ΟΙΚΟΝΟΜΙΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ ΑΘΗΝΩΝ

ATHENS UNIVERSITY OF ECONOMICS AND BUSINESS

Athens University of Economics and Business

Department of Management Science and Technology

Undergraduate Thesis

Behavioral Profiling of Popular Messaging Apps Using Kernel-Level Tracing with ML Techniques

Student:

Foivos - Timotheos Proestakis

Student ID: 8210126

Supervisors:

Prof. Diomidis Spinellis

Dr. Nikolaos Alexopoulos

Submission Date:

April 5, 2025

Abstract

This thesis examines kernel-level tracing techniques to create behavioral profiles of popular messaging applications using Machine Learning. The main goal is to analyze the operational characteristics of such apps and employ ML algorithms to detect patterns regarding security. The study covers topics such as kernel-level data collection, big data processing and analysis, and the design of ML models for behavior identification and classification.

Acknowledgments

I would like to express my heartfelt gratitude to my supervisors, Prof. Diomidis Spinellis and Dr. Nikolaos Alexopoulos, for their invaluable guidance, insightful feedback, and continuous support throughout the duration of this thesis. Their expertise and encouragement were instrumental in the successful completion of this work.

I would also like to sincerely thank my exceptional fellow students, Vangelis Talos and Giannis Karyotakis, for their contribution, collaboration, and for being true companions in this academic journey.

A special thanks goes to my family, whose unwavering support, both emotional and practical, made this endeavor not only possible but also deeply meaningful. Their presence and encouragement were a constant source of strength.

Contents

Chapter 1

Introduction

Smartphones have become an integral component of modern society, with the number of global users surpassing 5 billion and continuing to grow rapidly [1]. Among the dominant mobile platforms, Android—an open-source operating system developed by Google—holds a stable global market share of approximately 75% [2]. Its open-source nature, flexibility, and widespread adoption have cultivated a vast ecosystem of applications that enhance user productivity and social interaction across various domains.

Among these applications, messaging platforms such as WhatsApp, Telegram, Facebook Messenger, and Signal have gained significant popularity, playing a central role in both personal and professional communication. However, the ubiquitous use of smartphones for such purposes has led to the accumulation of sensitive personal data on user devices, including photos, contact lists, location history, and financial information, thereby raising serious privacy and security concerns [3].

Incidents such as Facebook's unauthorized collection of SMS texts and call logs from Android devices [3] underscore the vulnerabilities within existing mobile ecosystems. In response, regulatory frameworks like the General Data Protection Regulation (GDPR) and national laws such as the UK Data Protection Act 2018 aim to enforce principles of transparency, data minimization, and user consent in data processing [4, 5].

Despite these legislative efforts, Android's current permission management system remains insufficient. Users frequently misinterpret the scope and implications of the permissions they grant, inadvertently exposing sensitive data to misuse [6, 7].

To address these challenges, it is essential to analyze application behavior—that is, the actual operations performed by an app, both in the foreground and background. Research has shown that discrepancies often exist between user expectations and actual app behavior, with applications executing hidden or unauthorized tasks [**?**, 8]. Many detection techniques rely on the assumption that user interface (UI) elements accurately represent application functionality, an assumption that is not always valid [9].

Behavioral analysis methods are typically divided into static and dynamic approaches. Static analysis examines application code without execution, identifying known malicious patterns. However, it is susceptible to evasion through obfuscation and polymorphism [10, 11]. In contrast, dynamic analysis evaluates applications during runtime, monitoring behaviors such as system calls, resource consumption, and network activity [12, 13]. Among these, system call analysis is particularly valuable, offering fine-grained visibility into application interactions with hardware and OS-level services [14].

Kernel-level tracing is a powerful form of dynamic analysis, capable of capturing low-level system interactions with high precision. Android is built on a modified Linux kernel that orchestrates resource management and system processes via system calls [15]. Tools such as `ftrace` and `kprobes` enable developers and researchers to trace kernel-level function calls, execution flows, and resource usage [16, 17].

`Ftrace` is a built-in tracing utility within the Linux kernel, optimized for performance and capable of monitoring execution latency and function call sequences. `Kprobes`, on the other hand, allows for dynamic instrumentation of running kernels, enabling targeted probing of specific code locations during runtime [18].

Applying kernel-level tracing to messaging applications, however, introduces unique technical challenges. These apps typically exhibit complex, multi-threaded behavior, frequent background processing, and diverse interactions with system resources. Accurately profiling such behavior requires collecting and interpreting high-volume, high-resolution kernel data [19, 20].

Despite the growing research interest in Android security and behavioral analysis, existing work has primarily focused on general application profiling or malware detection. Few studies have concentrated specifically on behavioral profiling of messaging apps using kernel-level data [21]. Meanwhile, recent reports concerning the usage of secure messaging apps such as Signal by government and military officials have emphasized the urgent need for transparent, robust behavioral analysis mechanisms [22].

