**SOC166 - Javascript Code Detected in Requested URL**

**Incident Report: Malicious XSS Attempt Detected**

**Date:** February 26, 2022  
**Time:** Between 03:34 PM and 03:56 PM  
**Source IP:** 112.85.42.13  
**Target Server:** 172.16.17.17  
**Target Port:** 443  
**Device Action:** Permitted  
**HTTP Response Codes:** 200 (initial traffic), 302 (for all malicious attempts)

A screenshot of a computer

Description automatically generated

**Incident Overview**

A malicious actor attempted to exploit the target server, **172.16.17.17**, by injecting **Cross-Site Scripting (XSS)** payloads into the q parameter of various HTTP GET requests. The malicious payloads included JavaScript and HTML code intended to execute unauthorized scripts on the server or client. However, these attempts were unsuccessful due to the server responding with an **HTTP 302 Found** status, redirecting the requests and preventing payload execution.

**Sequence of Events**

1. **Initial Legitimate Requests:**
   * At **03:34 PM**, the attacker accessed the root URL, followed by the "About Us" page at **03:35 PM**. These requests returned **HTTP 200 OK**, indicating normal browsing activity.

A screenshot of a computer

Description automatically generated

1. **Testing and Exploitation Attempts:**
   * Between **03:45 PM and 03:56 PM**, the attacker submitted multiple requests containing malicious payloads, such as:
     + prompt(8) to trigger a JavaScript popup.
     + <$img src=q onerror=prompt(8)$> and <$svg><$script ?>$alert(1) to exploit HTML-based vulnerabilities.
     + <$script>$for((i)in(self))eval(i)(1)<$/script> to execute arbitrary JavaScript functions.
     + <$script>javascript:$alert(1)</script> for direct execution of JavaScript.
   * Each request was permitted but resulted in an **HTTP 302 Found** response, redirecting the payloads and neutralizing the attack.

A screenshot of a computer program

Description automatically generated

1. **Payload Analysis:**
   * All payloads leveraged typical XSS injection techniques targeting input fields or parameters.
   * The **HTTP Response Size of 0** confirms that no data was processed or rendered from the injected scripts.

A screenshot of a computer

Description automatically generated

**Key Observations**

* **Attack Outcome:** All malicious attempts were unsuccessful due to the server's redirection behavior.
* **Risk Mitigation:** The server handled requests securely by redirecting malicious input, effectively preventing script execution.
* **Attacker's Intent:** Likely to exploit XSS vulnerabilities for unauthorized script execution or reconnaissance.

**Recommendations**

1. **Implement Input Validation and Output Encoding:**  
   Ensure strict input sanitization at all endpoints to block malicious scripts at the entry point.
2. **Strengthen Web Application Firewall (WAF):**  
   Enhance firewall rules to detect and block XSS payloads proactively.
3. **Monitor and Block Malicious IPs:**  
   Add the source IP **112.85.42.13** to the blocklist and monitor for similar activity.
4. **Regular Security Audits:**  
   Conduct routine application testing to identify and patch XSS vulnerabilities.
5. **Enable Threat Intelligence:**  
   Integrate threat feeds to detect and mitigate similar patterns of attack.

**Conclusion**

This investigation highlights a series of unsuccessful XSS attempts, where the attacker’s payloads failed to execute due to the server's secure handling of input via HTTP 302 redirection. While no breach occurred, this incident emphasizes the need for ongoing vigilance and robust input sanitization to mitigate similar threats.

**References**

1. OWASP Foundation. *OWASP Top Ten Web Application Security Risks*. Available at: https://owasp.org/www-project-top-ten/
2. Mozilla Developer Network (MDN). *JavaScript Security*. Available at: <https://developer.mozilla.org/en-US/>
3. MITRE ATT&CK Framework. *Exploit Public-Facing Application (T1190)*. Available at: https://attack.mitre.org/
4. Let's Defend. *SOC166 - JavaScript Exploit Analysis Lab*. Available at: <https://letsdefend.io/>