Mobile API testing. Where the real damage is.

Most of a mobile app's real attack surface is the API behind it, not the app on the device. The app is a client the attacker controls and can decompile, instrument, and modify. The API is the actual security boundary: it is where authentication, authorisation, and business logic are enforced. The most damaging mobile findings (accessing other users' data, bypassing limits, privilege escalation) are almost always API findings, not client findings. A mobile pentest must include a full API penetration test, or it misses the part that matters most.

HOW IT WORKS

01 How does a tester reach the API?

Before testing the API, the tester has to see its traffic. The steps:

- Intercept the traffic. Route the app's connections through an interception proxy so every request and response is visible.
- Bypass certificate pinning if the app uses it, since pinning blocks the interception proxy. See certificate pinning bypass.
- Map the endpoints. Catalogue every API endpoint the app calls, the parameters each takes, and the authentication each requires.
- Capture authenticated sessions. Get valid tokens for two or more test accounts, so cross-account authorisation can be tested.

Once the traffic is visible and the endpoints are mapped, the API test proceeds the same way a standalone API penetration test would.

02 What does mobile API testing cover?

The same categories as any API penetration test, applied to the endpoints the mobile app uses:

- Broken Object Level Authorization (BOLA). Can one user access another user's data by changing an ID? The most common and most damaging API flaw. See BOLA.
- Broken authentication. Are tokens issued, validated, and revoked correctly? Can sessions be hijacked or forged? See broken authentication.

SOURCES

[1] OWASP API Security Top 10 (2023)

[2] OWASP MASVS

[3] OWASP MASTG Network and API Testing

- Function-level authorisation. Can a normal user reach admin-only endpoints?
- Client-side trust. Does the API re-check the rules the app enforces locally (prices, limits, permissions), or does it trust the client?
- Input validation and injection. Are the API parameters validated, or are they vulnerable to injection?
- Rate limiting. Can the endpoints be brute-forced or scraped? See rate limit bypass.

The full OWASP API Security Top 10 applies. See the API security topics for the detail on each.

03 The most common mobile API mistake: client-side trust

The single most common pattern we see in mobile engagements is the API trusting the app to enforce something the attacker can bypass.

Examples from real engagements:

- The app shows a price and sends it to the API on checkout. The API trusts the price the app sent. An attacker changes it.
- The app hides an admin feature from non-admin users in the UI, but the API endpoint behind it does not check the caller's role. An attacker calls it directly.
- The app enforces a daily transfer limit and sends transactions to the API. The API does not re-check the limit. An attacker removes the client check.
- The app validates input before sending it. The API assumes the input is already validated. An attacker sends raw input.

The root cause is the same every time: a security decision was made on the client, which the attacker controls, and the server trusted it. The fix is the same every time: the server must independently enforce every security-relevant rule, regardless of what the client sends.

04 How does SecureLayer7 test the mobile API?

Every mobile engagement includes a full API penetration test of the backend the app talks to.

We intercept the app's traffic, bypass pinning where present, and map every endpoint. Then we run the API attack matrix: BOLA across multiple accounts, authentication and session testing, function-level authorisation, client-side-trust testing (replaying modified requests the app would never send), input validation, and rate limiting.

The findings are mapped to the OWASP API Security Top 10 and the relevant OWASP MASVS categories, each with a reproducible request and the specific server-side change required. Because the API is the real boundary, this is usually where the report's most serious findings are.

Test the API behind your mobile app.
securelayer7.net/learn/mobile-security/mobile-api-testing

Open online