To address these gaps, this thesis proposes a structured methodology for profiling the behavior of popular messaging applications on Android through kernel-level tracing using `ftrace` and `kprobes`. The proposed approach integrates Machine Learning techniques to process and classify behavioral patterns, aiming to enhance security diagnostics, user privacy, and system transparency.

1.1 Motivation and Problem Statement

Motivation The motivation behind this research arises from the necessity to bridge existing gaps between user expectations, regulatory compliance, and the actual operational behavior of popular messaging applications. Messaging apps process extensive personal data, creating substantial risks related to privacy violations and security breaches. Recent incidents involving unauthorized data collection by prominent messaging applications, along with revelations about governmental use of supposedly secure messaging platforms, underscore significant concerns regarding transparency and user trust.

Problem Statement Current literature lacks comprehensive kernel-level behavioral analyses of messaging applications, leaving critical privacy and security risks inadequately addressed. Thus, this research seeks to systematically explore kernel-level behaviors to enhance transparency, improve user trust, and provide rigorous technical evaluations of messaging applications' privacy implications.

1.2 Research Objectives

The specific research objectives addressed in this thesis are categorized as follows:

Primary Objectives

- Record the actual kernel-level behavior of widely used messaging applications.
- Identify potential violations of the principle of data minimization.
- Analyze mismatches between granted permissions and real-time resource usage.
- Detect unauthorized or hidden access to sensitive user data.
- Compare the behavioral profiles of privacy-focused apps (e.g., Signal) and more commercial alternatives.

Analytical and Technical Sub-Objectives

- Develop a tracing and profiling framework using ftrace and kprobes.
- Classify system calls into functional categories (file access, networking, IPC).
- Monitor transitions between app states (idle, active, background).

- Collect and analyze kernel-level usage statistics per application.
- Identify potential indirect data leakage through side-channel patterns.
- Correlate traced behaviors with declared permissions.
- Implement a web-based dashboard for behavior visualization.

Broader Goals

- Enhance transparency in how messaging apps behave at system level.
- Improve user awareness of hidden behaviors executed in the background.
- Demonstrate the value of kernel-level tracing for security and privacy evaluation.
- Provide a structured and reproducible methodology for privacy-respecting behavior analysis.

1.3 Research Questions

Based on the motivation and objectives, this thesis aims to address the following research questions:

Q1. What kernel-level operations do popular messaging applications perform during normal usage?

Q2. Are there deviations between the declared permissions of these applications and their actual behavior at runtime?

Q3. Can kernel-level tracing techniques identify unexpected or potentially invasive operations performed without user interaction?

Q4. How does the behavior of privacy-focused apps compare to that of commercial messaging platforms at the kernel level?

Q5. What kind of patterns in system calls can be used to characterize privacy-relevant behavior?

1.4 Limitations

- **Platform Scope**: Analysis restricted to Android 10+ due to kernel API dependencies.
- **Dynamic Analysis Constraints**: Real-world noise (e.g., background services) may affect system call traces.

- **App Selection Bias**: Focus on top-tier apps (WhatsApp, Signal, Telegram) may omit niche platforms.

1.5 Contributions of this Thesis

1.6 Thesis Outline

This thesis is organized into the following chapters:

- **Chapter 1 – Introduction:** Provides background context, outlines the motivation and objectives, presents the research questions, contributions, and a high-level overview of the thesis structure.
- **Chapter 2 – Related Work and Technical Background:** Reviews existing literature on Android architecture, messaging app privacy implications, static and dynamic analysis techniques, system call tracing, and identifies key research gaps.
- **Chapter 3 – Methodology and System Design:** Describes the research design, experimental setup, data collection using kernel-level tracing, and the analysis framework.
- **Chapter 4 – Results:** Presents the observed behavioral patterns, differences among messaging apps, and key findings related to privacy-relevant behaviors.
- **Chapter 5 – Discussion:** Interprets the results in light of the research questions, discusses limitations of the study, and suggests potential improvements.
- **Chapter 6 – Conclusions:** Summarizes key contributions, highlights findings, and suggests directions for future research.
- **Appendix A – Additional Data Tables:** Includes supplementary statistical data and traces.
- **Appendix B – Code:** Provides relevant shell scripts, Python tools, and configuration details used in the implementation.

Chapter 2

Related Work and Technical Background

2.1 Android Architecture and Kernel-Level Access

A review of previous research, articles, and studies relevant to this thesis. Emphasis is placed on identifying gaps in current literature and highlighting the thesis's focus areas.

