BATTLE TOWN

26150901

solidity

Executive summary

Manual source review, exact compilation, baseline and instrumented execution, production-only coverage analysis, targeted EVM tests, bounded stateful campaigns and static-analysis classification were performed for the supplied BATTLE TOWN source snapshot.

The review identified one Medium and four Low technical findings, together with two Informational observations. No Critical or High-severity findings were identified within the reviewed source snapshot and the methods described in this report.

Audit scope

The review covers the supplied Solidity source snapshot and its technical implementation: access checks, state transitions, arithmetic, external calls, repeated execution, resource bounds and interactions between the supplied contracts.

The official BATTLE TOWN Whitepaper v3.3 was used to identify the intended contract set and its declared technical capabilities. Correspondence checking was limited to that purpose. Business assumptions, economic viability, game rules and commercial merits were not audited.

Production deployments, deployed bytecode, live balances and addresses, administrator conduct, key custody, external-service configuration, actual game results and operation of the final project were outside scope. Local deployments used by tests were execution fixtures, not a deployment audit.

Imported and inherited dependencies were considered where their behavior affected the reviewed contracts. This did not extend the review to every contract in those third-party packages. Deployment scripts, mocks and baseline tests were supporting material rather than separately audited production components.

Files in scope

No.	File	SHA-256
1	contracts/BattleTownConstants.sol	a52a68ca54daecb18a1abe7387794e97d7d3c3461ee02a9b13e9926c91275b03
2	contracts/BattleTownNftRewards.sol	7912b6373549ea314347aa4b46c7564e15085d70674b3582b67fe6b5373d817a
3	contracts/BattleTownPresale.sol	0060c059c5d7a758eff984838aa95e06bef175a00cf0f8545ccbb5fd69a27cc7
4	contracts/BattleTownRewardsVault.sol	b64e32de9469a53aa920f4cf474d67f54e9e32b62b71d4f1e7eca4c689fc3ebb
5	contracts/BattleTownTank.sol	d56372195180c923e9b72a4a98fd87612bcb12cb013e1f9fff17f3a37d3be69e
6	contracts/BattleTownTankLens.sol	0df7c84f4ffc7fa0ddb4be07bdaef5a32a4fdb0bc48da374fe774ec25fc6a0dc
7	contracts/BattleTownTankSale.sol	4a2dee4efd097580345b1a67b133b6ab09dc468f02d6b2bbd633017bb69db91d
8	contracts/BattleTownToken.sol	0534b62cc24d8013e2b908529207d310c97555a2168190d17d849c5bd5236140
9	contracts/BattleTownTournament.sol	ebf3fcf19af4a8ad6377d222bd52656940567e40d72efed019b6777e795423c3
10	contracts/BattleTownTournamentLens.sol	f31f4f9eaf449013abaf13852ec2e37dccdd663053772c791709c8769a88d80b
11	contracts/BattleTownVesting.sol	dbf6180189c3812c1084a0c55b47b6ef362a7f121e904127539d71ba510c53e0
12	interface/IBattleTownRewardsVault.sol	86c8576ec83362ed942e5c2d265d6cae8e29e7824a326e1f8ee3959201f8d41b

No.	File	SHA-256
13	interface/IBattleTownTankMint.sol	10caf67ce6e5e991fe9558417e635a7e8851a29875385b98d233bd3e126e711c
14	interface/IBattleTownTankView.sol	65807015797242e32279662bebe7bb6ee513b062af4fc4dd61c95e9f0969efb8
15	interface/IBattleTownVesting.sol	67eedfb0d3da0eae291c3ce5dfeecd81cd043d81183ed5415dad939e75e276c2

SHA-256 identifies the exact UTF-8 bytes of each reviewed source file. The `contracts/interfaces` symlink resolves to the same four interface files.

