

Five Reasons Why You Need a **Virtual Device Platform** for Secure Mobile Application Development





Mobile devices, and the applications that run on them, are more embedded into business-critical workflows than ever before. Yet the security of the applications running on these devices is questionable. Researchers found that in the official Android store, the Goldoson Android malware, which can collect data on installed apps, WiFi and Bluetooth-connected devices, was detected in 60 legitimate apps with 100 million collective downloads.¹ Additionally, according to a recent Verizon survey of enterprise organizations, 59% of IT leaders allow employees to access work email from their personal phones/devices² increasing the risk of a compromised mobile application affecting their organization.

59% of IT leaders allow employees to access work email from their personal phones/devices²

Mobile devices are an attractive target for many reasons, in part because the applications that run on them have access to sensitive data which can be profitable. This risk is so critical that the Open Worldwide Application Security Project (OWASP) Top 10 Mobile Risks for 2024 list of vulnerabilities includes insecure data storage.³ Organizations that develop

and deploy mobile applications for their employees, customers, and partners, need to ensure thorough testing of their mobile applications. Building with data security and compliance in mind, particularly in highly regulated industries, is critical. The risks are significant. There is a delicate balance of developing mobile applications at the speed of the market with building them securely. Data security compromise can lead to loss of business, reputation, and significant legal penalties for compliance violations and class action lawsuits.

Much of the challenge around building security, particularly data security, into mobile application development cycles, is due to the time and cost to perform deep security testing on jailbroken or rooted devices. It is critical to test using these devices to understand how application data is stored and could potentially be exploited. In addition, building, testing, and enabling security for mobile apps is often a siloed process with limited visibility. Developers, testers, and security teams each use their respective tool stack, and sharing information is a manual, time consuming process. Moreover, mobile application security testing tools alone are not sufficient for testing for mobile application data security and compliance.

¹ <https://www.bleepingcomputer.com/news/security/over-200-malicious-apps-on-google-play-downloaded-millions-of-times/>

² Verizon 2024 Mobile Security Index Report

³ <https://owasp.org/www-project-mobile-top-10/>

1.

Physical Devices Are Time Consuming and Have Limitations

Organizations try to solve the challenge of building and testing secure mobile apps by buying and bringing a few mobile devices in-house to test during the development process. However, mobile devices are expensive to procure, difficult to maintain, and introduce long wait times and delays in the development process. The procurement of a physical device is costly financially and resource-wise. The use of physical devices can add days of time to the application development process since the team must spend cycles waiting for procurement and budget approval, shipping time, and configuring and setting up the devices before testing can occur. This method just isn't scalable for testing in a large enterprise with a large volume of applications. Logistics of shipping devices between members and locations causes delays in application development, and securely managing sensitive data on the device also adds a layer of complexity.

There is also the complexity of keeping up with constant testing on the latest mobile device models, which can get costly, especially if multiple devices are needed for testing. And there is also the decision of how many models and which ones to procure to ensure thorough testing. By the end of 2024, Apple had launched 46 distinct iPhone models since 2007.⁴ and Android's approach has resulted in thousands of vendors offering tens of thousands of Android devices. Physical devices have to be managed individually, so testing across multiple configurations requires additional effort and once

devices are purchased, they also require time-consuming maintenance including updates and hardware repairs.

However, the even more significant time allocation for physical devices is the time spent on device configuration and jailbreaking or rooting and set up. Mobile application security testing is allocated a finite window of time, so every hour is critical. Setting up complex network security testing or rooting or jailbreaking to remove restrictions from a device so apps can be installed and tested, takes time away from doing exploit and vulnerability testing of the application. Additionally, devices that run iOS continue to be made more secure and more difficult to jailbreak.

2.

Emulators and Virtual Device Farms are Limited in Scope and Lack Critical Functionality

Simulators and emulators are tools that can be used in developing mobile applications, however they come with limitations, both for iOS and Android devices. An emulator tries to mimic both the hardware and software of a mobile device, while a simulator is focused on replicating the software environment and application behavior, not necessarily the hardware.

