



# Cross-Site WebSocket Hijacking in Bokeh: CVE-2026-21883

24 January 2026 • Yunus Aydın • 4 min read

During a security review of Bokeh, I found a vulnerability in the WebSocket origin validation logic that allows Cross-Site WebSocket Hijacking (CSWSH). The `match_host` function in Bokeh's server code uses flawed hostname matching that can be bypassed by registering malicious subdomains.

## The Vulnerability

The issue is in `src/bokeh/server/util.py`. The `match_host` function compares hostnames against allowlist patterns using Python's `zip()` function, which stops when the shortest iterable ends. This creates a security flaw.

### Vulnerable Code:

```
def match_host(host: str, pattern: str) -> bool:
    # ...
    host_parts = host.split('.')
    pattern_parts = pattern.split('.')

    if len(pattern_parts) > len(host_parts):
        return False

    for h, p in zip(host_parts, pattern_parts):
        if h == p or p == '*':
            continue
```

```
    else:
        return False
return True
```

The code checks if the pattern is longer than the host, but it doesn't check if the host is longer than the pattern. This means a host that starts with the pattern (but has extra segments) will incorrectly match.

## How the Attack Works

If a Bokeh server has an allowlist like `['dashboard.corp']`, an attacker can:

1. Register a domain like `dashboard.corp.attacker.com`
2. Create a malicious website on that domain
3. Lure a victim to visit the malicious site
4. The malicious site initiates a WebSocket connection to the vulnerable Bokeh server
5. The Origin header (`http://dashboard.corp.attacker.com`) matches the allowlist due to the flawed logic
6. The connection is accepted, and the attacker can interact with the Bokeh server on behalf of the victim

### Example:

```
from bokeh.server.util import match_host

allowed_pattern = "example.com"
malicious_origin = "example.com.evil.com"

if match_host(malicious_origin, allowed_pattern):
    print(f"[VULNERABLE] {malicious_origin} matched {allowed_pattern}")
```

### Output:

```
[VULNERABLE] example.com.evil.com matched example.com
```

The matching happens because:

1. `host` parts: `['example', 'com', 'evil', 'com']`
2. `pattern` parts: `['example', 'com']`
3. `zip` compares `example==example` (OK) and `com==com` (OK)
4. Iteration stops, and the function returns `True`

## Impact

This vulnerability enables Cross-Site WebSocket Hijacking attacks. Once an attacker establishes a WebSocket connection:

- They can access sensitive data from the Bokeh server
- They can modify visualizations
- They can execute actions on behalf of the victim (if the Bokeh app allows such actions via WebSocket messages)

### Scope:

- Only affects deployed Bokeh server instances
- No impact on static HTML output, standalone embedded plots, or Jupyter notebook usage
- Does not bypass authentication hooks if they're in place
- Cannot make private/internal network servers accessible from outside

## The Fix

Bokeh fixed this in version 3.8.2. The fix ensures that host and pattern have the same number of parts before comparison.

## Recommended Fix:

```
if len(pattern_parts) != len(host_parts): # Enforce equal length
    return False
```

This prevents hosts with extra segments from matching patterns.

## Affected Versions

- Bokeh < 3.8.2

## Patched Versions

- Bokeh 3.8.2 and later

## Disclosure Timeline

- **Initial Report:** Reported to Bokeh security team
- **Security Advisory:** [GHSA-793v-589g-574v](#)
- **CVE:** CVE-2026-21883
- **Fix:** Version 3.8.2 released

## Conclusion

Origin validation in WebSocket connections is critical for preventing CSWSH attacks. The flawed hostname matching logic in Bokeh's `match_host` function allowed attackers to bypass allowlist checks by registering malicious subdomains. The fix ensures strict length matching before comparison.

**Key takeaway:** When validating hostnames or origins, always check that both the pattern and the host have the same number of segments. Don't rely on partial matches.

## Related Content

- [SQL Injection Vulnerability: Security Issue in GeoPandas to\\_postgis\(\) Function](#)
- [CVE-2025-66019: LZW Decompression DoS Vulnerability in pypdf](#)

## References

- [Bokeh Security Advisory GHSA-793v-589g-574v](#)
- [CVE-2026-21883](#)
- [CWE-1385: Missing Origin Validation in WebSockets](#)

Share:

 Twitter

 LinkedIn

 Copy

## Related Posts

### If Nobody Reads Code, Why Not Write in Assembly? So Here's Redis in Assembly

Everyone's writing code these days. AI assistants generate functions, classes, entire applications. But here's the thing: nobody's reading it. We're...

14 February 2026

### CVE-2026-22787: Cross-Site Scripting (XSS) Vulnerability in html2pdf.js Library

A Cross-Site Scripting (XSS) vulnerability has been identified in the html2pdf.js library. The vulnerability exists due to unsanitized user input...

17 January 2026

## Game Hacking 101: Memory Manipulation in Mount and Blade Warband

Game hacking opens a window into how games store and manage data in memory. By understanding memory manipulation, you can...

03 January 2026

---

[← Back to home](#)

[All posts →](#)

---

© 2026 Yunus Aydın. All rights reserved.



RSS Feed