Systematic Analysis of Smartphone Architectural Security: A Multi-Layered Vulnerability Assessment of Android and iOS Ecosystems (2020-2025)

Maunik Shah[1], Shiva Kumara[2]
[1]Indian Institute of Technology, Bombay; maunik.shah27989@gmail.com
[2]Independent Researcher, Kuvempu University, reachkumaras@gmail.com

Abstract
Android and iOS constitute the major smartphone players on the world stage, but there is still limited knowledge on their architectural security flaws. This paper is a multi-layered vulnerability evaluation of the Android and iOS platform on the basis of vulnerabilities that are publicly disclosed between 2020 and 2025. A total of 5647 vulnerabilities were analyzed, of which 3192 are for Android and 2455 are for iOS. Vulnerabilities were assigned to a structured architectural security model that includes hardware and firmware, kernel and operating system core, system services and middleware, application framework, and ecosystem and update mechanisms. The results show that vulnerabilities are disproportionately distributed across the various layers of the architecture, with the kernel layer, system services, and application framework layers contributing to the greatest number of disclosures with the highest severity levels. More than half of all the reported issues are cross-layer vulnerabilities which illustrate the weakness of a rigid layer isolation. Platform-specific differences in vulnerability concentration and patch latency distributions as one of the comparative analysis indicators portray different architectural design and ecosystem governance models. This paper places vulnerabilities into their own architectural context to offer a longitudinal and comparative overview of systemic security risk, which can be applied to platform designers, researchers and security practitioners aiming to enhance smartphone architectures.

Keywords: Smartphone security architecture, Android and iOS vulnerabilities, Multi-layer vulnerability analysis, Operating system security

1. Introduction

Since the advent of smartphones, the device has become a primary interface in the modern digital life, providing a means for us to communicate, make financial transactions, monitor our health and access sensitive personal and organizational data. As the use of mobile platforms has grown, so has the effects of security failures in smartphone ecosystems. Over the last decade, and especially in the period of 2020-2025, mobile devices have been increasingly attacked by sophisticated threats that leveraged weaknesses that are not only found at the application layer, but throughout this underlying system architecture. This evolution has revealed the shortcomings of security approaches based on application-level defenses that ignore security weaknesses that are embedded within operating systems, system services, and supporting infrastructure.

Modern mobile threat environment is becoming more complex, more large-scale, and more automated. The increase in malware writing, reverse engineering software, and automated vulnerability exposure has reduced the obstacle by which the attackers can find and take advantage of vulnerabilities in the mobile software stacks. Empirical works analyzing Android malware detection and analysis methods show that adversaries constantly adapt to the defensive mechanism with the aim of exploiting the weaknesses in platform internals and runtime environments instead of only using malicious applications (Daoudi et al., 2021; Daoudi et al., 2023). Similarly, research on application analysis tools and datasets illustrate the increasing number and range of vulnerabilities associated with mobile applications and their interaction with the system-level components (Pagano et al, 2023; Arikan & Yilmaz, 2025).

Despite the diversity of the research that has been published, a great deal of the literature has been fragmented and narrowly-scoped. A large part of the previous work is dedicated to malware detection, machine learning-based classification, or security analysis of applications, which usually deals with the evaluation of isolated components without taking into account their location in the overall smartphone architecture. For example, analyses of the decompilation performance for Android applications or the privacy compliance in the given app categories give valuable insights into the security issues related to the localization of services, but do not reflect the propagation of the vulnerabilities across the architectural layers and the effect of the design choices made at lower levels on the overall security of the platform (Hatamian et al., 2021; Kargoten et al., 2023). As a result, there is a lack of a system wide understanding of vulnerability distribution across hardware, operating systems, system services, and application frameworks in the field, especially in a comparative context across both the Android and iOS ecosystems.

This disconnect is becoming very problematic because in current smartphone platforms, the strict multiple layers of design often cause vulnerabilities at one level to compromise the defense of the other. Application facing vulnerabilities can be symptomatic of deeper architectural problems, system services or runtime environments flaws can make application level safeguards ineffective. Technically rigorous but in most cases failing to assess these interactions or place vulnerabilities in the context of an integrated architecture, the currently available

empirical studies and tool-based analyses (Pagano et al., 2023; Arikan and Yilmaz, 2025). Consequently, security practitioners and platform designers do not have empirically-grounded knowledge about which architectural layers account for the largest contributions to systemic risk and how patterns of vulnerability change over time.

Inspired by these constraints, this paper fulfills the requirement of systematic and architecture-sensitive vulnerability analysis of smartphone platforms. Rather than focusing on vulnerabilities independent of each other, the study takes a multi-dimensional view that explicitly represents the disclosure of vulnerabilities to architecture strata in the Android and iOS systems. The analysis aims at shedding light on steady vulnerabilities, pattern exposure to platforms, and the performance of current mitigation procedures in the next five years (20202025) by combining longitudinal data of vulnerability and a structured architectural model.

Accordingly, this research is driven by the following research questions: How are the publicly disclosed vulnerabilities distributed among security layers of smartphones? What are the architectural differences between Android and iOS and how does this affect the concentration and the exploitability of vulnerabilities? To what extent and how do current mitigation mechanisms seem to reduce high impact vulnerabilities across various layers over time? To answer these questions, it is necessary to go beyond individual case studies and adopt a comparative and data-driven methodology that can help to capture systematic trends.

The main contributions of this study are three fold and each is entirely realized in the methodology and analysis presented in this work:

- A multi-layer vulnerability taxonomy: a systematic mapping of smartphone vulnerabilities onto different architectural layers, allowing architecture-aware security-analysis to be performed beyond the application-level viewpoints.
- Empirical evidence on the platform-specific vulnerability, severity distribution, and temporal trends of Android and iOS vulnerabilities through a comparative longitudinal analysis (20202025).
- Architectural design information that points to trade-offs in the current smartphone security models and uncovers structural weak points at multiple levels..