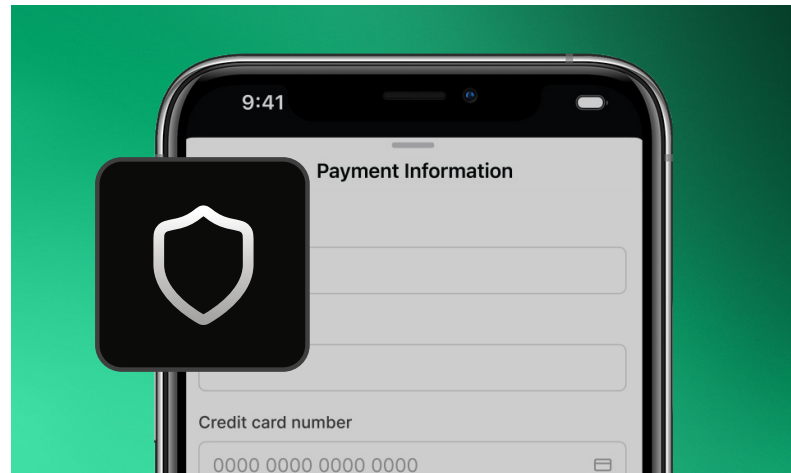
⁴ <https://www.simplymac.com/iphone/how-many-iphones-are-there>

However, both simulators and emulations merely mimic the behavior of a system. They are often limited to only one device or OS platform which makes scale challenging and not suited to large enterprises.

Application development testers have many choices when it comes to Android emulators. Although emulators mimic the software and hardware of an Android device, they do not mimic a rooted device. Testing on a rooted device is critical for mobile application development because it enables testing of application behavior. This includes how the application handles system level modifications, whether the application can identify it's running on a rooted device, and how the application would handle security risks from root access like unauthorized access or tampering with the application.

Emulators are not available for iOS devices, although iOS simulators exist, they have numerous limitations and are only useful during application development. iOS simulators are designed for application development, not testing. The simulator operates in a macOS sandbox so testing at the system level is not possible, and it cannot replicate a jailbroken environment. This means it also doesn't allow for testing for security risks such as how an application would handle running on a jailbroken device, or security risks like unauthorized access or tampering of an application.

Although iOS simulators exist, they have numerous limitations and are only useful during application development.



3.

MAST is Not Enough

Enterprises need to incorporate mobile application security testing for compliance and include mobile applications in their compliance audits as part of their cybersecurity resilience best practices. Building compliance into mobile application security testing is critical, however compliance remains elusive to many organizations, with more than half of CISOs stating that it is not embedded into their CI/CD pipeline.⁵ But building in a compliant way is more critical than ever before due to the increasing number of compliance regulations enacted globally. In addition to countries enacting regulations, there are twenty states in the United States that have enacted their own data privacy laws.⁶ Unaddressed data security issues within mobile applications can put organizations at risk for failure of compliance with security, compliance, (GDPR, HIPAA, PCI-DSS) and data privacy regulations. This could result in potential loss of reputation, legal action, fines and class action lawsuits resulting in millions of dollars of lost revenue.

⁵ The CISO Society 2025 State of Continuous Controls Monitoring Report

⁶ <https://pro.bloomberglaw.com/insights/privacy/state-privacy-legislation-tracker/#states-with-narrow-privacy-laws>

4.

Stronger Operating System Security Results in Less Secure Testing

Industry frameworks for managing these risks and testing for mobile app security are helpful but they are also challenging. OWASP MASVS MASTG as well as CWE and CVE guidelines⁷ are helpful in identifying risks, but they are also time-consuming to apply. Manual testing for these risks can take time away from deep exploitation and vulnerability testing. If the testing time allocated is a week to 10 days, and half of that is spent manually testing for compliance checks, that leaves less time for the deeper analysis that is really needed. Tight application development timelines and time-consuming manual efforts can lead to risk in overlooked vulnerabilities or exploits, especially those for compliance.

MAST scanning tools can be helpful however these tools are not completely sufficient. They may be used in conjunction with a physical device that may or may not be jailbroken, or with a device emulator that has limited scope. Other automated MAST scanning tools try to simulate real device usage but are a black box – with the OS and model of device remaining unknown. These tools don't allow a tester to mix and match various OS and Device combinations for testing an application and may miss findings because they are not run using a jailbroken device.

Additionally, many of these scanning tools only create a static report of findings, such as a pdf file, instead of being delivered on a dynamic platform that shows impact status and evidence of findings and remediation guidance that can be easily actioned and brought into development pipelines.

