billion in 2020, and cloud security is projected to grow by 33% despite the economic downturn caused by the global pandemic.

But the odds are tough:

- More than 20% of data breaches discovered during the last year occurred due to code errors, and over 40% of those attacks targeted web applications.
- Hackers launch an attack every 39 seconds, an average of 2,244 times per day.
- On average, only 5% of the apps in a company's portfolio are properly protected.
- The average cost of a data breach in 2020 was $3.86 million per company.
- 39% of British businesses have already fired employees for security breaches during the pandemic.

With new breaches hitting the headlines every day, business leaders not only want – but need – to better prioritize security in their processes.

IF EVERYONE IS ALREADY DOING APPLICATION SECURITY, WHY ARE SO FEW DOING IT WELL?

So, if everyone is already doing application security, why are so few doing it well?

DIGITAL DISRUPTION BRINGS NEW SECURITY CHALLENGES

Over the last decade, application development has undergone a massive transformation in response to ever more demanding business expectations. More than half of organizations now claim that without applications, they cannot operate. Another 67% believe that digital transformation efforts, such as IT and business process optimization, improves release velocity of new products and services.

As a result, development teams are using advanced technologies like cloud computing and microservices, in conjunction with DevOps principles, to innovate faster and stay competitive in an increasingly crowded market.

But like most things, this progress has come at a cost. The fast pace of DevOps has left security teams scrambling to keep pace and still install appropriate security guardrails. Traditional security practices used to occur in the final stages before release, but this no longer works well with the rapid release cycles of agile software development. For instance, Amazon reached 50 million production deployments per year even as far back as 2014 – that comes out to about one release every second.

It's no surprise that the speed of development is fast outpacing the rate at which security teams can check configurations and scan for vulnerability, especially considering that developers now outnumber security professionals 500 to 1.

MODERN APPLICATION ENVIRONMENTS REPRESENT A NEARLY INFINITE ATTACK SURFACE

Additionally, modern application environments represent a nearly infinite attack surface. 87% of enterprises are now multi-cloud, and most are struggling to provide consistent security across multiple application architectures and infrastructure. There is no margin for error – organizations must stop attacks every time or risk a breach, public embarrassment, or billions of dollars in damages. Malicious actors only need to be successful once to win.

The pressure to keep up, and do so securely, has led to a gradual reframing of DevOps as DevSecOps, where security is "shifted left" and introduced earlier in the software development life cycle, where capacity is more flexible and open to change. In other words, security is baked into the processes and tools so that issues can be found just like any other defect and addressed more efficiently.

ORGANIZATIONS ARE STRUGGLING TO ACHIEVE SECURE DEVOPS

Ask someone to describe DevSecOps, and you might get one of the following answers:

- "Bringing security to DevOps"
- "Security as code"
- "Everyone is responsible for IT security"

But understanding the need to incorporate security practices earlier in development is very different from implementing it – and DevSecOps has proved to be far more nuanced in reality. While most businesses understand the intent of DevSecOps, they are still unsure of how to go about actually doing it. In fact, just 14% fully integrate security throughout the software development lifecycle.

Recent research also shows that while 65% of security teams report shifting left, less than a fifth are doing the scans necessary to verify that claim. The same report suggests that most security teams don't have processes in place to monitor and protect cutting-edge application technologies, such as microservices, APIs, and cloud-native/serverless.

NEARLY HALF OF ENTERPRISES ADMIT TO CONSCOUSLY DEPLOYING VULNERABLE APPLICATIONS

This lack of visibility leaves security organizations running blind when problems surface in production – and the later in the development cycle vulnerabilities are discovered, the more costly it is to fix them. A study by IBM System Science Institute suggests that compared to a defect identified during early design phases, fixing a defect found during implementation can cost 6x as much, and a defect uncovered in production 100x.

Even more disturbing, nearly half of enterprises admit to consciously deploying vulnerable applications to meet tight deadlines.

Everyone may want secure applications, but it's clear that compliance and security oversight are often overlooked, even deliberately avoided, in favor of achieving faster and more frequent deployments.

THE GREAT DEVOPS-SECOPS DIVIDE

One of the most significant issues lies with older, ingrained perceptions. In the siloed, waterfall world of the past, teams worked independently from one another with rigidly scheduled handoffs required between distinct phases. Security operations (SecOps) teams often introduced security functions only during the final stages of the development and release process, resulting in delays. And while cross-team collaboration has had to improve over time to accommodate more agile development practices, the fact remains that development and security teams strive towards different core goals, often causing organizational misalignment – and inevitably, friction.

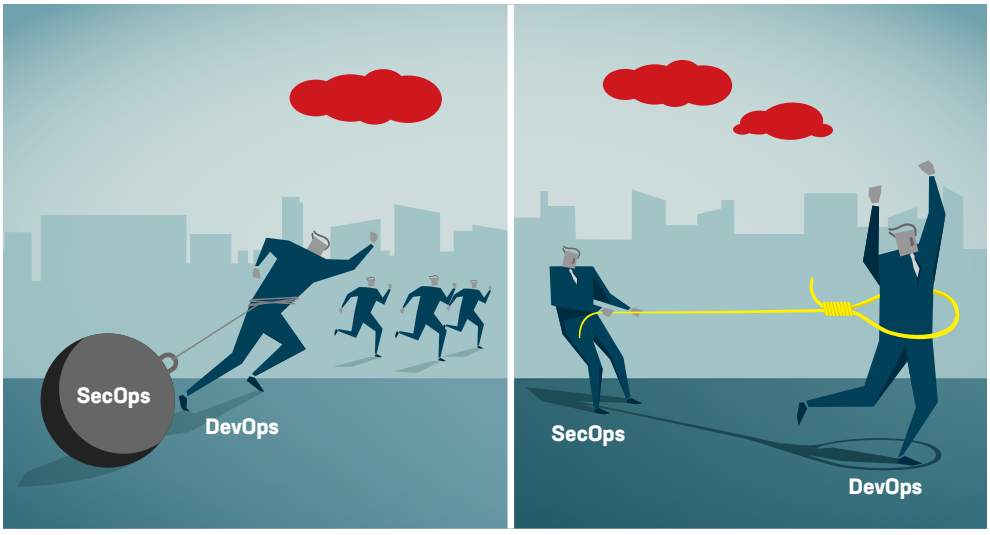


Figure 1: How DevOps Views SecOps and Vice Versa

If DevOps teams are the engine driving innovation, SecOps is often viewed as the brakes that stop forward motion. 48% of technical professionals believe security is a major constraint on the ability to deliver software quickly.

DEVOPS AND SECURITY TEAMS ARE BOTH CONTRIBUTING TO THE COMPANY'S OVERARCHING GOAL

On the other hand, SecOps and application security teams are focused on thorough testing to identify vulnerabilities, potential risks, and possible compliance issues. They feel developers are running rampant, especially with the specter of shadow IT in the application stack looming large – and they struggle to hold developers back as they charge ahead at full speed.

Successfully putting the "Sec" into DevSecOps hinges on changing these older cultural biases, reinforcing the need to embrace security, and empowering teams with the right tooling and automation to make smarter decisions without slowing down the entire organization. Despite apparently different priorities, DevOps and security teams are both contributing to the company's overarching goal – producing a high-quality, timely product. The difference lies in how they are used to measuring and defining what that means.

The truth of the matter is that development won't be slowing down any time soon. 38% of developers now release monthly or faster, and 54% of containers live for 5 minutes or less. Continuing to view security as a separate entity bolted onto the code, instead of a key feature that must be embedded end-to-end, only slows processes down and reduces efficiency.

More than ever before, now is the time to transform how security is applied to DevOps.

BRIDGING THE GAP TO DELIVER SPEED *AND* SECURITY

The core questions for app security then become: What makes it easier for DevOps teams and application security teams to collaborate? What does built-in security really look like?

For organizations that want to improve their DevOps security, here are some key points to focus on:

Automate Security as Much as Possible

Invest in security solutions that can be embedded directly into the CI/CD pipeline using automation. This makes it easier to secure apps without having to sacrifice as much development speed for the sake of security. Adopting technologies like static code analysis, dynamic analysis, and pen testing reduces risks, and alerts developers to potential problems. There will never be enough security professionals to handle all security issues on their own, so use automation whenever possible.

