International Journal of Advanced Research in Engineering Technology & Science

ISSN: 2349-2819

Email: editor@ijarets.org Volume-10 Issue-10 October-2023 www.ijarets.org

ADVANCED TECHNIQUES FOR SECURITY ASSESSMENT IN MOBILE APPLICATIONS

Ngangom James Singh,

Research Scholar, Department Computer Applications,

Maharaja Agrasen Himalayan Garhwal University

Dr. Sayed Mohd. Saqib,

Assistant Professor, Department Computer Applications,

Maharaja Agrasen Himalayan Garhwal University

Abstract

Mobile applications have become an integral part of daily life, offering a range of services from communication to financial transactions. However, the increased use of mobile applications has also raised significant security concerns. This paper explores advanced techniques for security assessment in mobile applications, focusing on static and dynamic analysis, machine learning approaches, and automated security testing tools. By evaluating these methods, the paper aims to provide a comprehensive understanding of how to enhance the security of mobile applications effectively.

Introduction

The proliferation of mobile devices and applications has revolutionized how individuals interact with technology. With millions of applications available, ensuring the security of these apps is crucial. Security assessment techniques play a vital role in identifying vulnerabilities and mitigating potential threats. This paper delves into various advanced techniques for security assessment in mobile applications, examining their effectiveness, challenges, and future prospects.

Several major players dominate the mobile operating system landscape, including Google, Microsoft, Apple, Symbian, and Palm. Each of these companies has developed its own OS, aiming to create a platform that is both user-friendly and secure. The evolution of these operating systems has been marked by continuous improvements and innovations, each striving to offer the best user experience and meet the diverse needs of consumers.

Google's Android OS is one of the most popular mobile operating systems worldwide. Known for its open-source nature, Android allows for significant customization and flexibility. It supports a vast ecosystem of applications available through the Google Play Store, catering to virtually every conceivable need. The OS is also designed to integrate seamlessly with Google's suite of services, including Gmail, Google Drive, and Google Maps, enhancing its utility for users who rely on these services.

Apple's iOS, on the other hand, is known for its closed ecosystem and stringent app review process, which ensures a high level of security and stability. iOS is renowned for its smooth and intuitive user interface, which has set a benchmark in the industry. The seamless integration of hardware and software in Apple's devices contributes to a cohesive user experience, making it a favorite among users who prioritize reliability and performance.

Microsoft's Windows Phone OS, although no longer a major player in the market, introduced unique features such as live tiles and seamless integration with Microsoft Office and other productivity tools. This made it an attractive option for business users and those deeply embedded in the Microsoft ecosystem.

Symbian, once a dominant player in the mobile OS market, was known for its efficiency and robustness, particularly in the early days of smartphones. However, it struggled to keep pace with the rapid advancements and the increasing demand for app-centric platforms, leading to its decline.

Palm OS, which also played a significant role in the early development of mobile operating systems, was renowned for its simplicity and ease of use. Despite its initial success, it eventually fell behind in the competitive landscape dominated by Android and iOS.

Despite the advancements and unique features of these mobile operating systems, none can be deemed perfect. Each platform has its strengths and weaknesses, and the choice of the best OS often depends on individual user preferences and requirements. Factors such as user interface, app availability, security, and integration with other services play crucial roles in determining which OS is most suitable for a particular user.

User-friendliness is a critical aspect that influences the popularity of a mobile operating system. A user-friendly OS ensures that even novice users can navigate through its features and functionalities with ease. Both Android and iOS have made significant strides in this area, offering intuitive interfaces and comprehensive support to help users make the most of their devices.

