Kick-off: Mobile Application Security

Chair for IT Security / I20
Prof. Dr. Claudia Eckert
Technical University of Munich

Dr. Julian Schütte
julian.schuette@aisec.fraunhofer.de

Konrad Weiss
konrad.weiss@aisec.fraunhofer.de

Alexander Küchler
alexander.kuechler@aisec.fraunhofer.de

July 16, 2019

Outline

Deliverables and Grading

The seminar will be organized as a scientific conference.

- Report (50%)
  - Scientific paper with exactly 10 pages in length
  - We provide a LaTeX template
  - Shall cover relevant work in that area, clear structure, clarity of presentation, proper bibliography & citations
  - Add your own thoughts, discussion
- Review (20%)
  - Each of you creates two anonymous reviews
  - Template will be provided
  - Approximately one page in Latex
- Presentation (30%)
  - 30 minutes presentation
  - 15 minutes discussion

Time Table

16.07.2019	[Today] Topic Presentations
After matching	Start of topic assignments
02.09.2019	Introduction to scientific writing (optional)
27.10.2019	Submit your outline + preliminary draft (80% of overall content)
29.10.2019	Meeting: Intermediate review and discussion (10:00 - 16:00)
12.12.2019	Submit your paper
21.12.2019	Submit your reviews
20.01.2020	Submit your rebuttal + camera-ready-version + slides
27.+28.01.2020	Meeting: Presentations and discussion (9:00 - 16:00)

Topics

- Interprocedural Analysis With Weighted Pushdown Systems
- Automated Dynamic Testing
- Dynamic Taint Analysis
- Intermediate Representations for Static Binary Analysis
- Practical IR Lifting of iOS apps: far from trivial?
- UI-Attacks
- Evolution of Mobile Malware Behavior
- Security Implications of Web Technologies in Mobile Applications
- Security Issues of 3rd Party Libraries in Android Applications
- App Integrity Assurance
- Application collusion and confused deputy attacks
- Covert Channels in Android
- Discover Privacy Violations in Mobile Apps
- (Semi-)Automatic Deobfuscation
- SE-Linux

Interprocedural Analysis With Weighted Pushdown Systems

Fraunhofer AISEC

- Understand & describe the problem of precise interprocedural program analysis, esp. in Android apps
- Understand weighted pushdown systems & the existing algorithms (such as `post-`$\star$, `pre-`$\star$) to make use of pushdown systems for program analysis
- Discuss the application of pushdown systems to malware analysis of mobile apps

- Initial literature
  - Reps et al.: "Precise interprocedural dataflow analysis via graph reachability"
  - Reps et al.: "Weighted pushdown systems and their application to interprocedural dataflow analysis. In Science of Computer Programming"
  - Lal et al.: "Extended Weighted Pushdown Systems"
  - Liang et al.: "Sound and precise malware analysis for android via pushdown reachability and entry-point saturation"

Automated Dynamic Testing

- What are the challenges with dynamic testing?
- Approaches of automated dynamic vulnerability finding
- Assess & classify approaches

- Initial literature
  - Hao et al.: "PUMA: Programmable UI-Automation for Large-Scale Dynamic Analysis of Mobile Apps"
  - Bhoraskar: "Brahmastra: Driving Apps to Test the Security of Third-Party Components"
  - Rasthofer et al.: "Harvesting Runtime Data in Android Applications for Identifying Malware and Enhancing Code Analysis"
  - Schwarz et al.: "All You Ever Wanted to Know about Dynamic Taint Analysis and Forward Symbolic Execution (but Might Have Been Afraid to Ask)"
  - Anand et al.: "Automated Concolic Testing of Smartphone Apps"

Dynamic Taint Analysis

- How does DTA work and what is it good for?
- Platform-level vs. application-level DTA
- Challenges in getting DTA right
- Propose ways to break DTA

- Initial literature
  - Enck et al.: "TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones"
  - You et al.: "TaintART: A Practical Multi-level Information-Flow Tracking System for Android RunTime."
  - Schütte et al.: "Practical Application-Level Dynamic Taint Analysis of Android Apps"
  - Schwarz et al.: "All You Ever Wanted to Know about Dynamic Taint Analysis and Forward Symbolic Execution (but Might Have Been Afraid to Ask)"

Intermediate Representations for Static Binary Analysis

- Intermediate Representations (IR) are not only used by compilers but also for reverse engineering
- The idea is simple and there are some well-known IRs like *VEX*, *LLVM*
- However, researchers tend to invent their own IRs and so there is *ESIL*, *BAP*, *Scratch*, *Binnavi REIL*, *rev.ng*, *HHVM IR*, *QBE IR*, *TCG IR*, etc.
- What is the point of this and what are the properties that are relevant to IRs?
- What are possible drawbacks of LLVM and VEX and what tried the others to make better?

