feroot

Best Practices

to Improve Your Web Applications and Your Client-Side Security

Protecting your customers and securing your client-side is paramount to your growth and continued success as a modern business in the digital era.

In this white paper, we outline best practices to improve your web application and client-side security including how to protect your customers against browser-level attacks and the top client-side security mistakes to avoid.

Contents

Your Website Is the Center of Your Universe

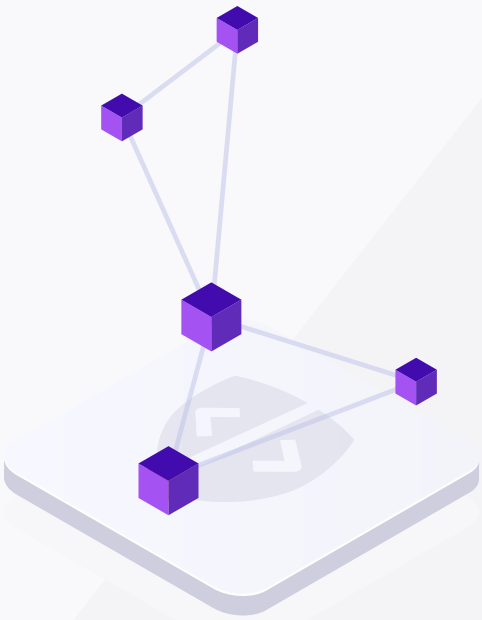
(and Client-Side Security Holds It Together)

To do business with any customer, organizations of all types and across all industries must have a digital presence. Marketers like to say “our website is the center of our universe,” and this definitely holds true. If a business does not have a website, consumers quickly question the validity of the business. Buying behaviors today dictate that a business must have an online presence available for the consumer to evaluate the goods and services sold as the first step in their decision-making process.

The customer experience doesn’t stop with websites. Many businesses also have web applications to drive the ease of doing business with customers. Businesses use web applications to bring in customers, engage them, and keep them coming back for more.

Your user experience, governed by your client-side web applications, is the force behind most of your organization’s cash flow. It keeps your organization in business, growing, and competitive. A good user experience helps you retain existing customers, gain new customers, upsell, cross-sell, and much more! However, it only takes one security breach in which customer data is stolen to damage your brand reputation and cause an exodus of customers fleeing to your competitors.

Client-Side Security Basics



What Is Client-Side Security?

Let's tell a story loosely based on a real breach to showcase a great example of the net outcome when an organization failed to provide client-side security.

Meet Jane. Jane is planning a trip to New York to visit an old friend from university. She would like to book a flight from London to New York. Jane's company has invested thousands of dollars in cybersecurity awareness training and, as a result, she is very conscious about protecting herself online. She is circumspect about most websites, especially those offering too-good-to-be-true bargains. She never shops on those websites offering 'act now,' last-minute deals.

To book her ticket, Jane goes directly to her favorite and trusted airline's website. She quickly picks a flight via the airline's web application, fills in the required information in the online form, and purchases the ticket. Later that day she receives her booking confirmation as well as her receipt from the airline.

A few weeks later Jane receives her credit card statement from her bank. The credit card balance and her bill for the month is much, much higher than usual. Confused, she closely reviews the statement and notes a variety of charges that she did not make. Jane feels anxious and frustrated. A number of thoughts race through her mind immediately:

- Was my card stolen?
- Who, when, and how did criminals gain access to my credit card?!
- What else did they steal?
- How long has this been going on?
- Will my bank be able to resolve this?
- Will I be reimbursed for these bogus chargers, or am I going to lose my hard earned money?

Jane immediately calls her bank. Of course, this is not the bank's first rodeo, so they void the charges and cancel Jane's credit card. The bank agrees to send her a replacement Visa card within 48 hours, with a new credit card number, CCV, and expiration date. Jane is thrilled that the bank resolved the credit card charges quickly, but now she can't make any credit card purchases for the next two days. However, the fun doesn't stop there. Jane must now change her payment details on every website and subscription service she used her old card on for payment. Jane thinks to herself: "Great, there goes several hours of my life I won't get back."

What Jane and her bank do not know is how and where her credit card information was stolen. Jane is a frequent online shopper and is now questioning if she still wants to do business with her usual organizations.

