Maximizing WAF Value

Version 1.6
Released: April 29, 2016

Securosis, L.L.C. 515 E. Carefree Highway Suite #766 Phoenix, AZ 85085 **T** 602-412-3051
info@securosis.com www.securosis.com

Author's Note

The content in this report was developed independently of any sponsors. It is based on material originally posted on the Securosis blog, but has been enhanced, reviewed, and professionally edited.

Special thanks to Chris Pepper for editing and content support.

This report is licensed by Akamai.

www.akamai.com

As the global leader in Content Delivery Network (CDN) services, Akamai makes the Internet fast, reliable and secure for its customers. The company's advanced web performance, mobile performance, cloud security and media delivery solutions are revolutionizing how businesses optimize consumer, enterprise and entertainment experiences for any device, anywhere. To learn how Akamai solutions and its team of Internet experts are helping businesses move faster forward, please visit www.akamai.com or blogs.akamai.com, and follow @Akamai on Twitter.

Copyright

Maximizing WAF Value

Table of Contents

WAF Value Disconnect

Web Application Firewalls (WAF) have been in use for well over a decade, evolving from a point solution primarily blocking SQL injection into a complex application security platform. WAF has continued this evolutionary track in response to new threats, new deployment models, and an increasingly demanding clientele's need to solve more complicated security problems. Progressing from SQL injection to Cross-Site Scripting (XSS), from PCI compliance to DDoS protection, and from Cross-Site Request Forgery (CSRF) to 0-day protection, WAF continued adding capabilities to address emerging use cases. WAF's most recent progression is a direct response to several disruptive innovations, most notably cloud computing and threat analytics.

WAF is back at the top of our research agenda because users continue to struggle with managing WAF platforms as threats evolve and their IT landscape shifts. The value is clear on paper, but too many organizations have trouble realizing that value, so we want to provide some guidance to help folks take WAF off the shelf and starting using it consistently and effectively.

The first challenge has been that many new application attacks require more than simple analysis of individual HTTP requests — it requires multi-request analysis across web application sessions. Detection of modern attack vectors including request forgeries, content scraping, fraud, and other types of misuse — often during a barrage of Bot generated requests to mask activity — requires more information and deeper analysis than older and simpler attacks. Second, as the larger IT industry flails to find security talent to manage WAF, customers struggle to keep existing devices up and running; these customers need to emphasize ease of set-up and management during product selection.

This paper will discuss the continuing need for Web Application Firewall technologies, and address the ongoing struggles to run WAF. We will also focus on decreasing time to value for WAF with updated recommendations for standing up a WAF for the first time, discussing what it takes to get a basic set of policies up and running, and covering the new capabilities and challenges facing customers.

WAF's Continued Popularity

The reason WAF emerged in the first place, and still a key driver of adoption, is that no other product provides comparable protection at the application layer. Many common attacks — such as command injection — which specifically target application stacks go undetected all too often. Intrusion Detection Systems (IDS) and general-purpose network firewalls are poorly suited to protecting the application layer, and still largely ineffective for that use case. To detect application misuse and fraud a security solution must understand the dialogue between application and end

user. WAF was designed for this role, with an understanding of application protocols to identify applications under attack. For most organizations, WAF is still the only way to provide adequate application protection.

For many years the main driver for WAF adoption was compliance, specifically a mandate in Requirement 6 of the Payment Card Industry's Data Security Standard (PCI-DSS) to either use a secure software development life cycle (very hard) or install a WAF in front of the application (much easier). The choice was clear for most organizations. You can basically plug a WAF in and pass that requirement.

Aside from compliance, WAF offers long-term value as well. Users have learned that leveraging a WAF is both faster and cheaper than fixing bug-ridden legacy applications. The focus has shifted from "get compliant fast" to "secure legacy apps for less!"

WAF Limitations

The value of WAF has been reduced by difficulties in deployment and ongoing platform management. A tool cannot provide sustainable value if it cannot be effectively deployed and managed. The last thing organizations need is yet another piece of software sitting on a shelf. Or even worse, an out-of-date WAF providing a false sense of security.

