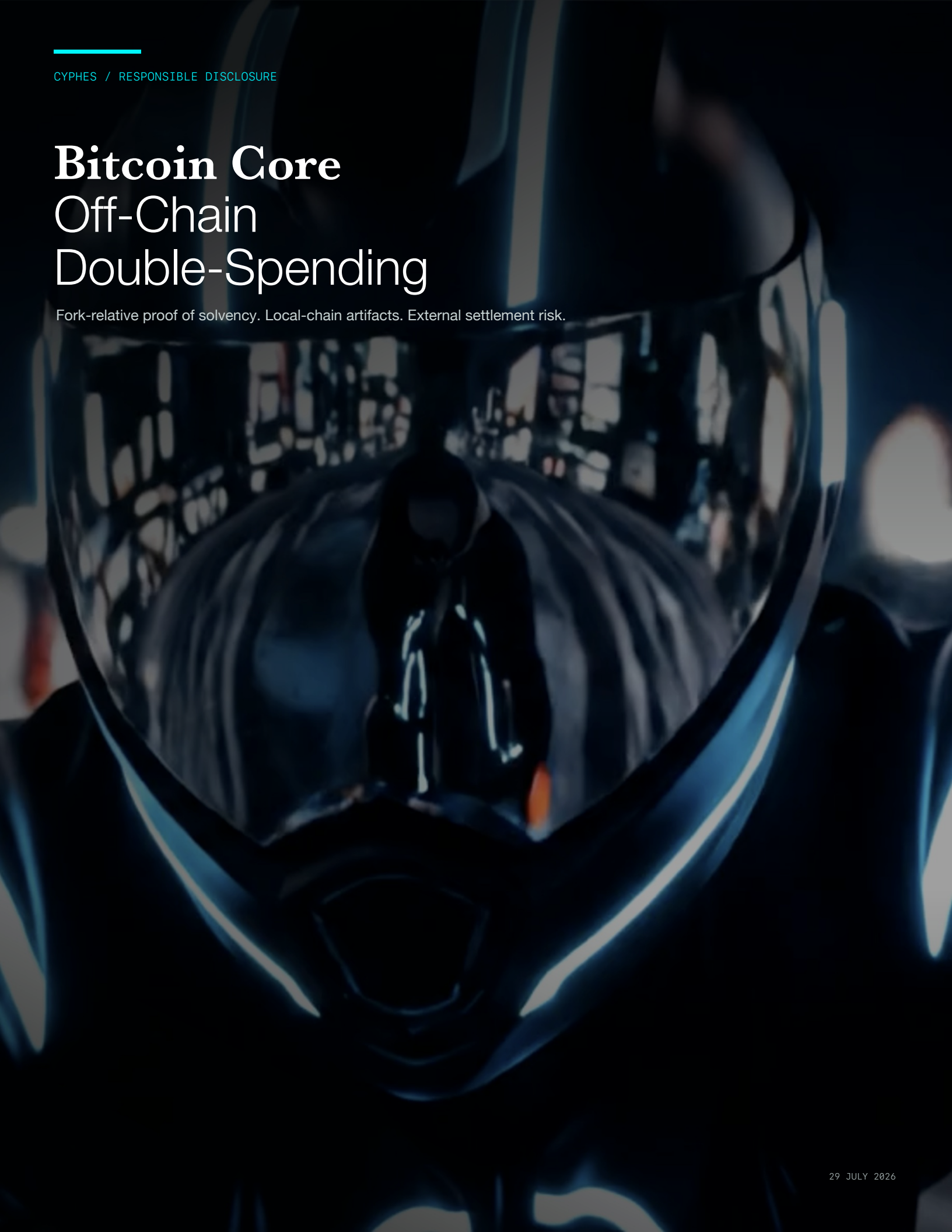
CYPHES / RESPONSIBLE DISCLOSURE

Bitcoin Core Off-Chain Double-Spending

Fork-relative proof of solvency. Local-chain artifacts. External settlement risk.



29 JULY 2026

SECTION / 00

Executive View

The issue is not that Bitcoin double-spends on-chain. The issue is that local-chain observations can be repackaged as proof of global solvency.

BOTTOM LINE

Medium-confidence API and UX hardening finding.

Bitcoin Core v31.1 can report a trusted 10 BTC balance, sign a fresh challenge, fully sign a transaction, accept that transaction to the local mempool, and return a non-null UTXO for the same outpoint while a better-work chain has already spent and buried the output. Every artifact is true relative to the local stale branch and false relative to economic reality.

