Phoenix5981 Security

Independent Security Research — willworachat@gmail.com

AUDIT COVERAGE SUMMARY

Report date: 2026-07-13
Target: Lightspark
Platform: HackerOne (Lightspark BBP)

Lightspark — Audit Coverage Summary

Target	Lightspark (Spark network — self-custodial Bitcoin L2 token protocol)
Platform	HackerOne — Lightspark BBP
Engagement dates	2026-05-23 (first submission) – 2026-07-13 (final verdict, program closed out)
Scope tested	Spark Operator (SO) gRPC services — SparkService (transfer/settlement), SparkTokenService (token create/mint/transaction lifecycle), authentication layer
Outcome	2 findings submitted, both closed with no payout; program fully worked and pivoted

Methodology Applied

- **H3 — signature/replay checklist:** reviewed gRPC message signing, per-operator payload hashing (V1 token transaction hash), and session authentication for replay or forgery gaps.
- **H8 — boundary conditions:** probed transaction validity windows, in particular the minimum-allowed `validity_duration_seconds` (1 second) against the mint-supply accounting path.
- **H6/H16 — state-machine and DoS-elevated analysis:** traced the SIGNED → FINALIZED transaction status lifecycle end-to-end, including the backfill task that promotes SIGNED to FINALIZED, looking for windows where stale or expired state is trusted.
- **Direct source review:** read the actual Go implementation (`spark/so/tokens/validation.go`, `ent/tokentransaction_extension.go`) rather than black-box probing alone, including an in-code TODO and a disabled unit test flagging a known gap.
- **Live PoC discipline (L1/L5/L6):** both findings were confirmed against the live mainnet gRPC endpoint (`0.spark.lightspark.com:443`) with a minimal Python PoC and actual observed request/response output, not simulated.

Findings Summary

#	Finding	Reported Severity	Outcome
F1	Broken gRPC array-length validation on SparkService enabling authenticated denial of service	—	DUPLICATE
F2	Expired SIGNED mint transactions still counted toward a token's `max_supply`, permanently blocking further minting once the (expired) SIGNED mint exhausts the cap (#3756591)	High	INFORMATIVE

> **Triager note on F2:** the underlying behavior was independently confirmed with a live PoC (create token → mint full supply with a 1-second validity window → wait for expiry → confirm the expired SIGNED mint still blocks all further minting). HackerOne triage closed this Informative on the basis that the described actor is exclusively the token's own issuer, the minted tokens remain fully usable, and reaching a hard supply cap after a legitimate full-supply

> mint is expected terminal behavior rather than a security-relevant denial of service — impact is confined to the issuer's own resource, with no path for a third party to trigger or be harmed by the condition.

Coverage

Both findings were the product of full source-level review of the token transaction validation and status-transition logic reachable from the public gRPC surface, plus targeted testing of the signature/session layer already covered under an earlier engagement pass. No further unauthenticated or authenticated attack surface remained to explore on the SparkTokenService/SparkService boundary within program scope; the engagement is considered complete.

Outcome

$0 confirmed. Both submissions closed (Duplicate, Informative) with no impact to reporter Signal/Reputation score. Program marked complete and deprioritized in favor of active targets.

Phoenix5981 Security — Independent Security Research · willworachat@gmail.com · This document summarizes research conducted under the Lightspark HackerOne bug bounty program's responsible disclosure terms.