Global Digital Forensics · Guide edition 2026-11-03 · evestigate.com/insights/application-security-too-often-overlooked-on-the-cyber-front-2/

APPLICATION SECURITY TESTING

Manual Application, API and Source Code Testing Finds the Business Logic Flaws Scanners Miss

A business logic flaw in a customer portal or API lets an attacker read another customer's data through the front door, past the firewall and the scanner. Have the application tested by people who think like an attacker, and know what to preserve if an attacker has already been in. Call (800) 868-8189 at any hour.

Published March 16, 2015 · Updated October 30, 2026

Primary Reference for This Guide

Start with NIST Cybersecurity Framework 2.0.

By Robert Knudsen, Computer forensics and cybersecurity examiner, Global Digital Forensics

How Attacks Pass the Firewall Into Your Application or API

Your enterprise can deploy the most expensive next generation firewall and endpoint detection suite on the market, and the customer facing web portal, mobile backend API or SaaS platform calls for testing of its own, because a business logic flaw there can lead to a serious data breach. HTTPS traffic flows right through perimeter firewalls by design. Automated dynamic application scanners check for basic, textbook syntax errors like simple SQL injection. Multi step checkout workflows, broken object level authorization where one user changes an account number in an API call to view another user financial statements, and subtle authentication token flaws call for a tester who understands what the application is meant to do.

Combining Human Led Application Penetration Testing With SourceScan Code Review

Global Digital Forensics delivers deep, manual web application and API penetration testing paired with expert source code review for software companies, financial institutions and enterprises. Our senior engineers test every user role boundary, API endpoint, session management workflow and microservice integration against OWASP Top 10 and OWASP API Security standards. When clients provide repository access for white box testing, we combine live exploitation with our SourceScan static analysis workbench to pinpoint the exact line of vulnerable code and verify the patch before production release.

© 2026 Global Digital Forensics, Inc. All rights reserved.

What Manual Testing Establishes Beyond Automated Scanning

An automated scanner knows what a vulnerability looks like in isolation. A manual tester learns what your application is supposed to do, and that knowledge reaches the business logic where most serious breaches occur. A manual test begins by learning the application's roles, workflows and trust boundaries, then asks what lies outside each role's authority and tries to do it. Can a customer change an account identifier in an API request and read another customer's statement? Can a step in a checkout or approval workflow be skipped, replayed or performed out of order? Does a password reset token expire, and is it bound to the account that requested it? Can two requests racing each other redeem a single credit twice? Those questions have answers only in the context of the business logic, and the tester who reaches them identifies something specific to your application: the ways the application can be made to betray its own users.

A manual test is also defined by its scope, and a report that states the scope precisely is one counsel can rely on. The scope is set by time, by the roles and environments provided and by the tester's knowledge of the application. A clean report records the result for that scope, that within those roles, environments and dates no exploitable weakness was found, and that precise wording is what counsel can repeat with confidence. Findings are confirmed by exploitation with evidence, so that a developer can reproduce the issue and a general counsel can understand what an attacker could actually have done. We rate each finding by the data it exposes and the conditions it requires, rather than by a generic severity label, because an authorization flaw that reveals one field to an authenticated user and one that lets any visitor enumerate every account are different business risks even if a scanner would score them the same.

Broken Access Control Leads the OWASP Lists for a Reason

The OWASP Top Ten places broken access control at the top of its current list, and the OWASP API Security Top 10 opens with broken object level authorization, the flaw in which an API trusts the identifier a caller supplies and returns the object without checking that the caller may see it. These are common conditions. They arise whenever a developer assumes the front end will only send valid requests, and they pass through perimeter controls because the request is well formed, authenticated and carried over HTTPS. The lists also cover broken authentication, excessive data exposure, mass assignment, security misconfiguration, server side request forgery and unrestricted access to sensitive business flows, each of which we test explicitly by hand.

Counsel care about these lists for a practical reason. When a breach occurs through an application, regulators, plaintiffs and contractual counterparties ask what testing was done and against what standard, and a documented program of manual testing against the OWASP categories, with remediation and retest records, is the answer that holds up. The Federal Trade Commission's enforcement history treats failure to test for well known application flaws as a reasonableness question, and customer contracts increasingly require periodic application penetration testing with reports available on request. Our application penetration testing engagements are structured to produce that record, and when the client provides repository access, source code review with SourceScan traces each confirmed finding to the line of code responsible so the fix is precise and verifiable.

© 2026 Global Digital Forensics, Inc. All rights reserved.

