

Forum:[Hack Forums](#)

Thread: [Hot XSS Exploit Trends in 2025](#)

vasper - June 23, 2025, 10:19 am

I'm going to dig into XSS exploits... I want to write a little about this...

In 2025, XSS will really boom... DOM Clobbering with a new trick. It's not just an id="window" injection anymore. Hackers are actively using Shadow DOM. By creating a custom element in Shadow Root and rewriting sensitive methods like document.createElement, you can inject malicious JS code directly into the browser's rendering engine. Since this is separated from the standard DOM, classic WAF filters are blind!

The second topic... toxic MutationObservers, which are really a pain in the ass. They can be configured to track any changes in the DOM and inject XSS payloads into dynamic elements in an infinite loop. Imagine a React SPA: if you launch a malicious Observer, you can throw a script into the DOM before the final render and completely destroy the application state. This is a client-side attack, so the server doesn't know at all!

The third point is bypassing CSP via JSONP and old endpoints. Many sites still use sloppy CSP, thinking that unsafe-inline will save them. Hackers exploit JSONP callbacks to load malicious scripts as API responses. For example, script src="/api/jsonp?cb=maliciousCode" can bypass all CSP policies. This method is as easy as pie for legacy applications that still use jQuery 3.x!

Fourth, XSS in WebAssembly! Believe me, some are already manipulating WASM to launch XSS at a lower level. By injecting a malicious WASM module into applications that use WebAssembly for rendering or heavy calculations, you can inject JS code directly into the browser's memory. Since this works at the binary level, antiviruses and WAFs are simply paralyzed. Also, WASM doesn't obey CORS, so it can leak sensitive data to any domain!

Fifth, a cool technique called Event Listener Hijacking. This is really sneaky! Hackers inject a malicious listener into sensitive elements like <form> or <button> and can track any events (clicks, keypresses) and execute their code before the main application. For example, attach a listener to the oninput of a password field... and everything the user types is in your pocket! This technique is often seen in Vue.js applications with sloppy event delegation.

Sixth... abusing postMessage in iframes or workers. Many sites use window.postMessage for communication between frames, but do not check the origin properly. A hacker can inject a malicious iframe and send a fake message to execute JS code in the context of the parent window. This technique is actively mowing down victims in chats or dashboards with Web Workers. There was a case with a well-known platform where a simple postMessage stole a user's session!

And lastly... exploitation of the Fetch API via vulnerable credentials. Hackers use fetch with credentials: include to send malicious requests to sensitive APIs. For example, if an application does not use the SameSite Cookie correctly, an XSS payload can send a fetch to /api/delete-account and... poof! — the user's account is deleted. Since the request comes from the victim's browser, the server thinks it is the user himself.

If you have tested something, throw it in the forum, we will see how to deal with it! ?

Igniter - June 23, 2025, 9:35 pm

Thank you I'm always on the lookout for new methods I have followed xss for a while. New methods involve using a variety of tools. I have utilized some tools for better efficiency.