



Start Secure. Stay Secure.™

Intelligent Engines

The next generation of Web application security

By SPI Labs

Intelligent Engines

Table of Contents

<i>Intelligent Engines – The Next Generation in Web Application Security</i>	3
<i>Why Intelligent Engine technology combined with Static Checks delivers the most accurate results</i>	3
<i>The problem with traditional static check technology</i>	4
<i>Why Intelligent Engines are faster and more accurate</i>	6
<i>Conclusion</i>	8
<i>About SPI Labs</i>	9
<i>Contact Information</i>	10

Intelligent Engines

Intelligent Engines – The Next Generation in Web Application Security

Why Intelligent Engine technology combined with Static Checks delivers the most accurate results

Today's Web application and Web services assessment products boast thousands of static checks for security vulnerabilities like Cross-site scripting and SQL Injection. The Web application assessment software vendors have essentially been in the game of who has the best vulnerability database with the most checks.

At the rate the industry is currently going with the growing number of checks, vulnerability databases will have tens of thousands of static checks in a few years. With that many checks, application scans will simply take too long. The industry has needed new technology to revolutionize the process of finding vulnerabilities.

Intelligent engine technology will drastically change how customers and analysts will evaluate Web application assessment products. Static checks will remain important, but the game will change. It will be the combination of intelligent engines and static checks that will define the leaders in this space.

Initial results from new intelligent engine technology are astounding. The intelligent engine technology when combined with static checks enables

Intelligent Engines

security professionals to complete assessments much faster, virtually eliminate false positives, and discover more true vulnerabilities than ever before. Using traditional application assessment static-check technology, running an automated test for a vulnerability like Cross-site scripting could take up to three hours and yield results with many false positives. Using intelligent engine technology, that time can be reduced to 12 minutes with almost no false positives. Furthermore, the combination of static checks with intelligent engines finds more true vulnerabilities than static checks alone.

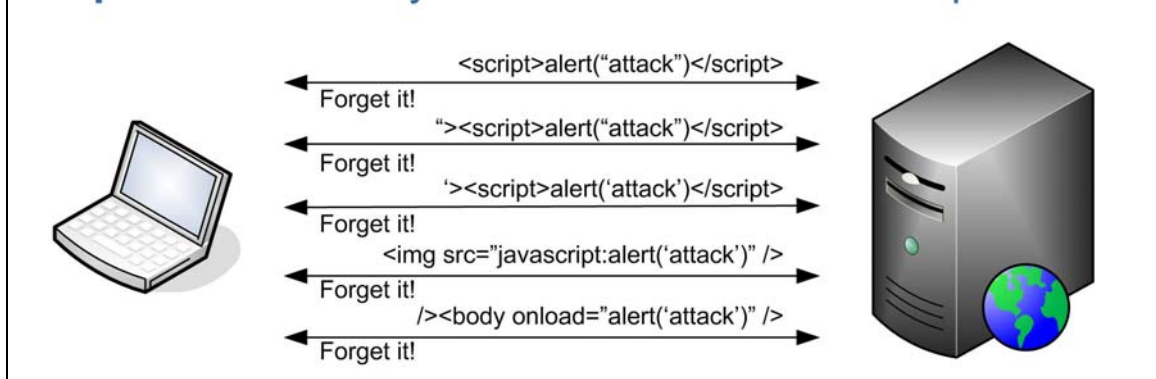
The intelligent engine technology is able to bring such significant advances because of its structured logic-based approach. Intelligent, sophisticated engines analyze the conditions in the application and then use that knowledge in a targeted attack. The static checks approach fires off every possible attack option without taking into account the uniqueness of the application.

The problem with traditional static check technology

For the most part, using a vulnerability database with static checks has been a successful approach. As Web applications and their functionality have grown in scale and complexity, however, there has been a consequent rise in problems with standard Web application scanning methodology. Here's an illustration showing how a traditional Web application scanning vulnerability assessment would be conducted when seeking an instance of Cross-Site scripting. The same attacks are repeatedly submitted against all avenues of

Intelligent Engines

input that were discovered during a “crawl” of the application to see if a dialog box can be opened, indicating that the application is indeed susceptible to Cross-Site scripting.

Old process: Blindly send static attacks and hope for a hit

Even if the Web application filters a potentially malicious character such as “>”, multiple attacks that include that character will still be submitted. Some of the problems that are becoming more evident with this approach follow:

Lack of Intelligence - One problem with the traditional approach is with the general lack of “intelligence” that is applied to it. That is not to say that the individuals creating the assessment methodology are lacking in acumen, but rather an observation of the limitations of the standard “bulk” approach. It’s an “all or nothing” methodology that doesn’t utilize any intelligence or logic other than yes or no and the sheer number of vulnerability signatures being submitted. Does the file exist? Does the application accept a specific

Intelligent Engines

character as input when submitted in a form? Has this particular default installation file been removed? And on and on, ad infinitum.

Assessments take too long - The number of potential vulnerabilities and their variants is constantly growing. Longer lists of static checks, each of which must be submitted against an application, means slower scanning times. It takes a large database of static checks just to ensure the accuracy of a scan.

