```html

If you've ever tried to visit certain websites and run into anti-bot or security challenge pages, you may have noticed your privacy browser plugins sometimes cause these pages to malfunction or fail. One example is **JShelter**, a privacy tool with strong anti fingerprinting features. This blog post explains why plugins like JShelter often disable or block modern JavaScript features that a site like *Anubis* depends on — and why this leads to the confusing "site breakage" many users experience.

## Table of Contents



## Why Anti-Bot Pages Exist

NoScript blocking proof of work

Websites like news outlets, finance portals, and ticketing platforms often get targeted by automated bots. These bots can scrape content, submit fake forms, or even launch denial-of-service attacks. This:

- Harms site performance and slows down your experience
- Costs the website money in bandwidth and infrastructure
- Reduces content value to readers and users

To protect themselves, these sites deploy *anti-bot pages*. These usually come as interactive challenges that block suspicious traffic but allow real humans through quickly. The challenge might look like:

- Checking if a visitor can run JavaScript
- Doing simple puzzles or waiting for a timer to complete
- Running a **Proof-of-Work** algorithm

Anti-bot measures are crucial to keep websites fast, secure, and affordable to run. But they rely on your browser running modern JavaScript and exposing certain data — features that privacy plugins sometimes block.

## Proof-of-Work in Plain English

Proof-of-Work (PoW) is a clever technique websites use to prove a visitor is real and willing to spend a small but measurable effort to access content. Think of it like showing your "work" before getting a hall pass in school:

1. You are given a tricky math puzzle.
2. You spend a few seconds solving it.
3. You show the answer to get through.
```

For websites, this effort takes the form of a computer puzzle your browser solves in the background. Bots harvesting data don't want to waste CPU time guessing millions of answers on you, so they give up or slow down.

Why can't we just block bots by IP or behavior?

Bots often use rotating IP addresses, proxies, or mimic human browsing patterns. IP blocks alone aren't enough. PoW ensures the visitor's device does real computational work, which is hard to fake at scale.

The Hashcash Background

Proof-of-Work in web security borrows from **Hashcash**, initially invented as an email spam deterrent. Hashcash requires senders to compute a partial hash collision—meaning a number that, combined with the message, produces a hash with certain zeros at the front. It's like finding a needle in a haystack.

Last month, I was working with a client who thought they could save money but ended up paying more.. Websites use a similar technique under the hood. The visitor's browser runs scripts that quickly:

- Guess numbers
- Generate hash values from those numbers + challenge data
- Check if the hash meets a target difficulty (e.g., starts with 4 zeros)

Once solved, the browser sends the solution back, allowing access.

JavaScript Requirements and Modern Features

To run Proof-of-Work and other anti-bot challenges, sites depend heavily on modern JavaScript APIs and features, including:

Feature Why It's Needed How Privacy Plugins Affect It Typed Arrays (e.g., Uint8Array) Perform fast, low-level operations for hashing May be disabled or mocked, causing calculations to crash or become invalid Web Crypto API Provides secure hashing functions efficiently Blocked or spoofed, breaking hash calculations JavaScript Timing APIs (performance.now()) Measure time elapsed or control challenge timeouts Reported times scrambled or

unavailable — breaks timing logic Typed Arrays and Bitwise Operations Used in hashing and Proof-of-Work algorithms Intercepted or replaced with fake data leading to failed calculations

Many modern privacy plugins aim to reduce browser fingerprinting by disabling or modifying these APIs. While this is good for privacy, it also breaks legitimate use cases security and anti-bot software rely on.

Why JShelter Blocks Features Anubis Needs

JShelter leads in anti fingerprinting technologies and privacy protection in browsers. Its goal is to limit what websites can learn about your device, browser setup, and behavior by restricting or altering APIs often abused to fingerprint and track users.

Unfortunately, sites like Anubis rely on some of these very APIs to:

- Run their Proof-of-Work checks
- Perform integrity verification
- Verify that the browser environment is not spoofed

When JShelter disables these features, for example by:

- Disabling the Web Crypto API or making hashing unpredictable
- Fuzzing timing functions or making them unavailable
- Blocking high-resolution timers and typed arrays

it causes Anubis's Proof-of-Work and anti-bot checks to fail. The site may show an error, infinite loading, or trigger more aggressive blocks.

Simply put: JShelter's *modern js disabled* settings break anti-bot proofs.

Anubis sees inconsistent or missing values it requires to confirm real humans and valid browsers. It cannot complete the Proof-of-Work, so it blocks page access.

Common Site Breakage Reasons & What You Can Do

I'll be honest with you: when you face broken site behavior caused by anti fingerprinting features, here are some common reasons and a quick checklist to help troubleshoot:

Checklist for Diagnosing Site Breakage

1. **Check if JavaScript is fully enabled:** Some plugins selectively disable modern JS or Web Crypto.
2. **Try disabling the privacy plugin temporarily:** See if the anti-bot page loads successfully — if yes, the plugin is likely interfering.
3. **Look for specific "modern js disabled" or "fingerprinting protection" settings in your plugin:** Some allow whitelisting trusted sites.
4. **Clear site cookies and localStorage:** Sometimes anti-bot checks use stored tokens that get out of sync.
5. **Use a stable, updated browser:** Old browsers or privacy-focused forks may lack required APIs by default.

Why "Clear Everything" Without Explanation is a Problem

Some advice online says "just clear your browser data." This can sometimes help but often fails to explain:

- What data affects bot challenges (like cookies, localStorage, or even IndexedDB)
- Why JavaScript environment integrity matters even more than stored data
- That sometimes the problem isn't user data but the plugins actively blocking safe APIs

Instead of guessing, it's better to understand if the root cause is the privacy tool altering JavaScript or a cache/cookie issue.

Final Note on Browser Privacy Tools and Anti-Bot Checks

Balancing privacy and security isn't easy. Privacy tools block tracking and fingerprinting but sometimes unintentionally block protection measures websites use to keep services running securely. If you trust a site like Anubis and want it to work smoothly:

- Consider temporarily reducing anti fingerprinting strictness on that site
- Look for "site exceptions" in your privacy plugin
- Understand that "site breakage" often means the privacy features are doing their job — just being too strict for the site's needs

Hopefully this post clarifies why **JShelter modern JS disabled** modes cause issues by interfering with the core *Proof-of-Work* and hashing features Anubis relies on, and what you can do about it.

Thanks for reading! For the smoothest experience, carefully balance your privacy settings with the sites you trust.

```
```