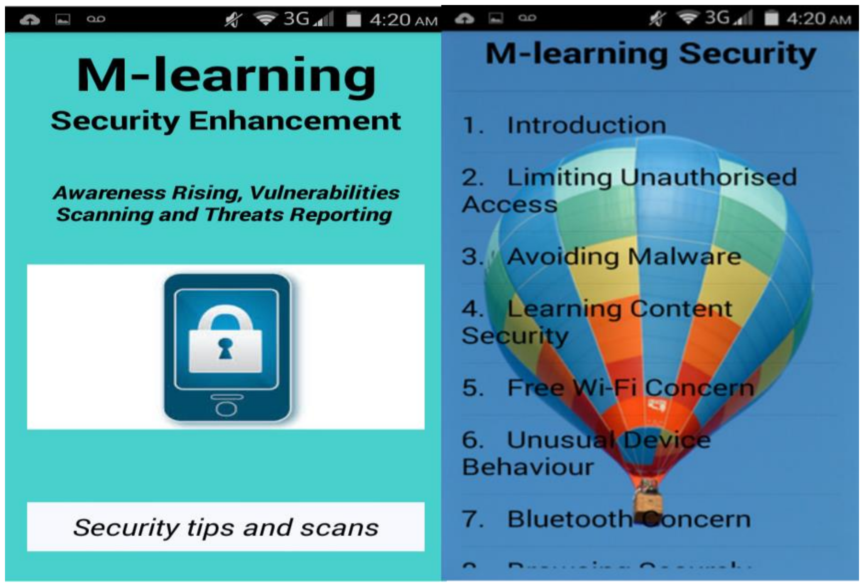


Fig: The app home page and activity list

Review of Literature

"Mobile Application Security: Enhancing Your Mobile Security Posture" by Jeff Forristal** is a comprehensive and practical guide to understanding and improving mobile application security. The book covers a wide range of topics, from foundational concepts and threat modeling to secure development practices and advanced security techniques. Forristal's expertise and practical insights make this book an invaluable resource for developers, security professionals, and anyone involved in mobile application security. By following the guidance provided in this book, readers can enhance their mobile security posture and better protect their applications and data from emerging threats.

"Android Security Internals: An In-Depth Guide to Android's Security Architecture" by Nico Golde (2014)

"Android Security Internals: An In-Depth Guide to Android's Security Architecture"by Nico Golde, published in 2014, is a comprehensive examination of the security mechanisms embedded within the Android operating system. This book provides a detailed analysis of Android's security architecture, offering valuable insights into the various components and strategies employed to protect the platform from a range of threats. Golde, with his extensive expertise in security, delves into the intricacies of Android's design and implementation, making this book an essential resource for security professionals, developers, and researchers interested in understanding and improving Android security.

Introduction to Android Security

The book begins by setting the stage with an introduction to Android security. Golde outlines the fundamental principles of security as they apply to the Android platform, emphasizing the importance of securing mobile devices in a world where Android is the most widely used mobile operating system. He explains the challenges and risks associated with Android security, including the diverse range of devices, variations in hardware and software, and the constant evolution of threats. This introductory section provides a foundational understanding of why security is a critical concern for Android and prepares readers for the detailed analysis that follows.

Android Security Architecture

Golde's book offers a deep dive into the core components of Android's security architecture. He begins with an examination of the Android operating system's structure, including its kernel, system services, and application framework. Golde explains how these components interact to create a secure environment for applications and user data. He provides a detailed look at the Android Linux kernel and its role in enforcing security policies, discussing concepts such as process isolation, memory protection, and access controls. The book also covers Android's custom security enhancements built on top of the Linux kernel, such as the Security-Enhanced Linux (SELinux) policy.

Application Sandbox Model

A significant portion of the book is dedicated to the Android application sandbox model. Golde describes how Android employs a sandboxing approach to isolate applications from each other and from the underlying system. He explains the principles of application isolation, including the use of unique user IDs, separate process spaces, and the permission model that governs access to resources. Golde also explores the implications of sandboxing for application security, highlighting both its strengths and limitations. He provides detailed examples of how sandboxing helps protect user data and prevent unauthorized access, as well as how attackers might attempt to bypass these protections.

Permission Model and Access Control

The Android permission model is a critical aspect of its security architecture, and Golde provides an in-depth analysis of how it works. He explains the concept of permissions and how they are used to control access to sensitive resources and APIs. The book covers the different types of permissions, including normal and dangerous permissions, and discusses how permissions are requested, granted, and enforced. Golde also examines the challenges associated with managing permissions, such as the potential for privilege escalation and the risks posed by overly broad permissions. He provides practical examples of how permissions can be used effectively and how they can be misconfigured or exploited.

Cryptography and Data Protection

Golde's exploration of Android's cryptographic and data protection mechanisms is another key aspect of the book. He discusses the various cryptographic algorithms and protocols used to secure data on Android devices, including encryption for data at rest and in transit. Golde provides a detailed look at Android's KeyStore system, which is designed to securely manage cryptographic keys and protect sensitive information. He also covers topics such as secure key storage, data encryption standards, and the use of hardware-backed security features. The book includes practical advice on implementing encryption and protecting data in mobile applications.