A few months later Jane reads the news. Her favorite airline had a significant data breach. Hackers managed to infiltrate the client side of her favorite airline's website and skim her credit card data when she booked her flight to New York.

As confirmation of her suspicion that her credit card information was stolen on the airline website, she also receives an email from the airline explaining that she is one of up to 380,000 people whose credit card details and traveler information, including email, name, and address were stolen in a web skimming attack by the Magecart criminal group.

Jane vows to try a different airline next time she is planning a trip.

Sounds pretty relatable, right? The majority of breaches of this type are malware-based skimming that go undetected for weeks or even months, on legitimate business websites.

So what happened?

The airline's booking webpage had a poorly secured script running on it. Criminals used that vulnerable component to add a few lines of malicious code to the website. By manipulating the website's code, the hackers were able to skim the data Jane entered into the payment form and send it back to their servers. The airline confirmed that its website and mobile app had been compromised and were actively being exploited by a group of cyber criminals. The aftermath for the airline was not pretty. They were fined $26 million by regulators and faced many other intangible costs that added up to tens of millions of dollars..

What now?

Hackers work hard to find your website's weaknesses.

The good news is that you can do something about it. There are ways to protect your customers, ensure a safe user experience, and make sure your websites and web applications don't get hacked. It is your business's responsibility to protect your customers by ensuring your client side is secure! By doing this, you can stay ahead of the threat and out of the news.

7 Security Measures to Detect and Defend from Client-Side Attacks

There are seven security measures
that help detect and defend from skimming attacks.

1. Web Application Firewall (WAF)
2. Content Security Policy (CSP)
3. Pentesting, Vulnerability Assessments, and Security Assessments
4. Client-Side JavaScript Vulnerability Scanning
5. Code Scramblers and Obfuscators
6. Client-Side Attack Surface Monitoring
7. JavaScript Security Permissions

What Is a Web Application Firewall (WAF)?

WAFs are a special category of firewalls that are designed specifically to protect web applications and businesses from falling victim to some skimming attacks. WAFs are deployed in front of web applications to analyze web traffic in order to detect and block malicious or unauthorized activity.

According to the Payment Card Industry Data Security Standards (PCI DSS), a WAF is defined as "a security policy enforcement point positioned between a web application and the client endpoint." Web application firewalls protect web applications from attacks, such as cross-site forgery, cross-site-scripting (XSS), file inclusion, and SQL injection.

However, WAFs are an open systems interconnection (OSI) layer 7 defense mechanism against application-layer attacks. They protect services that user-facing web applications apply to collect, store and utilize data. WAFs are not designed to protect the browser-level user interface itself. In other words, if a web application and its user experience is a house, then the WAF protects the walls, not the furniture or the people inside. In the end, WAFs are not able to detect and protect businesses from sophisticated skimming malware, drive-by skimming, supply chain attacks, or side-loading and chain-loading attacks..

WAFs & Drive-by Skimming and Supply Chain Attacks

WAFs are unable to detect manipulated JavaScript code or if data is being exfiltrated.

Threat actors don't like to waste their time or money. They tend to go after low hanging fruit in order to maximize their return on investment (ROI), and sometimes might follow a 'spray and pray' approach, rather than a targeted attack approach, to find vulnerable websites and web applications to exploit. It's relatively easy for a threat actor to add skimming to third-party JavaScript code. Once the code base has been infected, any organization that uses the code will automatically load it, allowing hackers to collect data from multiple organizations at once. This type of attack is commonly called "drive-by skimming." Third-party code is also highly vulnerable to supply chain attack, since the code is not part of the business's internal security oversight. This makes it open to tampering and malicious use. Attacking third-party code and tools gives hackers the ability to penetrate almost any website using the third-party code.

WAFs & Sophisticated Skimming Malware

WAFs are not able to detect and protect businesses from more sophisticated skimming malware.

There are a few variants of web skimming code that are notoriously difficult to detect and remediate. Pipka, for example, is a web skimming code with anti-forensic, self-cleaning, and stealth capabilities. It is able to remove itself from a webpage's code after it has been executed, thereby making it exceptionally difficult to detect.

WAFs & Sideloading and Chainloading Attacks

WAFs do not protect against skimming performed by a sideloaded JavaScript code.

