Barracuda

First the Security Gate, Then the Airplane

Whitepaper

What needs to be heeded when checking web applications?

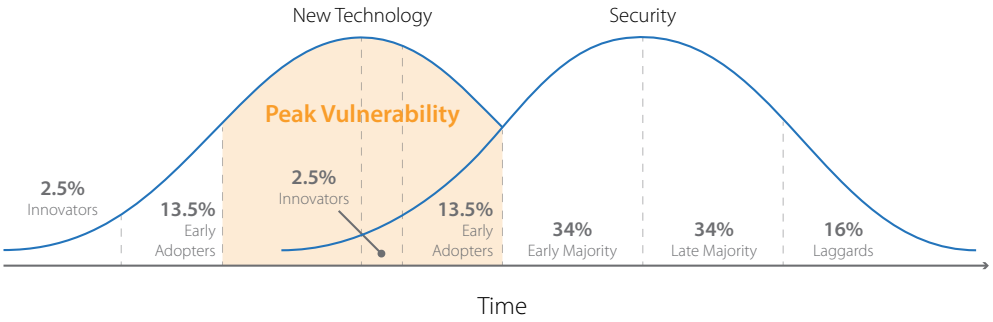
Release 1

Anyone developing a new software program will usually have an idea of the features and functions that the program should master. The subject of security is, however, often an afterthought. But with web applications, the backlash comes quickly because many are accessible for everyone worldwide. They are currently being used by hackers on a grand scale as gateways into corporate networks. Web Application Firewalls (WAFs) make it a lot more difficult to penetrate networks.

In most commercial and non-commercial areas the Internet has developed into an indispensible medium that offers users a huge number of interesting and important applications. Information procurement of any kind, buying services or products but also bank transactions and virtual official errands can be conducted easily and comfortably from the screen. Waiting times are a thing of the past and while we used to have to search laboriously for information, we now have the search engines that deliver the results in a matter of seconds. And so browsers and the web today dominate the majority of daily procedures in both our private as well as working lives. In order to facilitate all of these processes, a broad range of applications is required that are provided more or less publically. Their range extends from simple applications for searching for product information or forms, up to complex systems for auctions, product orders, Internet banking or processing quotations. They even control access to the company's own intranet.

A major reason for these rapid developments is the almost unlimited possibilities to simplify, accelerate and make business processes more productive. Most enterprises and public authorities also see theweb as an opportunity to make enormous cost savings, benefit from additional competitive advantages and open up new business opportunities. This requires a growing number of - and more powerful - applications that provide the Internet user with the required functions as fast and simply as possible.

Developers of such software programs are under enormous cost and time pressure. An increasing number of companies want to use the functionality of these so-called web applications for their business processes and offer their products, services and information as quickly as possible, simply and in a variety of ways. So guidelines for safe programming and release processes are usually not available or they are not heeded. In the end, this results in programming errors because major security aspects are deliberately disregarded or are simply forgotten. The productive use usually follows soon after development, without developers having checked the security status of the web applications sufficiently.



This model (based on Everett M. Rogers adoption curve from "Diffusion of innovations") shows a time lag between the adoption of new technology and the securing of the new technology. Both exhibit the similar Technology Adoption Lifecycle. There is an inflection point when a technology becomes widely enough accepted and therefore economically relevant for hackers resulting in a period of Peak Vulnerability. Bottom line: Security is an afterthought.

Above all, the common practice of adapting tried and tested technologies for developing web applications is dangerous, without having subjected them to prior security and qualification tests. In the belief that the existing network firewall would provide the required protection if possible weaknesses were to become apparent, those responsible unwittingly grant access to systems within the corporate boundaries. And thereby, they disclose sensitive data and make processes vulnerable. But conventional protection systems do not guard against apparently legitimate connections that attackers build up via web applications.

As a result, critical business processes that seemed secure within the corporate perimeter are suddenly freely accessible in the web. Conventional security strategies such as network firewalls or Intrusion Prevention Systems are no longer expedient here. Particularly in association with the web, the security requirements for applications have a different focus and are much higher than for traditional network security. The requirements of service providers who conduct security checks on business-critical systems with penetration tests should then also be respectively higher.

