POSTMORTEM SEPTEMBER 2026

Exploit of Nimiq's legacy gas-abstraction contracts on Polygon PoS

TOTAL LOSS

50,496.11

USDC, USDT0 and USDC.e (USD stablecoins)

TAKEN BY ACTOR A

50,463.79

from the swap-liquidity address

TAKEN BY ACTOR B

32.32

from two Wallet users

DRAIN

2026-09-16 23:26:49 UTC

Polygon block 93,930,784

HANDLERS DISABLED

2026-09-17 18:20:39 – 18:40:01 UTC

18 h 54 min after the drain

EXPLOIT TRANSACTIONS

4 successful

plus 4 reverted attempts by actor A

CHAIN	Polygon PoS mainnet (chain id 137). The stolen funds were later bridged to Ethereum mainnet (chain id 1).
DATA	On-chain state as of 2026-11-12 (Polygon head block 97,207,231, Ethereum head block 26,401,463).
CONVENTIONS	All times are UTC. Token amounts are in token units, rounded to two decimals; the linked transactions show the exact amounts. USDC is native USDC on Polygon, USDC.e is bridged USDC, and USDT0 is the token the Gas Abstraction Update calls USDT.
EVIDENCE	Statements are read from on-chain receipts, logs, traces and storage or from verified contract source, unless they are marked Inference.

Contents

1 **Summary** 3

2 **Contracts involved** 5

3 **Root cause** 9

3.1 The design 9

3.2 Why the forged call succeeded 10

4 **Timeline (UTC)** 13

4.1 Before the exploit 13

4.2 Actor A: funding, preparation, exploit 13

4.3 Actor A: movement of the stolen funds 15

4.4 Actor B 19

4.5 Response 21

4.6 Public reports (for comparison with the chain) 23

5 **Addresses** 24

6 **Scope check** 27

6.1 Which handler calls were forged 27

6.2 Other HTLCs 29

6.3 Transfer handlers and the USDC.e→USDC conversion 29

7 **Appendix** 30

7.1 All transaction hashes 30

SECTION 1

Summary

On 16 September 2026, an attacker used a flaw in Nimiq's legacy gas-abstraction contracts on Polygon to take 50,463.79 in USD stablecoins from the swap-liquidity address operated by Green Ocean. About 48 minutes later, a second attacker used the same flaw to take 32.32 from two Nimiq Wallet users. Nimiq started to disable the contracts 18 h 54 min after the first theft. No other loss was found (section 6).

- **Terms.**
  - **Handler.** One of the Polygon contracts of the OpenGSN-based integration described in the Gas Abstraction Update. Nimiq deployed them. They carried out gas-abstracted USDC and USDT transfers and swaps for the Nimiq Wallet, and USDT transfers for Nimiq Pay (section 2).
  - **HTLC** (hash time-locked contract). The escrow of a swap. It locks the stablecoins until the counterparty redeems them with a secret, or until the sender takes a refund after a timeout. Two of the handlers, the HTLC handlers, held these escrows for stablecoin swaps in the Nimiq Wallet. The update calls them the legacy swap contracts.
  - **RelayHub.** OpenGSN's shared third-party hub contract. Every OpenGSN relay submits its relayed calls through it.
  - **Paymaster.** The OpenGSN contract that accepts a relayed call and pays for it.
  - **EOA** (externally owned account). A plain address controlled by a private key.
- **What happened.** Two independent actors used one flaw in Nimiq's legacy OpenGSN handler contracts. Both are attackers. They are called Actor A and Actor B below, and their identities are unknown. Only the two HTLC handlers were exploited.
  - **Actor A**, the main attacker, used the EOA `0x2258…c45d`.
    - On **2026-09-16 at 23:26:49** (block 93,930,784, tx `0xb067…58c0`), A registered itself as an OpenGSN relay. In the same transaction it made the RelayHub call `execute()` on the two HTLC handlers, three times in total.
    - Each call carried a forged request in the name of the swap-liquidity address `0x24cb…6773` (see Victims below). The handlers pulled that address's entire balance of three tokens through its standing approvals, and locked the tokens in HTLCs whose recipient and hash A had chosen.
    - At 23:28:34 (tx `0xb2ca…d169`), A redeemed the three HTLCs to the fresh address `0x4b0e…9e04` (called A's receiver below).
  - **Actor B**, a second attacker with no known link to A (section 4.4), used the EOA `0xfa20…a618`. About 48 minutes later, on 2026-09-17 at 00:14:39 and 00:14:43, B used the same `execute()` path with the "close" branch. B paid out to itself two expired, never-refunded HTLCs that ordinary Nimiq Wallet users had opened for swaps in September 2025 (**H1** and **H2**, section 4.1).
- **Victims.**
  - The swap-liquidity address `0x24cb…6773` (EOA), lost to A. It is operated by Green Ocean, the independent liquidity provider that powers Fastspot's swaps. Fastspot is the swap service whose API the Nimiq Wallet uses (`fastspot` in the Wallet's `src/config/config.mainnet.ts`). The address is the other side of every stablecoin swap in the Wallet: it redeems the HTLCs that users open (for example **H1** and **H2**, section 4.1) and opens HTLCs of its own, calling `open` and `redeem` on `0xF615…` directly. Verichains and SlowMist call it "Nimiq's swap-liquidity wallet".
  - Nimiq Wallet user `0x9db8…b1c4` (a private user's account; the **H1** swap in section 4.1), lost to B. B took this user's HTLC refund.
  - Nimiq Wallet user `0xc451…a5bc` (a private user's account; the **H2** swap in section 4.1), lost to B. B took this user's HTLC refund.

- **Contracts used.** `ERC20PermitHTLCHandler` `0x0cFD…959b` (native USDC) and `ERC20MetaHTLCHandler` `0xF615…359f` (USDC.e and USDT0), both reached through OpenGSN RelayHub `0x6C28…750d`. The transfer handlers and the USDC.e→USDC swap handler were not used (section 6).
- **Other approvers.** Ordinary users who had given a handler a token approval were exposed until the contracts were disabled, but only through small residual allowances. These are Nimiq Wallet users and, for USDT, also Nimiq Pay users, whose config uses the same handlers.
  - Before the drain, about 2,230 (owner, token) pairs together had about 12.15 in stablecoins within reach, at most 0.07 per user.
  - No forged call was found that took any of it. From 2026-09-16 10:37 to the last `setRelayHub(0)` the check is complete: in that window the RelayHub logged no forged call other than A's and B's. For the time before, it has one gap: the transactions of two other dapps' paymasters were not decoded (section 6.1, item 3).
  - The users who did lose money lost it through B's raid on stale HTLCs (section 6).
- **Loss per token.** All three tokens are USD stablecoins.

Token (contract)	Actor A, victim `0x24cb…6773`	Actor B, Nimiq Wallet users	Total
USDC (native) `0x3c49…3359`	26,130.64	4.54 from `0xc451…a5bc`	26,135.18
USDT0 `0xc213…8e8F`	24,332.49	27.78 from `0x9db8…b1c4`	24,360.27
USDC.e `0x2791…4174`	0.66	0	0.66
Sum	**50,463.79**	**32.32**	**50,496.11**

- **Transaction count.** Four successful exploit transactions:
  - A: `0xb067…58c0` (drain) and `0xb2ca…d169` (redeem).
  - B: `0x5868…3b16` and `0x15cc…8a63`.
  - A also sent four reverted attempts (**P3**, **P4**, **P6**, **P8** in section 4.2).
- **Where the money went.**
  - A swapped the USDT0 to USDC on KyberSwap at 23:34:37, about 6 minutes after the redeem.
  - A bridged 50,447.83 USDC to Ethereum with LI.FI's Eco route, and converted it to DAI in Sky's LitePSM at 23:37:35 on 2026-09-16.
  - On 2026-10-29, 50,447.38 DAI went into a Uniswap v4 pool that A's receiver had just created against a token it had just deployed. In the same block, a different fresh address took 50,443.75 DAI out of that pool and shielded 50,317.64 DAI into Railgun. That this second address belongs to A is inference (section 4.3).
  - A's own starting gas also came out of Railgun on Ethereum.
- **Disabling.** Nimiq's deployer EOA `0xDB88…9DDA` deployed the legacy contracts and is their owner. It called `setRelayHub(0x0)` on six handlers between **2026-09-17 18:20:39 and 18:40:01** (18 h 54 min after the drain). These calls are what the update describes as "we disabled these contracts". Since then `execute()` can no longer be reached, because `onlyRelayHub` compares the caller with `address(0)`.
- **Not disabled.** An older handler deployed by the same owner, `ERC20MetaHandler` `0x4b76…c7a2` from 2023-02-22, still has `getHubAddr() == RelayHub` at the head block. Its verified `execute()` is an earlier code version that does check the EIP-712 signature and the nonce. That it therefore does not have this flaw is inference from the source; it was not tested by simulation. It has never processed a relayed call, and no token approval has ever named it as spender (section 2).

SECTION 2

Contracts involved

All Nimiq handlers were deployed by the EOA `0xDB88…9DDA`, and it is their only owner. `owner()` returns this address at block 93,930,783 and at the head block for every contract below. The owner's complete transaction history contains no `transferOwnership` call. The owner never set any hub other than `0x6C28…750d` or `0x0`. All handlers were compiled with Solidity v0.8.17+commit.8df45f5f, optimizer on.

Transfer handler, USDC.e + USDT0 (Wallet config; also in the Nimiq Pay config for USDT)

`0x98E69a6927747339d5E543586FC0262112eBe4BD`

Contract (verified name)	ERC20MetaHandler
Deploy tx (block, UTC)	`0xcf939076fd7df7cbe47be9434843156d30f1be4edfb1138ce6d1d2fbd03f3292` (39,784,005; 2023-02-27 22:46:42)
Hub set to RelayHub	`0x372ed11cfee07652e67348143460599fac44c92edbb84f779a62b9a69eb7e224`
Hub at 93,930,783	RelayHub
Disabled by `setRelayHub(0)`	`0xc874e863e98b5ace14b1d5243e7b163f437603468723dd897f8a41622f4d423c` (93,976,188; 2026-09-17 18:21:55)
Verified source	Blockscout verified; Sourcify exact_match

HTLC handler, USDC.e + USDT0 (Wallet config; also in the Nimiq Pay config for USDT) — EXPLOITED

`0xF615bD7EA00C4Cc7F39Faad0895dB5f40891359f`

Contract (verified name)	ERC20MetaHTLCHandler
Deploy tx (block, UTC)	`0x0f388c1122bb5a94184397f919113bcbbe9ae17e938a215185c04141c01a14e0` (40,086,681; 2023-03-07 23:15:43)
Hub set to RelayHub	`0x0db7f5ea84c5883e52c5daed92088babe398cc82ba009708e08246e2e1d0507d`
Hub at 93,930,783	RelayHub
Disabled by `setRelayHub(0)`	`0xf42a363a9bebcc266642eaaa8bcd920ae22f098c4b1ff9b568a6189654dbd0cf` (93,976,364; 18:26:19)
Verified source	Blockscout verified; Sourcify exact_match

Transfer handler, native USDC (Wallet config)

`0x3157d422cd1be13AC4a7cb00957ed717e648DFf2`

Contract (verified name)	ERC20PermitHandler
Deploy tx (block, UTC)	`0x50247ac8d1fa4491744dfb9e3b634983db362be93239f2bd59f451470e6aa14a` (51,869,807; 2024-01-02 23:09:30)
Hub set to RelayHub	`0x91388ff08ed66ab2b16540d256137cc5d6f35b5785d8a1c646f7038b7161f399`
Hub at 93,930,783	RelayHub
Disabled by `setRelayHub(0)`	`0x62efc65a3a903b6623bbc7b320cceae914425f7c2498b811eed16d00da90c42c` (93,976,137; 18:20:39)
Verified source	Blockscout verified; Sourcify exact_match (runtime)