Build Security as a Guardrail, Not a Gate

Provide appropriate guidance and tooling so that security becomes a guardrail rather than a gate. For example, ensuring that DevOps teams have access to templated policies enables them to align their applications with security requirements right from the start, without adding unnecessary delays to the development process. Applications and security policies can also be tested as part of the CI/CD pipeline, so they are checked like any other functional specification. In short, it's important to provide developers with everything they need to create

and test applications with the appropriate security controls applied, or they won't do it. Empowering developers with a better understanding of security and self-service compliance reduces vulnerabilities and risk to the organization. Developers always want to drive faster, so make it possible for them to speed safely.

Abstract Security to Enable More Proactive Responsibility

The average developer cannot be expected to have the level of expertise necessary to stay current with all the latest security trends – they have enough trouble keeping their programming skills up to date. Reduce complexity and make it easier to gain developer buy-in by selecting solutions that provide simplified, easy-to-understand insights within the CI/CD feedback loop.

Make Security Adaptable, Scalable, and Reliable

Choose solutions that offer consistent, centralized, and self-service security for any environment, including modern and distributed environments. For instance, AI-driven security policy engines are one way to enable the adaptability necessary to support rapid change in applications resulting from CI/CD methodologies. Being able to adapt security policies in response to the latest attacks and identify dependencies makes it easier to assess risk and take action faster.

MODERN APPLICATION SECURITY MUST SET DEVOPS FREE

IN TODAY'S WORLD, INTEGRATED SECURITY MUST BECOME A NORMAL PART OF ANY DEVOPS IMPLEMENTATION

Gone are the days when security could simply be bolted on at the end of a process – in today's world, integrated security must become a normal part of any DevOps implementation. Making security frictionless and adaptable enables development teams to power ahead without fear. Instead of a painful extra step that must be dealt with, modern application security can be a robust support system that empowers organizations to reach their business goals and guides them to even higher heights.

2. The Importance of Securing Real-Time APIs

By Rajiv Kapoor, Senior Product Marketing Manager, NGINX (part of F5)

Increasingly, digital transformation and customer expectations are driving organizations to employ creative approaches to serving the needs of a diverse mix of end users and experiences. From telemedicine to online banking, real-time APIs are the foundation upon which digital business is built, allowing app developers to create apps that can serve the needs of their customers.

With the explosion of apps in the digital world that are served instantly by APIs, the need to rapidly detect and protect against API breaches becomes critical.

APIs form the chassis for modern applications. They are everywhere, enabling developers to obtain valuable information from other software components and integrate it into their applications, for example embedding Google Maps in a rideshare app or YouTube videos in a web page. APIs are key components at every stage of a user's interaction with an app, from logging in to leaving feedback. The rise of 5G, with its promise of high speeds (1 Gbps), is likely to make today's impatient users even less tolerant of poor app performance. API-driven businesses that don't achieve real-time API responsiveness – which we define as processing an API call end-to-end in under 30ms – are sure to lose digital market, and the loss of revenue might even put their digital transformation efforts at risk.

APIs ARE KEY COMPONENTS AT EVERY STAGE OF A USER'S INTERACTION WITH AN APP

Like all things created with good intentions, the prominent use of APIs has its downside – it provides actors with malicious intentions a new avenue for exploiting applications. Gartner has predicted that by 2022 API abuses will be the most frequent attack vector against enterprise web applications, resulting in data breaches.

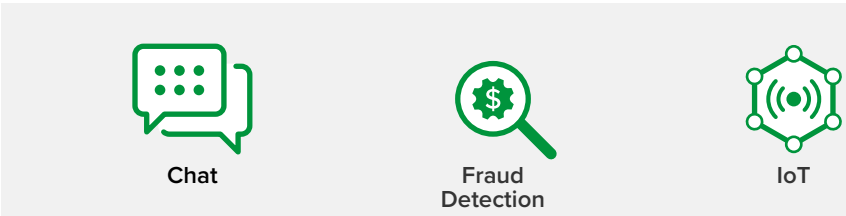


Figure 2: Three Everyday Use Cases for Real-Time APIs

WHY IS API SECURITY IMPORTANT?

APIs ARE OFTEN GRANTED ACCESS TO ALL DATA WITHIN THE APPLICATION ENVIRONMENT

A combination of factors makes APIs rich targets for security attacks. One of the biggest problems is failure to set appropriate access permissions. Because they are not intended for direct access by users, APIs are often granted access to all data within the application environment. Access is then controlled by granting specific permissions to the users making the initial requests that are translated into API calls, and having the API inherit only those permissions. This works fine until an attacker manages to bypass the user authentication process and access the downstream app directly via the API. Because the API has unrestricted access, the attacker gets visibility into everything.

Like basic HTTP web requests, API calls incorporate URIs, methods, headers, and other parameters. All of these can be abused in an attack. Unfortunately, most typical web attacks, such as injection, credential brute force, parameter tampering, and session snooping work surprisingly well on APIs. To attackers, APIs are an easy target.

HOW DO YOU SECURE APIs?

It's vital to build security into an API at every phase of its lifecycle. During the design and development stages, engineers need to build in the logic required for integrating with the WAF, bot protection, API management solution, API gateway, and other tools that will secure the API as it's delivered in development, testing, and production environments.

Figure 3: Four Components of Secure APIs

You then deploy those technologies to protect the API during delivery, as discussed in the following sections.

WAF

A WAF recognizes requests that are in fact illegitimate, designed not to exercise the API's intended functionality but to exploit vulnerabilities in application code that allow attackers to steal information or execute malicious code. It's crucial that at minimum the WAF protects against the most common attack types, like the OWASP API Security Top 10.

NGINX App Protect is based on F5's Advanced WAF product. Optimized for CI/CD and DevOps workflows, it supports XML, JSON, text, and HTML request and response payloads. Its advanced API protection profiles protect against attacks with parsing and structure enforcement, attack signatures, method enforcement, and path enforcement.

Bot Protection

HTTP APIs can be subject to bot and other forms of malicious or unwanted automation-based traffic. Shape, now part of F5, offers API Defense™ as a solution that provides visibility, throttling, and mitigation options to protect HTTP-based APIs from bots and other forms of automated attacks that generate online fraud and application abuse.

API Management

Among other functions, API management solutions provide the interface for defining security policies which the API gateway then applies as it processes API calls.

The NGINX Controller API Management Module includes important protections like implicit URI allowlisting based on the API specification, as well as programmable rate limiting, multiple rate-limiting policies, and throttling to protect against denial of service attacks.

API Gateway

An API gateway like NGINX Plus secures API calls in its role of guardian responsible for three key functions.

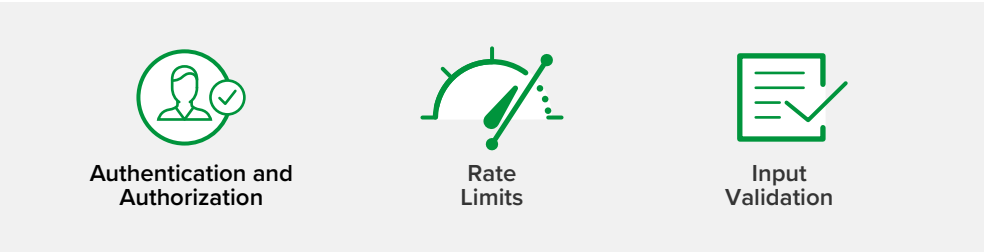


Figure 4: Key Functions of API Gateways

1. **Authentication and Authorization**

   API authentication is about allowing access only to recognized clients – those that can prove they are who they claim to be.

   Because authentication is not core to what an API does, it makes sense to perform it outside the application code. This frees API developers from having to write their own authentication code and means you can centrally manage authentication for all APIs while still making authentication requirements flexible. For example, you might allow unauthenticated use of the API that returns game scores at a sports website, but you definitely need to authenticate the people who use an API to edit the content.

   Now let's look at the difference between authentication and authorization. Authentication is the process of verifying user identity. Authorization is what comes next – determining which actions a particular user is entitled to perform and conveying that information to the server.

2. **Rate Limits**

   Rate limits control how frequently a given client can make an API call. They have two main purposes: protecting backend services from being overloaded and ensuring fair use for clients. An example of rate limiting might be to allow 100 transactions per second during periods of heavy demand. Rate limits can be applied to individual user names, specific IP addresses or ranges, or to all users (for example, during peak traffic times).

