Feroot

Securing the client side

THE THREAT OF WEB SKIMMING

www.feroot.com

FEROOT SECURITY

THE THREAT OF WEB SKIMMING

Web-based data skimming, also known as e-skimming, online skimming, and Magecart, is a client-side threat that all organizations with web services must be aware of.

These attacks infect websites with malicious code, known as skimmers, sniffers, or JavaScript sniffers that are very difficult to detect. An infected website skims user information from the payment checkout forms, login pages, marketing forms, or anywhere else companies are ingesting personal information without the organization or users being aware of the compromise of their information.

Threat actors target information, including usernames, passwords, payment details, credit card details, billing address, name, email, phone number, and other types of personal, health, and commercial information.

Who is at risk?

Any organization in financial services, banking, e-commerce, healthcare, government, technology, and SaaS industry that offers services on its website that does not have effective client-side security controls in place is potentially vulnerable. More than 19,000 web skimming breaches, including high-profile international organizations, were identified in 2019.

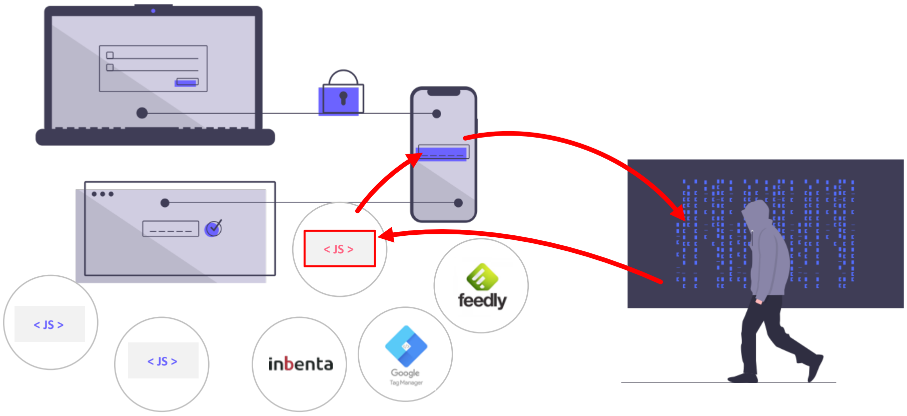
How do client-side attacks work?

These client-side attacks compromise target websites often via a third-party's software libraries that organizations rely upon. Threat actors use methods to inject malicious code onto the client-side code base, which is executed within the browser session that then redirects or copies or otherwise sends exfiltrated information off from that client device to a malicious infrastructure under the attacker's control.

Examples of third-party applications targeted by attackers include live chatbots, customer service functions, advertising scripts, marketing forms, marketing tags, opensource libraries, and various other elements loaded by the client's browser.

Because multiple websites often use third-party software libraries, the compromise of one of these third-party elements with a malicious JavaScript can allow an attacker to compromise many websites simultaneously. This attack is known as a "drive-by web skimming." Additionally, web skimming code with anti-forensic, self-cleaning, and stealth capabilities such as Pipka uses techniques to remove itself from the web page's code after execution, making it exceptionally difficult to detect or trace.

Web-based data skimming is continually evolving and **persistent threat (PT).** According to Willem de Groot's security research report, one in five Magecart's victims are re-infected within days.



What are some of the best practices for detecting client-side skimming threats?

A critical part of client-side inventory discovery and management is to determine what scripts have access to sensitive data that are not being used, often referred to as 'zombie scripts.' Always know what data is going through the client-side of the website, what sensitive data is collected, what application elements and third-party elements have access to the data, or are touching your data (i.e., scripts and libraries):

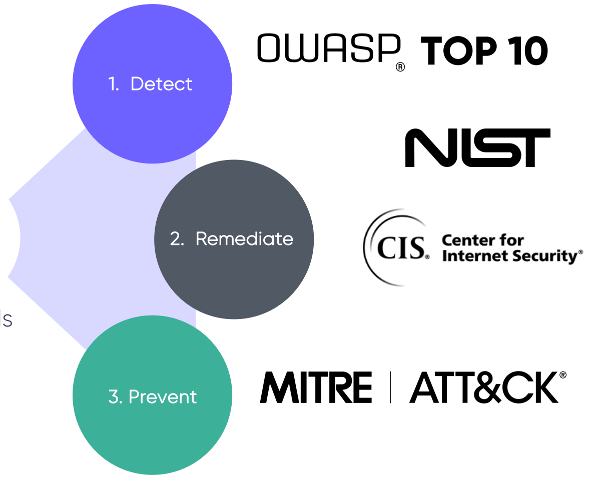
1. Reviewing code and security control configuration to identify potential vulnerabilities and misconfigurations
2. Performing security and vulnerability assessment of all scripts and code elements that the website loads into the browser
3. Use of vulnerability security assessment tools to test web applications for vulnerabilities
4. Use security control-integrity, file-integrity monitoring, and change-detection automation systems.
5. Implement client-side intrusion-detection to detect run-time browser-level intrusions

What are some of the best practices for preventing client-side skimming data breaches?

