**Teaching Android Security: A Public Educational Activity of Vulnerable Android Applications**



**Abstract**

This paper describes a public educational activity to assist in the instruction of both students and developers in creating secure Android apps. Our set of activities includes example vulnerable applications, information about each vulnerability, steps how to repair the vulnerabilities, and information about how to confirm that the vulnerability has been properly repaired. Our goal is to make these activities available to other instructors for use in their classrooms ranging from the K-12 to university settings. A secondary goal of this project is that it also fosters interest in security, and computing.

**Body**

Security is hard. Teaching security can be even harder. The mobile revolution has allowed anyone with a basic understanding of development to upload their applications (app) to an app store, making them available to millions of potential users. With this great power comes great danger as inexperienced developers have the capability to create vulnerable apps that can affect millions of users. More so, even experienced developers make mistakes due to the challenging nature of creating secure software.

Developers frequently create vulnerable software for a wide range of reasons, some of which include ignorance of how to create secure apps, simple developer error, or a lack of understanding of the importance of secure app development. In order to help educate developers about not only how to create secure apps, but the importance of secure app development, we have created a public educational set of Android apps. Each example contains a clear demonstration of the negative ramifications of the vulnerability, steps to repair the vulnerability, and then actions to ensure that it has been resolved.

We have two primary goals for the project. The first is that we want to educate Android developers about the specific example vulnerabilities in our study. Secondly, we would like to demonstrate the importance of security on a more general level to a diverse set of Android developers from different experience levels. Our goal is to get developers, from all experience levels, more interested in security and to see how much of an important topic it is.

There were several primary concerns which lead us to create this oracle, including:

* **Development Effort:** Creating activities, especially those with vulnerabilities can be a long, and difficult process, something which many instructors do not have the necessary resources to build. Creating activities such as these will free instructors to do what they do best, teach. Additionally, since these activities will be refined through regular usage, instructors can be more confident that any issues will have already been resolved.
* **Required Skillset:** Teaching students and developers to create secure software is a difficult process. Not all instructors teaching mobile development are expected to be adequately prepared to create a diverse, and informative set of activities such as we have created.

Our project has several goals that we want to achieve:

* **Ease of use:** All activities should be usable ‘out of the box’ by instructors. Additionally, students and developers working through the activities on their own should be able to do so with as little challenges as possible due to the documentation and clear instructions sets which are provided with each of the activities.
* **Relevance:** One way of making students interested in software security is by demonstrating its relevance. We have found that creating relevant, real world projects help to foster student interest in the activity. Students want to work on exercises with real world applicability. Each of our activities contains relevant examples from the real world, and contains steps to clearly recreate each vulnerability and demonstrate its relevance.
* **Meet a diverse range of skillsets:** Not all students or developers are at the same skill level, and security is important to all experience and skill levels. Additionally, anyone can learn about security and mobile development; ranging from elementary school students to retirees. We wanted to create a way to assist this diverse range of students. In order to meet this demand, our activities are designed to assist students and developers of all experience levels, and of all ages.
* **Ease of Use:** We wanted to make the activities as easy to use as possible. We did not want its users to struggle with setup or any other issues. We wanted them to focus on the task at hand, learning about the vulnerability and how to protect your apps from it. In order to address this issue, all activities have a minimal amount of preparation, and have clearly defined setup instructions.

We have created 10 publicly available activities, which can be found, on our project website: **hidden**. This list will continue to grow, both through our own development efforts in addition from submissions from the external sources. A secondary benefit of these exercises is that it will demonstrate the importance of security to students and developers and further foster interest in these areas. In order to support a diverse set of users, activities are geared for beginners to advanced users. Our apps were created by an experienced Android developer with over 1 million app downloads on Google Play alone.

**Activities**

Although our list of activities is growing, we currently have 10 vulnerability examples ranging from basic examples as proper Intent protection to more complicated activities such as correct use of content providers. The process outline of each activity is outlined in Figure 1.



Figure 1: App Repair Process

Each of the exercises contains:

1. Mobile apps which contain well defined vulnerabilities.
2. Documentation about the adverse effects of the vulnerabilities and how they may be exploited.
3. Step by step documentation how to repair the vulnerabilities.
4. Instructions how to verify that the vulnerability has been repaired.

