dr laurence e. day / @functi0nZer0; ethcc5

Never Send To Know For Whom The Bell Tolls:
A Walk Through The Wasteland

What We're Talking About

A brief discussion of interesting breaches, hacks and exploits that have taken place since the *last* EthCC

Vectors, vulnerabilities, lessons learned

Will be light on code (I'm sorry but *no one* is reading Solidity at 18:00 on a projector)

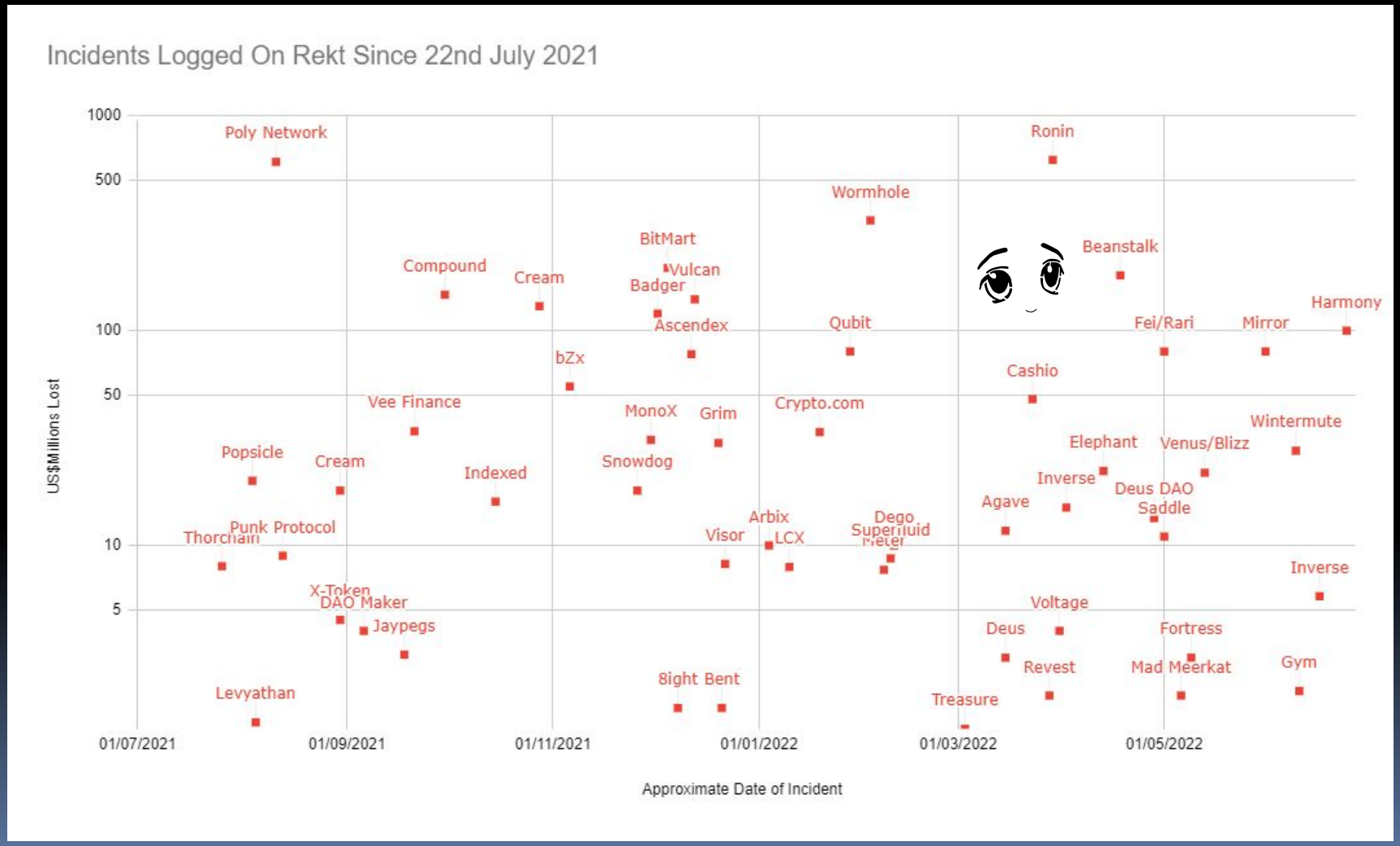
A Note To The Affected

I'm a dev, but I'm not *completely* fluent in the mechanics of all of the incidents presented here: this is a best effort, *brief* recap of noteworthy (non-rug) incidents

This is *not* intended as a pile-on – bugs creep in, mistakes are made or overlooked: this is a *ruthless* arena we work in, and *everyone hit here has my respect*