This study is relevant in that it provides a better understanding of the limitations of the existing research on mobile security tools, data sets, and detection methods by basing vulnerability analysis on architectural context and time scale, as well as in interpreting the technology at the system level. The results are intended to assist the platform designers, security researchers and developers in not just knowing the locations of vulnerabilities, but the reasons behind them as well as how architectural choices influence future security.

2. Literature Review

Smartphone security architectures have become sophisticated and multi-layered, aiming to have a balance between usability, performance, and security against more sophisticated attacks. As the leading mobile operating systems, Android and iOS, follow different architectural philosophies which have an impact on the way security mechanisms are implemented and on the manifestation of vulnerabilities. Android is stipulated by an architecture focussing on modularity and openness to allow massive customization over equipment and phones, and deploy permission models, application sandboxing, and runtime enforcement to isolate untrusted code. In contrast, iOS has a very controlled architecture, with centralized control, rigorous application review processes, and robust default system component and application isolation. Although the two platforms both utilize layered defenses, the architectural openness, update mechanisms, and ecosystem control of the two platforms determine their threat surfaces respectively.

A large amount of previous research has been conducted on mobile operating system vulnerabilities, specifically on the application layer. Numerous researches are involved in malware detection techniques using machine learning and deep learning, which represents the growing interest in the field of automatic threat detection. Systematic reviews of Android malware detection point out the range of learning models, feature extraction approaches, and datasets that are used to detect malicious behavior and also note challenges in terms of scalability, evasion, and real-world deployment (Senanayake et al., 2021; Kouliaridis & Kambourakis, 2021). Such methods are further developed to other neural structure models, such as transformer models and graph convolutional networks to achieve more accurate and robust detection (Gomez and Munoz, 2023; Kacem and Tossou, 2025; Zhao et al., 2025).

Despite their technical sophistication, studies of malware detection often have a narrow focus on application behavior and do not expressly place vulnerabilities in the context of the overall operating system architecture. This is a limitation found in the research where the detection algorithm or classification pipeline is evaluated but there is no discussion of the role of underlying system services, kernel interfaces, or update mechanisms in exploitability. Even thorough surveys of artificial intelligence-based malware detection focus on algorithmic performance and not on architectural exposure or system risk (Alkahtani & Aldhyani, 2022). As a result, such studies do not offer much insight about how vulnerabilities are distributed between architectural layers, or how architectural design decisions affect security in the long-term.

In addition to malware detection, empirical studies of application security and privacy provide useful yet no more broad-based points of view. Researches on Android applications deployed in sensitive situations, e.g. in the context of the Covid-19 contact tracing, have shown extensive privacy and security vulnerabilities due to misuse of

permissions, insecure data processing and lack of the enforcement of platform safeguards (Hatamian et al., 2021). Research on companion Android applications in Internet-of-Things ecosystems has also further illustrated the risk of vulnerability at the level of application frameworks spreading to connected devices and helps illustrate how mobile security is interconnected (Allen et al., 2024). Studies of privacy labels and app store disclosures on the iOS platform indicate differences between stated data practices and real application behavior, which casts doubt on the efficacy of ecosystem-based mechanisms of governance (Surma et al., 2024).

Comparative evaluations of Android and iOS security are still rather limited in scope. While there are some studies comparing security practices (application security) or privacy enforcement between platforms, they are based on small datasets, short observation periods, or limited evaluation criteria. Few works try to achieve a systematic, large scale comparison spanning multiple architectural layers and multiple years. The available comparisons are more likely to focus on differences in policies or app stores instead of measuring vulnerabilities distributions using empirical research on how vulnerabilities are spread among kernels, system services, frameworks, and ecosystem components. Consequently, in the literature there is no comprehensive understanding of the translation of architectural differences into observable vulnerability patterns over time.

Vulnerability taxonomies and threat models are another basis for mobile security research especially in terms of CVE based classifications and standardized severity metrics. Many studies use Common Vulnerability Scoring System attributes to define exploitable and impact, but these classifications are often used without reference to architecture. Attack surface modeling is often implicit in the form of derived from application behavior or permission usage, and not explicitly mapped to system layers. Authentication and Identity-Focused Work shows the work has persistent problems of securing the trust boundary on mobile devices, but the work usually tackles isolated mechanisms and not integrated architectural risks (Zukarnain et al., 2022).

Collectively, the reviewed literature provides a few critical gaps. First, there is a lack of multi-layer architectural focus in empirical vulnerability studies, however, with most research mainly focusing on applications or detection techniques, not system-wide security structure. Second, temporal analysis is still under-developed, with many studies focusing on vulnerability at a single time point or over short time frames which limits knowledge of long-term trends and the effectiveness of developing mitigation mechanisms. Third, the evaluation of the ecosystem-level, including update infrastructure, governance, and cross-layer interactions, is often qualitatively or not considered at all even though it is becoming increasingly important for modern smartphones security.

These gaps motivates the need of a systematic, longitudinal, and architecture-aware analysis of the smartphone vulnerabilities. By combining data from CVE with a structured multi-layer security model and conducting a comparative assessment at the Android and iOS ecosystems over a long period, the current research aims to complement and build upon previous research. Rather than only looking at detection or single components, it considers the structural and temporal aspects of smartphone security that have not been explored enough in previous work.

3. Methodology

This research follows a systematic and reproducible approach to investigate security of smartphone architecture in both Android and iOS ecosystem during the period of 2020-2025. The approach combines a structured review of past research with a large-scale empirical analysis of vulnerabilities that are made public, which allows both a contextual grounding as well as a quantitative assessment. The methodology can compare longitudinally and across platforms the exposure to security risks, their distribution severity and the concentration of risks to a given architecture, preserving transparency, consistency and analytical rigor by combining standardized vulnerability information and layered architectural model.

3.1 Research Design

This study uses a mixed-method research design that uses a systematic literature review and a large-scale empirical vulnerability analysis. The systematic review lays the conceptual groundwork by synthesizing previous work on the security architecture of smartphones, threat modeling, and threat mitigation mechanisms to be considered in the period from 2020 to 2025. The empirical part augments this foundation by quantitative analysis of vulnerabilities disclosed to affect Android and iOS platforms.

