BVCC Wallet

BLOCK VENTURE CHAIN CAPITAL

# Agent Wallet Security & Test Report

Static analysis, vulnerability remediation & on-chain agent-permission test battery

| | |
|---|---|
| SUBJECT | BVCC Agent Wallet (ERC-4337 / ERC-7579 smart account) |
| NETWORK | Arbitrum Sepolia (421614) · V2 factories also on Arbitrum One (42161) & BNB Chain (56) |
| PATCHED WALLET | `0x1ED261BA084c4E6117daf289f3ca233BBA3cF0C3` |
| ENGAGEMENT | Internal security testing & remediation verification |
| DATES | 2026-06-06 → 2026-06-11 |
| STATUS | 2 HIGH found · Fixed & redeployed · 11/11 tests pass · Recovery verified · V2 multichain |

## 1 Executive Summary

This report documents the security testing of the BVCC Agent Wallet — an ERC-4337 smart account that lets an owner delegate bounded spending authority to an autonomous agent (EOA). Work covered static analysis, a high-severity finding with its remediation, a redeploy of the patched contracts, and a full re-verification of the agent permission system on the patched bytecode. A follow-up V2 round fixed a mainnet gas-griefing issue in the fee-snapshot logic and concluded with a deterministic multichain deployment (Arbitrum One, BNB Chain, Arbitrum Sepolia — see §9).

| 2 | 131/131 | 11/11 | 0 |
|---|---|---|---|
| HIGH FINDINGS (FIXED) | UNIT TESTS (FORGE) | ON-CHAIN AGENT TESTS | OPEN ISSUES |

### Key outcomes

- **High-severity vulnerability identified and fixed.** An `approve` cap-bypass let an agent move tokens past its daily/total budgets via `transferFrom`. Reproduced on-chain, patched in source, and re-verified after redeploy.
- **Mainnet gas-griefing issue fixed (V2).** Arbitrum precompiles matched by the fee-snapshot calldata scan burned all forwarded gas, inflating gas estimates to ~32M and breaking swaps. Every `balanceOf` probe is now gas-capped (`PROBE_GAS_CAP = 100k`); estimates return to ~400k for a real swap (see §9).
- **Static analysis clean.** Slither 0.11.3 produced no actionable findings; a defensive zero-address check was added to both factory constructors.
- **Patched contracts redeployed** to Arbitrum Sepolia with the deployer key holding no privileges (control retained by a separate admin address).
- **Full agent-permission battery passes** on the patched wallet: per-transaction, daily, total-budget, period-budget (with auto-rollover), expiry, pause, and recipient allow-listing — enforced for both native ETH and ERC-20 tokens.

> **Overall assessment.** The agent permission model is sound and, after remediation, enforces every configured constraint. No open issues remain at the time of writing.

## 2 Scope & Methodology

### In scope

- `BVCCAgentWallet.sol` — agent permission logic (`executeAsAgent`, per-token & ETH caps, pause, expiry).
- `BVCCWallet.sol` — base smart account (execution, fees, guardian recovery).
- `BVCCAgentWalletFactory.sol` / `BVCCWalletFactory.sol` — CREATE2 factories.

### Methodology

- **Static analysis** — Slither 0.11.3 over both source repositories.
- **Unit testing** — Foundry (`forge test`), including regression tests for the finding.

- **On-chain testing** — the agent EOA drives `executeAsAgent` directly; reverts are confirmed by simulation (`eth_call`) against live state, valid cases as real transactions.

### Agent execution model

The agent calls `executeAsAgent(bytes32 mode, bytes executionData)` as an ordinary EOA transaction (not a UserOp). `mode` is the ERC-7579 batch selector (`0x0100…00`); `executionData` is an ABI-encoded `Execution[]` of `(target, value, callData)`. Validation order inside the call is **per-tx** → **recipient/protocol** → **period** → **daily** → **total**, with state updated **before** external execution (checks-effects-interactions) and a `nonReentrant` guard.