While most companies in the meantime protect their networks to a relatively high standard, the hackers have long since moved on to a different playing field. They now take advantage of security loopholes in web applications. There are several reasons for this: Compared with the network level, you don't need to be highly skilled to use the Internet. This not only makes it easier to use legitimately, but also encourages the malicious misuse of web applications. In addition, the Internet also offers many possibilities for concealment and making action anonymous. As a result, the risk for attackers remains relatively low and so does the inhibition threshold for hackers.

Many web applications that are still active today were developed at a time when awareness for application security in the Internet had not yet been raised. There were hardly any threat scenarios because the attackers' focus was directed at the internal IT structure of the companies. In the first years of web usage in particular, professional software engineering was not necessarily at the top of the agenda. So web applications usually went into productive operation without any clear security standards. Their security standard was based solely on how the individual developers rated this aspect and how high their respective knowledge was.

The problem with more recent web applications: Many offerings demand the integration of additional browser plug-ins and add-ons in order to facilitate the interaction in the first place or to make it dynamic. These include, for example, Ajax and JavaScript. While the browser was originally only a passive tool for viewing web sites, it has now evolved into an autonomous active element and has actually become a kind of operating system for the plug-ins and add-ons. But that makes the browser and its tools vulnerable. The attackers gain access to the browser via infected web applications and as such to further systems and to their owners' or users' sensitive data.

Some assume, that an unsecured web application cannot cause any damage as long as it does not conduct any security-relevant functions or provide any sensitive data. This is completely wrong. The opposite is the case. One single unsecured web application endangers the security of further systems that follow on, such as application or database servers. Equally wrong is the common misconception that the telecom providers' security services would protect the data. Providers are not responsible for a safe use of web applications, regardless of where they are hosted. Suppliers and operators of web applications are the ones who have the big responsibility here towards all those who use their applications, one which they often do not fulfill.

Web applications under fire

The security issues for web applications have not escaped the attackers and they have been exploiting these shortcomings in the IT environments for some time now. There are numerous attack scenarios using which they can obtain access to corporate data and processes or even external systems via web applications. For years now the major types have been:

All injection attacks (such as SQL Injection, Command Injection, LDAP Injection, Script Injection, XPath Injection)

- Cross Site Scripting (XSS)
- Hidden Field Tampering
- Parameter Tampering
- Cookie Poisoning
- Buffer Overflow
- Forceful Browsing
- Unauthorized access to web servers
- Search Engine Poisoning
- Social Engineering

The only more recent trend: The attackers have recently started to combine the methods more often in order to obtain even higher success rates. And here it is no longer just the large corporations who are targeted because they usually guard and conceal their systems better. Instead, an increasing number of smaller companies are now in the crossfire.

One example: Attackers know that a certain commercial software program is widely used for shopping carts in online shops, and that the smaller companies rarely patch the weak points. They launch automated attacks in order to identify - with high efficiency - as many worthwhile targets as possible in the web. In this step they already gather the required data about the underlying software, the operating system or the database from web applications, which give away information freely. The attackers then only have to evaluate this information. As such they have an extensive basis for later targeted attacks.

How to make a web application secure

There are two ways of actually securing the data and processes that are connected to web applications. The first way would be to program each application absolutely error-free under the required application conditions and security aspects according to predefined guidelines. Companies would have to increase the security of older web applications to the required standards later.

However, this intention is generally doomed to failure from the outset, because the later integration of security functions in an existing application is in most cases not only difficult, but also above all, expensive. Another example: a program that had until now not processed its inputs and outputs via centralized interfaces is to be enhanced to allow the data to be checked. It is then not sufficient to just add new functions. The developers must start by precisely analyzing the program and then making deep inroads into its basic structures. This is not only tedious, but also harbors the danger of making mistakes. Another example is programs that do not just use the session attributes for authentification. In this case it is not straightforward to update the session ID after logging in. This makes the application susceptible for Session Fixation.

If existing web applications display weak spots – and the probability is relatively high – then it should be clarified whether it makes business sense to correct them. It should not be forgotten here that other systems are put at risk by the unsecured application. A risk analysis can bring clarity, whether and to what extent the problems must be resolved or if further measures should be taken at the same time. Often however, the program developers are no longer available and training new developers as well as analyzing the web application results in additional costs.

