feroot

FEROOT.COM

The Ultimate Guide to

Client-Side Security

Learn everything you need to know about client-side security to protect your JavaScript web applications and customer data.

Contents

01

Introduction

In today's world, businesses, economies, and lives are connected by a complex spiderweb of code and software applications. This code and these applications drive e-commerce, financial transactions, and data input. They impact our ability to quickly transfer money from one account to another, to fill out an online mortgage application, and to order supplies from a vendor.

The code that drives these systems is complicated. And, well...let's face it. Murphy's Law is working overtime when it comes to software code. If something can go wrong, it will.

The important thing to understand about software code is that hackers and attackers actively search for vulnerabilities to exploit, in even the most well-written code. And increasingly, they're looking for vulnerable code on the 'client side.'

Take a Walk on the Client Side

Modern websites, and the software that drives them, carry risk. Any customer—be it an individual consumer or another business—that wants to conduct a transaction via a website is going to expect a seamless and safe user experience with minimal or no risk. The first step in protecting customers is making sure the entry point into your business—the client-slide—is as secure as it possibly can be.

In this e-book, we offer businesses a comprehensive look at client-side security and the type of attacks that are increasingly targeting businesses that deliver client-side products and services. The security gap around the client-side is growing and organizations need to be prepared to secure their front-end operations if they want to ensure business growth and customer safety. The unique client-side attack surface requires a specific and dedicated security approach that is different from traditional server-side security. Protecting the client side means understanding and acknowledging the risks and taking appropriate action to protect any person or business that comes in contact with client-side operations.

Know Your Terminology: Client Side vs. Server Side

What Is the Client Side?

Within the context of IT, the words “client” and “server” relate to the basic structure of connected devices. So, end-user devices, like desktops, laptops, mobile phones, and tablets are considered ‘clients,’ while the systems that the devices are connected to are referred to as ‘servers.’ For web developers, the term ‘client side’ means any activity that is taking place on the client device. This could be a webpage as it is being displayed, including text, videos, and images, or any operation or calculation underway in the background.

vs

What Is the Server Side?

Server side refers to anything happening on the server instead of the end user’s device. In the short span of internet history, there was a time when everything operated on the server-side, even things like dynamic webpages. However, the process of transmitting data from the end user’s device (client-side) to the server side took far too long—something referred to as latency. To combat this problem, web developers started building code (sometimes called scripts) that operated more on the client side, making webpage operations much faster.

What Is Client-Side Security and Why Is It Important?

Client-side security refers to the technologies and policies used to protect an end user from malicious activity that is occurring on dynamic webpages accessed from the end user's own device. Client-side security is designed to protect the end user from incidents, vulnerabilities, and attacks that occur within an end user's browser. It is also sometimes referred to as the "front end" in the context of code development for web applications. Client-side attacks have been increasing in both scale and cost since the beginning of 2020 as companies expand their investment in the end-user digital experience. This has created an unprecedented opportunity for threat actors to exploit end-user activities.

To fully manage the risk against breaches and attacks, companies must simultaneously protect both the client side and back end of their application portfolios. This includes everything that the customer sees, such as text, images, stylesheets, and the rest of the user interface (UI), along with anything the customer interacts with, such as what the website or web application does within the user's browser.

One of the most important actions any business can take is protecting their customers from client-side threats. Unfortunately, because of the sophisticated and subtle nature of these attacks, they can be hard to detect until it's too late. To ensure that businesses are offering a safe and secure digital experience, they must be diligent about securing their website and web applications from dangerous client-side attacks.

02

Client-Side Attacks and JavaScript Code

What is JavaScript?

JavaScript (JS) is a text-based programming language used on both the client side and the server side. JS allows web and application developers to build websites and web applications with interactive elements that engage the user and enhance their experience.
JS is relatively easy to learn and easy to share.

98%+

of websites use JavaScript for client-side webpage behavioral elements.

JavaScript serves as one of the core technologies used to build web applications and websites accessed by consumers. Over 98% of websites use it for client-side webpage behavioral elements. Eighty percent of websites use a third-party JS library or web framework for their client-side scripting. What this means is that websites are assembled with various pieces of third- or fourth-party JS code, which does not have any security permissions built into it. JS is inherently vulnerable to cyberattacks. It allows threat actors to deliver malicious scripts to run on a client computer via the web.

Developers and marketers love JavaScript because of its flexibility and ease of use. Security teams are nervous about it, because JS does not have "security permissions" built into it and is difficult to keep safe from client-side-focused threat actors.

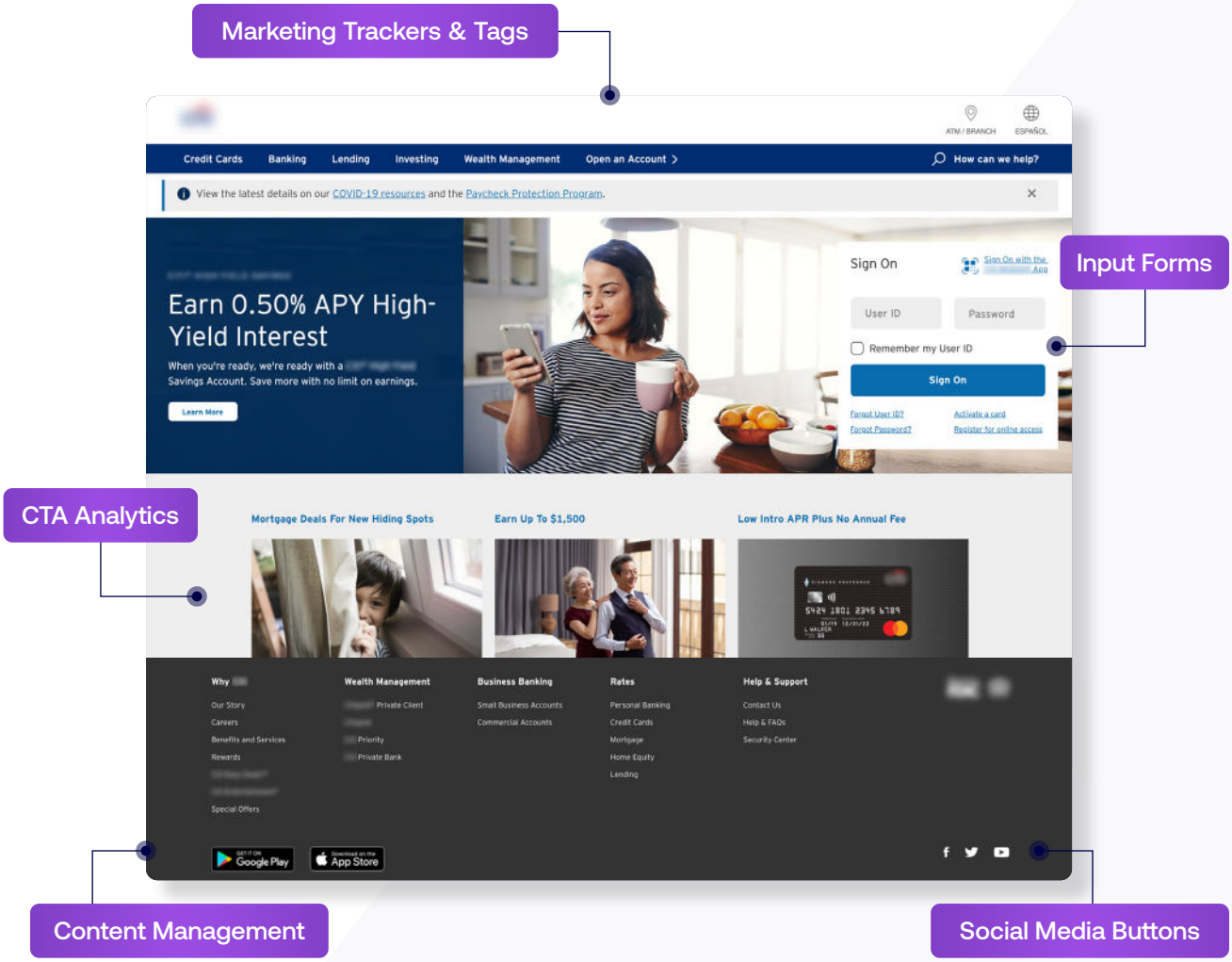
What Is the Relationship Between the Client Side and JavaScript?

JavaScript is the most commonly used client-side process

Many webpage operations that an end user sees are actually happening only on the end user's device and not on the server. Webpages achieve this by creating code within the web application to annotate or format text on a webpage. This code is called "markup language" (e.g., HTML).

Web developers also use something called 'client-side processes' in their application architecture. Client-side processes enable dynamic webpages to operate on the client device, instead of the server. (Dynamic webpages display different content depending on user input.) JavaScript is the most commonly used client-side process. Markup languages and client-side processes enable the web developer to build an interactive web application architecture that responds to something that the end user is doing on the client device. For example, if a person is shopping on a website, a client-side process would recognize the end user's mouse moving over to a form and clicking on a blank space to fill in a credit card number.

Businesses are putting too much faith in the security of the dynamic supply chain

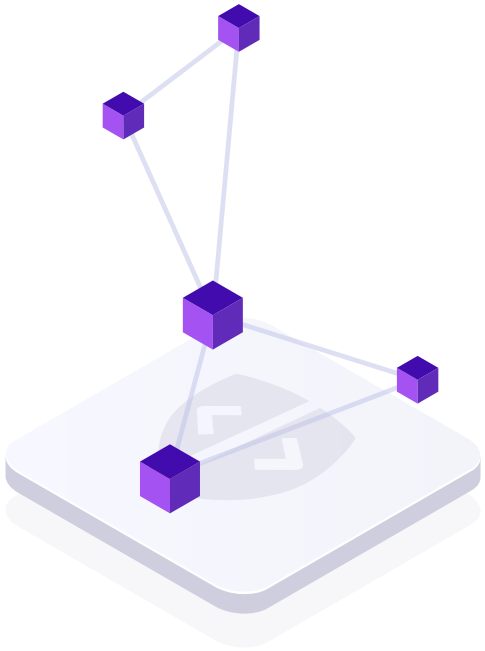


Client Side

- CDNs, APIs
- WebSockets
- XHR/Fetch
- Open Source
- Documents
- Stylesheets
- Scripts
- Images

Server Side

- Database
- Content
- APIs
- Static Assets
- Storage
- Libraries



What Does JavaScript Have to Do With Security?

Because client-side activity happens outside a business's security perimeter, typical security technologies will not protect the end user from malicious activity that is occurring on dynamic webpages accessed from the end user's own device. So, let's say you operate a small online shop that uses JavaScript. A threat actor has injected some malicious code into your JavaScript code that enables them to capture a customer's credit card number when the customer types it into the payment page. Your standard security won't see this happening because it is not happening on your server. Instead, It is happening on the dynamic webpage that your customer sees and interacts with. In a nutshell, your customer has downloaded malicious code from your server, which is then interpreted and rendered by the user's browser on the user's device.

Recent research suggests that web application attacks account for more than a quarter of all data breaches. Since many web applications operate on the client side, your customers become vulnerable to attacks like Magecart, e-skimming, sideloading, cross-site scripting (XSS), and formjacking. But vulnerable web applications are not the only way that threat actors access client-side assets; server-side website security misconfigurations and the addition of third- and fourth-party code or supplemental applications can also cause security problems.

Therefore, implementing effective client-side security is critical if you want to protect your customers.

Why Is JavaScript Vulnerable?