HTLC handler, native USDC (Wallet config) — EXPLOITED

`0x0cFD862bE942846Cebad797d7c1BC6e47714959b`

Contract (verified name)	ERC20PermitHTLCHandler
Deploy tx (block, UTC)	`0xab827b175e2e4cb407dd402c7f7a51c7d42d995d9c4cefdf33c68767c27f8dac` (54,383,444; 2024-03-07 19:01:03)
Hub set to RelayHub	`0xda699c7e153729f70556bca71a70aea8c37260df5fbefd4af94047e130c26bc0` (first attempt `0xe4c8b7421bd0366eb33192f5544cc76f6282a6478b56968e388ab74512091dc8` reverted)
Hub at 93,930,783	RelayHub
Disabled by `setRelayHub(0)`	`0x22570a382b21efa372bec4dbccc9faf62fafb338900bacefc8d0ccf74db03252` (93,976,404; 18:27:19)
Verified source	Blockscout verified; Sourcify exact_match

USDC.e→USDC swap handler (Wallet config)

`0xfAbBed813017bF535b40013c13b8702638aC25CD`

Contract (verified name)	ERC20MetaUniswapHandler
Deploy tx (block, UTC)	`0x10a5a2dbfcdec1b398042d2243cd33427f4286b1c4b513fc6d24f8e7a729e5ff` (54,384,181; 2024-03-07 19:27:41)
Hub set to RelayHub	`0x9f58e8de8d2148b21ba2b6fd768b1e5f00d9729bd28c0247ee9bd1bc4ff6e8c5`
Hub at 93,930,783	RelayHub
Disabled by `setRelayHub(0)`	`0xc98a49f05011da6c50b560b56d8beb20121e0318a3ef963e51a0dc6a9a667f49` (93,976,547; 18:30:54)
Verified source	Blockscout verified; Sourcify exact_match

Transfer handler, USDT0 (the "sixth handler": deployed for USDT cashlinks, a feature that was never released; never used)

`0x3c870b039BF82D2F883b6E89D41d06d26b9F4486`

Contract (verified name)	ERC20MetaHandler
Deploy tx (block, UTC)	`0xf34d1cc288e1e8826d9a1a78dd747653d06a3561394b21ec3b2e5f7c9b3cfd10` (77,626,862; 2025-10-13 12:26:39)
Hub set to RelayHub	`0xe8c6e2d5c48e466cb89dc556221716f125826c40cc5e3580f6783dc41be79ecd`
Hub at 93,930,783	RelayHub
Disabled by `setRelayHub(0)`	`0x6169a24393268d29bf74d9cb8fed7401551a752c9e1c39cb788a9875265a56bd` (93,976,912; 18:40:01)
Verified source	Blockscout verified; Sourcify match

Unconfigured handler

`0xe4491D94a6f594273157965872be1ef4aEb26501`

Contract (verified name)	unverified; `versionRecipient() = 2.2.6+opengsn.recipient.erc20meta.handler`
Deploy tx (block, UTC)	`0x3ce4e668e774eba40f0cc7858e0478eb1eb425d3b8cd03df2847c8c33d70cc01` (77,625,843; 2025-10-13 11:52:41)
Hub set to RelayHub	never
Hub at 93,930,783	`0x0`
Disabled by `setRelayHub(0)`	not needed
Verified source	not verified

Oldest transfer handler, USDC.e (never in a Wallet config). Earlier code: its `execute()` verifies the signature and nonce

`0x4b766a07d464451415f43cd9334e77fe4122c7a2`

Contract (verified name)	ERC20MetaHandler
Deploy tx (block, UTC)	`0x009afe209b525fea0fcfbff2b0e4cded1e6e48503b4e24f8f5e7b8451ba4f9ac` (39,602,325; 2023-02-22 22:44:12)
Hub set to RelayHub	`0xc81080a213c815bf2a911416ea1c3a263b96f90fe307aaac0ac9f48d28215227`
Hub at 93,930,783	RelayHub
Disabled by `setRelayHub(0)`	**not disabled; hub is still RelayHub at head**
Verified source	Blockscout verified

USDC.e/USDC Uniswap v3 pool used by the swap handler (not a Nimiq contract)

`0xD36ec33c8bed5a9F7B6630855f1533455b98a418`

Contract (verified name)	UniswapV3Pool, created by Uniswap factory `0x1F98431c8aD98523631AE4a59f267346ea31F984`
Deploy tx (block, UTC)	`0xd19d05adc8fe2fde345712b13f80a227550bdfcc6151e1245f23446ea13bf64d` (48,926,712; 2023-10-20 02:25:52)
Verified source	verified

OpenGSN RelayHub (shared public infrastructure)

`0x6C28AfC105e65782D9Ea6F2cA68df84C9e7d750d`

Contract (verified name)	RelayHub, `versionHub() = 2.2.0+opengsn.hub.irelayhub`
Deploy tx (block, UTC)	`0x23f82fcb75cdeea8a82eedb31d77493500a5c7e360b694284c9c220d6c6b8555` (14,119,440; 2021-05-05 11:55:49), deployer `0x8C1FD2DE219c98f5F88620422e36a8A32f83324E`
Verified source	Blockscout verified; Sourcify match

OpenGSN StakeManager

`0x15C7B7CE10f3A9AE63554bCE7C54d0a818E967C7`

Contract (verified name)	StakeManager, deployer `0x8C1FD2DE219c98f5F88620422e36a8A32f83324E`
Deploy tx (block, UTC)	`0x0849a963f3fe2dba3ceb1bd5ce2414684dcca50fe45c099c32fbf239c665fc90` (14,119,432; 2021-05-05 11:55:33)
Verified source	Blockscout verified

Notes on the contract list:

- **The "sixth handler".** The owner disabled six handlers (R1 – R6), and the Wallet config lists five. The sixth is `0x3c87…4486`.
  - **What it was for.** It is the contract of USDT cashlinks, a feature that was developed in October 2025 and never released. Unmerged branches of the public Keyguard and Hub repositories name it as the USDT cashlink contract, and the Keyguard and the Wallet now list it with the comment "OpenGSN USDT cashlink, never used".
  - Up to `adc1ec51…` of 2026-10-25, it appears in no revision of the Wallet's `src/config/config.mainnet.ts`.
  - The RelayHub never relayed a call to or for it.
  - No `Approval` event on any token contract has ever named it as spender, from block 39,602,000 to the head (section 6.1 item 4). So although its `execute()` was unprotected until R6, it had no allowance to spend.

- **Two more contracts of the same owner.** The owner also deployed `0x4b76…c7a2`, which is still hub-enabled, and `0xe449…6501`, which never had a hub.
  - Neither address appears in the code of the Wallet, the Hub, the Keyguard or Nimiq Pay.
  - `0x4b766a07…` was configured by owner nonces 1–7 on 2023-02-22 and 2023-02-23 (**C1**–**C7** in section 7.1): `setRelayHub(RelayHub)`, `setSwapRouter(0x68b3465833fb72a70ecdf485e0e4c7bd8665fc45)`, `setWrappedChainToken(WPOL)`, `registerToken(USDC.e, 500)`, `updateRelayGas` twice, and a 1 POL RelayHub `depositFor`. The deposit was withdrawn on 2023-02-27 (**C8**, tx `0xfd49…a327`, `withdrawRelayHubDeposit`).
  - The RelayHub's logs contain no `TransactionRelayed` or `TransactionRejectedByPaymaster` naming `0x4b766a07…`.
  - No `Approval` event on any token contract has ever named `0x4b766a07…` or `0xe4491D…` as spender, from block 39,602,000 to the head (section 6.1 item 4).
  - It holds no USDC, USDC.e, USDT0, WPOL or POL.
  - Its verified source differs from the later handlers. `execute()` (`ERC20MetaHandler.sol` lines 124–135) has no `onlyRelayHub`, but it calls `verifyInternal(request, domainSeparator, requestTypeHash, suffixData, signature)` (line 126) and requires `request.nonce == getNonce(request.from)` (line 127). A forged request that `request.from` did not sign therefore reverts with "Base: signature mismatch" (`BaseCombinedGsnHandler.sol` line 63). This is **inference** from the source; it was not tested by simulation.
  - The verified source of `0x3c870b03…` has the same unprotected `execute()` as `0x98E6…`.
- **Token naming.** The contract `0xc213…8e8F` reports the symbol `USDT0` today. It is the token the Wallet config calls `usdt_bridged`.

SECTION 3

Root cause

3.1 The design

Every Nimiq handler is three OpenGSN roles in one contract: recipient (the contract the user's request is addressed to), paymaster (the contract the RelayHub asks, in `preRelayedCall()`, whether to accept a relayed request and pay for it) and forwarder (the contract whose `execute()` the RelayHub calls to carry out the request).

- `BaseCombinedGsnHandler is IRelayRecipient, IPaymaster, IForwarder, Ownable` (`BaseCombinedGsnHandler.sol`, line 14).
- Its `isTrustedForwarder(f)` returns `f == address(this)` (lines 69–71).

The authentication of a gasless request sits only in the handler's **paymaster** hook. `preRelayedCall()` calls `verifyCallPrivate()`, which checks the EIP-712 signature (`verifyInternal`), the nonce, and the business preconditions (`checkOpen`, `checkRedeem`, `checkRefund`). See `ERC20MetaHTLCHandler.sol` lines 176–219:

```
// ERC20MetaHTLCHandler.sol (0xF615…), lines 205-210
function preRelayedCall(GsnTypes.RelayRequest calldata relayRequest, bytes calldata signature, bytes
calldata approvalData, uint256 maxPossibleGas) external override virtual onlyRelayHub
onlyToSelf(relayRequest) returns (bytes memory context, bool revertOnRecipientRevert) {
    (relayRequest, signature, approvalData, maxPossibleGas);
    require(relayRequest.request.to == address(this), "Meta: illegal request.to");
    require(relayRequest.request.nonce == getNonce(relayRequest.request.from), "Meta: Invalid nonce");
    (OpenRequestData memory openRequestData, CloseRequestData memory closeRequestData, FeeInformation
memory feeInformation, ApprovalRequestData memory approvalRequestData, bytes32 secret) =
verifyCallPrivate(relayRequest.request, relayRequest.relayData, signature, approvalData);
```

The handler's **forwarder** hook, `execute()`, moves the tokens. It checks nothing except that the caller is the RelayHub:

```
// ERC20MetaHTLCHandler.sol (0xF615…), lines 221-236
function execute(ForwardRequest calldata request, bytes32 domainSeparator, bytes32 requestTypeHash,
bytes calldata suffixData, bytes calldata signature) public override payable onlyRelayHub returns (bool
success, bytes memory ret) {
    (request, domainSeparator, requestTypeHash, suffixData, signature);

    bytes4 methodId = GsnUtils.getMethodSig(request.data);
    // chainTokenFee is not needed here, so we just set it to 0
    (OpenRequestData memory openRequestData, CloseRequestData memory closeRequestData, FeeInformation
memory feeInformation) = decodeRequestDataPrivate(methodId, request.from, request.data, 0);
    nonces[request.from] = nonces[request.from] + 1;
    if (methodId == this.open.selector || methodId == this.openWithApproval.selector) {
        openPrivate(request.from, openRequestData);
    } else {
        closePrivate(closeRequestData, feeInformation);
    }

    success = true;
    ret = "";
}
```

```
// BaseCombinedGsnHandler.sol, lines 39-42
modifier onlyRelayHub() {
    require(msg.sender == getHubAddr(), "Base: illegal msg.sender");
    _;
}
```

```
// ERC20MetaHTLCHandler.sol, lines 54-58 and 108-112
function openPrivate(address userAddress, OpenRequestData memory requestData) private {
    require(requestData.htlc.token.transferFrom(userAddress, address(this), requestData.htlc.amount),
"HTLC: Deposit transfer failed");
    deposits[requestData.htlc.token] = deposits[requestData.htlc.token] + requestData.htlc.amount;
    htlcs[requestData.id] = requestData.htlc;
}
function closePrivate(CloseRequestData memory requestData, FeeInformation memory feeInformation) private
{
    htlcs[requestData.id].token.transfer(requestData.target, htlcs[requestData.id].amount -
feeInformation.fee);
    deposits[htlcs[requestData.id].token] = deposits[htlcs[requestData.id].token] -
htlcs[requestData.id].amount;
    delete htlcs[requestData.id];
}
```

The native-USDC HTLC handler `0x0cFD…` has the same functions:

- `execute()`: `ERC20PermitHTLCHandler.sol` lines 222–237
- `openPrivate()`: line 54
- `closePrivate()`: line 109
- `preRelayedCall()`: line 206

The transfer and swap handlers have the same `execute()` pattern:

- `ERC20MetaHandler.sol` (transfer, USDC.e/USDT0) lines 122–134 → `transferPrivate(request.from, …)` = `token.transferFrom(request.from, requestData.target, amount)` (lines 15–17)
- `ERC20PermitHandler.sol` (transfer, native USDC) lines 122–134 → same `transferPrivate`
- `ERC20MetaUniswapHandler.sol` (swap) lines 181–190 → `swapPrivate(request.from, …)`

3.2 Why the forged call succeeded

The missing check is this: `execute()` does not verify that the request was signed by `request.from`, that its nonce is current, or that the RelayHub ran the contract's own `preRelayedCall()`. It trusts `onlyRelayHub` instead.