Methodology

The review combined:

- exact input hashing and source reconciliation;
- manual inspection of all first-party production files and interfaces;
- compilation with the locked Solidity, optimizer, viaIR and Cancun profile;
- the supplied baseline suite and targeted reproduction/control tests;
- production-only statement, branch, function and line coverage;
- bounded Echidna stateful campaigns and deterministic closure sequences;
- local resource-bound and bytecode-size measurements;
- Slither classification and targeted compiler-advisory screening;
- structural whitepaper correspondence; and
- a scoped SCSVS/SCSTG evidence map.

The nine principal Echidna runs used three campaigns, three fixed seeds per campaign and 600 seconds per run. Twenty main properties completed without a discovered counterexample. The models were bounded by their actions, actors, sequence

reachability, seeds and duration. Large Presale and Tank caps, coordinator rotation, BT-L01 ownership transfer and several direct per-pair accounting properties were not all reached in the original campaigns; deterministic closure tests separately exercised the material identified gaps.

Coverage gaps primarily consisted of inlined reentrancy-rejection arms and defensive states that are unreachable through the reviewed lifecycle. Coverage, fuzzing and static analysis were treated as supporting evidence rather than proof that defects are absent.

Contract architecture

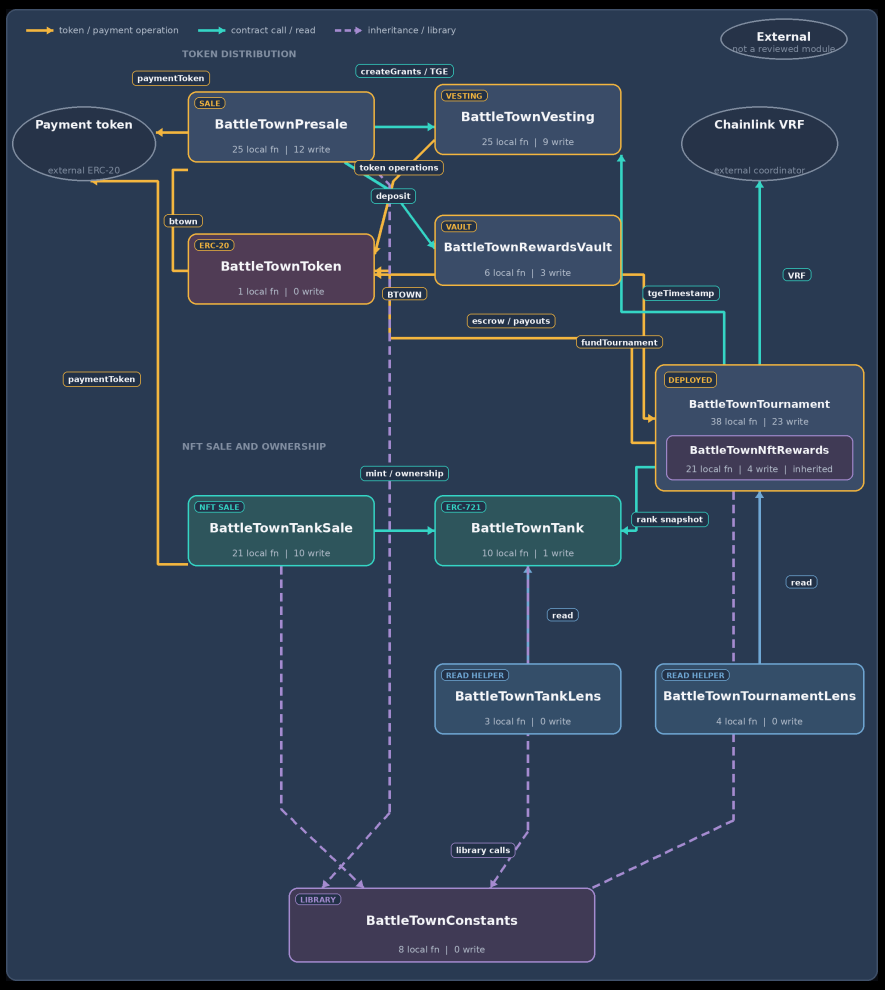

Figure 1. BattleTown contract relationship map. Compiler-derived source topology covering 11 reviewed modules; arrows run from caller to dependency.