3. **Input Validation**

   Input validation is verifying that input supplied by a user or application is correct: consists of the right type of characters (digits, letters, punctuation), is the right size, is one of a predefined set of acceptable values, is consistent with another value being provided, and so on. For example, you might check that the zip code matches the supplied address, or that a birthdate is not in the future. Input validation prevents improperly formed data from entering an information system and threatening its integrity. It's also an important way to detect malicious users, who can then be blocked from making further requests.

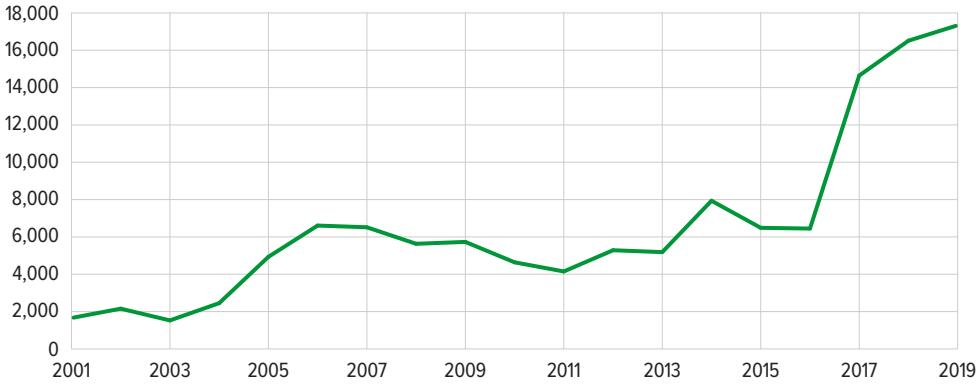
APIs are a strategic necessity to give your business the agility and speed needed to succeed in today's business environment. But with the increasing cost of security breaches, organizations want to ensure that exposing their data via APIs does not create security risk which impacts their top line and bottom line.

3. Rising Open Source Software Vulnerabilities Require a Modern WAF

By Kevin Jones, Senior Product Manager, NGINX (part of F5)

Today, security has become a crucial part of the development, deployment, and delivery of web applications. We deliver our applications with an ever-increasing velocity that allows us to stay competitive. In a constant process of transformation, we adopt new technologies and methodologies in order to stay agile – often including the latest and greatest open source tooling, components, and application stacks. This velocity enables us to provide what our customers want and need from our applications, but what are the risks of such fast-paced innovation?

Figure 5: Number of CVE Vulnerability Incidents, 2001 - 2019

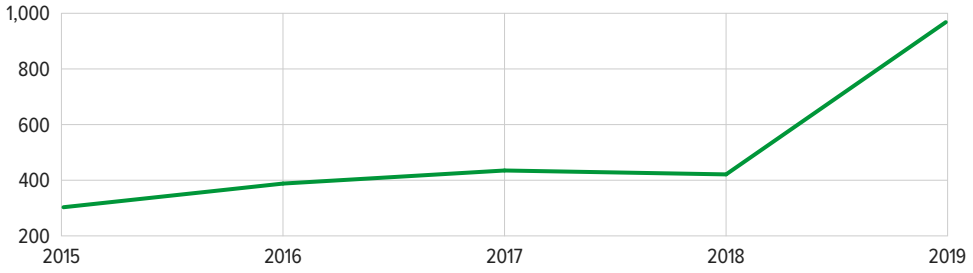


Source: National Institute of Standards and Technology

THE RISE OF OPEN SOURCE VULNERABILITIES

RiskSense, a vulnerability management firm based in Silicon Valley, recently published a study titled The Dark Reality of Open Source. The goal was to identify the threats to application security that come from open source products. Alarmingly, the number of Common Vulnerabilities and Exposures (CVEs) for open source software increased by 130% between 2018 and 2019, from 421 to 968. Moreover, it took 54 days on average for vulnerabilities to be added to the National Vulnerability Database after they were publicly disclosed, leaving organizations that use the software "exposed to serious application security risks for almost two months".

Figure 6: Open Source Vulnerabilities per Year, 2015 - 2019



Source: RiskSense Spotlight Report: The Dark Reality of Open Source

In recent years, open source software vulnerabilities have been the cause of many major data breaches, such as the Apache Struts exploit (CVE-2017-5638). This exploit allowed attackers to pass specific HTTP request data containing Object-Graph Navigation Language (OGNL), which allows reading and setting properties within Java, as well as method execution. This allowed attackers to perform a remote code execution (RCE) attack and in 2017 led to the breach of over 143 million accounts at Equifax. Given the number of vulnerabilities in the open source landscape, how do you protect your users, your network, and most importantly your data, against these malicious attacks?

HOW WAFs HELP WITH OPEN SOURCE SOFTWARE SECURITY

THE MOST IMPORTANT PART OF THE PUZZLE IS A WEB APPLICATION FIREWALL

The overall solution to security can be complicated, but the most important piece of the puzzle is a web application firewall (WAF). A WAF protects your web applications by filtering, monitoring, and blocking malicious HTTP/S traffic destined for the web application, and preventing unauthorized data from leaving the app. It does this by adhering to a set of policies that help determine what traffic is malicious and what traffic is safe. As with any security strategy, you need defense in depth. A WAF doesn't block all exploits. Patching and maintaining a supported version ensures you can mitigate zero-day exploits.

Let's look at another example. According to the Open Web Application Security Project (OWASP), one of the most common and malicious attacks that can performed on your web application is SQL injection. The attacker exploits a vulnerability to execute malicious SQL statements from within the web application, which in turn expose sensitive data from a database. This attack is so common that it is #1 on the list of OWASP Top 10 attacks.

Here at NGINX, we know how important security is to our users. When creating NGINX App Protect, we leveraged NGINX's powerful architecture to help you achieve your security goals. NGINX App Protect is a modern, high-performance, and reliable application security solution. With a design based on F5's market-leading WAF, the product runs natively on top of NGINX Plus and integrates security controls directly into your application. When developing NGINX App Protect, we kept the same philosophy as with previous NGINX products, focusing on performance, scalability, and a lightweight architecture.

For further discussion of NGINX App Protect and how it can provide security protection throughout your infrastructure, see the following chapters.

4. Introducing NGINX App Protect: Advanced F5 Application Security for NGINX Plus

By Mark Campbell, Senior Manager, Product Marketing, F5, Inc.

Companies going through digital transformation have clear business imperatives. They include improving the customer experience with modern business applications, adopting agile practices to outpace competitors in the market, and leveraging market advantages to drive new revenue streams. Supporting these efforts are new application architectures that increase development efficiency and incorporate containers, microservices, and APIs.

SECURITY CONTROLS FOR TRADITIONAL APPLICATIONS DON'T ALWAYS MAP WELL TO BUSINESS REQUIREMENTS

For modern applications, agility and time to market are key. Security is often a secondary consideration, or is neglected entirely. Why? Security controls for traditional applications don't always map well to business requirements. For example, the kind of sophisticated web application firewalls (WAFs) that are traditionally configured and operated by SecOps teams are not generally well suited for agile applications deployed by the DevOps teams supporting specific lines of business. The result can be inadequate or misconfigured security, delays in go-to-market timing, and a poor user experience.

INTRODUCING NGINX APP PROTECT

"We're very excited to make available yet another product offering that demonstrates why NGINX and F5 are better together, just a few months after our vanguard milestone release of NGINX Controller 3.0 in January," says Gus Robertson, Senior Vice President and General Manager of NGINX. "We intend to continue our accelerated pace of innovation, delivering more and more value to our customers as they continue their digital transformation journeys."