That trust is misplaced, because the RelayHub calls whichever paymaster and forwarder the **relay request names**. This is OpenGSN RelayHub v2.2.0, verified source `RelayHub_flattened.sol`:

- `relayCall()` (lines 1429–1460) checks only the relay worker (the account that sends the transaction on behalf of a relay: `msg.sender == tx.origin`, a registered and staked manager), gas limits and packing.
- `innerRelayCall()` (lines 1537–1610) calls `relayRequest.relayData.paymaster.call(preRelayedCall…)`, then `GsnEip712Library.execute(relayRequest, signature)`, then `relayRequest.relayData.paymaster.call(postRelayedCall…)`.
- `GsnEip712Library.execute()` (lines 678–692) calls `relayRequest.relayData.forwarder.call(IForwarder.execute.selector, relayRequest.request, …)`.
- Nothing binds the paymaster to the forwarder or to `request.to`.

Step by step, for the USDT0 leg of tx `0xb067…58c0`. The steps follow its call trace.

1. **Relay setup.** EOA `0x2258…c45d` sends a type-2 transaction to itself. Earlier, in tx `0xc7cb9bbc…`, it had installed an EIP-7702 delegation to `0x0404…e251`; its code at blocks 93,930,783 and 93,930,784 is `0xef0100 0404853a…`. Running that code, `0x2258` becomes an OpenGSN relay:
   - StakeManager: `setRelayManagerOwner(0x2258…)`, `stakeForRelayManager(0x2258…, unstakeDelay 1000)` with 1 POL, `authorizeHubByOwner(0x2258…, RelayHub)`
   - RelayHub: `addRelayWorkers([0x2258…])`
   - Logs: `StakeAdded`, `HubAuthorized`, `RelayWorkersAdded`.
2. `0x2258` calls `RelayHub.relayCall(maxAcceptanceBudget 1,000,000, relayRequest, signature = 65 zero bytes, approvalData = 0x, externalGasLimit 830,414)`.
   - `msg.sender` is `0x2258` and equals `tx.origin`, so the hub's "relay worker must be EOA" check passes.
   - Relay request:
     - `request.from = 0x24cb173ae221aea93369f34bdcf0ddb35b436773` (the victim)
     - `request.to = 0xF615…`, `request.nonce = 0`, `gas 500,000`, `validUntil 2026-09-17 00:26:49` (one hour after the block)
     - `request.data = open(id 0x7f6a2442add17a6ba28d2bc4d301450f789e81dc2df2b214d33bc3f8745704db, token USDT0, amount 24,332.49, refund = recipient = 0x8ed97c79a95b311e28b91eed6077af1576533fcd, hash 0xec4916dd28fc4c10d78e287ca5d9cc51ee1ae73cbfde08c6b37324cbfaac8bc5, timeout 2026-09-16 23:26:50, fee 0)`. The timeout is one second after the block time.
     - `relayData.paymaster = 0x2258…` (the attacker itself), `relayData.forwarder = 0xF615…`
     - `relayData.gasPrice = pctRelayFee = baseRelayFee = 0`
3. The hub calls `0x2258.getGasAndDataLimits()`, then `innerRelayCall` on itself, then `0x2258.preRelayedCall(…)` (selector `0x00be5dd4`). The attacker's code accepts. **The handler's own `preRelayedCall()` is never called, so `verifyCallPrivate()` never runs.** Because gas price and fees are 0, the maximum charge is 0. The attacker made no RelayHub deposit: there is no `Deposited` event for `0x2258`.
4. The hub calls `0xF615.execute(request, …)` (selector `0xe024dc7f`). `msg.sender` is the RelayHub, so `onlyRelayHub` passes. `execute()` decodes `request.data`, increments `nonces[victim]`, and calls `openPrivate(request.from = victim, …)`.
5. `openPrivate` calls `USDT0.transferFrom(0x24cb…, 0xF615…, amount)` for the 24,332.49 USDT0, with the handler as `msg.sender`. This spends the victim's standing approval to the handler. The `Approval` event in the same receipt shows the remaining allowance as `0xff…e4f135f75f`, which is still close to 2^256. The handler stores `htlcs[id]` with the attacker's recipient and hash, and adds the amount to `deposits[USDT0]`.
6. The hub calls `0x2258.postRelayedCall(…)` (`0x76fa01c3`). **The handler's `postRelayedCall()`, the only place where the relay path emits `Open` (lines 238–252), is not called. That is why no `Open` event exists for these three HTLCs.** The direct `open()` emits `Open` itself (line 67), but it was not used. The hub emits `TransactionRelayed(relayManager 0x2258, worker 0x2258, from 0x24cb…, to 0xF615…, paymaster 0x2258…, selector 0x3878a2a3, status 0, charge 0)`.

The same transaction does the same for native USDC on `0x0cFD…` (26,130.64 USDC) and USDC.e on `0xF615…` (0.66 USDC.e). The amounts equal the victim's balances, which the attacker's code read with `balanceOf` just before each relay call. All three use the same recipient and hash. The hash `0xec4916dd28fc4c10d78e287ca5d9cc51ee1ae73cbfde08c6b37324cbfaac8bc5` is `sha256(bytes32(uint256(1)))`.

The redeem (tx `0xb2ca…d169`) uses no forged path:

- `0x2258` calls the CREATE2 deployer `0x4e59…956c`, which deploys `0x8ed9…3fcd` (the precomputed HTLC recipient).

- That contract's constructor calls the normal public `redeem(id, target = 0x4b0e8b2c38b0044405e24db481d00be7d5419e04, secret = bytes32(1), fee = 0)` (selector `0x506bb7ba`) on each handler. `checkRedeem` passes because `msg.sender` is the stored recipient and `sha256(secret)` matches.
- The handlers emit `Redeem(id, 0x…01)` three times and transfer the full amounts to `0x4b0e…9e04`.
- The deployed contract's runtime code is a 57-byte revert stub. Blockscout's label "SupplyChainTypes" for it is a bytecode-match artifact.

Actor B's variant (`0x5868…3b16`, `0x15cc…8a63`) used the `else` branch of `execute()`:

- `request.data = refund(id, target = 0xfa20…, fee 0)`. `execute()` calls `closePrivate()` with no refund-address, timeout or signature check. The signature was empty (`0x`).
- This paid out existing HTLCs to B. B used its own freshly deployed paymasters (`0x821e…af47`, `0xcdc6…a2cb`) and a plain EOA relay worker with no EIP-7702.
- So EIP-7702 was not needed for the exploit. A used it only to put the relay setup and three relay calls into one transaction.

The transfer handlers would have allowed the same forgery with `transferPrivate(request.from, target, amount)`, sending tokens directly to any target. The swap handler's `swapPrivate` pays the swap output to `request.from`, so a forged call there could force a user's swap but not redirect its output. That last point is inference from the code: `swapUniswapCallback` requires a registered pool, and `pool.swap(userAddress, …)`. Neither handler type was used (section 6).

SECTION 4

Timeline (UTC)

4.1 Before the exploit

H2 POLYGON **2025-09-02 18:31:42 UTC** Block 75,971,924

`0xbc30531fa287d1ab0968388a0ef1c4f11f8fde93f977b11fe3903332d21f864c`

From	`0x750dd91b9948792c0850c83b059e9abaefa243c0` (Fastspot relay worker: the worker of the OpenGSN relay registered as `https://gsn.main.fastspot.io`, which Fastspot ran for the Nimiq Wallet, see section 5)
To	RelayHub
Function	`relayCall` → `openWithPermit` on `0x0cFD…`
Token / amount	USDC 4.54 from `0xc45101224affe424c8528304a174ccf0479fa5bc`
Note	Legitimate user swap HTLC, id `0x0ba929317e74edbf93e8f9403beef2637789a0dbe217603d6a71aa0fcf86c7bf`, recipient `0x24cb…6773`, timeout 2025-09-02 20:16:25. Never redeemed or refunded. B stole it later (B10).

H1 POLYGON **2025-09-03 02:44:40 UTC** Block 75,985,843

`0x81c2ef136f6ff98efb78578c14f6881556e55c36bf0ea851610593852dd09674`

From	`0x750dd91b9948792c0850c83b059e9abaefa243c0`
To	RelayHub
Function	`relayCall` → `openWithApproval` on `0xF615…`
Token / amount	USDT0 27.78 from `0x9db8c021407755f5537ca90c8a34dcbb0197b1c4`
Note	Legitimate user HTLC, id `0x52df369ab40c5e01a8d1c89ffb1e8644ff447cf22824ffea7880df321bf0fbd7`, timeout 2025-09-03 04:29:00. Never closed. B stole it later (B9).

These two were the only open HTLCs at block 93,930,783. `deposits(USDT0)` on `0xF615…` was 27.78 USDT0 and `deposits(USDC)` on `0x0cFD…` was 4.54 USDC.

4.2 Actor A: funding, preparation, exploit

E1 ETHEREUM **2026-09-16 22:13:35 UTC** Block 25,992,951

`0xe7bd285ec3b1eaebba1bb56494ef0288e63c0a33dc51226972b2e2c0a1fbc2c6`

From	`0xd6bb56a50c6a701148b7e0ac706fc4e63e89f75f` (Railgun broadcaster)
To	`0xAc9f360Ae85469B27aEDdEaFC579Ef2d052aD405` (Railgun RelayAdapt)
Function	Railgun unshield
Token / amount	0.009975 ETH to `0x2258…c45d`. Railgun Treasury fee 0.000025 WETH
Note	A's first funds come from the Railgun privacy pool (`RailgunSmartWallet` `0xFA7093CDD9EE6932B4eb2c9e1cde7CE00B1FA4b9`).

E2 ETHEREUM **2026-09-16 23:01:23 UTC** Block 25,993,190

`0x033d4870257c0cefd055b829d69c33e1b99ab1bb8849574c73ac7faa8ae0fdb2`

From	`0x2258…c45d` (nonce 0)
To	LiFiDiamond `0x1231DEB6f5749EF6cE6943a275A1D3E7486F4EaE`
Function	`swapAndStartBridgeTokensViaAcrossV4`
Token / amount	0.007277 ETH
Note	Across V3 `FundsDeposited` on Ethereum SpokePool `0x5c7BCd6E7De5423a257D81B442095A1a6ced35C5`: destination 137, depositId 4,591,126, depositor `0x2258…`.

P1 POLYGON **2026-09-16 23:01:28 UTC** Block 93,929,770

`0xc350aa9a8ab6608e1f0c2f1dd9dc36bdaf85287c03021712c8061119f2cd2db8`

From	`0x394311a6aaa0d8e3411d8b62de4578d41322d1bd` (Across relayer)
To	Polygon SpokePool `0x9295ee1d8c5b022be115a2ad3c30c72e34e7f096`
Function	`fillRelay` (FilledRelay, depositId 4,591,126)
Token / amount	WETH 0.007236 swapped to 184.36 POL sent to `0x2258…`, plus a dust amount of WETH
Note	A's gas money on Polygon.

P2 POLYGON **2026-09-16 23:08:04 UTC** Block 93,930,034

`0xadd5d2fa5d9b88047278ad9b5813fe73c15eb36136970226b5023d4587f9fadf`

From	`0x2258…` (nonce 0, type 4)
To	CREATE2 deployer `0x4e59b44847b379578588920ca78fbf26c0b4956c`
Function	CREATE2 + EIP-7702 authorization
Note	Deploys `0x70e8139d854c9304159844c7eb5474db2877216a` (unverified) and delegates `0x2258` to it. The authorization uses nonce 1.