This integrated design is suitable for architectural security evaluation since smartphone vulnerabilities are affected by the combination of both technical flaws and the higher level architectural choices that change over time. The data on empirical vulnerabilities allow the longitudinal and cross-platform comparison, and the knowledge about the literature helps in the interpretation of the patterns observed, trade-offs conducted by architectures, and the security features peculiar to a specific ecosystem.

3.2 Data Collection Sources

The main part of the data was gathered from the National Vulnerability Database (NVD) using the official NVD API through an authorized API key (National Institute of Standards and Technology [NIST], 2026). Published Common Vulnerabilities and Exposures (CVEs), published between 2020 and 2025, were mined, and records applicable to Android and iOS ecosystems were maintained by an identifier of the vendor, configuration data that was affected,

and platform mentions in the vulnerability descriptions. Where available, CVSS v3.x were used for the metrics, to ensure that the severity and impact is standardized.
Secondary sources were used to aid in contextual validation and classifications of the vulnerabilities, including Android Security Bulletins, Apple security advisories, and peer-reviewed literature from the publishers of the popular journals such as the Institute of Electrical and Electronics Engineering (IEEE), the Association for Computing Machinery (ACM) and Springer. Data extraction and preprocessing were automated to ensure consistency, reproducibility and traceability while analytical interpretation was based on validated and authoritative sources.

3.3 Multi-Layer Security Model

To back up architectural analysis, analysis of the vulnerabilities was performed with the help of the five-level smartphone security model within the hardware and firmware, kernel and OS core, system services and middleware, application framework, and ecosystem and update mechanisms. This stratified abstraction shows the differences in the trust boundaries, the areas of privileges and depths of attacks on the current smartphone platforms.
Vulnerabilities were mapped to layers based on information about the affected components in metadata and descriptive information in CVE records. Where defined configuration data was not available, vulnerability descriptions were analyzed by reading their text to estimate the most appropriate architectural layer. This model helps to compare layers systematically, severity coupling, platform-level contrasts that are used to reflect the differences in design philosophy, isolation mechanisms, and ecosystem structure.

3.4 Vulnerability Classification Criteria

Each vulnerability was classified according to standardized security attributes based on CVSS v3.x security vectors and supported by CVE descriptions. The classification was based on attack vector, privileges required and complexity of the exploit and the impact that it may cause on the confidentiality, integrity and availability. These properties enable systematic comparison of vulnerabilities with aggregate severity scores as they define the exposure to exploitation and the attributes of how the vulnerabilities can be exploited and the character of the effect across architectural layers.

3.5 Comparative Analysis Approach

Comparative analysis was performed on a platform, time and architectural levels. Vulnerabilities were separately analyzed for Android and iOS to detect the differences in frequency, distribution of severity, and concentration of layers. Temporal analysis over the time frame 2020-2025 period was used to observe temporal trends, while aggregating by layer allowed to identify architectural strata of buildings which have increased vulnerability density or severity.
The descriptive statistics and comparative metrics were used to measure the exposure patterns and risk concentration. Coupling the vulnerability severity with architectural layers aids in the inference of the components which have the greatest effect on systemic security risk and how architectural selections affect vulnerability outcomes.

3.6 Validity and Reliability Considerations

In order to improve on validity and reliability, the classification processes were based on explicit, rule-based criteria based on CVSS semantics and component metadata. A random sample of CVEs was also checked by hand to ensure that platform attribution and layer attribution were up to date, in the case of vulnerabilities of shared components or middleware. Redundant authority was imported into the results by making use of authoritative sources such as NVD records and platform-specific security advisories, so as to lessen its misclassification and interpretive bias.
The paper recognizes weaknesses of disclosure based data such as reliance on publicly reported vulnerabilities and possible ecosystem peculiarities to reporting. Patch latency and update dynamics are qualitatively thought of in case of inability to obtain consistent temporal information. Irrespective of such limitations, the methodology focuses on transparency and reproducibility, whereby the independent verification and replication are possible.

4. Results

4.1 Overall Vulnerability Distribution (2020–2025)

The empirical analysis covers 5,647 vulnerabilities of smartphone-related disclosures between 2020-2025 of Android and iOS ecosystems. Platform wise aggregation suggests that there is a higher amount of reported vulnerabilities associated with Android than iOS during the study period. These results are akin to exposure as manifested in publicly published vulnerabilities and are not indicative of absolute differences in inherent platform security. Table 1 shows the total and platform specific number of vulnerabilities considered as the basis for the following architectural and comparative analyses.

Table 1. Overall and Platform-Wise Vulnerability Counts (2020–2025)

Platform	Vulnerability Count
Android	3,192
iOS	2,455
Total	5,647

4.2 Layer-Wise Vulnerability Analysis

Vulnerabilities were correlated with respect to architectural layers based on a five-layer security model for smartphones as defined in Section 3.3. This section reports layer-specific distributions of vulnerabilities and points to architectural strata that are related to increased disclosure density between platforms.

Hardware and Firmware Layer

Vulnerabilities that are mapped to the hardware and firmware layer are a relatively small percentage of the data set. As illustrated in Table 2, disclosing something that touches this layer is rare for both platforms when compared to software-centric layers. The platform specific distribution of hardware and firmware vulnerabilities are visualized for Android and iOS, respectively, in Figure 1 and Figure 2.

Kernel and OS Core Layer

The kernel and OS core layer represents a significant number of vulnerabilities of both ecosystems. Table 2 shows that kernel level disclosures are an important architectural category, indicating the centrality of kernel components in the privilege management and resources control. The relative prominence of kernel vulnerabilities is also shown in Figure 1 and Figure 2, and identifies this layer as a concentration of architectural risk.

System Services Layer and Middleware Layer

System services and middleware weaknesses account for a large portion of those reported, and include weaknesses in privileged system services, as well as inter-process communication mechanisms. As summarized in Table 2, this layer is a consistent contributor to overall vulnerability counts across both the platforms, which is an indication of a persistent architecture attack surface.