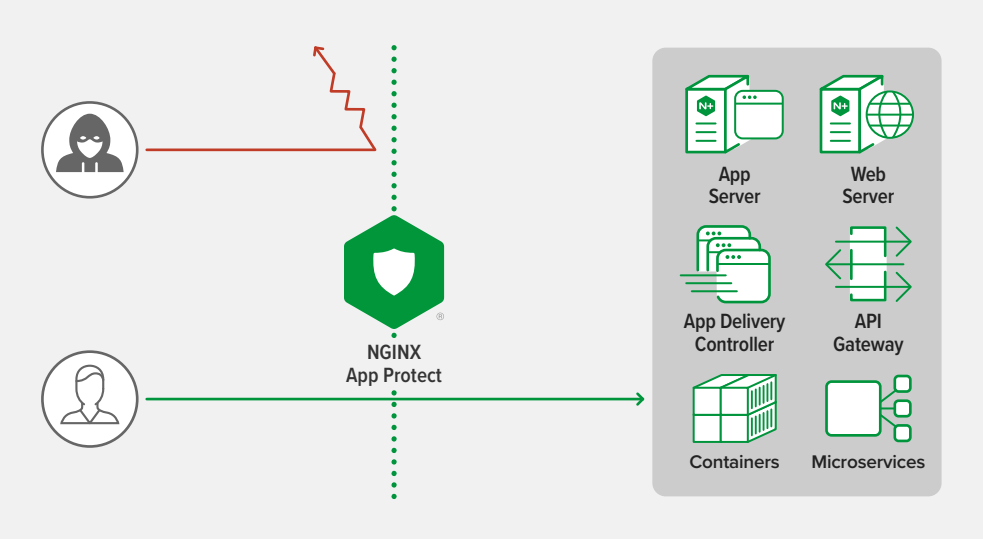


Figure 7: Use NGINX App Protect to Secure Apps with Rapid Threat Defense and Analytics

NGINX App Protect is a new application security solution that combines the efficacy of advanced F5 WAF technology with the agility and performance of NGINX Plus.

The solution runs natively on NGINX Plus and addresses some of the most difficult challenges facing modern DevOps environments:

- Integrating security controls directly into the development automation pipeline
- Applying and managing security for modern and distributed application environments such as containers and microservices
- Providing the right level of security controls without impacting release and go-to-market velocity
- Complying with security and regulatory requirements

STRONG F5 APPLICATION SECURITY

YOU CAN CONFIDENTLY DEPLOY NGINX APP PROTECT SIGNATURES IN "BLOCKING MODE"

NGINX App Protect's security controls are ported directly from F5's advanced WAF technology, providing a significant upgrade from community-supported solutions like ModSecurity. Its comprehensive set of WAF attack signatures has been extensively field-tested and proven to generate virtually no false positives, so you can confidently deploy signatures in "blocking mode" even in production environments. NGINX App Protect protects against the OWASP Top 10 web application security risks, enforces protocol compliance, defends against common evasion techniques, provides denylisting, checks cookies, protects APIs, and prevents sensitive data leakage with F5's Data Guard.

OWASP TOP 10 WEB APPLICATION SECURITY RISKS

1. Injection
2. Broken Authentication
3. Sensitive Data Exposure
4. XML External Entities (XXE)
5. Broken Access Control
6. Security Misconfiguration
7. Cross-Site Scripting (XSS)
8. Insecure Deserialization
9. Using Components with Known Vulnerabilities
10. Insufficient Logging and Monitoring

BUILT FOR MODERN APPLICATIONS

Strong security controls don't help if they can't be implemented in the application's operating environment. NGINX App Protect is built to support modern application deployment topologies. Common deployment modes for NGINX Plus include:

- Load balancer
- API gateway
- Ingress controller for Kubernetes Pods
- Per-Pod proxy for microservices

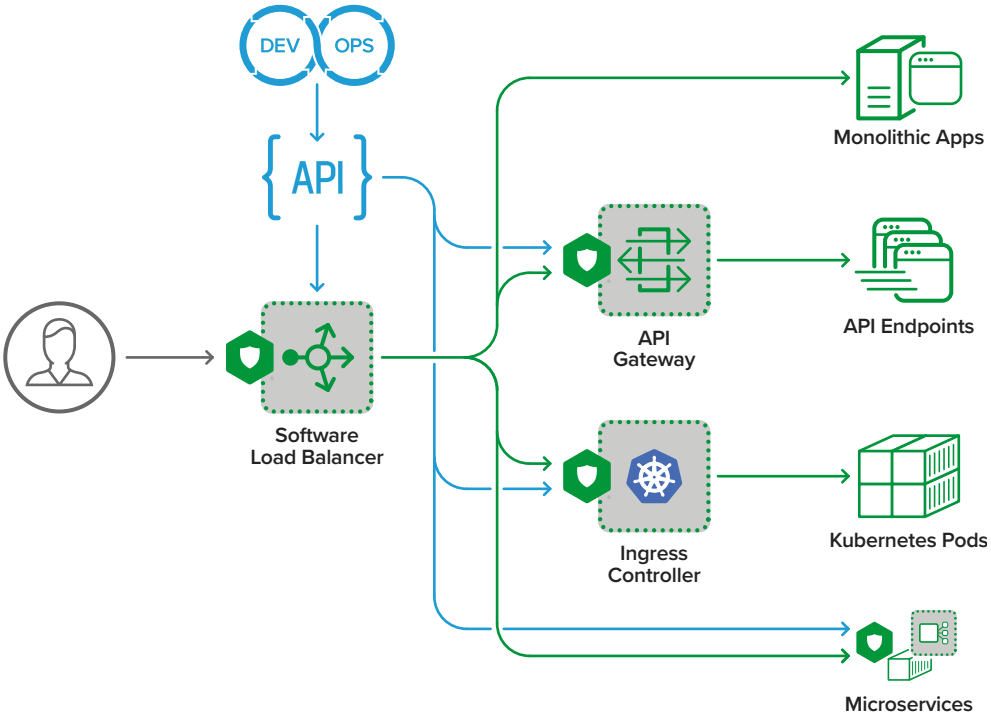


Figure 8: NGINX App Protect Deploys Throughout Your Infrastructure

UNCOMPROMISED SPEED

NGINX APP PROTECT CONTROLS ARE COMPILED INTO BYTECODE, SO TRAFFIC IS PROCESSED LIGHTNING FAST

Unfortunately, you often have to sacrifice performance for security, and vice versa. ModSecurity controls, for example, involve evaluation of regular expressions, so each additional control you enable directly degrades performance – leading many administrators to implement a very small number of controls. In contrast, NGINX App Protect controls are compiled into bytecode, so traffic is processed lightning fast regardless of how many attack signatures you enforce. The net result is up to 20x the throughput and requests per second compared to a ModSecurity implementation with the Core Rules Set v3 enabled.

KEEP DEVOPS FOCUSED ON INNOVATION

The relationship between SecOps and DevOps can often get uncongenial, especially if security requirements get in the way of release velocity. Static application security testing (SAST) and software composition analysis (SCA) are great tools for catching security defects early in development, but many vulnerabilities are not discovered until after applications are pushed through the release gates. Sending apps back to development increases costs and hurts productivity – catching defects while the app is still in the development pipeline is substantially more efficient, whether that involves adjusting the security policy or fixing the code.

