



# IJRASET

International Journal For Research in  
Applied Science and Engineering Technology



---

# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

---

**Volume:** 14    **Issue:** IV    **Month of publication:** April 2026

**DOI:** <https://doi.org/10.22214/ijraset.2026.80531>

[www.ijraset.com](http://www.ijraset.com)

Call:  08813907089

E-mail ID: [ijraset@gmail.com](mailto:ijraset@gmail.com)

# AssessArc: System Architecture and Methodology of a Mobile Workbench for Authorized Security Assessment

Prof. Rushikesh S. Bhalerao, Vishal Satle, Satyam Katkade, Tanvi Sangale, Tanishq Medhane

Department of Information Technology SVIT, Nashik, Maharashtra, India

**Abstract:** We present AssessArc, an Android application structured as a modular workbench for authorized security assessment, oriented toward analysts who lack a conventional workstation. The phone becomes the primary reconnaissance and manual-testing surface for discovering and documenting vulnerabilities in scope. The system integrates rules-of-engagement (RoE) recording, vulnerability-assessment (VA) reconnaissance, manual penetration-testing (PT) aids, offline utilities, and on-device review of saved scan artifacts. Its architecture is deliberately operator-driven: functional areas share governance concepts (gated network activity for the highest-risk emitters, local textual evidence with optional cryptographic fingerprints, and manual hand-off between tools) instead of a monolithic automated pipeline that could obscure intent or exceed scope silently. We articulate (i) the RoE lifecycle and its selective enforcement model, (ii) methodological patterns of the four VA modules and twelve PT capabilities, (iii) offline utilities and the report repository including deduplication semantics, (iv) cross-cutting workflows and persistence trade-offs, and (v) limitations relevant to research reproducibility, security properties of on-device storage, and responsible use. The contribution is architectural and methodological: a coherent field-oriented decomposition aligned with mainstream PT/VA frameworks, suitable for adaptation, extension, or formal verification in future work.

**Index Terms:** Mobile security, penetration testing, vulnerability assessment, rules of engagement, Android, security workbench, reconnaissance methodology, local evidence.

## I. INTRODUCTION

MOBILE devices are increasingly viable as field kits when laptops are impractical, and for many students, practitioners, and field roles without desktop or laptop access they are often the *only* platform available. They are carried during walk-throughs, physical engagements, and travel-heavy assignments. Yet technical testing must still respect *scope*, *authorization*, and *traceability*. Cloud-centric platforms solve collaboration and central policy enforcement but introduce dependency, data residency questions, and operational friction offline. *Local-first* toolchains address connectivity and custody but risk under-documented testing if governance is left entirely implicit.

AssessArc targets that setting without abandoning governance: authorized analysts run reconnaissance scans, manual HTTP experimentation, encoding and analysis helpers, and local review of findings on the handset itself to identify vulnerabilities in permitted targets, all *without* a separate PC toolchain or vendor-hosted backend. The design question is how to structure such an application so that (a) high-risk network emission is prefaced by explicit in-app RoE state,

(b) evidence types remain distinguishable (formal scan exports vs. informal notes), and (c) operators retain fine control over sequencing and interpretation.

### A. Design Principles

Four principles recur:

- 1) *Human-in-the-loop control*. Scans start and stop under explicit user action; modules do not silently chain into one another. This reduces accidental amplification of traffic and keeps causal attribution with the operator.
- 2) *Governance where exposure is highest*. Active network reconnaissance and the primary HTTP laboratory require a recorded engagement before traffic is emitted. Lower-risk or offline transformations remain available under procedural policy.
- 3) *Separation of evidence types*. Flat-file scan exports, encrypted engagement context, and informal workflow notes follow distinct persistence paths, so reviewers can reason about evidential strength.
- 4) *Provenance without over-claiming*. In-app RoE fields and optional metadata footers support traceability but do not replace legal contracts, change-control records, or external ticketing systems.

**B. Contributions and paper organization**

This paper is *methodological*: it explains what AssessArc does, how subsystems interact, and which trade-offs follow, at a level suitable for peer review. Symbol-level implementation detail is intentionally secondary and may be supplied as supplementary material. Section II situates the work. Section III deepens the RoE lifecycle. Section IV describes application composition. Sections V–VIII develop VA, PT tools, utilities, and the report repository. Section IX integrates cross-cutting workflows. Section X discusses limitations and ethics. Section XI concludes.

