Kevin Zhang

Application Security CS GY 9163

**Homework 4: Mobile Mess**

On NYU Classes, submit a link to your GitHub repository. The repository should be **private**, and you should add the instructor/TA's GitHub account as a contributor to give them access for grading. The repository should contain all of the files of the Android project, plus the text files BUG.txt and difference.txt.

For this section, your instructor is: **Kevin Gallagher**, GitHub ID `**kcg295**`. Your TA is: **Evan Richter**, GitHub ID `**evanrichter**`

Part 1: Setting up Your Environment

Android Environment

* Set up Android Studio on Windows 10
* Set up Android emulator, Pixel 3a, image R, API 30, x86 ABI, Target Android 11.0 (Google Play)
* Imported “GiftCardSite” project
* Emulator run successfully, able to start up virtual Pixel 3a

Part 2.1: It's all about intent – What’s the difference?

I have included difference.txt in the assignment repository; however, I have also included the answers to the questions below. Please note (#) refers to the specific questions referenced.

1. What are the two types of Intents?
2. Which of the two types of Intents are more secure?
3. What type of Intent is shown on lines 69 to 73 of SecondFragment.kt?
4. What type of Intent is shown on lines 68 to 70 of ThirdFragment.kt?
5. Which of these two Intents is the proper way to do an Intent?

a) Questions 1-5: The two types of Intents used for Android are Implicit Intents and Explicit Intents (1); in terms of security, Explicit Intents are much more secure, due to their definite specification of responding/interacting services. Implicit Intents do not allow identification of services - messages can be sent to wrong components (2). "Line 69-73 of "SecondFragment.kt" is an example of an Implcit Intent for a website, while lines 68-70 of "ThirdFragment.kt" is an Explicit Intent (3,4) - "Third Fragment.kt" and its Explicit Intent are the proper way, as they cause a web page external of the app to be opened (5).

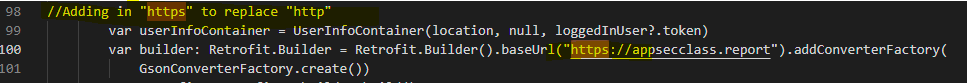
b) Fixing the incorrect Intent: In order to fix this, we modify "SecondFragment.kt" to use Explicit Intent by specifying the action "scrollingprofileactivity" rather than utilizing the "intent.actionall" key, avoiding ambiguity and not utilizing unecessary components. This is done with "var intent = Intent(activity, ProductScrollingActivity::class.jav".

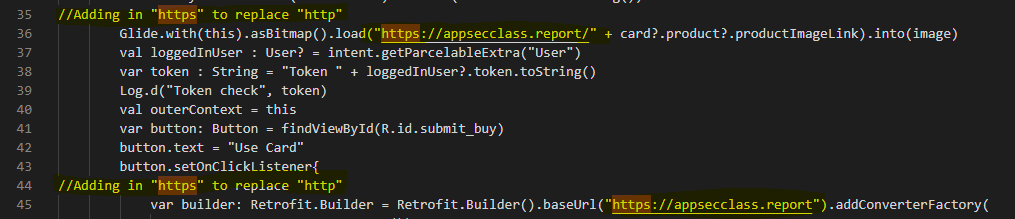
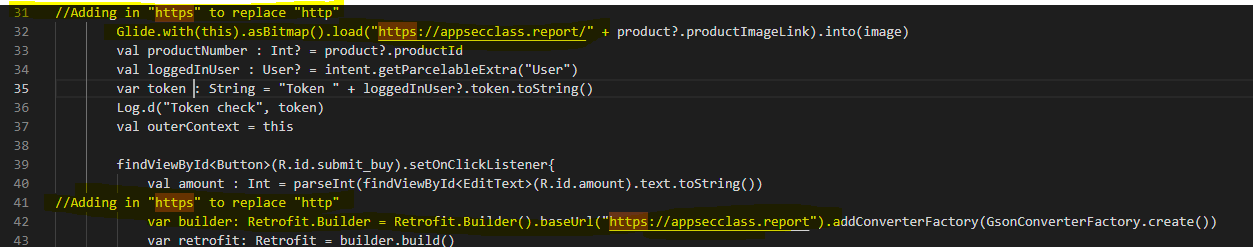
Part 2.2: It's all about intent – Shutting out the world

