SOPHOS

Malicious JavaScript Attacks: What Can You Do?

Part I of this series discussed why hackers use malicious JavaScript attacks, and why you should be concerned. In this paper, we'll explore solutions.

As an IT manager, you need to be able to effectively secure all of the websites you manage to avoid attack and the subsequent infection of site visitors. You also need to protect the users within your organization from becoming victims of malware. Organizations can use specific technologies and layered protection strategies to safeguard websites and shield employees. We'll review what to watch for, as well as the security strategies that will help keep you safe.

The Malicious JavaScript Attack Threat

The web is a perfect vehicle for threat delivery. Malware authors use the web to deliver their attacks for three primary reasons: it's easy to reach you; it's easy to infect you; and traditional defenses are failing.

Web attacks are high-volume attacks, and cybercriminals get more money when their malware tricks lure more people. And, this is not just from rogue pages set up specifically to trick site visitors—many legitimate websites are compromised to play a major role in attacks. In fact, our threat experts analyze malicious URLs each day, and a vast majority of these are legitimate websites that were hacked.

Why is JavaScript used? JavaScript is a programming language that underpins today's web; browsing the web without JavaScript support is no longer a realistic option. Attackers take advantage of this and inject inline JavaScript to hide redirects more effectively than with simpler attack methods.

Cybercriminals can buy black market exploit kits that let even non-programmers create sophisticated, malicious websites. Once a victim is silently redirected to a malicious site, their data can be stolen, or they may fall prey to a fake antivirus software scam, which gains revenue for the attackers.

Malicious JavaScript can impact your business in a variety of ways. Your websites can be compromised, and as a result, inadvertently expose anyone who browses your site to malicious code. Second, your users can become malware victims if the machines you manage get infected via a drive-by download, SEO poisoning, or other attacks explained in our first paper. Both of these possibilities have very real impact, in terms of costs, productivity, corporate reputation and data theft.

As an IT manager, you need to be vigilant and defend against these malicious JavaScript threats, but how do you do that?

Securing Your Websites

Securing the websites you manage is the most effective way to avoid your site being involved in a malicious JavaScript attack. As an IT manager, you should not only protect web servers, but also the website's foundation—its operating system (OS)—and ensure secure application coding practices are being used. You also need to remember to update any software components that run on a web server. We recommend this type of layered approach to get the best malicious JavaScript protection.

You need to work with your website administrators, programmers and designers and make sure secure application coding practices are in order and applied. You also need to continually check that your site's antivirus software, OS and access permissions are up to date.

Web Servers

Web servers act as the backbone of the Internet, yet they're particularly vulnerable to attack as they're "open" by nature, with users encouraged to send information to them and receive information from them. In most cases, attacks are unobtrusive, and servers and websites are injected with malware designed to infect as many users as possible. Criminals can modify the HTTPD (HTTP server daemon), database software and website code to modify their original function, and drive traffic to a rogue site.

It's important to remember that patching is also important for web servers. Content management systems (CMS) and blogging applications are frequently targeted for attack, and sites running vulnerable versions can be compromised very easily, using little more than a search engine.

Secure Application Coding

Your websites security also depends on secure application coding. The Open Web Application Security Project (OWASP) has developed security principles that should be followed for secure application coding. This will help to avoid SQL injection and other similar threats.
The recommended guidelines include:

- **Minimize attack surface area**—Every feature that is added to an application adds a certain amount of risk to the overall application. The aim for secure development is to reduce the overall risk by reducing the attack surface area.

- **Establish secure defaults**—By default, the experience should be secure, and it should be up to the user to reduce their security, if they are allowed.

- **Principle of least privilege**—Accounts should have the least amount of privilege required to perform their business processes.

- **Principle of defense in depth**—Where one control would be reasonable, more controls that approach risks in different fashions are better.