JavaScript is vulnerable because it is easy for hackers and other threat actors to input query strings into forms to access, steal, or contaminate protected data. JS is the standard for the processing of personal information in client-side websites and applications. There are numerous open-source and third-party libraries available today that include solutions written in JavaScript. These libraries offer easy access to threat actors and often contain code with known vulnerabilities. Malicious or vulnerable third-party code can provide a threat actor with unrestricted access to sensitive data at the browser level, so the attack surface is broad and wide open.

By default, JavaScript environments do not have a security permissions model built in. The World Wide Web Consortium standard is that security permissions (what code is able to execute and what script activity is allowed) are housed in browsers, and the responsibility to manage them lies with the site or application owner. Therefore, the onus is on site owners (and not the end user) to implement policies and procedures to detect and mitigate JS borne cyberattacks.

03

Modern Web Application Architecture 101

Front-End vs. Back-End Frameworks

The front end and the back end are two very specific terms in the software development world. Front end developers and back end developers complete very different tasks, but they must be closely aligned to build reliable and secure web applications. The front end, or the client side, is what your users see and interact with.

Front-End or Client-Side Web Application Development

When a customer or user goes to your website, they interact with your front end directly to learn about your business or utilize its web-based services. The front end includes everything that your users experience, such as the graphics, the third-party JavaScript code the team included to enhance the user experience, the buttons, the colors, the navigation menu, etc. Currently there are three technologies that are used to develop client-side web applications: HTML, CSS, and JavaScript. JavaScript is the actual programming language which is responsible for the business logic, user interaction, data models, and more.

Front-End Frameworks and Libraries

JavaScript-based, front-end web applications tend to be built on a few discrete frameworks and libraries.
The most popular ones are:

Angular JS

Developed by Google, Angular JS is a JavaScript open-source, front end framework. It's used by enterprises to develop some of the largest web applications in the world. Like most JavaScript frameworks and libraries, it is open-source and hence is a continuously growing and expanding framework.

React JS

Developed by Facebook, React JS is another widely leveraged framework used to build user interfaces. It is also an open-source, component-based, front-end framework.

Bootstrap

Bootstrap is an open-source library used for quickly building user interfaces. It is used by developers to create responsive websites and web applications. Many developers like Bootstrap and favor it over other frameworks when developing JavaScript-based, responsive, mobile-first websites.

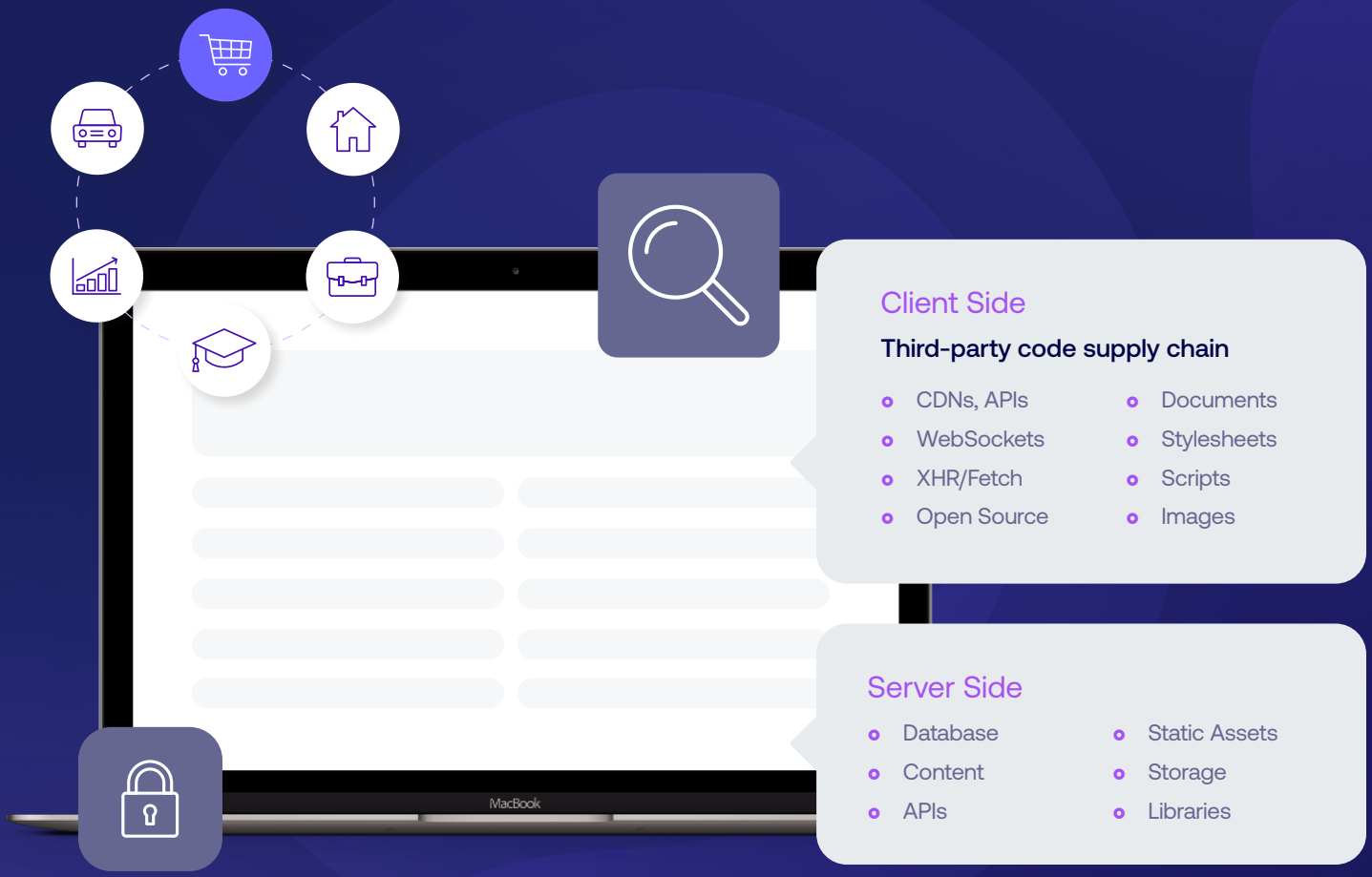
jQuery

jQuery, yet another open-source library, is used by developers to simplify the interactions between an HTML/CSS document, and the Document Object Model (DOM) and JavaScript.

Back-End or Server-Side Web Application Development

The back end, or the server side of the website, stores and arranges your data. Back-end developers make sure that everything the client-side development team built works flawlessly. Arguably, the back end of the website is the more difficult piece of the puzzle to develop and maintain. If something breaks on the back end, the entire front end could collapse. Your users, customers, and colleagues do not ever see what happens on the back end. The only people at your organization that might see the back end (beyond your developers), is your marketing team when they make adjustments to the website.

1. Outside-in gives quick time to value.
2. Complete all-in-one solution, no need to cobble different tools.
3. Operates autonomously, saves countless employee hours.



Client Side

Third-party code supply chain

- CDNs, APIs
- WebSockets
- XHR/Fetch
- Open Source
- Documents
- Stylesheets
- Scripts
- Images

Server Side

- Database
- Content
- APIs
- Static Assets
- Storage
- Libraries

Why Is Front-End Business Logic Becoming More Prevalent?

The front end, also known as the client side, is what facilitates the digital user journey. It is the common ground for modern-day business logic. Why? Performance. The less requests made to the server, the more performant a web application can become. Utilizing JavaScript libraries and frameworks, developers can build modern digital experiences directly in the browser.

Unfortunately, developers are not the only team delving into the the client side. Threat actors are taking a deep look at the front-end business logic to see how they might be able to deface the front end, deploy malware, or infiltrate the back end to cause damage.

Some developers prefer not to work on the client side, because they feel that the front end is not true programming. This characterization is antiquated in today's digital world since most new businesses are building cloud-native web applications that are supported by software-as-a-service (SaaS) solutions.

Risk of Leveraging Third Parties for Innovation

Historically, websites were coded line by line, by one developer or one development team, usually in pure HTML/CSS/JavaScript without many third-party libraries. Now, they are assembled: large chunks of prewritten code are appropriated from open source communities like Github. Themes, scripts, templates, plugins, and blocks of code from disparate sources are commonly pulled together to make a website. In fact, modern web applications load an average of over 20 third- and fourth-party scripts as part of the user experience. While third-party code and applications can quickly add functionality, they often contain vulnerabilities that can be easily manipulated by threat actors. WordPress is the world's largest web platform, with tens of thousands of ready-to-use plugins, available as easily as hitting the download and install buttons. These components load hundreds of JavaScript scripts from all over the world.

Many web application development teams use third-party plugins and scripts for convenience purposes. It's easier and much more cost effective to deploy a tried, true, and tested script than write something similar from scratch. The greatest benefit of using third-party scripts is being able to take advantage of the creativity and ingenuity of other developers who are not part of your immediate development team. The application development community is filled with brilliant minds all over the globe who have great ideas, build innovative scripts, and then share them with their community on open-source platforms. There are thousands of developers out there who feel intimately connected to their work and their community, and nothing makes them happier than helping their colleagues get better and build cooler web applications. Developers are just awesome like that. They love to build interesting stuff and make it available for others to use.

As mentioned earlier, third- and fourth-party code is often accessed from open-source repositories like GitHub. And herein lies the issue. When you bring in somebody else's code, you not only incorporate the code's logic, but you also incorporate everything else the developer has chosen to embed—including vulnerabilities and coding errors. When building a website using third-party code, you are trusting the developer to have considered application security while they wrote it.

Unfortunately, for many developers, security is an afterthought, so businesses risk inadvertently introducing vulnerabilities, code weaknesses, and, at worst, malware into their web app or site.

The downside to leveraging third parties for innovation is that their scripts might be infected with spyware and keyloggers purposefully built to steal sensitive data without the company's knowledge. If installed, this malicious code can lie undetected and seemingly benign, while performing countless nefarious acts. For instance, it can scrape customer information from a form field, so when someone is completing a landing page, the malicious code is also mirroring and capturing their data. It may do the same thing on payment pages or chatbots—essentially anywhere the user is keying in information to interact with the website. So leveraging third parties is a gamble. With a roll of the dice, you are taking risks on the safety of the third-party scripts.

What Is Authentication and Authorization?

It doesn't matter whether you have a simple, single-page application or a large, enterprise-scale web application, protecting your users and their data is paramount. Over the last few years, the use of JavaScript has ballooned because it is flexible, easy to use, and easy to modify. JavaScript, as described earlier, is flexible, easy to use and easy to modify. Troves of developers prefer to use it for building unique and user-friendly web applications.

With increased use comes not only innovation, but also increased issues. Over the past few years, there are many processes previously done on the server side, now move to the client side. In some cases, this is a great development; in others it produces more challenges. So, as Uncle Ben said to Peter Parker in Spiderman: "With great power comes great responsibility."

Authentication

One of the key technologies that still sits in both the client side and the back end is identity management. While you can roll your own authentication, it's more common to use a third party whose sole focus is providing secure identity services.

When the user goes through the authentication process, they are issued a token inside the browser which can be used for subsequent logins (until expiry). On the front end of the application, the developer can use this token to provide dynamic navigation, profile pages, and more.

Authorization

Typically controlled by the back end through middleware and role-based access, authorization has less of a play on the client side but is still important to understand for completeness of security when it comes to protecting web applications. Notably, broken access control is one of the biggest risks to web applications today, moving up to the number one spot in the OWASP Top 10.

Client-side Data Storage

One area where web applications have not changed much is with the utilization of databases. Using a back-end data store allows web applications to make queries and render the selected data into the user's view (e.g., browser).

