SecCode Analyzer

Analysis Results

Analysis

1. SQL Injection Vulnerability

- Severity: Critical

- The code is vulnerable to SQL injection attacks in both the `add_user` and `authenticate` functions because it directly concatenates user input into SQL queries. An attacker can manipulate the input to execute arbitrary SQL commands.

2. Plain Text Password Storage:

- Severity: High

- Storing passwords in plain text in the database is a significant security risk. If the database is compromised, all user passwords can be easily exposed.

Improvements

1. Parameterized Queries

- Use parameterized queries to prevent SQL injection attacks. Parameters will be properly sanitized by the database driver.

```python

def add_user(username, password):

cursor.execute("INSERT INTO users (username, password) VALUES (?, ?)", (username, password))

conn.commit()

## def authenticate(username, password)
```

cursor.execute("SELECT * FROM users WHERE username = ? AND password = ?", (username, password))

user = cursor.fetchone()

if user:

print("Autenticado com sucesso!")

else:

print("Falha na autenticação!")

```

## 2. Password Hashing

- Hash the passwords before storing them in the database. Use a strong hashing algorithm like bcrypt to securely store passwords.

```python

import bcrypt

## def add_user(username, password)

hashed_password = bcrypt.hashpw(password.encode('utf-8'), bcrypt.gensalt())

cursor.execute("INSERT INTO users (username, password) VALUES (?, ?)", (username, hashed_password))

conn.commit()

## def authenticate(username, password)

cursor.execute("SELECT * FROM users WHERE username = ?", (username,))

user = cursor.fetchone()

if user and bcrypt.checkpw(password.encode('utf-8'), user[2]):

print("Autenticado com sucesso!")

else:

print("Falha na autenticação!")

```

3. Closing Database Connection

- Ensure to close the database connection properly after its use to prevent resource leaks.

```python

conn.close()

```

By implementing these improvements, you can significantly enhance the security of the application and protect it from common security vulnerabilities.
```