Security Challenges and Threats

In this section, Golde addresses the security challenges and threats that Android faces. He provides a thorough analysis of various attack vectors, including malware, vulnerabilities in third-party applications, and exploits targeting the Android operating system itself. The book discusses different types of malware, such as viruses, worms, and spyware, and examines how these threats can compromise Android devices. Golde also explores the role of security updates and patch management in addressing vulnerabilities and mitigating risks. He provides insights into how attackers operate and how security measures can be improved to defend against evolving threats.

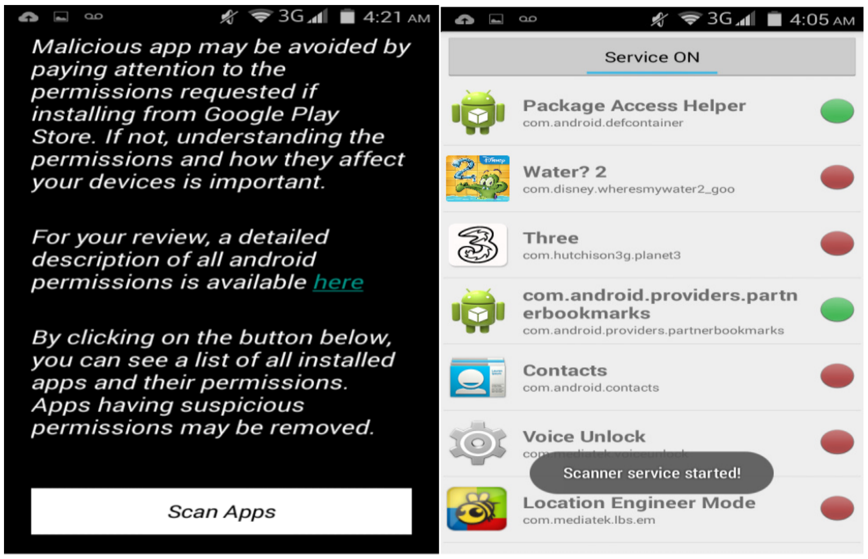
Static Analysis

Static analysis involves examining the application's source code or binary without executing it. This technique helps identify vulnerabilities, such as insecure coding practices and potential security flaws, early in the development process.

1. Code Review Tools: Tools like Fortify, Checkmarx, and SonarQube analyze the source code for security issues, providing detailed reports and recommendations for remediation.

2. Binary Analysis: Tools like IDA Pro and Ghidra disassemble and analyze the binary code, helping identify potential vulnerabilities in compiled applications.

3. SAST (Static Application Security Testing): SAST tools integrate into the development pipeline, allowing continuous security assessments and early detection of vulnerabilities.



Malicious app may be avoided by paying attention to the permissions requested if installing from Google Play Store. If not, understanding the permissions and how they affect your devices is important.

For your review, a detailed description of all android permissions is available here

By clicking on the button below, you can see a list of all installed apps and their permissions. Apps having suspicious permissions may be removed.

Package Access Helper
com.android.defcontainer

Water? 2
com.disney.wheresmywater2_goo

Three
com.hutchison3g.planet3

com.android.providers.partnerbookmarks
com.android.providers.partnerbookmarks

Contacts
com.android.contacts

Voice Unlock
com.mediatek.voiceunlock

Location Engineer Mode
com.mediatek.lbs.em

Fig 2: Service on Pages.

Dynamic Analysis

Dynamic analysis involves executing the application in a controlled environment to observe its behavior and identify security issues during runtime.

1. DAST (Dynamic Application Security Testing): DAST tools like OWASP ZAP and Burp Suite simulate attacks on running applications, identifying vulnerabilities that occur during execution.

2. Runtime Analysis: Tools like Frida and Xposed Framework monitor the application's behavior at runtime, identifying issues such as unauthorized data access and API misuse.

3. Emulators and Sandboxes: Using emulators and sandboxes allows for safe testing of applications, enabling security analysts to observe how the app behaves under different conditions and identify potential security gaps.

Machine Learning Approaches

Machine learning techniques have emerged as powerful tools for enhancing security assessments in mobile applications.

1. Anomaly Detection: Machine learning models can be trained to detect anomalous behavior in mobile applications, identifying potential security threats that deviate from normal patterns.

2. Malware Detection: Machine learning algorithms, such as neural networks and support vector machines, can classify applications as malicious or benign based on their behavior and characteristics.

3. Automated Threat Hunting: Machine learning models can continuously analyze application data, identifying emerging threats and providing real-time security insights.

Automated Security Testing Tools

Automated security testing tools streamline the process of identifying and mitigating vulnerabilities in mobile applications.