P3 POLYGON **2026-09-16 23:08:31 UTC** REVERTED Block 93,930,052

`0xd4035e6ce423ca3d2439365554970f371c9c1074f5dae612fa900df10fa58a9f`

From	`0x2258…` (nonce 2)
To	`0x2258…` (self)
Function	`0x17f859b7` (attack entry) with 1 POL
Note	Reverted: RelayHub "invalid externalGasLimit". Relay setup plus a USDC `balanceOf` read in the trace.

P4 POLYGON **2026-09-16 23:09:03 UTC** REVERTED Block 93,930,073

`0xc8066f7052bf748e9ddd8fcad93f2ecda6b68b5f8c0abb7fd00974abd582c140`

From	`0x2258…` (nonce 3)
To	self
Function	same
Note	Reverted: "invalid externalGasLimit".

P5 POLYGON **2026-09-16 23:22:18 UTC** Block 93,930,603

`0xc7cb9bbcb2b38f902576d980488917da09e92875a392744289116dc245bbaec1`

From	`0x2258…` (nonce 4, type 4)
To	`0x4e59b448…`
Function	CREATE2 + EIP-7702 authorization
Note	Deploys `0x0404853a9d3fdd57073f06be08a72401e85ee251` (unverified) and re-delegates `0x2258` to it. The authorization uses nonce 5.

P6 POLYGON **2026-09-16 23:22:54 UTC** REVERTED Block 93,930,627

`0x7f02ce7c3e24e87c2cecb84e8e06169a6fdd1db0b775a65bf0ede39dd7ee0a52`

From	`0x2258…` (nonce 6)
To	self
Function	`0x17f859b7` with 1 POL
Note	Reverted ("low gas") after the first forged open had succeeded inside the call, so all of it was rolled back. A test run.

P7 POLYGON **2026-09-16 23:26:49 UTC** Block 93,930,784

`0xb067efae73637f3564f58af7f6027afc497e81e47624b0048636085c678858c0`

From	`0x2258…` (nonce 7)
To	self
Function	`0x17f859b7` with 1 POL: relay setup, then 3 × `RelayHub.relayCall` → handler `execute` → `openPrivate`
Token / amount	USDC 26,130.64 victim→`0x0cFD…`; USDT0 24,332.49 victim→`0xF615…`; USDC.e 0.66 victim→`0xF615…`
Note	**The drain.** Three `TransactionRelayed` events, paymaster `0x2258`, status 0, charge 0. No `Open` events. HTLC ids `0x2b3a57f2002b67744eb75a2a4eb3dedb6abde2d662b6f98858fc424e19adf293`, `0x7f6a2442add17a6ba28d2bc4d301450f789e81dc2df2b214d33bc3f8745704db`, `0x572d9d5397b187ed644dd0a0ed9071b9522e9bb7bb12bf23f9cc6de476c1d068`.

P8 POLYGON **2026-09-16 23:27:27 UTC** REVERTED Block 93,930,809

`0x49eb8c3fd19e57b4cec670a41b25c1068078dc175f157304072bc655dc52a139`

From	`0x2258…` (nonce 8)
To	`0x4e59b448…`
Function	CREATE2 redeem contract
Note	Reverted : inside the constructor of `0x8ed97c79…`, the first two redeems succeeded. The third (USDC.e) reverted, and the deployment failed with "contract creation code storage out of gas", so the whole transaction rolled back.

P9 POLYGON **2026-09-16 23:28:34 UTC** Block 93,930,854

`0xb2ca76dfbfe571742b4b66465b777ab1e06988a8632be1631bef9654cc64d169`

From	`0x2258…` (nonce 9)
To	`0x4e59b448…`
Function	CREATE2 of `0x8ed97c79…`, whose constructor calls `redeem` ×3
Token / amount	USDC 26,130.64 from `0x0cFD…`, USDT0 24,332.49 and USDC.e 0.66 from `0xF615…`, all to `0x4b0e8b2c38b0044405e24db481d00be7d5419e04`
Note	Three `Redeem` events with secret `0x…01`.

4.3 Actor A: movement of the stolen funds

P10 POLYGON **2026-09-16 23:30:13 UTC** Block 93,930,920

`0xb1324674bf0006695d934477aabec73cf48307ac6092497ee90aa8fb4142ae81`

From	`0x2258…` (nonce 10)
To	`0x4b0e…9e04`
Function	native transfer
Token / amount	150 POL
Note	Gas for the receiver.

E3 ETHEREUM **2026-09-16 23:30:47 UTC** Block 25,993,337

`0x0ec1815b7e0104ebef71cc0d2f515e206ccf6452562a089160f36702a9079555`

From	`0x2258…` (nonce 1)
To	`0x4b0e…9e04`
Function	native transfer
Token / amount	0.0015 ETH
Note	Gas for the receiver on Ethereum.

P12 POLYGON **2026-09-16 23:34:03 UTC** Block 93,931,073

`0x843dcac4189133a329bfbdc613c00cd2b2e41cdfabf460d65ae989a06c78b81f`

From	`0x4b0e…` (nonce 0)
To	USDT0
Function	`approve(0x6131B5fa…, 2^256-1)`
Note	Approves KyberSwap's MetaAggregationRouterV2.

P13 POLYGON **2026-09-16 23:34:37 UTC** Block 93,931,096

`0x7afc8746937851d5a512b7c2ab6b19c06bba56ffbbf4f55174396cb5b6d086b4`

From	`0x4b0e…` (nonce 1)
To	KyberSwap MetaAggregationRouterV2 `0x6131B5fae19EA4f9D964eAc0408E4408b66337b5`
Function	`swap`
Token / amount	USDT0 24,332.49 in, USDC 24,317.18 out
Note	Through executor `0x8F10B468b06c6FD214B65F87778827F7D113f996` and Uniswap v4 PoolManager `0x6736678280587006015138 3f4bbff9dab53e5cd6`.

P14 POLYGON **2026-09-16 23:35:57 UTC** Block 93,931,149

`0xc76ed1f76e08878d67d89b20fddeb056fd932399f934ff693b95b2f195926326`

From	`0x4b0e…` (nonce 2)
To	LI.FI Permit2Proxy `0x89c6340B1a1f4b25D36cd8B063D49045caF3f818`
Function	`callDiamondWithEIP2612Signature`
Token / amount	USDC 50,447.83
Note	`LiFiTransferStarted`: bridge `eco`, integrator `jumper.exchange`, destination chain 1, receiver `0x4b0e…9e04`. Funds go into Eco Portal `0xec000064576f9c95a8623bc0eff3db6d296ea6df`.

E4 ETHEREUM **2026-09-16 23:36:11 UTC** Block 25,993,364

`0x483f483b70fb1e980f7940f3565577a081e4ac3eab20404c6eeb82ac2bc9e518`

From	`0x80fbefbee7e2bb2ac085f082674dc8c0bccfa135` (Eco solver)
To	`0xec2c96e75b09e29b66bf2ee5c37fa749ef9aa7c7`
Function	Eco fill
Token / amount	USDC 50,447.83 to `0x4b0e…9e04`, via Eco Executor `0x645dADD5Bf354526b68A2befd9A305F7E03b91Ca`
Note	The bridged USDC arrives on Ethereum.

E5 ETHEREUM **2026-09-16 23:37:23 UTC** Block 25,993,370

`0x7ad470f1b167f3fc9d2a2e2c5e9ff97884ccfa697d7491a9c9f8e42248caf984`

From	`0x4b0e…` (nonce 0)
To	USDC (Ethereum) `0xA0b86991c6218b36c1d19D4a2e9Eb0cE3606eB48`
Function	`approve`

E6 ETHEREUM **2026-09-16 23:37:35 UTC** Block 25,993,371

`0x3dceca78faac4d5c4e9bdfce9c3a9063b766077292b78942a95b4a04d8783bc4`

From	`0x4b0e…` (nonce 1)
To	Sky/Maker `DssLitePsm` `0xf6e72Db5454dd049d0788e411b06CfAF16853042`
Function	`sellGem`
Token / amount	USDC 50,447.83 in; DAI 50,447.83 out
Note	USDC→DAI 1:1.

E7 ETHEREUM **2026-10-14 04:20:11 UTC** Block 26,188,308

`0x94a2b5e2f56f9bc2520a3240e707c940df8db3462ae6bbd151b22a95b289c0ba`

From	`0x4b0e…` (nonce 2)
To	`0x52272e33e9c92ac0CDD3fB720877fBE37E223984` (EOA, unknown)
Function	native transfer
Token / amount	0.0001 ETH
Note	Purpose unknown.

E8 ETHEREUM **2026-10-23 03:54:35 UTC** Block 26,252,768

`0xfa0092fc0c7699f2dba65c2a730dd01167688d457631c107e3dbaa58b670aced`

From	`0x4b0e…` (nonce 3)
To	DAI `0x6B175474E89094C44Da98b954EedeAC495271d0F`
Function	`approve`

E9 ETHEREUM **2026-10-23 03:55:11 UTC** Block 26,252,771

`0xffa00d0e5c5051521328cb0bf3d3f13c32608b7b4d26c557ad8f00d0eef95dc2`

From	`0x4b0e…` (nonce 4)
To	RailgunSmartWallet `0xFA7093CDD9EE6932B4eb2c9e1cde7CE00B1FA4b9`
Function	`shield`
Token / amount	DAI 89.17, plus fee 0.22 to Railgun Treasury `0xE8A8B458BcD1Ececc6b6b58F80929b29cCecFF40`
Note	A test shield.

E10 ETHEREUM **2026-10-23 15:17:23 UTC** Block 26,256,168

`0x4929d160fd4cfc7f38a0bf5adb7f95b63372b08460bf616131064fe0e0ee7a5a`

From	`0x53a1f685d66c14962e7ebbc58e5f30b17b0c06df` (broadcaster)
To	RailgunSmartWallet
Function	`transact` (unshield)
Token / amount	DAI 88.95 back to `0x4b0e…`
Note	The test amount returned to the same address. Inference: either a deliberate round trip or a shield that was not accepted. Railgun's off-chain checks were not looked at.

E11 ETHEREUM **2026-10-29 02:35:11 UTC** Block 26,295,447

`0x811014956d3dabe9f29ee8b6eb9c0376952dce65f96ac87b8c13025a61c54c55`

From	`0x7a2ea6ab500c17fd9e11c5d6f26fd4b2c1afd465` (broadcaster)
To	Railgun `RelayAdapt7702` `0xb7E02B99Ae9CD8ae9f995CA1B7d7686cd404Cbc5`
Function	Railgun unshield
Token / amount	0.0029925 ETH to `0x0302b34b58b268591f70f0828720e7fcd2091375`
Note	Funds the address that later takes the DAI (E16).

E12 ETHEREUM **2026-10-29 02:36:59 UTC** Block 26,295,456

`0xa3d288565c6571b0b8140db023c4f1f86b24664fe8408b6444ea8da97f15fb8b`

From	`0x4b0e…` (nonce 5)
To	CREATE
Function	contract `0x2bd4b5079faf4ab27d1c06507925a483a74d1eec`
Note	Role not analysed.

E13 ETHEREUM **2026-10-29 02:36:59 UTC** Block 26,295,456

`0x6ed0c5dbbcf31b4246e9f741d44878fb3b36f5768531892f3e444742940447b5`

From	`0x4b0e…` (nonce 6)
To	CREATE
Function	token "VALE" `0x5aFf13902410A027c52f3bdd2A376C2912257Def` (unverified)
Token / amount	mints 1,000,000 VALE each to 8 addresses
Note	The receipt shows mints to `0xf74c…48c8`, `0xda08…79ae`, `0x9a49…6738`, `0x9455…50fb`, `0x9c5a…4ffe`, `0x6c1c…8b67`, `0xf567…9e2f`, `0xe27f…c174`.

E14 ETHEREUM **2026-10-29 02:36:59 UTC** Block 26,295,456

`0x88e69038cf1e4a28ab9451a14d90faa3515caa4a5b569e52a3598acf8c720582`

From	`0x4b0e…` (nonce 7)
To	DAI
Function	`approve`

E15 ETHEREUM **2026-10-29 02:36:59 UTC** Block 26,295,456 (tx index 119)

`0xecab35bcc7c25e7b8907b1d316b278638d638744e3b001ea161adbd25e70c076`

From	`0x4b0e…` (nonce 8)
To	CREATE `0xd02e6d495e494b123979a972c98c55a6501237da`
Function	constructor: Uniswap v4 `Initialize` of pool `0xe7f83f3b0fabb15d7707285f7022af355e558ee1005877533f7604dd65f12f3d` (VALE/DAI, fee 0, tickSpacing 60, no hook) and 2 × `ModifyLiquidity`
Token / amount	DAI 50,447.38 into PoolManager `0x000000000004444c5dc75cB358380D2e3dE08A90`. Position NFTs 486356 and 486357 to `0x4b0e…`
Note	The stolen DAI becomes pool liquidity.