## 3 Static Analysis — Slither

Slither 0.11.3 was run on both repositories prior to redeploy. It reported 39 results across 59 contracts; **none were actionable**. The table summarizes the categories and their disposition.

| CATEGORY | EXAMPLES | DISPOSITION |
|---|---|---|
| By-design patterns | `block.timestamp` comparisons, inline assembly, low-level calls, calls-in-loop (batch) | Accepted |
| False positives | "uninitialized" mapping; reentrancy in `createWallet` (only an `emit` after trusted `setGuardians`) | No action |
| Dead code — `_feeNumerator()` | Flagged as unused | False positive — verified in use (see note) |
| Missing zero-check — factory `owner_` | Constructor sets owner without validation | Fixed |
| Cosmetic | Naming, literal digits, cyclomatic complexity | Accepted |

> **Verification note — tool findings are not facts.** Slither marked `_feeNumerator()` as removable dead code. A source check showed it **is** used to compute the protocol fee (base 0.05% / agent override 0.15%); removing it would have broken fee logic. The finding was a false positive caused by virtual-function dispatch and was **not** acted upon.

### Remediation applied

A defensive zero-address check — `if (owner_ == address(0)) revert ZeroOwner();` — was added to both factory constructors, synchronized across both repositories. Full suite green: **forge test = 128/128** in each repo. Slither was re-run on the V2 contracts after the gas hardening (§9): **0 results**; the unit suite now stands at **131/131**.

## 4 Finding — Token cap bypass via `approve`

HIGH FIXED & REDEPLOYED

### Description

An agent's per-token **daily** and **total** budgets could be bypassed entirely. The agent validation accounted spend for `transfer` but charged `approve` only against the per-transaction cap — not the daily or total budgets, and without restricting the spender. An agent could therefore `approve` an allowance (only per-tx-capped) and drain it with an external `transferFrom`, never touching the cumulative counters.

### Proof of concept (executed on-chain, budgets exhausted at daily 2/2 · total 2/3)

| STEP | ACTION | RESULT |
|---|---|---|
| 1 | `executeAsAgent → USDC.approve(B, 1)` | Passes — per-tx OK, daily ignored |

| | | |
|---|---|---|
| 2 | As EOA: `USDC.transferFrom(wallet, B, 1)` (outside `executeAsAgent`) | 1 USDC extracted; counters unchanged |

A plain `transfer` of the same amount correctly reverted `TokenDailyLimitExceeded`; `approve` did not. Because `approve` had no daily limit, the `approve(1) → transferFrom(1)` cycle was repeatable — draining the wallet past both the daily (2) and total (3) caps. The PoC funds were returned to the wallet.

### Remediation

In the `executeAsAgent` accumulation loop the condition was extended with `|| _isApprove(batch[i].callData)` so that `_applyTokenSpend` charges `approve` against the same daily/total caps and updates the counters. The choice is deliberately conservative (an `approve` overwrites the allowance, so it may over-count — the safe direction). Six regression tests were added; **5 fail without the fix, all pass with it.**

> **Bounded surface.** Tokens accept only `transfer` and `approve` by selector; `transferFrom` and `increaseAllowance` are blocked. ETH has no delegation primitive, so it is unaffected. `approve` was the sole vector — now closed and re-verified post-redeploy (see §6, test #4 context).

## 5 Remediation Deployment

Both factories changed bytecode (zero-check), producing new deterministic CREATE2 addresses on Arbitrum Sepolia. The frontend network configuration was updated accordingly.

| CONTRACT | NEW ADDRESS (PATCHED) | PREVIOUS (DEPRECATED) |
|---|---|---|
| AgentWalletFactory | `0xc87aa10747A92B472EF6B36e190B84c897a2953e` | `0xe7ad…7e04` |
| SmartWalletFactory | `0xa5290A51a73903176e09C864E1542a07da67BD12` | `0x6890…F58d` |

### Privilege separation

- **Deployer (no control):** `0xA3e06B6443D911915bb4eD157832d43C25D6617D` — the factory's `owner` is set from a constructor argument, not `msg.sender`, so the deployer holds no privileges.
- **Admin / kill-switch owner:** `0x3145Bd5e2489d8bDdAb17F23F26F07Ac5aD55A4c` — immutable; the only privileged action is the one-way `kill()`.

Runtime sizes within the EIP-170 limit (AgentWallet 21,169 B; AgentWalletFactory 23,803 B — 773 B of headroom).