1. CI/CD Integration: Integrating security testing tools into the Continuous Integration/Continuous Deployment (CI/CD) pipeline ensures that security assessments are conducted regularly, reducing the risk of vulnerabilities in production.

2. Mobile App Security Testing Platforms: Platforms like MobSF and AppScan provide comprehensive security testing for mobile applications, offering static and dynamic analysis, malware detection, and more.

3. Automated Penetration Testing: Tools like Metasploit and Core Impact automate penetration testing, simulating attacks to identify vulnerabilities and assess the application's security posture.

Methodologies:

In the growth of the Android user base presents both opportunities and challenges. The open nature of the Android ecosystem fosters innovation and diversity in the application market. However, it also opens the door to malicious applications that can compromise user security. By implementing advanced security techniques and promoting safe practices, it is possible to grant users the freedom to explore the vast array of applications available while ensuring their safety and privacy. The work of Dar and

Parvez provides a solid foundation for these efforts, and the proof-of-concept implementation outlined in this chapter offers a practical roadmap for enhancing the security of smartphone applications. Through continued research, collaboration, and user education, the goal of a secure and open mobile ecosystem can be achieved.

Need-Based Security (NBS) is an innovative approach designed to address one of the most pressing issues in mobile application security—the overreliance on extensive permissions. When users install applications on their Android devices, they are often presented with a lengthy list of permissions that the app requires to function. These permissions can range from benign requests, such as access to the internet, to more intrusive demands, like access to contacts, camera, and location services. The problem with this approach is that it grants applications more access than they actually need, potentially opening up significant security vulnerabilities. The objective of Need-Based Security, or user-approved security, is to refine this process by removing unnecessary permissions from the AndroidManifest.xml file, thereby creating more secure applications. The implementation of NBS, as outlined by M. A. Dar and Parvez in 2014, involves several critical processes.

At the heart of NBS is the principle that an application should only request permissions that are essential for its core functionality. This need-based approach ensures that users are not unknowingly granting excessive access to their personal information and device capabilities. The first step in implementing NBS involves a thorough analysis of the application’s requirements. Developers must carefully evaluate which permissions are truly necessary for the app to perform its intended functions. This step is crucial because it sets the foundation for creating a secure application.

Challenges and Future Prospects

While advanced techniques for security assessment offer significant benefits, they also present challenges. These include the complexity of mobile application environments, the need for specialized skills, and the constant evolution of security threats. Future research should focus on developing more sophisticated machine learning models, improving the integration of security tools into development workflows, and addressing the unique security challenges of emerging technologies such as IoT and 5G.

Conclusion

Advanced techniques for security assessment are essential for ensuring the safety and reliability of mobile applications. By leveraging static and dynamic analysis, machine learning approaches, and automated security testing tools, developers and security professionals can identify and mitigate vulnerabilities effectively. Continuous research and innovation are necessary to keep pace with the evolving threat landscape and enhance the security of mobile applications.

One of the innovative techniques we propose is the development of a layered security approach. This approach involves implementing multiple layers of security measures, each designed to address different aspects of smartphone security. For example, the first layer could involve enhancing the permission system to provide more granular control over app permissions. The second layer could focus on continuous monitoring and anomaly detection, using machine learning algorithms to identify suspicious behavior. The third layer could involve user education and awareness programs to ensure that users understand the importance of security and privacy.

Furthermore, our research highlights the potential benefits of integrating biometric authentication methods into the security framework. Biometric methods, such as fingerprint recognition and facial recognition, provide an additional layer of security by ensuring that only authorized users can access the device and its data. By combining biometric authentication with other security measures, we can create a more robust and secure environment for smartphone users.

In addition to these techniques, we also explore the potential of using blockchain technology to enhance smartphone security. Blockchain technology offers a decentralized and transparent approach to data security, making it more difficult for malicious actors to compromise the system. By leveraging blockchain, we can create a more secure and trustworthy environment for app transactions and data sharing.

Our study also examines the role of artificial intelligence (AI) in improving smartphone security. AI can be used to develop advanced threat detection systems that can identify and respond to security threats in real-time. By analyzing patterns of behavior and detecting anomalies, AI-powered systems can provide early warnings of potential security breaches and help mitigate risks before they escalate.