**II. RELATED WORK**

Structured penetration-testing methodologies, including PTES phases, the OWASP Web Security Testing Guide (WSTG) categories, and NIST SP 800-115 technical examination guidance, emphasizes scoping, authorization, repeatable procedures, and traceable findings [1]–[3]. AssessArc maps naturally to *intelligence gathering, threat modeling, and vulnerability analysis* phases in such frameworks, while leaving exploitation and formal reporting to organizational process outside the app. Mobile platforms impose *sandboxing*, permission models, and storage isolation that shape how tools persist credentials, findings, and network history [4]. Local assessment artifacts therefore inherit platform confidentiality guarantees (app-private storage) but also platform limitations (no built-in enterprise ACL on device, plaintext files unless additionally protected). Prior work on Mobile security tools pans malware analysis, network monitors, and educational labs; fewer publications focus on *field PT/VA workbenches that foreground RoE before reconnaissance*. AssessArc contributes an explicit decomposition: encrypted RoE preferences, gated VA and primary HTTP lab, file-based VA evidence with deduplication, and Shared Preferences-based PT workflow notes; this split mirrors both *risk* and *evidentiary* roles.

**III. RULES-OF-ENGAGEMENT LIFE CYCLE**

**A. Semantic data model**

Before high-risk operations, the application records a structured snapshot intended to mirror a lightweight RoE packet:

- Organizational identity and authorized scope text (human-readable boundaries).
- Ticket or reference and named approver and assessor fields (accountability anchors).
- A validity window: duration from activation, expressed in hours and bounded to a configurable maximum, so that long-forgotten engagements do not indefinitely authorize traffic.

Operators create, update, or clear the record from the home surface; reopening the dialog pre-fills existing values to support renewal or correction mid-engagement.

**B. Confidentiality and freshness**

The record is protected using *encrypted application preferences* (platform-backed key derivation and authenticated encryption). This mitigates *casual disclosure* if device backups or rooted inspection expose preference XML, compared with cleartext key-value stores. It does *not* constitute a hardware security module or enterprise key escrow.

*Temporal validity* is enforced on read: expired records are treated as absent and cleared, reducing the risk that stale scope silently authorizes new sessions. This is a *conservative* default: false negatives (blocked scans until renewal) are preferred over false positives (unbounded reuse).

**C. Selective enforcement rationale**

All VA modules that perform *network reconnaissance* consult engagement state before running.

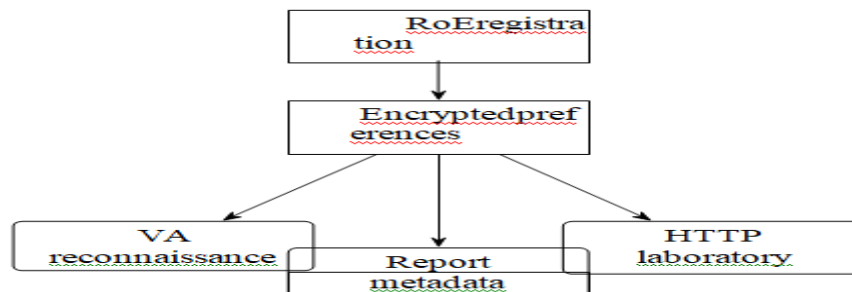


Fig. 1. Engagement registration, secure store, gated network features, and optional report metadata.

The HTTP Request Laboratory in Pentest Tools applies the same check before transmitting requests. Rationale: these paths generate attacker-observable traffic at scale or arbitrary HTTP toward user-supplied endpoints.

Conversely, offline helpers (encoding, hashing, JSON for-matting), paste-based static analyzers, and several auxiliary HTTP testers remain available without that *software* gate. Rationale: (i) benign transformations are often needed during analysis without implying target contact; (ii) some live tests are lower-volume or analyst-judgment-driven; (iii) over-gating can encourage workarounds outside the instrumented path. *Organizational policy* remains the ultimate control; the architecture documents where software enforcement is strict vs. procedural.

#### D. Coupling to save artifacts

When VA or SSL outputs are written to flat files, the system may append an *engagement metadata block* if an engagement exists at save time, echoing organization, scope, ticket, approver, assessor, and validity end. This strengthens *artifact provenance* when reports are exported or archived. It must not be mistaken for standalone legal authorization.

### IV. APPLICATION COMPOSITION

The home surface routes to five areas: RoE registration, utilities (offline helpers), vulnerability assessment, pentest tools (tabbed hub), and saved reports. Navigation is *hierarchical*: each area exposes a hub (card list or tab strip) that leads to specialized screens. This favors *discoverability*, *incremental testability* of modules, and *low coupling* between feature teams or future contributors.

The VA and PT hubs differ superficially (cards vs. tabs) but share the same intent: aggregate many tools without embedding or chestration logic that would couple scan ordering to code paths.