Client-side data storage works in a similar way, but instead of being stored in the back end, the data is stored within the browser. JavaScript can then be used to access the browser directly (no back end required). This allows businesses to develop and maintain fantastic web applications and websites that deploy many unique features to enhance the user experience, including:

- Personalized site preferences for the user based on previous data to delight the user on a personal level.
- Maintaining previous website activity and interactions to make it easier for the user to do business, (e.g., keeping shopping cart contents or keeping the user logged in).
- Storing data and web assets locally so the site will load quicker for the user.
- Saving documents generated by the web application locally so that the user can still read the contents even when offline.

04

Client-Side Risks and Threats

There are dozens of methods that cyber threat actors can use to steal personally identifiable information (PII), financial transaction data, and other confidential and proprietary data from businesses. Traditionally, this involved breaching a target's corporate network. Today, there are a multitude of new vectors that can be used to steal data from companies, including the use of malicious scripts directly on the front end of a website or web application to start skimming data as a user enters information into a form. Digital skimming is a client-side cyber threat that requires improved security on any website or web application to protect customers interacting with the website from losing sensitive information.

Client-Side Security Threats and Attacks

Even as businesses are shoring up client-side vulnerabilities, bad actors are stepping up their threats and attacks. Fortunately, the most popular of these attacks are well known. By educating your staff and organization on the most common client-side security threats, it's easier to learn how to stop them in their tracks.

01 Client-Side Vulnerability Attacks

Cross-Site Scripting

Cross-site scripting (XSS) is one of the most common ways for bad actors to launch a client-side attack. In a typical XSS attack, malicious code is injected into website content which targets unsuspecting users viewing that site. This threat is so common that it's regularly on The Open Web Application Security Project's (OWASP) list of Top 10 Vulnerabilities.

DOM-based XSS

In a document object model (DOM)-based, cross-site scripting (XSS) attack (sometimes called "type-0 XSS"), the threat actor's payload is executed as a result of modifications to the DOM environment in the victim's browser, which was used by the original client-side script. As a result, the client side code runs differently than originally designed. The page itself does not change, but the client-side code on the page executes malicious DOM-environment modifications.

Directory Traversal or Path Traversal

Directory traversal or path traversal attacks exploit weak security validation or sanitization of user-supplied file names, such that characters represent "traverse to parent directory," which are passed through to the operating system. Directory traversal or path traversal are HTTP-borne attacks which grant attackers access to restricted directories and allows them to execute commands outside of the web server's root directory.

02 Web Skimming Attacks

E-skimming

E-skimming, commonly referred to as a 'Magecart' attack, is a process in which malicious threat actors, nation-state-sponsored hackers, and financially motivated hackers gain access to an online store of a company. These threat actors inject skimming code onto payment card processing pages of the website in order to make financial gain.

Magecart

Magecart is a commonly used name for loosely affiliated threat groups that use digital skimming or e-skimming techniques to steal customer data. The term "Magecart" is a portmanteau of the words "Magento" (a popular open-source, e-commerce software platform) and "shopping cart." It operates using a small piece of code, which is inserted into a website to skim information from customers. Most often, it's inserted into payment pages, where it acts as an online credit-card skimmer, pulling the personal information and credit card details of anyone who comes across it.

E-commerce Platform Skimming

Beyond Magecart, some of the world's most popular e-commerce platforms, like Volusion, have also been breached by Magecart e-skimming code. These e-commerce platforms provide checkout services to thousands of merchants, while collecting and utilizing massive amounts of customer credentials, personal data, and financial information. Thousands of online stores were compromised during the Volusion platform hack in 2019, which went undetected for weeks.

Drive-by Web Skimming

In drive-by web skimming, a threat actor compromises third-and fourth-party code with malware, with the hope that multiple organizations use this code and infect their websites and web applications inadvertently. Modern web applications load an average of over 20 third- and fourth-party scripts as part of the user experience. Compromising one of these third- or fourth-party elements with malicious JavaScript allows an attacker to compromise multiple websites simultaneously.

Trusted Cloud-Hosted Platform Skimming

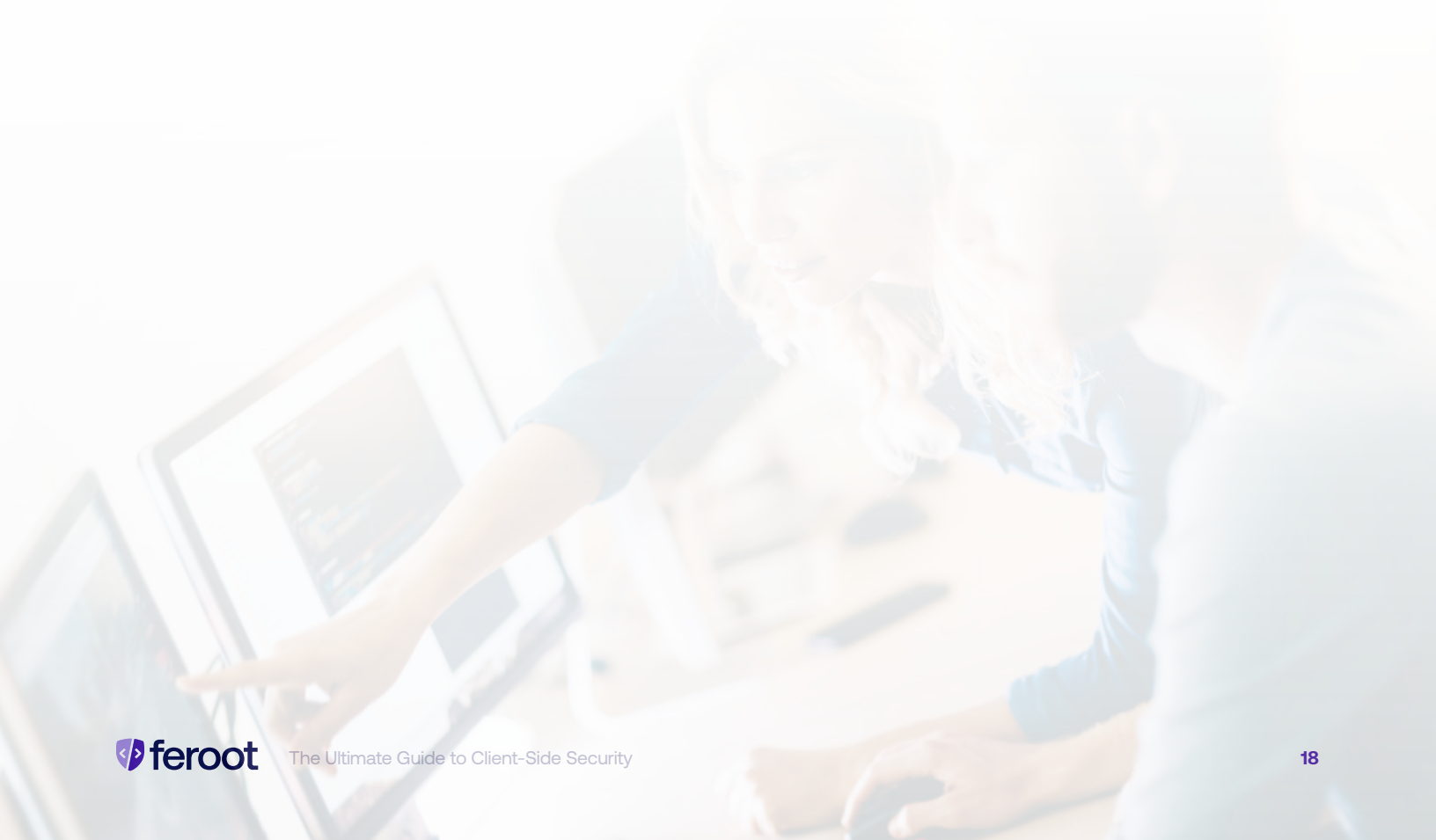
Malicious e-skimming code has been found hosted by popular cloud platforms, including Salesforce Heroku, Amazon CloudFront CDN, and many others. Magecart e-skimmers were found following a spray and pray approach that targeted misconfigured Amazon S3 buckets. Skimmers were able to open backdoors and insert malicious code into JavaScript libraries used by thousands of organizations.

Public Wi-Fi Skimming

IBM researchers discovered this type of e-skimming attack on public Wi-Fi hotspots. In a public Wi-Fi skimming attack, threat actors infiltrate a large number of users in public spaces, such as airports and hotels, by skimming off unprotected public Wi-Fi. Threat actors insert skimming code via Wi-Fi hotspots allowing them to exfiltrate the data users enter on web forms. These forms may include e-commerce checkout pages, marketing landing pages, or login pages. Public Wi-Fi skimming is quite a simple attack for threat actors to execute, since vulnerable Wi-Fi routers allow attackers to automatically inject skimming scripts into all websites accessed by users through their connected devices.

Anti-forensic, Self-Cleaning, and Stealth Data Skimming

There are a few variants of web skimming codes that are notoriously difficult to detect and remediate. Pipka 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. Pipka-like threats require focused attention by cyber defenders and an automated scanning solution that alerts them to questionable code behaviors.



03 Malicious Script Injection Attacks

JavaScript Injection

During a JavaScript injection attack, a hacker gains website or web application parameter information and changes their values. This allows the threat actor to manipulate the website or application and collect sensitive data, such as PII or payment information. Threat actors tend to use JavaScript code obfuscators or scramblers to make it difficult for web application developers or security experts to see the malicious code and detect threats.

SQL Injection

SQL injection is a code injection technique used by hackers to attack data-driven applications. Threat actors insert malicious SQL statements into an entry field and then execute the statements. Attackers can thereby interfere with the queries the target application makes to its database, allowing the attacker to view and collect proprietary or protected data.

XML Entity Injection

XML External Entity (XXE) Injection is a web security vulnerability that allows attackers to interfere with an application's processing of XML data. Attackers exploit these vulnerabilities to view files on the application server file system and to interact with any systems that the application can access.

Formjacking

In formjacking, cyber criminals, financially motivated threat actors, and other hackers insert malicious JavaScript code into their targets' website. Their goal is to take over the functionality of the site's form pages to collect sensitive user information or valuable data. This information can include credit card information, personally identifiable information (PII), and other data that can be used to breach networks or be sold on the dark web.