The last thing organizations need is yet another piece of software sitting on a shelf. Or even worse, an out-of-date WAF providing a false sense of security.

Our research highlighted several issues contributing to insecure WAF implementations, allowing penetration testers and attackers to easily evade WAF to target applications directly.

- **Ineffective Policies:** Most firms complain about maintaining WAF policies — the rules that determine what a WAF blocks and what passes through to applications. The main policy complaints involve not keeping pace with new application features and emerging threats. Equally troubling is the lack of information on which policies are effective, which can leave security professionals flying blind. Better metrics and analytics are needed to tell application security professionals what's working, what's not, and how to improve.
- **Breaking Apps:** Security policies can and do sometimes block legitimate traffic. Web application developers are incentivized to push new code as often as possible. Code changes and new functionality often violate existing 'positive' security policies, so unless someone updates the whitelist of approved application requests for significant application updates, a WAF will block legitimate requests. Predictably, this upsets both customers and IT staff who field customer complaints. Firms trying to "ratchet up" security with more restrictive policies may block legitimate requests, or generate too many false positives, with

the resultant flood of alerts overwhelming the SOC and leading to legitimate attacks going unseen and unaddressed.

- **Skills Gap:** As we all know application security is non-trivial. The skills to understand spoofing, fraud, non-repudiation, denial of service attacks, and application misuse within the context of an application are rarely all found in any one individual. But they are all needed to be an effective WAF administrator. Many firms — especially those in retail — complain that "they are not in the business of security" — and would rather outsource WAF management to someone with the necessary skills. Others find their WAF in purgatory after the administrator is offered more money and leaves the organization, creating a huge gap because nobody understands the policies. But outsourcing is no panacea — thanks to the application specificity of WAF rules, even a third-party service provider needs the configuration to be reasonably stable and burned-in before they can accept managerial responsibility. Without in-house talent to set up and manage the WAF, you'll be forced to outsource maintenance to third parties.

- **Cloud Deployments:** Your on-premise applications may be covered by WAF, and you might be reasonably happy with your team's technical proficiency, but as your company moves applications into public cloud infrastructure (and they *are* moving — the question is simply whether you know about it or not), you may find yourself unable to migrate your WAF and its policies to this new and fundamentally different application architecture. Your WAF vendor might not provide a suitable substitute for the appliance you run on-premise, or they might not offer the APIs you need to orchestrate WAF deployment in a dynamic cloud environment, where applications and servers come and go much faster than ever before.

Going It Alone

If all those problems sound like a nightmare scenario, don't worry about it. The *real* nightmare scenario is actually not using some type of WAF or filter to protect your applications, leaving them wide open to attack. Sure, a few of you have been fortunate enough to build your applications from scratch with security in mind, which is awesome. But unicorns are rare — the rest of us are relegated to fixing vulnerabilities in existing applications… you know, the ones designed and coded without any security involvement.

> It is often simply not economically feasible to fix the application, so some other approach must be used — which is where WAF comes in.

In most software stacks the majority of defects are discovered in the underlying platform, third-party software, and in-house applications, rather than in the (much smaller) corpus of in-house code. That's just how software works now. Of course attackers know this, so they probe applications for new vulnerabilities. Legacy platforms take years to meet modern security requirements, if they ever get there. And the cost of these efforts is inevitably many times greater than the

original investment. It is often simply not economically feasible to fix the application, so some other approach must be used — which is where WAF comes in.

Our research shows that WAF failures result far more often from *operational* failure than from fundamental product flaws. Make no mistake — WAF is not a silver bullet — but a correctly deployed WAF makes it much harder to successfully attack an application, and for attackers to avoid detection. The effectiveness of WAF is directly related to the quality of people and processes maintaining them. The most serious problems with WAF are with management and operational processes, rather than the technology. We need a pragmatic process to manage Web Application Firewalls to overcome the issues.

Deploying the WAF

Now we will dig into the myriad ways to deploy a Web Application Firewall (WAF), including where to position it, and the pros & cons of on-premise devices versus WAF services. A key part of deployment is training the WAF for specific applications and setting up the initial ruleset. We will also highlight effective practices for moving from *visibility* (alerting) to *control* (blocking). Finally we will present a Quick Wins scenario because every security technology needs an early 'win' to prove its value.