Sideloading and chainloading are attack techniques that allow threat actors to load malicious or infected JavaScript code onto a target webpage using legitimate scripts and tools. As an example, Magecart Group 12 breached over 300 e-commerce websites by adding e-skimming code to the Adverline JavaScript library, a third-party digital ad platform. Adverline's retargeting script was then sideloaded with a skimming code, which in turn, skimmed customer data and sent it to a command & control (C2) server. Sideloading-based e-skimming breaches can go undetected for long periods of time because infected code is loaded directly by web browsers well outside of the security perimeter that cybersecurity analysts are tasked with protecting.

Curious about these security terms? Check out our Education Center to learn more about web application and client-side security.

Content Security Policy (CSP)

A Content Security Policy (CSP) is an added layer of security that helps businesses and security teams detect and mitigate certain types of attacks. These attacks include cross-site scripting (XSS), JavaScript code injection, and data skimming attacks. Threat actors try to circumnavigate CSP in order to steal data, distribute malware, or deface websites.

CSPs are a great addition to a web security stack, however, they do present some weaknesses when used as a sole security control:

- **Excessive and complex "allow list" rules**
- **CSP bypass techniques**
- **Incorrect CSP implementation**
- **CSP implementation tradeoffs**

In addition, in the case of most CSPs, it's not easy to develop and deploy a comprehensive policy. Because most websites and web apps are assembled using third- or fourth-party JavaScript libraries, developers end up "allowlisting" all external and internal hosts of scripts to avoid breaking the functionality of the code. Businesses face a tradeoff between security and functionality of their websites. The main problem with the "allowlisting" approach is that it removes or reduces the very protection CSP is supposed to provide.

The best way to use CSP is part of a layered approach to security when protecting web applications.

Pentesting, Vulnerability Assessments, and Security Assessments

A penetration test (pentest), is a deliberate and scheduled attempt by an organization to evaluate the security of its IT infrastructure by safely trying to exploit vulnerabilities. A vulnerability assessment is a systematic analysis and review of security weaknesses in a technology, system, application, or network. During security assessments, testers focus on security processes, governance, and compliance. Security professionals recommend that penetration tests, vulnerability assessments, and security assessments be conducted on a regular basis. In fact, regular pentesting and assessments are often required by cyber insurance providers in order to attain and retain coverage. Ultimately they are best practices you might already follow or need to start following to keep your websites and web applications safe. Penetration tests, vulnerability assessments, and security assessments have some limitations due to the fact that they:

- **Are time and resource intensive.**
- **Are limited in scope to certain applications, technologies, and networks.**
- **Require a skilled and experienced tester to be successful.**
- **Require specialized tools and technologies to uncover vulnerabilities and threats.**

Typically, pentests and assessments are performed as short-term projects and repeated on a quarterly or annual basis. Finding good pentesters is hard and they demand a high wage because of the specialized skills and experience they possess. Many organizations hire a managed security service provider (MSSP) to conduct the pentest. Now let's assume that a penetration test, vulnerability assessment, or security assessment is 100% accurate and provides actionable results. That's great. However, results are a snapshot in time, thereby arming hackers with the ability to execute attacks between quarterly or annual assessments. Hackers are always looking for new vulnerabilities to exploit and likely will know about new exploits before a pentest has been completed. Relying on quarterly or annual vulnerability assessments is a great start, but companies still remain exposed to breaches. Threats and cyber threat actors move much faster than any company can.

Client-Side JavaScript Vulnerability Scanning

There are quite a few vulnerability scanning products on the market today, many of which have been around for a long time. These tools are designed to scan and assess computers, software, applications, servers, and networks to uncover known weaknesses (a.k.a. vulnerabilities) that could be used for malicious purposes by hackers. Vulnerability scanners are used to identify and detect vulnerabilities arising from misconfigurations or flawed programming within network-based assets such as firewalls, routers, web servers, application servers, and more. Cybersecurity teams deploy vulnerability scanners to find potential inroads hackers could use to breach their networks and defenses.