- **Fail securely**—Examine how applications fail to determine if an application is secure or not.
- **Don't trust services**—Implicit trust of externally run systems is not warranted.
- **Separation of duties**—Certain roles have different levels of trust than normal users.
  For example, administrators are different to normal users; but, in general, administrators should not be users of the application.
- **Avoid security by obscurity**—The security of key systems should not be reliant upon keeping details hidden.
- **Keep security simple**—Attack surface area and simplicity go hand in hand. Developers should avoid the use of double negatives and complex architectures when a simpler approach would be faster and simpler.
- **Fix security issues correctly**—Once a security issue has been identified, it's important to develop a test for it, and to understand the root cause of the issue.

Antivirus Software

Antivirus software is a must for any web server. When combined with a flexible firewall, it's one of the strongest forms of defense against security breaches. When a web server is targeted, the attack will often upload hacking tools or malware immediately to take advantage of the security breach before it's fixed.

To prevent this, you should enable real-time, on-access scanning for your web servers at all times. On-access scanning significantly reduces the chance of malicious code running on the system as it can scan in both "on read" and "on write" modes, and can then deliver an immediate notification as soon as any piece of malware tries to store itself on the server. Without good antivirus software, a security breach can go unnoticed for a significant amount of time.

Patching

Just as up-to-date antivirus software is essential in defending against computer viruses, so too is protecting your systems with the latest security patches to fix software vulnerabilities.

Patching your OS against security vulnerabilities is critical. Since vulnerabilities are often discovered before virus writers get the chance to exploit them, it is strongly advised that patches are applied as soon as possible. Patching any running applications such as blogs and guest books is also essential.

End User Protection

We also recommend a layered security strategy to protect end users. As the name implies, different types of protection layers are used to boost the overall strength of security scanning. This is the best defense in the ongoing game of cat and mouse where cybercrooks try to hide their malicious JavaScript attacks and security vendors work diligently to root them out.

Unfortunately, there's no silver bullet for security scanning, but a layered approach can go a long way to boost protection. Here's a summary of protection layers for end user machines:

- **Live URL filtering**—Blocks access to known malicious sites across all categories, includes filtering on the web gateway and endpoint
- **Content scanning**—Blocks content containing malicious code, scanning it on both the web gateway and on the endpoint
- **Exploit blocking**—Buffer Overflow Prevention System (BOPS) on the endpoint offers a significant level of generic protection against exploit driven attacks
- **Payload detection**—Real-time, on-access content scanning on the endpoint can block the attack's payload
- **Runtime protection**—If all else has failed and a user is infected with some undetected malware, you may still be able to block or remove the threat once it's running with Host Intrusion Prevention System (HIPS). This monitors runtime behavior in order to identify malicious activity.
- **Proper patching**— Patch all end user machines, and update as new fixes are made available

But, what about user education? Can't you just teach your users how to avoid scams and stay away from bad websites? Certainly user education can help, but in a classic drive-by attack, the user does nothing wrong. They simply browse to a perfectly legitimate site that's been infected. Users visiting a hijacked site have no way of knowing the site is compromised because the malicious code is invisible, and is executed as soon as the page loads in the user's browser.

Conclusion

Since JavaScript is an integral part of today's website functionality, attackers have taken advantage of its flexibility to obfuscate malicious code and hide attack payload from security scanners. Malicious JavaScript attacks are now a favorite way for cybercrooks to infect websites, and subsequently users' machines.
Your company's website could be hijacked, or your users could fall victim to one of these sites in their normal day-to-day activities.

With the right knowledge, you can know what to watch for, and understand how to protect your company and its users. A layered protection strategy enables multiple security defenses to work together to help to protect and defend against web threats, and particularly malicious JavaScript attacks.

Additional Resources:

If you are interested in learning more about this topic, please visit these Sophos resources:

- Why Hackers have turned to Malicious Java Script Attacks (Part I in this series)
- Web Attacks: Using Malicious JavaScript to Deliver Malware (web seminar)
- Malware with your Mocha (technical paper)
- Securing websites (technical paper)

Sources:

OWASP – Secure Coding Principles