E16 ETHEREUM **2026-10-29 02:36:59 UTC** Block 26,295,456 (tx index 120)

`0xedda9cd1e825ad97926ccc30a9e7123c521544dc37a3c8abf13f0c6354ce8f96`

From	`0x0302b34b58b268591f70f0828720e7fcd2091375` (nonce 0)
To	CREATE `0x708cd62b88d16e760ec78f3b949067e649f962d6`
Function	constructor swaps in pool `0xe7f8…2f3d`
Token / amount	VALE 1,000,000 in; DAI 50,443.75 out to `0x0302…1375`
Note	The pool is emptied in the very next transaction. Inference: `0x0302` is controlled by A. It was funded from Railgun 1 min 48 s earlier and acted at the next transaction index. Its VALE balance has no incoming `Transfer` event: VALE has exactly 10 `Transfer` events in total, the 8 mints in E13 and the 2 hops in E16. So the token contract credited it without an event, and none of the 8 minted VALE balances was used.

E17 ETHEREUM **2026-10-29 02:36:59 UTC** Block 26,295,456

`0x0a355ab2b920e6140a8e8f1186f7f7b82f7e9bb9329126e6acb456dab80f18e3`

From	`0x0302…` (nonce 1)
To	CREATE `0x31fea00325d88d102f26b238d2aa977f004def81`
Note	Role not analysed.

E18 ETHEREUM **2026-10-29 02:40:23 UTC** Block 26,295,473

`0x0474285b3bde33ad97ea445236ff001e4a67d99514e557f79a4cc7597132b70f`

From	`0x0302…` (nonce 2)
To	DAI
Function	`approve`

E19 ETHEREUM **2026-10-29 02:41:11 UTC** Block 26,295,477

`0x5bac0938daaf014ed904a1224a4abc5b255d0d0a6dc01548a77b9c089f70ddef`

From	`0x0302…` (nonce 3)
To	RailgunSmartWallet
Function	`shield`
Token / amount	DAI 50,317.64, plus fee 126.11 to Railgun Treasury
Note	**End of the traceable path**: the funds enter the Railgun shielded pool.

Balances at the head blocks:

- `0x4b0e…9e04` on Polygon: 0.66 USDC.e (the stolen USDC.e, never moved) and 149.77 POL.
- `0x2258…` on Polygon: 31.54 POL. Its 1 POL relay stake is still in the StakeManager, with no unlock transaction.
- On Ethereum, `0x4b0e…` and `0x0302…` hold 0 DAI and 0 USDC.

Excluded as address poisoning (not attacker actions):

- About 60 "USDC" transfers from `0x4b0e…` to `0x3730882A9ADaA5a812DeE9dA3e5876a8FcfD7341` on Ethereum. They use non-canonical token contracts and imitate the LitePSM pocket `0x37305B1cD40574E4C5Ce33f8e8306Be057fD7341`.
- A `Transfer` event "from `0x2258…`" to `0x4b0E9cB6f835DAB60996620316e722605aa49e04` on Polygon (**X1**, 2026-09-17 23:30:31, tx `0x471c…8881`). It was emitted by a fake "POL" token contract `0x4c436f446fCCf0974E287d2F2247bE969d1F6Bb0`, in a transaction sent by `0x3994871213eab017e6b5cbb11a0d3b7d532119ca`, not by A.
- A 17.1 gwei transfer to `0x2258…` from `0x4B012f8F8807701a7b66cfa88e0635b445a39e04`, a lookalike of the receiver (tx `0x3e4a…c34e`).
- A JPYC 100 transfer to `0x4b0e…` on Polygon from `0xcbf0e9ccb0b1f6a0edc098c156259739798e8b4d` (**X2**, 2026-09-17 21:30:16, tx `0xe6a2…7e72`).

4.4 Actor B

B1 POLYGON **2026-09-17 00:14:15 UTC** Block 93,932,681

`0xff15a47f4f574110a67e543a097862f4f95fc2dba06591e696959604d7092220`

From	`0xfa20…a618` (nonce 7)
To	StakeManager
Function	`setRelayManagerOwner`

B2 POLYGON **2026-09-17 00:14:18 UTC** Block 93,932,683

`0x6d41f99dd71e0d886f94cbf107e5db6dd6e77f934d35ae1f88cbd34cf07f9dd8`

From	`0xfa20…` (nonce 8)
To	StakeManager
Function	`stakeForRelayManager`
Token / amount	1 POL

B3 POLYGON **2026-09-17 00:14:21 UTC** Block 93,932,685

`0x508e9f0d260818118eb587615602c36833f049bbd96deb1aeda67076cd73e989`

From	`0xfa20…` (nonce 9)
To	StakeManager
Function	`authorizeHubByOwner`

B4 POLYGON **2026-09-17 00:14:24 UTC** Block 93,932,687

`0x80686704b7a191753bfc3b947200a8b1eca600ce2949bf96039dfb7483ed55ee`

From	`0xfa20…` (nonce 10)
To	RelayHub
Function	`addRelayWorkers([0xfa20…])`

B5 POLYGON **2026-09-17 00:14:27 UTC** Block 93,932,689

`0xef98c0a0830412016db10d45d22bfdd10990e070a810651969cdf398c0318b58`

From	`0xfa20…` (nonce 11)
To	CREATE
Function	paymaster `0x821e11e98334f99027aff6fef94c6bd6d09faf47`
Note	unverified

B6 POLYGON **2026-09-17 00:14:30 UTC** Block 93,932,691

`0x0658f6353eb86d9e9f073e7556e6a64946a0dec6921dd534a2c9a5bdb518d079`

From	`0xfa20…` (nonce 12)
To	CREATE
Function	paymaster `0xcdc62dd43a0799fe04cb24bfc72e039ac21ca2cb`
Note	unverified

B7 POLYGON **2026-09-17 00:14:33 UTC** Block 93,932,693

`0x5214416dc2db06b85d861008678d29954aea529712ad118c628240c7c5641a61`

From	`0xfa20…` (nonce 13)
To	RelayHub
Function	`depositFor(0x821e…)`
Token / amount	0.05 POL

B8 POLYGON **2026-09-17 00:14:36 UTC** Block 93,932,695

`0x9e05ea6a9acb6333fc2fa5082a100b1129d3661894b05d606993607287d11e31`

From	`0xfa20…` (nonce 14)
To	RelayHub
Function	`depositFor(0xcdc6…)`
Token / amount	0.05 POL

B9 POLYGON **2026-09-17 00:14:39 UTC** Block 93,932,697

`0x5868d4aba1562b7eea3ac9dfc9fa119d413df516c3f6d8383674f22408c73b16`

From	`0xfa20…` (nonce 15)
To	RelayHub
Function	`relayCall` → `0xF615.execute` → `closePrivate` (`refund(0x52df…0fbd7, 0xfa20…, 0)`)
Token / amount	USDT0 27.78, `0xF615…`→`0xfa20…`
Note	The H1 user's refund, stolen.

B10 POLYGON **2026-09-17 00:14:43 UTC** Block 93,932,700

`0x15cc67b04020c4ce9bfd3ab75de1a04cb6fd4793f96b402d484d97d84c868a63`

From	`0xfa20…` (nonce 16)
To	RelayHub
Function	`relayCall` → `0x0cFD.execute` → `closePrivate` (`refund(0x0ba9…c7bf, 0xfa20…, 0)`)
Token / amount	USDC 4.54, `0x0cFD…`→`0xfa20…`
Note	The H2 user's refund, stolen.

B11 POLYGON **2026-11-12 15:20:31 UTC** Block 97,194,250

`0x260b49aaeaab3140ce4302b2c3e42becadfecc6543c71ed5b35e6c277a280e35`

From	`0xfa20…` (nonce 22)
To	LI.FI Permit2Proxy `0x8eEd5E28216C0Bc6F817df40Ae6CA11C2294CC1c`
Function	`callDiamondWithEIP2612Signature`
Token / amount	USDT0 29.78
Note	`LiFiTransferStarted`: bridge `relaydepository`, destination chain 1, receiver `0xfa20…`. This amount mixes the 27.78 USDT0 stolen here with USDT0 from B's other activity.

B's history begins in July 2026 with contract deployments whose constructors collect small USDC.e, USDC and USDT0 balances from various contracts. Inference: B is an opportunistic sweeper that copied A's technique. No on-chain link between A and B was found. B still holds 4.87 USDC on Polygon, which includes the 4.54 USDC from B10.

4.5 Response

POLYGON **2026-09-17 04:01:45 – 16:27:01** Block 93,941,781 – 93,971,592

(legitimate relays continued)

Actor	Fastspot relay (worker `0x750d…`, see H2)
Action	Five more legitimate relayed transfers went through `0x3157…` and `0x98E6…` after the drain. The last was `0xe9f2723b03245f63f295b567f4e0e4293ff02d359851d45f5ec95ae5af5774bc`.

R1 POLYGON **2026-09-17 18:20:39 UTC** Block 93,976,137

`0x62efc65a3a903b6623bbc7b320cceae914425f7c2498b811eed16d00da90c42c`

Actor	owner `0xDB88…9DDA` (nonce 71)
Action	`setRelayHub(0x0)` on `0x3157…`

R2 POLYGON **2026-09-17 18:21:55 UTC** Block 93,976,188

`0xc874e863e98b5ace14b1d5243e7b163f437603468723dd897f8a41622f4d423c`

Actor: owner (nonce 72)

Action: `setRelayHub(0x0)` on `0x98E6…`

W1 GITHUB **2026-09-17 18:25:48 (git author/commit time)**

`nimiq/wallet` commit `22520ca984`

Actor: Wallet team

Action: "config: show a maintenance message for swaps in the Wallet". Mainnet `enabledSwapAssets` changed from `[NIM, BTC, USDC_MATIC, USDT_MATIC]` to `[]`.

R3 POLYGON **2026-09-17 18:26:19 UTC** Block 93,976,364

`0xf42a363a9bebcc266642eaaa8bcd920ae22f098c4b1ff9b568a6189654dbd0cf`

Actor: owner (nonce 73)

Action: `setRelayHub(0x0)` on `0xF615…`

R4 POLYGON **2026-09-17 18:27:19 UTC** Block 93,976,404

`0x22570a382b21efa372bec4dbccc9faf62fafb338900bacefc8d0ccf74db03252`

Actor: owner (nonce 74)

Action: `setRelayHub(0x0)` on `0x0cFD…`

R5 POLYGON **2026-09-17 18:30:54 UTC** Block 93,976,547

`0xc98a49f05011da6c50b560b56d8beb20121e0318a3ef963e51a0dc6a9a667f49`

Actor: owner (nonce 75)

Action: `setRelayHub(0x0)` on `0xfAbB…`

R6 POLYGON **2026-09-17 18:40:01 UTC** Block 93,976,912

`0x6169a24393268d29bf74d9cb8fed7401551a752c9e1c39cb788a9875265a56bd`

Actor: owner (nonce 76)

Action: `setRelayHub(0x0)` on `0x3c87…`

W2 GITHUB **2026-09-17 21:11:11 UTC**

`nimiq/wallet` commit `0b0122ab2f`

Actor: Wallet team

Action: Swap maintenance shown in a WarningModal.

W3 GITHUB **2026-09-17 22:31:35 UTC**

`nimiq/wallet` commit `32b484909f`

Actor: Wallet team

Action: "config,router: put gas abstraction for sending stablecoins under maintenance". Adds `isGasAbstractionUnderMaintenance: true` to mainnet.

W4 GITHUB **2026-09-19 09:16:33 (committed 09:18:45)**

`nimiq/wallet` commit `47fe85945a`

Actor: Wallet team

Action: "config: re-enable NIM-BTC swaps". `enabledSwapAssets: [NIM, BTC]`.