Evidence boundary: Relationships are derived from the compiler-matched Solidity source and direct-call inventory. External addresses are contextual dependencies; this diagram does not assert deployed addresses, ownership, network configuration, or post-deployment wiring. OpenZeppelin base modules are omitted.

Contract-to-whitepaper correspondence

The source set corresponds structurally to the contract architecture described by the official whitepaper. Token, Tank, TankSale, Presale, Vesting, Tournament, RewardsVault and both lens contracts have the declared principal responsibilities. Constants is an internal library and NftRewards is inherited by Tournament.

No unrelated first-party production contract was identified. This establishes structural and declared-capability correspondence only; it is not blanket verification of every statement in the whitepaper or approval of the document.

Vulnerability analysis

Severity	Total
Critical	0
High	0
Medium	1
Low	4
Informational	2

NO CRITICAL OR HIGH-SEVERITY FINDINGS IDENTIFIED

In the reviewed source files and within the scope and methods described in this report.

Evidence and validation

Activity	Result	Evidence boundary
Source and reference hashing	Completed	Exact reviewed source and reference-document bytes identified
Production source reconciliation	15 of 15 hashes match	First-party production files and interfaces
Manual source review	Completed	All 11 production files and four interfaces
Structural whitepaper correspondence	Completed	Component and declared-capability mapping only
Normal compilation	Completed	solc 0.8.28+commit.7893614a; optimizer 200; viaIR; Cancun
Compiler-input verification	Completed	56 sources comprising 15 scope files, eight supporting repository sources and 33 imported dependency sources
Baseline tests	597 passing; zero failing; zero skipped or pending	Supplied suite on the in-process Hardhat network
Production-only coverage	99.76% statements; 97.38% branches; 99.38% functions; 99.82% lines	Coverage metadata for the 11 production files; not a security score
Instrumented supplied suite	596 passing; one infrastructure-only pending	The size/build-infrastructure test explicitly skips while coverage instrumentation is active
Audit-assisted instrumented suite	608 passing; one infrastructure-only pending	Supplied suite plus narrow audit-only coverage tests; production totals unchanged

Activity	Result	Evidence boundary
Technical reproductions	7 of 7 passing	Each pass confirms the behavior named by that reproduction
Positive technical controls	5 of 5 passing	Named access, accounting, claimant, VRF-gate and calendar assertions
Targeted closure	9 passing; zero failing	Tank rebinding, coordinator rotation, exact NFT accounting and deterministic allocation/supply boundaries
Echidna 2.3.3	Three campaigns; three fixed seeds each; 600 seconds per run; 20 main properties without discovered counterexamples	Bounded harness models and campaign duration; not proof of defect absence
Local resource bounds	22 tests and 38 measurements; N-1/N/N+1 and combined-dimen sion calls exercised	Local Hardhat gas, calldata, transaction and RPC limits only
Local Cancun/EIP-1153 path	One passing transient-guard execution after audit-harness calibration	Local backend support only; target-network support is a deployment prerequisite
Slither 0.11.6	Analysis completed; 164 raw entries classified	Detector labels were not adopted as audit severities
Scoped SCSVS/SCSTG map	Completed as an evidence map	Not a certification or claim of complete standard conformance

Coverage, static analysis and bounded fuzzing are supporting evidence. They do not establish an exhaustive absence of defects.