Sideloading and Chainloading

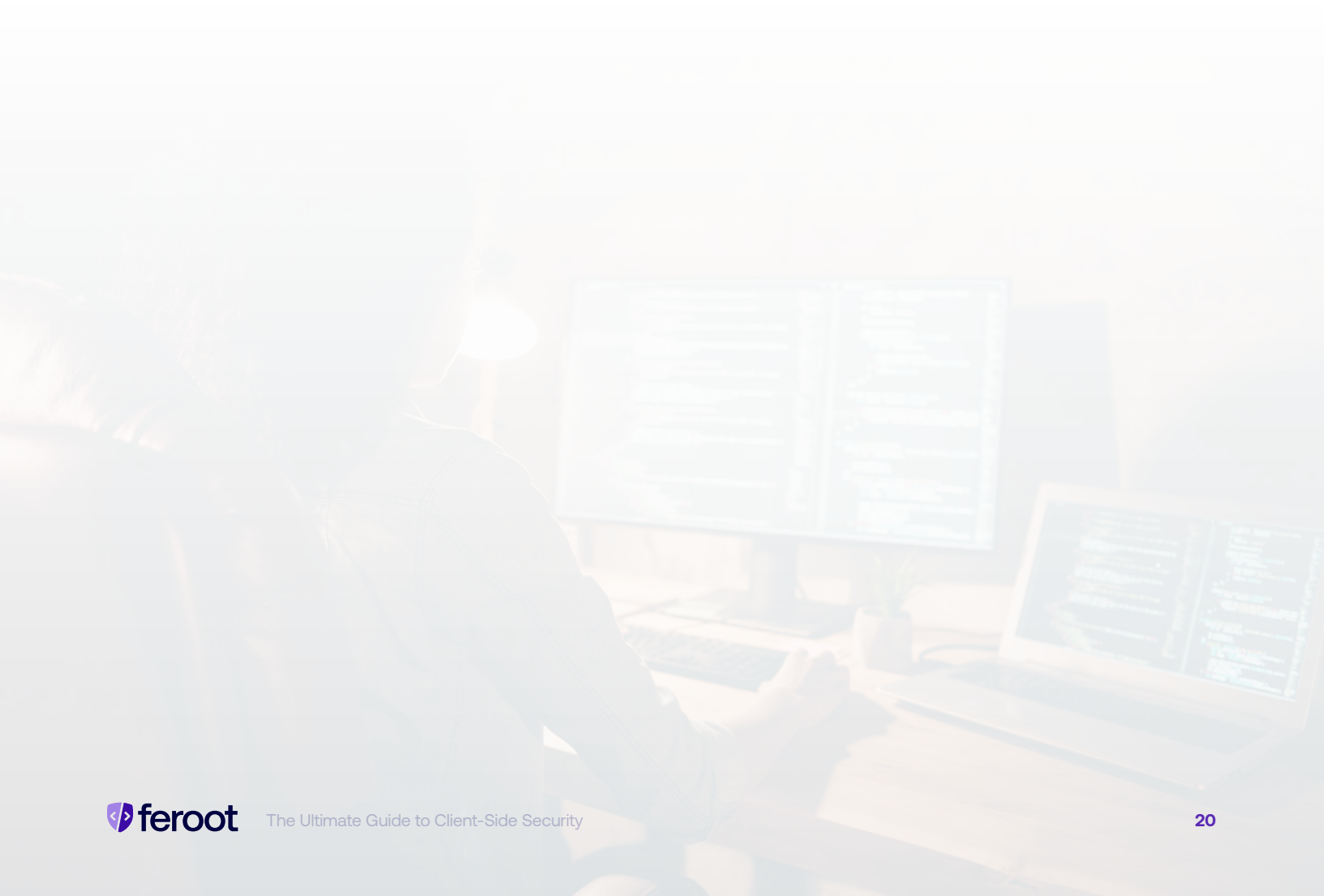
Sideloading and chainloading are threat actor attack techniques that allow them to load malicious or infected JavaScript code onto a target webpage using legitimate scripts and tools.

JavaScript Sniffing

A JavaScript sniffer is a form of malware that is designed to steal financial transaction data from websites, web applications, and online forms when a customer decides to purchase a good or service through an e-commerce website or e-store.

Broken Link Hijacking

Companies that are not careful and leave unused, expired, or invalid links on their websites or web applications put themselves at risk for broken link hijacking. This client-side attack relies on companies forgetting about broken links, which are then hijacked and used to load malware or other types of malicious code into the user's browser. To protect your company from broken link hijacking, ensure that all links that are no longer valid or in use are removed or replaced immediately.



04 Request Forgery

Server-Side Request Forgery

Server-Side Request Forgery (SSRF) is an exploit where a threat actor abuses the functionality of a server causing it to access or manipulate information that would otherwise not be accessible to the hacker. One example of a SSRF attack includes a hacker causing the server to connect to services within the organization's infrastructure. Hackers could also force the server to connect to external systems, in order to exfiltrate data such as authorization credentials like usernames and passwords.

Cross-Site Request Forgery

Cross-Site Request Forgery (CSRF), also known as a one-click attack or session riding, is a type of cyberattack that forces an end user to execute unwanted actions on a web application in which they are currently authenticated. CSRF directly manipulates end-user browsers.

Client-Side Attacks and Threats: Who Is at Risk?

Threat actors love to exploit the obvious, so perhaps a better question is "who is not at risk?" To do business with any consumer, organizations 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 the consumers' decision-making process.

That being said, threat actors also 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, especially if they purchased malware on the dark web and need to recoup that initial investment. Threat actors also try to attack many targets at once in order to hedge their bets; all they need to recoup their initial investment is to breach one business's website.

High Risk

- Financial Services & Banking
- Insurance
- Healthcare & Medical
- E-commerce & Retail
- Travel & Hospitality
- Communication, Social Media, & Content Producers
- Cryptocurrency Exchanges & Blockchain

Medium Risk

- Real Estate
- Technology & Cybersecurity
- Distribution & Transportation
- Education
- Entertainment

Low Risk

- Manufacturing
- Energy
- Distribution & Transportation
- Consulting & Legal Services

Client-Side Kill Chain

JavaScript Malware Attack Defense

Client-side attacks are on the rise. The majority of attacks use some form of JavaScript to deliver malware in order to collect data from unsuspecting users and send it to command and control domains for processing. Client-side attacks need to be managed similarly to more traditional server-side attacks. There are many cyber defense frameworks that can help security analysts defend their business from attacks. The most prominent one is Lockheed Martin's Cyber Kill Chain™.

The following illustrates the Cyber Kill Chain™ to map out and defend against a JavaScript malware attack, let's walk through a quick example.

Reconnaissance

The threat actors research a variety of e-commerce websites to determine to determine the first- and third-party script composition. From their research, they identify that the majority of these websites use a specific open-source, third-party JavaScript on payment pages to enhance the user experience. The threat actors determine that they can build or buy specific JavaScript malware that can skim customer information from the page at the point of purchase and automatically send that information to their command and control domain.

Weaponization

The threat actors hunt for Magecart-like malware available for purchase on the dark web. The malware kit costs $1,000. They decide to acquire the JavaScript malware kit, since it has been proven to work over the past year for similar third-party scripts. After acquiring the malware, they find the correct open-source script on GitHub to corrupt.

Delivery

The threat actors corrupt the open-source, third-party JavaScript code in the GitHub library where it is openly shared. Now every website that uses this third-party script is part of a drive-by web skimming attack. In essence, the skimming malware was delivered by the web application owner themselves via the infected GitHub JavaScript code, not by the threat actors.

Exploitation

The threat actors have now introduced a severe vulnerability into the web applications of multiple e-commerce businesses. Server-side security technologies are unable to find the malware, because they are not designed to do so. The threat actors sit back and wait for the target e-commerce webpages to refresh, so that exploitation, aka data skimming, can begin.

Installation

As soon as the e-commerce web applications refresh, the malicious JavaScript code is loaded in the user's browser. This is outside of the security team's oversight. The JavaScript malware is now able to receive commands from the threat actors' command and control server.

Command and control

The threat actors can now execute commands from their command and control server to start collecting data. The threat actors can also monitor all web applications infected by their third-party JavaScript code and can adjust the malware to provide them with as much value as possible.

Action on objectives

The threat actors' JavaScript malware campaign goes unnoticed for three months. The third-party script they targeted only gets updated quarterly, so they have plenty of time to collect credit card information. Over the span of three months the threat actors accumulate 2,000 credit card records. They decide to sell them on the dark web for $200 each, netting them a profit of $399,000.

Not every client-side attack follows the Cyber Kill Chain™ model this closely, but having a rough understanding of how threat actors execute client-side attacks can be very helpful.

05

JavaScript Security Approaches & Technologies

Protecting client-side web applications and websites is a goal that straddles both the application development and cybersecurity industries. Bugs and vulnerabilities in web applications represent a significant portion of the most common attack paths. However, there remains a bit of a struggle as to who is responsible for protecting the client side. Is it the security team or is it the application development team? While application security is shifting further to the left, both teams need to improve collaboration throughout the software development lifecycle to better integrate security earlier in the process and keep the client side safe from e-skimming, Magecart, formjacking, and other attacks.

To protect their customers from client-side attacks, businesses have seven primary tools at their disposal:

- Web application firewalls (WAF)
- Content security policy (CSP)
- Penetration testing and assessments (vulnerability and security)
- Client-side vulnerability scanning
- Code scramblers and obfuscators
- Client-side attack surface monitoring
- JavaScript security permissions

Web Application Firewalls

A WAF helps businesses protect their web applications by filtering and monitoring HTTP traffic between the application and the internet. It protects web applications from attacks, such as cross-site forgery, cross-site-scripting (XSS), and SQL injection. WAFs are deployed in front of web applications and analyze bidirectional, web-based (HTTP) traffic, detecting and blocking anything malicious. They are great tools to use when protecting your business from skimming attacks, but they can only do so much.

What WAF Limitations Do I Need to Be Aware of?

WAFs are a special category of firewall that are designed specifically to protect web applications and your business from falling victim to 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 sits between a web application and the client endpoint and serves as a security policy enforcement point.

It is important to note, however, that 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 sideloading and chainloading attacks.

WAFs cannot protect businesses or their customers from:

- **Sophisticated Skimming Malware**—WAFs are not able to detect and protect businesses from more advanced skimming malware.
- **Drive-by Skimming and Supply Chain Attacks**—WAFs are unable to detect manipulated JavaScript code or data exfiltration.
- **Sideloading and Chainloading Attacks**—WAFs do not protect against skimming performed by a side-loaded JavaScript code.

Are WAFs Right for Me?

Absolutely. WAFs are great security tools to help protect your business from client-side attacks, however, they can only block some client-side threats. They cannot block all of them. As with everything in security, there is no silver bullet to protect your business and your customers from all cyber threats. A WAF will protect the connection between your servers and your customers, but the protection ends there. They can't monitor or protect your business from browser-level threats outside of your security perimeter.

Content Security Policy

A Content Security Policy (CSP) is an added layer of security that helps businesses and security teams detect and mitigate certain types of client-side attacks. A CSP can help uncover cross-site scripting (XSS), JavaScript code injection, and data skimming attacks.

What CSP Limitations Do I Need to Be Aware of?

First, it's not easy to add CSP to an existing website. Most websites and web apps are assembled using third- or fourth-party JavaScript libraries and code. Since web browsers load these scripts from external website domains or subdomains (e.g., code[.]company[.]com) developers end up whitelisting all external and internal hosts of scripts to avoid breaking the code's functionality.

What this means is that application development and security teams face a tradeoff between security and functionality of their websites. Whitelisting removes or reduces the very protection CSP is supposed to provide, thereby making the value of CSP inherently questionable. The complexity involved in building and maintaining CSP makes it easier for threat actors to find holes within those policies in order to steal data, distribute malware, or deface websites.

There are four main weaknesses in CSP that can expose you to e-skimming breaches:

- Excessive "allow list" rules or whitelisting
- CSP bypass techniques
- Incorrect CSP implementation
- CSP implementation tradeoffs

Is CSP Right for Me?

Sure, that is, if you have the time and energy to deploy and manage CSP on a continuous basis. CSP is a great way to launch a pseudo-Zero Trust approach to protecting your web assets, but, as described earlier, CSP comes with significant risks. CSP is also extremely vulnerable to cross-site scripting attacks. For example, even when CSP is applied, scriptless or post-XSS attacks can still be executed, some XSS vulnerabilities simply aren't mitigated, and some browsers do not support CSP at all. So, while CSP can add a level of increased client-side protection, it will only stop a limited number of client-side threats.

Client-Side Pentesting, Vulnerability, and Security Assessments

Penetration Testing

A penetration test, more commonly referred to as a "pentest," is a deliberate cybersecurity attack, conducted with permission from the organization by professional cybersecurity experts and designed to uncover weaknesses and vulnerabilities across an organization's security controls. Companies either use internal red teams to carry out these attacks or hire an external security service provider that specializes in penetration testing. During the pentest, red teams attempt to enumerate and infiltrate their target's digital infrastructure, networks, and endpoints. Once vulnerabilities have been identified, pentesters try to mimic threat actor tactics, techniques, and procedures (TTPs) to forage deeper into their target's systems and networks. The final output of the pentest is a report that outlines what security gaps exist and what needs to be addressed to secure the business from cyber threats.