Deployment Models

The first major challenge for anyone using a WAF is getting it set up and effectively protecting applications. Start with deciding where you want the WAF to work: on-premise, cloud-hosted, or a combination. On-premise means installing multiple appliances or virtual instances to balance incoming traffic, and ensuring they don't degrade the user experience. With cloud services you have the option of scaling up or down with traffic as needed.

Next you need to determine how you want your WAF to work. You can choose either inline or out-of-band. Inline entails installing the WAF "in front of" a web app so all traffic to and from the app runs through it. This blocks attacks directly as they come in, and in some cases before content is returned to users. Both on-premise WAF devices and cloud WAF services provide this option. Alternatively some vendors offer an out-of-band option to assess application traffic via a network tap or spanning port. They use indirect methods such as TCP resets and network device integration to shut down attack sessions. This approach has no side effects on application operation because traffic still flows directly to applications.

When possible we recommend inline deployment because it gives you flexibility to enforce as many or as few blocking rules as you want while allowing legitimate traffic.

Obviously there are both advantages and disadvantages to having a WAF inline, and we don't judge folks who opt for out-of-band rather than risking the application impact of inline deployment. But out-of-band enforcement is easier to evade via tactics like command injection, SQL injection, and Cross-Site Scripting (XSS) attacks which don't require responses from the application. Additionally attacks are more likely make it through to reach applications because there will be a latency between

when the enforcement action is triggered and the session is interrupted. It's not always a clear-cut choice, but balancing risks is why they pay you the big bucks, right?

When possible we recommend inline deployment because it gives you flexibility to enforce as many or as few blocking rules as you want while allowing legitimate traffic. In other words, just because you can block doesn't mean you should block, and with an inline deployment you can decide exactly how aggressive you want to be in terms of blocking attacks. The blocking limitations and relatively easy evasion of out-of-band WAF deployment make it hard for us to support.

Rule Creation

Once the WAF is deployed you need to figure out what rules to run on it. Rules encapsulate what you choose to block and what you let pass through to applications. The creation and maintenance of these rules will occupy the vast majority of your time operating the WAF, so this paper will spend quite a bit of time on them. The first step in rule creation is understanding how rules are built and employed. The two major categories are negative and positive security rules: the former are geared toward blocking known attacks, while the latter list acceptable actions for each application.

Negative Security

"Negative Security" policies essentially block known attacks by detecting patterns of known malicious behavior using *signatures*. Things like content scraping, SQL injection attacks, XML attacks, cross-site request forgeries, suspected botnets, connections from Tor nodes, and even blog spam, are universal application attacks that affect all sites and can be detected via profiles. Most negative policies to block these standard attacks come "out of the box" from vendors' internal teams, who research and develop signatures for customers.

The downside of this method is its fragility: if the signature fails to match any unrecognized variation, it will bypass the WAF. If you think this sounds unpleasantly like traditional endpoint AV, you're right. Signatures are only suitable when you can reliably and deterministically describe attacks, and don't expect signatures to be immediately bypassed by simple evasion. This is where the quality of the provided rules, as well as agility to discover and respond to evolving threats helps you differentiate one vendor from another.

WAF usually provides myriad other detection options to compensate for the limitations of static signatures: heuristics, reputation scoring, detection of evasion techniques, and proprietary methods for qualitatively detecting attacks. Each method has its own strengths and weaknesses, and use cases for which it is better or worse suited. These techniques can be combined to provide a risk score for each incoming request, driving flexible blocking options based on the severity of the attack and your confidence level. This is similar to the "spam cocktail" approach used by email security gateways for years. But the devil is in the details: there are thousands of attack variations, and figuring out how to apply policies to detect and stop attacks remains difficult.

Finally there are rules you'll need specifically to protect your web applications from a class of attacks designed to find flaws in the way application developers code, targeting gaps in how they enforce process and/or transaction state. These include rules to detect fraud, business logic attacks,

content scraping, and data leakage. Example attacks include issuing order and cancellation requests in rapid succession to confuse the web server or database into revealing or altering shopping cart information, replaying legitimate transactions, and fiddling with the order of events to attack transaction integrity.