### V. VULNERABILITY-ASSESSMENT METHODOLOGY

The VA suite comprises four modules as presented in the product interface: SSL Check, Directory Scan, Subdomain Scan, and VHost Scan. They address complementary recon-naissance questions: transport and surface hardening, content path discovery, hostname inventory expansion, and name-based virtual hosting on shared infrastructure.

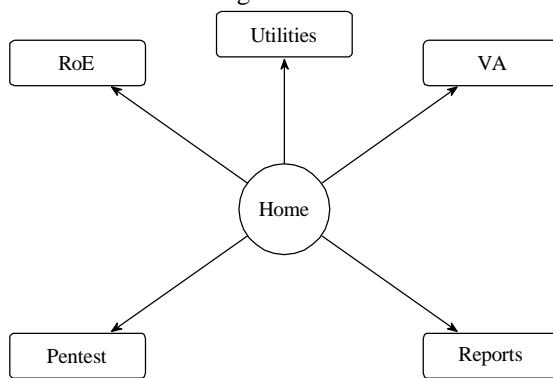


Fig.2. Primary routing from the home surface to major subsystems.

#### A. Shared platform behavior

Modules share a common *HTTP client layer* with config-urable timeouts and redirect policy. *No-redirect* clients are used where automatic following would mask status codes (e.g., redirect classification and brute-force style path probes). A consistent user-agent string reduces fingerprint variance across modules.

*Input normalization* validates URLs and domains, trims wordlists, and composes candidate paths or hostnames. *Up-per bounds* on candidate counts reduce accidental denial-of-service against targets or local resource exhaustion.

Long-running work executes off the UI thread; operators may *stop* iterative scans atomically, preserving partial results and progress messaging.

#### B. SSL Check

SSL Check evaluates a primary URL in layers: (1) for non-HTTPS entry URLs, an HTTP GET *without* following redirects inspects status and Location to infer whether traffic is steered toward TLS;

(2) a TLS handshake yields certificate chain and not-after information; (3) selected security-sensitive response headers are collected; (4) Set-Cookie attributes are parsed for common hardening attributes. A risk score aggregates weighted signals for triage. Persistence is operator-initiated: the analyst saves a snapshot after review, serializing the layered results into a textual report.

### C. SubdomainScan

Given a validated root domain and a label wordlist, the tool forms candidates  $\langle label \rangle . \langle root \rangle$ . For each candidate it performs DNS resolution; on success it may probe  $https://$  and then  $http://$  roots to obtain application-layer status codes. A short inter-request delay may throttle burstiness. Results aggregate hostname, resolved address, and best-effort HTTP status, supporting manual prioritization for follow-on testing.

### D. DirectoryScan

Given a base URL and a path/wordlist (plus optional file-name extensions), the tool composes candidate URLs, requests each, and classifies responses to surface reachable paths. Candidate volume is capped before execution. Interpretation remains heuristic: status codes and lengths indicate candidates for review, not definitive vulnerability.

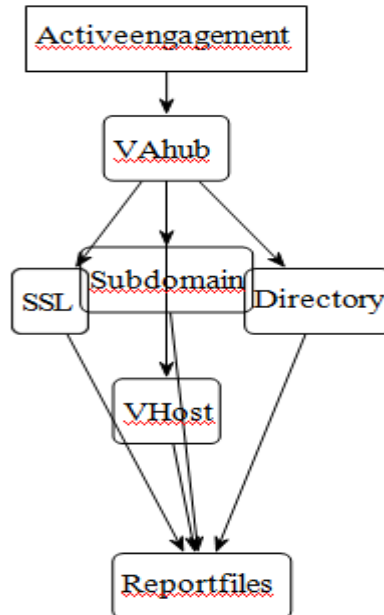


Fig.3. VAhub, modules, engagement precondition, and flat-file outputs.

### E. VHostScan

Given a fixed base URL (often an IP or shared endpoint) and host-header candidates (optionally suffixed with a root domain), the tool establishes a baseline response using a control hostname, then compares each candidate's status and body length (and related signals) to flag divergences suggestive of distinct virtual hosts. This supports shared-hosting reconnaissance; false positives arise from caching, CDNs, and dynamic content.

### F. Evidencepolicy

SSL Check writes reports on explicit save. The three iterative scanners write flat files when the in-memory finding list is non-empty at stop or completion; empty run skip file creation to avoid noise.

A recommended reconnaissance ordering (illustrative only) is: SSL Check → Subdomain Scan → Directory Scan → VHost Scan. The software does not enforce ordering; analysts may iterate or repeat modules as methodology dictates. PENTEST-

### VI. PENTEST-TOOLS METHODOLOGY

Pentest Tools organizes twelve capabilities into four tabs (Core, Analysis, Helpers, and Workflow), as summarized in Table I.