The situation is not much better with web applications that are to be developed from scratch. There is no software program that ever went into productive operation free of errors or without weak spots. The shortcomings are frequently uncovered over time. And by this time correcting the errors is once again time-consuming and expensive. In addition, the application cannot be deactivated during this period if it works as a sales driver or as an important business process. Despite this, the demand for good code programming that sensibly combines effectiveness, functionality and security still has top priority. The safer a web application is written, the lower the improvement work and the less complex the external security measures that have to be adopted.

The second approach in addition to "secure programming" is the general safeguarding of web applications with a special security system from the time it goes into operation. Such security systems are called Web Application Firewalls (WAF) and safeguard the operation of web applications.

A WAF should protect web applications against attacks via the Hypertext Transfer Protocol (HTTP). As such it represents a special case of Application-Level-Firewalls (ALF) or Application-Level-Gateways (ALG). In contrast with classic firewalls and "Intrusion Detection" systems (IDS) a WAF checks the communications at the application level. Normally, the web application to be protected does not have to be changed.

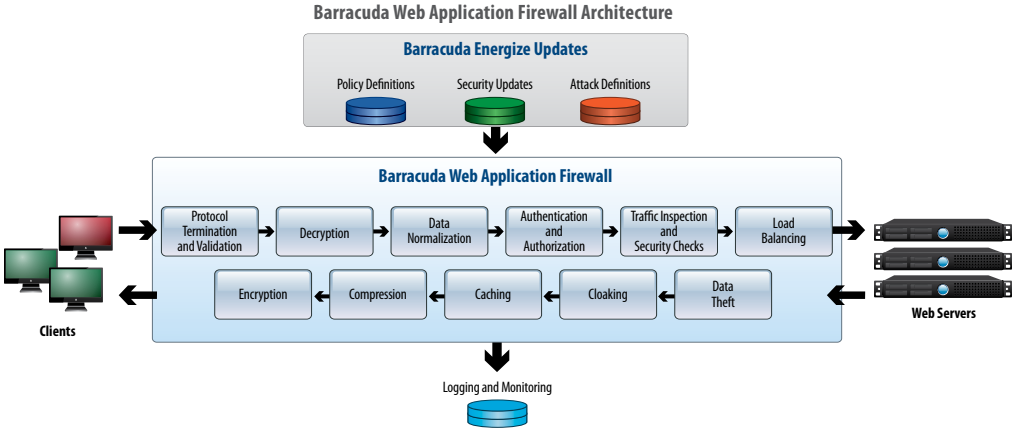
Secure programming and WAFs are not contradictory, but actually complement each other: Analog to flight traffic it is without doubt important that the airplane (the application itself) is well serviced and safe. But even the perfect airplane can never replace the security gate at the airport (the Web Application Firewall), which, as the first security layer, considerably minimizes the risks of attacks on any weak spots.

After introducing a WAF, it is still recommendable to check the security functions, as conducted by Penetration Testers. This might reveal for example that the system can be misused by SQL Injection by entering inverted commas. It would be a costly procedure to correct this error in the web application. If a WAF is also deployed as a protective system, then this can be configured to filter the inverted commas out of the data traffic. This simple example shows at the same time that it is not sufficient to just position a WAF in front of the web application without an analysis. This would lead to misjudging the achieved security status: Filtering out special characters does not always prevent an attack based on the SQL Injection principle. Additionally the system performance would suffer, as the security rules would have to be set as restrictively as possible in order to exclude all possible threats. In this context too, penetration tests make an important contribution to increasing the Web Application Security.

WAF functionality

A major advantage of WAFs is that one single system can close the security loopholes for several web applications. If they are run in redundant mode they can also conduct load balancing functions in order to distribute data traffic better and increase the performance for the web applications. With content caching functions they reduce the load on the backend web servers and via automated compression procedures they reduce the band width requirements of the client browser.

