COURSE RECAP AND EXAM INFO
Lecture 36

COMPSCI 702
Security for Smart-Devices

Muhammad **Rizwan** Asghar

June 3, 2021

THE UNIVERSITY OF AUCKLAND
NEW ZEALAND

COURSE STRUCTURE: PART 1

- Lectures (Week 1 to Week 7-8)
  - Introduction (<1 week)
  - Access control (<1 week)
  - Android security (<4 weeks)
  - iOS security (2 weeks)

COURSE STRUCTURE: PART 2

- Individual seminars by students (3 weeks)
  - 21 seminar presentations
  - Each student chose one from 15 research articles
  - 13 unique research articles presented
    - Only these 13 will be examinable!
- Group presentations (1 week)
  - 6 groups
- Course recap and exam info
  - Focus of this lecture!

EXPECTED FROM STUDENTS

- Attend lectures and presentations
- Active class participation
- Present a research article
- Work in a team on a group project
  - Come up with ideas to obfuscate apps
  - Reverse engineer apps developed by other groups
  - Group size 3-4
  - 6 project reports

SUPPORT DURING THIS COURSE

- Feedback on your seminars
- Feedback on development phase
- Feedback on challenge phase

FUTURE POSSIBLITIES

- Extending report as a research article
- Dissertation, thesis, or project
- Internship
- Job

LEARNING OUTCOMES

- Give basic advice on securing smart devices (Themes 1-6)
- Criticise and appreciate technical literature on mobile security (Themes 1-5)
- Demonstrate technical skills to increase security of smart devices (Themes 1-6)
- Prepare and deliver an oral presentation on an advanced topic in mobile security (Themes 1,2,4,5)
- Develop novel problem solving and research-informed ideas (Themes 1-6)

All themes are defined in graduate profiles

ASSESSMENTS

- 10% individual seminar
  - 5% introduction
  - 5% solution
- 30% group project
  - 4% app
  - 15% development phase
  - 10% challenge phase
  - 1% project presentation
- 60% final exam
  - Will discuss this next

OUR FEEDBACK

- Project development
- Challenge phase
- Seminars
  - Seminars are being evaluated
  - Grades and comments will be provided soon
- Group projects
  - Under evaluation

YOUR FEEDBACK: SET EVALUATIONS

- 23 students
- 5 responded
- Response rate: <22%
- Thanks to those who already responded!
- Request all other students to help us in achieving our target of 80% by Sunday, June 6

FINAL EXAM

- Written exam (short essay-based)
- Standard 2 hours
- Study material for final exam
  - Lecture slides and resources
  - Projects and seminars
  - Presentations and class discussions
- 8 questions
- 60 marks

EXAMINABLE 13 ARTICLES

1. [Li-WWW20] (3)
2. [Shen-NDSS21] (3)
3. [Kim-NDSS21] (2)
4. [Lei-NDSS21] (2)
5. [Lu-CCS20] (2)
6. [Tang-USENIX20] (2)
7. [Tuncay-USENIX20] (2)
8. [Weir-USENIX20] (2)
9. [Chen-USENIX20] (1)
10. [Hu-WWW20] (1)
11. [Mahmud-USENIX20] (1)
12. [Mi-NDSS21] (1)
13. [Ruge-USENIX20] (1)

SAMPLE QUESTION (1)

- *Imagine that you are building a new app store, say AlphaStore.*

  a) In order to protect intellectual property rights of app developers, we consider that AlphaStore can accept obfuscated apps. Discuss the major challenge in analysing obfuscated apps submitted to AlphaStore.

  b) Outline a solution to address the challenge you identified.

  c) For inspecting an app submitted to AlphaStore, what would be your design choices when it comes to app execution environment (i.e., real device or emulated environment) and analysis nature (i.e., manual analysis or automated analysis). Justify your answer.

ANSWER (1)

a) The main challenge is to run analysis tools that might not be able to fully understand and decide whether the app submitted to AlphaStore is benign or malicious...

b) To properly analyse the apps submitted to AlphaStore, reverse engineering and deobfuscators should be put in place. After a round of these tools, AlphaStore would be able to use state of the art static and dynamic analysis tools...

c) The real device is an option to understand actual behaviour as otherwise malicious apps can suppress malicious behaviour; whereas, emulated environment is more scalable, but might not be able to properly inspect the apps. The manual analysis might be thorough, not scalable. The automated analysis is scalable but might not be able to completely understand the code. A hybrid approach in both cases might be a possible solution.

SAMPLE Q&A (2)

- ***Question:*** *From app distribution point of view, there are two models: an app could be obfuscated once for all the users, or a unique instance of obfuscated app could be created for each user. Analyse both models from the point of view of debugging and updates.*

- **Answer:** The first model is easier to be updated and debugged, but not that secure, since the attacker just needs to reverse engineer only a single instance in order to break the app. However, in the second model, even if the attacker is able to reverse engineer a specific instance of the app, the app could be still secure as each instance is unique, but it is challenging for debugging and updates as the developer has to work with each instance.

SAMPLE Q&A (3)

- ***Question:*** *Design a mechanism to mitigate mobile app squatting.*

- **Answer:**
  Apply app squatting-generation models to produce potential candidates. Next, incentivise users to mark potentially real/fake apps…
  App stores can also actively look for potentially fake apps based on the number of downloads, star rating, reviews, …

CANVAS, COURSE WEBSITE, AND PIAZZA

- Canvas for almost everything
- All lectures were recorded and recording links were distributed through Canvas
- Canvas (week/lecture wise organisation)
  - https://canvas.auckland.ac.nz/courses/60529/modules
- Course website (topic wise organisation)
  - https://www.cs.auckland.ac.nz/courses/compsci702s1c
- Piazza
  - https://piazza.com/aucklanduni.ac.nz/semester12021/compsci702

Questions?

Thanks for your attention!