#### A. Networkvs.localtools

Manytoolsoperateonpastedortypeddatawithoutnetwork I/O, supporting *air-gapped analysis* of captured traffic. Live HTTP tools share client conventions with VA. As stated in Section III-C, *only* the HTTP Request Lab requires an active engagement before sending; other live testers rely on operator judgment unless policy extends gates.

#### B. Workflowpersistencesemantics

Workflow notes and checklists persist in *lightweight local preferences*, distinct from flat-file VA reports. This supports session continuity and personal methodology tracking but is *not* presented as a tamper-evident or multi-user evidence vault.

TABLE I  
PENTESTTOOLSTABSANDCAPABILITIES(PRODUCTLABELS).

Tab	Capabilitiesandmethodologicalrole
Core	HTTP Request Lab: arbitrary method, URL, headers, and body with timing and full response capture (engagement-gated before send). Encoder: ordered encode/decode chains (URL, Base64, hex, Unicode, HTML) with optional double pass for chained obfuscation studies. JWT Inspector: Base64url decoding of header and payload [5], calendar expiry interpretation, and heuristic “risk flags” around algorithms and claims (analyst must validate). Diff Viewer: line-oriented comparison in text, JSON, or header modes to summarize additions, removals, and edits after mutations.
Analysis	Response Analyzer: static analysis of pasted response headers/body (and optional probe string) for CSP, CORS, cache, cookie attributes, and reflection hints; intended as triage, not exhaustive policy proving. Auth Header Tester: replays a target URL with Bearer, Basic, and API-key style variants to compare behavior. CORS Tester: issues crafted cross-origin requests to observe browser-relevant policy behavior for configured origins.
Helpers	Wordlist Mutator: generates prefix/suffix/case/numeric permutations for downstream manual use. SSRF URL Helper: composes candidate URLs for <i>authorized</i> SSRF hypothesis testing. Open Redirect Tester: builds redirect-parameter payloads from a base URL and parameter name for manual validation.
Workflow	Scope Notes: free-form target/endpoint/auth/finding-draft text. IDOR Checklist and JWT Attack Checklist: structured checklists with template load/save for repeatable methodology tracking.

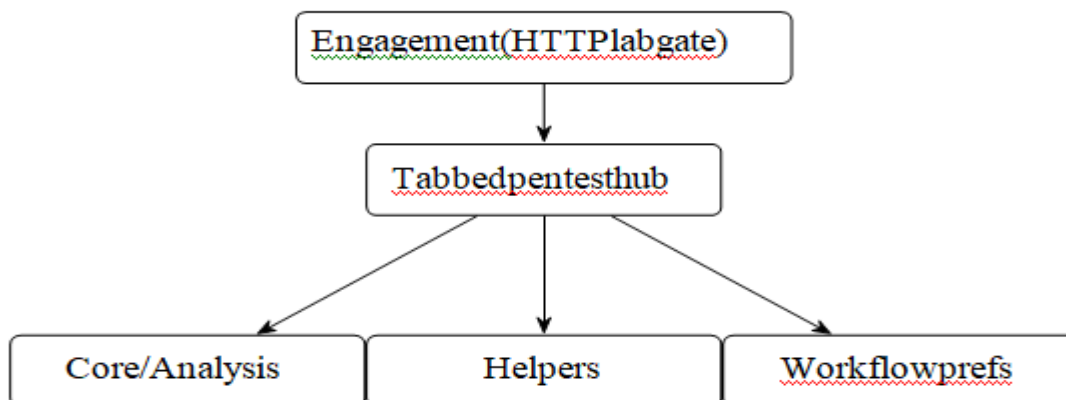


Fig.4. Pentest hub: engagement gate on the HTTP laboratory; workflow state in separate storage.

### VII. UTILITIES METHODOLOGY

Utilities exposes three cards in the shipped configuration: Encoder (the same transformation engine as in Pentest Tools, ensuring consistent semantics across entry points), Hash tool (hexadecimal digests over UTF-8 input with selectable algo-rithms from a configured set), and JSON tool (pretty-printing and syntactic validation). Paths are predominantly *offline* and not engagement-gated in software, reflecting a design bet that benign transforms should remain available while misuse is governed by policy.

Cross-area composition is *manual*: encoded payloads or formatted JSON are copied or retyped into VA/PT surfaces, preserving explicit analyst intent at each hop.

### VIII. SAVED REPORTS METHODOLOGY

Saved Reports is the *reader* for flat textual outputs from SSL Check and the three iterative VA scanners.