Vulnerability Assessments

A vulnerability assessment is a systematic analysis and review of security weaknesses in a technology, system, application, or network. During these assessments a security analyst will determine if the system is susceptible to any known or exploitable vulnerabilities, assign severity levels to them, recommend remediation or mitigation, and prioritize the order in which remediation must occur based on the severity level.

Security Assessments

Unlike pentesting and vulnerability assessments, which are focused on the tools and technology, security assessments are more concerned with process, governance, and compliance. Essentially, it is an evaluation of your tools, applications, websites, and technologies, and how they are used to determine if they are secure from cyber risks and threats. Security assessments identify and evaluate if your business has the proper security policies and controls in place across all of your assets inside and outside of your security perimeter. The end result of a security assessment should be deep insights into the security gaps of your organization, aligned to both your overall security program and a governance model (e.g. NIST). The undertones of the report should also provide a risk level of your organization in its current state. Generally speaking, security assessments are a core piece of any organization's risk management process.

Security assessments can be performed by consultants or internal teams. It depends on how mature the organization's processes and security teams are. Consultants and security analysts who perform assessments conduct in-depth audits of the organization's defensive measures against various attack methods used by threat actors, including internal staff (insider threat and human error) or external actors (hackers trying to breach the business via client-side or server-side attacks).

What Is Tested During a Client-Side Security Assessment?

Client-side security assessments are actually quite uncommon at this point in time. Unfortunately, this lack of client-side assessments presents a huge problem, given the dramatic rise in client-side attacks like cross-site scripting, formjacking, and Magecart. With the increased use of front-end frameworks, libraries, and third-party tools, it's time for organizations to expand the scope of traditional security assessments and testing to include the client-side attack surface of their websites and web applications.

Client-side security assessments are tedious if done manually. **There are five categories of questions a consultant or security analyst needs to answer to uncover potential client-side issues and their associated risks:**

01

What Client-Side Assets Do We Have?

If you don't know what you have, you can't protect it. The first step in a security assessment is to inventory all webpages, web applications, landing pages, forms, payment forms, marketing trackers, and other client-side assets that might pose a risk to the business if corrupted.

02

What Technologies Do We Use? What First- and Third-Party Code are We Using? What Does Our JavaScript Supply Chain Look Like

This is critical. Websites today are assembled in real time using a variety of protocols, connections, and data sources. Businesses must have an inventory of all webpage and web application components. Assessors need to have a full picture of all their scripts, where they are loaded, how they are loaded, and how they interact with other JavaScript code across the client side. The easiest way for threat actors to steal protected data is by corrupting third-party JavaScript. Without continuous testing of your client side, you might never see that they have breached your JavaScript supply chain and are stealing customer information.

03

Who Has Access to Our Data (in Real Time)?

Once you have a list of your assets and the associated technologies, it's time to start looking into who has access to them (and what type of access). Are third parties reading all of your customer data during every form submission? How are you protecting our user's privacy? Being able to shape data access insights across the client side is the next big step in protection.

04 Are We in the Midst of an Attack Right Now?

Once you have a full inventory of your client-side pages, applications, and the code you use, it's time to see if your client-side web assets are only doing what you want them to be doing... ahem... that is, is the data you are collecting only being collected by you or is it being sent to a threat actor's command and control domain in Uzbekistan? You want to look at your keyloggers, your WebSockets, any anomalous behaviors, and if there is any data transfer to unauthorized countries or servers.

05 What Needs to Be Fixed Now?

Once the security assessor has inventoried your client-side assets and the code used to build and maintain them, and any potential breaches and exploited vulnerabilities have been uncovered, the assessor should provide a detailed report on what the organization's security team should do to secure the business. Client-side security assessments should point out:

Security configuration gaps:

- Current access controls: This identifies who currently has access to what and how to limit access to ensure only authorized individuals can modify or utilize client-side assets.
- Overly permissive access: Clear recommendations on how to deploy a Zero Trust approach to client-side web applications and websites to reduce the risk of tampering. This helps ensure who has full access, read only access, and data transfer access.

Malicious elements:

- Malicious host scripts: Are any malicious hosts actively stealing data? What can be done to fix this issue?
- Malicious scripts: Is the business currently using any first- or third-party script that has been corrupted and is exfiltrating data or modifying the webpage or application in any way? What can be done to fix this issue?

Vulnerabilities:

- Exploited vulnerabilities: Are there any known vulnerabilities currently being exploited? Is there a patch available to fix these vulnerabilities and which ones are the most critical to patch?
- Other vulnerabilities: Are there any known vulnerabilities that we can patch proactively to reduce client-side cyber risk? Is a patch available and how critical is it to patch the vulnerability now?

What Pentesting, Vulnerability Assessment and Security Assessment Limitations Exist?

Typically, pentests, vulnerability assessments, and security assessments are performed as short-term projects that are repeated on a quarterly or annual basis. Finding good pentesters is hard and they demand a high wage because of the specialized skill set and experience they possess. Many organizations hire a managed security service provider (MSSP) to conduct the pentest.

Let's assume that a penetration test or assessment is 100% accurate and provides actionable results. That's great. However, results are a snapshot in time, which means hackers have the ability to execute attacks between quarterly or annual assessments. Additionally, 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. Ultimately, threats and threat actors can move much faster than any company.

Penetration tests and assessments also present limitations because they:

- Are time and resource intensive.
- Are limited in scope to certain applications, technologies, and networks.
- Require a skilled consultant, tester, or employee with the know-how to be successful.
- Rely on the use of specialized tools and technologies to uncover vulnerabilities and threats.

Are Pentests, Vulnerability Assessments and Security Assessments Right for Me?

Yes! Yes they are! They are a necessary aspect of any cybersecurity program. But keep in mind that they are not continuous. The information gleaned during a pentest or assessment represents only those issues that exist at that moment, and there might be a laundry list of new vulnerabilities and issues that a different pentest or assessment will uncover in a week or a month. Threat actors move faster than any government or business. To stay ahead of the threat, you need more than a period-in-time pentest or vulnerability assessment.

Client-Side Vulnerability Scanning

Software Vulnerabilities

Simply put, a software vulnerability is an error or defect in software. Threat actors and hackers love to scan networks, systems, applications, and software to find vulnerabilities that could be leveraged to gain control of the system or software. Vulnerabilities stem from the way the software is designed. For example, there could be a flaw in the way the application or software was coded, errors could have appeared in a software update or release unexpectedly, or code errors could have been injected inadvertently when first- or third-party code was added to the software or application.

Why Is JavaScript Vulnerable?

As discussed earlier in this e-book, JavaScript, like many other coding languages, wasn't designed with security in mind. Being able to protect JavaScript is something that we are now having to reckon with given the explosion of usage in all modern digital applications.

JavaScript vulnerabilities have become a favorite for threat actors given the ease with which it can be exploited and leveraged for more advanced attacks like digital skimming.

What Is Vulnerability Management?

Vulnerability management is the continuous process of identifying, analyzing, prioritizing, and remediating weak points in an organization's cybersecurity posture. Vulnerabilities include potential exposures or risks that need to be mitigated to ensure threat actors cannot exploit them to access the network for malicious purposes.

What Can Vulnerability Scanners Do?

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.

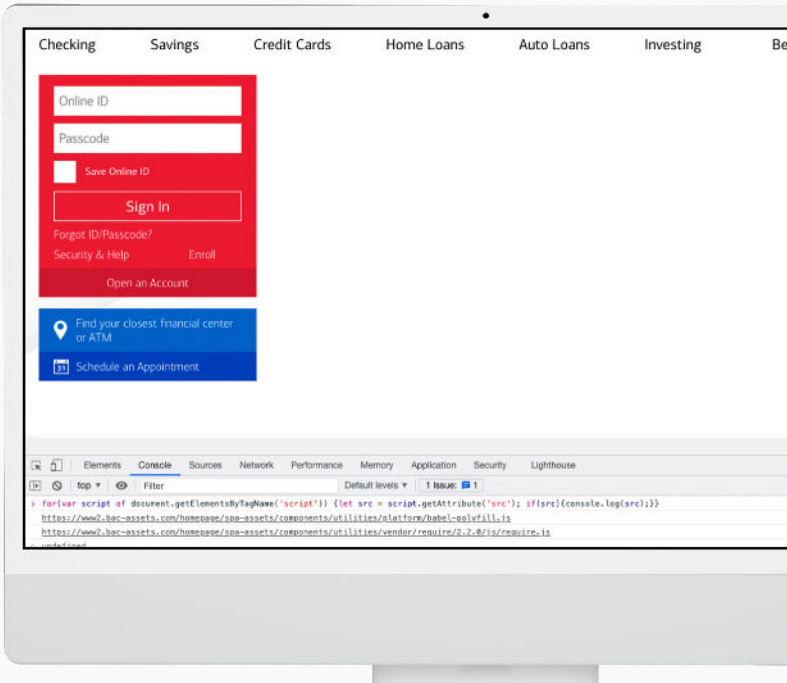
Cybersecurity teams deploy vulnerability scanners to find potential inroads hackers could use to breach their networks and defenses. Once a vulnerability has been found, vulnerability management and security teams then patch the vulnerabilities with software updates. If vendor software updates are not 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 entrypoint.

What Limitations Do Vulnerability Scanners Have When Used for Assessing Client-Side JavaScript?

Vulnerability scanners are designed to scan back-end code and systems, typically those digital assets that live on the server side. They aren't able to detect and enumerate all JavaScript scripts and vulnerabilities. To be blunt, they just can't see them. Vulnerability scanners can only see the client-side after it's been assembled together, not in real time. Vulnerability scanners also can only see a single domain, not all of the links that are part of it.

What Are Typical Vulnerability Scanners Able to See and Scan?

Traditional vulnerability scanners are more closely related to bots than they are to humans. As such, there are several human-like actions they cannot replicate, therefore they do not detect client-side threats in their entirety. In this example, only a few JavaScript items are detected. The screenshot below shows this in a very simple way. A vulnerability scanner will only pick up a handful of active scripts.



