SCENARIO

The application possesses a NoSQL Injection Vulnerability and is powered by a MongoDB NoSQL database. We will try to exploit the vulnerability in order to gain administrator privileges using MongoDB operators.

**PROCEDURE**

1. Open the web application and navigate to the user login page.
2. Login using the given credentials.
3. Send GET /user/lookup?user=wiener request from the Proxy tab to BurpSuite’s Repeater.
4. First, we will inject Payload 1 in order to get the length of the administrator’s account’s password in URL encoded form.
5. Forward the request to the Intruder as we need to mount a brute force attack on the password’s characters one by one.
6. Now in order to mount a brute force attack place Payload 2 in place of the user parameter and go for Cluster Bomb attack and in Payload 1 for the attack select 0-7 and a-z for Payload 2.
7. After the attack is completed, sort by descending order on the length of the content and note down the password.

**PAYLOAD**

1. administrator' && this.password.length == 8|| 'a'=='b
2. administrator' && this.password[§0§]=='§a§

**REMEDIATION**

1. **Avoid Direct User Input:** As with SQL, the primary method of exploitation in NoSQL databases like MongoDB is the use of unsanitized user input in queries. You should never construct database queries with raw user input. Always sanitize and validate any data that comes from an untrusted source before using it in a query.
2. **Use Parameterized Queries:** When possible, use parameterized queries or prepared statements. MongoDB offers methods that allow you to use parameters which are not susceptible to injection.
3. **Strong Data Validation:** Use strong data validation – every piece of data your application accepts should be validated and sanitized. Use a library or framework that provides built-in functionality for this.
4. **Database User Permissions:** Limit the permissions of the application's database user. Ideally, your application should not be using a database user that has permission to drop tables or make administrative changes.
5. **Regularly Update Database Software:** Ensure that you're using the most recent version of your NoSQL database. Updated versions often come with security patches.
6. **Error Handling:** Do not expose database error details to the user. This information can provide attackers valuable insights into the internal workings of your application.
7. **Enable Database Authentication:** Always set up and use authentication for your MongoDB databases. Set strong, unique passwords for each user and avoid using the root user.
8. **Use Encryption:** Encrypt sensitive data before storing it in the database. This adds an additional layer of protection.