State-changing entry points

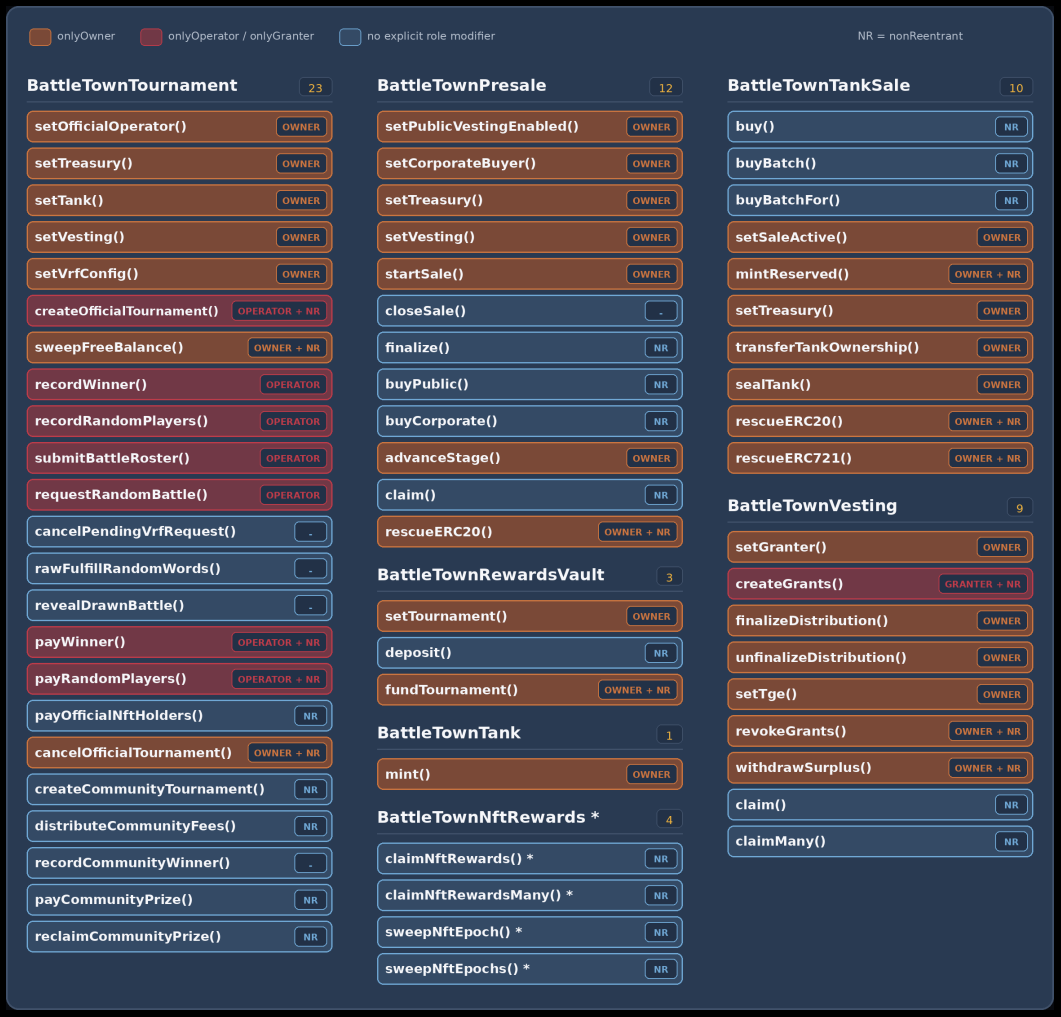

Figure 2. BattleTown state-changing entry points. 62 public/external function definitions, grouped by reviewed module and derived from the compiler-matched source.

Scope note: Constructors, view/pure functions, and inherited dependency ABI methods are excluded. * Four BattleTownNftRewards functions are inherited by BattleTownTournament. A missing explicit role modifier does not imply missing authorization: body-level checks are not summarized here.

BT-M01

Replacement commitment after a randomness-request timeout

Medium

Description

Locations: `BattleTownTournament.sol:614`, `:646`, `:690`, `:719`.

`submitBattleRoster` rejects changes while `pendingVrfRequest` is nonzero. After the one-day timeout, an owner or operator can cancel the request. Cancellation clears the pending marker and deletes the routing entries without making the draw terminal. The operator can then store a different roster commitment and issue another request for the same record.

