STOP MALWARE BEFORE IT STOPS YOU

WHITE PAPER

Trustwave®
Smart security on demand

CATCH MALWARE IN THE ACT

DISCOVER THE ADVANTAGES OF TRUSTWAVE'S SWG MALWARE ENTRAPMENT ENGINE

The Web world is rapidly changing. A typical Web page is no longer a simple, static hypertext page, but rather a page being generated by diverse technologies. The most common is dynamic Web page technology, such as dynamic frameworks, like PHP, J2EE or ASP.NET on the server side, or Dynamic HTML functionality of the browsers utilizing AJAX or CSS on the client side.

Furthermore, Web pages can use online scripting languages like JavaScript, bringing flexibility to the page behavior without having to make changes to its underlying code. The browser can also utilize AJAX technologies to interact with the server asynchronously without interfering with the behavior of the page, or to take advantage of the features of CSS to control the page layout. The modern Web page is adaptive to diverse users, highly modular, rich and efficient, and acts as an application delivery platform.

The same dynamic technologies that benefit businesses also benefit malicious code authors, allowing them to exploit browser and browser plug-ins in run-time. JavaScript, as the de facto standard for creating interactive Web applications, has become the most common tool used for such attacks.

These massive changes and evolving trends in the Web landscape prompted Trustwave to develop the Malware Entrapment Engine, able to detect and eliminate modern malicious activities, adding significant value to the detection mechanism with minimal latency and minimal impact on end user experience. This new intelligent engine is offered on top of other Trustwave SWG engines.

This paper highlights the unique advantages and features of Trustwave's SWG Malware Entrapment Engine.

TRUSTWAVE MALWARE ENTRAPMENT ENGINE - WHAT IS IT ALL ABOUT?

It is no secret that the internet is full of malware and that millions of end machines are attacked every day by various types. In the past, malware may have been the domain of “script kiddies” and curious minds pushing boundaries, but these days malware is an industry developed to make money. Malware is bought, traded and managed by criminals who find it to be extremely lucrative and are willing to put a lot of resources back into it. Criminals perform market research and target specific demographics that they feel are the most profitable. They will look for financial information, such as credit card numbers and bank account details, as the potential for profit is high.

In such evolving attacks, malicious code authors are using HTML elements to hide JavaScript code elements.

The attacks are built dynamically. Each part of an attack seems completely benign at rest, but when put together with the other parts and executed, can become malicious. Even if the code looks suspicious when being statically scanned, the purpose of the obfuscation remains unknown unless one runs an emulation of the code.

Trustwave’s SWG Malware Entrapment Engine addresses such obfuscated attacks, which have become widely used by current cybercriminals. It does so by conducting DOM[1] analysis on the fly with minimal latency impact.

For example, an attacker can store a shellcode string as a value of a hidden HTML tag or attribute and execute it in runtime while using the HTML DOM.

This specific method allows the code to overcome other security engines that are not DOM-aware and unable to determine whether an embedded script will exploit the browser at runtime.

VULNERABILITY-DRIVEN, BUT EXPLOIT-AWARE

The Malware Entrapment Engine doesn’t just try to check for exploits, which can be delivered in different forms. Rather, it identifies conditions that would result in exploitation in order to detect attacks exploiting certain vulnerabilities.

The engine disregards the specifics of exploitation code but remains aware of the code’s malicious behavior, if it exists. This behavior enables real zero-day protection against unknown vulnerabilities.

[1] DOM: Document Object Model (DOM), an application programming interface for HTML and well-formed XML documents. It defines the logical structure of documents and the way a document is accessed and manipulated.

EVER-CHANGING BATTLE

Trustwave's Malware Entrapment Engine employs internal detection logic, which can be easily updated as part of the ongoing security updates deployment. By utilizing this mechanism, the system is able to rapidly address new threat techniques and protect from both known and unknown exploits.

The Malware Entrapment Engine protects against various vectors of attack, including HTML, JavaScript, VBScript, PDF and Flash. This ability, combined with the other SWG engines, covers the full spectrum of the common attack scenarios. The engine is an extensible and flexible platform, allowing more file types' support in the future to address emerging and new threats.

THE BIG PICTURE

One characteristic of a Web application is that one document may load scripts from various sources. By default, each script loaded into the main document possesses access to all of the data and functionality of the other scripts loaded into that document. This functionality is often used to include advertising sections on a webpage or analytics tracking.

Trustwave's Malware Entrapment Engine inspects the webpage as a whole, including all embedded objects or resources such as JavaScript or CSS files. The engine can detect attacks that leverage these dynamic capabilities and comprehend the real intent of the full document's code.

