GOVERNED TEMPLATE RENDERING REFERENCE: RED-TEAM REPORT DEFAULT V2 (DRAFT)

definition f998a8c2…b10a

RED-TEAM REPORT · PROJECT ARTIFACT

SPECIRA

Red-Team Report: Dispatch Modernization

Meridian Field Services: instructional example, not project evidence

DRAFT · WATERMARK POLICY: DRAFT_ONLY

template red_team_report v2 · pack: specira_default_delivery

the test record to the threat model's prediction; the gaps are the output

§0 What This Document Is

Read this first if you have not seen a red-team report before. It takes one minute and makes the rest of the document mean something.

Meridian has a **threat model**: a document in which engineers sat down, looked at the design, and reasoned about how someone might attack it. That is a *prediction*. This document is the *test*. A tester was authorized to attack the running system for two weeks and to write down what actually happened.

The two documents are meant to disagree in useful ways, and the disagreements are the whole point:

- **The model predicted an attack, and the defence held.** Good: the reasoning was right and the control is real. Recorded as `Refuted`.
- **The model predicted an attack, we tried it, and the control stopped it.** Also good, and different: the control was not merely present, it was exercised. Recorded as `Confirmed`.
- **The tester found something the model never imagined.** This is the most valuable result in the document, because it is the one that design review could not have produced. Recorded as `New path`.

Each finding below is told as a story first (what was tried, what happened, and what it would have meant) and only then summarized in a table. If you read nothing else, read the three narratives in §1.

§1 Findings

mandatory | 1 decision | 1 evidence rule

validators: every_finding_has_plain_language_narrative_before_any_table · narrative_states_attempt_method_response_control_and_consequence · every_named_control_exists_and_is_cited_never_assumed · no_finding_is_only_an_id_and_a_verdict · three_dispositions_present

ENGAGEMENT SCOPE AND RULES

In scope: the dispatch pilot's staging environment, namely the dispatcher console, the technician mobile API, and the telematics webhook ingress at the load balancer (network topology[1]). *Out of scope*: the phase-two customer portal (not built yet) and the payment processor integration, because the pilot handles no cardholder data at all (cardholder-data scope[2]). *Window*: two weeks, authorized, against staging. Staging runs one database tier below production (environment strategy[1]), so **no absolute-throughput claim is made from these results**; the scope note is part of the finding, not a disclaimer. *Disclosure*: coordinated, to the security owner.

RT-01 : Forging a vehicle position Refuted

Why anyone would try this. Meridian's dispatch board decides who is nearest to a job using vehicle positions that arrive from the telematics vendor as webhook calls. If an attacker could invent positions, they could make a technician appear to be somewhere they are not and steer profitable work toward them. The threat model predicted exactly this at the feed boundary (TM-01, PB-1[3]).

What the tester did. Composed a well-formed position for a real vehicle, placed it beside a lucrative job, and posted it to the public webhook endpoint, the same endpoint the vendor uses.

What the system did. Rejected it *before the message was ever parsed*. The vendor computes an HMAC-SHA256 signature over the exact bytes of the request body plus a timestamp header; the API recomputes that signature using a shared secret the tester never had, and compares (SR-021[4]; webhook signature[5]). No valid signature means no parsing, so a forged body never becomes data.

What it would have meant. Had the check been absent, forged positions would have reached the position store and silently biased the board's ranking: a corruption that looks like normal operation, which is the hard kind to notice. **Disposition (refuted):** the predicted attack failed and the control is proven. A negative result, recorded precisely *because* the model predicted it.

RT-02 : A technician reaching for a dispatcher's power Confirmed

Why anyone would try this. Reassigning a job is a dispatcher's authority. A technician who could reassign work could hand themselves the good jobs: the privilege-escalation path the model predicted (TM-03[3]).

What the tester did. Signed in with a legitimate technician credential, obtained a valid session, and called the reassignment endpoint *directly*, bypassing the user interface entirely. This is the important detail: the console never shows a technician that button, but **hiding a button is not a control**. An attacker does not use your interface.

What the system did. Refused the call. Every request is authorized on the server against the caller's role and hub scope, not against what the interface happened to offer (enforcement points[6]; SR-004[4]). The refusal was written to the audit stream, so the attempt is visible to an investigator rather than merely blocked.