Client-side security technologies are able to pick up

50+

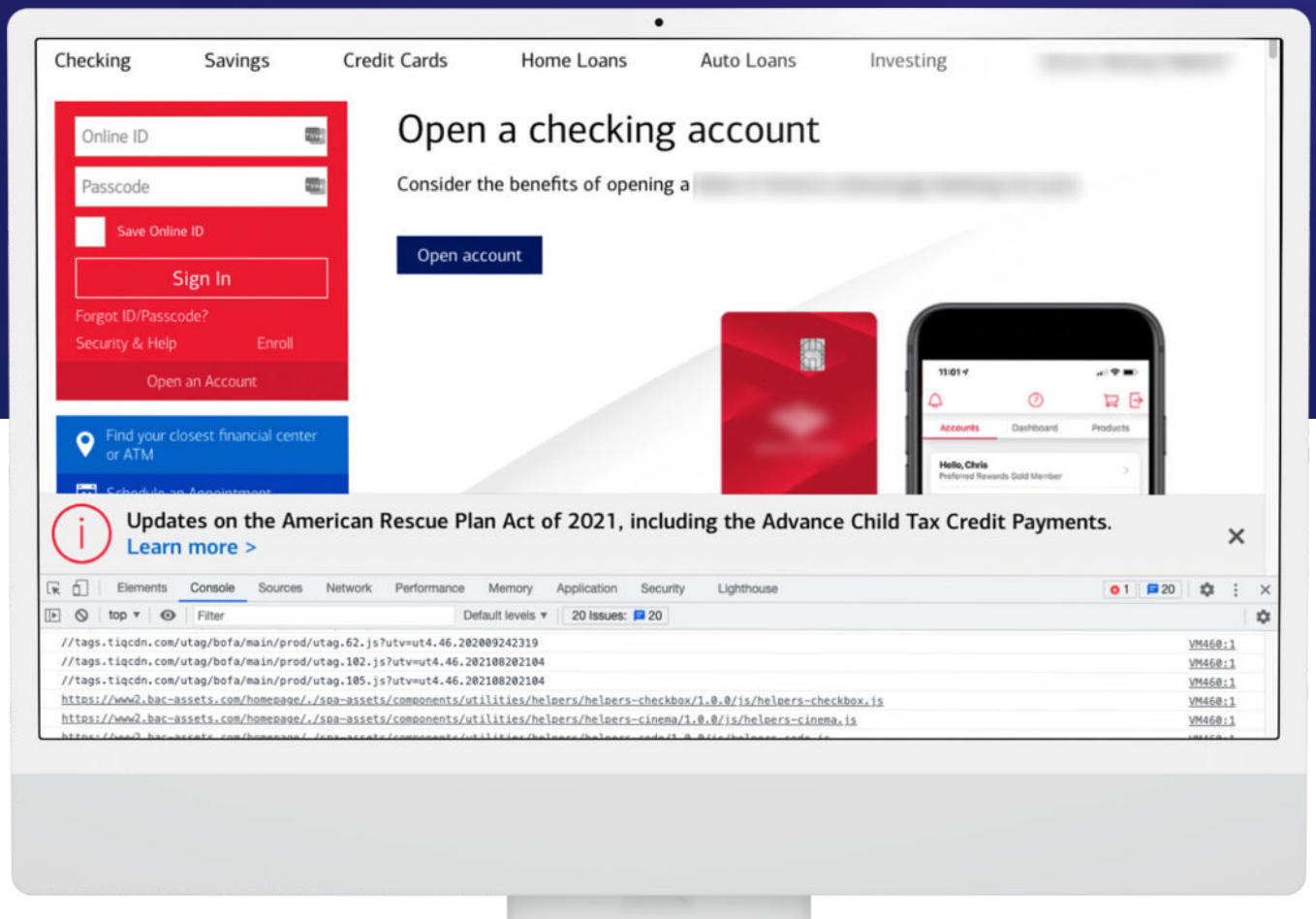
active scripts.

What Are Client-Side Security Technologies Able to See and Scan?

Client-side security technologies replicate actual user behavior on a webpage, including the ability to execute custom user journey scenarios. Needless to say, there is a massive detection gap that leaves businesses vulnerable to client-side cyberattacks.

Are Vulnerability Scanners Useful for Client-Side Security?

Vulnerability scanners are necessary for any cybersecurity program. However, they are not useful for client-side security. Vulnerability scanners are not designed to support client-side security efforts. There are client-side security technologies specifically built to scan JavaScript web applications and websites. They see all scripts, network requests, and resources in each scan. Since JavaScript is so easy to manipulate, threat actors can move very fast when hijacking them for malicious purposes. This type of technology continuously scans client-side web applications and websites, and alerts on issues immediately. Furthermore, a new product category recently hit the market which deploys JavaScript security permissions, so that hackers can't exfiltrate data. Security teams can rest easy that their client-side applications are protected, and they can fix vulnerabilities and issues when they are ready and not under duress.



Code Scramblers and Obfuscation Technologies

What Are Code Scramblers and Obfuscators?

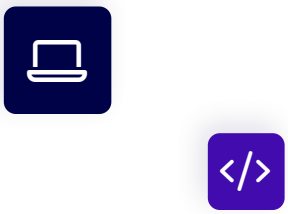
Code scrambling or code obfuscation is a process by which easy-to-read code is distorted to make it difficult to comprehend. The goal of code scrambling and obfuscation is to make it difficult for competitors, other developers, and threat actors to reverse engineer or modify it. Code obfuscators allow 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.

What Limitations Do Code Scramblers and Obfuscators Have?

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.

Are Code Scramblers and Obfuscators Right for Me?

Well, it depends. If you have a simple, single-page web application that doesn't collect user data and security isn't an issue, then, sure, code obfuscation might be a viable option for you. However, if you have a sophisticated web application or webpage that you use to interact with your customers, you cannot risk not being able to see vulnerabilities in your code. If your web application is using third-party scripts, and those are attacked in a drive-by web skimming attack, you will not be able to see the malicious code. A skimming attack could run on your web application in perpetuity. If your web application collects payment information, or if you conduct business in Europe where GDPR violations are a clear and present danger, you are leaving your business open to tremendous fines, legal action, and troves of upset customers



Feroot Security Inspector

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

What Limitations Do I Need to Be Aware of?

The limitations are minimal for this client-side security approach. 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 e-book. 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 in this e-book. With strong collaboration between security and application development teams, businesses can secure their client-side web applications with ease.

Are Client-Side Attack Surface Monitoring Solutions Right for Me?

If your business interacts with customers via web applications or webpages, then yes, client-side attack surface monitoring solutions will enable your business to stay ahead of client-side cyber threats. Client-side attack surface monitoring solutions condense manual processes that take security analysts and web application developers days into minutes. With automated alerts and detailed issue enumeration, these technologies can enable security teams to automate client-side security tasks beyond any scope available with other client-side security approaches.

Feroot Security
PageGuard

Client-Side JavaScript Security Permissions

I know what you're thinking. There is no such thing as JavaScript security permissions!!! Hold up! There is a new product category to address the growing client-side security gap.

What Are Security Permissions?

Permissions are a way for application developers and security analysts to control access to a specific system and device-level functions in an application, page, or other software. Traditional applications, that is, those not written in JavaScript, generally come with a menu of options or functions that may be made visible or hidden from a user based on their permission level. Most permissions must be granted at runtime by the user. The user has the right to revoke permissions at any time.

Types of permissions include the applications ability to:

- Access features on the user's machine, such as their camera or mic.
- Review and collect personal data, such as personally identifiable information, data entered into forms, IP address, and location data.
- Grant rights to modify the functionality of the application or software.

What Are JavaScript Security Permissions?

Traditional software and applications, a.k.a. those not written in JavaScript, come with a menu or functions to set user permissions. JavaScript is the wild, wild west. 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. So do JavaScript security permissions exist? Yes, they do.

JavaScript security permissioning technologies 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, then these solutions automatically apply security configurations and permissions for continuous protection from malicious client-side activities and third-party scripts. These products integrate directly into the runtime environment of every user browser session to enable proactive monitoring and defense.

They essentially deploy a 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 applications.

In short, JavaScript security permissioning solutions monitor and respond to browser-level security events in real time by auto-instrumenting itself on every website and by applying security configurations to every user browser session.
I assure you, it's not too good to be true.

What JavaScript Security Permission Limitations Do I Need to Be Aware of?

There are none. 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.

Are JavaScript Security Permissions Right for Me?

If you work for a company that uses marketing landing pages, e-commerce technologies, user portals, and other technologies to communicate and collaborate directly with your customers, then **YES, you do need to add JavaScript security permissions!**

06

Operationalizing Client-Side Security

Most security organizations struggle with finding recommendations on how to build a client-side security program. There's a good reason for that. To date, cybersecurity has been focused on mapping out the attack surface and managing security operations solely from the businesses or server-side perspective. Client-side security that is protecting customers has been an afterthought or a tedious task at the best of times.

As an industry, we need to focus not only on “who is attacking me,” but also on “who is attacking my customers.”

Cyber threat actors have found that it is relatively easy to exploit constantly changing JavaScript code. Client-side security requires a significant amount of manual work to ensure the security of web applications and websites and the data that they capture, share, and store. The client side is a fundamentally different attack surface from any server side technology. To successfully deploy and manage a client side security, security teams need to adopt a completely new point of view, approach, and skill set. For many security teams, the concept of client-side security is still relatively foreign.

Let’s walk through some of the basic components of how to operationalize client-side security.

Client-Side Security Operationalization Basics

Inventory Management

Key Action: Map your client-side attack surface to make an inventory of your web assets.

First and foremost, you need to know what web applications and websites your business maintains today. If you don’t know what needs to be protected, you obviously can’t protect it. The discovery and ‘baselining’ of all IT systems and processes is critical to any cybersecurity program. The same holds true for client-side security programs. Start by enumerating all of your websites, individual webpages, web applications, and other web assets. Inventory your client-side attack surface from soup to nuts.

Data Discovery

Key Action: Discover client-side architectural elements and the data they use.

Look at all of your web assets from the hacker's perspective. Seriously, do what the bad guys might do to your websites and web applications. The first step in the Lockheed Martin Kill Chain™ is reconnaissance. The first thing hackers do is examine your websites and web applications like your customers would. Then they start to investigate your scripts. Next they start to figure out your weaknesses. So, just like a hacker, you need to answer the following questions:

- What data is being exchanged via client-side assets?
- What sensitive data do my websites and web applications collect?
- What sensitive data might be exposed inadvertently?
- What application elements are touching our data (i.e., scripts and libraries)?
- What third- or fourth-party elements have access to data that is being exchanged on the client side?
- Where is the sensitive data we collect going, and do we have control over it?

Script Management

Key Action: Catalog all scripts on your web assets and the data accessible by these scripts. Quarantine zombie scripts immediately.

Next you need to determine what scripts have access to sensitive data. Follow the Zero Trust model on your client side just like you do on the server side. If the script doesn't need to have access to data, don't let it have access to data. Simple as that. Moreover, if your web assets are running scripts that are not actually being used, a.k.a. 'zombie scripts', then ask yourself if these scripts really need to have access to your data? Hackers love zombie scripts. They are the easiest backdoors to exploit. If a hacker is currently evaluating your business as a target and they find a zombie script, they'll start rubbing their hands together and say: "Hmm...this script is inactive and it has access to sensitive data? Gold mine! They'll never know I was using it to steal their customer data!" Therefore, when a script becomes a 'zombie,' it should immediately be quarantined.

You have to have a full inventory of all the scripts running on your web assets and know what their access to data might look like. To properly evaluate your web assets and the scripts that built them, you will have to do static code analysis, dynamic code scanning, and manual review of front-end code. It's the only way you will be able to keep hackers at bay. No two scripts are the same, so while conducting code analysis and review, please keep in mind that it is difficult to:

- Find all zombie scripts. You will have to hunt for them on a regular basis.
- Gain total visibility into which scripts are or are not in use.
- Establish visibility into security control and configuration variations.
- Block all access to sensitive data on the client side.

Change Tracking

Key Action: Continuously analyze your web assets for code changes, and deploy applicable security measures.

Here is where client-side security becomes extremely tedious and time consuming. Today, many websites and web applications aren't built by a team of developers line by line. They are assembled like a sophisticated jigsaw puzzle. Chunks of pre-written code with varying functionality, built by multiple developers with varying capabilities, are pieced together to create the final product. Modern web applications load an average of over 20 third- and fourth-party scripts as part of the user experience.

If your web assets are assembled using third- or fourth-party scripts, then you can be sure that the scripts are changing all the time. Staying on top of this change and ensuring continuous client-side security is a full-time job. Security teams often lose the client-side security battle due to how applications are developed and enhanced over time. On top of that, front-end code changes for every user session. So, how can someone review all of those code and script changes and keep track of zombie scripts, benign scripts, vulnerabilities and exploits, and data exposure?

