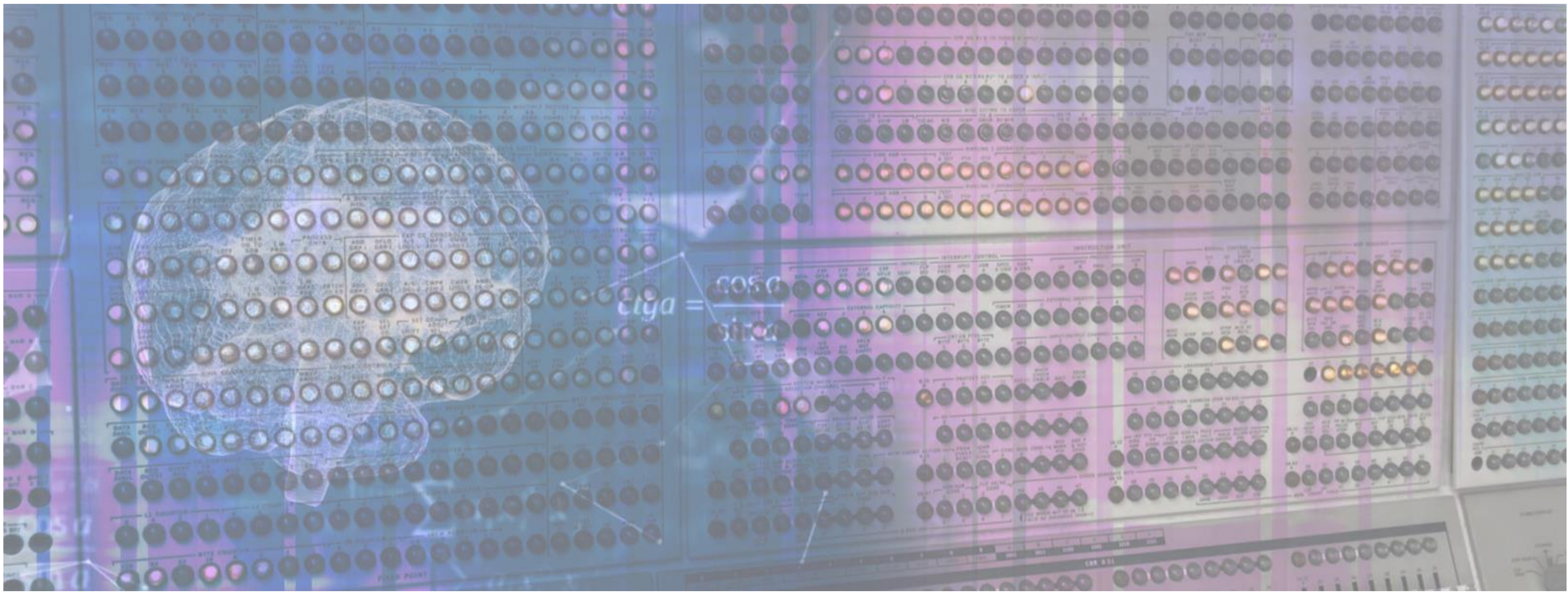


Understanding the Four Quadrants of Mobile App Security

An Intellyx Whitepaper for Verimatrix
by Eric Newcomer, CTO and Principal Analyst
April 2025

Introduction

Mobile ecommerce is growing dramatically, and so is cybercrime targeting mobile applications. Advances in AI technology add fuel to the fire, giving attackers a boost. Security solutions need to be tailored for mobile applications and devices.

Understanding the specific security threats against mobile applications is the first step in effectively countering those threats. You cannot just treat mobile app security as just another part of the overall IT landscape security.

Mobile applications have their own unique characteristics, which you can observe when you use them. They have specialized hardware, operating systems, programming languages, apps, and deployment mechanisms – all of which comprise a unique environment, vulnerable to cyberattack.

Furthermore, mobile devices typically operate outside the corporate security perimeter, requiring a different approach than traditional corporate perimeter centric defense mechanisms.