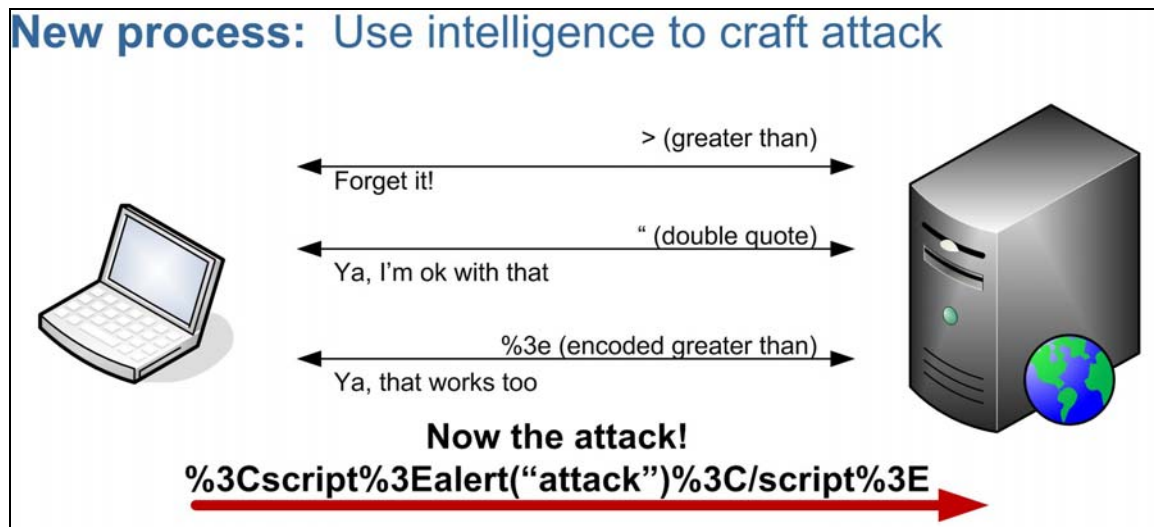
High Number of False Positives - As Web vulnerabilities and technologies change with time it is difficult for checks to stay accurate. Vulnerability signatures are “hard coded” and static, and heavily technology dependent. In essence, they cannot be dynamic or intelligent about what the server is responding with. This can lead to a high number of “false positives”, when an automated assessment tool flags a vulnerability that in actuality does not exist. Each false positive has to be manually verified, a time intensive task.

Why Intelligent Engines are faster and more accurate

How can intelligence be applied to this process without losing accuracy, producing more false positives, or increasing scan times dramatically? By converting from signature based assessments to ones that rely on intelligent engines that can dynamically craft attacks based on how the server reacts to requests. These intelligent engines can dramatically reduce scanning time by potentially doing the work of hundreds of checks in a fraction of the time.

Intelligent Engines

The next image illustrates how an Intelligent Engine would submit attacks seeking an instance of Cross-Site Scripting.



First, the engine determines exactly on the page where the input is being accepted. Where are we (inside a frame, etc.)? Then, it determines what is allowed as input on that page. What's allowed? Only then is the actual attack string built and submitted against the application. For instance, if the application properly filters a ">", then the intelligent engine would convert that character to its HTML entity before it was submitted again. It's a very specific, focused approach. If the HTML standard allows it, then the vulnerability will be found. It's the difference between standing at a locked door with a ring of a hundred keys, or being a locksmith and creating a custom key on the spot. After a response from the server is returned, a Javascript parser is then employed to parse it and verify that an actual

Intelligent Engines

vulnerability exists. This intelligent approach dramatically reduces incidents of false positives by determining the right question to ask, and even more so by then only responding to the proper input.

Conclusion

In every industry, there comes a time when the limitations of the current method of doing things become self-evident and starts to collapse under its own weight. Web application security applications that rely on static libraries of checks are rapidly approaching that point. Functionality has become too complex to rely upon enormous lists of static signatures which are required to find the proverbial needle in the haystack. There is an obvious need for a revolutionary innovation in how Web application security assessments are conducted. By applying the proper intelligence, Web application security assessments can be conducted faster, performed more flexibly, and serendipitously be more accurately done by virtually eliminating false positives. An individual who knows how to use a card catalog can find a specific book in a library much faster than a large group who must pull every volume off the shelf to see if it is the one they want. It's the same with Web application security. A little intelligence goes a long way.

Intelligent Engines

About SPI Labs

SPI Labs is the dedicated application security research and testing team of S.P.I. Dynamics, Inc. (www.spidynamics.com). Composed of some of the industry's top security experts, SPI Labs is specifically focused on researching security vulnerabilities at the Web application layer. The SPI Labs mission is to provide objective research to the security community and give organizations concerned with their security practices a method of detecting, remediating, and preventing attacks upon the Web application layer.

SPI Labs' industry leading security expertise is evidenced via continuous support of a combination of assessment methodologies which are used in tandem to produce the most accurate web application vulnerability assessments available on the market. This direct research is utilized to provide daily updates to SPI Dynamics' suite of security assessment and testing software products. These updates include new intelligent engines capable of dynamically assessing web applications for security vulnerabilities by crafting highly accurate attacks unique to each application and situation, and daily additions to the world's largest database of more than 5,000 application layer vulnerability detection signatures and agents. SPI Labs engineers comply with the standards proposed by the Internet Engineering Task Force (IETF) for responsible security vulnerability disclosure.

Information regarding SPI Labs policies and procedures for disclosure are outlined on the SPI Dynamics Web site at:

<http://www.spidynamics.com/spilabs.html>.

Intelligent Engines

Contact Information

S.P.I. Dynamics
115 Perimeter Center Place
Suite 1100
Atlanta, GA 30346

Telephone: (678) 781-4800
Fax: (678) 781-4850
Email: info@spidynamics.com
Web: www.spidynamics.com