Crazy Calculatorz Risk Assessment

Cross-Site Scripting (XSS):

Likelihood of Exploitation: High

The likelihood that a threat actor could exploit is high. Cross-site scripting is not a terribly complex attack. Testing for input validation is one of the primary steps to testing a web page containing forms. Because the “calculator.html” page does not have any validation whatsoever, it would be very easy for threat actors to exploit it.

Impact if Exploited: High

The vulnerable “calculator.html” page uses POST requests to execute the “calc.php” server-side script. Because of this, threat actors cannot craft malicious links for victims to click. That being said, XSS attacks using a POST form can still result in significant security risks. Attackers can create their own forms, manipulating submissions, allowing them to change user information or post malicious content. While the calculator page does not make use of cookies or other user data, attackers could potentially gain access to sensitive information used on other pages of the application.

Overall Risk: High

Based on the high likelihood and impact of exploitation, the overall risk of an XSS attack is high. The use of POST requests just slightly decreases the risk from Critical to High. POST XSS attacks remain a large threat and have the potential to cause substantial harm but require slightly more effort than GET XSS attacks. A successful exploit still relies on social engineering but is not as simple as users clicking a malicious link.

CVSS Metrics

8.1 - AV:N/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:N

Attack Vector – Network: Attack is executed over a network via a web application.

Attack Complexity – Low: No special access conditions, user setup, or complex interactions needed. Form submission is the only requirement.

Privileges Required – None: Navigating to “calculator.html” and form submission does not require any privileges.

User Interaction – Required: Requires a user to visit the page and interact with the form.

Scope – Unchanged: When the malicious script is executed, it does not affect other parts of the application or other users.

Confidentiality Impact – High: XSS can be used to steal session tokens or other sensitive information.

Integrity Impact – High: XSS can be used to manipulate users and trick them into performing unintended actions.

Availability Impact – None: This attack does not disrupt access to the application.

Relevant CWEs

- CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
- CWE-80: Improper Neutralization of Script-Related HTML Tags in a Web Page (Basic XSS)
- CWE-20: Improper Input Validation

ASVS Impact

- **V1.5 Input and Output Architecture:** The application directly echoes user input back to the client. This allows attackers to inject JavaScript into a form, the server reflects it back, and it is executed in the target browser.
- **V5.1 Input Validation:** The application does not verify or validate user input, allowing for untrusted input to be echoed back and executed in the user's browser.
- **V5.2 Sanitization and Sandboxing:** The application does not sanitize user input which would strip out malicious content.
- **V5.3 Output Encoding and Injection Prevention:** The application does not encode or escape output, which allows attackers to inject JavaScript for execution.
- **V14.4 HTTP Security Headers:** Due to the lack of HTTP security headers like Content Security Policy (CSP), attackers can use this vulnerability inject and execute JavaScript.

PHP Injection/Remote Code Execution:

Likelihood of Exploitation: Critical

The likelihood of this vulnerability being exploited is critical. This vulnerability does not require any additional privileges, is relatively easy to exploit, and is a high value vulnerability that can have serious consequences.

Impact if Exploited: Critical

PHP injection through the use of an insecure use of the "eval()" function can have devastating effects on a web application. This vulnerability has the potential to compromise the confidentiality, integrity, and availability of a system. Not only can attackers inject any number of php commands, but they can leverage the "shell_exec()" function to execute shell commands on the host. Using this vulnerability attackers can gain unauthorized access to the system allowing them to view privileged data, impersonate legitimate users, exfiltrate data, manipulate records, shut down the system, and more. PHP injection vulnerabilities are extremely dangerous and should be a top priority for remediation.

Overall Risk: Critical

The overall risk of this vulnerability is critical. PHP injection is a relatively simple attack that does not require additional permissions or excessive knowledge of the inner workings of the application. Using trial and error attackers can inject PHP expressions going as far as remotely executing shell commands. Remote code execution can have devastating effects to confidentiality, integrity, and availability of a web application. PHP injection remediation should be a top priority and treated with due diligence.

CVSS Metrics

10.0 - AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:H/A:H

Attack Vector – Network: Attack is executed over a network via a web application.

Attack Complexity – Low: Attack does not require special conditions. Developing a payload requires some trial and error but is still relatively simple to execute.

Privileges Required – None: Navigating to “calculator.html” and form submission does not require any privileges.

User Interaction – None: Users do not need to be involved for threat actors to successfully exploit this vulnerability.

Scope – Changed: This vulnerability allows attackers to execute shall commands which is out of the scope of the web application.

Confidentiality Impact – High: Attackers have shell access meaning there is a wide variety of actions they could take to violate confidentiality. An example of this is to read the contents of the “/etc/passwd” file.

Integrity Impact – High: With the ability to execute shell commands, attackers could alter files and data, greatly impacting the integrity of the application.

Availability Impact – High: Attackers could execute shell commands that could potentially disrupt services such as stopping processes, deleting configuration files, or disabling the system entirely.

Relevant CWEs

- CWE-78: Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
- CWE-94: Improper Control of Generation of Code ('Code Injection')
- CWE-95: Improper Neutralization of Directives in Dynamically Evaluated Code ('Eval Injection')

ASVS Impacts/Violations

- **V1.5 Input and Output Architecture:** The application uses the “eval” function to carry out critical application processes (math calculations). It fails to validate any input or output, allowing users to execute code remotely.
- **V5.1 Input Validation:** The application does not verify or validate user input, allowing for untrusted input to be used in command execution.
- **V5.2 Sanitization and Sandboxing:** The application does not sanitize user input which would strip out malicious content.
- **V5.3 Output Encoding and Injection Prevention:** The application does not encode or escape output, which allows attackers to inject commands for execution.
- **V6.4 Secret Management:** This vulnerability exposes sensitive information such as configuration files, credentials, keys, or environment variables.
- **V10.1 Code Integrity:** The integrity of the code execution is compromised as external input can affect the path the code takes.
- **V10.3 Application Integrity:** This vulnerability allows attackers to take advantage of the lack of manipulation prevention mechanisms. Attackers can make system modifications which violate the integrity of the application.