These application-specific rules are constructed using the same analytic techniques, but rather than focusing on the structure and use of HTTP and XML grammars, a fraud detection policy examines user behavior as it relates to the type of transaction being performed. These policies require a detailed understanding of how both attacks and your web applications work.

Positive Security

The flip side of this coin is the positive security model, also called 'whitelisting'. Positive security only allows known and authorized web requests, blocking all others. Old-school network security professionals use the term *default deny* — this is the web application analogue. It works by observing and cataloging legitimate application traffic, establishing 'good' requests as a baseline for acceptable usage, and blocking everything else. You need to ensure you do not include any attacks in your 'clean' baseline, and then set up policies to block anything not on your list of valid behaviors.

Positive security only allows known and authorized web requests, blocking all others. Old-school network security professionals use the term default deny — this is the web application analogue.

The good news is that this approach is very effective at catching malicious requests you have never seen before (0-day attacks), even without explicit code signatures for each attack. You understand the folly of trying to maintain a ruleset which covers every possible attack. This is also an excellent way to pare down the universe of all threats described above into a smaller and more manageable subset of attacks to include in a blacklist.

The bad news is that applications are dynamic and change regularly, so unless you update your whitelist with every application update, your WAF will effectively disable new application features or crash applications. But for those willing and able to do the work positive security is a big win. Understand that this approach is becoming more complicated as continuous deployment, DevOps, and code trickery such as "feature tagging" all ratchet up the cadence of WAF rule updates. We believe the best bet is for organizations to move testing WAF rules *inside* their development pipelines to ensure the WAF doesn't break new functionality.

In general, the negative security model should reflect the current state of attacks as we know them, and positive models should reflect the current state of your applications. Both have limitations, with negative security always playing catch up with new attacks, and positive constantly trying to keep pace with your application development. The good news is these models complement each other and work well together.

Establishing Rules

Once the WAF is installed it is time to put basic rules in place. You will start with the WAF in monitor-only mode (also known as alert mode) until your rules are established and vetted. This involves three steps:

1. **Detect attacks:** First turn on any built-in (negative) rules to detect known bad behavior. They should be part of the vendor's basic bundle.
2. **Learning mode:** Next the WAF automatically learns web traffic to help build policies, saving you time and effort. Most WAF platforms include this as basic functionality.
3. **Tuning:** You will need to operate in this mode for days to weeks, depending on applications and traffic levels, to give the WAF enough time to learn and generate a decent clean baseline.

Let's dig into learning mode, which starts with discovery. By looking at traffic logs to see what the pre-packaged rules *would have blocked* if they were enabled, you learn what your applications actually do. This provides a proverbial smorgasbord of application activities, from which you identify what needs to be secured and which positive rules are appropriate to enable. This initial discovery process is essential to ensure your initial ruleset covers what each application *really* does — not just what you *think* it does.

By looking at traffic logs to see what the pre-packaged rules would have blocked if they were enabled, you learn what your applications actually do.

After the initial learning process you are ready for your first round of tuning, to get rid of some false positives and (if you tested using actual attacks) false negatives. You will need to tweak WAF rules, and possibly add new ones, to ensure your security policies comprehensively protect your applications. Once you understand your application functions and have a good idea of what attacks to expect, you need to determine how to counter them. For example if your application has a known defect which provides a security vulnerability you cannot address in a timely fashion through code changes, WAF provides several options:

- **Block the request:** You can remove the offending function from your whitelist; if your deployment supports blocking this will disable the threat. The downside is that this may also break part of the application.
- **Create a signature:** You can write a specific signature to detect attacks on that defect so you can detect attempts to exploit it. This will stop known attacks, but effective security requires you to account for *all* possible variations and evasion techniques, which may be impossible.

- **Use heuristics:** You can use one or more heuristics as clues to abnormal application use or attacks on a vulnerability. Heuristics use malformed requests, unusual customer geolocations, customer IP reputation, knowledge of weak or insecure application areas, requests for sensitive data, and various other indicators to recognize attacks. As our blog commenters pointed out, the indicators change over time, and the quality of the offerings from different vendors varies widely. You'll need to evaluate effectiveness of heuristic based rules on your own traffic in order to differentiate the quality of different vendor's offerings.