#### A. Repository behavior

Files are listed in reverse chronological order by last modification. Substring search filters filenames. Each row expandstoshowfullbodytextondemand. The platform *share* intent exports plaintext to other applications (email, notebooks, ticketing) under user control. Bulk delete requires confirmation and removes report files and associated deduplication indexes.

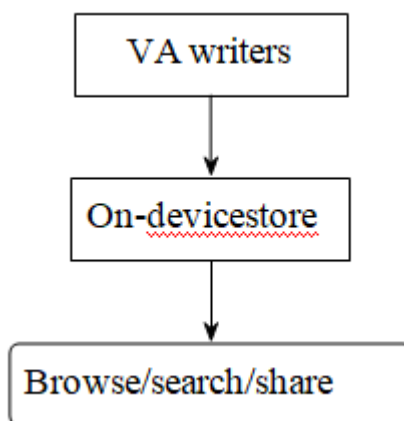


Fig.5. Producers, local textual store, and review UI for saved reports.

#### B. Deduplication and integrity labeling

Each save constructs a typed header (module class, times-tamps, target context, progress), body lines, optional engagement footer, and a *content fingerprint* line derived from SHA-256 over a *canonicalized* representation that excludes volatile timestamp and signature lines. Before writing, the store compares against existing material so that *logically duplicate* reports need not multiply files, which supports repeated runs without clutter while preserving distinct runs when content differs.

Pentest workflow notes do not populate this repository, reinforcing the separation between *scanevidence* and *methodology scratchpad* data.

### IX. CROSS-CUTTING WORKFLOW AND INTEGRATION

A coherent field session may proceed as: register or refresh engagement; run VA reconnaissance and save reports; review exports in Saved Reports; use Pentest Tools for targeted testing (HTTP laboratory respecting engagement); use Utilities for encoding and JSON checks; share artifacts under policy; clear local reports or engagement on handset reassignment.

#### A. Manual composition as a design trade-off

Data do not traverse a single internal pipeline object. Operators bridge tools by copying, retyping, or rereading outputs. This trades *automation efficiency* for *transparency*: each emission or transformation step remains visible and deliberate, which aids training, court-defensible notes, and scope discipline at the cost of integration labor.

#### B. Positioning

AssessArc is best described as a modular workbench: one home composition, heterogeneous persistence models, one cross-cutting engagement concept for the riskiest network features, and file-based evidence for reconnaissance-heavy outputs, with field adaptability preferred over end-to-end orchestration.

## X. LIMITATIONS AND ETHICAL CONSIDERATIONS

- 1) *Governance*: Engagement fields are self-asserted; there is no built-in verification against enterprise ticketing or identity providers. Enforcement is partial (Section III-C). Only one active engagement is modeled.
- 2) *Technical accuracy*: Reconnaissance interpretation depends on heuristics (status codes, lengths, headers); false positives and negatives require human validation. Network results may be influenced by middleboxes, CDNs, and rate limiting.
- 3) *Storage security*: Report files are local and plaintext within the app sandbox; there is no cloud sync or on-device role-based access control. Encrypted preferences protect RoE relative to clear text settings but not all threat models (e.g., full device compromise).
- 4) *Ethics and law*: Researchers and practitioners must use such tools only on systems they own or are explicitly authorized to test, in compliance with law and organizational policy. The tool enables assessment; it does not substitute for authorization.

## XI. CONCLUSION

We described AssessArc as a mobile, modular workbench coupling RoE recording, VA reconnaissance, pentest aids, utilities, and local report review, with deep methodological patterns for each surface. The architecture emphasizes selective gating, heterogeneous persistence, deduplicated textual evidence, and manual composition between tools. Future work may explore external ticket integration, stronger evidence packaging (e.g., signed exports), formal models of engagement state, and broader automation that stays within policy.

## REFERENCES

- [1] Penetration Testing Execution Standard (PTES) Technical Guidelines. <http://www.pentest-standard.org> (accessed 2026).
- [2] OWASP Foundation, "OWASP Web Security Testing Guide," stable released documentation. <https://owasp.org/www-project-web-security-testing-guide/> (accessed 2026).
- [3] NIST, "Technical Guide to Information Security Testing and Assessment," SP 800-115 Rev. 1, 2020.
- [4] Google, "Android Security Overview," Android Open Source Project documentation (app sandbox, storage model).
- [5] M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," IETF RFC 7519, May 2015.



10.22214/IJRASET



45.98



IMPACT FACTOR:  
7.129



IMPACT FACTOR:  
7.429



# INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089  (24\*7 Support on Whatsapp)