For now most organizations deploy quarterly penetration tests (pentests) and vulnerability assessments to stay on top of their client-side code security. However, these tests and assessments are snapshots in time. Application developers move much, much faster and businesses leave their client-side attack surface wide open to breaches for extended periods of time.

07

Client-Side Threat Detection & Prevention

Client-Side Attack Detection

The first step in detecting client-side attacks is having a full inventory of all your web assets. This includes websites, marketing forms, payment portals, web applications, etc. Your cybersecurity team needs to know what their client-side attack surface looks like, so that they know where threat actors might attack your digital assets and your customers.

The next step in the client-side security process is to determine what scripts are running across your website and web applications as well as what they have access to. To detect client-side attacks, businesses need to be 100% aware of what data is being collected or exchanged via the client side, and what application elements or third-party elements have access to or are touching your data (i.e., scripts and libraries). Once these basics have been covered, security teams must follow the five steps below on a continuous basis to detect client-side attacks.

1. Review code and security control configurations to identify potential vulnerabilities and misconfigurations.
2. Perform security and vulnerability assessments of all scripts and code elements that your websites or web applications load into the browser.
3. Test web applications for vulnerabilities using assessment tools. You may or may not find a vulnerability that must be addressed.
4. Protect your website by using control-integrity monitoring and change-detection automation systems.
5. Implement client-side intrusion detection to detect runtime and browser-level intrusions.

Client-Side Attack Prevention

The best prevention and defense strategy for client-side attacks is a layered security strategy, also commonly referred to as defense in depth. Businesses need to deploy and nurture a multi-layered defense against web skimming attacks in order to prevent or significantly minimize the impact of client-side cyber threats.

1. Harden and tamper-proof the client side of your web applications.
2. Grant data access only to those websites and web applications which absolutely require access to that data.
3. Carefully restrict access to prevent all unsanctioned scripts and JavaScript libraries from accessing data, to ensure unauthorized access of sensitive data is contained at the browser level.
4. Continuously analyze all scripts from the client side to detect unauthorized activities.
5. Deploy vulnerability and malware monitoring technologies and processes on the client side of your web applications.
6. Implement client-side intrusion prevention policies and procedures to prevent runtime, browser-level intrusions in real time.

08

Client-Side Security Teams and Collaboration

Cybersecurity is a team sport. It doesn't matter if you are detecting and defending your business from threats, you need a great team that has your back to reduce cyber risk and secure your mission critical web applications. In client-side security, there are four core teams which security professionals need to collaborate closely with to avoid costly Magecart-like attacks or other client-side breaches.

Web Application Development Teams

Cybersecurity professionals must build strong relationships with their application development teams in order to keep web applications secure. Understanding how the applications are built, deployed, and configured will help with the testing and security of those applications in production environments.

Goal Alignment

First things first. Web application developers are constantly under pressure to deliver new features, fix bugs, and innovate faster...all of which goes against the grain of the cybersecurity professional's role in the business, which is to make sure the web application and the code it is built on is as secure as possible.

Once a web application has reached the point where it can be sent to a security team for testing, the developers have spent weeks, months, or even years pouring over every little detail in it. It is only natural that web developers are emotionally invested in their work. Developers are also excited to move on to the next project to again build something amazing and learn new skills. Once the security team has tested the application and uncovered vulnerabilities in the code, it can be painful for some developers to have to go back and fix the code. It also disrupts their workflows. To go back and fix an existing application means that they have to adjust their sprint schedule and delay release dates of new apps, features, or functionalities. Repairing vulnerabilities can throw off a lot of moving parts in the web app development lifecycle.

Cybersecurity professionals need to understand that many developers may not have the security knowledge to understand what a specific vulnerability means or how to fix it. Security experts need to be patient and collaboratively walk through vulnerabilities and potential solutions to fix them with the developers. It is important to train the developers on security concepts, vulnerability management, and what the outcome of a vulnerable application might be. **Mutual education is key.**

Web Application Development Team Roles

It is uncommon for businesses only to have one front-end developer who is solely charged with designing and developing websites, web applications, or other JavaScript-based applications running from web technologies. Most businesses have a broad team that includes the following roles:

Visual Designer:

This highly creative individual is an artist at heart. Visual designers love figuring out how to give your website or web application the flair it deserves, to ensure your users and customers are visually delighted every time they visit your web assets. They love to find the right fonts, colors, spacing, visuals, graphics, and design elements to make your business look good.

UI/UX Designer:

This individual loves taking on complex challenges. UI/UX designers dig deep into the experience your users have when they use your web application. They make sure that if something can be accomplished in one click, it is in fact accomplished in one click. They will spend hours building wireframes and specifying how your users interact with each piece of your web application.

Front-End Web Developer:

Nothing makes a front-end web developer happier than building an awesome web application that your users love to use. Front-end developers specialize in bringing fantastic visuals and unique digital interactions to life, to ensure the user experience is legendary all day, every day.

Back-End Developer:

Back-end developers live on the server side. They build the backbone of your web applications, often by developing data models, APIs, and integrating middleware to extend application functionality.

How to Build Security and Developer Relationships

Web application development team members are highly trained professionals who take a lot of pride in their work. They have a wealth of unique skills and knowledge that any cybersecurity professional can learn from. When interacting with these fine folks, please keep the following best practices in mind:

Ask questions

Never make assumptions when working with developers. Everything they do and every decision they make is well thought out and meticulously calculated. Before passing judgments on or making demands to enhance the security of their web applications, ask them as many questions as possible to build your understanding first.

Keep security lingo to a minimum

Developers have their own language, just like security professionals. When working with developers, keep the security lingo to a minimum. Remember, developers are lifelong learners at heart. They get excited when you teach them something new. So feel free to ask them if they want to learn more. They might be intrigued. You can build great relationships with them by showing interest in their craft and sharing your craft with them.

Use security issues as team-building and learning exercises

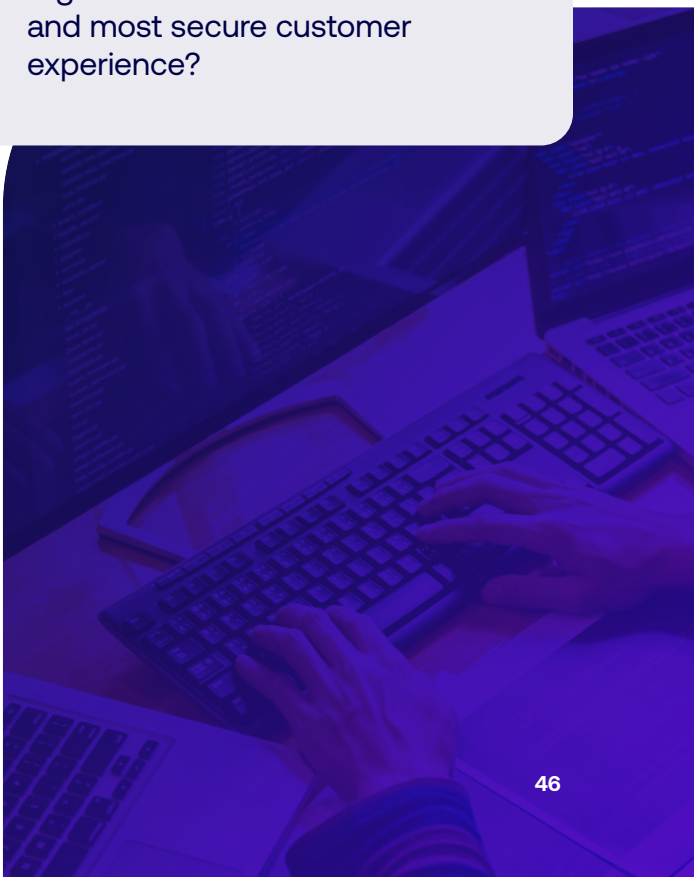
If there is a breach or security issue, don't alienate developers with misplaced frustration. Use the breach to talk through solutions. Developers love problem solving just as much as they love learning. Partner with them during a breach or security incident to learn and grow together. You'll be able to build a strong relationship with them based on mutual trust and understanding.

Understand their pain and empathize with it

Building web applications is a lot of work. As mentioned earlier, developers spend weeks, months, and years building awesome applications. They might interpret security requests or vulnerability reports as a criticism of their work. Take time to explain that you understand their perspective and want to partner with them to make their web application even better, even more robust, and foolproof. Explain to them how they can integrate the learnings into their next web application to get stronger and better.

Great questions to ask include:

- Why did they build the web application the way they did?
- Why did they pick the specific first- and third-party scripts the web app is made up of?
- What was the goal of each deployed/included script?
- Which other first- and third-party scripts did they investigate as alternatives and why didn't they choose them?
- Are there more secure scripts that can replace insecure scripts that offer the same desired outcome?
- How can the security and development teams work together to ensure the best and most secure customer experience?



Marketing Team

There's a natural tension between cybersecurity and marketing teams. Marketing wants to get stuff out, have it be easily accessible, and ensure prospects and customers enjoy the business interaction. Security teams are focused on locking things down to protect the business. In fact, some cybersecurity professionals fear that the marketing team might be the weakest link in the security chain. The waters between marketing and security can be choppy, but they are navigable.

Goal Alignment

Some cybersecurity professionals dislike marketing and see it as a risk and cost center to the business. To them, it might seem that everything the marketing team does on the website is a risk. To those who are unfamiliar with marketing, a lot of what marketers do can seem strange and intimidating. It is the marketing team's job to make sure prospects know the business exists and convince them to interact with the business by reading blogs, downloading content, and sharing their contact information. Collecting a prospect's contact information and safely storing that information is key to building business relationships, conveying product/service value propositions, and, ultimately, generating revenue. Yes, this might open up additional avenues for cyberattacks, but it is a necessary condition to run and grow a successful modern business. After all, this isn't the 1980's and a yellow pages listing no longer works to drum up business.

Reducing website breach risk is crucial, because website attacks are a very real and dangerous thing. Fortunately, with tolerance, understanding, and mutual respect, marketing and cybersecurity can have a fantastic partnership to drive business growth. Cybersecurity professionals need to keep in mind that most marketers are not technical people. They aren't able to differentiate malicious from legitimate code. If they haven't been trained on what phishing emails are, they might click on them to evaluate the effectiveness of the email. Explain security goals, challenges, and best practices with the marketing team. They will listen and learn. You just have to give them the information they need in a way they can understand. **By patiently explaining the cause and effect of insecure digital actions, they will learn and grow, and then adjust their approach to partner with you to secure the business.**

How to Build Security and Marketing Relationships

Let's be clear here, the marketing team is not qualified nor is it able to find vulnerabilities in websites. They are not cybersecurity professionals. They are just trying to make your business look good to generate demand for your services. The worst thing a security professional can do is get upset with marketing or burn bridges with them over a malicious script on the website. Marketers have a completely different, but equally valuable, skill set. Keep this in mind when you collaborate with the marketing team.

Good security teams and marketing teams work together to securely promote the business, remove any friction in the customer journey, and enable each other to be successful. Marketers need care, feeding, understanding, and patience from cybersecurity teams. If you are a security analyst, don't get upset with the marketing team and dismiss them or their profession. They help your business grow so that you get a paycheck and perhaps some new colleagues to reduce your workload. Get to know them and teach them what they need to know to secure the business. Marketers tend to have open minds and love learning from experts. To a marketer, knowledge is power. Share your knowledge with them and they will support you through thick and thin.