With your traffic baseline you can enable whitelisting to provide positive security. Once your rules have been tuned and whitelisting enabled, and you are confident in your results, it is time to enable blocking. You will need to be present and alert when you do this, because you are certain to miss something, so expect another small round of tuning here.

Quick Wins

Deploying a WAF for the first time is very difficult — more art than science. It's also high-profile because the affected *applications* are usually high-profile; so you will face scrutiny from the CISO, developers, and IT team. Improved security is much harder to demonstrate or see than a broken application or bad performance, which are visible to everyone… including customers. We have a few tips to get you up and running quickly and safely so you can demonstrate positive momentum to your boss.

> WAF requires considerable specialization to operate effectively. We recommend leaning on your WAF vendor for services early in the process, especially during Proof of Concept (PoC) testing before purchase.

- **Start with learning mode:** As mentioned above, learning mode can dramatically accelerate building your initial ruleset. It used to be very crude but over the years this capability has matured. In some cases we have seen useful whitelists created in under 24 hours.
- **Leverage threat intel:** It's a big world out there, and odds are the attack you saw last week hit someone else before that. Global threat intelligence, threat feeds, IP reputation services, and the like can all help you identify attacks — even when you haven't seen them before. And in many cases threat intelligence can be automatically integrated into your existing ruleset to improve protection with minimal effort. We have seen great success with threat intel because it's easy to employ and quickly blocks basic DDoS, bots, and malware.
- **Feed your other security systems:** WAF outputs event data in several formats, including `syslog` — which can be ingested by just about any SIEM, log management tool, or analytics repository. These feeds provide another source of security data to supplement your security monitoring efforts, and help compliance teams substantiate the controls in place to meet compliance mandates.

- **Reporting:** There are several other organizations in your company that are interested in security and event data from the WAF. Risk management, compliance, development teams — as examples — all benefit from understanding what the WAF protects and what sort of threats are being seen. Getting some basic reports sent to these groups helps get their cooperation and support for WAF functions. While most WAFs have some built in reports and reporting capabilities, you will need to tune and adjust according to your needs, and tweak the output for these groups as well. For example, risk teams often want summary information while development teams are more focused on areas where the underlying application was vulnerable. Data generated by the WAF can be exported into other reporting tools as well.

- **Use vendor services where appropriate:** Given the industry-wide lack of sufficient security talent, most firms have trouble finding people to manage their WAF. Most WAF admins, like other security folks, have other responsibilities, but WAF requires considerable specialization to operate effectively. We recommend leaning on your WAF vendor for services early in the process, especially during Proof of Concept (PoC) testing before purchase. They are far more familiar with their products than you are, and can help steer you past common pitfalls. Some even offer monitoring services to watch your WAF in action, especially those which offer cloud-based WAF services. They essentially help you tune your protection by detecting false positives and negatives. You should try to bundle in additional services when negotiating fees or purchasing, as expert help can go a long way toward getting your WAF up and useful quickly.

Managing the WAF

Deploying a WAF requires knowledge of both application security and your specific application(s). WAF management requires an ongoing effort to keep the WAF current with emerging attacks and frequent application changes. Your organization likely adds new applications and changes network architectures at least a couple times a year, but we see more and more organizations embracing continuous deployment. Their application functions and usage change constantly. You will need to adjust your defenses regularly to keep pace.

Test & Tune

The deployment process focuses on putting rules in place to protect applications. Managing a WAF involves monitoring it to figure out how well your rules are actually working, which requires spending a bunch of time examining logs to learn what works and what doesn't.

Tuning policies for both protection and performance is not easy. As we mentioned, you need someone who understands the rule 'grammars' and how web protocols work. That person must also understand your applications, the types of data your customers should have access to within them, what constitutes bad behavior/application misuse, and the risks web applications pose to your business. An application security professional needs to balance security, applications, and business skills, because the WAF policies they write and manage touch all three disciplines.