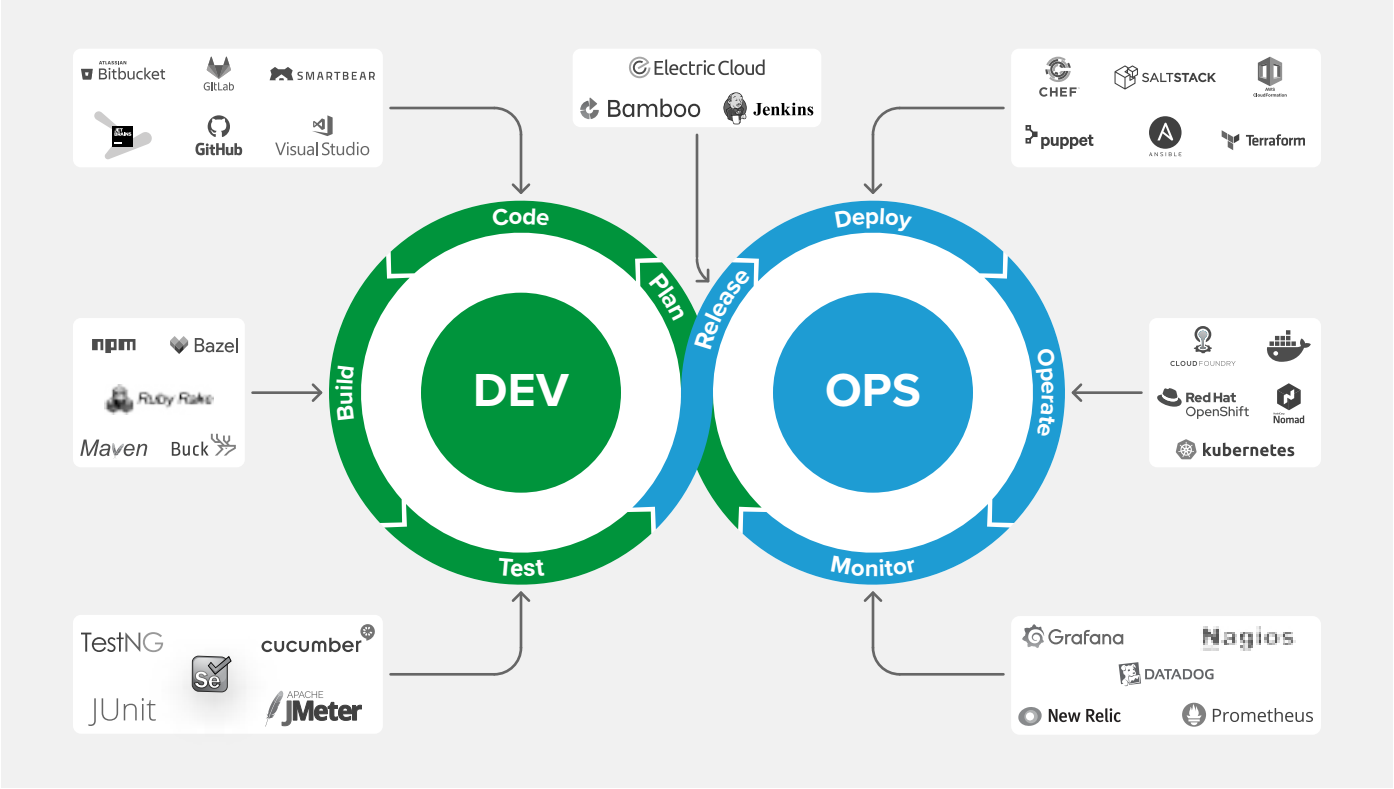


Figure 9: DevOps Application Life Cycle and Ecosystem

NGINX App Protect is DevOps-friendly and integrates into common development pipelines. Using NGINX App Protect's declarative configuration capabilities, security can become part of DevOps CI/CD automation, getting tested just like any other part of an application's functional specification. In essence, the security policy and configuration are consumed as "code" pulled from a source code repository. The SecOps team creates and maintains security policy to ensure the controls required to protect the business are in place. Not only does this help to maintain release velocity, it also helps to bridge gaps between DevOps and SecOps teams.

NGINX App Protect is a modern WAF designed to sit close to your applications, providing additional protection beyond a perimeter WAF. It can be fully integrated into DevOps and CI/CD frameworks to:

- Enable strong security controls to be integrated seamlessly with NGINX Plus
- Outperform other WAFs for improved user experience
- Reduce complexity and tool sprawl while delivering modern apps

5. Achieving PCI DSS Compliance with NGINX App Protect

By Yaniv Sazman, Senior Product Manager, F5, Inc.

Digital transformation has changed the security landscape. Traditional digital security no longer exists as organizations transition from monolithic applications to cloud-native micro-services architectures to increase business agility. Because microservices communicate over the network, modern websites and web applications are more vulnerable to cyberattacks than monoliths and have become one of the easiest ways to compromise the networks of companies of all sizes. Organizations need to find the right balance between security and agility.

The credit card industry continues to be a frequent target for cyberattacks. This chapter discusses the specific security and compliance challenges that enterprises face when they handle credit card transactions, and how technologies like a web application firewall (WAF), and NGINX App Protect in particular, help them meet regulatory requirements.

PCI DSS COMPLIANCE IS CRITICAL TO TODAY'S MODERN APPLICATIONS

The Payment Card Industry Data Security Standard (PCI DSS) describes the actions that all parties involved in processing credit card payments must take to protect cardholder data. The very first requirement is to "Install and maintain a firewall configuration to protect cardholder data". Requirement 6.6 further states that owners of public-facing web applications must protect them by "installing an automated technical solution that detects and prevents web-based attacks (for example, a web application firewall). . .".

INSTALLING A WAF IS NOT A SIMPLE MATTER OF "SET IT AND FORGET IT"

Unfortunately, installing a WAF is not a simple matter of "set it and forget it". There is a wide variety of possible attacks and attackers are constantly coming up with new ones. That makes maintaining PCI DSS compliance one of the most significant challenges faced by modern applications.

Requirement 6.5 of the standard lists the vulnerabilities that a WAF must defend against "at a minimum":

- Injection flaws, particularly SQL injection, but also OS Command Injection, LDAP and XPath injection flaws, and others
- Buffer overflows
- Insecure cryptographic storage
- Insecure communications
- Improper error handling
- Cross-site scripting (XSS)

- Cross-site request forgery (CSRF)
- Broken authentication and session management
- Improper access control (such as insecure direct object references and failure to restrict URL access)

The PCI DSS list doesn't even overlap completely with another commonly used list of vulnerabilities, the Open Web Application Security Project (OWASP) Top 10, which adds XML external entities, misconfiguration (such as using default configs), insecure deserialization, and insufficient logging and monitoring.

NGINX APP PROTECT MEETS AND EXCEEDS PCI DSS REQUIREMENTS

To comply with PCI DSS and protect your apps against the ever-growing set of vulnerabilities, you need a modern WAF solution like NGINX App Protect. It protects against the listed PCI DSS vulnerabilities, the OWASP Top 10, and beyond.

NGINX App Protect is designed for modern infrastructure and can be installed anywhere. It slots directly into your CI/CD pipeline "as code", and being closer to your applications than traditional WAFs enables you to rapidly update security policies. Because NGINX App Protect deploys on all platforms (public and private clouds, VMs, containers, and more) and use cases (including API gateway and Kubernetes Ingress controller), you get consistent performance and the same level of protection across your entire infrastructure.

NGINX APP PROTECT COVERS MORE THAN 6,000 SIGNATURES

NGINX App Protect covers more than 6,000 signatures that are updated at least every two months to cover the latest known attacks.

Also, beyond the signatures, NGINX App Protect:

- Performs HTTP protocol and evasion technique checks on a per-request basis to detect errors such as illegitimate metacharacters in the contents of the HTTP message, invalid length, and more. Such anomalies can indicate a possible attack that is unknown (zero-day) and their presence reinforces other evidence that may exist in the traffic.
- Processes JSON and XML content, and can check the payload for potentially malicious injections.
- Provides a unique capability that prevents responses from exposing sensitive information by masking the data (also known as response scrubbing). We always recommend enabling response scrubbing when the application returns confidential data that must not be exposed.

6. Agile Perimeter Security with NGINX App Protect

By Isaac Noumba, Senior Enterprise Network Engineer (Security) at F5, Inc.

In the context of computer security, the *perimeter* is a conceptual line that establishes a "zone of trust" for applications and other infrastructure components inside it.

In traditional infrastructure environments using a *castle-and-moat* approach, the perimeter separated the intranet (internal network) from the extranet or Internet. The intranet was presumed safe, and threat was assumed to come only from outside it. Security posture was rather static and consisted of establishing fortification around the intranet with packet inspection and access control measures.

EXTERNAL ATTACKERS HAVE FOUND WAYS TO CIRCUMVENT SECURITY CONTROLS

Over time, however, external attackers have found ways to circumvent security controls and compromise components in the intranet, or craft external attacks and make it look like their requests originate from within the perimeter – since the intranet was considered safe. This has motivated a change to a *Zero Trust security model*, where all entities (internal and external) are continually assessed before trust is established. The intranet is no longer assumed safe.

Digital transformation and new app architectures like microservices have also introduced additional security challenges. Many corporate apps are hosted in public clouds, or distributed across the cloud and on-premises topologies, meaning the security infrastructure protecting apps is no longer entirely under the control of a local administrator. As a result, many organizations now establish the perimeter around individual apps (or small groups of apps with direct structural dependencies among them). In this graphic the green dotted line represents the perimeter.