Victim approvals after the exploit: none. `0x24cb…6773` emitted no `Approval` event towards any handler after block 93,930,784, except the allowance decreases inside **P7** itself. Its remaining allowances are listed in section 6. The owner sent no transaction at all after **R6**: its nonce is 77 both right after **R6** (block 93,976,913) and at the head block.

The times of the Wallet commits are their git author and commit timestamps. They say nothing about when the change was deployed to wallet.nimiq.com.

4.6 Public reports (for comparison with the chain)

Date (UTC)	Source	Statement	Matches chain?
2026-09-18 17:52	OKX Orbit post relaying Defimon Alerts, https://www.okx.com/en-us/orbit/insight/87713771195840	About $50,400 stolen on Sept 16. Paymaster plus forwarder flaw, own relay, forged opens, predictable hashlock. "Nimiq had **previously** suspended Gas Abstraction stablecoin trading."	Mechanism matches. "Previously" holds only relative to the post. The first Wallet maintenance commit (2026-09-17 18:25) and the disabling (18:20–18:40 that day) both come after the drain, so nothing was suspended before the exploit.
2026-09-18	Coinfomania, https://coinfomania.com/nimiq-reports-50463-loss-due-to-exploit/ (mirrored at cryptonews.net/news/security/33461163/)	$50,463 lost "due to a vulnerability in its ERC20PermitHTLCHandler"	Names only the native-USDC handler. `ERC20MetaHTLCHandler` `0xF615…` lost USDT0 and USDC.e as well. Says "Nimiq has reported" but cites only @SlowMist_Team.
2026-09-16 entry	SlowMist Hacked, https://hacked.slowmist.io/?c=Polygon	"Single-tx loss was about $50,463"; sha256(1) hashlock; CREATE2 redemption	The drain and the redeem were two transactions (**P7**, **P9**). The neighbouring "$23,000" entry belongs to a different incident.
2026-09-20	DeFiHackLabs PoC `src/test/2026-09/Nimiq_exp.sol` (commit `480eb811928cda63d83bf25a2355dd7e92d5745f`), README entry "20260916 Nimiq … Lost: ~$50,463.79"	Calls `0xb067…` the "Setup tx" and `0xb2ca…` the "Exploit tx". Describes the victim as holding approvals to "two ERC20PermitHTLCHandler instances".	The victim's `transferFrom`s happen in `0xb067…`, not `0xb2ca…`. `0xF615…` is an `ERC20MetaHTLCHandler`.
2026-09-25	Verichains, https://blog.verichains.io/p/nimiq-hack-analysis	26,130.64 USDC + 24,332.49 USDT0 + 0.66 USDC.e. Notes that the `closePrivate` branch could pay out any open HTLC and that user approvals were exposed. Says Nimiq suspended stablecoin trading.	Matches. Verichains does not report that this actually happened (actor B).
2026-10-02	sanbir/evm-hack-registry `2026-09-Nimiq_exp/Nimiq_exp.md` (commit `52d0b80becc6a02b4a0a4abba4c5b3f748dab217`)	Total 50,463.79. PoC deposits 1 POL for its paymaster and uses nonces 0 and 1 for USDT0/USDC.e. Cites @DefimonAlerts (x.com status 2100868892742602793) and @SlowMist_Team (status 2100896541859054006).	Total matches. The real attack made no paymaster deposit and used nonce 0 for all three requests (gas price 0, so no charge). It also calls `0xb067…` "setup".

No source mentions actor B.

SECTION 5

Addresses

Address	Chain(s)	Role	First / last activity seen
0x2258491525c21f334c5a2dc22ce55e55023fc45d		**Actor A** (the main attacker), EOA. On Polygon it is EIP-7702-delegated to 0x0404853a… (code 0xef0100…); it acted as its own relay manager, relay worker and paymaster.	Ethereum: funded 2026-09-16 22:13:35 (E1), last tx 23:30:47 (E3). Polygon: first 23:01:28 (P1), last own tx 23:30:13 (P10). Nonce 11 at the Polygon head, 2 on Ethereum.
0x70e8139d854c9304159844c7eb5474db2877216a		A's first delegate code (unverified, 4,429 bytes)	Deployed in P2
0x0404853a9d3fdd57073f06be08a72401e85ee251		A's second delegate code (unverified, 4,702 bytes), active during P6 – P10	Deployed in P5
0x8ed97c79a95b311e28b91eed6077af1576533fcd		A's CREATE2 HTLC recipient. Its constructor redeemed the three HTLCs; the runtime code is a revert stub.	P9
0x4b0e8b2c38b0044405e24db481d00be7d5419e04		**A's receiver**: redeem target, swapper, bridger, LitePSM user, pool creator	Polygon 2026-09-16 23:28:34 – 23:35:57. Ethereum 23:36:11 – 2026-10-29 02:36:59
0x5aFf13902410A027c52f3bdd2A376C2912257Def		"VALE" token deployed by A's receiver	2026-10-29
0xd02e6d495e494b123979a972c98c55a6501237da		Pool-setup contract deployed by A's receiver	2026-10-29
0x0302b34b58b268591f70f0828720e7fcd2091375		Took the DAI out of the pool and shielded it into Railgun. Inference: controlled by A.	2026-10-29 02:35:11 – 02:41:11
0xfa20e2bd6cc9c41218d9811ffd984ae333fca618		**Actor B** (the second attacker), EOA: relay manager, worker and request sender	First tx 2026-07-28; exploit 2026-09-17 00:14:15 – 00:14:43; last 2026-11-12 15:20:31
0x821e11e98334f99027aff6fef94c6bd6d09faf47, 0xcdc62dd43a0799fe04cb24bfc72e039ac21ca2cb		B's paymasters (unverified, 1,169 bytes each)	2026-09-17 00:14:27 / 00:14:30

Address	Chain(s)	Role	First / last activity seen
`0x24cb173ae221aea93369f34bdcf0ddb35b436773`		**Victim of A**: swap-liquidity EOA, the counterparty of stablecoin swaps in the Nimiq Wallet, operated by Green Ocean, the independent liquidity provider that powers Fastspot's swaps (see section 1). It had approved `0xF615…` for USDC.e (first approval 2023-03-08, block 40,123,746, tx `0xfd35867e90f85165bbede4893d2d006b2aba42556d6e0ce92d40c09d1d6734b5`, 2^256−1) and USDT0 (from block 63,755,548, tx `0x6d3ddbc3d49baccb7feb2fa4158fb5d79945f81bf5de8b4b8703a5d2c647fc73`), and `0x0cFD…` for native USDC (block 54,385,221, tx `0x9abd8adca8abb67e8e92891fc950078ff383f0d9f23a8384497bbb12a91bc599`, 2^256−1).	Active since 2023. Holds 0 USDC, USDT0 and USDC.e and 1,545.25 POL at the head.
`0x9db8c021407755f5537ca90c8a34dcbb0197b1c4`		**Victim of B**: Nimiq Wallet user (private account), refund address of HTLC `0x52df…0fbd7`. Before that, the RelayHub had relayed 327 calls of this address to the Wallet's handlers between blocks 57,381,117 and 75,985,843: 40 HTLC opens, 255 HTLC redeems and 32 transfers, on all four configured transfer and HTLC handlers.	HTLC opened 2025-09-03 (H1)
`0xc45101224affe424c8528304a174ccf0479fa5bc`		**Victim of B**: Nimiq Wallet user (private account), refund address of HTLC `0x0ba9…c7bf`. Before that, the RelayHub had relayed 27 calls of this address to the Wallet's handlers between blocks 42,788,537 and 75,971,924: 5 HTLC opens, 2 HTLC redeems and 20 transfers, on all four configured transfer and HTLC handlers.	HTLC opened 2025-09-02 (H2)
`0xDB88bf6328D4778bd8Ce653ADf51E079b5bF9DDA`		Nimiq's deployer and owner of all handlers (EOA)	2023-02-22 – 2026-09-17 18:40:01 (R6)
`0x750dd91b9948792c0850c83b059e9abaefa243c0`		Relay worker of the legitimate Nimiq path (relayed H1 / H2). Its relay manager registered on the RelayHub with the URL `https://gsn.main.fastspot.io` (`RelayServerRegistered` events). The Nimiq Wallet keeps `.fastspot.io` relays even without recent activity (`src/lib/usdc/OpenGSN.ts` line 274 in the Wallet), and Nimiq Pay prefers them (`src/lib/polygon/config.ts` line 21 in Nimiq Pay). The relay was run by Fastspot. It received plain transfers from the owner on 2024-01-02 and 2024-10-18 (owner nonces 22 and 46).	—

Address	Chain(s)	Role	First / last activity seen
0x3c499c542cEF5E3811e1192ce70d8cC03d5c3359		USDC token contract (native USDC on Polygon)	—
0xc2132D05D31c914a87C6611C10748AEb04B58e8F		USDT0 token contract (the token the Gas Abstraction Update calls USDT)	—
0x2791Bca1f2de4661ED88A30C99A7a9449Aa84174		USDC.e token contract (bridged USDC)	—

SECTION 6

Scope check

6.1 Which handler calls were forged

Every call to a handler's `execute()` must come from the RelayHub (`onlyRelayHub`). The hub calls a forwarder only inside `innerRelayCall()`, which always ends in a `TransactionRelayed` event (success) or a revert that is logged as `TransactionRejectedByPaymaster`.

Method.

1. **All 321,038 RelayHub logs** from block 39,784,000 (before the first configured handler went live) to head block 97,207,231 were examined, plus 1,138 logs from blocks 39,602,000–39,784,000 for the oldest handler.
2. **Events where a Nimiq handler is `to` or `paymaster`.** There are 14,234 `TransactionRelayed` and 401 `TransactionRejectedByPaymaster` events.
   - Five `TransactionRelayed` events name a paymaster other than the handler itself: the three in **P7** and one each in **B9** and **B10**. All five are listed above.
   - All 401 rejections and all other 14,229 relayed calls had `paymaster == to == handler`, the legitimate Nimiq path. That count includes four `RejectedByForwarder` (status 3) results on `0x98E6…`, which moved no tokens.
   - Per handler, legitimate relayed calls were:
     - `0x0cFD…`: 4,340 (blocks 54,412,143 – 93,913,626)
     - `0x3157…`: 3,842 (51,978,134 – 93,971,592)
     - `0xF615…`: 3,645 (40,424,924 – 93,903,391)
     - `0x98E6…`: 2,248 (39,799,413 – 93,963,081)
     - `0xfAbB…`: 154 (54,411,999 – 86,723,553)
     - `0x3c87…`, `0xe449…` and `0x4b76…`: 0
3. **Calls that name a handler only as forwarder.** `TransactionRelayed` does not log the forwarder, and the handlers' `execute()` ignores `request.to`. A forged call could therefore name a Nimiq handler as forwarder while `to` and the paymaster are unrelated. The **P7** trace shows the hub makes no `isTrustedForwarder` check on `request.to`. To cover this:
   - **Exploit window, blocks 93,900,000 – 93,976,912** (2026-09-16 10:37:13 to 2026-09-17 18:40:01, the last `setRelayHub(0)`): there is **no** relayed or rejected event at all other than the Nimiq-handler events above. The window is fully covered.
   - **Full history before the window**: 147,530 relayed or rejected events involve other dapps and 63 other paymasters.
     - Each paymaster's `preRelayedCall` was replayed at its last active block, using a request that names `0xF615…` as forwarder.
     - **47 paymasters** revert with "Forwarder is not trusted" or "preRelayedCall: Unauthorized forwarder". They cannot have accepted a Nimiq forwarder, as long as their code did not change before their last block.
     - **14 paymasters** accepted the test request or reverted for other reasons. `relayData.forwarder` was decoded from the `relayCall` calldata of **all 1,491** of their transactions. Every forwarder is the OpenGSN Forwarder `0xda78a11fd57af7be2edd804840ea7f4c2a38801d` or Enzyme's forwarder `0x14c6b99affc61e9b0753146f3437a223d0c58279`, so **no Nimiq contract appears**.