Tuning involves a lot of trial and error, figuring out which changes have unintended consequences like adding attack surface or breaking application functionality, and which are useful for protecting applications from emerging attacks.

Tuning involves a lot of trial and error, figuring out which changes have unintended consequences like adding attack surface or breaking application functionality, and which are useful for protecting applications from emerging attacks. You need dual tuning efforts, one for positive rules which must be updated when new application functionality is introduced, and another for negative rules which protect applications against new attacks uncovered by industry research.

By the time a WAF is deployed customers should be comfortable creating whitelists for applications, having gained a decent handle on application functionality and leveraging automated WAF learning capabilities. It's fairly easy to observe these policies in monitor-only mode, but there is inevitably a bit of nail-biting as new capabilities are rolled out. You will be waiting for users to exercise a function before you know if things *really* work, after which reviewing positive rules gets considerably easier.

Tuning and keeping negative security policies current still rely heavily on assistance from WAF vendors and third parties. Enterprises don't have research groups studying emerging attack vectors every day. In fact, neither do some of the WAF vendors. This is an important area to discuss with your vendor as the more you can leverage their knowledge and expertise, the better off you will be. These knowledge gaps around attacker techniques and cutting-edge attack techniques create challenges when writing specific blacklist policies. So you will depend on your vendor as long as you use WAF, which is why we stress finding a vendor who acts as a partner, and building support into your contract.

As difficult as WAF management is, there is hope on the horizon, as firms embrace continuous deployment and DevOps, and accept daily updates and reconfiguration. Their security teams have no choice but to build and test WAF policies as part of their delivery processes. WAF policies *must* be generated in tandem with new application features, which requires security and development teams to work shoulder-to-shoulder, integrating security into release management. New application code goes through several layers of functional testing, and WAF rules get tested as code goes into a production environment, but before exposure to the general public.

There is hope on the horizon, as firms embrace continuous deployment and DevOps, and accept daily updates and reconfiguration. Their security teams have no choice but to build and test WAF policies as part of their delivery processes.

This integrated release process is called Blue-Green deployment testing. In this model both current (Blue) and new (Green) application code run, on their own servers, in parallel in a fully functional production environment, ensuring applications run as intended in their 'real' environment. The new code is gated at the perimeter firewall or routers, limiting access to in-house testers. This way in-house security and application teams can verify that both the application and WAF rules function effectively and efficiently. If either fails the Green deployment is rolled back and Blue continues on. If Green works it becomes the new public production copy and Blue is retired. It's early days for DevOps and a lot of operational disciplines need to refined to truly keep pace with DevOps. Yet this approach enables daily WAF rule tuning with immediate feedback on iterative changes. And more importantly it avoids surprises when updated code goes into production behind the WAF.

WAF management is an ongoing process — especially in light of the dynamic attack space which blacklists address, false-positive alerts which require tuning your ruleset, and the application changes driving your whitelist. Your WAF management process needs to continually learn and catalog user and application behaviors, collecting metrics along the way. Which metrics are meaningful, and which activities you need to monitor closely, differ between customers. The only certainty is that you cannot measure success without logs and performance metrics. Reviewing what has happened over time, and integrating that knowledge into your policies, is key to success.

Machine Learning

The term 'Machine Learning' has generated substantial confusion, so let's first discuss what it means in a security context. In its simplest form, machine learning looks at application usage metrics to predict bad behavior by specific clients. These algorithms examine data including stateful user sessions, user behavior, application attack heuristics, function misuse, and (high) error rates. Additional data sources include geolocation, IP addresses, and known attacker device fingerprints (IoC, or Indicators of Compromise). The goal is to detect subtler application misuse, and to catch attacks quickly and accurately. Think of it as a form of 0-day detection. You want to spot behavior that looks shady, even if you haven't seen that kind of badness before.

Machine learning is a useful technique. Detecting attacks as they occur is our ideal, and automation is critical because you cannot manually review all application activity to figure out what's an attack. So you need some level of automation to both scale scarce resources and fit better into new continuous deployment models.

But it is still early days for this technology — this class of protection has a way to go in terms of maturity and effectiveness. We see varied success: some types of attacks are spotted, but false positive rates can be high. And with some vendors "machine learning" is applied to their 'learning mode' or even the automation of old manual functionality. You need to vet when it's really "machine learning" or just placing lipstick on the pig.