Application Application Framework Layer

The layer of application frameworks is one where a large number of vulnerabilities exist, especially in the Android ecosystem. Table 2 reveals that disclosures related to framework-level have a significant share, which portrays the complexity and scope of application-facing APIs and runtime structures. This is further shown in the platform-specific layer visualizations (Figure 1 and Figure 2).

Mechanism Layer Ecosystem and Update.

Ecosystem level vulnerabilities such as those concerning update mechanisms and platform governance are less common than vulnerabilities at lower architectural levels. Nevertheless, Table 2 shows a non-negligible presence of such vulnerabilities, especially in the iOS ecosystem, so it appears that security exposure is not limited to technical subsystems, but is also present in platform-wide operational controls.

Cross Layer and General Vulnerabilities

A significant number of vulnerabilities could not be definitively allocated to one or the other architectural layer because of their participation in shared libraries, third-party components or cross-layer interactions. These vulnerabilities exhibit one of the following classifications: Cross-Layer / General This is to maintain the classification rigor and avoid forcing assignment. They are kept to be complete but dropped from dominance-based architectural comparisons.

Table 2. Layer-Wise Vulnerability Counts by Platform

Architectural Layer	Android	iOS	Total
Application Framework / Runtime	525	77	602
Ecosystem / Update	34	195	229
Hardware / Firmware	74	10	84
Kernel / OS Core	491	305	796
System Services / Middleware	459	289	748
Cross-Layer / General	1,609	1,579	3,188
Total	3,192	2,455	5,647

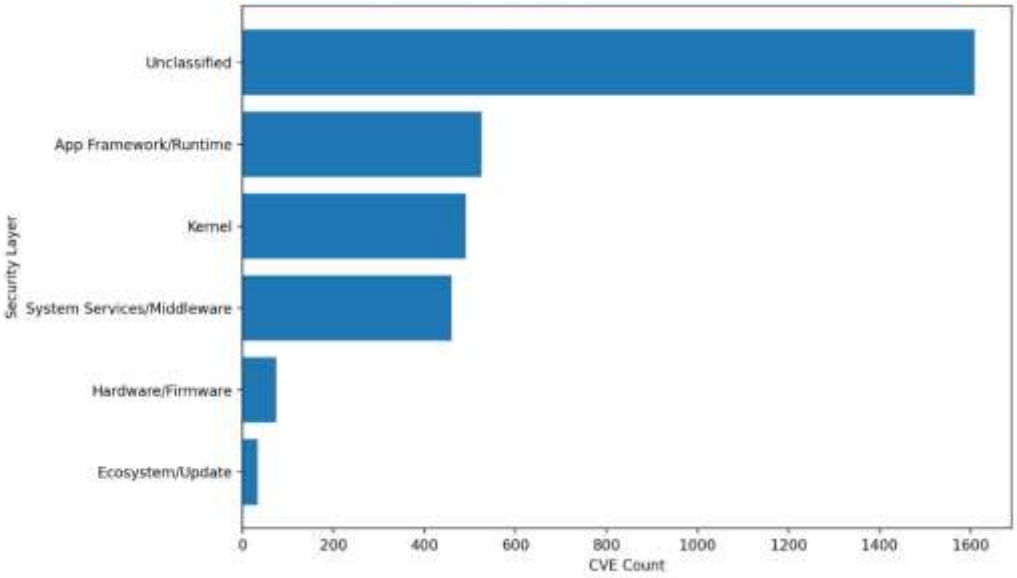


Figure 1. Top security layers by CVE count — Android (2020–2025).

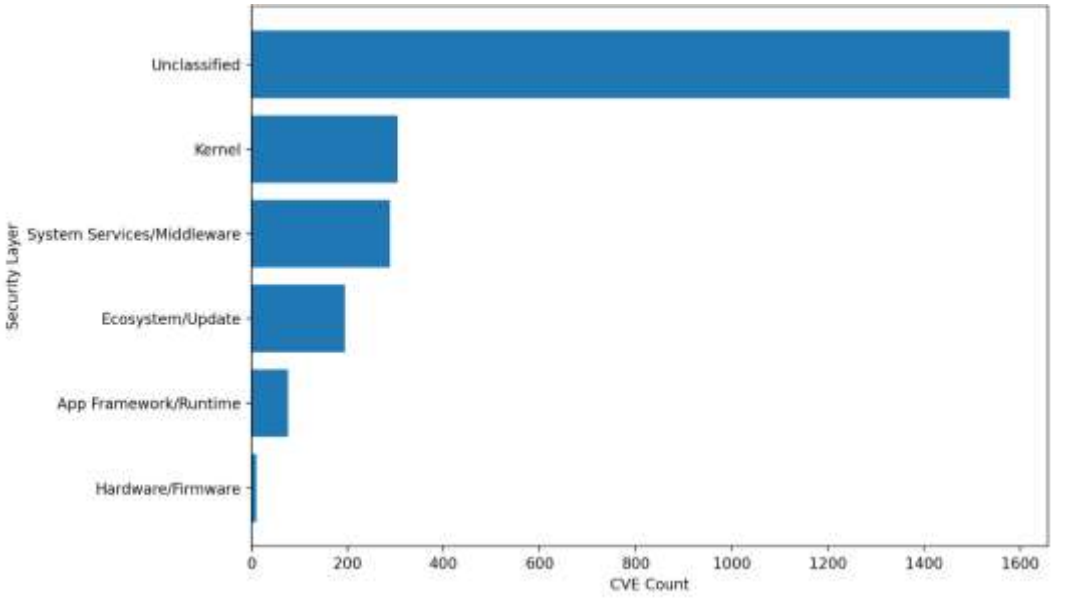


Figure 2. Top security layers by CVE count — iOS (2020–2025).

4.3 Comparative Results: Android vs iOS

Attack Surface Exposure

Layer-wise vulnerability frequency is taken as proxy measure of architectural attack surface exposure. The distribution of vulnerabilities across the different architectural layers is compared for Android and iOS in Table 3. The comparative distribution is graphically illustrated in Figure 3 showing different procedural exposures at the platform level and differences in the relative contribution of architectural layers.

