ICS 491 Assignment 5: SDL Release

Authors: Team Gamma (Cynthia Okuno, Christie Reindle, Christopher Delp)

Date: July 31, 2016

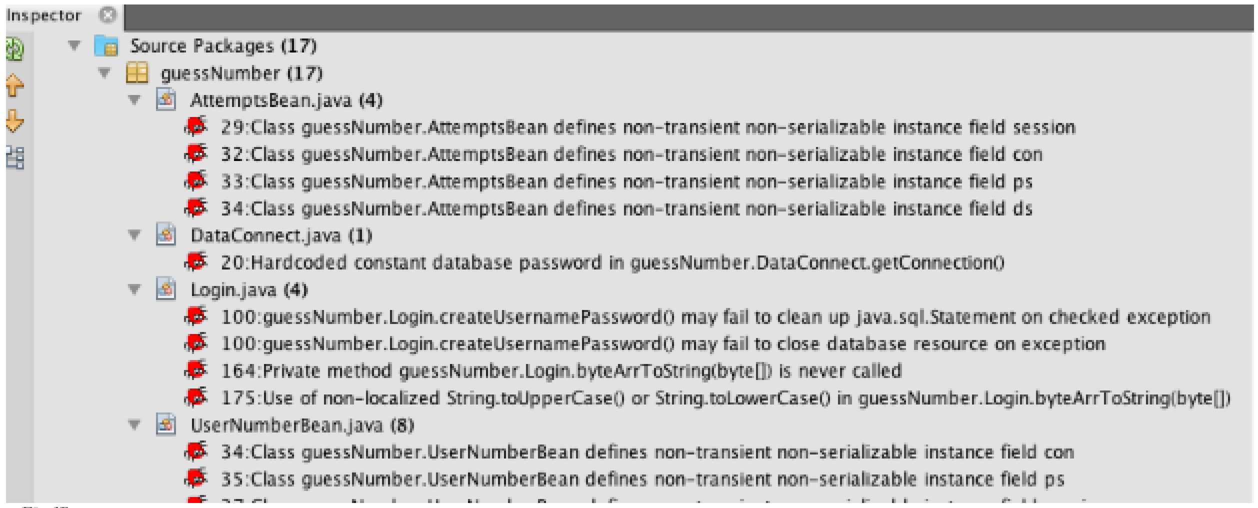
1 Incident Response Plan.

a. Privacy Escalation Team
    i. Escalation Manager - Christie Reindle
    Orders, structures, and brings in the management's attention to a major incident or a problem, which has escalated beyond its limits. The Escalation Manager ensures that unresolved problems don't linger, these issues are promptly addressed, and restores the normal service operation as quickly as possible to drive the process to completion.(1)
    ii. Legal Representative - Christopher Delp
    Responsible for helping resolve any legal or PR concerns consistently throughout the process.
    iii. Public Relations Representative - Cynthia Okuno
    Responsible for helping resolve any PR concerns consistently throughout the process and alerting the public of the issue
b. okunoc@hawaii.edu
c. Privacy Escalation Response Process
    i. Escalation manager
        1. Is e-mail notified of the issue
        2. Finds source of escalation, determines impact and breadth on program
        3. Ensures incident is valid and violator has indeed perpetuated the application
        4. Gathers list of known facts to take action against perpetrator(s)
        5. Creates timeline expectations to include taking action against the perpetrator(s), find area of vulnerability within program, and give information to rest of team
        6. Seeks to fix pothole in program
    ii. Legal Representative
        1. Sees to the extent of which the perpetrator has violated the program
        2. Takes into account the list of known facts of the incident(s) and perpetrator(s) from the Escalation Manager and team
        3. Sets to persecute perpetrator(s) based on degree of violations
    iii. Public Relations Representative
        1. Once incident is known to be valid and actions are to be taken against violator, seeks to notify other users of the program of the incident

2 Final Security Review.

Some of the threats identified in our initial threat model are not applicable to the application in its current version. A few features, like paypal integration, have not been finalized, or are not possible in a demo version. For example, threats involving authentication or exposed data when making paypal purchases, are not an issue since that feature hasn't been enabled. Other than denial of service, which again isn't applicable to a demo version, the majority of our threats involved the database. Particularly user authentication and confidentiality, and database integrity in general. Although several new database queries have been added since our initial version, it is our assessment that the database still remains secure.

