Companion Notes on CVE-2018-3883

An exploitable SQL injection vulnerability exists in the authenticated part of ERPNext v10.1.6. Specially crafted web requests can cause SQL injections resulting in data compromise. The employee and sort order parameter can be used to perform an SQL injection attack. An attacker can use a browser to trigger these vulnerabilities, and no special tools are required.
(https://www.cvedetails.com/cve/CVE-2018-3883)

Terms used in the Report

1. What is CVE (common vulnerabilities and exposures) ?

- It is a program launched in 1999 by MITRE, a non-profit that operates research and development centers sponsored by the federal government, to identify and catalog vulnerabilities in software or firmware into a free “dictionary” for organizations to improve their security.
  Ref: https://www.csoonline.com/article/3204884/what-is-cve-its-definition-and-purpose.html

2. What could be a vulnerability in computer world ?

- According to theCVE website, a vulnerability is a mistake in software code that providesan attacker with direct access to a system or network. It could allow an attacker to pose as a super-user or system administrator with full access privileges.An attacker indirect access to a system or network. It could allow an attacker to gather customer information that could be sold.

3. What is ERP (Enterprise resource planning) ?

- ERP could be categorised as business management software—typically a suite of integratedapplications—that an organization can use to collect, store, manage, andinterpret data from these manybusinessactivities.ERP systems track business resources—cash,raw materials,production capacity—and the status of business commitments: orders,purchase orders, andpayroll.The applications that make up the system share data across various departments (manufacturing, purchasing, sales,accounting, etc.) that provide the data.ERP facilitates information flow between all business functions and manages connections to outsidestakeholders.
  Ref: https://en.wikipedia.org/wiki/Enterprise_resource_planning

4. What is ERPNext software ?

- ERPNextis afree and open-sourceintegratedERP software developed by Frappé Technologies Pvt. Ltd. and is built onMariaDBdatabase system using aPythonbased server-side framework.
  ERPNext is a generic ERP software used by manufacturers, distributors and services companies. It includes modules like accounting, CRM, sales, purchasing, website + e-commerce,

point of sale, manufacturing, warehouse, project management, inventory and services. Also, it has domain specific modules like schools, healthcare, agriculture and non-profit.
Ref: https://en.wikipedia.org/wiki/ERPNext#Architecture

5. What is SQL Injection ?

- SQL Injection is a type of injection attack with which an attacker may inject malicious SQL statements into a database server. The SQL injection vulnerabilities may be used to bypass an application's security measures. The authentication and authorization of the application can be bypassed, and content can be retrieved from the entire SQL database. Information can be added, deleted or modified in the database with such an attack.
- This vulnerability may affect web applications that use an SQL database in the server-side for storage of data. This attack is one of the most prevalent and dangerous web application vulnerabilities. The Open Web Application Security Project (OWASP) lists injections as the top threat to web applications in their OWASP top 10 2017 document.

CVE Details

Description of CVE:

An exploitable SQL injection vulnerability exists in the authenticated part of ERPNext v10.1.6. Specially crafted web requests can cause SQL injections resulting in data compromise. The employee and sort order

parameter can be used to perform an SQL injection attack. An attacker can use a browser to trigger these vulnerabilities, and no special tools are required.
(https://www.cvedetails.com/cve/CVE-2018-3883)

Impact factors:

CVSS v3.0 Security and Metrics:

- **Base Score**: 8.8 HIGH
- **Vector**: AV:N/AC:L/PR:L/UI:N/S:U/C:H/I:H/A:H (Legend given below)
- **Impact Score**: 5.9
- **Exploitability Score**: 2.8
- **Attack Vector**: Network
- **Attack Complexity**: Low
- **Priviledge Required**: Low
- **User Learning**: None
- **Scope**: Unchanged
- **Confidentiality**: High
- **Integrity**: High
- **Availability**: High

CVSS v2.0 Severity and Metrics:

- **Base Score**: 6.5 MEDIUM
- **Impact Score**: 6.4
- **Exploitability Score**: 8.0

- **Vector**: AV:N/AC:L/Au:S/C:P/I:P/A:P

CWE: 89 (Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection'))

Vulnerability Details

The Vulnerability was found on September 9th, 2018 and consequently reported to the company's vulnerability disclosure program. Consequently, the vulnerability was assigned a CVE and disclosed to the users and patched in the next software update.

The following are the parameters where the Injection of SQL code could be done to get the access and/or modify the user database and consequently the system configs of the ERPNext v10.1.6 software

searchfield parameter

The searchfield parameter can be used to perform an SQL injection attack as shown below:

```
GET /?txt=a&searchfield=name<SQLINJECTION>&query=erpnext.c
ontrollers.queries.employee_query&doctype=Employee&cmd=fra
ppe.desk.search.search_widget&_=1522110063950 HTTP/1.1
Host: 192.168.239.140
User-Agent: Mozilla/5.0 (Windows NT 6.3; Win64; x64; rv:59
.0) Gecko/20100101 Firefox/59.0
Accept: application/json, text/javascript, /; q=0.01
```

```
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.239.140/desk
X-Frappe-CSRF-Token: 14ee26a793805ed02dbd172b28d514503da3d
31fb5e9392930567947
X-Requested-With: XMLHttpRequest
Cookie: user_image=; user_id=Administrator; system_user=ye
s; full_name=Administrator; sid=dd26a9f121a4177ed22d8f5ff0
a93508eb095cbf18cecaa020cccdd4; io=-cQWmng9Wch23ijkAAAF
DNT: 1
Connection: close
```