Mobile applications bring the same categories through a different door. The mobile client is one more caller of the API, and attackers go around it. They intercept its traffic, read the embedded secrets and endpoints out of the binary and call the API directly with whatever identifiers and tokens they choose. Testing therefore covers the API as the mobile application uses it, the storage of credentials and tokens on the device, certificate handling and the business flows the mobile client exposes that the web interface may not. Testing the mobile API alongside the web portal often surfaces older versions of the same functions, kept alive so that outdated installations keep working, and those older versions are where we look hardest at the authorization checks.

How an Application Test Runs From Threat Model to Retest

The engagement begins with a threat model conversation rather than a port scan. We map the application's users and roles, its data classifications, its integrations and its most valuable transactions, and we agree with the security and engineering leads which environment will be tested, what credentials and roles we will hold and what is out of bounds. Testing then proceeds role by role. Each role's view of every endpoint and function is enumerated and compared against the others, and every object reference, state transition and token is tested across role boundaries. API testing follows the published or discovered specification, probing authentication, authorization, rate limiting, input handling and the undocumented endpoints that specifications tend to leave out. Where source code is available, the two efforts run together, with the code pointing to risky constructs and the live test confirming whether they are exploitable.

Every confirmed finding is written with reproduction steps, the evidence of exploitation, the data or function exposed and a remediation recommendation specific to the code or configuration involved. The report is delivered in a working session with engineering so that questions are answered and priorities set, and after remediation we retest each finding and document the result. Clients who release frequently often move to a cadence tied to major releases, with the threat model updated as the application changes. The engagement protects the client in two directions: the obvious one, by finding and closing the flaws before an attacker does, and the less obvious one, by producing the dated record of methodology, scope, findings and retests that counsel will need if a breach or a contractual audit later asks what the company did to secure its software.

When the Application Has Already Been Exploited

A breach through an application leaves its record in different places than a network intrusion, and preserving those places quickly is what makes the analysis possible. Web server access logs record every request with its path, parameters, source address, user agent and response code, and an attacker enumerating object identifiers or abusing a workflow produces a pattern in those logs that stands out once you know to look. Web application firewall and load balancer logs add the blocked and allowed decisions. Application logs, if the developers wrote them, record the authenticated user behind each action. Database query logs and audit tables record what was read or changed, where the logging covered it. Export each of these sources with its timestamps intact first, then patch the flaw and rotate the logs, and image the application and database servers if the attacker may have gained execution on them.

The analysis then reconstructs what the attacker did and which records were exposed, which is the question counsel need answered for notification and the question engineering needs answered to fix the right thing. We identify the first exploitation attempt, the technique, the accounts and objects touched and the duration, and we distinguish the attacker's requests from legitimate traffic using the session, address and timing evidence.

© 2026 Global Digital Forensics, Inc. All rights reserved.

The findings reach counsel in a report that follows NIST SP 800-61 Revision 3 in structure and states plainly which conclusions each log source supports. Our data breach analysis page describes the engagement in full, and Robert Knudsen leads the forensic work that turns server logs into a defensible account of the exposure.

Test Your Applications and APIs With Veteran Security Engineers

Call Global Digital Forensics at (800) 868-8189 to scope a manual web application, API or source code security assessment. The initial consultation is free.

What Manual Testing Establishes Beyond Scanning

A scanner knows what a vulnerability looks like in isolation, and a manual tester learns what the application is supposed to do. This table lists the business logic questions a manual test asks, the OWASP category each maps to and what the tester brings to it, so a security lead can see where the serious breaches occur. The last column matters because a well formed, authenticated request over HTTPS passes right through perimeter controls. A clean manual report states its result for the scope and the dates tested.

Business logic questions a manual application test asks, the OWASP category each maps to and what the tester brings to it.

Question the tester asks	OWASP category	What the tester brings
Can a user read another user's object by changing an identifier	Broken object level authorization	Compares each role's access to the same object
Can a checkout or approval step be skipped or replayed	Unrestricted access to sensitive business flows	Knowledge of the intended workflow
Does a reset token expire and bind to its account	Broken authentication	Understanding of the token lifecycle
Can two requests redeem a single credit twice	Business logic abuse	Timed concurrent requests built for the race
Does an API return fields the role should not see	Excessive data exposure	Compares each field with the role's entitlement
Can a request set fields it should not	Mass assignment	Tests each writable field against the role
Does the mobile API carry older unprotected functions	Broken access control	Tests the mobile API alongside the web portal

© 2026 Global Digital Forensics, Inc. All rights reserved.