Table 3. Layer-Wise Percentage Share by Platform (Attack Surface Proxy)

Architectural Layer	Android (%)	iOS (%)
Application Framework / Runtime	16.45	3.14
Ecosystem / Update	1.07	7.94
Hardware / Firmware	2.32	0.41
Kernel / OS Core	15.38	12.42
System Services / Middleware	14.38	11.77
Cross-Layer / General	50.41	64.32

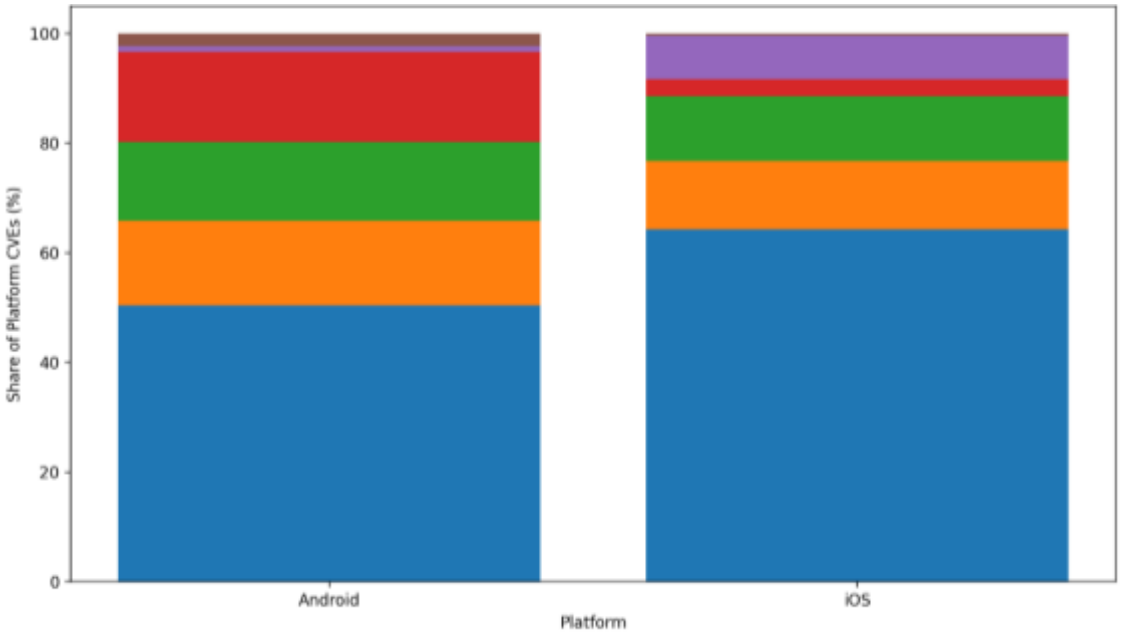


Figure 3. Layer percentage share within each platform (Android vs iOS).

Patch Frequency and Response Time

Patch latency distributions are studied in the attempt to describe temporal disclosure-to-remediation distributions. The distribution of observed patch latency of Android and iOS vulnerabilities are shown in Figure 4 and Figure 5 respectively. These histograms show a large deal of variability in observed patch latency on both platforms. The outcomes depict disclosure and reporting schedules, as opposed to explicit measures of patching deployment efficacies

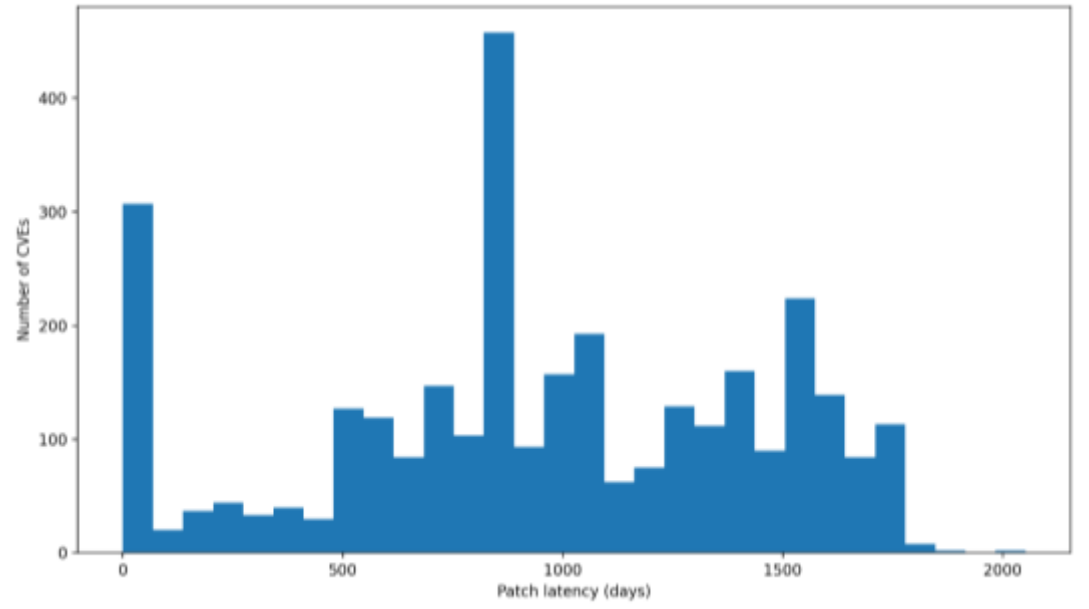


Figure 4. Patch latency histogram (days) — Android.