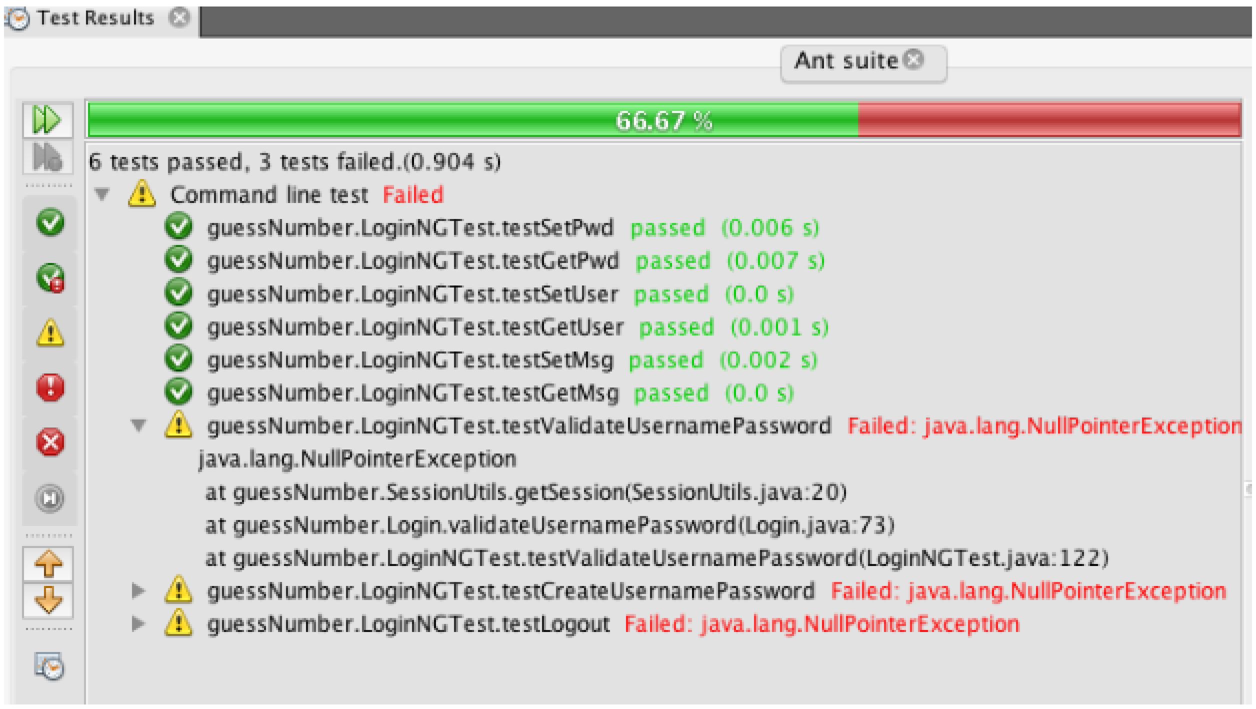
Recent static analysis using FindBugs revealed a possible security vulnerability in our prepare statements. Specifically the potential for SQL injection via our use of the user defined "username" variable in our SQL queries. As a result, changes were made to the way our affected prepare statements were created, parametrizing user input so that it will not be treated as a string. Otherwise, static analysis results were the same as previous times. The primary known flaw from these being a hard-coded password, for which we have no solution at this time.



```
Inspector
Source Packages (17)
  guessNumber (17)
    AttemptsBean.java (4)
      29:Class guessNumber.AttemptsBean defines non-transient non-serializable instance field session
      32:Class guessNumber.AttemptsBean defines non-transient non-serializable instance field con
      33:Class guessNumber.AttemptsBean defines non-transient non-serializable instance field ps
      34:Class guessNumber.AttemptsBean defines non-transient non-serializable instance field ds
    DataConnect.java (1)
      20:Hardcoded constant database password in guessNumber.DataConnect.getConnection()
    Login.java (4)
      100:guessNumber.Login.createUsernamePassword() may fail to clean up java.sql.Statement on checked exception
      100:guessNumber.Login.createUsernamePassword() may fail to close database resource on exception
      164:Private method guessNumber.Login.byteArrToString(byte[]) is never called
      175:Use of non-localized String.toUpperCase() or String.toLowerCase() in guessNumber.Login.byteArrToString(byte[])
    UserNumberBean.java (8)
      34:Class guessNumber.UserNumberBean defines non-transient non-serializable instance field con
      35:Class guessNumber.UserNumberBean defines non-transient non-serializable instance field ps
```