2.2 Messaging Apps: Characteristics Privacy Implications

2.3 Static vs. Dynamic Analysis

2.4 System Call Analysis and Kernel Tracing

2.5 Machine Learning for Behavior Profiling

2.6 Related Research Gaps in Literature

Chapter 3

Methodology and System Design

3.1 Research Design

3.2 Experimental Setup

A discussion of experimental settings, including how experiments were conducted and what evaluation metrics (e.g., accuracy, precision, recall, F1-score) were used.

3.3 Data Collection and Preparation

A detailed description of how kernel-level data is collected and the preprocessing steps taken to ensure suitability for ML algorithms.

3.4 Feature Extraction from System Calls

3.5 Machine Learning Models and Tools

An overview of the ML algorithms (e.g., Random Forest, SVM, Neural Networks) and the software tools (e.g., Python, scikit-learn) employed in the study.

Chapter 4

Results

4.1 Behavioral Patterns Observed

4.2 Model Performance

4.3 Comparison Between Messaging Apps

4.4 Classification or Pattern Recognition Outcomes

Presentation of evaluation tables, charts, and analysis derived from the ML algorithms.

4.5 Comparisons and Interpretations

Comparison of different models or configurations, with emphasis on interpreting discrepancies and assessing each model's performance.

4.6 Visualizations

Chapter 5

Discussion

5.1 Interpretation of Results

5.2 Limitations of the Study

5.3 Opportunities for Improvement

A detailed discussion of how the findings relate to the initial research objectives and the broader literature. The contribution and limitations of this study are highlighted.

Chapter 6

Conclusions

6.1 Key Findings

A summary of the main results and how they address the initial research questions.

6.2 Future Research Directions

Suggestions for expanding this research, including improvements or new avenues for study.

Bibliography

[1] Statista, *Number of smartphone users worldwide from 2014 to 2029*, 2024. Available: `https://www.statista.com/forecasts/1143723/smartphone-users-in-the-world`

[2] F. Laricchia, "Mobile operating systems' market share worldwide from January 2012 to July 2020," Statista, 2021. `https://www.statista.com/statistics/272698/global-market-share-held-by-mobile-operating-systems-since-2009/`

[3] The Verge, "Facebook has been collecting call history and SMS data from Android devices," 2018. `https://www.theverge.com/2018/3/25/17160944/facebook-call-history-sms-data-collection-android`

[4] GDPR.EU, *General Data Protection Regulation*, 2018. `https://gdpr.eu/`

[5] UK Government, *Data Protection Act 2018*, `https://www.legislation.gov.uk/ukpga/2018/12/contents/enacted`

[6] Z. Feng et al., "A Survey on Security and Privacy Issues in Android," IEEE Communications Surveys & Tutorials, vol. 22, no. 4, pp. 2445-2472, 2020.

[7] A. Felt et al., "Android Permissions: User Attention, Comprehension, and Behavior," SOUPS, 2012.

[8] A. Gorla et al., "Checking App Behavior Against App Descriptions," ICSE, 2014.

[9] Y. Nan et al., "UIPicker: User-Input Privacy Identification in Mobile Applications," IEEE TSE, 2019.

[10] S. Arzt et al., "FlowDroid: Precise Context, Flow, Field, Object-sensitive and Lifecycle-aware Taint Analysis for Android Apps," PLDI, 2014.

[11] W. Enck et al., "TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones," OSDI, 2010.

[12] Z. Xu et al., "Crowdroid: Behavior-Based Malware Detection System for Android," SPSM, 2011.

[13] M. Lindorfer et al., "ANDRUBIS - 1,000,000 Apps Later: A View on Current Android Malware Behaviors," BADGERS, 2014.

[14] G. Canfora et al., "Detecting Android Malware Using Sequences of System Calls," IEEE TSE, 2015.

[15] R. Love, *Linux Kernel Development*, Addison-Wesley, 2010.

[16] S. Rostedt, "Ftrace: Function Tracer," Linux Kernel Documentation, 2023.

[17] Linux Kernel Organization, "Kernel Probes (kprobes)," Linux Kernel Documentation, 2023.

[18] J. Corbet, G. Kroah-Hartman, A. Rubini, *Linux Device Drivers*, 4th ed., O'Reilly Media, 2015.

[19] J. Tang et al., "Profiling Android Applications via Kernel Tracing," IEEE TMC, 2017.

[20] J. Kim et al., "Understanding I/O Behavior in Android Applications through Kernel Tracing," ACM MobiSys, 2016.

[21] M. Backes et al., "Boxify: Full-fledged App Sandboxing for Stock Android," USENIX Security, 2015.

[22] The Washington Post, "Pentagon officials used Signal messaging app, raising security concerns," March 2023.

Appendix A

Appendix A: Additional Data Tables

Any further data tables, graphics, or supplementary material.

Appendix B

Appendix B: Code

Source code or additional scripts too extensive to include in the main chapters.