WHITE PAPER

THINK APP SECURITY FIRST

ADVANCED APPLICATION THREATS
REQUIRE AN ADVANCED WAF

The threat landscape is dramatically different than it was just 5 years ago. A traditional web application firewall (WAF) was once a very effective solution for mitigating application layer attacks, but now has trouble keeping up with the advanced capabilities and agility of attackers. Signatures often lag behind new exploits. Even when a traditional WAF is capable of mitigating the threat, implementing and managing it properly can be a challenge. Today, new methods are needed to effectively automate the mitigation of fast-evolving threats.

Why are traditional WAFs inadequate?

Traditional WAFs were created to address the problem of web application servers running code that was vulnerable to a myriad of known attacks, especially cross-site scripting (XSS) and SQL injection. WAFs have been deployed over the years to address these common vulnerabilities, but not without issues of false positives and operational complexity. The original open source WAF, ModSecurity, is often the target of bypass attacks or evasion techniques that attempt to defeat the largely passive, filter-based mechanisms it uses to detect malicious requests

Next-gen firewalls (NGFW) claim “application-aware” features and can also stop some injection attacks (XSS, SQLi, and so on). But, NGFW still relies on passive filter detection and doesn’t examine every HTTP request. Instead, it works much like an IPS, sampling requests and examining their first few bytes, not the full request payload. As a result, application layer bypass attacks against NGFW technologies are common. Plus, IP address reputation feeds implemented on NGFW and other firewall technologies have proven ineffective against botnets and other automated threats.

WAF technology has improved over the years, but it’s still largely based on those passive, filter-based methods used to detect malicious payloads and check for protocol compliance in web requests. In addition, the operational complexity of managing WAF policies has caused many organizations to leave some applications unprotected. In many high-profile breaches, a known application vulnerability was exploited because the targeted organization couldn’t patch the application server or deploy WAF policy quickly enough.

Compounding these challenges, advances in automation technologies and the easy availability of botnets-for-hire have made detecting threats much harder. Automation technologies like headless browsers make it difficult to differentiate a human user from a bot, even with the use of CAPTCHA challenges. Botnets are built on the inexhaustible supply of easily compromised IoT devices, cable modems, and browsers, rendering the source IP address almost useless for detection and mitigation of botnet attacks.

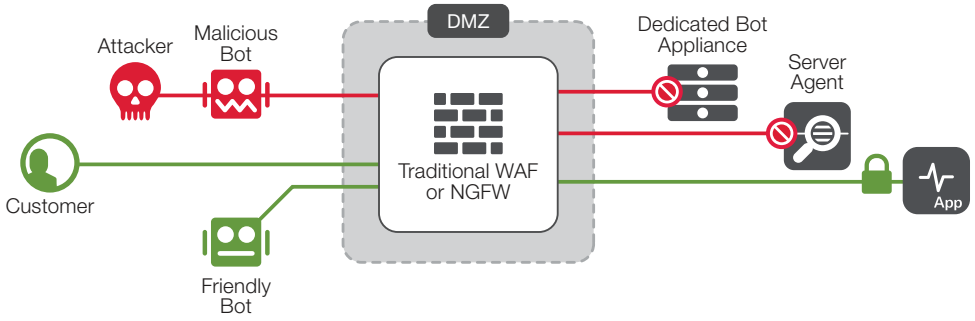

Figure 1: Automated threats bypass Traditional WAF and NGFW.

Automated threats drive today's attacks.

The source of most attacks, regardless of type, is automated. DDoS attacks, data breaches, vulnerability scans, credential stuffing, brute force, resource hoarding, and other attack types are almost all automated. Attackers use automation to launch large-scale attacks and probe for vulnerabilities despite often having less financial and human capital than the organizations they target. In many cases, these automated attacks contain no malicious payload and are crafted to bypass defenses by mimicking legitimate user traffic.

Application layer (or layer 7) DDoS attacks have become a more common attack vector because they can target a resource-intensive URL with legitimate requests and simply overwhelm the application infrastructure. Similarly, credential stuffing (the automated use of compromised usernames and passwords) and brute force attacks designed to bypass login authentication are crafted by the attacker as legitimate requests. These login attacks are often "low and slow" to avoid detection as a DoS attack.

Malicious automated traffic and bots make up 30-40 percent of traffic on a typical site, but as much as 90 percent or more of the traffic to a targeted asset within that same site. The target might be a login page, as in brute force or credential stuffing, or a heavy URL, as in a layer 7 DoS attack. In a resource hoarding attack, the attacker might target the purchase pages for desirable tickets, sneakers, or other items. Scraping attacks similarly target data that'll be mined for later use. These targeted attacks aren't just difficult to detect, they also consume a disproportionate amount of infrastructure resources.

The tools used to automate these attacks include headless browsers (for example Phantom.js and Selenium), vulnerability scanners (the same ones used by penetration testers), command-line scripts, browser extensions, and even malware-infected machines.

The weakest link? The browser.