FINTEH.ORG TECHNICAL RECOMMENDATION

Bind each draw's original outcome-relevant input to its first randomness
request and reject replacement of that input while the draw remains
recoverable.

Evidence and impact

An authorized owner or operator and an unfulfilled request that reaches its timeout are required. The demonstrated sequence does not establish advance knowledge of a VRF output, real-network transaction ordering or a profitable manipulation.

Source review and targeted EVM execution confirmed replacement of the committed roster, rejection of the earlier response and acceptance of a replacement response. Stateful calibration also reached the behavior.

Chainlink's VRF security guidance advises against cancellation or re-requesting for the same commitment and against changing outcome- relevant input after a randomness

request. The whitepaper describes a cancellation capability, so this issue is not classified as a contradiction of that document.

BT-L04 is a separate routing issue. Its recovery can reach this timeout path and expose the same replacement-commitment behavior, but BT-L04 alone does not demonstrate selection between two accepted VRF results.

FINTEH.ORG TECHNICAL RECOMMENDATION

Define a single recovery rule: cancellation must either make the draw terminal or leave the valid original response safely processable. Test timeout, callback and retry orderings and demonstrate that recovery cannot select between alternative committed inputs or accepted results.

BT-L01

Role revocation rejected after ownership acceptance

Low

Description

Locations: `BattleTownVesting.sol:152`, `BattleTownTournament.sol:336`, and inherited `Ownable2Step` ownership acceptance.

Both role setters reject `account == owner()` before considering whether the operation enables or disables the role. If an already enabled granter or operator becomes owner, that role remains enabled and a direct request to disable it is rejected.

The new owner cannot revoke the role while retaining ownership. Ownership must first be transferred elsewhere to remove it through the existing setter.

FINTEH.ORG TECHNICAL RECOMMENDATION

Permit the current owner to disable its existing granter or operator
role. If role separation is intended as a persistent invariant, reconcile
that role explicitly during ownership acceptance.

Evidence and impact

This is a role-management consistency issue rather than unauthorized acquisition of ownership. Whitepaper page 21 already discloses the role overlap.

Targeted reproductions exercised enablement, ownership nomination, acceptance and the rejected disable operation in both Vesting and Tournament. Verification should repeat that lifecycle while retaining rejection of unauthorized callers.

BT-L02

Reserve mint blocks first sale activation

Low

Description

Locations: `BattleTownTankSale.sol:227`, `:246`; `BattleTownTank.sol:26`.

`mintReserved` accepts an authorized reserve mint before the sale has ever been activated. The initial `setSaleActive(true)` requires `totalMinted == 0`. After an early reserve mint, that requirement cannot become true because the supplied Tank exposes no burn operation.

FINTEH.ORG TECHNICAL RECOMMENDATION

Reject reserve minting before first activation, or define an explicit
initial-activation rule that accounts for reserve supply while preserving
protection against unrelated pre-mints.

Evidence and impact

The first activation of that TankSale instance is unreachable after an accepted call sequence. An authorized reserve allocation and owner action are required; this is not an outsider-triggered denial of service. Whitepaper page 14 describes the relevant behavior.

Targeted EVM execution confirmed that a pre-activation reserve mint permanently prevents first activation of the same sale instance. Verification should show that the unsafe sequence is rejected at its first invalid step and that activation, deactivation, reserve minting and reactivation remain valid in the intended order.

BT-L03

New asynchronous work accepted after cancellation

Low

Description

Locations: `BattleTownTournament.sol:614`, `:646`, `:746`, `:853`, `:1045`.

`cancelOfficialTournament` marks the record cancelled and closes its payout flags. The record lookup only verifies existence. `submitBattleRoster` and `requestRandomBattle` do not check `cancelled` or `randomPaid`, so a cancelled record with no recorded random recipients can still accept a roster and a new request.