How to Begin and What Happens on the First Call

Have ready a description of the application or API, its user roles and the data it handles, the environments available for testing, whether source code access is possible, the dates of any previous tests and the contractual or compliance drivers behind the request. The examiner asks about the technology stack, the authentication design, the integrations and the business transactions that matter most, and whether the engagement should include code review or a retest provision. You then receive a written scope describing the threat model session, the roles and environments to be tested, the methodology, the deliverables and the timing, with a proposal. The initial consultation is free, and the hotline at (800) 868-8189 answers 24/7.

Technical References and Further Reading

Earlier reporting on this subject: Forbes, Cyber Security Threats to Information Systems Today, March 12, 2015.

Authorities cited in this guide: OWASP Top Ten. Ranks the most critical web application security risks, with broken access control in first position in the 2021 edition. OWASP API Security Top 10. Identifies the leading API risks beginning with broken object level authorization, 2023 edition. NIST SP 800-61 Revision 3, Incident Response Recommendations and Considerations for Cybersecurity Risk Management. Provides the incident response structure applied when an application has been exploited, final version published 2025.

FREQUENTLY ASKED

Questions Counsel and Security Leaders Ask About Application Security Testing

How is an API penetration test different from a web application test?

The attack surface and the tooling differ, and so does the way authorization is tested. A web application test works through the interface a user sees, following workflows and testing what each role can do. An API test works from the specification or from traffic captured between the client and the server, enumerating every endpoint and method, including the ones the documentation omits, and testing authentication, object level and function level authorization, rate limiting and input handling directly. Most modern applications benefit from both, because the web or mobile front end is one of several callers of the API and an attacker can call the API directly.

Do you need access to our source code?

Source code access is optional, and it makes the test more thorough and the fixes more precise. A black box test with credentials for each role finds the exploitable flaws an outside attacker could find. Adding source code lets us trace each confirmed finding to the responsible code, identify similar patterns elsewhere in the application beyond the paths the live test reached and verify the fix at the line level. Repository access is handled under a nondisclosure agreement and the code is examined in an isolated environment. Clients choose the depth that fits their risk and their release schedule.

© 2026 Global Digital Forensics, Inc. All rights reserved.

Will testing take down our production application?

Manual testing is controlled, and we agree on the rules before it starts. Testing in production gives the truest result because staging environments drift, so most clients test production with limits on destructive actions, with test accounts and with an escalation contact available throughout. Techniques that could affect availability, such as high volume or denial of service testing, are reserved for a maintenance window the client chooses. The rules of engagement name what is in bounds and how testing pauses if something unexpected happens, and the testers follow them.

How often should applications be tested?

Often enough that the test reflects the application as it exists. A full manual test at least annually is the common baseline, driven by customer contracts and compliance frameworks, with additional testing after major releases, architecture changes, new integrations or a change in the data the application handles. Applications that release continuously benefit from a program in which the threat model is maintained and targeted testing follows each significant change. The cadence should be documented, because the dated record of testing is itself part of what counsel will rely on after an incident.

What do we receive at the end of an application penetration test?

A written report that states the scope, environment, roles and dates tested and the methodology followed, lists each confirmed finding with reproduction steps, exploitation evidence, the data or function exposed and a specific remediation recommendation, and records the retest result for each finding after your team fixes it. We walk the report through with engineering and security leadership. Where code review was included, findings reference the responsible source. The report is written so that it serves two readers: the developer who fixes the issue and the counsel or auditor who later needs to show what testing was done.

Web Addresses for the Sources Cited

1. NIST Cybersecurity Framework 2.0: https://csrc.nist.gov/pubs/cswp/29/the-nist-cybersecurity-framework-csf-20/final
2. OWASP Top Ten: https://owasp.org/www-project-top-ten/
3. OWASP API Security Top 10: https://owasp.org/API-Security/
4. NIST SP 800-61 Revision 3: https://csrc.nist.gov/pubs/sp/800/61/r3/final
5. Forbes, Cyber Security Threats to Information Systems Today, March 12, 2015: https://www.forbes.com/sites/sungardas/2015/03/12/cyber-security-threats-to-information-systems-today/

Reviewed by Joseph Caruso, Founder, CEO and CTO, CISSP. Last reviewed October 31, 2026.

© 2026 Global Digital Forensics, Inc. All rights reserved.

Start With a Free Consultation

The initial consultation is free, and requests are handled in confidence. On the first call a senior member of our team listens to the question you need answered, identifies the records and systems involved, and recommends the next step.

Call (800) 868-8189, answered 24/7, request a consultation at evestigate.com/contact, or email info@evestigate.com.

© 2026 Global Digital Forensics, Inc. All rights reserved.