Activities begin with providing the user background about the specific vulnerability being targeted in each exercise. This serves to provide the student the necessary background of when, why, and how the vulnerability may occur. Whenever possible, students are also provided a real world example of occurrences of vulnerability and where they occurred in specific apps. We also discuss some basic reasons about why the vulnerability occurs and common developer mistakes which lead to the vulnerability.

Each activity has an associated app, which contains an instance of the discussed vulnerability. These apps, which were created specifically for these exercises, are typically very simple with the sole intention of conveying the example vulnerability. Using the provided instruction set, students may then re-create the vulnerability to demonstrate its possible negative ramifications. An example recreation of a vulnerability is shown in Figure 2. In some cases, activities also utilize free, 3rd party tools such as Fiddler[[1]](#footnote-1) as shown in Figure 3 to demonstrate vulnerabilities or show that the vulnerability has been repaired.

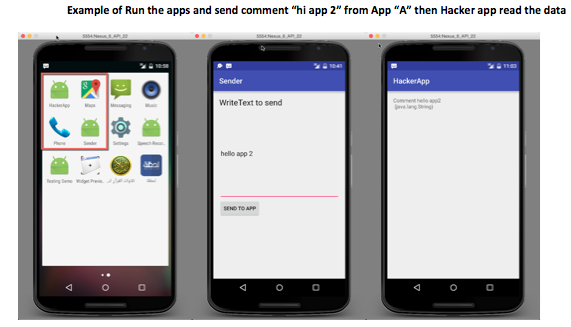


Figure 2: Example Vulnerability Demonstration

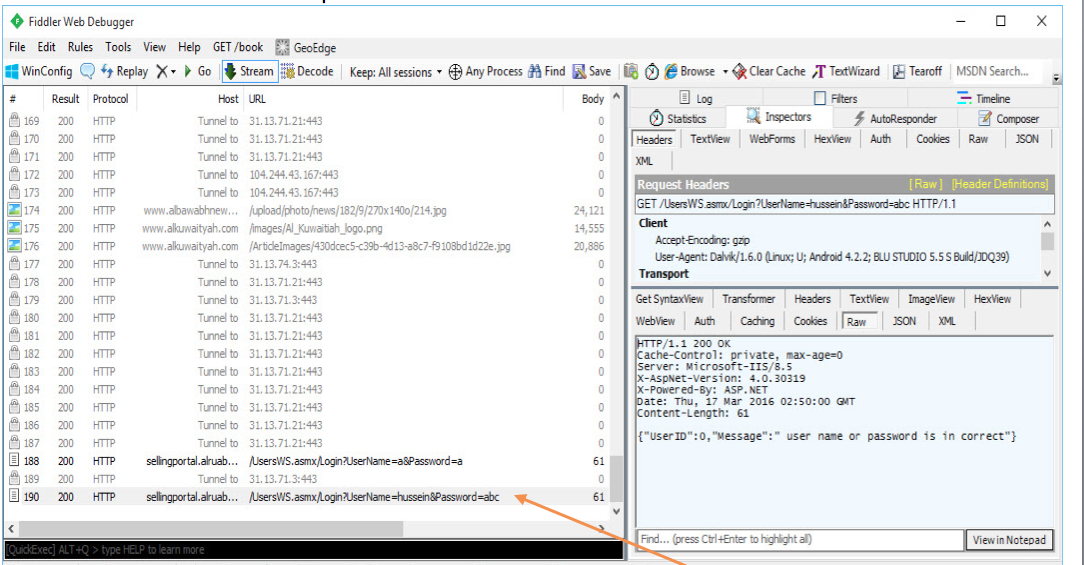


Figure 3: Example Vulnerability Demonstration Using Fiddler

The user is then provided background information how to repair the vulnerability, which includes any relevant code snippets and proper information about why the introduced defensive coding practices protects against the vulnerability. Our documentation also provides clear steps about how to repair the vulnerability inside of the provided app.