Privacy and Compliance Teams

The European Union imposed the General Data Protection Regulation (GDPR) regulation a few years ago. Ever since then cybersecurity and legal teams across the globe have been feeling a tremendous amount of pressure to protect customer data and ensure adherence to data privacy regulations. Every business collects personally identifiable information (PII) on their customers. Depending on the industry, you can add additional types of PII to the security roster, like personal health information (PHI).

In some businesses, the Chief Information Security Officer (CISO) is also responsible for maintaining and supporting data privacy and privacy compliance programs. If you work for such an organization, you're in luck! Your leader will tell you exactly what you need to do to protect your business from regulatory fines. If you're not, you'll have to align yourself with the goals and needs of your privacy and compliance organization.

Goal Alignment

This one is actually pretty easy when compared to working with developers or marketers. To a privacy and compliance professional much of what cybersecurity people do is magic.

Privacy and compliance professionals want to make sure customer data or proprietary information isn't stolen, sold, or leaked. They will ask the cybersecurity team to get it done and will rely on your team's expertise to find the right tools, processes, and technologies to get the job done. They might even throw some budget your way to ensure privacy and compliance projects, tasks, and ongoing processes are completed in a professional manner.

However, and this is where it gets kind of complicated, since privacy and compliance teams will want to see reports from the security team on what has been done to protect user privacy and what protective measures have been put in place to secure the business from the privacy perspective. Before you take on a new privacy and data security project, talk to your privacy and compliance team to outline goals and set expectations. Get their buy in on outputs and outcomes early. Get your ducks in a row to avoid rework and to meet their expectations. Also, don't get frustrated with your privacy and compliance team if a new regulation pops up and you have to change anything. **After all, they have no control over governments, foreign or domestic, and the policies that are enacted.**

Privacy and Compliance Team Roles

Privacy and compliance teams vary greatly. In smaller organizations, your CEO might have outsourced privacy and compliance to a legal services provider. In midsize businesses, you might have a risk manager. In large businesses that are under high regulatory scrutiny, you might have a true team led by a Chief Compliance & Privacy Officer, Data Protection Officer (DPO) as mandated by GDPR in the EU, or General Counsel. This individual is in charge of all things privacy and compliance related at your business. They are responsible for the strategic oversight, guidance, and management of your business's data privacy program. They ensure data privacy awareness and compliance across the business. It's a tough task to ensure everyone at your company is aware of privacy regulations and complies with requirements at every turn. Heads of privacy and compliance work very hard to assess emerging privacy trends and develop response strategies. Often new regulations come out of the blue, so be patient with them as they ask you for support from the cybersecurity perspective.

Product Security Teams

Cyber threat actors have pushed businesses to not only invest mountains of resources into security, but they have also driven business to innovate. Over the past few years cyber criminals have created a new role that is critical for software-as-a-services (SaaS) and other technology businesses. It is now quite common to find directors or managers of product security whose sole responsibility is to make sure any applications that the development team builds and maintains are safe and secure for customers to use. These fine folks protect applications from harmful attacks or unauthorized access. Product security professionals can be found working for the CISO, for the VP of product management, or the VP of development. They participate in engineering projects to identify threats and vulnerabilities, collaborate with the cybersecurity team to define security requirements and security concepts, and work with the engineering team to make sure application security is a critical part of the software development lifecycle (SDLC).

Goal Alignment

Product security staff can be a cybersecurity professional's best friend and are definitely a strong advocate for your mission. They are the folks that will help you secure your business from the inside out, and, if you have web applications, from the outside in. Product security professionals are completely vested in securing your business just like you are, so you're going to love working with them. You can partner with them early in the SDLC to develop your security architecture and build out security requirements for software products and web applications before they go live and increase your cyber risk. Product security professionals will do threat modeling and security analysis of your products and web applications that they will share with you so that you can secure your business together. If you don't have a director or manager of product security at your company yet, this is a new critical role to fill, and we suggest you find one soon. **They will be a key ally in your mission to stay ahead of cyber threats.**

09

How to Recover from a Client-Side Attack

A Client-Side Attack is Discovered: What Should Businesses Do?

A Client-Side Attack is Discovered: What Should Businesses Do?

Step 1

Keep Calm and Carry On

Step 2

Contain the Breach

Step 3

Investigate

Step 4

Shut Down Malware, Malicious Scripts, and Backdoors

Step 5

Identify the Point-of-Attack Origin

Step 6

Recover

When you do notify your customers, they will want to know what information was exposed and how you plan to ensure a breach doesn't happen again. There are a few key items to outline for them if you decide to announce the breach:

Breach Details

- What was taken, broken, or stolen?
- How did the attacker get in, what did they deploy, and what might they do with what they stole?
- How significant was the breach?

Future Protective Measures

- What will your business do to secure your client-side applications in the future?
- What should your customer do to protect themselves moving forward?
- What did your business learn from the breach and how will you do better?
- Is your business still vulnerable to similar breaches?

Three Key Steps to Prepare for Future Attacks

Once the breach has been contained and you're certain your business and customers are no longer at risk, then you need to prepare for future attacks, because in spite of your best efforts, no web application or code is going to be perfect, and the threat of future attacks is always looming in the distance.

Step 1

Learn, Learn, Learn

Step 2

Scan for Vulnerabilities

Step 3

Test Your Defenses

Step 1: Learn, Learn, Learn

If you are like most organizations, a client-side attack is something new and unknown. In order to properly prepare for future attacks, you have to collect as much data, intelligence, and information as possible. Here's a quick list of the types of information your organization should capture:

- An inventory of all your web applications and websites.
- A list of access details and credentials for databases and technologies.
- A list of customer, prospect, or user data collected.
- An inventory of all approved first- and third-party scripts used on your website.
- An inventory of all approved data first- and third-party scripts used on the website.
- Configuration file names and paths.
- A list of applicable log files collected from your website, web applications, software, system files, etc.
- A list of API keys, security tokens, and integration details.
- Vendor contact information (if your website is hosted externally, and if you have any marketing or other technologies integrated with your client-side applications).

Step 2: Scan for Vulnerabilities

Threat actors love to find vulnerabilities in your code so that they can exploit them. When preparing for future attacks, it is important to ascertain whether your website is up to date. Are you using the most secure code libraries? Have you deployed all available patches? Take your time to scan your website or web application and mitigate risk as much as possible. If there are any vulnerabilities without an available patch, track it or find a work around. Make sure you identify:

- Web server misconfigurations
- Old software versions
- Known vulnerabilities in software versions
- Available patches for known vulnerabilities
- Inappropriate access to file and system locations or data

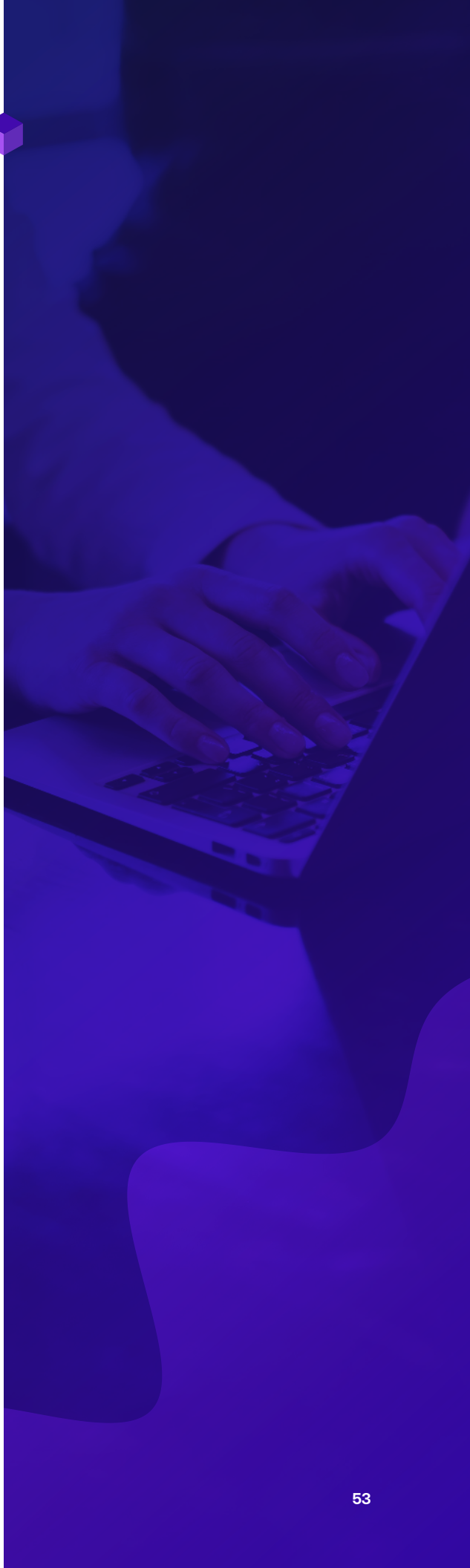
One thing to note, traditional vulnerability scanners are designed to scan back-end code and systems, aka server-side technologies. It is unlikely that they will be able to detect every client-side vulnerability that an industrious threat actor might use against you. To learn more about client-side vulnerability scanning, check out the vulnerability scanning section earlier in this e-book.

Next, patch all vulnerabilities you uncovered immediately. If there isn't an available patch, do some research to find out if there is a way to reduce the risk of vulnerability exploitation. You will likely have to end any unauthorized or foreign processes, remove corrupted or unauthorized files, and, if the threat actor modified any client-side script, replace modified files with approved or sanitized versions.

Step 3: Test Your Defenses

Once you have regained control over your web applications and websites, use the intelligence you have collected to validate and, if necessary, upgrade your defenses. Use the threat actor tactics, techniques, and procedures (TTPs) that successfully breached your website previously to see if the same TTPs still work. Use the client-side cyber threat intelligence (CTI) you have collected and look it up in open-source CTI feeds or in paid feeds like VirusTotal. Learn more about your attacker, see how else they might attack you, and pressure test your website to see if those other TTPs might work.

You may also wish to create a clone of your website to test your defenses. If you are using real TTPs, you don't want to harm your customer-facing website. The goal is not to disrupt your business's ability to grow and interact with your customers.



10

Conclusion

The client-side security problem is becoming a big challenge for many businesses. Organizations struggle to ensure client-side code is secure and to stay ahead of the latest threats targeted at the business and their customers. Unfortunately, traditional security, which focuses primarily on the server side, won't protect businesses or consumers from client-side attacks. In the meantime, attacks like Magecart, web skimming, and formjacking are wreaking havoc, as businesses struggle to get their arms around client-side intelligence, vulnerabilities, and malware.

Client-side threats have generated an extremely large security gap that businesses must be prepared to address if they wish to continue to operate on the internet. The client-side attack surface is unique and requires its own security program to reduce risk. Client-side protection includes understanding the risks specific to the client side, cataloging all code associated with web assets and data, applying principles of least privilege and Zero Trust, and ensuring permissions are set correctly. Comprehensive protection also requires that businesses begin to implement security solutions specially designed to protect from client-side attacks. These solutions are able to analyze client-side applications, web browsers, and other front-end technologies in real time. The technologies continuously scan web assets to detect anomalous behavior and then provide crystallized insights and contextual details to security analysts to enable them to quickly take corrective action. In addition, these products collect client-side telemetry and cyber threat intelligence that help security teams mitigate cyberattacks on web applications and browsers.

It's time for businesses to meet client-side risks and threats with action to enable them to successfully navigate the disruptive forces of cyberattacks.

We are Feroot Security

Businesses come to Feroot to enable proactive client-side security programs. Our data protection capabilities take the pain and ambiguity out of client-side security threat analysis, detection, response, and prevention.

Our products help organizations uncover supply chain risks and protect their client-side attack surface.

Ready to Protect Your Client-Side?

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

feroot

About Feroot

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