Password Management Mitigation SOP

# Development Mitigation SOP

Password management issues occur when a password is stored in plaintext in an application’s properties or configuration file.

# Defense Against [DEFECT]

A password should never be stored in plaintext. Instead, the password should be entered by an administrator when the system starts. If that approach is impractical, a less secure but often adequate solution is to obfuscate the password and scatter the de-obfuscation material around the system so that an attacker has to obtain and correctly combine multiple system resources to decipher the password. At the very least, passwords should be hashed before being stored.

# Examples

…

Properties prop = new Properties();

prop.load(new FileInputStream(“config.properties”));

String password = prop.getProperty(“password”);

DriverManager.getConnection(url, usr, password);

…

## Explanation

This code will run successfully, but anyone who has access to config.properties can read the value of password. If a devious employee has access to this information, they can use it to break into the system.

## Recommendation

The code below demonstrates how to integrate SQLCipher into an Android application after downloading the necessary binaries, and store credentials into the database file.

Import net.sqlcipher.database.SQLiteDatabase;

…

SQLiteDatabase.loadLibs(this);

File dbFile = getDatabasePath(“credentials.db”);

dbFile.mkdirs();

dbFile.delete();

SQLiteDatabase db = SQLiteDatabase.openOrCreateDatabase(dbFile, “credentials”, null);

db.execSQL(“create table credentials(u, p )”);

db.execSQL(“insert into credentials(u, p) values(?, ?)”, new Object[]{username, password});

…

**Resources**

1. [INPUT-1: Validate inputs, Oracle](http://www.oracle.com/technetwork/java/seccodeguide-139067.html#5)