In conclusion, our research provides a comprehensive analysis of the current state of smartphone security, with a particular focus on the Android operating system. We have identified significant vulnerabilities in the existing security framework and proposed a series of novel techniques to address these issues. Our findings underscore the importance of user awareness, continuous monitoring, developer accountability, and community collaboration in enhancing smartphone security. We believe that our study has the potential to significantly improve the state of practice in smartphone security and provide a safer digital environment for users worldwide. Through ongoing research and collaboration, we can continue to develop innovative solutions to address the evolving challenges of smartphone security and ensure that users can enjoy the benefits of mobile technology without compromising their privacy and safety.

In our detailed security comparison of the world's leading smartphone operating systems, we aimed to uncover if any OS holds a particular advantage over others. Our research concluded that novice users, especially Android users, often remain unaware of the permissions they have granted during app installation, which poses significant security risks. To address this, we conducted an experiment where we removed dangerous permissions from apps based on their functionality, without altering the underlying Android OS structure, thereby providing an additional layer of security.

This dissertation outlines several key steps and novel techniques to enhance the smartphone security framework. Our investigation into how Android permissions and their usage evolve within the Android ecosystem revealed that the security framework is becoming increasingly vulnerable. We propose several innovative techniques to remedy this situation, which could be beneficial to researchers, developers, and users alike, and have the potential to improve the overall state of smartphone security, particularly for Android OS.

A critical aspect of our research involved analyzing the evolution of Android permissions. Over time, permissions requested by apps have become more invasive, with many applications seeking access to sensitive data that is not essential for their primary functionality. This trend poses a significant risk to user privacy and security, particularly for those who lack the technical knowledge to evaluate the necessity of these permissions.

To mitigate these risks, we propose a dynamic permission system that provides real-time notifications and contextual information to users about the specific resources being accessed by an app. This system would help users understand why a particular permission is needed and the potential consequences of granting it, enabling them to make more informed decisions and reduce the risk of unauthorized access to their data.

References

1. OWASP. (2023). Mobile Security Project. Retrieved from [OWASP Mobile Security Project](https://owasp.org/www-project-mobile-security/)

2. SonarQube. (2023). Continuous Inspection. Retrieved from[SonarQube](https://www.sonarqube.org/)

3. Fortify. (2023). Application Security Solutions. Retrieved from [Fortify](https://www.microfocus.com/en-us/cyberres/application-security/fortify)

4. Checkmarx. (2023). Software Security Platform. Retrieved from [Checkmarx](https://www.checkmarx.com/)

5. OWASP ZAP. (2023). Zed Attack Proxy. Retrieved from [OWASP ZAP](https://www.zaproxy.org/)

6. Burp Suite. (2023). Web Vulnerability Scanner. Retrieved from [Burp Suite](https://portswigger.net/burp)

7. MobSF. (2023). Mobile Security Framework. Retrieved from [MobSF](https://mobexler.com/)

8. Metasploit. (2023). Penetration Testing Software. Retrieved from [Metasploit](https://www.metasploit.com/)

9. ProGuard. (2023). Java and Android Obfuscator. Retrieved from [ProGuard](https://www.guardsquare.com/en/products/proguard)

10. DexGuard. (2023). Android Protection Suite. Retrieved from [DexGuard](https://www.guardsquare.com/en/products/dexguard)

11. SSL Labs. (2023). SSL/TLS Best Practices. Retrieved from [SSL Labs](https://www.ssllabs.com/)

12. OWASP. (2023). API Security Top 10. Retrieved from [OWASP API Security](https://owasp.org/www-project-api-security/)

13. National Institute of Standards and Technology (NIST). (2023). Security and Privacy Controls for Information Systems and Organizations. Retrieved from [NIST](https://csrc.nist.gov/publications/detail/sp/800-53/rev-5/final)

14. An Efficient Common Substrings Algorithm for On-the-Fly Behavior-Based Malware Detection and Analysis Authors: J. C. Acosta, H. Mendoza, B. G. Medina Year: 2012

15. A Proposal to Realize the Provision of Secure Android Applications - ADMS: An Application Development and Management System Authors: H. Agematsu et al.Year: 2012

16. Security Comparison of Android and IOS and Implementation of User Approved Security (UAS) for Android Authors: D. M. Ahmad, P. Javed Year: 2016

17. Android vs. iOS: The Security Battle Authors: F. Al-Qershi et al. Year: 2014

18. Enhancing Stealthiness & Efficiency of Android Trojans and Defense Possibilities (EnSEAD): Android's Malware Attack, Stealthiness and Defense: An Improvement Authors: M. Ali, H. Ali, Z. Anwar Year: 2011