Take My Hand, Let’s Start

Since We Were Last Here



Some Descriptive Statistics

Number of incidents: 53

Total loss (US$mm): 3,438

Average loss (US$mm): 64.87

Median loss (US$mm): 16

Largest losses: Ronin, Poly, Wormhole
(guess the common theme here)

Poly Network

Root: bruteforced signature hash allowed attacker to update the list of addresses that can sign off on cross-chain txes

```
01;
(bytes4(keccak256(abi.encodePacked(_method, "(bytes,bytes,uint64)"))),
```

```
http://ethers.utils.id ('putCurEpochConPubKeyBytes(bytes)').slice(0, 10)

'0x41973cd9'

http://ethers.utils.id ('f1121318093(bytes,bytes,uint64)').slice(0, 10)

'0x41973cd9'
```

Poly Network

Using EthCrossChainManager to call EthCrossChainData in order to bypass an ownership modifier…

"Hi, I'm updating keepers – sighash match!"

"Hi again, here are some transactions signed by the keeper (me) authorising the release of assets you have held in escrow to the keeper's address"

Poly Network

Loss: US$611mm (33mm realised)

After some bizarre speeches, dumping a bit into Curve, and getting USDT blacklisted, the attacker gave most of it back

Lesson: keep concerns separate – unless there's a *very* good reason to allow contract A to own contract B, *do not do it* – especially in a cross-chain setting

Beanstalk

Root: flashloan enough BEAN to force immediate execution of an emergency governance proposal draining the treasury

Attacker also sent 250,000 BEAN to Ukraine via a separate distraction proposal

Some intermediate steps omitted here (used the BEAN to generate Seeds via BEAN3CRV-f and BEANLUSD-f)

Beanstalk

```
Hacker 0x1c5dcdd006ea78a7e4783f9e6021c32935a10fb4
Hacker Contract 0x79224bc0bf70ec34f0ef56ed8251619499a59def
BIP18  0xe5ecf73603d98a0128f05ed30506ac7a663dbb69

Propose BIP18 tx: 0x68cdec0ac76454c3b0f7af0b8a3895db00adf6daaf3b50a99716858c4fa54c6f
1. Hacker proposes a malicious proposal BIP with initAddress @ 0xe5ecf73603d98a0128f05ed30506ac7a663dbb69

Launch the hack tx: 0xcd314668aaa9bbfebaf1a0bd2b6553d01dd58899c508d4729fa7311dc5d33ad7
1. Flashloan 350,000,000 DAI, 500,000,000 USDC, 150,000,000 USDC, 32,425,202 BEAN, and 11,643,065 LUSD
2. Vyper_contract_bebc.add_liquidity 350,000,000 DAI, 500,000,000 USDC, 150,000,000 USDT to get 979,691,328 3Crv
3. LUSD3CRV-f.exchange to convert 15,000,000 3Crv to 15,251,318 LUSD
4. BEAN3CRV-f.add_liquidity to convert 964,691,328 3Crv to 795,425,740 BEAN3CRV-f
5. BEANLUSD-f.add_liquidity to convert 32,100,950 BEAN and 26,894,383 LUSD and get 58,924,887 BEANLUSD-f
6. Deposit 795,425,740 BEAN3CRV-f and 58,924,887 BEANLUSD-f into Diamond
7. Diamond.vote(bip=18)
8. Diamond.emergencyCommit(bip=18) and hacker proposed _init contract is excuted to get 36,084,584 BEAN and 0.54 UNI-V2_WETH_BEAN,
   874,663,982 BEAN3CRV-f, 60,562,844 BEANLUSD-f to hacker contract
9. BEAN3CRV-f.remove_liquidity_one_coin 874,663,982 BEAN3CRV-f to get 1,007,734,729 3Crv
10. BEANLUSD-f.remove_liquidity_one_coin 60,562,844 BEANLUSD-f to get 28,149,504 LUSD
11. Flashloan back LUSD 11,795,706 and BEAN 32,197,543
12. LUSD3CRV-f.exchange to swap 16,471,404 LUSD to 16,184,690 3Crv
13. Burn 16,184,690 3Crv to get 522,487,380 USDC, 365,758,059 DAI, and 156,732,232 USDT
14. Flashloan back 150,135,000 USDT, 500,450,000 USDC, 350,315,000 DAI
15. Burn UNI-V2_WETH_BEAN 0.54 to get 10,883 WETH and 32,511,085 BEAN
16. Donate 250,000 USDC to Ukraine Crypto Donation
17. swap 15,443,059 DAI to 15,441,256 USDC
18. swap 37,228,637 USDC to 11,822 WETH
19. swap  6,597,232 USDT to  2,124 WETH
20. Profit 24,830 WETH is sent to hacker
```