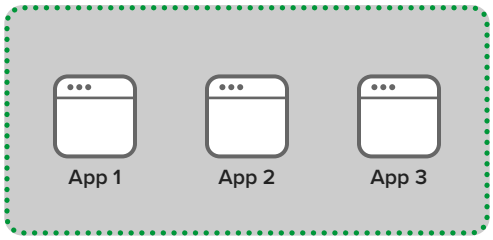
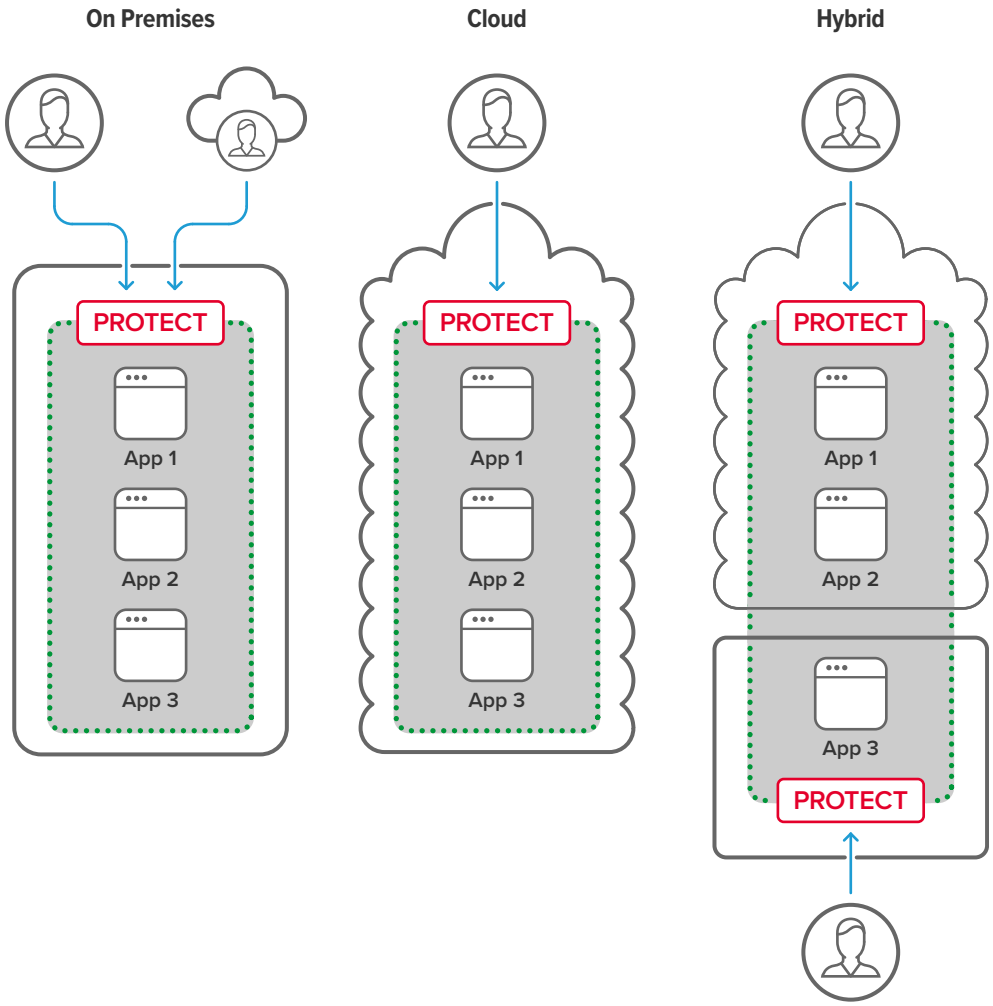


Figure 10: Perimeter Around a Group of Related Apps

Regardless of the architectural model, there's a "gatekeeper" that sits on the perimeter to inspect incoming traffic and enforce security policies that protect the apps inside it. We refer to this gatekeeper as the *edge*. In the three common deployment patterns shown in Figure 11, the edge is represented by the red box enclosing the word **PROTECT**.

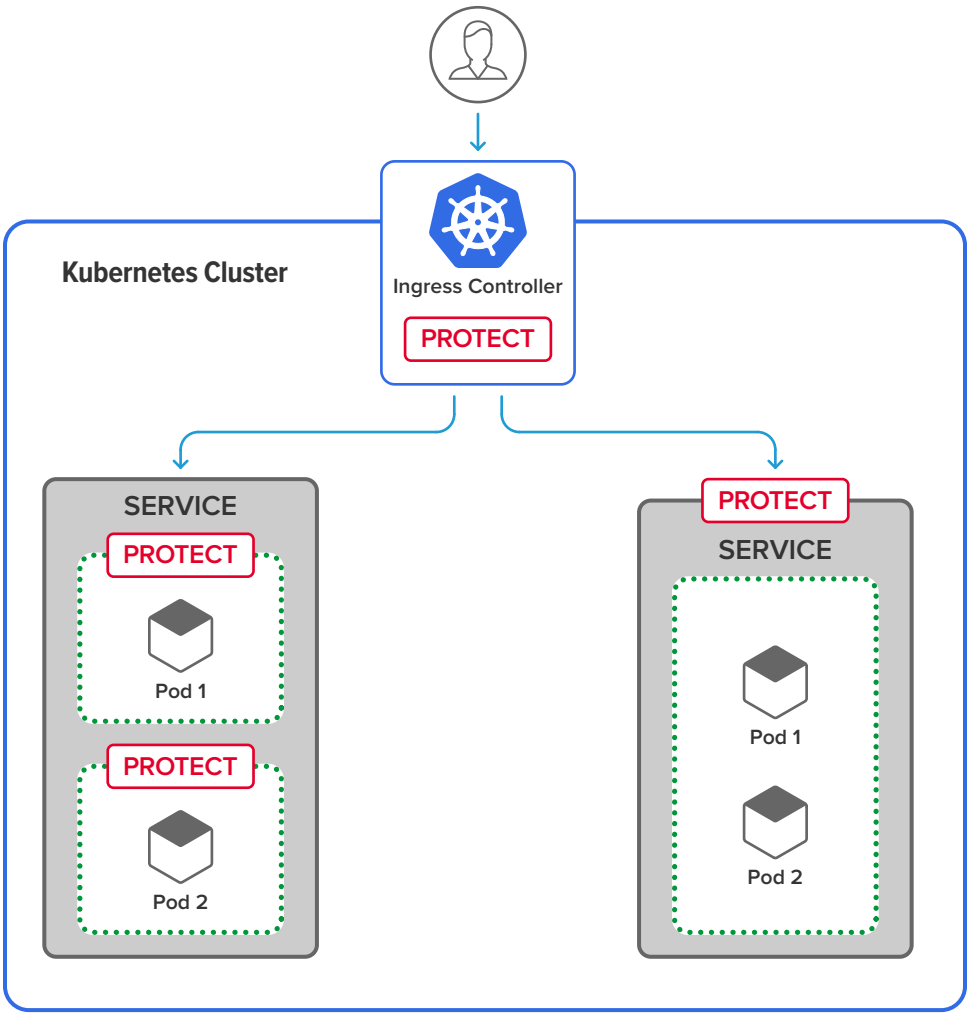
Figure 11: The Edge in Three Common Deployments



In containerized architectures, such as the Kubernetes framework, the same concepts apply. An Ingress controller acts as the edge for an entire Kubernetes cluster, managing access from external clients and routing requests to the Kubernetes services in the cluster. As shown in Figure 12, however, security policies can be enforced at a more granular level within the cluster as well – per-Pod and per-Service:

- With per-Pod protection (depicted on the left), the Pod defines the perimeter containing an app or app component in one or more containers.
- With per-Service protection (depicted on right), a Service exposes the instances of an app deployment through one or more Pods. The perimeter is established around the pods behind the Service.

Figure 12: The Edge in a Kubernetes Cluster



SECURING THE PERIMETER WITH NGINX APP PROTECT

NGINX App Protect is a modern application security solution built on F5's market-leading web application firewall (WAF) technology. With NGINX App Protect, you can enforce security for your apps with agility by protecting the perimeter, regardless of the deployment environment or app architecture: on-premises, cloud, hybrid, microservices-based, or containerized. For more information, see chapter 4, Introducing NGINX App Protect: Advanced F5 Application Security for NGINX Plus.