## 6 Agent Permission Test Battery

Executed against the patched wallet `0x1ED261…F0C3` with the agent EOA `0x3852…4aB4`. Reverts confirmed by simulation; valid cases as real transactions. Allowed recipients were the two recovery guardians (bot-controlled, funds recoverable).

| # | TEST | EXPECTED | RESULT |
|---|---|---|---|
| 1 | ETH per-tx (0.0002 > max 0.0001) | revert | ExceedsPerTxLimit |
| 2 | ETH within limit (0.0001) | pass | Success |
| 3 | USDC per-tx (2 > max 1) | revert | ExceedsTokenMaxAmount |
| 4 | USDC daily (1+1+1, cap 2) — real txs | 3rd reverts | TokenDailyLimitExceeded |
| 5 | Token send to non-allow-listed dest | revert | RecipientNotAllowed |
| 6 | ETH send to non-allow-listed dest | revert | RecipientNotAllowed |
| 7 | Total budget (0.0002+0.0001 = 0.0003) — real txs | next reverts | AgentBudgetExceeded |
| 8 | Period budget (0.0003+0.0002 = 0.0005) — real txs | next reverts | PeriodBudgetExceeded |
| 9 | Period auto-rollover (periodStart 0) | window resets on 1st spend | Verified |
| 10 | Paused (owner pauses) | agent blocked | EnforcedPause |
| 11 | Expiry (lapsed naturally) | agent blocked | AgentPermissionsExpired |

> **Observation.** The recipient allow-list applies to **token transfers** as well as ETH — a token send to a non-allow-listed address reverts `RecipientNotAllowed`. Re-authorization via "Edit" does not reset the daily counter (keyed by UTC day); with `periodStart = 0` the first spend rolls the period window.

### Privilege-escalation resistance (from the 06-06 session)

An agent targeting the wallet itself to call owner functions (`pauseAgents`, `revokeAgent`, `authorizeAgent`) reverts `AgentCannotCallWallet`; arbitrary contract calls outside the protocol allow-list are blocked. The agent can move only ETH and allow-listed tokens to allow-listed recipients.

## 7 Supporting Frontend Fixes

Three issues in the wallet web application were fixed and synchronized to the public repository.

| AREA | ISSUE | FIX |
|---|---|---|
| `lib/wallet.ts` | Credential recovery queried only the base factory event, so re-opening an *agent* wallet by address failed (“no active wallet”). | Also query `agentFactory` / `AgentWalletCreated`. |
| Agents page | Per-token limits shown as text only. | Read `getTokenSpent` on-chain; render Daily/Total progress bars (verified live). |
| Deploy flow | App-supplied gas under-estimated the ~21 KB agent-wallet deploy on Arbitrum, causing signing failures. | Defer to the wallet’s network estimate. |

## 8 Guardian Recovery — Executed & Verified

The base wallet supports social recovery: a 2-of-3 guardian quorum can rotate the owner passkey after a timelock. This was exercised end-to-end on wallet `0xEBc851bA9e8Ce32A2cb8eB99544F5F024f06E61d` and is now **complete**.

### Process

- **Initiation (2026-06-06).** Recovery proposed toward a new owner passkey; guardians 1 and 2 (`0xa337…5908`, `0x8e06…2CcC`) approved — quorum 2/2.
- **Timelock.** On-chain `recoveryReadyAt` = 2026-06-08 17:24 UTC; throughout the window the owner retained the ability to cancel.
- **Execution (2026-06-08).** After the timelock elapsed, `executeRecovery()` was called from the wallet `/recover` UI. The on-chain `signer` rotated to the new passkey (X `0x303591…` / Y `0xb858ed…`), replacing the previous owner (X `0x070f7a…`).
- **Positive check.** The new owner immediately signed a real **19 USDC** transfer with the new Face ID — confirmed on-chain.

### Negative test — the old passkey is neutralized

To confirm the rotation actually revoked the previous key, a transfer was attempted with the **old** passkey, whose WebAuthn credential is still present in the test authenticator. The signature is produced locally but must be validated on-chain against the current owner.

| # | ACTION | EXPECTED | RESULT |
|---|---|---|---|
| 1 | Send 0.0001 ETH, UserOp signed by the OLD passkey | rejected at validation | FailedOp(0, AA24 signature error) |
| 2 | On-chain effect (ETH / USDC balances & nonce) | no change | Unchanged |

**Result.** Recovery completed successfully: ownership moved to the new passkey, the new key operates the wallet, and the old key is cryptographically rejected by the EntryPoint during signature validation — it can no longer move funds even though its local credential still exists.