What this is

- A fork-relative proof-of-solvency failure in wallet/RPC/API semantics.
- An off-chain double-spending primitive when external systems release value from stale local-chain evidence.
- A design and documentation gap around aggregate balance, signmessage, testmempoolaccept and gettxout provenance.

What this is not

- Not a Bitcoin consensus break.
- Not inflation.
- Not a valid network double spend.
- Not proof that a well-peered verifier node is unsafe.

CRITICAL	HIGH	MEDIUM	LOW
0	0	1	3
no consensus break	no direct theft in Core	API/UX composition risk	docs and hardening

CYPHES assessment: disclose privately first. Expect maintainers to classify the core issue as hardening, API documentation, or UX risk rather than a critical vulnerability. The useful outcome is a safer integration path.

EVIDENCE SNAPSHOT

The stale node looks healthy by the usual signals.

Check	Stale node result	Why it matters
initialblockdownload	false	Node does not present as bootstrapping.
verificationprogress	1	Progress-based sync heuristic passes.
blocks == headers	105 == 105	Node has no known better header chain.
tip age	seconds	Preserves local tip-age heuristics and GUI sync optics.
networkactive	true	The node is not offline.
connections	1	A live attacker peer keeps the stale branch moving.
wallet tip == node tip	height/hash match	The exposed provenance comparison still passes.

Artifact divergence

Artifact	Stale node	Real chain
getbalance	10.00000000 BTC	0.00000000 BTC
signmessage(nonce)	verifies	also verifies
signrawtransactionwithwallet	complete: true	unspendable
testmempoolaccept	allowed: true	missing-inputs
gettxout(outpoint)	unspent 10 BTC	null

SECTION / 01

CYPHES Nodes

Multiple review nodes converge on the same conclusion: the artifacts are locally valid, but local validity is being confused with global solvency.

01 NODE/01 PRIMARY EVIDENCE

Source and API surface

The wallet RPC surface exposes aggregate balance as a scalar and relies on local-chain synchronization. It does not attach a most-work-chain provenance guarantee.

CONTRIBUTION

- Mapped getbalance, getbalances, getwalletinfo, signmessage, gettxout and GUI sync language.
- Confirmed getbalance returns bare value while getbalances/gettxout carry block provenance fields.
- Confirmed lastprocessedblock is provenance-only and signmessage documentation omits solvency limits.

SOURCE

Bitcoin Core tag v31.1, local binary v31.1.0, wallet RPCs, GUI copy, shipped RPC help text, and public security policy.

CALIBRATION

High confidence in API behavior. The controversial part is not whether the RPCs behave this way; it is whether the behavior should be classified as a security vulnerability or hardening issue.

02 NODE/02 VALIDATED POC

Proof of concept

The PoC is meaningful because the victim node is not offline. It stays network-active and receives fresh lower-work blocks from an attacker-controlled peer.

CONTRIBUTION

- Three-node regtest topology reproduces fork-relative balance and UTXO divergence.
- Re-run reproduces all artifacts: balance, signature, signed spend, mempool acceptance and gettxout divergence.
- Augmented probes show minconf can be restored by mining more stale-branch confirmations.

SOURCE

poc_fork_relative_solvency.sh, run_output.txt, run_output_rerun.txt, augmented_probes.sh and augmented_probes_output.txt.

CALIBRATION

Under eclipse the RPC artifact bundle is free. On mainnet, hashrate is only needed to preserve freshness optics or deepen stale-branch confirmations. That decomposition bounds severity without weakening the finding.

03

NODE/03 ECONOMIC ARBITRATION

Integration risk

The downstream failure mode is irreversible external value released from local-chain observations. The unsafe integration path is natural because Core answers plainly and challenge-response feels stronger than it is.

CONTRIBUTION

- Separated key control from current UTXO ownership.
- Separated signed-spend authorization from most-work-chain spendability.
- Separated instant proof-of-control from collateral encumbrance over time.

SOURCE

Bridge, lending, exchange-credit, reserve-attestation and proof-of-funds compositions.

CALIBRATION

Strongest impact is off-chain or cross-chain systems that use a claimant node or a poorly peered observer. A verifier's own well-peered node largely defeats the scenario.