employee parameter

The employee parameter can be used to perform an SQL injection attack as shown below:

```
POST / HTTP/1.1
Host: 192.168.239.140
User-Agent: Mozilla/5.0 (Windows NT 6.3; Win64; x64; rv:59
.0) Gecko/20100101 Firefox/59.0
Accept: application/json, text/javascript, /; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://192.168.239.140/desk
Content-Type: application/x-www-form-urlencoded; charset=U
TF-8
```

```
X-Frappe-CSRF-Token: 14ee26a793805ed02dbd172b28d514503da3d
31fb5e9392930567947
X-Requested-With: XMLHttpRequest
Content-Length: 194
Cookie: user_image=; user_id=Administrator; system_user=ye
s; full_name=Administrator; sid=dd26a9f121a4177ed22d8f5ff0
a93508eb095cbf18cecaa020cccdd4; io=-cQWmng9Wch23ijkAAAF
DNT: 1
Connection: close

    employee=EMP%2f0001<SQLINJECTION>&date=2018-03-07&leav
e_type=Leave+Without+Pay&consider_all_leaves_in_the_alloca
tion_period=true&cmd=erpnext.hr.doctype.leave_application.
leave_application.get_leave_balance_on
```

sort_order parameter

The sort_order parameter can be used to perform an SQL injection attack as shown below:

```
POST / HTTP/1.1
Host: 192.168.239.140
User-Agent: Mozilla/5.0 (Windows NT 6.3; Win64; x64; rv:59
.0) Gecko/20100101 Firefox/59.0
Accept: application/json, text/javascript, /; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
```

```
Referer: http://192.168.239.140/desk
Content-Type: application/x-www-form-urlencoded; charset=U
TF-8
X-Frappe-CSRF-Token: 14ee26a793805ed02dbd172b28d514503da3d
31fb5e9392930567947
X-Requested-With: XMLHttpRequest
Content-Length: 113
Cookie: user_image=; user_id=admin%40admin.com; system_use
r=yes; full_name=asd; sid=dd26a9f121a4177ed22d8f5ff0a93508
eb095cbf18cecaa020cccdd4; io=ELCOCSQzSPt1L6_fAAAE
DNT: 1
Connection: close

    item_code=asdasd&start=0&sort_by=projected_qty&sort_or
der=asc<SQLINJECTION>&cmd=erpnext.stock.dashboard.item_das
hboard.get_data
```

Prevention Techniques for SQL Injection Attacks

Web framework

As of late, some Web Frameworks offer different SQL Injection attack prevention techniques. For example, if you are using PHP in your application or product you can use Magic Quotes. It is a function utilized when any blend of 4 special characters „,",/, NULL is in the data sent from POST, GET, COOKIES. It prefixes a “\ before the special characters. But “this method can be easily bypassed by an attacker and should be avoided because firstly, Magic Quotes supports only 4 special characters. It’s not the best way to perform escaping of malicious data and certainly not the most secure.

Instruction-Set Randomization

Instruction-set randomization technique inputs arbitrary values into the runtime SQL query statement of a web application and checks for instability, vulnerability to recognize SQL injection attacks. SQLrandplaces a proxy between the web server and the database server and randomizes SQL queries. On the off chance that the random value can be predicted, this technique isn’t very effective and should be used only as a basic testing procedure.

Input validation and whitelist validation

We can apply input validation to our application or product. For example, aswe have seen when we fill an online form that warns you about the length or strength of your password, that is an example of input validation. Assume that all the input we get is malicious. We can use a whitelist of acceptable inputs that carefully adjust to specifications. Reject any information that does not carefully comply with specifications or change it into something that does. While building SQL query strings, utilize stringent whitelists that limit the character set dependent on the expected value of the parameter in the request. This will indirectly limit the extent of an assault.

Prepared statements with parameterized queries

Use of prepared statements with parameterized queries probably your best bet against a SQL injection attack. Instead of writing dynamic queries, which fails to differentiate between application code and data, prepared statements force developers to use static SQL query and then pass in the external input as a parameter to query. This approach ensures the SQL interpreter always differentiates between code and data because the root cause of SQL injection is that the malicious data gets interpreted as code, prepared statements take care of that by sending the code and the data separately to the database server.

Stored procedures

Stored procedures are the SQL statements defined and stored in the database itself and then called from the application. These are very similar to the prepared statements defined above. Developers are usually only required to build SQL statements with parameters that are automatically parameterized. However, it’s possible for a developer to generate dynamic SQL queries inside stored procedures. Implement stored procedures safelyby avoiding dynamic SQL generation inside. This method is said to be faster than prepared statements because the procedures are already stored which makes it more efficient.

References

- Joshi, P., Ravishankar, N., Raju, M. and Ravi, N. (2018).

Encountering SQL Injection in Web Applications.2018 Second International Conference on Computing Methodologies and Communication (ICCMC).

- Li Qian, Zhenyuan Zhu, Jun Hu and Shuying Liu (2015). Research of SQL injection attack and prevention technology.2015 International Conference on Estimation, Detection and Information Fusion (ICEDIF).
- https://www.synopsys.com/software-integrity/resources/knowledge-database/sql-injection.html
- https://cwe.mitre.org/data/definitions/89.html
- http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.681.8691&rep=rep1&type=pd
- https://www.acunetix.com/websitesecurity/sql-injection/
- https://www.acunetix.com/vulnerability-scanner/owasp-top-10-compliance/
- https://talosintelligence.com/vulnerability_reports/TALOS-2018-0560