Combining defense-in-depth and the zero-trust model is the best defense strategy against intelligent web skimming attacks, including Pipka. A multi-layered defense against web skimming attacks, you will be able to prevent or significantly minimize the client-side threats.

1. Hardened and tamper-proof the client-side of your web applications.
2. Restrict access to only what is absolutely required and prevent all legitimate or unsanctioned scripts and JavaScript libraries from unauthorized access of sensitive data at the browser-level.
3. Continuously analyze all scripts from the client-side for presence activities that you didn't authorize.
4. Implement vulnerability and malware monitoring for the client-side of the web application.
5. Implement client-side intrusion-prevention to prevent run-time browser-level intrusions in real-time

- **Identify all data assets** across the client-side risk surface area of your web
- Gain in-depth real-time **visibility** into **risks** within your dynamic client-side code
- **Autonomously** enforce security and GRC controls across your client-side risk surface area



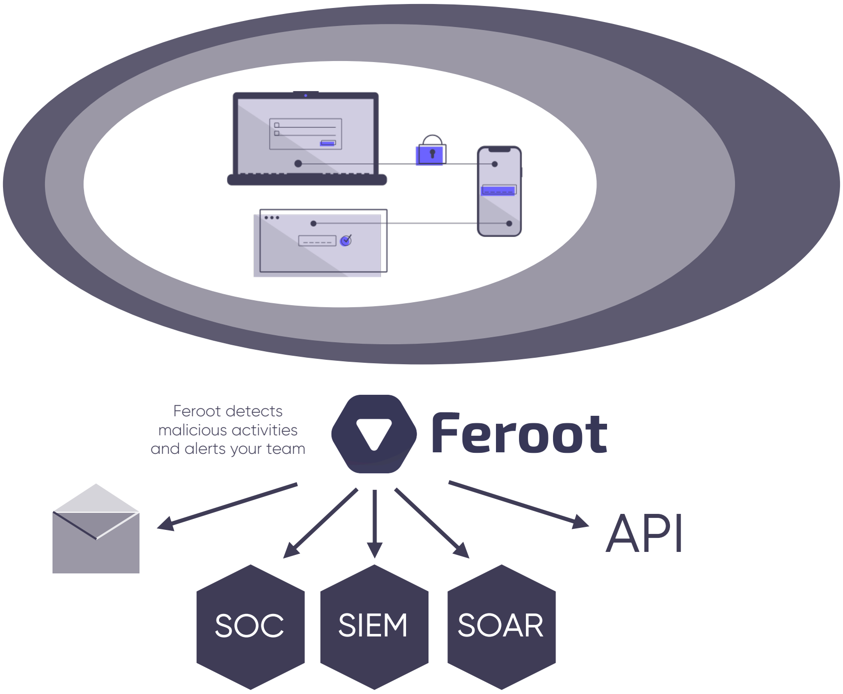
FEROOT SECURITY

About Feroot

Feroot Security Platform specializes in protecting web applications and enterprise websites against client-side threats helping organizations deal with web skimming attacks, data harvesting attacks, and attempts to execute malicious code at the browser level by legitimate and unsanctioned code, and much more. It implements in hours, not months, and it doesn't need a rocket scientist to operate it.

Feroot Inspector is an automated security assessment solution for the client-side of websites. It continuously assesses user journeys paths for exposure to data loss, vulnerabilities, threats from dynamically loaded content and third-party JavaScript libraries, and security misconfigurations that can lead to incidents such as Magecart web skimming.

Feroot PageGuard detects unauthorized scripts files, code behavior and utilizes state-of-the-art defense mechanisms to block unauthorized and unwanted behavior in real-time.



PagerDuty
splunk>
DATADOG
servicenow
Opsgenie
AWS CloudWatch
Slack
Webhooks
sumo Sumo Logic
Atlassian Jira

QUICK BASE
AdRoll
save on foods
ElectricBrain
ADDEPAR

www.feroot.com