04

NODE/04 NETWORK VERDICT

Final arbitration

Report as medium-severity hardening. Do not overclaim a protocol double spend. Ask for provenance, warnings and safer API patterns.

CONTRIBUTION

- Private disclosure first to security@bitcoincore.org.
- Primary fix path: documentation, GUI wording and a new provenance-rich balance RPC.
- Public framing after maintainer clearance: off-chain double-spending, not Bitcoin consensus failure.

SOURCE

Collective synthesis over code, PoC output, augmented probes and disclosure policy.

CALIBRATION

This is exactly the kind of finding where professional tone matters. The report should be useful even if Bitcoin Core classifies it as non-vulnerability design hardening.

SECTION / 02

Technical Analysis

The code-level basis is simple: wallet synchronization is local-chain synchronization, and that distinction disappears in common proof-of-funds workflows.

Finding

Fork-relative solvency artifacts can be economically false.

Bitcoin Core can produce internally valid balance, key-control, signed-spend, mempool-acceptance and UTXO-existence artifacts while the backing output has already been spent on the most-work chain. The failure is not Bitcoin consensus. It is the absence of provenance and warnings on local-chain wallet/RPC observations consumed by external settlement systems.

Field	Assessment
Title	Off-chain double-spending through fork-relative proof of solvency
Target	Bitcoin Core v31.1 / binary v31.1.0
Severity	Medium for Bitcoin Core API/UX hardening; high for unsafe downstream settlement systems
Status	Validated in regtest PoC; not a consensus break, not inflation, not a valid network double spend
Core invariant	Wallet and RPC answers are synchronized to the local active chain, not independently proven against the most-work chain
Primary impact	External systems may release irreversible off-chain or cross-chain value after accepting local-chain balance, signature, signed-spend, or UTXO artifacts

Severity rationale

- No BTC is created; no consensus invariant is violated.
- Loss requires unsafe composition above Bitcoin Core: bridge minting, lending, proof-of-funds, exchange crediting, reserve attestation or payout automation.
- The PoC demonstrates that common freshness and challenge-response checks pass on the stale view.
- Bitcoin Core's own security policy allows an issue to be serious without ultimately being classified as a vulnerability; this report is framed accordingly.

CODE BASIS

getbalance returns a bare scalar.

At tag v31.1, getbalance obtains the wallet, waits for it to synchronize to the node's current chain, computes balance, and returns a single amount. The result carries no scan status, block hash, header height, chainwork, peer topology, network activity, or tip-age context.

```
GetWalletForJSONRPCRequest(request)
pwallet->BlockUntilSyncedToCurrentChain()
LOCK(pwallet->cs_wallet)
GetBalance(*pwallet, min_depth, avoid_reuse)
return ValueFromAmount(bal.m_mine_trusted)
```

Local synchronization is not global synchronization.

BlockUntilSyncedToCurrentChain guarantees the wallet has caught up with chain().Tip() for this node. On a stale branch fed by an attacker peer, that invariant still holds. The wallet is synchronized to the stale branch, not to Bitcoin as a whole.

```
uint256 last_block_hash = WITH_LOCK(cs_wallet, return m_last_block_processed);
chain().waitForNotificationsIfTipChanged(last_block_hash);
```

getbalances, getwalletinfo and gettxout are better because they attach block provenance. That provenance is still local-node provenance. An integrator naturally compares wallet_lastprocessedblock to the node tip and stops there, because Core exposes the fields but does not explain what would make a balance trustworthy for external settlement.

GUI AND SIGNATURES

The user-facing language presents the stale number as current.

Bitcoin-Qt labels the wallet amount as Available and describes it as the current spendable balance. The sync presentation can show up-to-date when the local tip is fresh enough. That freshness decision is local-tip-only, so a lower-work branch can satisfy the icon while every wallet and RPC artifact remains branch-relative.

Surface	Observed wording or behavior	Risk
GUI balance	Available / current spendable balance	A stale local value is presented as current.
Sync icon	Up to date when local tip age is below threshold	Fresh lower-work branch can satisfy the indicator.
signmessage	Description covers key signing behavior only	Verifier may mistake key control for solvency.
gettxout	Returns UTXO plus bestblock	Good provenance seed, but still local to one node.

Crucially, none of getbalance, getbalances, gettxout, testmempoolaccept or signmessage consults tip age. Under eclipse the artifact bundle is therefore free; hashrate is only needed to satisfy the GUI icon or a verifier-authored freshness rule.