These all-too-predictable marketing shenanigans make it very difficult for customers to compare apples to apples when selecting a WAF, and it is doubly difficult for people without deep background in security and application development to know which capabilities will be useful. For a real comparison you need vendors to show you how these features make WAF policies easier and better in practice. If these capabilities help tune your WAF policies without a lot of work on your part, you're getting huge value. If it's just another way to look at data, and you still need to figure out how to use that data to tune your rules, it's not worth it.

Security Analytics and Threat Intelligence

We have performed a lot of research on the impact of threat intelligence on security operations. You can just Google "Securosis threat intelligence" for enough reading to keep busy for weeks. We believe in TI because it enables you to benefit from the misfortune of others and learn from attacks being used against other organizations.

For WAF, TI services monitor "threat actors" across the globe, collecting events from thousands of organizations. These events are fed into large data warehouses and mined for application misuse patterns.

For WAF, TI services monitor "threat actors" across the globe, collecting events from thousands of organizations. These events are fed into large data warehouses and mined for application misuse patterns. This data also enables vendors to determine which 'users' are regularly scraping content or

sending malicious requests to identify devices, domains, and IP addresses involved in attacks. Additionally, security research teams monitor adversary activity and how threat actors are evolving Tactics, Techniques, and Procedures (TTPs). Patterns and TTPs are fairly consistent for each threat actor, so you can look for likely threat actors' patterns in your logs.

TI offers two key products that help maximize the value of your WAF. The first is the source IP addresses of people and bots involved in attacking web sites — perhaps even those currently attacking *your* site — which you can incorporate into a blacklist to block. These lists change continually as new devices are compromised and altered to probe and attack web sites, so the feed needs to be directly integrated into your WAF for dynamic blocking.

The second offering identifies exploits and malicious requests currently being seen on other sites, with a feed either providing attack details or rules from your WAF vendor to detect and block sets of attacks. These services help evolve your rules without trying (and inevitably failing) to surveil the entire Internet, or needing to employ people who understand how the attacks work and how to block them.

It's not exactly a crystal ball, but TI gives you a heads-up about attacks which may be coming your way.

Putting It All Together

Once your WAF is deployed you need a strong operational process to keep it current. This requires a consistent and reliable testing process to keep pace with application changes. You also can leverage more automated techniques like machine learning (internal analysis of your own applications), and threat intelligence (external analysis of attacks on other organizations), to scale your processes and increase accuracy.

We won't pretend WAF is a "set it and forget it" technology. Work is needed before, during, and after deployment. You need a strong underlying process to make sure you get a quick win to maximize short-term value, and then derive long-term sustainable benefit from your investment.

Summary

Web Application Firewalls (WAF) continue to be an underused yet important security control to protect applications from attacks. A WAF is usually the easiest way to both address compliance requirements for application security and secure legacy applications, which are often infeasible or impossible to update. But WAF technology can be difficult to deploy and even harder to manage due to the lack of staff with sufficient security and application skills. Compounding the challenge is the increasing rate of application change, especially in a DevOps-centric world where new technology can be deployed multiple times a day.

Maximizing the value of WAF involves getting a quick win by deploying negative policies which look for common application attacks. Leveraging threat intelligence from the WAF vendor enables you to get the device up and running quickly, blocking attacks. The second stage of deployment involves adding positive policies, which allow only authorized application traffic by having the WAF 'learn' how the application acts, and then tuning policies to ensure there is no degradation of application operations.

Managing WAF requires a process that integrates rules to stop new application attacks, as well as updating rules to factor in new application changes. To make this all work you must gather data and perform analysis to determine which rules are useful and which are not, and then optimize rules for performance.

Application security is not *optional* — either for compliance or protection of critical corporate data. WAF is integral to application security, so organizations need to figure out how to integrate Web Application Firewalls into their suite of security controls. The information in this paper can help organizations deploy WAF more effectively, decrease time to value, and gain sustainable value by ensuring the WAF stays current with changes in both attacks and applications.