FindBugs output

Dynamic tests, done using TestNG, revealed no significant changes for all classes and methods existing at the at the time of initial testing. Some of the methods that were failing the initial testing have always functioned perfectly in the actual app, so the problem must be occurring in the testing setup. Some new test cases were needed though, as a few significant methods were added as features were implemented. No new security issues were discovered using the TestNG dynamic testing.



```
6 tests passed, 3 tests failed.(0.904 s)
▼ Command line test Failed
    guessNumber.LoginNGTest.testSetPwd passed (0.006 s)
    guessNumber.LoginNGTest.testGetPwd passed (0.007 s)
    guessNumber.LoginNGTest.testSetUser passed (0.0 s)
    guessNumber.LoginNGTest.testGetUser passed (0.001 s)
    guessNumber.LoginNGTest.testSetMsg passed (0.002 s)
    guessNumber.LoginNGTest.testGetMsg passed (0.0 s)
  ▼ guessNumber.LoginNGTest.testValidateUsernamePassword Failed: java.lang.NullPointerException
    java.lang.NullPointerException
     at guessNumber.SessionUtils.getSession(SessionUtils.java:20)
     at guessNumber.Login.validateUsernamePassword(Login.java:73)
     at guessNumber.LoginNGTest.testValidateUsernamePassword(LoginNGTest.java:122)
  ▶ guessNumber.LoginNGTest.testCreateUsernamePassword Failed: java.lang.NullPointerException
  ▶ guessNumber.LoginNGTest.testLogout Failed: java.lang.NullPointerException
```

Example of partially failed testing.

From a user's perspective there are no security issues. Some features aren't implemented as nicely as they could be, but nothing crashes and there are no significant failures when using the app. However, on the shop page, the payPal link currently opens payPal and lets you add fake items to a cart. Since these items are listed as having a price of 1 cent, a user could potentially fill the cart with fake items and maybe somehow actually pay real money for them. Also on the shop page, all items are currently 'free'. In reality, they would be purchasable via payPal, and non-repudiation would become an issue. As it stands, shop functionality could be given a **Passed FSR (with exceptions)**.

As the developers we are aware of other issues. As mentioned previously, the hard coded database password is a significant issue, and would receive either a **Passed FSR (with exceptions)** or a **FSR escalation**. All information (tutorials and examples) we've found so far do the same thing we're doing. And to do something else at this point may require adding an admin access login, or other signficant change. So due to time constraints, the hard coded password will ultimately be **Passed FSR (with exceptions)**. Looking at the database setup itself, the current arrangement has the user's username in three different tables. This may or may not be an issue security wise, but limiting this data to one table would be a future goal. So database setup will be a **Passed FSR (with exceptions)**. Access to the DB seems secure, so will receive a **Passed FSR.**

3 Certify Release and Archive.

1) Make sure that all the necessary code to run your program is accessible on your online repository.

https://github.com/cdelp/WaifuV3
Yes, link is in User Guide https://www.youtube.com/watch?v=bCzaPmntJ2I&feature=youtu.be

(2) In the main folder, include a .pdf version of your final report, and name it "Secure Development Lifecycle.pdf".
(3) Ensure that your README is up to date.
(4) To complete the documentation, create GitHub wiki pages for: Home, Contact Us, and User Guide. The Home page should just include the name of the application, and 3-4 sentences about what it does. The Contact Us page should have the names of the group members and contact information (emails suffice). The User Guide should describe step by step how to use your program, given the code in the GitHub directory. This should be easy to follow, even for fairly non-technical people. Include screenshots.
(5) Finally, release your final version of the program. On your report, for this section, simply include a link to your GitHub release.
https://github.com/cdelp/WaifuV3/releases/tag/1.0.1

4 References

1) http://www.cisjournal.org/journalofcomputing/archive/vol5no8/vol5no8_8.pdf