

SELF-ASSESSMENT

20 Questions to Assess Your Mobile App Security Posture

Twenty questions for security decision-makers, to be answered honestly and privately. Fifteen minutes will tell you where your mobile apps are exposed.

How to use this assessment

This is a diagnostic, not a quiz. Its value comes entirely from answering it as things actually are rather than as they are meant to be, so it is designed to be filled in privately before it is discussed with anyone.

Who should answer it

One person can complete it, but the answers get sharper with three: whoever owns mobile security, whoever owns the app itself, and whoever would be called at two in the morning if something went wrong. Where those three give different answers to the same question, that disagreement is the finding.

Scoring

Answer	Score	Means
Yes	2	You could demonstrate this today, on request, without preparation
Partly	1	True for some apps, or true in principle but never tested
No or unsure	0	Not in place, or nobody present knows

Score "unsure" as zero. It is not a harsh rule. If the answer cannot be found quickly by the people in the room, it is not available at the moment an incident happens either.

 Four questions are marked  critical. A zero on any of them matters more than your total score, because each one describes an exposure that the other nineteen answers cannot compensate for.



SECTION A

Runtime threat detection

What the app can actually observe about the environment it is running in.

#	Question	Yes 2	Partly 1	No 0
1	Does your app detect rooted, jailbroken, emulated or virtualised environments while it runs? Virtualisation is the newer of these and the one most often missing — several current malware families use it specifically to hide.	☐	☐	☐
2	Does it detect hooking or instrumentation frameworks attached to the running process? Hooking changes behaviour in memory, so nothing on disk changes and file-integrity checks report all clear.	☐	☐	☐
3	Does it detect malicious accessibility services and overlays drawn on top of your screens? ♦ critical This is the single most abused mechanism in mobile banking malware, and static protection cannot see it at all.	☐	☐	☐
4	Is detection behaviour-based, rather than a signature list the vendor updates? A signature list recognises the tools attackers were using when the list was written.	☐	☐	☐

TIP IronWall covers this ground by watching what the runtime is actually doing — rooting, jailbreaking, emulators, virtualisation, hooking, repackaging, debugging and malicious accessibility services all surface through behaviour, so a technique doesn't need to be on a known list to be caught.



SECTION B

Protection integrity

Whether the protection itself survives contact with someone trying to remove it.

#	Question	Yes 2	Partly 1	No 0
5	Has anyone tested what happens when your protection is removed from the binary? ♦ critical Most mobile protection ships as identifiable code with an identifiable entry point. If it can be stripped out, the app runs on and reports nothing.	☐	☐	☐
6	Would you find out if a repackaged clone of one of your apps was in circulation? ♦ critical Clones are distributed outside the official stores, where no review applies, and they produce traffic that looks correct.	☐	☐	☐
7	Is your protection applied so that it still holds after the app has been modified and repackaged? Repackaging is how a clone is built: decompile, alter, rebuild. Protection that only holds for the binary you shipped does not hold for the one your customer installed.	☐	☐	☐



SECTION C

Visibility and evidence

Detection is only half of it. These four ask whether what the app detects reaches your team, and whether you could prove it afterwards.

#	Question	Yes 2	Partly 1	No 0
8	If one of your apps detected a compromised device right now, would your security team know within the hour? Protection that decides on the device and reports nothing leaves you with a quiet app and no idea what it survived.	☐	☐	☐
9	Could you produce a dated, exportable record of mobile threat incidents for the past 90 days without asking a vendor to run it for you? ♦ critical This is what a regulator, a board or a post-incident review will ask for. "We deployed a security product" is not an answer.	☐	☐	☐
10	Do mobile app threat detections reach the same place as the rest of your security telemetry? Mobile incidents that live in a separate portal get reviewed when someone remembers to look.	☐	☐	☐
11	Do you know your current tamper or infection rate across your live user base, as a number? Most teams discover this figure only after switching detection on, and it is usually higher than expected.	☐	☐	☐



SECTION D

Response and user experience

What happens at the moment of detection, and what it costs the customer.

#	Question	Yes 2	Partly 1	No 0
12	<p>Can you set different responses for different threats and different apps, or is it one policy everywhere?</p> <p>The right response for a banking app is wrong for a loyalty app. Blanket policies generate the support tickets that get protection switched off.</p>	☐	☐	☐
13	<p>Do you know your false-positive rate, and does a named person own it?</p> <p>An unmeasured false-positive rate is the most common reason protection is quietly dialled down.</p>	☐	☐	☐
14	<p>When protection blocks a customer, do they see a message written for a customer to read?</p> <p>A message that names what was found and what to do next turns a frightening moment into a trust-building one.</p>	☐	☐	☐

TIP IronSky lets you set exactly this per threat and per app — warn the user, restrict a single function, or end the session — so the response fits what actually happened instead of one setting applied everywhere.



SECTION E

Deployment and release fit

Whether protecting the next release competes with your product roadmap.

#	Question	Yes 2	Partly 1	No 0
15	Can a new app version be protected without engineering effort inside the sprint? Protection that costs developer hours competes with the roadmap, and loses — quarter after quarter.	☐	☐	☐
16	Does protection run automatically in your CI/CD pipeline rather than as a manual step? A manual step is a step somebody eventually forgets, usually under release pressure.	☐	☐	☐
17	If a new attack technique appeared this month, could your protection be updated within days rather than a release cycle? Attack tooling updates continuously. Protection that does not degrades quietly, and the reporting looks the same either way.	☐	☐	☐



SECTION F

Programme and governance

Whether anyone owns this, and whether you know what you are protecting.