In order to protect the web applications, the WAFs filter the data flow between the browser and the web application. If an entry pattern emerges here that is defined as invalid, then the WAF interrupts the data transfer or reacts in a different way that has been predefined in the configuration diagram. If for example, two parameters have been defined for a monitored entry form, then the WAF can block all requests that



An overview of how a Barracuda Web Application Firewall works

contain three or more parameters. In this way the length and the contents of parameters can also be checked. Many attacks can be prevented or at least made more difficult just by specifying general rules about the parameter quality, such as their maximum length, valid number of characters and permitted value area.

Of course, an integrated XML Firewall should also be the standard these days, because increasingly more web applications are based on XML code. This protects the web server from typical XML attacks such as nested elements or WSDL Poisoning. A fully-developed rule for access numbers with finely adjustable guidelines also eliminates the negative consequences of Denial of Service or Brute Force attacks. However, every file that is uploaded to the web application can represent a danger if it contains a virus, worm, Trojan or similar. A virus scanner integrated into the WAF checks every file before it is sent to the web application.

Several WAFs have the option of monitoring the data sent by the web server to the browser in such a way that they can "learn" their nature. In this way these filters can - to a certain extent - automatically prevent malicious code from reaching the browser, if for example a web application does not conduct sufficient checks of the original data. Learning Mode is a profiling mode that indexes every URL and parameter in

a stream of traffic in order to build a whitelist of acceptable URLs and parameters. However in practice, a whitelist only approach is quite cumbersome, requiring constant re-learning if there are any changes to the application. As a result, whitelist only approaches quickly become out of date due to the constant tuning required to maintain the whitelist profiles. However, the contrary blacklist only approach offers attackers too many loopholes. Consequently the ideal solution should rely on a combination of both whitelisting and blacklisting. This can be made easy-to-use by using templated negative security profile (e.g. for standard usages like Outlook Web Access, Sharepoint or Oracle applications) augmented by a whitelist for high value sub-section like an order entry page.

To prevent a high number of false positives some manufacturers, such as Barracuda Networks, provide an "exception profiler". This flags entries in possible violation of the policies but can still be categorized as legitimate based on an extensive heuristic analysis to the administrator. At the same time the exception profiler makes suggestions for exemption clauses that prevent a similar false positive from being repeated.

Some WAFs provide different operating modes: Bridge Mode (as Bridge Path) or Proxy Mode (as One-Arm Proxy or Full Reverse Proxy). In Bridge Mode the WAF is used as an In-Line Bridge Path and works with the same address for the virtual IP and the backend server. Although this configuration avoids changes to the existing network structure and as such can be integrated very easily and fast to protect an endangered web application. Bridge Mode deployments sacrifice security and application acceleration for network simplicity. Additionally this mode means that all data is passed on to the web application, including potential attacks - even if the security checks have been conducted.

The by far safer operating mode for a WAF is the Full Reverse Proxy configuration, as this is used in line and uses both of the system's physical ports (WAN and LAN). As a proxy WAFs have the capability to protect web applications against attacks such as Session Spoofing or Cross-Site Request Forgery, which is not possible in Bridge Mode. And, only special functions are available here. As a Full Reverse Proxy the Barracuda WAF provides for example Instant SSL that converts http-pages into HTTPS without making changes to the code.

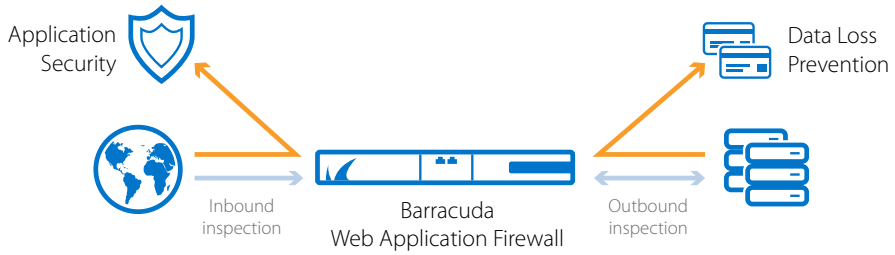
Proxy WAFs also provide a whole range of further security functions. They facilitate the translation of web addresses and as such the overwriting of URLs used by public requests to the web application's hidden URL. This means that the application's actual web address remains cloaked. With Proxy WAFs SSL is much faster and the response times for the web application can also be accelerated. They also facilitate cloaking techniques, Level 7 rules (to avoid Denial of Service attacks) as well as authentication and authorization. Cloaking, the concealment of one's own IT infrastructure, is the best way to evade the previously mentioned scan attacks which attackers use to seek out easy prey. By masking outgoing data, protection can be obtained against data theft and Cookie-Security prevents identity theft. But the Proxy-WAFs must also be configured to correspond with the respective terms. Penetration tests help with the correct configuration.