## 9 V2 — Mainnet Gas Hardening & Multichain Deployment

HIGH (availability / overpayment) FIXED — V2 DEPLOYED 2026-06-11

### Description

On Arbitrum mainnet, swaps through the wallet failed at gas estimation. The Case-3 fee logic scans calldata for token candidates and probes each one with `balanceOf`. Arbitrum system precompiles (`0x64–0x6f`) report bytecode (`0xfe`), so the code-length guard did not filter them — and calling them consumes **all forwarded gas** instead of reverting cheaply. The estimator kept raising the gas limit until ~32M so the probe would "fit"; that inflated limit is what the wallet would have prefunded (a 64× overpayment risk), and in practice the operation failed.

| | BEFORE (BUG) | AFTER (V2 FIX) |
|---|---|---|
| Frontend / bundler gas estimate | ~32,000,000 | ~400,000 |
| Gas-burning candidate effect | burns all forwarded gas | capped at 100k, execution continues |
| Outcome | swap fails (or 64× overpayment) | normal swap, ~$0.02–0.05 on Arbitrum |

### Remediation (V2)

- **Gas-capped probes.** A shared constant `PROBE_GAS_CAP = 100_000` caps every `balanceOf` staticcall in both `_tryBalanceOf` (snapshot) and `_collectFeesOnIncrease` (fee collection). Worst case across a full scan is bounded to ~2.2M gas vs 32M before; real ERC-20s need well under the cap, so fee accounting is unaffected.
- **Fee collection can never break a user call.** If a token misbehaves after the swap, its fee is skipped instead of reverting the user's transaction.
- **Explicit accounting bound.** `executeAsAgent` now requires `totalEth ≤ type(uint128).max` before casting, removing a silent-truncation edge case for agents configured with no ETH limits.

### Verification

- **forge test = 131/131** — regression tests added: a gas-burning candidate does not starve the swap; a reverting `balanceOf` is filtered non-fatally; a fat-return token is still snapshotted.
- **Slither 0.11.3 re-run on V2: 0 results.** (Tool note: Slither cannot generate IR for OpenZeppelin's transient-storage `ReentrancyGuard` internals — a known parser limitation, not a finding.)
- **EIP-170 sizes:** AgentWallet 21,214 B; AgentWalletFactory 23,848 B (728 B headroom); SmartWallet 13,432 B; SmartWalletFactory 16,030 B.

### V2 deployment — deterministic CREATE2, same address on every chain

| CONTRACT | ADDRESS (IDENTICAL ON ALL NETWORKS) |
|---|---|
| SmartWalletFactory V2 | `0x230b7010529AB6977Dd8581B3eF018ef865BdEf1` |
| AgentWalletFactory V2 | `0x8D9e24022777173AD6336e00884b6C87c7EF054c` |

Deployed 2026-06-11 to Arbitrum One (42161), BNB Chain (56) and Arbitrum Sepolia (421614). Wallets created by pre-V2 factories do not receive the fix — new wallets must be created through the V2 factories.

## 10 Conclusion

The BVCC Agent Wallet enforces a comprehensive, layered permission model. One high-severity bypass was identified, reproduced on-chain, remediated, redeployed, and re-verified. Static analysis surfaced no further actionable issues. The complete on-chain test battery passes on the patched bytecode, and no issues remain open. Guardian recovery was additionally executed and verified end-to-end — ownership rotation to a new passkey plus on-chain rejection of the old key (see §8). A follow-up V2 round fixed a mainnet gas-griefing issue in the fee snapshot, was verified by tests and a clean Slither re-run, and concluded with the deterministic deployment of the V2 factories to Arbitrum One, BNB Chain and Arbitrum Sepolia (see §9).

### Appendix — key identifiers

| ITEM | VALUE |
|---|---|
| Network | Arbitrum Sepolia (421614) |
| Patched wallet | `0x1ED261BA084c4E6117daf289f3ca233BBA3cF0C3` |
| Agent (role B) | `0x38529C66F3cf22453D66B9E2A20FdF2676544aB4` |
| AgentWalletFactory (Sepolia, pre-V2) | `0xc87aa10747A92B472EF6B36e190B84c897a2953e` |
| SmartWalletFactory (Sepolia, pre-V2) | `0xa5290A51a73903176e09C864E1542a07da67BD12` |
| SmartWalletFactory V2 (all networks) | `0x230b7010529AB6977Dd8581B3eF018ef865BdEf1` |
| AgentWalletFactory V2 (all networks) | `0x8D9e24022777173AD6336e00884b6C87c7EF054c` |
| V2 networks | Arbitrum One (42161) · BNB Chain (56) · Arbitrum Sepolia (421614) |
| Tooling | Slither 0.11.3 · Foundry (forge 1.5.1) · viem 2.x |