A simple example:

```
<!DOCTYPE html>
<head>
    <script language="javascript" src="test.js"></script>
</head>
<body>
    <div id="y" x="646f63756d656e742e777269746525323825323248656c6c6f25323125323225323925334 2"/>
    <script>
        var m = document.getElementById('y').getAttribute('x');
        var code = doSomething(m);
        eval(code);
    </script>
</body>
```

The page itself does very little:

1. Loads a JavaScript resource by the name of "test.js"
2. Creates an HTML element of the type DIV
3. Executes a short piece of JavaScript code that reads a property of the DIV tag, sends it to a method called doSomething (which is defined in test.js) and evaluates the response

The test.js file doesn't reveal much either:

```
function doSomething(m) {
    o = "";
    for(var i=0;i<m.length;i+=2){o+=unescape("%"+m.substr(i,2));}
    return unescape(o);
}
```

The code defines a JavaScript method that receives a string, manipulates its characters and returns another string as a result, thus constructing code at runtime that adversely changes the behavior of the document.

Examining each page individually does not help much in the way of understanding the underlying intentions of the code. The main HTML page calls a function by the name of doSomething, but we would not know what it does; whereas the test.js file declares a function by the name of doSomething, but we don't know what the parameter "m" that it receives is, so we do not know what it would return to the user. This kind of behavior could hide anything from an email address to malicious code that will infect the end machine with a banking Trojan.

It is clear from this example that understanding the connection between the test.js resource and the parent HTML page is essential to knowing what code this page will construct and execute on the client machine in runtime.[2]

In order to improve efficiency and have minimal impact on latency, the engine employs an internal prefetching and caching mechanism of the correlated webpage objects or resources.

LIKE A BROWSER

Another webppage feature is its ability to dynamically generate and execute JavaScript code. One of the common options for implementing dynamic execution is using the JavaScript eval() function, which evaluates a string containing JavaScript code as if it was embedded in the document's code. Malicious code authors often utilize this ability to construct code at runtime that adversely changes the behavior of the document. The malicious author can also split the malicious content between a few benign parts, which are put together only in runtime.

The easiest, most profitable way for an attacker to infect machines is using an exploit kit. An exploit kit is a collection of exploits put together into a single package. Like any professional service offered, it has an administration panel, statistics, feature list, periodic updates, etc. The exploit kit enables an attacker who successfully infects a website to inject it with a script or iframe that, when loaded by the browser, will examine the client machine, detect the operation system, browser and plugins installed, and choose an optimal attack to execute on the specific machine. At the end of the process, an executable file of the attacker's choice will be downloaded and executed on a client machine.

Therefore, a completely legitimate website that is vulnerable or uses a vulnerable CMS, or even a plugin that was not updated, can be turned into an attack site when it is infected with an exploit kit.

When this method of operation started, the injected scripts/iframes and their content were easy to spot—it was just a matter of identifying and blocking them. The Application Level Vulnerabilities scanner performed this role very well.

As the popularity of exploit kits grew and security engines began blocking them, obfuscation and evasion became key ingredients. Common exploit kits use different methods of code obfuscation[3] in order to wrap their malware payloads to execute an attack in a stealthy manner.

For example, the following use of string splitting and concatenating will turn what appears to be a string full of "junk" into code that, when evaluated, will actually execute:

```
<script>
    var rand1 = 'd9o99c9u99m9e99n9t999.999w99r9'
    var rand2 = '9i99t99e99(999"h99e9ll999o99"9999)99;'
    eval(rand1.concat(rand2).split("9").join(""));
</script>
```

[2] http://labs.m86security.com/2012/01/web-hijacks-with-ajax/

[3] Code obfuscation in the programming world means making code harder to understand or read.

Statically examining this code shows the use of various string manipulation methods, all of which have completely legitimate uses in benign code. Only when these are put together and used on the particular data will they execute malicious code.

Another example is this malicious code which can be de-obfuscated to reveal a hidden iframe leading to the attack page:

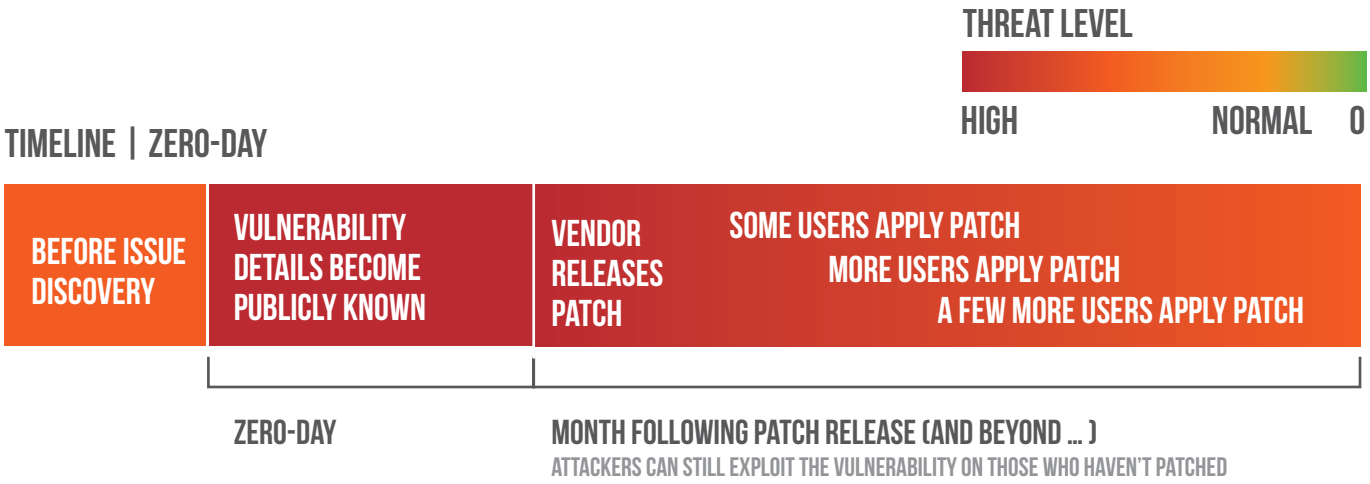
These techniques make such attacks virtually undetectable by common security engines. To overcome the techniques, Trustwave's Malware Entrapment Engine emulates browser behavior.