Challenge-response does not close the gap.

Property	Established by	PoC result
Key possession	signmessage over fresh nonce	Yes, genuinely.
Authorization to spend outpoint	signrawtransactionwithwallet	Yes, locally.
Outpoint unspent on most-work chain	Requires independent verifier view	No.
Coins locked through loan term	Script, multisig or custody	No.

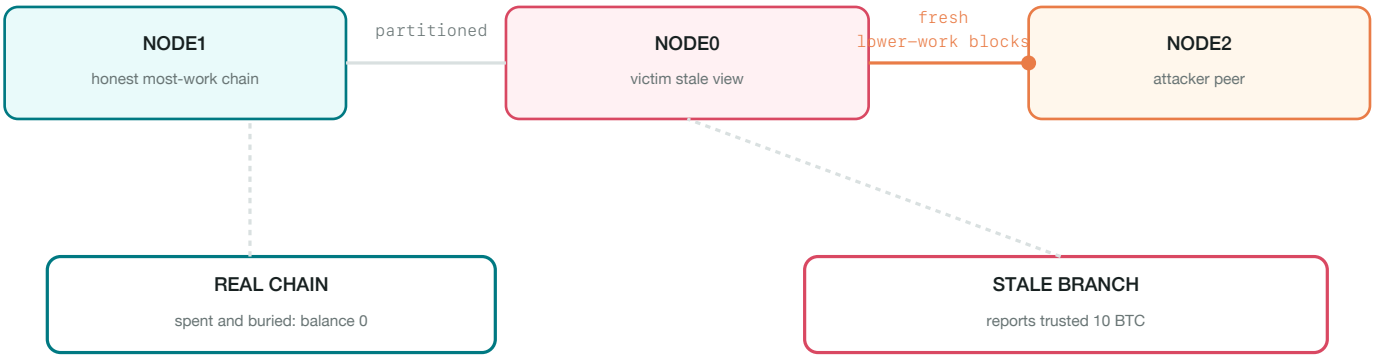
SECTION / 03

Proof Of Concept

A local regtest fork models the stale-observer condition without hiding behind an offline-node demonstration.

TOPOLOGY

Victim view remains online, connected and fresh.



Node1 mines the real most-work chain where the 10 BTC collateral is spent and buried. Node0 is partitioned from node1 but remains connected to node2, which mines fresh blocks on a lower-work branch. Node0 reports itself active, out of IBD, fresh, connected, and internally synchronized.

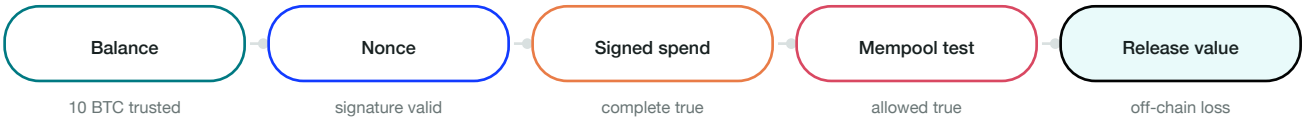
The critical design choice: node0 is never taken offline.

An offline-node demo would be easy to dismiss. This PoC keeps networkactive true and keeps a live peer attached, so ordinary checks that integrators reach for all pass.

```
./poc_fork_relative_solvency.sh
Runtime: about 30 seconds
Requires: bitcoind / bitcoin-cli v29+ on PATH
Set KEEP_DATADIR=1 to preserve regtest datadirs
```

ARTIFACT PIPELINE

This is the off-chain double-spending path.



Fork-relative artifacts are internally valid. They are economically false against the most-work chain.

The artifacts form a convincing bundle: a trusted balance, a fresh signed challenge, a signed transaction, a local mempool accept result, and a UTXO proof. The bundle fails only when checked against an independent view of the most-work chain.

Step	Evidence from PoC	Security meaning
1. Balance	node0 getbalance: 10 BTC; node1: 0 BTC	Aggregate balance is local-chain-relative.
2. Challenge	verifymessage true on both nodes	The signature proves key control, not current UTXO ownership.
3. Signed spend	signrawtransactionwithwallet.complete true	Wallet can sign a spend that is already dead on the real chain.
4. Mempool accept	node0 allowed true; node1 missing-inputs	Spend validity is also local-chain-relative.
5. UTXO	node0 gettxout object; node1 null	The outpoint exists only on the stale branch.