Once a vulnerability has been found, vulnerability management and security teams then patch the vulnerabilities with software updates. If vendor software updates are not available, security teams find ways to reduce the potential harm or cyber risk they might incur if a hacker attempts to breach their network using the vulnerability as their entry point. Vulnerability scanners are a necessary technology for any cybersecurity program. However, they are not useful for client-side security, as they are not designed to support client-side security efforts. Vulnerability scanners are designed to scan back-end code and systems, typically those digital assets that live on the server side.

They also aren't able to detect and enumerate all JavaScript scripts and vulnerabilities. Vulnerability scanners can only see the client side after it's been compiled together, not in real-time. They also can only see a single domain, not all of the links that are part of it.

Code Scramblers and Obfuscation Technologies

What Are Code Scramblers and Obfuscators?

Code scrambling or code obfuscation is a process by which easy-to-read code is distorted to make it difficult to comprehend. The goal of code scrambling and obfuscation is to make it difficult for competitors, other developers, and threat actors to reverse engineer or modify it. Code obfuscators allow websites to load properly in the browser without being different to the naked eye.

Web application developers and security teams acquire code obfuscators in order to hide JavaScript web application code that threat actors or competitors might target. By concealing portions of the code, developers hope to reduce the risk that threat actors might use the code for malicious purposes. It is quite easy and cheap to scramble or obfuscate code.

There are many free code obfuscation tools available, but they come with some great limitations. First, with time and effort, web applications that were obfuscated with free code obfuscating technologies can be reverse engineered to uncover a version of the original application. This is why hackers use code obfuscation to their advantage. There are also code de-obfuscators on the market. These simply do not work. Some paid code obfuscating technologies pollute the original web application code to such an extent that you can't even get remotely close to the original code if you try to unscramble it. Herein lies a substantial problem. If you obfuscate your web application code, you can't unscramble it. You can't go back to the original code. So it gets extremely hard to spot security issues and vulnerabilities in the code. Normal code and malicious code in a scrambled web app looks exactly the same. Your web developers can no longer see issues in the code with the naked eye and are likely to miss critical vulnerabilities. Some businesses circumvent this by implementing dynamic code analysis to track issues, but this is time-consuming, costly, and inefficient.

Client-Side Attack Surface Monitoring

What Are Client-Side Attack Surface Monitoring Solutions?

Client-side attack surface monitoring solutions are a relatively new cybersecurity technology that automatically discover all of a company's web assets and report on their data access. These solutions use headless browsers to navigate through all the JavaScript contained on the website and web application pages. The technology gathers real-time information about how the scanned website works from the end-user perspective.

A key component of the technology is synthetic users, which are deployed during threat detection scans to act and interact the way a real human would when completing websites and web application tasks. These synthetic users can complete a variety of activities, including but not limited to:

- Scrolling through pages
- Submitting forms
- Solving CAPTCHAs
- Entering financial information
- Clicking active links
- Watching embedded videos
- Waiting for pages to load
- Navigating between pages
- Clicking on, opening, and closing pop-up messages

Each interaction conducted by a synthetic user with the web application is logged and monitored. These solutions then engage behavioral analyses and inject logic into each page to gather information that is difficult to collect manually, including:

- The type of data collected by forms.
- The type of data third-party scripts have access to.
- Any first- and third-party scripts that are fingerprinting users and their browsers.
- The types of trackers that are deployed on the page and their activities.
- The existence of any forms or third-party scripts transferring data across international boundaries or to unauthorized entities.
- Any first- and third-party scripts which are being loaded directly into the user's browser.
- Any first- and third-party scripts that are being sideloaded or chainloaded into the user's browser.
- The presence of any malicious hosts exfiltrating data.
- Whether any data is being exfiltrated via WebSockets and the location where it is being exfiltrated.

Evaluation of web applications from a security perspective isn't the only thing that client-side attack surface monitoring solutions do. They also perform post-scan informational analyses to offer businesses synthesized intelligence to secure web applications from harm. In addition, they analyze all information synthetic users collect and enumerate client-side threat intelligence for security teams to act on quickly and effectively.

Built-in machine learning capabilities also identify and classify data to detect and report on a variety of client-side security challenges.