If you have any questions on this topic, or want to discuss your situation specifically, feel free to send us a note at info@securosis.com.

About the Analysts

Mike Rothman, Analyst and President

Mike's bold perspectives and irreverent style are invaluable as companies determine effective strategies to grapple with the dynamic security threatscape. Mike specializes in the sexy aspects of security — such as protecting networks and endpoints, security management, and compliance. Mike is one of the most sought-after speakers and commentators in the security business, and brings a deep background in information security. After 20 years in and around security, he's one of the guys who "knows where the bodies are buried" in the space.

Starting his career as a programmer and networking consultant, Mike joined META Group in 1993 and spearheaded META's initial foray into information security research. Mike left META in 1998 to found SHYM Technology, a pioneer in the PKI software market, and then held executive roles at CipherTrust and TruSecure. After getting fed up with vendor life, Mike started Security Incite in 2006 to provide a voice of reason in an over-hyped yet underwhelming security industry. After taking a short detour as Senior VP, Strategy at elQnetworks to chase shiny objects in security and compliance management, Mike joined Securosis with a rejuvenated cynicism about the state of security and what it takes to survive as a security professional.

Mike published The Pragmatic CSO <http://www.pragmaticcso.com/> in 2007 to introduce technically oriented security professionals to the nuances of what is required to be a senior security professional. He also possesses a very expensive engineering degree in Operations Research and Industrial Engineering from Cornell University. His folks are overjoyed that he uses literally zero percent of his education on a daily basis. He can be reached at mrothman (at) securosis (dot) com.

Adrian Lane, Analyst and CTO

Adrian Lane is a Senior Security Strategist with 25 years of industry experience. He brings over a decade of C-level executive expertise to the Securosis team. Mr. Lane specializes in database architecture and data security. With extensive experience as a member of the vendor community (including positions at Ingres and Oracle), in addition to time as an IT customer in the CIO role, Adrian brings a business-oriented perspective to security implementations. Prior to joining Securosis, Adrian was CTO at database security firm IPLocks, Vice President of Engineering at Touchpoint, and CTO of the secure payment and digital rights management firm Transactor/Brodia. Adrian also blogs for Dark Reading and is a regular contributor to Information Security Magazine. Mr. Lane is a Computer Science graduate of the University of California at Berkeley with post-graduate work in operating systems at Stanford University.

About Securosis

Securosis, LLC is an independent research and analysis firm dedicated to thought leadership, objectivity, and transparency. Our analysts have all held executive level positions and are dedicated to providing high-value, pragmatic advisory services. Our services include:

- **Primary research publishing**: We currently release the vast majority of our research for free through our blog, and archive it in our Research Library. Most of these research documents can be sponsored for distribution on an annual basis. All published materials and presentations meet our strict objectivity requirements and conform to our Totally Transparent Research policy.
- **Research products and strategic advisory services for end users**: Securosis will be introducing a line of research products and inquiry-based subscription services designed to assist end user organizations in accelerating project and program success. Additional advisory projects are also available, including product selection assistance, technology and architecture strategy, education, security management evaluations, and risk assessment.
- **Retainer services for vendors**: Although we will accept briefings from anyone, some vendors opt for a tighter, ongoing relationship. We offer a number of flexible retainer packages. Services available as part of a retainer package include market and product analysis and strategy, technology guidance, product evaluation, and merger and acquisition assessment. Even with paid clients, we maintain our strict objectivity and confidentiality requirements. More information on our retainer services (PDF) is available.
- **External speaking and editorial**: Securosis analysts frequently speak at industry events, give online presentations, and write and speak for a variety of publications and media.
- **Other expert services**: Securosis analysts are available for other services as well, including Strategic Advisory Days, Strategy Consulting engagements, and Investor Services. These tend to be customized to meet a client's particular requirements.

Our clients range from stealth startups to some of the best known technology vendors and end users. Clients include large financial institutions, institutional investors, mid-sized enterprises, and major security vendors.

Additionally, Securosis partners with security testing labs to provide unique product evaluations that combine in-depth technical analysis with high-level product, architecture, and market analysis. For more information about Securosis, visit our website: <http://securosis.com/>.