This whitepaper takes you through the various approaches and techniques you can use to protect mobile apps so that you can determine the best approach for your apps. As a general rule, the more sensitive and important the apps are, the greater the requirement to defend them from attack.

We will review the mobile application vulnerability landscape and discuss various defense strategies, and how to defend those applications from intrusion, break in, theft, and fraud by explaining the four security quadrants:

1. Basic
2. Wrapper
3. Layered
4. Combined

From reading this paper, you should be able to determine which of these security quadrants is right for you. But let's set some important context first.

Build Apps Securely from the Start

Mobile applications are not secure by default. And even though listing an app on the App Store or Google Play store requires a review and a basic security scan, it's not enough to guarantee your app is protected.

Mobile app security must be carefully thought out and built into the app, ideally from the design phase. It's much harder to add security into an application later than it is to develop it securely from the start.

And with the mobile device typically operating in an uncontrolled environment, protection absolutely must be within the app itself to be effective. Many threats can originate from within the mobile device itself.

In fact, those waiting to include security into the app after the app is in the market put themselves in a very risky situation, like building a castle without a moat in the dark ages.

Releasing an app with no protection from attackers risks instant attack. Cyber criminals are very proactive, constantly searching for vulnerabilities to exploit.

Once an app is under attack, the pressure to defend it creates a temptation to react quickly that can easily miss some other protections, and allow attacks to continue in a vicious cycle.

In addition, some attacks are not easy to defend against, such as account takeover attacks, which can originate from specific IP addresses that need to be surgically blocked so as to avoid blocking real traffic.

Determining the right level of security for your app at the beginning, and investing appropriately in it, is critical for success in today's competitive mobile application environment.

Finding the Right Approach to Mobile Security

Many organizations consider adding minimal security because of cost barriers and time and effort constraints. They also may assume that the App Store and Google Play provide sufficient app screening to ensure an app has sufficient security.

But a breach can be costly if it impacts other applications on the device, allows hackers to access back-end systems, and can cause reputational damage even if nothing is stolen. Organizations must ensure customers trust their apps by ensuring they are secure.

The old saying about an ounce of prevention being worth a pound of cure is appropriate here. Another good analogy is driving a car without insurance. It's a lot more expensive to recover from a security related incident than it is to prevent the incident from occurring in the first place.

Advances in AI technology means that hackers now use AI to improve their chances of breaking in and stealing identities and funds. It's important therefore to use AI to counter them as well.

In that light, consider which of the following four quadrants you find yourself in, and ask yourself whether it's the right one for your app.

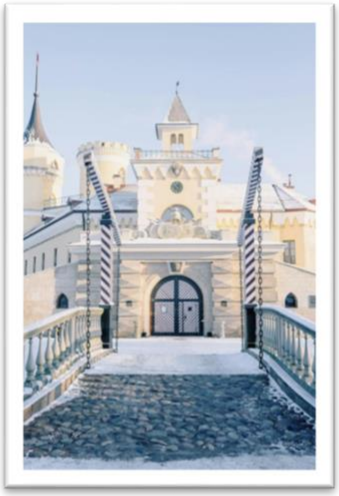
1. Basic Security

It’s tempting to think basic security measures are sufficient, but it can be costly to find out that it isn’t. Organizations frequently pay the price for taking shortcuts.

The misperception that Apple and Google provide security or mandate that only secure apps enter their stores contributes to this sort of overconfidence.

It’s true that both firms care about security – but they focus on how the app plays well with other apps in their ecosystem – they do not verify that in-app protection exists within the app. That is the app developer/publisher’s sole responsibility.

If the app collects any user information – even a minimum such as a username and password, that information can be stolen and used in criminal activities, such as breaking into other apps.



Here a good analogy is a medieval castle that digs a small moat or erects a basic wooden fence. It’s some protection but too easily breached.

Basic security measures include at a minimum using and enforcing strong authentication methods such as multi-factor authentication and biometric login.

Strong authentication methods are usually complemented by data and communications encryption (i.e. HTTPS), and open-source app code obfuscation tools such as ProGuard and Obfuscapk.