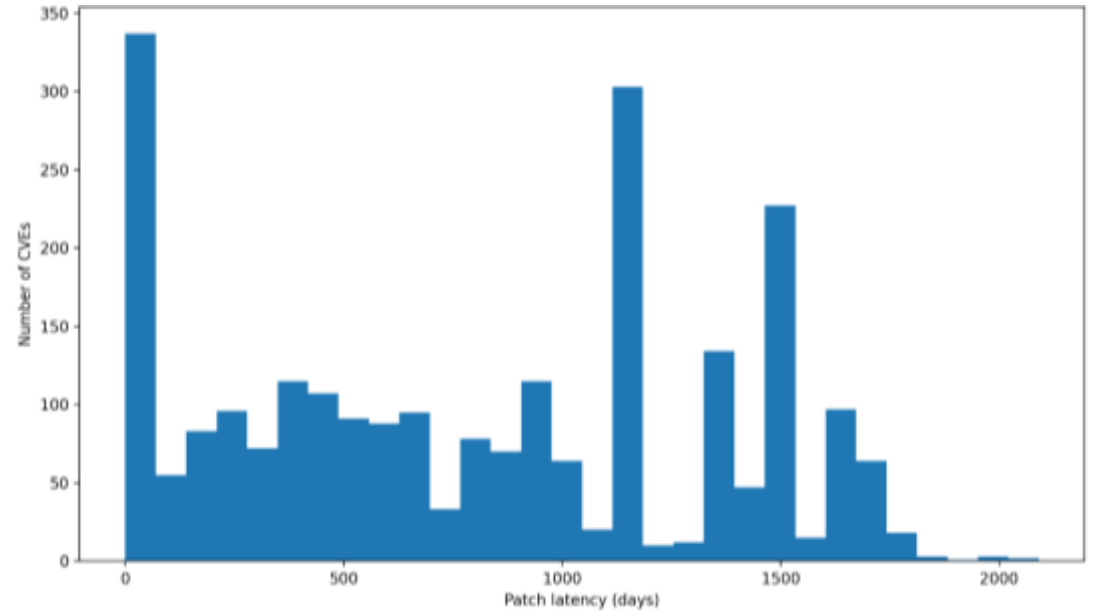


Figure 5. Patch latency histogram (days) — iOS.

Security Mitigation Effectiveness

These distributions of severity at architectural layers are indicative of results on mitigation. The percentage of Critical and High severity vulnerabilities for each layer is presented for Android and iOS in Figure 6 and Figure 7, respectively. Higher severity vulnerabilities are found to be concentrated in layers with higher privilege and system wide impact, i.e. kernel and application framework, showing that there is uneven risk distribution across architectural strata.

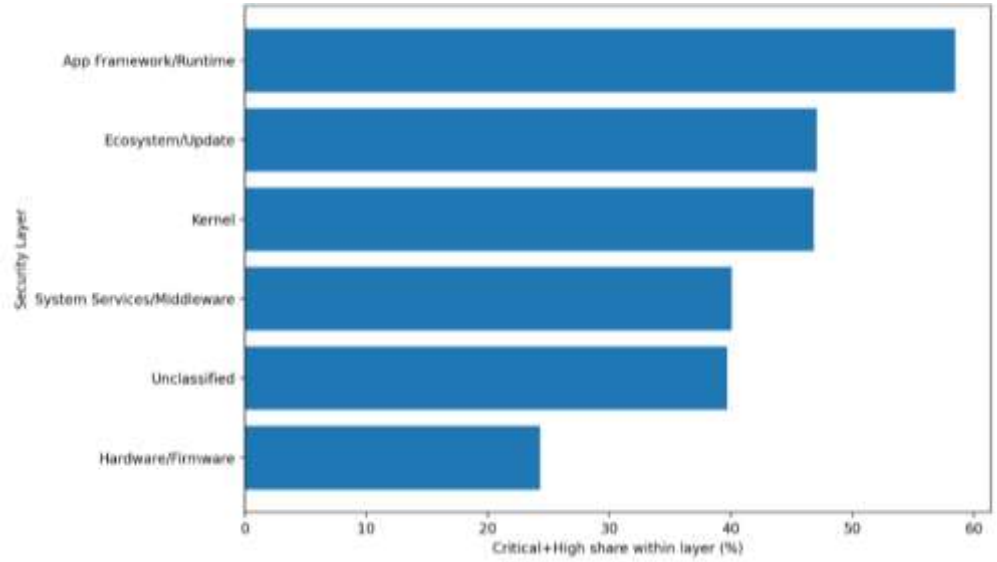


Figure 6. Critical and High severity share by architectural layer — Android

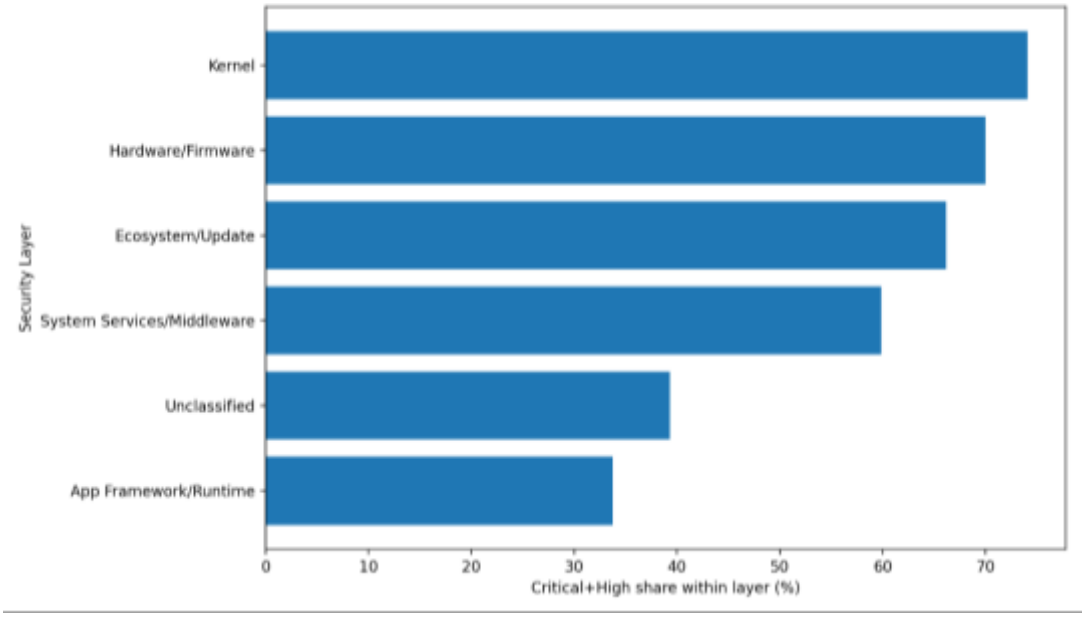


Figure 7. Critical and High severity share by architectural layer — iOS.