The type of intelligence available includes::

- Active malware
- Live marketing or other tracking software
- Geographic IP information
- Obfuscated scripts
- Data assets collected (financial, PII, etc.)
- Historical overview of your client-side attack surface
- Client-side security trends
- Types of webpages (login, billing, etc.)
- SSL issues
- Known JavaScript vulnerabilities

Client-side attack surface monitoring solutions are easy to set up and maintain on existing web applications, and can discover more client-side cyber threats than any of the approaches discussed in this white paper. These cyber defense technologies do require interaction between cybersecurity and application development teams. To properly secure businesses from client-side threats, teams need to understand the ins and outs of client-side application structures, as outlined earlier.

With strong collaboration between security and application development teams, businesses can secure their client-side web applications with ease.

JavaScript Security Permissions

Traditional software and applications, that is, those not written in JavaScript, come with a menu or functions to set user permissions.

By default, JavaScript environments do not have a security permissions model built-in. Third-party JavaScript code can have an unrestricted level of access to sensitive data at the browser level, so the attack surface is broad and wide open. There are a few client-side security products on the market today. They add security permissions and controls to JavaScript. Application developers and security teams simply have to add a few lines of code to their websites and web applications.

The JavaScript security solution automatically applies security configurations and permissions for continuous protection from malicious client-side activities and third-party scripts. These solutions integrate directly into the runtime environment of every user browser session to enable proactive monitoring and defense. JavaScript security permission solutions essentially deploy the Zero Trust model on JavaScript applications and run continuously in the background to automatically detect unauthorized scripts and anomalous code behavior.

After detection, they block all unauthorized and unwanted behavior in real time across an organization's web assets. JavaScript security solutions monitor and respond to browser-level security events in real time by auto-instrumenting themselves on every website and applying security configurations to every user browser session. If an application development or security team deploys JavaScript security permissions on all of their client-side pages and applications, then third-party JavaScript code can't be tampered with and data can't be exfiltrated by threat actors. Coupled with proactive scanning of client-side assets, application security and cybersecurity teams will receive alerts with context, to repair client-side security issues, all while being protected.

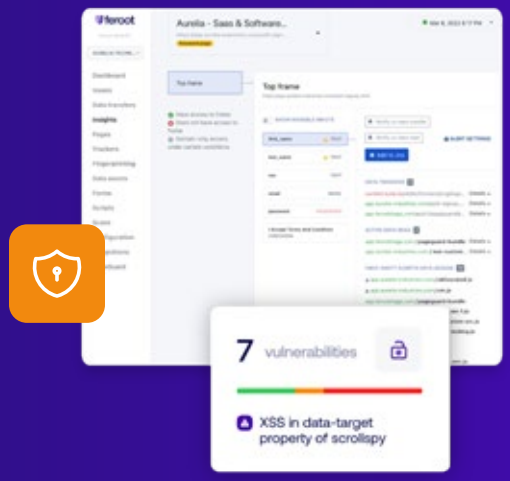
Client-Side Security Technologies

There are limitations in older existing client-side technologies.

Client-Side Security Technologies	Limitations
Web Application Firewall (WAF)	Can't protect businesses from: • Sophisticated skimming malware. • Drive-by skimming. • Supply chain attacks. • Sideloading attacks. • Chainloading attacks.
Content Security Policy (CSP)	If used as a sole security control, may expose businesses to e-skimming breaches due to: • Misconfigurations and excessive "allow list" rules. • Bypass techniques. • Incorrect implementation or tradeoffs. • Difficulty developing and deploying a comprehensive policy.
Penetration Testing and Vulnerability and Security Assessments	Even if conducted on a regular basis, pentesting and vulnerability and security assessments: • Are time and resource intensive. • Are expensive. • Reflect conditions based on a single point in time. • Are limited in scope to certain applications, technologies, and networks. • Require skilled and experienced personnel to conduct the tests or assessments. • Rely on specialized tools and technologies.
Vulnerability Scanners	Because vulnerability scanners are not designed to support the client side, they: • Scan only server-side (back-end) assets, not web applications and websites. • Aren't able to detect and enumerate JavaScript code and vulnerabilities. • Can only view a single domain, not all of the links that are part of it.
Code Scramblers and Obfuscators	Code scramblers and obfuscators come with some significant issues, such as: • Scrambled code cannot be easily unscrambled. • It is much more difficult to find vulnerabilities and problems in obfuscated code. • If the obfuscated script includes third-party code, it becomes difficult to identify malicious or corrupted third-party content.