FINTEH.ORG TECHNICAL RECOMMENDATION

Apply active-state checks before accepting new roster input or issuing a new randomness request. Cancelled or financially terminal records should reject new asynchronous work.

Evidence and impact

A callback and reveal can write selection state on the cancelled record. Cancellation therefore does not close every entry point and asynchronous side effect associated with it. The payout flag remains closed, preventing payment of the random leg again. The local mock does not establish actual VRF subscription charges.

Targeted EVM execution and stateful calibration confirmed that a cancelled record can issue a request and later receive selection-state changes while its payout remains closed.

FINTEH.ORG TECHNICAL RECOMMENDATION

Define explicit handling for responses to requests issued before
cancellation. Tests should show that late callbacks cannot reopen
payouts, release accounting twice or create another request.

BT-L04

Cross-coordinator request ID reuse can overwrite live VRF routing

Low

Description

Locations: `BattleTownTournament.sol:133-140`, `:415-448`, `:646-739`.

`setVrfConfig` permits the owner to replace the coordinator while a request is pending. `vrfRequestTournament` and `vrfRequestCoordinator` are keyed only by numeric `requestId`. A request through the replacement coordinator writes those slots without rejecting an existing live route for the same numeric ID.

FINTEH.ORG TECHNICAL RECOMMENDATION

Namespace every VRF route by both coordinator address and request ID, and store that exact pair in the tournament's pending state. Reject any write that would overwrite another live route.

Evidence and impact

A deterministic coordinator-rotation test created a pending request through coordinator A, changed configuration to coordinator B and issued a request with the same numeric ID for another tournament. The later request overwrote both routing entries. Coordinator A's callback was rejected, coordinator B's callback was routed to the later tournament, and the earlier tournament remained pending until timeout.

The bundled Chainlink VRF v2.5 coordinator derives `requestId` from key hash, consumer, subscription ID and a coordinator-local nonce; the coordinator address is absent. Equal inputs and nonce values across coordinator instances can therefore reuse the same numeric ID. This is cross-coordinator ID reuse, not a cryptographic hash collision.

The condition requires an owner configuration change, a live earlier request, an authorized later request and reuse of the numeric ID. The standard Chainlink subscription-migration path rejects migration while its subscription has pending requests. The reproduced overlap therefore requires manual or custom coordinator reconfiguration rather than a successful standard subscription migration. No such configuration was established for a deployed instance.

The earlier callback becomes irrecoverable through the overwritten route and the tournament is delayed until timeout, cancellation and retry. Recovery can reach BT-M01, but this finding alone does not show a permissionless trigger, profitable manipulation or selection between two accepted results.

FINTEH.ORG TECHNICAL RECOMMENDATION

Prohibit coordinator replacement while affected routes remain live, or define an explicit migration policy. Cancellation must delete only the exact pending pair. Equal-ID tests must verify both callbacks and ensure that fulfillment or cancellation of one request cannot alter the other.

BT-I01

Preview aggregation and claim input have different semantics

Informational

Description

Locations: `BattleTownTournamentLens.sol:30`, `:48`; `BattleTownNftRewards.sol:255`, `:297`.

The lens sums every supplied entry, including duplicates, while the claim paths require strictly increasing token IDs. A duplicate-containing preview can therefore report a larger aggregate even though a claim using that list is rejected. View batch limits also exceed claim batch limits.

FINTEH.ORG TECHNICAL RECOMMENDATION

Document that previews aggregate every supplied entry, including duplicates, or align preview validation and batch limits with the claim interface.

Evidence and impact

This is an API-consistency observation. It does not demonstrate duplicate payment or an authorization bypass. A targeted reproduction confirmed both the inflated duplicate preview and rejection of the corresponding claim input.

BT-I02

tokensOfOwner does not explicitly enforce its documented order

Informational

Description

Location: `BattleTownTank.sol:64`.

NatSpec promises ascending token IDs, but the implementation copies `tokenOfOwnerByIndex` output without sorting it. The locked OpenZeppelin Contracts 5.6.1 implementation uses swap-and-pop on removal and explicitly permits enumeration order to change.

