Malicious OPEN Redirection *CVE-2025-61166*

Nikhil Thakur 2 min read · Feb 11, 2026

SigningHub by Ascertia v.8.6.8 — Malicious OPEN Redirection

> Discovered by: Nikhil Thakur | Yazan Abu-Nadi

```
################################################################################
# Title: SigningHub by Ascertia v.8.6.8 - Malicious OPEN Redirection
# Author: Mr. Nikhil Thakur | Yazan Abu-Nadi
# Vendor Homepage: https://www.signinghub.com/
# Version: v.8.6.8
# Tested on: Latest version of Chrome, Firefox on Windows and Linux.
# CVE: CVE-2025-61166
################################################################################
```

— — — — — — — — — — — — — — — OPEN Redirection — — — — — — — — — — — — — —

> Open redirection arises when a script writes controllable data into the target of a redirection in an unsafe way. An attacker may be able to use the

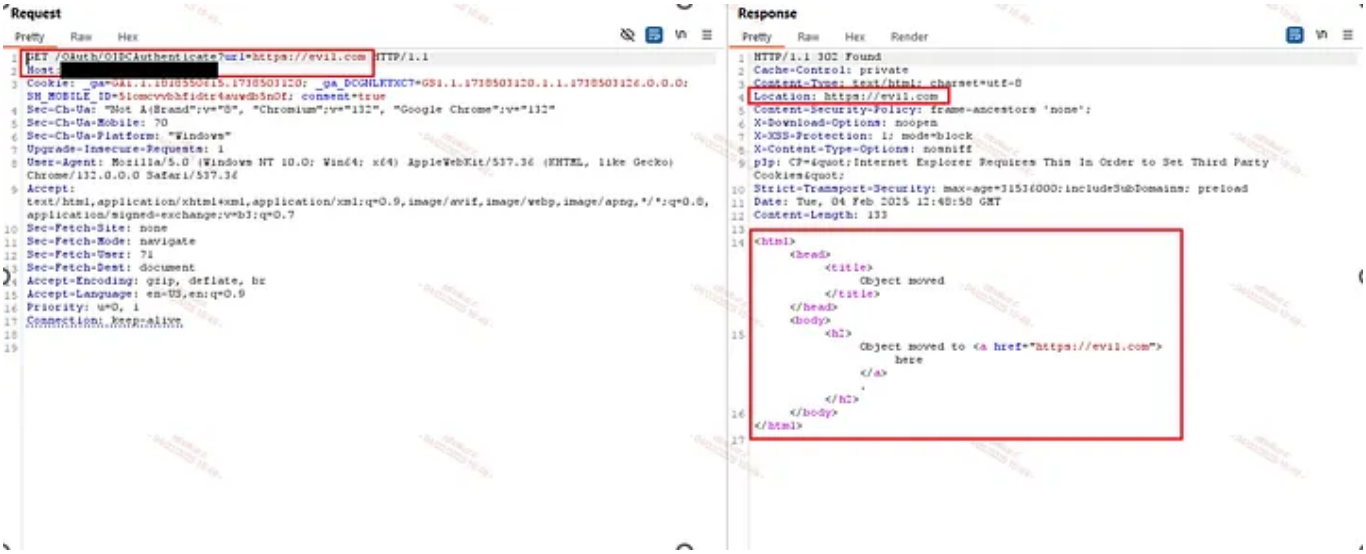
> vulnerability to construct a URL that, if visited by another application user, will cause a redirection to an arbitrary external domain. This behavior can be leveraged to facilitate phishing attacks against users of the application. The ability to use an authentic application URL, targeting the correct domain and with a valid SSL certificate (if SSL is used), lends credibility to the phishing attack because many users, even if they verify these features, will not notice the subsequent redirection to a different domain. If an attacker can control the start of the string that is passed to the redirection API, then it may be possible to escalate this vulnerability into a JavaScript injection attack, by using a URL with the javascript: pseudo-protocol to execute arbitrary script code when the URL is processed by the browser.

— —

Reproduction Steps:

> *Step 1: Initial Discovery and Reconnaissance:*
> *As an unauthenticated user, navigate to the target application's base URL (e.g., https://*.signinghub.com). Using browser developer tools (F12 → Network tab) or an intercepting proxy like Burp Suite, spider/crawl the application to identify all endpoints. Look specifically for parameters named url, redirect, return, next, or destination. You should identify the endpoint: /OAuth/OIDCAuthenticate?url=*
>
> *Step 2: Craft the Malicious Request:*
> *Construct a GET request to the vulnerable endpoint with an external domain in the url parameter:*
> *https://*.signinghub.com/OAuth/OIDCAuthenticate?url=https://evil.com*
> *(Replace url= value with the actual subdomain with a domain you control or a trusted external site like https://evil.com for testing)*

> *Step 3: Execute and Observe:*
> *Submit the crafted URL directly in a browser address bar or via a crafted link. Observe if the application immediately redirects the browser to https://evil.com without any user warning, interstitial page, or validation prompt.*
>
> *Step 4: Traffic will be redirected to https://evil.com ;)*



Successfully Redirected to https://evil.com

An unauthenticated attacker can able to find this vulnerable “url=” parameter endpoint by just spidering the SigningHub application portal url to identify associated endpoints.

— — — Many Thanks — — —

Written by Nikhil Thakur

3 followers · 1 following

No responses yet