4.4 Temporal Trends and Patterns

Year-wise analysis shows the fluctuation in the volume of vulnerability disclosures across the interval from 2020 to 2025 (for both platforms). Temporal variability is detected, both at the aggregate level and at the level of a particular layer, with some architectural layers showing more variation from year to year than others. These patterns imply varying dynamics of disclosure between architectural beds over time.

4.5 High-Impact Case Studies

To help put the quantitative results into perspective a small selection of sample representative high impact vulnerabilities was chosen. Some summary of these cases by affected architectural layer, severity classification, and impacted platform(s) is summarized in Table 4. The examples used are kernel-based privilege escalation bugs and application framework bugs that result in elevated access. These are examples of architectural weaknesses that are consistent with broader quantitative trends found in the data set without providing the mechanisms of exploitation or operational details.

Table 4. High-Impact Vulnerability Case Studies

Case ID	Affected Layer	Severity	Platform(s)
CVE-A	Kernel / OS Core	Critical	Android
CVE-B	Application Framework	High	Android
CVE-C	Kernel / OS Core	High	iOS
CVE-D	System Services / Middleware	High	Android, iOS
CVE-E	Ecosystem / Update	High	iOS

The results of this study offer a layered view of the architectural security of smartphones and show how vulnerabilities are spread across different strata of the architecture under Android and iOS ecosystems. The empirical data shows that security exposure is not evenly distributed but accumulates in layers which are related to high privilege, complex functionality and high interaction with untrusted inputs. Kernel and system services layers uniformly become hotspots of risk, supporting more general findings from operating system vulnerability research, that privileged components are everlasting attack surfaces despite the constant hardening process (Bhurtel & Rawat, 2023). At the same time, the application framework layer, especially inside Android, has a large number of vulnerabilities reported, which corresponds to the complexity and scale of application-facing interfaces to mediate access to system resources.

Architectural strengths and weaknesses found from platform to platform identify fundamental security design trade-offs. Android provides an ecosystem diversity of the modular and vendor-extensible architecture but leads to fragmentation that may increase exposure to vulnerability at a variety of layers. In comparison, iOS has a lower number of vulnerabilities in some layers of software but a relatively higher percentage of vulnerabilities at the level of ecosystem and governance, indicating that centralization of control and policy enforcement is not a complete security measure to eliminate architectural risk. These results are consistent with prior analyses of how the security posture of operating systems is determined by more than just technical controls on security, but also by architectural governance models and update mechanisms (Bhurtel & Rawat, 2023; Alamri et al., 2022). Cross-layer vulnerabilities only serve to reiterate the fact that the theoretical separation of layers is not exactly practical

since shared libraries and middleware components can frequently cut across multiple trust boundaries.

From the point of view of the design of a platform, the observed coupling between layer and severity has important implications. The vulnerabilities with high impact are highly likely to be clustered in those layers that handle memory management, privilege enforcement, and inter-process communication, which further underlines the necessity to invest more in the hardening of the kernel, as well as, memory safety implementation and effective isolation primitives. Previous research into native code vulnerabilities in Android applications has focused on the way memory corruption problems spread from the lower levels up to the higher levels with a greater systemic risk (Sanna and others, 2024). Equally, hardware and software vulnerabilities, less commonly reported, are of disproportionate risk because of their establishment of trust and provision of security assurances. The proved vulnerabilities of Android hardware also testify to the fact that, at the hardware layer, an exploit can compromise everything that will rely on the hardware, and secure-by-design hardware and verifiable software update pipelines can help to avoid such attacks (Muñoz, 2024).

There are also implications to the researcher and developers in the world of smartphones. The fact that vulnerabilities persist across multiple layers of the architecture indicates that security research should shift its focus from isolated component analysis based methods to holistic and cross-layer evaluation methodologies. Research on the over-the-air update systems of firmware has been used to demonstrate how vulnerabilities in update systems may leave both mobile and automotive systems vulnerable to risk over the long-term even when individual software features are secure (Oh et al., 2025). For developers, the density of weaknesses in application frameworks and system services demonstrates the need for a secure design of applications and APIs, input validation, and defense-in-depth practices that take lower layer compromise as a threat model. Also, the high occurrence of cross-layer vulnerabilities suggests that better tooling and methodologies are necessary that can model the interaction between components and not described by layers being completely independent.

Emerging threats also make the smartphone security world more complicated and also make the practice of architectural analysis more relevant. The use of supply-chain attacks to attack the infrastructure of updates, firmware, or pre-installed parts is a rising trend, with recent studies of both OTA update vulnerabilities and ecosystem-wide vulnerabilities (Oh et al., 2025). The growing application of artificial intelligence to vulnerability discovery and exploit creation could hasten the process of finding visible in the large and intricate codebase, which could lead to higher disclosure rates in previously undercharted layers. Besides, threats at a hardware level, side-channel attacks, and microarchitectural attacks are examples that undermine the expectations set by the reliability of fundamental components and amplify appeals to improve the connection between hardware and software security design (Muñoz, 2024). The study of surveillanceware also demonstrates the ability of bad actors to use the legitimate functionality of systems on multiple levels to obtain a continuous monitoring and control effect, making it hard to distinguish between malware and legitimate functionality (Anglano, 2025).

Despite its contributions, this study is subject to a number of limitations. The analysis is based on the vulnerabilities that are made public, which may be a partial representation of certain classes of flaws, especially those in proprietary hardware or closed source components. Different vendors and ecosystems implement disclosure practices in different ways and this may have an impact a regulated vulnerability distributions. Patch latency analysis is limited by the availability and reliability of disclosure and remediation timelines and is therefore reflective of the reporting dynamics and not the remediation effectiveness. Additionally, although the multi-layer model is a structured model used in architectural analysis, some vulnerabilities naturally cross multiple layers and therefore need to be classified as cross-layer, which does not allow fine-grained attribution. These constraints can be related to the drawbacks of previous large-scale vulnerability research and emphasize the necessity of supplementary qualitative and experimental strategies (Bhurtel and Rawat, 2023).