FINTEH.ORG TECHNICAL RECOMMENDATION

Correct NatSpec to describe enumeration order and the caller's sorting
requirement, or explicitly sort returned IDs if ascending order is part
of the intended API.

Evidence and impact

This is an API-compatibility observation rather than an ownership bypass. A targeted test minted IDs 1, 2 and 3 to one owner and transferred ID 1 away. `tokensOfOwner` returned `[3, 2]`. Passing that result directly to `claimNftRewards` reverted with `TokenIdsNotAscending`; sorting it to `[2, 3]` allowed the claim.

Operational funding boundary

Under a model in which the first official tournament is created at TGE and subsequent tournaments occur at the minimum 60-day interval for 3,650 days, 61 tournaments require `87,294,880 BTOWN`. The accumulated requirement first exceeds the initial `80,000,000 BTOWN` funding on day 3,120, reaching `80,135,840 BTOWN`. At a 90-day interval, 41 tournaments require `58,888,480 BTOWN`.

The implementation uses the Tournament contract's free balance and attempts to obtain any shortfall from the operator through `transferFrom`. If operator balance or allowance is insufficient, creation reverts atomically: the tournament, timestamp and escrow are not partially recorded, and creation can be retried after funding becomes available.

This is an operational funding boundary rather than a ranked vulnerability. The initial 80 million allocation does not by itself guarantee funding for every schedule permitted by the contract.

> The calculation assumes the stated intervals, a 3,650-day horizon, full scheduled payments, no additional funding and no reuse of cancelled amounts.

Documented technical capabilities

Component	Capability or restriction present in the source
Vesting	Owner can revoke grants before TGE and choose a nonzero recipient; grants do not carry a separate paid-sale or non-revocable flag
RewardsVault	Owner can rebind the destination to another address containing code and push tokens to the currently bound destination
Tournament	Owner can change Treasury, Tank, Vesting and VRF configuration and sweep calculated free balance
Presale	Purchase entry points take quantity without maximum-cost, delivery-mode or deadline arguments; current state controls execution
Tournament randomness	Manual random-player recording is available before a nonzero coordinator is configured
Tournament payouts	Recorded player payouts require the configured operator or another explicitly authorized role; no permissionless claim requirement is inferred
Official calendar	A nonzero future TGE permits period-zero official creation under the reviewed implementation
NFT rewards	Current NFT ownership controls claiming; each epoch retains its original Tank address and eligibility cutoff
Source set	No project-specific pause or proxy-upgrade mechanism is implemented in first-party contracts

These statements describe source-level capabilities and do not identify which accounts or contracts hold those powers in a deployed project.

Technical controls and execution results

Area	Validated result	Practical limit
Access control	Baseline tests and controls exercised owner, operator, granter and coordinator gates; BT-L01 records the ownership-transfer role edge	No claim about custody or conduct of real privileged accounts
Reentrancy	Baseline probes passed; local Cancun execution rejected a nested call through OpenZeppelin ReentrancyGuard Transient while preserving accounting	Only exercised paths, reviewed source and local backends are covered
State and accounting	Presale liabilities, one-time claims, NFT claimant changes, exact per-token/epoch payments and zero-surplus backing were exercised	No live balances or deployed configuration were reviewed
Duplicate execution	Repeat claims and payouts were exercised; BT-I01 is limited to preview semantics and does not permit duplicate payment	UI behavior outside the contracts is out of scope
Arithmetic	Checked operations, source bounds and calendar calculations were reviewed; 80 anniversary cases matched an independent oracle	No formal arithmetic proof is claimed
NFT behavior	Rank and supply N-1/N/N+1 boundaries, pinned Tank A/B epochs and exact ERC721Enumerable ordering were exercised	Large-cap setup used deterministic local storage checkpoints before production getter and mint-path assertions

