WHEN I FOUND A SECURITY VULNERABILITY IN ZIVAME

Khushank Raj Mahawan

Vulnerability

REFLECTED XSS in input parameter(search-box) on the website.

Description:-

Cross-Site Scripting (XSS) attacks are a type of injection, in which malicious scripts are injected into otherwise benign and trusted websites. XSS attacks occur when an attacker uses a web application to send malicious code, generally in the form of a browser side script, to a different end user. Flaws that allow these attacks to succeed are quite widespread and occur anywhere a web application uses input from a user within the output it generates without validating or encoding it.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

You Can Read More at:

https://www.owasp.org/index.php/Cross-site_Scripting_(XSS)

Proof-of-Concept(PoC)

Firefox Version : 70.0.1 (64-bit)

OS Version: 10.0

URL:- https://www.zivame.com/

Vulnerable Parameter: search box

Other browsers tested:

Chrome, Safari, Edge(All are showing same output)

What steps will reproduce the problem?

1.Normally Open the URL in browser

2.Type <script>alert(1)</script> in the search-box

3.Pop with ‘1’ will show up on screen.

What is the expected result?

Any kind of script should not be executed through input parameters on client side as it allows a hacker or any other unauthorized user to inject malicious code inside the script for attacks like cookie stealing and session hijacking etc.

What happens instead of that?

Script tags and html tags are executing directly on the client side that is a major concern and vulnerability that can be critical, allowing the attacker to take full control of the website and compromise all other users and their data.

Suggestive Measures to Prevent such vulnerability:

In general, effectively preventing XSS vulnerabilities is likely to involve a combination of the following measures:

- **Filter input on arrival.** At the point where user input is received, filter as strictly as possible based on what is expected or valid input.
- **Encode data on output.** At the point where user-controllable data is output in HTTP responses, encode the output to prevent it from being interpreted as active content. Depending on the output context, this might require applying combinations of HTML, URL, JavaScript, and CSS encoding.
- **Use appropriate response headers.** To prevent XSS in HTTP responses that aren't intended to contain any HTML or JavaScript, you can use the Content-Type and X-Content-Type-Options headers to ensure that browsers interpret the responses in the way you intend.
- **Content Security Policy.** As a last line of defense, you can use Content Security Policy (CSP) to reduce the severity of any XSS vulnerabilities that still occur.

I am also attaching a small PoC(proof –of-concept) screenshots for more clarification on this responsible disclosure.

Conclusion:-

I am a Cyber Security Student & found this vulnerability while Bug Hunting so it was my responsibility to report you about the same to maintain Cyber safety and disallow hackers or unauthorized people to make a misuse for the same. I did not tamper any data or disclosed this vulnerability to anyone.

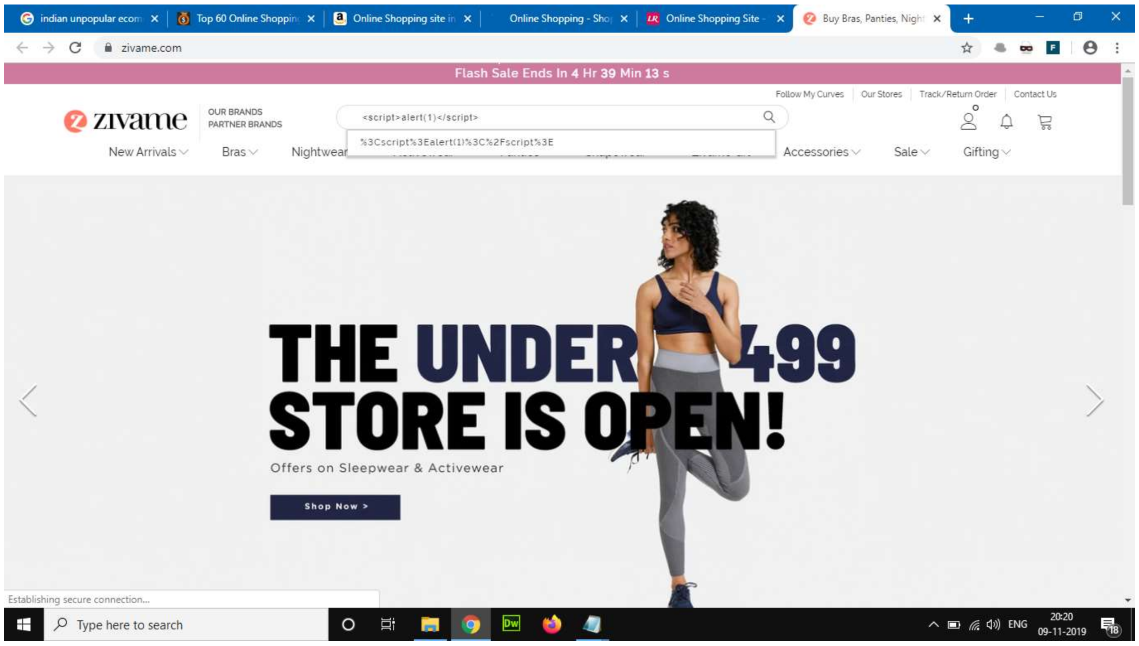
Any sort of Reward/Bounty and Hall Of Fame/Certificate of Appreciation from your side to me will be appreciated and keep me motivated to help more organisations and users from cyber risks.