The final step has the user attempt to recreate the vulnerability in the app. If properly conducted, the vulnerability should not longer exist. This concluding portion of the activity provides several important benefits to the user including demonstrating the importance and proper procedures of basic security testing, along with providing a sense of accomplishment for the user. Here are some of the activities that we have created, along with their recommended experience level:

1. **AdLibraries** (Advanced):Ad library could use the permissions that given to the app which contains the Ads library, even the people did not give this permission to Ads library. This can open up various security and privacy issues within the app.
2. **Android Javascript** (Medium)**:** This demonstrates the negative implications of using Javascript in Webview to pass data from an Android app to a server. This is considered bad practice because anyone could use malicious Javascript code on their website to gain private user information within the app.
3. **Broadcast** (Medium)**:** Broadcast data sent by the app is easy to access by any app in the system, so when we do Broadcast to specific apps, we have to encrypt the data. Unencrypted data could be accessed by unintended apps, which could lead to serious security and privacy issues within the app.
4. **Activities Access** (Beginner):Security issues arise when people try to access specific unauthorized activity. An example could be like in bank app, where users try to access a balance management activity without ever logging to the system.
5. **Content\_Providers** (Advanced):Content providers share data between apps, and any app in the system can access the Content\_Providers database. This means that data stored here must be kept secure and encrypted so that it can only be read only from an authorized app.
6. **Data Storage** (Medium):When an app does not secure storage data like Files, Shared references, SQLite, it could be read very easy from anyone. This means that important information stored in these files like, such as a database connection, must also be encrypted.
7. **DataOverHTTP** (Medium):Data that moves over an HTTP connection which is not encrypted is vulnerable to Man in the Middle attacks. One example of this is credit card information. Information passed over an HTTP connection must be encrypted to remain secure from unintended listeners.
8. **DOS** (Medium):Denial of Service (DoS) attacks is a common problem with Android, because an adversary could create many HTTP requests going to a specific server. These requests must be managed to make them less vulnerable to these types of attacks.
9. **Intent Protection** (Beginner):Android uses Intents to pass data between apps. Examples include Facebook and Facebook Messenger. Passing data between these apps may be easily ready by hacker apps. This module explains how to protect information being sent via Intents between apps.
10. **XML** (Beginner):XML is very easy to read using reverse engineering, so we have to avoid saving important information like Ads code, or Map Code

Although there are innumerable families of Android vulnerabilities, we have initially selected these for several factors including their prevalence in existing apps, their ease of demonstration, and negative ramifications.

**Conclusion**

We have created a publicly available instruction set of vulnerable Android apps which includes 10 groups of vulnerabilities. Our goal is for instructors to use these activities in a diverse set of courses, and may be used by a diverse set of skill levels ranging from beginning level developers, to more advanced levels. All course material may be found on our project website: **hidden**

**Acknowledgements**

Elements of this work are sponsored by a SIGCSE Special Projects Grant.

**Appendix**

% Not sure if this is necessary

XXXXXX

**Categories & Subject Descriptors**

XXX

**General Terms**

XXX

**Keywords**

XXX

References

(Test)

# References:

Test. (n.d.). *Test*. Retrieved from Test: Test

<<<Temp saving>>>

**Related Work**

Although this project is the first <how to word this?>

% Other work that supports security/mobile education

% Work that has identified vulnerabilities

% How have vulnerabilities in Android/mobile development been addressed?

% What websites contain exercises with these, and how does our work differ

There are several well-known resources for instructing students in creating secure software. However, our project is the first we are aware of that are tailored specifically to mobile app development, and provides clear demonstrations of vulnerabilities, how to recreate and then repair the vulnerability. A few popular resources which focus on other aspects of secure software development include DVWA[[2]](#footnote-2), Safecode[[3]](#footnote-3), as well as guides for creating secure software such as the one provided by OWASP[[4]](#footnote-4).

%%% I am not sure how much I should get into the academic side of things since I don’t think the paper is oriented to that direction

**Uses** %Maybe make this a subsection

XXXXXX

% In addition to various classes….

% Outreach activities, especially those for undrepresented students. Can serve as a mechanism to get students not involved in security, but in computing as well.

% Learning environments: Both the classroom and self-taught

**Future Work Work**

XXXXXX

% Create iOS examples

% Continue to grow the example list

% Plan on holding workshops to both instruct educators how to use these activities, but in creating new activities as well.

**Challenges**  (not sure if we should keep this section)

XXXXXX

1. http://www.telerik.com/fiddler [↑](#footnote-ref-1)
2. http://www.dvwa.co.uk [↑](#footnote-ref-2)
3. https://training.safecode.org [↑](#footnote-ref-3)
4. https://www.owasp.org/index.php/OWASP\_Secure\_Coding\_Practices\_-\_Quick\_Reference\_Guide [↑](#footnote-ref-4)