Demands on Penetration Testers

When penetration testers look for weak spots, they should also take into account the Payment Card Industry Data Security Standard (PCI DSS) 2.0. This defines rules for distributing and storing Primary Account Number (PAN) information. The companies are required to develop secure web applications and maintain them constantly. Further points define formal risk checks and test processes that are intended to uncover high risk weak spots. In order to check whether systems comply with PCI DSS, penetration testers must heed the following requirements:

- Does the system have a Web Application Firewall?
- Does the web traffic occur via a WAF Proxy function?
- Are the web servers shielded against direct access by attackers?
- Is there a simple SSL encryption for the data traffic, even if the application or the server do not support this?
- Are all known and unknown threats blocked?

A further point is the protection against data theft. This involves checking whether the protection mechanism checks the outgoing data traffic for the possible withdrawal of sensitive data and then stops it.



A Web Application Firewall should also protect the outgoing traffic to make data theft more difficult.

Penetration testers can fall back on web scanners to run security checks on web applications. Several WAFs provide extra interfaces to automate tests. The Barracuda Web Application Firewall for example contains APIs for the two common penetration test tools IBM AppScan and Cenzic Hailstorm. In addition there is an open API and XML interface to integrate further testing tools, even those designed by a company itself.

Since, by its very nature, a WAF stands on the frontline, certain test criteria should be applied to it as well. These include in particular the identity and access management. In this context the principle of "least privileges" applies: The users are only awarded those privileges on a "need-to-have" basis for their work or the use of the web application. All other privileges are blocked. A general integration of the WAF in Active Directory, eDirectory or other RADIUS- or LDAP-compatible authentication services makes this work easier.

The user interface is also an especially critical point, because it is the basis for safe WAF configuration. Unintelligible or poorly structured user interfaces lead to incorrect settings, which cancel out the protective functions. If, by contrast, the functions can be recorded intuitively, are clearly displayed, easy to understand and to set, then this in practice makes the greatest contribution to system security. An example can be seen at www.barracudanetworks.com/demos.php. A further plus is a user interface which is identical across several of the manufacturer's products or, even better, a management center like the Barracuda Control Central (BCC)which administrators can use to manage numerous other network and security products with as well as the WAF. The administrators can then rely on the known settings processes for security clusters. This ensures that security configurations for each clusters are consistent across the organization. An extensive penetration test of web applications should therefore take the ergonomics of the WAF interface into account for the evaluation along with the consistency of security deployment across applications and sites.

In summary, any web application, old or new, needs to be secured by a WAF in Full Proxy Mode. Penetration testers should check whether the WAF reliably cloaks system information in order to make attacks on the infrastructure less likely in the first place. It should also check whether it prevents the hacking of the application itself with common or new means, whether it secures all the backend systems the application connects to and it stops leakage of sensitive data when the web application has weaknesses which the WAF cannot level out. If penetration testers are not only looking for a security snap shot, but want to help their customers in creating sustainable security, they should always include the WAF's administration into their assessment.

To learn more about Barracuda's web security solutions, please visit www.barracuda.com/products or call Barracuda for a free 30-day evaluation at 1-408-342-5400 or 1-888-268-4772 (US & Canada).

About Barracuda Networks, Inc.

Protecting users, applications, and data for more than 150,000 organizations worldwide, Barracuda Networks has developed a global reputation as the go-to leader for powerful, easy-to-use, affordable IT solutions. The company's proven customer-centric business model focuses on delivering high-value, subscription-based IT solutions for security and data protection. For additional information, please visit www.barracuda.com.

Barracuda

Barracuda Networks
3175 S. Winchester Boulevard
Campbell, CA 95008
United States
408-342-5400
888-268-4772 (US & Canada)
www.barracuda.com
info@barracuda.com