Thank You

KHUSHANK RAJ MAHAWAN

POC

Step1

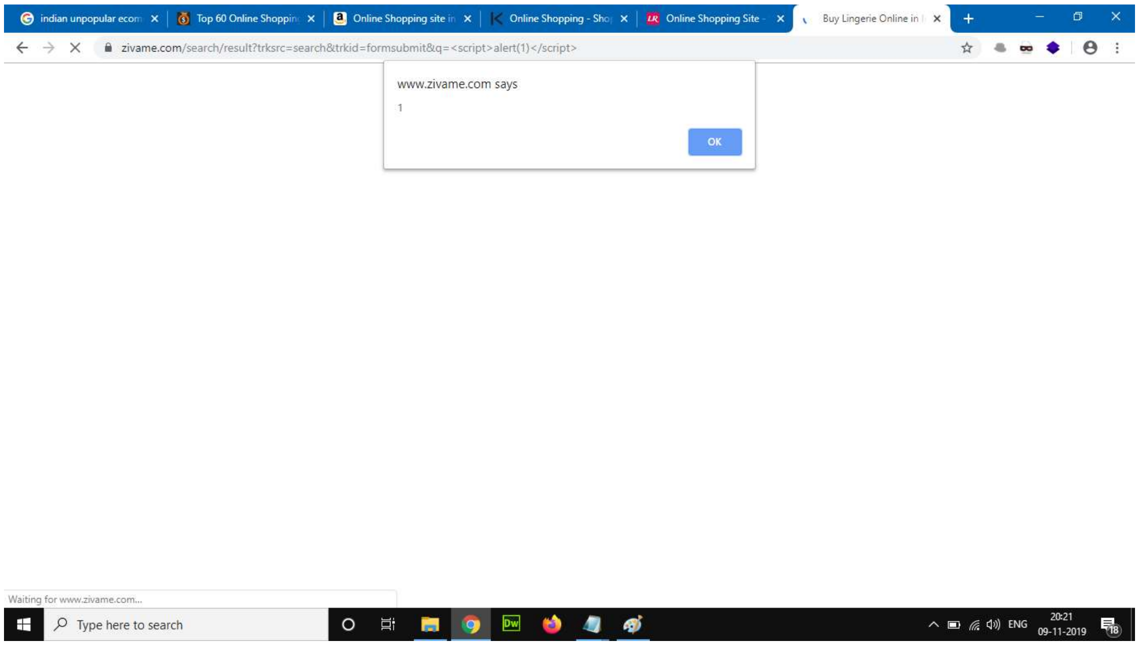


zivame.com

<script>alert(1)</script>

%3Cscript%3Ealert(1)%3C%2Fscript%3E

Step2



zivame.com/search/result?trksrc=search&trkid=formsubmit&q=<script>alert(1)</script>

www.zivame.com says

1