What it would have meant. A successful bypass would have let any technician redirect work to themselves: a fraud path, not just a bug. **Disposition (confirmed):** the attack was real, and the control stopped it under attack rather than in theory.

`RT-03` : Replaying yesterday's work `New path`

Why anyone would try this. Technicians work where there is no signal, so the mobile app queues their updates and sends them when the connection returns (offline model[7]). Anything queued can be captured, and anything captured can be sent twice.

What the tester did. Captured one batch of queued updates, waited, and sent the identical batch a second time.

What the system did. Accepted it. A stale update overwrote a newer job state.

What it means. The next full sync would eventually correct the record, so this is not destruction, but in the meantime a dispatcher sees the wrong status and may act on it. The deeper point is *why the model missed it*: the threat model reasoned about attackers trying to **reach** the API, and this is a perfectly legitimate payload, correctly signed and correctly authenticated, simply replayed at the wrong moment. Design analysis asks "who can get in?"; it does not naturally ask "what if a valid message arrives twice?" **Disposition (new path):** the most valuable finding in the engagement, because it is the one no amount of design review would have produced. It drives a threat-model revision and a new security requirement (§4).

The findings, indexed

Finding	Attack attempted	Outcome	Threat model
RT-01	Forged a telematics position for a real vehicle	**Refuted**: HMAC-SHA256 signature check held; rejected before parsing (SR-021[4])	TM-01[3]
RT-02	Technician credential called a dispatcher-only endpoint directly	**Confirmed**: server-side authorization refused and logged it (SR-004[4])	TM-03[3]
RT-03	Replayed a captured batch of queued mobile updates	**New path**: accepted; a stale update overwrote a newer state	none → drives TM-09[3]

§2 Severity

mandatory · 1 decision · 1 evidence rule

validators: severity_anchored_to_risk_register_scale · no_report_only_severity_scale · reachability_qualified · false_positive_rejected_with_evidence

Severity uses the risk register's scale, the same likelihood-times-impact currency every other Meridian risk carries (scoring scale[8]). That is deliberate: the sponsor can weigh a red-team finding against a delivery risk without translating between two vocabularies, and the translation is where judgment quietly gets lost. A report that invents its own severity words forces exactly that translation.

Finding	Likelihood, and why	Impact, and why	Net
RT-03	**Medium.** The attacker must first obtain a genuine batch: either a technician's unlocked device, or a position on the network path while it is in flight. Neither is exotic; neither is available to an anonymous stranger.	**Medium.** A work order shows a stale status until the next full sync corrects it, so a dispatcher may make one decision on wrong information. Disruptive and confusing; not destructive, not permanent.	Medium
RT-01 RT-02	No residual severity: the attacks ran and the controls held. They are recorded anyway, because **a tested path with no finding is evidence**; its absence from this report would later read as a path nobody thought to try.		None

Reachability, stated, because severity without it misleads. RT-03's replay must be delivered over the authenticated mobile API, so it requires a valid session. It is not reachable by an unauthenticated stranger on the internet. A "medium" that a stranger can trigger and

a "medium" that requires a stolen session are not the same medium, and the register's number alone cannot tell them apart.

> **FALSE POSITIVE: REJECTED, WITH EVIDENCE**
>
> The tester observed a metrics endpoint answering without authentication and flagged it as an exposure. It is not one. That endpoint is bound to the application tier inside the private subnets, which has no public address and is reachable only from the load balancer's security group (network topology[1]); the tester reached it only because they were already inside the tier. It is recorded here rather than deleted so a future reader sees the question was asked and answered: **an unrecorded rejection is indistinguishable from an oversight.**

§3 Resolution

mandatory · 1 decision · 1 evidence rule

validators: finding_not_resolved_without_passing_retest · retest_verifies_by_rerunning_attack · unfixable_finding_promoted_to_risk_register

One rule governs this section, and it is worth stating plainly: ***a finding is not resolved when the patch merges. It is resolved when the original attack is run again and fails.*** *Reading a diff tells you the author believed the fix works; re-running the attack tells you it does.*

The fix. Each queued batch now carries a sequence number that the server tracks per device, so a batch it has already applied is recognized and discarded instead of replayed.