AUGMENTED PROBES

Depth and freshness are branch-relative too.

A natural objection is that a careful verifier would require confirmations. The augmented probe shows that confirmation depth on the stale branch is attacker-controllable: after the attacker mines additional lower-work blocks, getbalance with minconf=6 again reports 10 BTC while the real chain reports 0.

```
funding tx confs on stale branch: 4
getbalance minconf=6 before attacker mines: 0 BTC
attacker peer mines 3 more blocks; node0 height: 108
funding tx confs on stale branch now: 7
getbalance minconf=6 after attacker mines: 10 BTC
node1 real chain getbalance minconf=6: 0 BTC
```

Two things catch this specific demo.

- chainwork differs across branches, but only helps if the verifier compares against an independent expected-work reference.
- peer topology is weak in the demo: one outbound peer, attacker-controlled. A full eclipse makes this harder to spot, but also raises the precondition.

SECTION / 04

Impact And Limits

Loss is realized by external systems that treat local-chain observations as global-chain facts.

IMPACT

The risk is compositional, not inflationary.

Bitcoin Core supplies a stale-observation primitive. External systems create the loss when they release irreversible value from that primitive.

Scenario	Relative risk	Reason
Cross-chain / off-chain settlement	High downstream	Observer releases non-BTC value from stale BTC evidence.
Eclipse plus automated payout	High downstream	The eclipse keeps the false view fresh until payout.
Lending / proof-of-funds	Medium	Fails when verifier relies on claimant machine or datadir.
Exchange credits stale deposit	Medium	Requires defective deposit accounting or poor observer setup.
Reserve attestation	Medium	Aggregate wallet balances can overstate solvency.
Withdrawal DoS	Low	Repeatedly selects outputs dead on the real chain.

Why it matters over time

Every new bridge, lender, custodian, payment processor and proof-of-reserves workflow reimplements freshness. The current APIs make the unsafe integration easy and the safe integration something the operator must already know.

LIMITATIONS

The report is intentionally scoped.

- No node can prove a higher-work chain does not exist. The ask is honest labeling and provenance, not omniscience.
- The PoC engineers the partition in regtest. Real-world delivery requires eclipse, hostile machine/datadir, hostile connectivity, operator error, or material hashrate.
- Under pure eclipse, the RPC artifact bundle is free: getbalance, getbalances, gettxout, testmempoolaccept and signmessage do not consult tip age, so withholding the honest chain is enough.
- Proof-of-work buys freshness optics only: the Bitcoin-Qt up-to-date icon and any verifier-authored tip-age rule. Rough 1/9 and 1/144 figures are Poisson order-of-magnitude means for 90-minute and 24-hour windows, not attack prices or thresholds; permissive block timestamps relax the heuristics further.
- Trusting the claimant's own machine or datadir is also free and unfixable in software. In that setup the claimant controls the view.
- Deepening the stale branch past a verifier's chosen minconf threshold does cost real proof-of-work on mainnet; the augmented probe keeps that boundary explicit.
- A well-peered verifier node checking specific outpoints defeats the strongest economic scenarios.
- The printed regtest tip age can be slightly negative because block timestamps may be ahead of wall clock; the relevant fact is that magnitude is seconds.

These limitations reduce severity. They do not eliminate the documentation and integration hazard.

SECTION / 05

Remediation

The safest fixes do not require changing consensus or breaking getbalance compatibility.

GUI

Label local observations as local observations.

When freshness cannot be established, the GUI should not render the amount as current spendable balance. It should show the provenance directly and mark the value provisional.

```
Locally known balance: 10 BTC
As of block 900000 (0000...abcd)
Network state not independently verified
```

Mark provisional when any freshness guard fails.

- wallet scanning is active
- initial block download is true
- wallet tip differs from node tip
- blocks are behind headers
- networkactive is false
- zero outbound peers
- tip age exceeds a conservative threshold

Offline signing must keep working. The fix is truthful labeling, not disabling valid offline workflows.

RPC AND DOCS

Preserve compatibility; add provenance.

Do not

- Do not change getbalance return type.
- Do not claim Core can guarantee no better chain exists.
- Do not present signmessage as proof of solvency.

Do

- Strengthen getbalance help text.
- Document signmessage as key-control only.
- Add provenance-rich getbalanceinfo.
- Add optional require_current fail-closed mode.