- Initial literature
  - Kim et al.: "Testing Intermediate Representations for Binary Analysis"
  - Märkl: "Case Study on LLVM as suitable intermediate language for binary analysis", TUM Technical Report

Practical IR Lifting of iOS apps: far from trivial?

- IR lifting: process of reverting a binary into a high level IR
- In theory it sounds simple and there are various tools available (*BAP*, *angr2*, *McSema*, *radare2*).
- In practice, however, things are not that simple anymore
- Research the exact process of binary lifting (*loader* $\rightarrow$ *dynlib resolver* $\rightarrow$ *disassembler* $\rightarrow$ *symbols reconstruction* $\rightarrow$ *CFG reconstruction* $\rightarrow$ *instructions lifting* $\rightarrow$ *IR compilation*)
- Walk through a tool like McSema or angr2 and try to lift an iOS app (practical part)
- Discuss your experience, suggest improvements

- Initial literature
  - Mcsema: Static translation of x86 instructions to llvm. A Dinaburg (`https://github.com/trailofbits/mcsemahttps://www.trailofbits.com/research-and-development/mcsema/`)
  - `https://lowlevelbits.org/parsing-mach-o-files/`
  - Egele et al.: "PiOS: Detecting Privacy Leaks in iOS Applications"
  - Lattner and Adve: "LLVM: A Compilation Framework for Lifelong Program Analysis & Transformation"

UI-Attacks

- Which attacks have been published on the UI of apps?
- Why is it interesting for an attacker to gain knowledge of the UI?
- How can apps (or the OS) protect against such attacks?

- Initial literature
  - Niemitz, Schwenk: "UI redressing attacks on android devices"
  - Abdow et al.: "UiRef: Analysis of Sensitive User Inputs in Android Applications"
  - Fernandes et al.: "Android UI Deception Revisited: Attacks and Defenses"
  - Fratantonio et al.: "Cloak and Dagger: From Two Permissions to Complete Control of the UI Feedback Loop"

Evolution of Mobile Malware Behavior

- Which categories of malware for mobile devices does exist?
- What are the techniques used by mobile malware? How do they compare e.g. to Windows malware?
- What are the trends when viewing the malware on a historic timeline?
- Focus on Android

- Initial literature
  - Weichselbaum et al.: "ANDRUBIS: Android Malware Under The Magnifying Glass"
  - Tam et al.: "The Evolution of Android Malware And Android Analysis Techniques"
  - Rasthofer et al.: "How Current Android Malware Seeks to Evade Automated Code Analysis"
  - Felt et al.: "A Survey of Mobile Malware in the Wild"
  - Andronio et al.: "HelDroid: Dissecting and Detecting Mobile Ransomware"

Security Implications of Web Technologies in Mobile Applications

Fraunhofer AISEC

- Types of apps: Native vs. Hybrid vs. Web Apps
- Which risks and vulnerabilities are intruduced by writing hybrid apps instead of native apps?
- What are common problems of hybrid mobile apps?

- Initial literature
  - Yang et al.: "Risk Analysis of Exposed Methods to JavaScript in Hybrid Apps"
  - Yang et al.: "Study and Mitigation of Origin Stripping Vulnerabilities in Hybrid-postMessage Enabled Mobile Applications"
  - Zuo et al.: "Automatically Detecting SSL Error-Handling Vulnerabilities in Hybrid Mobile Web Apps"
  - Mutchler et al.: "A Large-Scale Study of Mobile Web App Security"

Security Issues of 3rd Party Libraries in Android Applications

- Why is library detection important (e.g., which security issues can be present in libraries)
- Explore available approaches for (resilient) library detection
- Show the resilience and accuracy of available approaches w.r.t. detected versions / security issues
- Suggest means to increase detection accuracy and/or resilience of an existing approach

- Initial literature
  - Backes et al.: "Reliable Third-Party Library Detection in Android and its Security Applications"
  - Derr: "LibScout: Third-party library detector for Java/Android apps"
  - Titze et al.: "Ordol: Obfuscation-Resilient Detection of Libraries in Android Applications"

App Integrity Assurance

- How can I check that the app I published was not modified before running? (app modification checking)
- How can I check that critical parts of my code (e.g., a native library) are executed only by my app? (code lifting protection)