Browsers are often the weakest link in application security. Attackers try to compromise the user via common phishing attacks embedded in email messages or social media posts. Clicking on malicious links enables attackers to embed malware on the target machine. That malware can be used to enlist the infected machine in a botnet army, to execute one of the attacks talked about in the last section.

More commonly, the malware takes the form of a Remote Access Trojan (RAT), keylogger, or some other method of data gathering. These methods let the attacker harvest sensitive data such as username and password credentials, contact lists, and other information with potential value on the black market. Protecting the user from credential theft is an especially daunting task when a browser or mobile app is the client. These client types offer limited options for enforcing the security posture of the endpoint device.

Users often don't know that their device has been compromised and still think the Internet service is protecting their sensitive data. While HTTPS encryption provides data protection in transit, it doesn't protect the data when it's entered at the endpoint.

Putting better application security controls in place.

The WAF has to evolve into an active security control, capable of interrogating the client endpoint and strengthening the security posture of the application dynamically. The good news is, F5 Advanced WAF employs countermeasures to detect and stop evolving application-layer threats. At a high level, Advanced WAF integrates behavioral analysis and dynamic code injections as its two main mechanisms available to more completely assess the threat associated with any given client session.

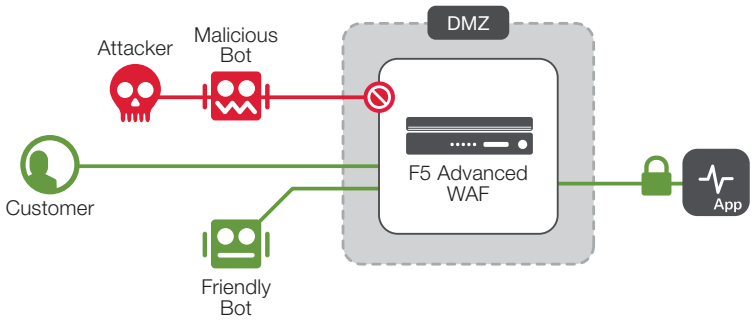

Figure 2: Advanced WAF detects bots without server agents or dedicated appliances.

By profiling a baseline of normal application traffic behavior, anomalous traffic patterns become easier to spot. Just as automation has increased attacker's capabilities, these technologies can differentiate normal from anomalous traffic in ways a human security engineer never could. F5 Advanced WAF uses advanced analytics and machine learning to generate dynamic signatures which block malicious traffic—without administrator intervention.

Using JavaScript injections to assess whether a client is a browser with a human user, Advanced WAF establishes a client fingerprint and enables easier detection of bots and other automated tools. With client fingerprints, the attacker can also be tracked beyond an IP address. The proactive bot defense of Advanced WAF challenges each client session, detecting the nature of the client as well as differentiating friendly bots from malicious ones. The challenges are transparent to the user, reducing or eliminating the impact to the user experience (UX) associated with CAPTCHA challenges.

Code can also be injected to encrypt user keystrokes dynamically, protecting a user from their own malware-infected device. F5's DataSafe[1] technology implements this protection for username and password fields, preventing credential theft. This protection is vital because the target of 86 percent of data breaches is identity or the application, based on research by F5 Labs[2].

Because the API can be notoriously difficult to secure against automated attacks, mobile APIs have increasingly become a target. With only the functionality of the mobile app instead of a browser, mobile app developers are challenged to implement more robust security controls. The new F5 Anti-Bot Mobile SDK[3] enables organizations to quickly integrate advanced security capabilities into their existing mobile apps with just a few mouse clicks.

Here's what you gain by focusing on automated threats.

By shifting the focus to automated threats and employing more active security countermeasures, organizations can realize significant benefits over traditional WAF approaches, including:

Operational improvements

All web applications share browsers and mobile apps as their clients, so it becomes easier to deploy a general WAF policy for most (ideally all) web applications. The web app without a WAF policy or active patching becomes a thing of the past.

Reduced risk

Eliminating automated scanning from the attacker arsenal increases the effort it takes for them to find the latest vulnerability in the application infrastructure. The risk that an unpatched server will be exploited on 0-day is dramatically reduced.

Better use of resources

With a traffic reduction of up to 40 percent, operating costs of the application servers are trimmed, particularly in public cloud environments. Application server performance may also improve under reduced load, resulting in an improved user experience. Reducing the baseline load also means web applications are less susceptible to application-layer DDoS attacks.

The methods outlined here are a blueprint for thinking about web application security in a new way. It represents how F5 sees the threat landscape evolving. By using a more active approach to security via the use of tools like Advanced WAF, security professionals can put in place more effective controls and secure more applications. F5 is pioneering the Advanced WAF space by including comprehensive bot mitigation for web and mobile apps, credential protection in the browser, and automated behavioral analytics using machine learning.

[1] https://www.f5.com/pdf/products/application-level_field_encryption_for_credential_and_data_protection.pdf

[2] https://f5.com/labs/articles/threat-intelligence/cyber-security/lessons-learned-from-a-decade-of-data-breaches

[3] https://www.f5.com/pdf/products/integrate_F5_anti-bot_mobile_SDK_with_any_mobile_app.pdf

DC0418 | WP-SEC-222132511