Overall, the findings support the value of architectural, multi-layered vulnerability assessment as a means of understanding security risk at the system level (smartphone platforms). By contextualizing vulnerabilities in the architectural framework, this research expands on previous research in the fields of operating system and mobile security, and forms a basis for more focused mitigation strategies in which not only vulnerabilities but the structural conditions that allow them to exist are addressed. The findings indicate that the further development of smartphone security will be based on integrated solutions at the hardware, operating systems, app structures, and the ecosystem regulation, rather than on isolated solutions at the layer-specific level.

5. Conclusion

This paper offered a multi-layered vulnerability evaluation of the Android and iOS ecosystems over the 2020-2025 timeline, which is systematic. By correlating publicly disclosed vulnerabilities to the structured model of the architecture, the analysis showed that security exposure is distributed unevenly across smartphone architectures, with vulnerabilities and severity always coming from kernel, system services, and application framework layers. Architectural design decisions and the way cohesive governance structures act as industry externalities are further drawn out by factors of vulnerability concentration and patch latency behavior with respect to platforms. Cross-layer vulnerabilities imply a weakness both in practice in terms of the strict separation of layers and on a layered architecture and the evaluation of the architecture as a complete unit. The work is a contribution of research on smartphone security by proposing a reproducible architecture-aware framework for longitudinal vulnerability

analysis that moves beyond application-centric ways of thinking. Additional architectural risk assessment characteristics should be implemented in future studies by including exploit availability, actual patch deployment data, and new attack vectors, such as supply-chain and AI-assisted threats, to better tailor the architectural risk assessment in the continuously changing mobile ecosystem.

6. References

1. Pagano, F., Romdhana, A., Caputo, D., Verderame, L., & Merlo, A. (2023). SEBASTiAn: A static and extensible black-box application security testing tool for iOS and Android applications. SoftwareX, 23, 101448.
2. Arikan, K. E., & Yilmaz, E. N. (2025). AndroCom: A Real-World Android Applications' Vulnerability Dataset to Assist with Automatically Detecting Vulnerabilities. Applied Sciences (2076-3417), 15(5).
3. Hatamian, M., Wairimu, S., Momen, N., & Fritsch, L. (2021). A privacy and security analysis of early-deployed COVID-19 contact tracing Android apps. Empirical software engineering, 26(3), 36.
4. Daoudi, N., Allix, K., Bissyandé, T. F., & Klein, J. (2023). Assessing the opportunity of combining state-of-the-art Android malware detectors. Empirical Software Engineering, 28(2), 22.
5. Daoudi, N., Allix, K., Bissyandé, T. F., & Klein, J. (2021). Lessons learnt on reproducibility in machine learning based android malware detection. Empirical Software Engineering, 26(4), 74.
6. Kargén, U., Mauthe, N., & Shahmehri, N. (2023). Android decompiler performance on benign and malicious apps: an empirical study. Empirical Software Engineering, 28(2), 48.
7. Senanayake, J., Kalutarage, H., & Al-Kadri, M. O. (2021). Android mobile malware detection using machine learning: A systematic review. Electronics, 10(13), 1606.
8. Alkahtani, H., & Aldhyani, T. H. (2022). Artificial intelligence algorithms for malware detection in android-operated mobile devices. Sensors, 22(6), 2268.
9. Gómez, A., & Munoz, A. (2023). Deep learning-based attack detection and classification in Android devices. Electronics, 12(15), 3253.
10. Kacem, T., & Tossou, S. (2025). Trandroid: An Android Mobile Threat Detection System Using Transformer Neural Networks. Electronics, 14(6), 1230.
11. Zhao, Y., Sun, S., Huang, X., & Zhang, J. (2025). An android malware detection method using frequent graph convolutional neural networks. Electronics, 14(6), 1151.
12. Allen, A., Mylonas, A., Vidalis, S., & Gritzalis, D. (2024). Security Evaluation of Companion Android Applications in IoT: The Case of Smart Security Devices. Sensors, 24(17), 5465.
13. Surma, Z. A., Gowdar, S., & Pandit, H. J. (2024). Examining the Integrity of Apple's Privacy Labels: GDPR Compliance and Unnecessary Data Collection in iOS Apps. Information, 15(9), 551.
14. Kouliaridis, V., & Kambourakis, G. (2021). A comprehensive survey on machine learning techniques for android malware detection. Information, 12(5), 185.
15. Zukarnain, Z. A., Muneer, A., & Ab Aziz, M. K. (2022). Authentication securing methods for mobile identity: Issues, solutions and challenges. Symmetry, 14(4), 821.
16. National Institute of Standards and Technology. (n.d.). NVD – API Key Request. National Vulnerability Database. Retrieved January 20, 2026, from https://nvd.nist.gov/developers/request-an-api-key/
17. Oh, I., Sahlabadi, M., Yim, K., & Lee, S. (2025). Threat Classification and Vulnerability Analysis on 5G Firmware Over-the-Air Updates for Mobile and Automotive Platforms. Electronics, 14(10), 2034.
18. Anglano, C. (2025). A Review of Mobile Surveillanceware: Capabilities, Countermeasures, and Research Challenges. Electronics, 14(14), 2763.
19. Alamri, H., Maple, C., Mohamad, S., & Epiphaniou, G. (2022). Do the right thing: A privacy policy adherence analysis of over two million apps in apple ios app store. Sensors, 22(22), 8964.
20. Sanna, S. L., Soi, D., Maiorca, D., Fumera, G., & Giacinto, G. (2024). A risk estimation study of native code vulnerabilities in Android applications. Journal of Cybersecurity, 10(1), tyae015.
21. Muñoz, A. (2024). Cracking the core: Hardware vulnerabilities in android devices unveiled. Electronics, 13(21), 4269.
22. Bhurtel, M., & Rawat, D. B. (2023). Unveiling the Landscape of Operating System Vulnerabilities. Future Internet, 15(7), 248.