- Initial literature
  - Anad: "Securing Android Code Using White Box Cryptography and Obfuscation Techniques"
  - Subhadeep et al.: "Analysis of Software Countermeasures for Whitebox Encryption"
  - Dagit et al.: "Code re-use attacks and their mitigation"
  - Jung et al.: "Repackaging Attack on Android Banking Applications and Its Countermeasures"
  - Zhou et al.: "DIVILAR: Diversifying Intermediate Language for Anti-Repackaging on Android Platform"
  - `https://android-developers.googleblog.com/2018/06/google-play-security-metadata-and.html`

Application Collusion and Confused Deputy Attacks

- What are Application Collusion and Confused deputy attacks on Android.
- How do they relate to the Android permission system, intents etc.
- How can they be detected or prevented.
- What are the necessary attack requirements.

- Initial literature
  - Marforio et al.: "Application Collusion Attack on the Permission-Based Security Model and its Implications for Modern Smartphone Systems"
  - Kalutarage et al.: "Towards a threat assessment framework for apps collusion"
  - Xu et al.: "AppHolmes: Detecting and Characterizing App Collusion among Third-Party Android Markets"
  - Wu et al.: "PaddyFrog: systematically detecting confused deputy vulnerability in Android applications"

Covert Channels in Android

- What are covert channels in Android.
- What external channels exist, focus on internal channels.
- What are covert channels used for.
- How can covert channels be detected.

- Initial literature
  - Caviglione et al.: "Seeing the Unseen: Revealing Mobile Malware Hidden Communications via Energy Consumption and Artificial Intelligence"
  - Lalande and Wendzel: "Hiding Privacy Leaks in Android Applications Using Low-Attention Raising Covert Channels"
  - Hansen, Hill and Wimberly.: "Detecting Covert Communication on Android"
  - Urbanski et al.: "Detecting local covert channels using process activity correlation on android smartphones"

Discover Privacy Violations in Mobile Apps

- Identify and list personally identifiable information that can be collected on Andoid and/or iOS
- Collect and evaluate methods for automated analysis of mobile apps to identify privacy concerns (e.g. Taint analysis)
- Describe counter-measures built into Android/iOS and/or provided by third party apps

- Initial literature
  - Mumtaz et al.: "Critical review of static taint analysis of android applications for detecting information leakages"
  - Enck et al.: "TaintDroid. An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones"
  - Egele et al.: "PiOS. Detecting Privacy Leaks in iOS Applications"
  - Wang et al.: "DroidContext. Identifying Malicious Mobile Privacy Leak Using Context"
  - Wu et al.: "Efficient FingerprintingBased Android Device Identification With Zero Permission Identifiers"

(Semi-)Automatic Deobfuscation

- Explore available research and tools for (semi-)automatic deobfuscation
- Analyze the quality of deobfuscation results for state-of-the-art-obfuscated applications (or custom samples)
- Show limits of deobfuscation
- Explain what a "perfect" deobfuscator may look like

- Initial literature
  - Karnick et al.: "A Qualitative Analysis of Java Obfuscation"
  - Klein, David: "Automating Removal of Java Obfuscation"
  - Leskov, Dmitry: "Protect Your Java Code - Through Obfuscators And Beyond"
  - Macbride et al.: "A Comparative Study of Java Obfuscators"
  - Sun, Sam: Deobfuscator (`https://github.com/java-deobfuscator/deobfuscator`)

SE-Linux

- What is SE-Linux?
- Why is it useful for Android/Android apps?
- What are its limitations?

- Initial literature
  - Chen et al.: "Analysis of SEAndroid Policies: Combining MAC and DAC in Android"
  - Smalley and Craig: "Security Enhanced (SE) Android: Bringing Flexible MAC to Android"
  - Shabtai et al.: "Securing Android Powered Mobile devices using SELinux"
  - Sambare et al.: "Securities in Android using SELinux"

Topic assignment

After matching phase (finishing 30.07.2019):

- Deregistration possible until 06.08.2019 without penality or brace yourself for a 5.0
- We'll ask you to send your 3 top choices via email
- You may add a letter of motivation to emphasize your top choice
- We'll assign topics to students with your input

Getting Started

- Objective: Get a comprehensive overview of the topic
  - Initial literature serves as a basis
  - Extension will be necessary
  - Check Sources, follow-up work, and related publications
  - Prioritize, classify, be critical
  - Keep in touch with your supervisor
- Make an outline
  - State your research question
  - Condense & review state of the art
  - Bring in your contribution
  - Provide an outlook to your fellow researchers
- Further info on writing & preparing talks will follow
  - Optional info session on writing a scientific paper
  - We give you information for every phase
- Language: English

Q&A ?