- **2 paymasters** were not decoded transaction by transaction:
  - GoldFever `0x025de272899de03e4a05cb35df56ffc31201df37` (72,316 events). Its verified `preRelayedCall` requires a target whitelist and a 65-byte `approvalData` signature from its own `signer`.
  - `0x81ffca177e749d619b05a7f3df331fce3c2965bd` (68,886 events, unverified on Blockscout). It reverts with "relay worker is not allowed worker" for foreign workers.
  - A forged call through either would need that dapp operator's cooperation. **Limitation**: the forwarder of these 141,202 calls is unverified.

4. **Exposure of approvers.**
   - **Approvals.** Every `Approval(owner, spender)` event with a Nimiq handler as spender was collected for USDC.e, USDT0 and native USDC: 18,629 events, 2,749 (token, owner, handler) pairs, blocks 39,784,000 – 97,207,231.
   - **Completeness of that list.** A search of all token contracts for approvals that name any of the eight handlers in section 2 as spender, over blocks 39,602,000 – 97,207,231, finds the same 18,629 events. All of them are on USDC.e, USDT0 or native USDC and name one of the five Wallet-configured handlers. **No** approval on any token has ever named `0x3c870b…` (the sixth handler), `0xe4491D…` or `0x4b766a07…` as spender.
   - **Allowances and balances.** For every pair, `allowance` and `balanceOf` were read at three blocks: 93,930,783 (before the drain), 93,976,912 (last `setRelayHub(0)`) and 97,207,231 (head).
   - **Exposure** is min(sum of allowances to all handlers, balance) per owner and token. It is the most a forged `execute()` could have pulled from that owner.

Block	USDC.e exposed (owners)	USDC exposed (owners)	USDT0 exposed (owners)
93,930,783, all owners	7.18 (666)	26,134.25 (1,117)	24,334.51 (450)
93,930,783, excluding `0x24cb…6773`	6.52 (665)	3.61 (1,116)	2.02 (449)
93,976,912 (all owners; `0x24cb…` now has 0 balance)	6.52 (665)	3.62 (1,116)	2.02 (450)
97,207,231 (head)	6.52 (665)	3.61 (1,113)	2.02 (448)

   - **Ordinary users.** Almost every Wallet approver keeps a small residual allowance. The Wallet approves `fee + amount` (`src/ethers.ts` line 1594 in the Wallet) and the charged fee is lower. The largest exposure of any user other than `0x24cb…` was 0.07 USDC.e (`0x3ce6cc16c46353dba3edce1fb261a1993784b2d6`), 0.06 USDC (`0x7845593de87aea70749385cc6d291e71d3e6f4f4`) and 0.04 USDT0 (`0x9fc87168709998502d20f47629c16154710a106a`). The sum over all ordinary approvers was about 12.15 in the three stablecoins together.
   - **Per handler** (pairwise, before deduplication per owner; the approver counts are as of block 93,930,783 and include `0x24cb…`, the exposure leaves it out):

Handler	Approvers	Exposure
`0x98E6…` USDC.e	443	3.99
`0x98E6…` USDT0	365	1.74
`0x3157…` USDC	807	2.58
`0xF615…` USDC.e	259	2.44
`0xF615…` USDT0	168	0.34
`0x0cFD…` USDC	562	1.25
`0xfAbB…` USDC.e	144	0.91

- **Was any of it taken?** None of these residual allowances was used by a forged call. The only forged calls found are the five events in item 2. For the exploit window this is complete. For the time before it, the limitation in item 3 applies (two paymasters not decoded).
- **The victim's allowances.** `0x24cb…6773` still has allowances close to 2^256 to `0xF615…` (USDC.e and USDT0) and `0x0cFD…` (USDC) at the head block, with zero balances. Since **R3** and **R4** these allowances cannot be used through `execute()`.

6.2 Other HTLCs

At block 93,930,783 the two HTLC handlers held exactly two open HTLCs (`deposits` = 27.78 USDT0 and 4.54 USDC), which B took. After **B10**, `deposits` is 0 for all three tokens on both handlers, so no open HTLC remains.

Apart from HTLC deposits, the two handlers hold small token balances of their own, about 31.49 in total:

- `0xF615…`: 18.12 USDC.e and 2.13 USDT0
- `0x0cFD…`: 11.25 USDC

These balances were the same before the drain and at the head block, so the exploit did not touch them. **Inference**: they are accumulated fee balances that only the owner can withdraw (`withdrawToken`).

6.3 Transfer handlers and the USDC.e→USDC conversion

The transfer handlers `0x98E6…`, `0x3157…`, `0x3c87…` and `0x4b76…` and the swap handler `0xfAbB…` show no forged relayed call. The full-history method and its limits are in 6.1. Their `execute()` was just as unprotected (section 3.1), so every account with a non-zero allowance to `0x98E6…`, `0x3157…` or `0xfAbB…` was exposed until **R1**–**R6** (amounts in 6.1 item 4). `0x3c87…` was equally unprotected until **R6** but never received an approval. `0x4b76…` was never disabled, but its `execute()` checks signatures (**inference** from source, section 2) and it has no approvals.

SECTION 7

Appendix

7.1 All transaction hashes

Every transaction hash cited above, in time order. Values other than hashes (HTLC ids, the hashlock, the Uniswap v4 pool id) are not transactions and are not listed.

ID	Chain	Tx hash	Block	UTC	Status	Short description
D2		0x0849a963f3fe2dba3ceb1bd5ce2414684dcca50fe45c099c32fbf239c665fc90	14,119,432	2021-05-05 11:55:33	OK	OpenGSN deployer 0x8C1F…324E deploys StakeManager
D1		0x23f82fcb75cdeea8a82eedb31d77493500a5c7e360b694284c9c220d6c6b8555	14,119,440	2021-05-05 11:55:49	OK	OpenGSN deployer 0x8C1F…324E deploys RelayHub
D3		0x009afe209b525fea0fcfbff2b0e4cded1e6e48503b4e24f8f5e7b8451ba4f9ac	39,602,325	2023-02-22 22:44:12	OK	Owner (nonce 0) deploys 0x4b766a07… (oldest ERC20MetaHandler)
C1		0xc81080a213c815bf2a911416ea1c3a263b96f90fe307aaac0ac9f48d28215227	39,602,450	2023-02-22 22:50:02	OK	Owner (nonce 1): 0x4b766a07… setRelayHub(RelayHub)
C2		0x96c82cd377462455cf68713c0d9125da0b752231ccaffdcc078646bd738d871e	39,603,042	2023-02-22 23:18:15	OK	Owner (nonce 2): 0x4b766a07… setSwapRouter(0x68b3…fc45)
C3		0x9b13c147a652729a294b5ea0a9d00fe7429240732fdfeddd4596875be667c344	39,603,190	2023-02-22 23:24:39	OK	Owner (nonce 3): 0x4b766a07… setWrappedChainToken(WPOL)
C4		0x06adebd153a935a739b2dd2662d970c4594091c91a50dc5125874de63626bba7	39,603,303	2023-02-22 23:31:35	OK	Owner (nonce 4): 0x4b766a07… registerToken(USDC.e, 500)
C5		0x16f4d9b32779addbfaf446d9e7575139e89be5069e31ce01a814067cc69ea376	39,603,364	2023-02-22 23:35:45	OK	Owner (nonce 5): 0x4b766a07… updateRelayGas
C6		0x055779f8e4c90fabc88a94b350f86415032b105f16351fe515d7e5e7fb47ed29	39,603,837	2023-02-22 23:59:05	OK	Owner (nonce 6): RelayHub depositFor(0x4b766a07…), 1 POL
C7		0xefdd8fe56ad8aa6791a6c5a352832b5980728db7f2e7dae4fc0d4a6173611dfb	39,633,376	2023-02-23 21:31:25	OK	Owner (nonce 7): 0x4b766a07… updateRelayGas
D4		0xcf939076fd7df7cbe47be9434843156d30f1be4edfb1138ce6d1d2fbd03f3292	39,784,005	2023-02-27 22:46:42	OK	Owner (nonce 8) deploys 0x98E6… (ERC20MetaHandler)
C9		0x372ed11cfee07652e67348143460599fac44c92edbb84f779a62b9a69eb7e224	39,784,049	2023-02-27 22:48:16	OK	Owner (nonce 9): 0x98E6… setRelayHub(RelayHub)
C8		0xfd495aab6a15c8660c6500a9d02400ec72f5d80c4ccee2449e2e6db31660a327	39,784,450	2023-02-27 23:04:04	OK	Owner (nonce 12): 0x4b766a07… withdrawRelayHubDeposit
D5		0x0f388c1122bb5a94184397f919113bcbbe9ae17e938a215185c04141c01a14e0	40,086,681	2023-03-07 23:15:43	OK	Owner (nonce 14) deploys 0xF615… (ERC20MetaHTLCHandler)
C10		0x0db7f5ea84c5883e52c5daed92088babe398cc82ba009708e08246e2e1d0507d	40,086,697	2023-03-07 23:16:17	OK	Owner (nonce 15): 0xF615… setRelayHub(RelayHub)
V1		0xfd35867e90f85165bbede4893d2d006b2aba42556d6e0ce92d40c09d1d6734b5	40,123,746	2023-03-08 22:56:37	OK	Victim 0x24cb… (nonce 0) approves USDC.e to 0xF615…, 2^256−1
D11		0xd19d05adc8fe2fde345712b13f80a227550bdfcc6151e1245f23446ea13bf64d	48,926,712	2023-10-20 02:25:52	OK	0xe63e…886b creates the Uniswap v3 USDC.e/USDC pool 0xD36e…a418
D6		0x50247ac8d1fa4491744dfb9e3b634983db362be93239f2bd59f451470e6aa14a	51,869,807	2024-01-02 23:09:30	OK	Owner (nonce 26) deploys 0x3157… (ERC20PermitHandler)
C11		0x91388ff08ed66ab2b16540d256137cc5d6f35b5785d8a1c646f7038b7161f399	51,870,017	2024-01-02 23:16:58	OK	Owner (nonce 27): 0x3157… setRelayHub(RelayHub)
D7		0xab827b175e2e4cb407dd402c7f7a51c7d42d995d9c4cefdf33c68767c27f8dac	54,383,444	2024-03-07 19:01:03	OK	Owner (nonce 31) deploys 0x0cFD… (ERC20PermitHTLCHandler)