These basic protection methods are frequently subject to successful cyber attackers using decompilers, hooking frameworks, and dynamic analysis techniques.

Furthermore, basic runtime threat detection software is easily overridden or defeated by patching the code. So not only is the app still vulnerable to attack, it also may not be able to notify anyone in case it is attacked.

An open-source RASP (Runtime Application Self-Protection) solution, such as Android-RASP provides additional protection against hooking frameworks, man-in-the-middle attacks, app repackaging, and rooted devices.

Free, open-source tools such as these provide basic obfuscation to rename classes, methods, and variables to make reverse engineering more difficult. Some have basic runtime threat detection to identify tampering, debugging, and emulation attempts. Others provide basic static analysis prevention using techniques like string encryption and control flow obfuscation to complicate code analysis.

But these basic tools are easily bypassed by attackers using decompilers, hooking frameworks, and dynamic analysis techniques. If obfuscation only renames elements without encrypting logic, code is reversible with deobfuscation tools.

Runtime threat detection relies on simple checks that can be overridden or patched. Static analysis defenses complicate but do not prevent deeper inspection, allowing attackers to modify or disable protections with advanced debugging and instrumentation tools.

These are all important protections but may not be sufficient. These are foundational protections for identity and data. But attackers continually search for, and find, new vulnerabilities to exploit.

2. Wrapper Based Security

Wrapper based solutions intercept calls to the mobile app to detect and prevent security intrusions from reaching app code. This is a good next level protection to consider.

Wrappers implement basic and some advanced security controls, and typically allow you to select which protections to include in your app.

Wrapper based products supply code to defend against common mobile app attacks, such as account takeover, malicious code injection, anti-tampering, anti-debugging, preventing emulators and simulators, and so on.

However, wrapper based protection requires you to identify which protections you need in advance, and you may end up limited by your selection,

When an attacker reaches the mobile app and tries to break in, for example, the wrapper detects the attack and executes the appropriate code to defend against it. This is however a single point of failure. Once an attacker breaches the wrapper, the application is vulnerable.

These solutions also detect tampering and malware but can lack comprehensive defense mechanisms that may not pass typical PEN testing.

If an attacker manages to penetrate the wrapper however, they can gain control of the app and modify its code, steal its data, penetrate its APIs, and so on.

The analogy here is that you set up an alarm system to alert you if anyone breaks into the castle, but the alarm does not prevent the break in.

3. Embedded Security

Embedded security leverages capabilities external to the app code, allowing you to combine multiple app shielding techniques, such as anti-tampering, code obfuscation, anti-debugging, with environmental checks and external services for more comprehensive app security.

Such comprehensive protection for example adds AI powered detection and SEIM integration to proactively detect and remediate vulnerabilities even external to the application and puts up additional barriers for hackers to overcome.

Embedded security typically also includes cryptographic key protection and vault access, both of which are independent from wrapper code, and complementary to it.

Embedded security also supports industry specific regulatory compliance that can speed deployments,

This level of protection is particularly appropriate and necessary for mission-critical apps.

The analogy here is of a safe room within the castle. Even if someone breaches the defenses and sets off an alarm, apps are still protected. However, it’s not the end of the story, since setting off an alarm within the app still requires a response.

4. Combining Protections: The Fortress

Combining all levels of security defenses protects an app at all levels, from the network touch point through to the running code.

Mobile apps hardened with the wrapped and embedded approach are resilient to static and dynamic analysis. When combined with intelligent monitoring, AI/ML threat mitigation, full SIEM/SOC integration into a DevSecOps infrastructure, threat forensics, auditing, and reporting your mobile app is truly protected.

For example, Verimatrix XTD offers such comprehensive protection within the mobile app, combined with external defense capabilities and proactive threat detection, alerting, and remediation.

XTD anti-tamper protection injects thousands of “micro-checks” that preserve the integrity of an app by detecting and preventing any changes to the app, such as repackaging, runtime manipulation, and code repackaging. If XTD detects any such changes, it stops app execution.