19. A Framework for GPU-Accelerated AES-XTS Encryption in Mobile Device Authors: M. A. Alomari, K. Samsudin Year: 2011

20. Oily Residuals Security Threat on Smart Phones Authors: K. AlRowaily, M. AlRubaian

Year: 2011

21. Performance Evaluation of Multi-Pattern Matching Algorithms on Smartphone Authors: A. Amamra, C. Talhi, J. M. Robert Year: 2012

22. Smartphone Malware Detection: From a Survey Towards Taxonomy Authors: A. Amamra, C. Talhi, J. M. Robert Year: 2012

23. Why is My Smartphone Slow? On the Fly Diagnosis of Underperformance on the Mobile Internet Authors: C. Amrutkar et al. Year: 2013

24. Permission-Based Android Malware Detection Authors: Z. Aung, W. Zaw

Year: 2013

25. Malware Detection Method Based on the Control-Flow Construct Feature of Software

Authors: J. Bai, Z. Zhao, J. Wang Year: 2014 Secure Software Installation on Smartphones

Authors: David Barrera, P. Van Oorschot Year: 2011

26. Automatically Securing Permission-Based Software by Reducing the Attack Surface: An Application to AndroidAuthors: A. Bartel et al. Year: 2012

27. Elliptic Curve Cryptography in Practice Authors: J. W. Bos et al. Year: 2014

28. Android Malware Past, Present, and FutureAuthor: C. A. Castillo Year: 2010

29. Malwise-an Effective and Efficient Classification System for Packed and Polymorphic Malware Authors: S. Cesare, Y. Xiang, W. Zhou Year: 2013

30. A Scalable Approach for Malware Detection Through Bounded Feature Space Behavior Modeling Authors: M. Chandramohan et al. Year: 2013

31. Offloading Android Applications to the Cloud Without Customizing Android

Authors: E. Chen, S. Ogata, K. Horikawa Year: 2012

32. A Lightweight Virtualization Solution for Android Devices Authors: W. Chen et al. Year: 2015

33. Identifying Smartphone Malware Using Data Mining Technology Authors: H. Sen Chiang, W. J. Tsaur Year: 2011

34. Security Assessment of Code Obfuscation Based on Dynamic Monitoring in Android Things Authors: T. Cho, H. Kim, J. H. Yi Year: 2017

35. Design and Development of a New Scanning Core Engine for Malware Detection

Authors: L. L. Chuan et al. Year: 2012

36. Android Application Development to Promote Physical Activity in Adolescents

Authors: D. Clark et al. Year: 2012

37. CRePE: Context-Related Policy Enforcement for Android Author: M. C. T. N. N. Crispo Year: 2010

38. Analysis of Zitmo (Zeus in the Mobile) Author: D. Desai Year: 2011

39. A Novel Strategy to Enhance the Android Security Framework Authors: M. Dar, J. Parvez Year: 2014

40. Role of Smartphone in Rural Development: A Case Study of Kashmir Author: M. A. Dar Year: 2016

41. Enhancing Security of Android & iOS by Implementing Need-Based Security (NBS)

Authors: M. A. Dar, J. Parvez Year: 2014

42. Smartphone Operating Systems: Evaluation & Enhancements Authors: M. A. Dar, J. Parvez Year: 2014

43. A Live-Tracking Framework for Smartphones Authors: M. A. Dar, J. Parvez Year: 2015

44. Novel Techniques to Enhance the Security of Smartphone Applications

Authors: M. A. Dar, J. Parvez Year: 2016

Smartphone Malware Threat, an Experimental Evaluation of Smartphone Security

Authors: M. A. Dar, J. Parvez Year: 2016

45. Security Enhancement in Android Using Elliptic Curve Cryptography Authors: M. A. Dar, J. Parvez Year: 2017

46. Emerging Security Threats for Mobile Platforms Authors: G. Delac, M. Silic, J. Krolo

Year: 2011 iPhone 2.0 SDK: The No Multitasking Myth Author: D. E. Dilger Year: 2008

47. Power Based Malicious Code Detection Techniques for Smartphones Authors: B. Dixon, S. Mishra Year: 2013

48. Enhancing User Privacy on Android Mobile Devices via Permissions Removal

Authors: Q. Do, B. Martini, K. K. R. Choo Year: 2014

49. A Flexible and Lightweight ECC-Based Authentication Solution for Resource-Constrained Systems Authors: N. Druml et al. Year: 2014

50. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones Authors: W. Enck et al. Year: 2014