“AndroidManifest.xml” is flawed in that it allows other (potentially unverified) applications to use “GiftCard” to run Activities - patching this issue will secure a lateral vulnerability. I have modified “.UseCard”, “.GetCard”, “.ProductScrollingActivity”, and “.CardScrollingActivity” with android:exported=”false,” which specifically defines that no other app can launch this activity (even the Android system itself).[[1]](#footnote-1) Additionally, we also have added the XML attributes tool “tools,” which also allow error handling. This will allow the android:exported=”false” to fail safety. Please see accompanying HTML comments within the AndroidManifext.xml document attached in the assignment repository.

Part 3: Can you read me out there?

Please see below changes to the files to implement HTTPS. HTTPS provides additional security through TLS (SSL) public key encryption.[[2]](#footnote-2) This provides authentication and, like changing Intents from Implicit to Explicit, will harden the mobile application.

1. SecondFragment.kt – in segments referencing the <http://appseclcass.report> modified it to HTTPS (<https://appseclcass.report>) 
2. ThirdFragment.kt – in segments referencing the <http://appseclcass.report> modified it to HTTPS (<https://appseclcass.report>) 
3. CardScrollingActivity.kt – in segments referencing the <http://appseclcass.report> modified it to HTTPS (<https://appseclcass.report>) 
4. ProductScrollingActivity.kt – in segments referencing the <http://appseclcass.report> modified it to HTTPS (<https://appseclcass.report>)
5. UseCard.kt – in segments referencing the <http://appseclcass.report> modified it to HTTPS (<https://appseclcass.report>)

GetCard.kt – in segments referencing the <http://appseclcass.report> modified it to HTTPS (<https://appseclcass.report>)

1. CardRecyclerViewAdapter.kt – in segments referencing the <http://appseclcass.report> modified it to HTTPS (<https://appseclcass.report>)
2. RecyclerViewAdapter.kt – in segments referencing the <http://appseclcass.report> modified it to HTTPS (<https://appseclcass.report>)

Part 4: Oops, was that card yours?

There exists a vulnerability in the REST API that allows users to use GiftCards that do not belong to them. In a file called BUG.txt, explain why this vulnerability may be occurring, and how it can be fixed. Your explanation should be no larger than a paragraph.

You can start looking for this vulnerability in the following files:

1. UseCard.kt
2. CardInterface.kt

Think about how the application is telling the server which card to use, and how that may be problematic.

Part 5: Privacy is Important

**Grading**

Total points: 100

Part 2 is worth 30 points:

10 points for answering the questions.

10 points for fixing the correct intent.

10 points for closing the application to outside intents.

Part 3 is worth 20 points:

2.5 points for each file correctly modified to use HTTPS.

Part 4 is worth 20 points:

10 points for identifying the cause of the vulnerability.

10 points for describing a potential solution.

Part 5 is worth 30 points:

10 points for removing unneeded permissions.

10 points for removing metric collection API calls.

10 points for removing interaction with sensors.

1. Description ol android:exported=”false”: “https://stackoverflow.com/questions/49471423/android-manifests-androidexported-false-prevents-app-from-running-on-device” [↑](#footnote-ref-1)
2. Description of HTTP: “https://www.cloudflare.com/learning/ssl/why-is-http-not-secure/#:~:text=HTTPS%20is%20HTTP%20with%20encryption,uses%20HTTPS%20has%20https%3A%2F%2F.” [↑](#footnote-ref-2)