Implementing security at the edge with NGINX App Protect offers the main advantage of performing security outside of the perimeter. Essentially, traffic inspection and access control occur at the edge to ward off threats before they cross the perimeter. As the last hop before the apps, the edge is where you can best see the type and number of threats against your apps.

The following snippet configures NGINX App Protect to secure three apps (**app1**, **app2**, and **app3**) that are accessed separately within a perimeter:

```
load_module modules/ngx_http_app_protect_module.so;
error_log /var/log/nginx/error.log debug;

http {
    # Enable NGINX App Protect in 'http' context
    app_protect_enable on;

    # Enable remote logging
    app_protect_security_log_enable on;

    # Default JSON security policy
    app_protect_policy_file "/etc/nginx/NginxDefaultPolicy.json";

    # Set remote logging options (in referenced file) and log server
    # IP address/port
    app_protect_security_log "/etc/nginx/log-default.json"
                             syslog:server=127.0.0.1:515;

    server {
        listen 80;
        server_name app1.com;
        # JSON policy for app1
        app_protect_policy_file "/etc/nginx/NginxApp1Policy.json";

        location / {
            proxy_pass http://www.app1.com:8080$request_uri;
        }
    }
```

(continues)

```
    server {
        listen 80;
        server_name app2.com;
        # JSON policy for app2
        app_protect_policy_file "/etc/nginx/NginxApp2Policy.json";

        location / {
            proxy_pass http://www.app2.com:8080$request_uri;
        }
    }
    server {
        listen 80;
        server_name app3.com;
        # JSON policy for app3
        app_protect_policy_file "/etc/nginx/NginxApp3Policy.json";

        location / {
            proxy_pass http://www.app3.com:8080$request_uri;
        }
    }
}
```

The following snippet configures NGINX App Protect to secure **app1**, **app2**, and **app3**, which are embedded in and presented through a single application within a perimeter:

```
load_module modules/ngx_http_app_protect_module.so;
error_log /var/log/nginx/error.log debug;

http {
    server {
        listen  80;
        server_name app.com;

        # Enable NGINX App Protect in 'http' context
        app_protect_enable on;

        # Enable remote logging
        app_protect_security_log_enable on;

        # Default JSON security policy
        app_protect_policy_file "/etc/nginx/NginxDefaultPolicy.json";
        # Set remote logging options (in referenced file) and log
        # server IP address/port
        app_protect_security_log "/etc/nginx/log-default.json"
                                 syslog:server=10.1.20.6:5144;
```

(continues)

```
        location / {
            # Main JSON policy file
            app_protect_policy_file "/etc/nginx/policy/policy_main.json";
            proxy_pass http://app.com$request_uri;
        }
        location /app1 {
            # JSON policy file for app1
            app_protect_policy_file "/etc/nginx/policy/policy_app1.json";
            proxy_pass http://app.com$request_uri;
        }

        location /app2 {
            # JSON policy file for app2
            app_protect_policy_file "/etc/nginx/policy/policy_app2.json";
            proxy_pass http://app.com$request_uri;
        }

        location /app3 {
            # JSON policy file for app3
            app_protect_policy_file "/etc/nginx/policy/policy_app3.json";
            proxy_pass http://app.com$request_uri;
        }
    }
}
```

In both configurations, there is a separate `app_protect_policy_file` directive for each of the apps, assigning each a distinct security policy because they have different security requirements.

For additional NGINX App Protect configurations, see the documentation.

PERIMETER SECURITY IN THE CI/CD PIPELINE WITH NGINX APP PROTECT

NGINX APP PROTECT ADDRESSES THE SECURITY CHALLENGES OF SCALABILITY AND AUTOMATION FOR MODERN APPS

NGINX App Protect addresses the security challenges of scalability and automation for modern apps. By inserting NGINX App Protect directly into multiple integration points in your CI/CD pipeline, you can secure your apps, Pods, and Services closer to their code, bridging the gap between development, operations, and security.

Integrating security directly in your app development cycle enables you to perform and automate security testing to discover the security risks to your apps and app components. You can define security policy enforcement and revert the publication of apps if the security compliance requirements are not met. Effectively, you continuously deliver secure apps by integrating NGINX App Protect into new versions of your apps before they are published.

7. Securing Your Apps in Kubernetes with NGINX App Protect

By Amir Rawdat, Technical Marketing Engineer, NGINX (part of F5)

Businesses know they need to bring services and applications to market quickly because if they don't, a competitor surely will. But web applications are prime targets for cyberattacks, and updating them fast and furiously increases the risk that potential security vulnerabilities slip through QA and make their way into production.

MANY FACTORS MAKE IT CHALLENGING TO CONSISTENTLY APPLY STRONG SECURITY STANDARDS

Many factors make it challenging to consistently apply strong security standards. The pressure to release code quickly into production makes it tempting to let security slide. Over-reliance on automated tools such as vulnerability scanners is dangerous, because they don't catch every issue. Combining code contributed by various cross-functional dev teams makes it less clear who is responsible for enforcing security. Running multiple applications and application versions in production multiplies the chinks in your applications' armor.

The net result is that the need for security tools such as web application firewalls (WAFs) has never been more acute. These security tools are often integrated with a load-balancing proxy, and deployed at the edge (or front door) of a corporate network to create a secure perimeter.

Security breaches of modern applications and infrastructures have revealed two necessary refinements to this approach:

- **Security at the perimeter is not sufficient.** There is rarely a single, easy-to-secure perimeter, and proxy-based security tools such as WAFs must be deployed closer to the applications they protect.
- **Security is no longer the sole domain of the CISO and SecOps team.** The DevOps team has a critical role in accepting, testing, and deploying security policies as part of its CI/CD pipeline.

NGINX PLUS INGRESS CONTROLLER WITH NGINX APP PROTECT

With NGINX Plus Ingress Controller for Kubernetes release 1.8.0 and later, you can embed the NGINX App Protect WAF in the Ingress Controller:

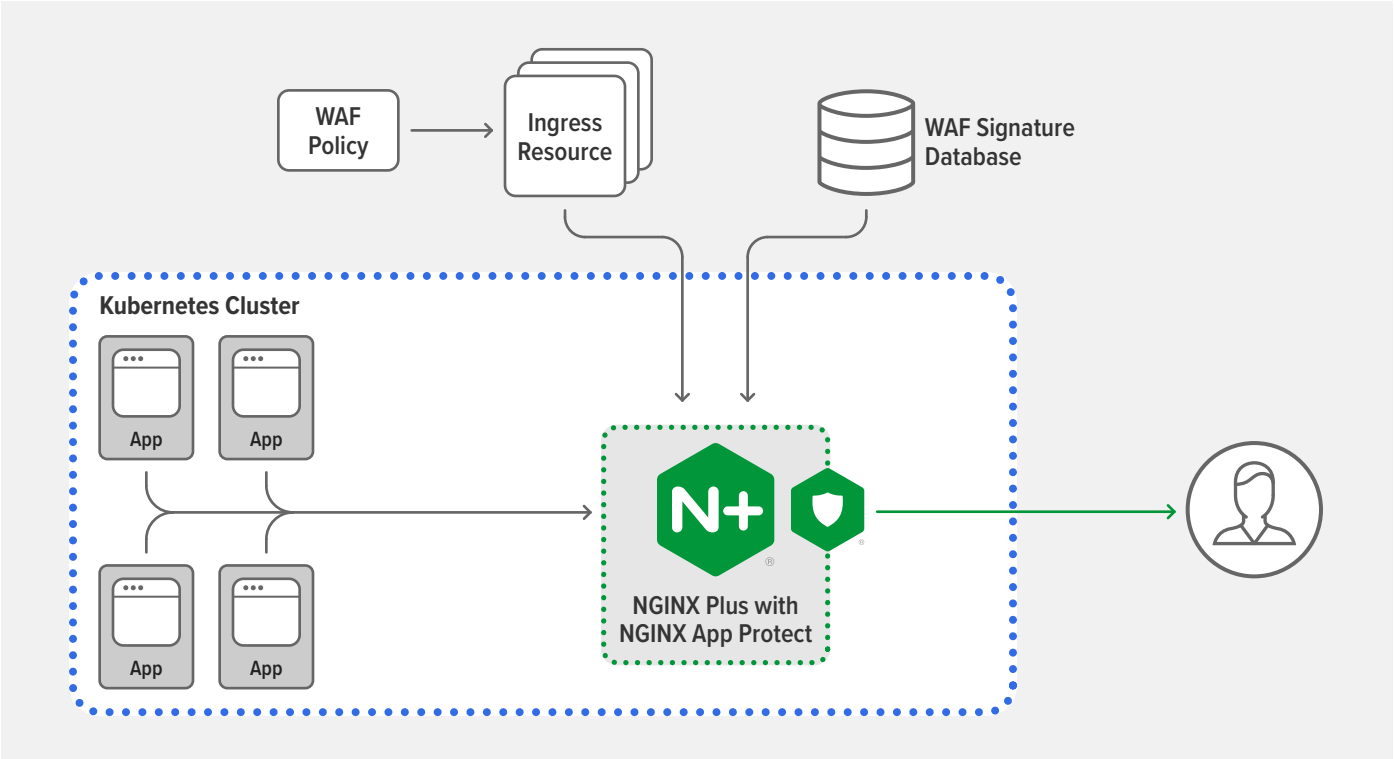


Figure 13: NGINX Plus Ingress Controller with Integrated NGINX App Protect

To build NGINX App Protect into NGINX Plus Ingress Controller, you must have subscriptions for both NGINX Plus and App Protect. A few simple steps are all it takes to build the integrated NGINX Plus Ingress Controller image (Docker container). You can deploy the image manually, with Helm charts, or with the NGINX Ingress Operator on supported platforms including Red Hat OpenShift. You then manage security policies and configuration using the familiar Kubernetes API.

WHY IS INTEGRATING THE WAF INTO NGINX PLUS INGRESS CONTROLLER SO SIGNIFICANT?

Integrating the NGINX App Protect WAF into the NGINX Plus Ingress Controller brings three unique benefits:

- **Securing the application perimeter** – In a well-architected Kubernetes deployment, the Ingress Controller is the only point of entry for data-plane traffic flowing to services running within Kubernetes, making it an ideal location for a security proxy.
- **Consolidating the data plane** – Embedding the WAF within the Ingress Controller eliminates the need for a separate WAF device. This reduces complexity, cost, and the number of points of failure.
- **Consolidating the control plane** – WAF configuration can now be managed with the Kubernetes API, making it significantly easier to automate CI/CD processes. The Ingress Controller configuration complies with Kubernetes role-based access control (RBAC) practices, so you can securely delegate the WAF configuration to a dedicated DevSecOps team.

The configuration objects for App Protect are consistent across both the Ingress Controller (using YAML files) and NGINX Plus (using JSON). A master configuration can easily be translated and deployed to either device, making it even easier to manage WAF configuration as code and deploy it to any application environment.

CONFIGURING APP PROTECT IN NGINX PLUS INGRESS CONTROLLER

You configure App Protect in NGINX Plus Ingress Controller with two new custom resources:

- APPolicy defines the WAF policy for App Protect to apply. The WAF policy is a YAML version of the standalone App Protect JSON-formatted policy.
- APLogConf defines the logging behavior of the App Protect module.

The Ingress Controller image also includes an App Protect signature set, which is embedded at build time.

Once you have deployed suitable APPolicy and APLogConf resources, you reference them from a Kubernetes Ingress resource, using a set of Annotations:

```
apiVersion: extensions/v1beta
kind: Ingress
metadata:
  name: cafe-ingress
  annotations:  a
    kubernetes.io/ingress.class: "nginx"
    appprotect.f5.com/app-protect-policy: "default/dataguard-alarm"
    appprotect.f5.com/app-protect-enable: "True"
    appprotect.f5.com/app-protect-security-log-enable: "True"
    appprotect.f5.com/app-protect-security-log: "default/logconf"
    appprotect.f5.com/app-protect-security-log-destination:
"syslog:server=10.27.2.34:514"
spec:
     ...
```

AppProtect then inspects and potentially blocks all requests handled by the Ingress Controller.

The APPolicy and APLogConf resources can be defined in a different namespace, perhaps one that is owned by the DevSecOps team. This allows for safe and secure separation of concerns, for example in larger enterprises that delegate security policies to a dedicated team.

App Protect policies protect your web applications against many types of threat, including the OWASP Top 10, cross-site scripting (XSS), injections, evasion techniques, information leakage (with Data Guard), and much more. The following sample APPolicy custom resource enables Data Guard violation in blocking mode.

```
apiVersion: appprotect.f5.com/v1beta1
kind: APPolicy
metadata:
  name: dataguard-alarm
spec:
  policy:
    applicationLanguage: utf-8
    blocking-settings:
    violations:
      - alarm: true
        block: true
        name: VIOL_DATA_GUARD
```

(continues)

```
    data-guard:
      creditCardNumbers: true
      enabled: true
      enforcementMode: ignore-urls-in-list
      maskData: true
      usSocialSecurityNumbers: true
    enforcementMode: blocking
    name: dataguard-alarm
    template:
    name: POLICY_TEMPLATE_NGINX_BASE
```

Logging

The logs for App Protect and NGINX Plus Ingress Controller are separate by design, to reflect how security teams usually operate independently of DevOps and application owners. You can send App Protect logs to any syslog destination that is reachable from the Kubernetes Pods, by setting the parameter to the `app-protect-security-log-destination` Annotation to the cluster IP address of the syslog Pod (see the Ingress resource above for an example). Additionally, you can use the APLogConf resource to specify which App Protect logs you care about, and by implication which logs are pushed to the syslog Pod. NGINX Plus Ingress Controller logs are forwarded to the local standard output, as for all Kubernetes containers.

Resource Thresholds

Lastly, NGINX App Protect on NGINX Plus Ingress Controller provides configurable resource protection thresholds for both CPU and memory utilization by the App Protect processes, to keep them from starving other processes. This is particularly important in multi-tenant environments such as Kubernetes which rely on resource sharing and can potentially suffer from the 'noisy neighbor' problem. The following sample ConfigMap sets resource thresholds for App Protect processes.

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: nginx-config
  namespace: nginx-ingress
data:
  app_protect_physical_memory_util_thresholds: "high=100 low=10"
  app_protect_cpu_thresholds: "high=100 low=50"
  app_protect_failure_mode_action: "drop"
```

The `high` threshold sets the percentage utilization at which App Protect enters failure mode, and the low threshold the utilization at which it exits failure mode. For memory utilization these are 100% and 10% respectively, while for CPU they are 100% and 50%. The value `drop` for `app_protect_failure_mode_action` means that App Protect rejects traffic while in failure mode, by closing the connection.

For more detailed information on configuring and troubleshooting NGINX App Protect in NGINX Plus Ingress Controller, see the Ingress Controller documentation. For information about other App Protect use cases, see the NGINX App Protect documentation.

FUTURE INTEGRATION

The Ingress resource configuration in release 1.8.0 uses Annotations to reference App Protect policies, which doesn't provide ideally granular control over which requests are inspected and which are not.

In a future release of NGINX Plus Ingress Controller, you can expect to see more detailed, customizable configuration that is integrated with the NGINX Ingress Resources. This will allow for additional control over how WAF policies are applied to requests.

SUMMARY

For modern, containerized applications, it's often safe to assume that all ingress traffic ("north-south") is untrusted, whereas internally generated traffic ("east-west") is well-formed and trustworthy. In this case, the Ingress Controller is an ideal location for a security proxy such as a WAF.

NGINX Plus Ingress Controller with NGINX App Protect is the only Ingress Controller implementation that integrates a fully supported WAF. Embedding the WAF in the Ingress Controller further improves efficiencies by consolidating data-plane devices into one, and by leveraging the Kubernetes API for its configuration.

8. Secure Your Apps with NGINX App Protect

Today's application landscape has changed dramatically. Modern apps are microservices that run in containers, communicate via APIs, and deploy via automated CI/CD pipelines.

DevOps teams need to integrate security controls authorized by the security team across distributed environments without slowing release velocity or performance. NGINX App Protect is a modern app-security solution that works seamlessly in DevOps environments as you deliver apps from code to customer.

Start your free 30-day trial of NGINX App Protect and NGINX Plus today or contact us to discuss your use cases. You can also read the product documentation and learn more about the full set of F5 web app and API protection solutions.

NGINX
Part of F5