Exploit kits also tailor their attacks for specific browsers, making use of the differences between the various browsers. The engine takes these aspects into account during the analysis.

TRUE ZERO-DAY PROTECTION: SWG MALWARE ENTRAPMENT ENGINE IN ACTION

What is Zero Day Protection?

Zero day exploits are attacks built by hackers based on vulnerabilities discovered in any piece of software (browsers, plug-ins, applications) that does not have an official patch to mitigate it. Zero day protection is the ability to provide protection against such zero day exploits. This can be achieved if the vendor knows of a particular vulnerability ahead of time and provides signatures for the attempt to take advantage of the vulnerability. This approach does not guarantee full protection. Furthermore, any method that is signature based cannot claim to offer true zero day protection, as there is a window in time needed for the vendor to build the update and distribute it.



Good security works in layers. SWG strives to provide protection in many layers and many stages in the exploitation process. A combination of detecting malicious behavior, as well as specific vulnerability conditions, allows both **true zero day protection** and accurate vulnerability protection.

What follows is a recent event demonstrating the ability of the SWG Malware Entrapment Engine to identify and block attacks based on vulnerabilities and provide true zero-day protection.

TIMELINE	DESCRIPTION
June 12	Microsoft publishes an advisory for the zero-day vulnerability in Internet Explorer: Microsoft XML Core Services 3.0, 4.0, 5.0, and 6.0 accesses uninitialized memory locations, which allows remote attackers to execute arbitrary code or cause a denial of service (memory corruption) via a crafted website. http://technet.microsoft.com/en-us/security/advisory/2719615 **The SWG Malware Entrapment Engine blocks all in-the-wild exploiting of this vulnerability without the need of a security update.**
June 15	A public exploit in the form of a Metasploit module (an infrastructure used by security people to generate exploit pages, a module essentially means it's very easy for someone to start using this attack without really knowing too much about it) becomes available.
June 18	Trustwave issues a security update with detection aimed specifically towards this vulnerability, adding yet another layer of protection for its customers.
June 19	During a staged targeted attack, the malicious code on a webpage may only become active after a certain period of time or for short spells during a day. This ability to hide, combined with the way active malicious code may change, means that it is essential to scan a website each and every time it is accessed from an untrusted link.

Note: Current Trustwave SWG users are protected against this exploit in-the-wild without having to install or update anything on their system due to the intelligent detection of unknown vulnerabilities by SWG generic proactive engines.

SUMMARY

Trustwave SWG technology is able to stop new, dynamic malware that other solutions overlook, as well as close the 60% malware gap missed by other gateways. SWG also reduces the financial costs associated with malware attacks incurred by desktop reimaging and remediation, data loss and compliance breaches. Plus, it helps to maintain productivity by virtually eliminating over-blocking and under-blocking. By introducing the Malware Entrapment Engine, Trustwave is now able to provide its SWG customers with better protection against increasingly sophisticated malicious attacks.

ABOUT TRUSTWAVE SECURE WEB GATEWAY

Trustwave Secure Web Gateway (SWG) is a security appliance located at the enterprise Web gateway designed to protect Web clients from malicious Web threats. Trustwave SWG's multi-layered anti-malware engine comprises intelligent technologies that detect and block next-generation, dynamic malware in real time.

Trustwave SWG is the only solution that protects against next-generation threats in real time. By accurately detecting new and targeted attacks, it virtually eliminates over- and under-blocking and enables users to access the Internet safely and productively. With the most flexible deployment options in the industry, it fits into existing infrastructures easily while providing full security coverage to every user, regardless of connection or location.

LEARN MORE AT TRUSTWAVE.COM

Trustwave®

Smart security on demand

Trustwave is a leading provider of compliance, Web, application, network and data security solutions delivered through the cloud, managed security services, software and appliances. For organizations faced with today's challenging data security and compliance environment, Trustwave provides a unique approach with comprehensive solutions that include its TrustKeeper® portal and other proprietary security solutions. Trustwave helps millions of organizations manage compliance and secure their network infrastructure—ranging from Fortune 500 businesses and large financial institutions to small and medium-sized retailers—manage compliance and secure their network infrastructures, data communications and critical information assets. Trustwave is headquartered in Chicago with offices worldwide. For more information: https://www.trustwave.com.