#	Question	Yes 2	Partly 1	No 0
18	<p>Do you have a current inventory of every mobile app your organisation publishes, including those owned by other business units?</p> <p>Apps published by a marketing team or a subsidiary carry your name and your risk, and rarely appear on the security team's list.</p>	☐	☐	☐
19	<p>Have you measured what your current protection costs in performance and crash rate?</p> <p>Security, performance and user experience pull against each other. Protection nobody has measured is protection nobody can defend in a product meeting.</p>	☐	☐	☐
20	<p>Is there a named owner for mobile app security, distinct from whoever owns the app?</p> <p>Where the app owner also owns its security, security loses whenever the two conflict — which is at every release.</p>	☐	☐	☐

Your score

Add up each section, then the total. Maximum 40.

Section	Questions	Maximum	Your score
A · Runtime threat detection	1–4	8	
B · Protection integrity	5–7	6	
C · Visibility and evidence	8–11	8	
D · Response and user experience	12–14	6	
E · Deployment and release fit	15–17	6	
F · Programme and governance	18–20	6	
Total	1–20	40	

Score	Where you stand	What to do next
32–40	Strong coverage. Your remaining risk is drift rather than absence — policy tuning, and keeping protection current as attack tooling moves.	Concentrate on Section D and the update cadence in Q17. Re-run this every six months
22–31	Partial coverage, usually strong on detection and weak on evidence and integrity. The most common profile among organisations that already bought something.	Work through your zeros in Sections B and C first. They are the cheapest gaps to close and the ones that hurt most in an incident
12–21	Significant gaps, typically a single-technique deployment doing one job well and nothing else.	Establish what you actually have before buying more. An assessment of your highest-risk app will tell you in days
0–11	Effectively unprotected at the device layer, whatever else is in place server-side.	Start with an inventory and one app. This is a weeks problem, not a years problem, if deployment does not require code changes

Five gaps this usually exposes

Across the organisations we work with, the same five patterns account for most of the lost points.

- 1 Detection without reporting.** The app blocks something, decides locally, and tells nobody. The control works and the organisation learns nothing from it, which also means it cannot be defended at budget time.
- 2 Protection nobody has tried to remove.** Question 5 is the one that most often gets a long pause. Vendors are rarely asked to demonstrate it, so it is rarely tested — and it is the assumption everything else rests on.
- 3 One policy for every app.** Deployed once, tuned never. It generates support tickets on the low-risk apps, and those tickets create the pressure that eventually weakens it on the high-risk ones.
- 4 An unmeasured false-positive rate.** Nobody knows it, so nobody defends it, so the first complaint from a product owner wins the argument by default.
- 5 Apps nobody counted.** A subsidiary's app, a campaign app, an app inherited in an acquisition. They carry your name and your regulatory exposure, and they are almost never on the list.

Take these to your current vendor

If you scored zero or one on a question, the follow-up belongs to whoever supplies your protection today. Each of these is answerable in a live session rather than a document — and asking for a demonstration rather than a statement is the whole point.

Ask	What a good answer looks like
Show me what happens when your protection is removed from our binary.	A live demonstration on a real device. Not a slide
How do you detect a malicious accessibility service or a screen overlay?	A behavioural explanation of what is observed, and why
What if the malware uses virtualisation or hooking to hide?	A specific mechanism, because this is what current families do
Is detection behaviour-based, signature-based, or both?	Both, with clarity about which one catches what
Can we set different responses per threat and per app?	Policy controlled by us, not a vendor default
What do we see when a detection happens, and how fast?	A live incident record with device conditions and action taken
Can we export that for an auditor without asking you?	A dated report readable without a product briefing
How quickly do you ship protection for a new technique?	Days, with a named process rather than a roadmap promise
How much source code do we change to deploy an update?	None. Protection applied to the built app
What does it cost us in startup time, memory and crash rate?	A measured figure stated on the record, verifiable on our own app

About SecIron

SecIron leads the marketing in providing end-to-end mobile app security. Sixteen years in this single category, more than **10,000 apps protected in production, and over 70 Global Fortune 500 organisations** as customers. Our deployment is codeless, which is why a proof of concept runs on your own app in days rather than quarters.

Set against the six sections above, here is where each part of that fits.

Section	What the questions test	How SecIron addresses it
A · Runtime detection	What the app observes about its environment	IronWall detects rooting, jailbreaking, emulators, virtualisation, hooking, repackaging, debugging and malicious accessibility services, through behaviour rather than signature lists alone
B · Protection integrity	Whether you know what you are protecting	IronScan builds the baseline: every app scanned for vulnerabilities, exposed secrets and risky third-party libraries
C · Visibility and evidence	Whether the protection survives removal	IronWall enforces its own presence. Strip it out of the binary and the app alerts, or exits — a capability no competitor fully matches
D · Response and UX	Whether you can see incidents and prove them afterwards	IronSky logs every incident the second it happens across the whole portfolio, with exportable records for audit
E · Deployment fit	What detection costs the customer	Response policy set per threat and per app by you: warn, restrict a function, or exit the session
F · Governance	Whether protection competes with the roadmap	Codeless deployment applied to the built app. No source-code changes, no SDK integration. Updates ship in days

The three products

IRONSCAN

Scans every app for vulnerabilities, exposed secrets and risky third-party libraries — the baseline.

[Discover now →](#)

IRONWALL

Applies protection to the built app and enforces its own presence, no source-code changes.

[Discover now →](#)

IRONSKY

Monitors and responds in real time, with every detection logged for audit.

[Discover now →](#)

What to do next

If your score surprised you, the fastest way to convert it into something actionable is to look at one real app rather than the portfolio. An assessment of your highest-risk app establishes the baseline in days, and it tends to settle internal disagreements that a spreadsheet cannot.

If you would rather talk it through first: what your answers mean, or what your regulator is likely to expect that conversation takes twenty minutes and costs nothing.

Talk to Us

✉ business@seciron.com
🌐 www.seciron.com