The retest. The identical captured batch that succeeded during the engagement was sent again. It was rejected, and the newer job state survived. *Only then* was RT-03 marked resolved. Owner: E. Sandoval.

Finding	Status	Owner	Evidence that closed it
RT-03	Resolved	E. Sandoval	The captured replay was re-sent post-fix and rejected; carried on the register as RSK-014[8] while the fix was in flight, closed on the passing retest
RT-01, RT-02	No action	none	Controls held under attack; nothing to fix

Why it sat on the risk register at all. That promotion is what a finding gets when it cannot be fixed immediately: it puts the open weakness somewhere the sponsor already reads, rather than only in a report that gets filed. **And because RT-03 was a path the design analysis missed, it produced two permanent changes rather than a patch alone:** a new threat-model row for replay of a legitimate queued payload (TM-09[3]), so the next model review starts from it, and a new security requirement (§4).

§4 Remediation Matrix

mandatory | 1 decision | 1 evidence rule

validators: remediation_maps_each_finding_to_sr_id · remediation_cites_forward_verification · new_path_drives_new_security_requirement

This matrix is the bridge back into the design; it is what stops a red-team engagement from being a document that is read once and filed.

Finding	Becomes
RT-03	NEW SR-031[4]: every queued batch carries a sequence number; the serv
RT-01, RT-02	No remediation; the controls held

Why a requirement and not just a fix. The fix lives in today's code; the requirement governs whatever replaces that code. Recording RT-03 as SR-031 means the protection survives the next rewrite.

Why two forward checks and not one. They answer different questions. The regression test proves *the control still exists*: it fails the build if anyone removes it. The alert proves *whether anyone is actually trying it* in production. Neither substitutes for the other.

The reconciliation closes. Every real finding traces to a security requirement and a forward check; the threat model gained the row it was missing; the risk register opened and closed the issue on evidence rather than on assertion.

§5 Open Questions

mandatory 1 decision

Question	Owner	Answer by	Blocks
Should the next engagement include the phase-two portal's payment redirect once it is built? The payment spec flags a skimming path against that page (incident playbook[2]), and it is exactly the kind of surface this engagement could not reach.	The security owner with the sponsor	At the portal integration design	The next engagement's scope, not this report's sign-off

Refs References & Package Contents

In the Specira workspace

specira [1] Infrastructure & deployment: the network topology register (the private subnets, the load balancer's webhook listener); the environment strategy app.specira.ai/projects/dispatch-modernization/artifacts/infrastructure-deployment#network-topology

specira [2] Payment security & PCI spec: the cardholder-data scope; the skimming incident path app.specira.ai/projects/dispatch-modernization/artifacts/payment-security-pci

specira [3] Threat model: TM-01 (feed spoofing at PB-1), TM-03 (privilege escalation), and the new TM-09 (replay of a legitimate queued payload) app.specira.ai/projects/dispatch-modernization/artifacts/threat-model

specira [4] Security requirements: SR-004 authorization, SR-021 feed authentication, and the new SR-031 batch-sequence validation app.specira.ai/projects/dispatch-modernization/artifacts/security-requirements

specira	[5] API contracts: the webhook signature scheme (HMAC-SHA256 over the raw body plus timestamp, 300-second replay window) app.specira.ai/projects/dispatch-modernization/artifacts/api-contracts#webhook-signature
specira	[6] Auth & authorization policy: the server-side enforcement points RT-02 tested app.specira.ai/projects/dispatch-modernization/artifacts/auth-authz-policy#enforcement
specira	[7] Mobile platform spec: the offline-tolerant model and its queued-batch sync app.specira.ai/projects/dispatch-modernization/artifacts/mobile-platform-spec#offline
specira	[8] Risk register: the scoring scale; RSK-014 app.specira.ai/projects/dispatch-modernization/artifacts/risk-register
specira	[9] CI/CD pipeline: the security regression suite that replays the captured batch app.specira.ai/projects/dispatch-modernization/artifacts/cicd-pipeline#security
specira	[10] Observability: the rejected-duplicate-batch alert app.specira.ai/projects/dispatch-modernization/artifacts/observability-monitoring#alerts

Generated by Specira · template red_team_report v2 (draft) · pack specira_default_delivery lineage f998a8c2…b10a · page 1 of 6