Suggested getbalanceinfo fields

Field class	Fields
Balance	trusted, untrusted_pending, immature, avoid_reuse state
Wallet provenance	wallet_lastprocessedblock hash, height and time; scanning state
Node provenance	node_bestblock hash, height, time; headers; chainwork; IBD
Network health	networkactive, connections_out, peer diversity summary
Freshness result	status current/provisional; freshness_reason; policy version

The field must be called a freshness assessment, not a guarantee. The goal is to make safe integrations obvious and unsafe integrations visibly provisional.

INTEGRATION GUIDANCE

Never settle from aggregate wallet balance.

The correct integration exists today: track specific outpoints on a verifier-operated, well-peered node. Aggregate balance and claimant-signed messages are auxiliary signals, not settlement gates.

- Use the verifier's own node, not the claimant's machine or datadir.
- Track the specific outpoint, script and amount.
- Require gettxout presence on the verifier's active chain with a confirmation threshold.
- Require scan complete, IBD false, acceptable headers-blocks gap, healthy outbound connectivity and live reorg handling.
- For bridges and high-value settlement, require independent observers to agree before minting or releasing value.
- Treat proof-of-control at one time as different from collateral locked over a term.

SECTION / 06

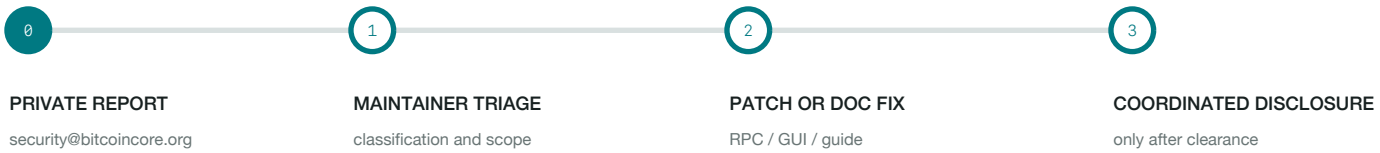
Disclosure Package

The report is ready for public disclosure, with explicit non-overclaiming language.

RESPONSIBLE DISCLOSURE

Submitted privately first.

Bitcoin Core's security policy says vulnerabilities should be reported to security@bitcoincore.org, and that an issue may be serious without requiring embargo or being classified as a vulnerability. This finding fits that boundary.



APPENDIX A

Independent verification record.

Evidence	Record
Binary	bitcoind --version returned Bitcoin Core daemon version v31.1.0 in supplied PoC output.
PoC rerun	run_output_rerun.txt reproduces balance 10 vs 0, valid signature, complete signed spend, allowed true vs missing-inputs, UTXO vs null.
Augmented probes	augmented_probes_output.txt confirms chainwork divergence, weak peer topology in demo, minconf escalation defeat and shipped help text excerpts.
Source paths	coins.cpp, wallet.cpp, overviewpage.ui, bitcoingui.cpp, blockchain.cpp and wallet RPC files were referenced at tag v31.1.
Official release	Bitcoin Core 31.1 was published on July 8, 2026 according to bitcoincore.org.
Official policy	Bitcoin Core security page directs vulnerability reports to security@bitcoincore.org.

APPENDIX B

References.

Reference	URL
Bitcoin Core 31.1 release notes	https://bitcoincore.org/en/releases/31.1/
Bitcoin Core 31.1 release announcement	https://bitcoincore.org/en/2026/07/08/release-31.1/
Bitcoin Core security advisories and disclosure policy	https://bitcoincore.org/en/security-advisories/
Bitcoin Core SECURITY.md	https://github.com/bitcoin/bitcoin/blob/master/SECURITY.md
Bitcoin Core contact page	https://bitcoincore.org/en/contact/
Bitcoin Core lifecycle policy	https://bitcoincore.org/en/lifecycle/
Source tag used by report	https://github.com/bitcoin/bitcoin/tree/v31.1

CYPHES final position: useful and reportable as private hardening disclosure. Public communication should wait for maintainer response and should preserve the distinction between off-chain double-spending and Bitcoin consensus double-spending.

FINAL / NETWORK

Medium / high confidence

Bitcoin remains consensus-intact. The failure is at the boundary where local-chain truth is treated as global solvency.

The report should ask for provenance, warnings and integration-safe APIs. It should not claim inflation, protocol theft or an on-chain double spend.