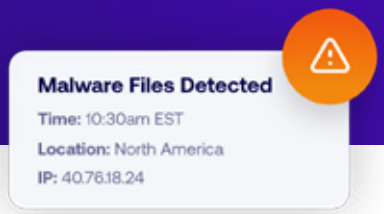
Advanced Client-Side Security Technologies

Advanced client-side solutions can protect from attack and data exfiltration through automation and advanced synthetic user technologies.



Client-Side Attack Surface Monitoring

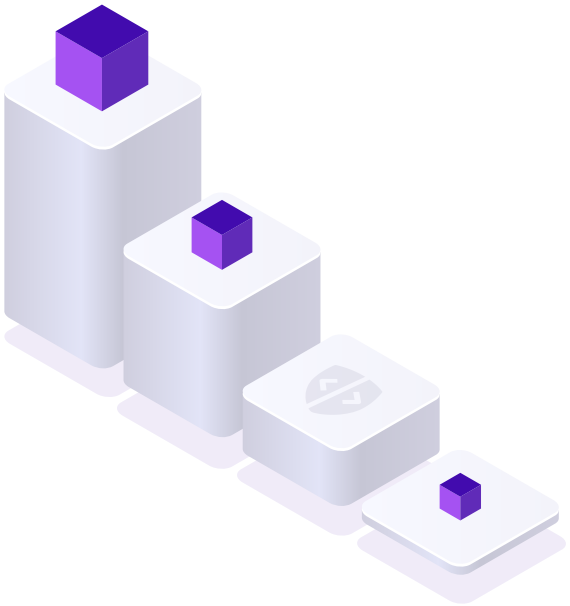
- Newer, advanced threat detection technology.
- Identifies and classifies data and threat intelligence to detect and report on client-side security vulnerabilities and attacks.
- Automatically discovers all web assets and reports on their data access.
- Uses headless browsers to navigate through all JavaScript contained on the website and web application pages.
- Gathers real-time information about how the scanned website works from the end-user perspective.
- Deploys synthetic users during threat detection scans to act and interact as a real human would.
- Logs and monitors each web application interaction.
- Uses behavioral analyses and injects logic into each page to gather information that is difficult to collect manually.



JavaScript Security Permissions

- Automatically applies security configurations and permissions for continuous protection from malicious client-side activities and third-party scripts.
- Blocks all unauthorized and unwanted behavior in real-time across an organization's web assets.
- Prevents data exfiltration.
- Integrates directly into the runtime environment of every user browser session to enable proactive monitoring and defense.
- Deploys the Zero Trust model on JavaScript applications.
- Runs continuously in the background to automatically detect unauthorized scripts and anomalous code behavior.
- Monitors and responds to browser-level security events in real-time by auto-instrumenting on every website and applying security configurations to every user browser session.

Grow Your Business and Avoid Data Breaches with the Right Type of Client-Side Security



Regardless of your role as cybersecurity professional, application developer, or marketer, you are responsible for protecting your most critical assets—your customers

In order to grow your business and avoid costly data breaches, it's your responsibility to prevent client-side attacks and the associated data exfiltration. The ultimate goal is to maintain secure client-side web applications and webpages to deliver a user experience without risk or compromise. The dangers that come through the client side are significant, as are the historic challenges to defeat them—complexity, a lack of visibility, and an inability to uncover, remediate, and prevent client-side attacks. But with knowledge of what is needed, it can be achieved.

Implementing effective client-side security threat defenses is crucial to ensure the safety of your customer data, the integrity of your user experience, the functionality of your web applications and websites, and the ability for your business to grow and succeed. If you are interested in automating your client-side security operations and hardening your front-end defenses please reach out to us. Our client-side security specialists stand ready to help you protect your business and your customers.

feroot

About Feroot

Feroot Security believes that customers should be able to do business securely online with any company, without risk or compromise. Feroot secures client-side web applications so businesses can deliver flawless digital user experiences to their customers. Leading brands trust Feroot to protect their client-side attack surface. Visit www.feroot.com.

Ready to Protect Your Client Side?

To get a personalized overview of our Inspector and PageGuard solutions, send an email to sales@feroot.com.