Ironically, one of the biggest challenges of securely developing and testing mobile applications is the increasing challenge of stronger security being developed for OS releases, particularly for iOS releases. In order to fully test mobile applications for CVEs and other potential exploits and vulnerabilities, testing must be done on a jailbroken device. Yet there is a continued push towards new iOS versions which are increasingly difficult to do thorough testing due to an inability for jailbreaking. iOS 17 began a trend of tighter security controls and iOS and the newest version 18, has seen this trend continue, which means there will not be a public jailbreak for these, or newer versions for Apple's mobile devices. Their aggressive update policy also means there are less and less devices running lower iOS versions. Security assessments for mobile applications can be done on non-rooted or non-jailbroken devices. However, it is much easier to do when there is access to the underlying OS as the root user. Since currently nearly 70% of devices run iOS 18,⁸ this means most application development teams are not doing deep testing for many mobile devices users.

Yet vulnerabilities, and applications with vulnerabilities being uploaded into the Google Play and Apple iStore remain persistent. Just as a few years ago Android malware Goldoson infiltrated 60 legitimate apps in the official Android store⁹ and this year 60 million Android users were infected by the malicious Vapor Application threat on Google Play¹⁰.

7 <https://owasp.org/www-project-mobile-app-security>

8 <https://developer.apple.com/support/app-store>

9 <https://www.bleepingcomputer.com/news/security/android-malware-infiltrates-60-google-play-apps-with-100m-installs>

10 <https://h0thardware.com/news/60-million-android-infected-malicious-google-play>

This year 60 million Android users were infected by the malicious Vapor Application threat on Google Play¹⁰

Additionally, the overall number of common vulnerabilities identified annually continues to grow. Although these are not all related to mobile applications, this trend is concerning, in the last ten years alone CVEs have increased from 7,000 to nearly 40,000 in 2024. These are just the CVEs that are identified, and the capabilities for this identification may decrease. Since 1999 the CVE's mission has been to identify, catalog and define disclosed vulnerabilities. However, because of its operation as a United States government-funded initiative, budget cuts and staff reductions may begin to affect the discovery and sharing of data about common vulnerabilities.

5.

Dependence on Third-Party Development Tools and App Stores Introduces Risk

Mobile application development teams are under pressure to continually build and develop and launch applications quickly. In fact, a recent article noted that more than half of security leaders (52%) say they are getting pressure to support speed at the cost of security¹¹.

As a result, many teams turn toward third-party Software Development Kits (SDK) to simplify their application development process, saving time and costs. SDKs contain reusable components that eliminate the need to write code from scratch and make creating applications easier. However, they also can be full of vulnerabilities within the libraries, code samples, and APIs they contain, and developers should not assume these SDKs are compliant with their organization's security and compliance standards. These vulnerabilities should be particularly concerning for mobile application development teams since they can expose applications to threats such as data breaches and unauthorized access which can lead to data leakage and exfiltration and result in compliance violations.

Similarly, many mobile application development teams are overly confident in their reliance on app stores to secure their mobile applications. There are security models in place by Apple and Google to help secure both iOS and Android mobile applications. However, the responsibility of the security of mobile applications cannot be outsourced to these platforms. Organization development teams are responsible for implementing secure coding practices and protecting the users of their applications, including protecting sensitive data.

Although Apple¹² and Google¹³ review apps for security and malware they don't guarantee protections for users, and developers are legally responsible for any damages that may result from their mobile application. This not only includes security issues such as data breaches, malware or fraud but also includes data privacy and compliance violations. This also extends to security third-party libraries or services used by the mobile application. Most importantly, this responsibility for securing the mobile application is continuous, and exists for as long as the mobile application remains available for download.

11 <https://www.morningstar.com/news/business-wire/20250414055945/organizations-fix-less-than-half-of-all-exploitable-vulnerabilities-with-just-21-of-genai-app-flaws-resolved>

12 <https://support.apple.com/guide/security/about-app-store-security-secb8f887a15/web>

13 <https://security.googleblog.com/2025/01/how-we-kept-google-play-android-app-ecosystem-safe-2024.html>



There is a Better Way to Securely Develop and Test Your Mobile Applications

Corellium Viper with MATRIX. Corellium Viper enables continuous, mobile application security testing, on a virtualized hardware platform, for deeper visibility into data security risk during development. It integrates seamlessly into your mobile application development process and helps identify risk before negative impact to the data security or compliance for mobile applications.

Visit the **Corellium Viper** page to see a demo of Viper with MATRIX at work or register for a free trial to get started today

About Corellium

The leader in Arm virtualization, Corellium helps security and developer teams build, test, and secure software for mobile, IoT, and automotive devices through the power of virtual hardware.

Over 500 organizations, governments and security practitioners worldwide use Corellium's native-fidelity virtualization solutions to strengthen security testing and streamline DevOps. With highly performant, scalable, and accurate virtual models, Corellium enables never-before-possible capabilities not possible on physical hardware.