XTD static protections include code and control flow obfuscation, native translation of bytecode into highly obfuscated ARM binary code; string, asset & resource encryption to prevent reverse engineering, basically hiding app internals. These protections prevent reverse engineering app code to expose vulnerabilities or allow attackers to inject code or modify code.

Additional protections guard against “man in the device” attacks, prevent app repackaging, dynamic modification, and reverse engineering mechanisms such as emulators and debuggers.

XTD’s dynamic protections detect rooted or jailbroken phones, debuggers, emulators, and hooking frameworks such as Frida. Anti-tampering preserves the run-time memory integrity of an executing app, which is susceptible to attack and running malicious code.

XTD Network protections prevent Man-in-the-Middle attacks, Network Proxy attacks, DNS spoofing and poisoning and also detects VPNs that might be used to circumvent business rules built around apps to mandate operating in certain geographic regions.

XTD also detects attempts to hide device rooting and defends against hooking attempts to inject code to modify or intercept function calls and gain unauthorized access, manipulate logic, or inject malware.

App integrity features of XTD include app signature checks, dex bytecode interaction checking, anti-tamper checking, and natural library/framework checking.

To enable this combined set of protections, you upload your Android or IoS application to the XTD web portal so that XTD can secure your app. XTD injects thousands of "micro-checks" that detect and prevent app repackaging, runtime manipulation, and IP theft (i.e. code theft).

Once XTD completes injecting the micro-checks and protecting the app, you can download a fully protected version of the app for submission to the Apple App Store or Google Play Store.

XTD also secures app communications to and from the backend servers, safeguards authentication protocols, and prevents access to sensitive databases and systems. This prevents anyone who manages to break into the mobile app from controlling its communication with BE systems and potentially breaking into those systems as well.

Protected apps send real-time threat events and periodic security health telemetry data to the XTD portal dashboard. The dashboard supports real-time reactions and responses to threats, including individual application instances on unmanaged end-user devices.

The monitoring dashboard displays all suspicious events, flagging detected threats, boiling them down to threat instances and number of apps. It also rates threats from low to high to help prioritize the threats.

The dashboard filters events for follow up analysis. It also has a predictive module that uses AI/ML to predict potential threats.

The analogy here is of using the moat, alarm system, and safe room to deter intruders at each level. Hackers basically look for easy wins and will attack someone else if they encounter too many barriers.

The Intellyx Take

Mobile apps are specifically designed for use on mobile devices -- they offer unique touchscreen capabilities and easy to use customer interactions. They make it easier to shop, take photographs, check a bank balance, plan a trip, and correspond with others via email and social networks.

They live in a special ecosystem, are typically developed using programming languages specifically adapted for mobile devices and are delivered to market using dedicated app marketplaces.

Their use for everyday ecommerce continues to explode. Recently developed generative AI technologies are also being adapted to mobile device use, which will extend the simplification of ecommerce activities even further.

All of this adds up however to an environment increasingly exposed to cybersecurity threats and attacks, and an ecosystem increasingly vulnerable to fraud, theft, and cybercrime.

Organizations developing mobile apps need to take a careful look at their security requirements and security posture, especially in light of new AI capabilities and initiatives.

It’s better to invest in mobile app security protection proactively than in response to a costly incident or breach.

Organizations should consider Verimatrix XTD, especially when they need the highest level of security and protection.

About the Author

Eric Newcomer is CTO and Principal Analyst at Intellyx, a technology analysis firm focused on enterprise digital transformation and AI transformation.

Eric was CTO of WSO2 before joining Intellyx in 2023 and was CTO of IONA Technologies until its acquisition by Progress Software in 2008.

Eric is an internationally recognized expert in transaction processing, web services, SOA, and cloud migration. His books on transaction processing, web services, and SOA have been translated into multiple languages and are used as textbooks in universities across the globe.

In financial services Eric served as Global Head of Security Architecture at Citi's Consumer Bank, Chief Architect at Citi's Treasury and Trade Services Division, and Chief Architect at Credit Suisse's Investment Banking Division.

Eric started his career in technology at Digital Equipment Corporation (now part of HP), where he was elected Distinguished Engineer as a Transaction Processing Architect.