Area	Validated result	Practical limit
Randomness lifecycle	Coordinator authentication, pending locking and ordinary coordinator rotation passed; BT-M01, BT-L03 and BT-L04 record retained lifecycle gaps	No live coordinator timing, subscription state or profitable manipulation was demonstrated
Resource bounds	One-dimensional maxima were measured; 4x256 NFT claims estimated successfully, 5x256 reached the first local estimator cap, and full 32x256/64x512 combinations exhausted local gas	Local limits are not production-network compatibility or fee evidence
Bytecode size	All size checks passed; largest production runtime was 17,806 bytes against the 24,576-byte EIP-170 limit	Applies to the reviewed compiler profile
Low-level constructs	No delegatecall, selfdestruct, tx.origin authorization or inline assembly appears in first-party production source	Imported libraries contain separately reviewed low-level utilities
Static analysis	All 164 Slither entries were classified; no separate retained finding resulted	Detector output is supporting evidence, not proof of defect absence

Deployment compatibility prerequisite

The reviewed contracts target the Cancun EVM. BattleTownVesting, BattleTownPresale, BattleTownRewardsVault and BattleTownTankSale use the EIP-1153 transient-storage path through the locked OpenZeppelin dependency.

Local Hardhat and Echidna backends compiled and executed that path. A nested reentrancy attempt was rejected while the outer transaction and accounting remained correct. These results demonstrate local tool support only.

The intended deployment network must independently support Cancun and EIP-1153. Target-network support, deployed bytecode and live configuration were not supplied or verified.

Compiler advisory screening

Advisory	Complete-input screening
SOL-2026-6	No `require` call exists in the 56 compiler-input ASTs; the named custom-error trigger is absent
SOL-2026-5	The reviewed build uses `viaIR: true`; the advisory applies to the legacy pipeline
SOL-2026-4 and SOL-2026-2	No source-level cycle among directly referenced internal functions or modifiers was identified
SOL-2026-3	Introduced in compiler 0.8.29, after the pinned 0.8.28 version
SOL-2026-1	No high-level transient variable declaration or delete operation exists; the imported guard uses explicit `tstore` operations
SOL-2025-1	No custom storage-layout specifier was identified

Trigger screening and successful execution do not prove compiler correctness.

References

1 Chainlink VRF v2.5 security considerations, consulted September 2026.

2 Solidity list of known bugs, consulted September 2026.

3 EIP-170: Contract code size limit.

4 OWASP Smart Contract Security Verification Standard and Smart Contract Security Testing Guide 0.0.1, used as a scoped evidence map.

5 BATTLE TOWN Whitepaper v3.3 (original audit reference), SHA-256: 84e27cb5d07a4fc497a7bcb6752bbac5b3726109a2be1af8fff78e66f6220175.

6 Current public edition: BATTLE TOWN Whitepaper v3.4.

Scope and limitations

This report describes a limited technical review of the identified source-code snapshot. It is not a guarantee of security, certification, deployment approval, endorsement, investment recommendation or assessment of any deployed instance.

Findings and conclusions depend on the supplied sources, stated assumptions and methods performed. Production configuration, privileged-account conduct, key security, deployed bytecode, live external integrations, target-network behavior and subsequent modifications were not assessed unless expressly stated.

Conclusion

The reviewed source snapshot was assessed for access control, state transitions, accounting, repeated execution, external-call and reentrancy behavior, randomness handling, resource bounds and compiler and deployment prerequisites. The work included exact compilation, 597 supplied baseline tests, targeted EVM execution, production-only coverage, classified static analysis and bounded stateful campaigns.

Within that source-only scope and the methods described in this report, no Critical or High-severity findings were identified. The executed controls behaved as expected in the tested scenarios, subject to the one Medium, four Low and two Informational items described above.

Subject to those findings and stated limitations, the reviewed snapshot is technically consistent with the tested controls and ready for the separate publication workflow. No conclusion is made about deployed configuration, target-network support or future source revisions.