ID	Chain	Tx hash	Block	UTC	Status	Short description
C12		0xe4c8b7421bd0366eb33192f5544cc76f6282a6478b56968e388ab74512091dc8	54,383,551	2024-03-07 19:04:49	REVERTED	Owner (nonce 32): 0x0cFD… setRelayHub, first attempt
C13		0xda699c7e153729f70556bca71a70aea8c37260df5fbefd4af94047e130c26bc0	54,383,924	2024-03-07 19:18:35	OK	Owner (nonce 33): 0x0cFD… setRelayHub(RelayHub)
D8		0x10a5a2dbfcdec1b398042d2243cd33427f4286b1c4b513fc6d24f8e7a729e5ff	54,384,181	2024-03-07 19:27:41	OK	Owner (nonce 37) deploys 0xfAbB… (ERC20MetaUniswapHandler)
C14		0x9f58e8de8d2148b21ba2b6fd768b1e5f00d9729bd28c0247ee9bd1bc4ff6e8c5	54,384,272	2024-03-07 19:30:55	OK	Owner (nonce 38): 0xfAbB… setRelayHub(RelayHub)
V2		0x9abd8adca8abb67e8e92891fc950078ff383f0d9f23a8384497bbb12a91bc599	54,385,221	2024-03-07 20:05:16	OK	Victim 0x24cb… approves USDC to 0x0cFD…, 2^256−1
V3		0x6d3ddbc3d49baccb7feb2fa4158fb5d79945f81bf5de8b4b8703a5d2c647fc73	63,755,548	2024-11-01 13:27:10	OK	Victim 0x24cb… approves USDT0 to 0xF615…
D9		0x3ce4e668e774eba40f0cc7858e0478eb1eb425d3b8cd03df2847c8c33d70cc01	77,625,843	2025-10-13 11:52:41	OK	Owner (nonce 48) deploys 0xe4491D… (unverified, never hub-enabled)
D10		0xf34d1cc288e1e8826d9a1a78dd747653d06a3561394b21ec3b2e5f7c9b3cfd10	77,626,862	2025-10-13 12:26:39	OK	Owner (nonce 49) deploys 0x3c87… (sixth handler, ERC20MetaHandler)
C15		0xe8c6e2d5c48e466cb89dc556221716f125826c40cc5e3580f6783dc41be79ecd	77,627,182	2025-10-13 12:37:19	OK	Owner (nonce 50): 0x3c87… setRelayHub(RelayHub)
H2		0xbc30531fa287d1ab0968388a0ef1c4f11f8fde93f977b11fe3903332d21f864c	75,971,924	2025-09-02 18:31:42	OK	User HTLC open (USDC 4.54) later stolen by B
H1		0x81c2ef136f6ff98efb78578c14f6881556e55c36bf0ea851610593852dd09674	75,985,843	2025-09-03 02:44:40	OK	User HTLC open (USDT0 27.78) later stolen by B
E1		0xe7bd285ec3b1eaebba1bb56494ef0288e63c0a33dc51226972b2e2c0a1fbc2c6	25,992,951	2026-09-16 22:13:35	OK	Railgun unshield 0.009975 ETH to A
E2		0x033d4870257c0cefd055b829d69c33e1b99ab1bb8849574c73ac7faa8ae0fdb2	25,993,190	2026-09-16 23:01:23	OK	A bridges ETH to Polygon (LI.FI/Across)
P1		0xc350aa9a8ab6608e1f0c2f1dd9dc36bdaf85287c03021712c8061119f2cd2db8	93,929,770	2026-09-16 23:01:28	OK	Across fill: 184.36 POL to A
P2		0xadd5d2fa5d9b88047278ad9b5813fe73c15eb36136970226b5023d4587f9fadf	93,930,034	2026-09-16 23:08:04	OK	A: deploy delegate 1, EIP-7702 delegation
P3		0xd4035e6ce423ca3d2439365554970f371c9c1074f5dae612fa900df10fa58a9f	93,930,052	2026-09-16 23:08:31	REVERTED	A: attempt 1
P4		0xc8066f7052bf748e9ddd8fcad93f2ecda6b68b5f8c0abb7fd00974abd582c140	93,930,073	2026-09-16 23:09:03	REVERTED	A: attempt 2
P5		0xc7cb9bbcb2b38f902576d980488917da09e92875a392744289116dc245bbaec1	93,930,603	2026-09-16 23:22:18	OK	A: deploy delegate 2, re-delegate
P6		0x7f02ce7c3e24e87c2cecb84e8e06169a6fdd1db0b775a65bf0ede39dd7ee0a52	93,930,627	2026-09-16 23:22:54	REVERTED	A: attempt 3
P7		0xb067efae73637f3564f58af7f6027afc497e81e47624b0048636085c678858c0	93,930,784	2026-09-16 23:26:49	OK	**A: relay setup + 3 forged opens (drain)**
P8		0x49eb8c3fd19e57b4cec670a41b25c1068078dc175f157304072bc655dc52a139	93,930,809	2026-09-16 23:27:27	REVERTED	A: redeem attempt
P9		0xb2ca76dfbfe571742b4b66465b777ab1e06988a8632be1631bef9654cc64d169	93,930,854	2026-09-16 23:28:34	OK	**A: CREATE2 redeem of 3 HTLCs**
P10		0xb1324674bf0006695d934477aabec73cf48307ac6092497ee90aa8fb4142ae81	93,930,920	2026-09-16 23:30:13	OK	A: 150 POL to receiver
P11		0x3e4a1ec7aa038abe2435351c1efd10207c85e603246f0e92b7fa7c203bf5c34e	93,930,929	2026-09-16 23:30:27	OK	Address-poisoning dust to A (not an attacker action)
E3		0x0ec1815b7e0104ebef71cc0d2f515e206ccf6452562a089160f36702a9079555	25,993,337	2026-09-16 23:30:47	OK	A: 0.0015 ETH to receiver
P12		0x843dcac4189133a329bfbdc613c00cd2b2e41cdfabf460d65ae989a06c78b81f	93,931,073	2026-09-16 23:34:03	OK	Receiver approves USDT0 to KyberSwap

ID	Chain	Tx hash	Block	UTC	Status	Short description
P13		0x7afc8746937851d5a512b7c2ab6b19c06bba56ffbbf4f55174396cb5b6d086b4	93,931,096	2026-09-16 23:34:37	OK	Swap USDT0 → USDC
P14		0xc76ed1f76e08878d67d89b20fddeb056fd932399f934ff693b95b2f195926326	93,931,149	2026-09-16 23:35:57	OK	Bridge 50,447.83 USDC to Ethereum (LI.FI/Eco)
E4		0x483f483b70fb1e980f7940f3565577a081e4ac3eab20404c6eeb82ac2bc9e518	25,993,364	2026-09-16 23:36:11	OK	Eco fill: USDC arrives
E5		0x7ad470f1b167f3fc9d2a2e2c5e9ff97884ccfa697d7491a9c9f8e42248caf984	25,993,370	2026-09-16 23:37:23	OK	Approve USDC
E6		0x3dceca78faac4d5c4e9bdfce9c3a9063b766077292b78942a95b4a04d8783bc4	25,993,371	2026-09-16 23:37:35	OK	LitePSM USDC → DAI
B1		0xff15a47f4f574110a67e543a097862f4f95fc2dba06591e696959604d7092220	93,932,681	2026-09-17 00:14:15	OK	B: setRelayManagerOwner
B2		0x6d41f99dd71e0d886f94cbf107e5db6dd6e77f934d35ae1f88cbd34cf07f9dd8	93,932,683	2026-09-17 00:14:18	OK	B: stake 1 POL
B3		0x508e9f0d260818118eb587615602c36833f049bbd96deb1aeda67076cd73e989	93,932,685	2026-09-17 00:14:21	OK	B: authorizeHubByOwner
B4		0x80686704b7a191753bfc3b947200a8b1eca600ce2949bf96039dfb7483ed55ee	93,932,687	2026-09-17 00:14:24	OK	B: addRelayWorkers
B5		0xef98c0a0830412016db10d45d22bfdd10990e070a810651969cdf398c0318b58	93,932,689	2026-09-17 00:14:27	OK	B: deploy paymaster 0x821e…
B6		0x0658f6353eb86d9e9f073e7556e6a64946a0dec6921dd534a2c9a5bdb518d079	93,932,691	2026-09-17 00:14:30	OK	B: deploy paymaster 0xcdc6…
B7		0x5214416dc2db06b85d861008678d29954aea529712ad118c628240c7c5641a61	93,932,693	2026-09-17 00:14:33	OK	B: depositFor 0.05 POL
B8		0x9e05ea6a9acb6333fc2fa5082a100b1129d3661894b05d606993607287d11e31	93,932,695	2026-09-17 00:14:36	OK	B: depositFor 0.05 POL
B9		0x5868d4aba1562b7eea3ac9dfc9fa119d413df516c3f6d8383674f22408c73b16	93,932,697	2026-09-17 00:14:39	OK	**B: forged refund, 27.78 USDT0**
B10		0x15cc67b04020c4ce9bfd3ab75de1a04cb6fd4793f96b402d484d97d84c868a63	93,932,700	2026-09-17 00:14:43	OK	**B: forged refund, 4.54 USDC**
—		0xe9f2723b03245f63f295b567f4e0e4293ff02d359851d45f5ec95ae5af5774bc	93,971,592	2026-09-17 16:27:01	OK	Last legitimate Nimiq relayed transfer
R1		0x62efc65a3a903b6623bbc7b320cceae914425f7c2498b811eed16d00da90c42c	93,976,137	2026-09-17 18:20:39	OK	setRelayHub(0) 0x3157…
R2		0xc874e863e98b5ace14b1d5243e7b163f437603468723dd897f8a41622f4d423c	93,976,188	2026-09-17 18:21:55	OK	setRelayHub(0) 0x98E6…
R3		0xf42a363a9bebcc266642eaaa8bcd920ae22f098c4b1ff9b568a6189654dbd0cf	93,976,364	2026-09-17 18:26:19	OK	setRelayHub(0) 0xF615…
R4		0x22570a382b21efa372bec4dbccc9faf62fafb338900bacefc8d0ccf74db03252	93,976,404	2026-09-17 18:27:19	OK	setRelayHub(0) 0x0cFD…
R5		0xc98a49f05011da6c50b560b56d8beb20121e0318a3ef963e51a0dc6a9a667f49	93,976,547	2026-09-17 18:30:54	OK	setRelayHub(0) 0xfAbB…
R6		0x6169a24393268d29bf74d9cb8fed7401551a752c9e1c39cb788a9875265a56bd	93,976,912	2026-09-17 18:40:01	OK	setRelayHub(0) 0x3c87…
X2		0xe6a2165b9f68205242d617dc692b12583d87579be3663e32dfb0b5d7f60f7e72	93,983,722	2026-09-17 21:30:16	OK	JPYC 100 to 0x4b0e… from 0xcbf0…8b4d (not an attacker action)
X1		0x471c70996b1f928dc781a51e6ef7d4ea65b4965547fa192cc8c45e466df48881	93,988,532	2026-09-17 23:30:31	OK	Fake "POL" `Transfer` event naming 0x2258…, sent by 0x3994…19ca (address poisoning)
E7		0x94a2b5e2f56f9bc2520a3240e707c940df8db3462ae6bbd151b22a95b289c0ba	26,188,308	2026-10-14 04:20:11	OK	Receiver: 0.0001 ETH to 0x5227…
E8		0xfa0092fc0c7699f2dba65c2a730dd01167688d457631c107e3dbaa58b670aced	26,252,768	2026-10-23 03:54:35	OK	Approve DAI

ID	Chain	Tx hash	Block	UTC	Status	Short description
E9	◆	0xffa00d0e5c5051521328cb0bf3d3f13c32608b7b4d26c557ad8f00d0eef95dc2	26,252,771	2026-10-23 03:55:11	OK	Railgun shield 89.17 DAI
E10	◆	0x4929d160fd4cfc7f38a0bf5adb7f95b63372b08460bf616131064fe0e0ee7a5a	26,256,168	2026-10-23 15:17:23	OK	Railgun unshield 88.95 DAI back
E11	◆	0x811014956d3dabe9f29ee8b6eb9c0376952dce65f96ac87b8c13025a61c54c55	26,295,447	2026-10-29 02:35:11	OK	Railgun unshield 0.0029925 ETH to 0x0302…
E12	◆	0xa3d288565c6571b0b8140db023c4f1f86b24664fe8408b6444ea8da97f15fb8b	26,295,456	2026-10-29 02:36:59	OK	Receiver deploys 0x2bd4…
E13	◆	0x6ed0c5dbbcf31b4246e9f741d44878fb3b36f5768531892f3e444742940447b5	26,295,456	2026-10-29 02:36:59	OK	Receiver deploys VALE
E14	◆	0x88e69038cf1e4a28ab9451a14d90faa3515caa4a5b569e52a3598acf8c720582	26,295,456	2026-10-29 02:36:59	OK	Approve DAI
E15	◆	0xecab35bcc7c25e7b8907b1d316b278638d638744e3b001ea161adbd25e70c076	26,295,456	2026-10-29 02:36:59	OK	Pool init + 50,447.38 DAI liquidity
E16	◆	0xedda9cd1e825ad97926ccc30a9e7123c521544dc37a3c8abf13f0c6354ce8f96	26,295,456	2026-10-29 02:36:59	OK	0x0302… swaps VALE → 50,443.75 DAI
E17	◆	0x0a355ab2b920e6140a8e8f1186f7f7b82f7e9bb9329126e6acb456dab80f18e3	26,295,456	2026-10-29 02:36:59	OK	0x0302… deploys 0x31fe…
E18	◆	0x0474285b3bde33ad97ea445236ff001e4a67d99514e557f79a4cc7597132b70f	26,295,473	2026-10-29 02:40:23	OK	0x0302… approves DAI
E19	◆	0x5bac0938daaf014ed904a1224a4abc5b255d0d0a6dc01548a77b9c089f70ddef	26,295,477	2026-10-29 02:41:11	OK	0x0302… shields 50,317.64 DAI into Railgun
B11	⬡	0x260b49aaeaab3140ce4302b2c3e42becadfecc6543c71ed5b35e6c277a280e35	97,194,250	2026-11-12 15:20:31	OK	B bridges 29.78 USDT0 (LI.FI/Relay) to Ethereum

Rows D1–D11 and C1–C15 are the deployment and configuration transactions of section 2.