Image Credit: PeckShield

Beanstalk

Loss: US$181mm

Attacker kept about US$76mm after repaying flash loans, headed into Tornado

Lesson: be careful about governance – allowing proposals to be executed in the same block as they're voted on was an oversight

Mirror

Root: validators on Luna 2.0 chain using out-of-date price oracle set LUNC/LUNA at parity for collateral

Deposit US$1 of LUNC: Mirror thinks it's actually worth approximately US$1,300

Take out as much m{BTC/ETH/DOT/GLXY} as you can against it – it's 99% bad debt

Mirror

mBTC Bitcoin	31.94 UST	31,289.59 UST	-99.89%
mCOIN Coinbase Global, Inc.	95.49 UST	75.26 UST	26.87%
mDIS The Walt Disney Company	133.00 UST	109.49 UST	21.46%
mDOT Polkadot	0.01 UST	10.54 UST	-99.89%
mETH Ether	2.27 UST	1,956.40 UST	-99.88%
mFB Facebook Inc.	242.83 UST	195.60 UST	24.14%
mGLXY Galaxy Digital Holdings Ltd	0.00 UST	6.90 UST	-99.88%

Image Credit: ChainLinkGod

Mirror

Loss: US$2mm

This one's interesting because it isn't actually Mirror's 'fault', but rather that of chain validators being slow on the uptake (massive operations failure)

Lesson: we need to be a *lot* more paranoid about the source of truth coming from oracles, *especially* when forks happen

Indexed

Root: Balancer-style AMM weights (and prices) knocked out of sync when introducing new index assets, constituents purchased when significantly undervalued

Key vectors: (1) estimating pool value via a single benchmark asset, and (2) the ability to settle accounting with unexpected deposits

Attacker identified, lawsuit filed in Canada

Indexed

Manipulate the TotalPoolValue Benchmark			
1	Trigger re-indexing to add SUSHI to DEFI5 index		1--2
2	Add leverage by borrowing $157 million in flash loans	A2	3--8
3	Purchase 98% of the UNI in DEFI5 using $109 million of borrowed tokens, causing the AMM to assign a massively inflated Pool Price to UNI		9--76
4	Exploit the inflated UNI Pool Price by causing the index controller to set a value for the TotalPoolValue benchmark far below the pool's NAV, and thus an inflated Initialization Price for SUSHI		77
5	Use $53 million in UNI ($48 million flash loaned in step 2 + $5.2 million swapped out in step 3) to mint 1.4 million DEFI5 tokens	A3	78--122
Hack the Trade Volume Limit on the Initialization Trade			
6	Circumvent the trade volume limit on the Initialization Trade by making a "gift" of $2.4 million of SUSHI and executing the "Gulp" function, causing the price glitch for the Initialization Price of SUSHI to affect the prices of all other assets		123--126
Minting and Burning DEFI5 Tokens at Deflated Minting Price			
7	Burn 1.4 million of DEFI5 minted in Step 5 for $155 million, including $2.1 million of SUSHI	A4	127--136
8	Use $2.1 million of SUSHI to mint more DEFI5	A5	137--154
9	Burn DEFI5 for $16.9 million, including $2 million of SUSHI	A6	155--164
10	Repeat steps 8-9 with $2.0 million of SUSHI, burning for $3.9 million	A7/A8	166--189
11	Cash out net gain of $12 million	A9	199--207
	Post-Attack Balance ("After")		A10

Image Credit: mf I am the source

Indexed

The Math Prodigy Whose Hack Upended DeFi Won't Give Back His Millions

An 18-year-old graduate student exploited a weakness in Indexed Finance's code, tucking racist epithets into his script, and opened a legal conundrum about theft on the blockchain. Then he disappeared.

By Christopher Beam +Follow

18 May 2022 at 21:01 GMT-7

From **Crypto**

Image Credit: Bloomberg

Indexed

Loss: US$16mm (median!)

Likely to be one of the first instances of Commonwealth case-law, whether attacker gets dragged out of their hole or not

Lesson: gas-saving optimisations can lead to catastrophe, fixing bugs identified in audits can open new faultlines, worth being ruthless about 'accidental' deposits of tokens

Inverse

Root: drive up price of INV on Sushi, force Inverse's platform to read TWAP oracle price, deposit INV as collateral, then borrow

Probably the first hyper-MEV-aware attack – attacker banked *very* hard on being able to hold off searchers and frontrunner bots – 300 ETH was used to manipulate INV price from US$500 to over US$20,000

No flash loan, all funds came from Tornado

Inverse

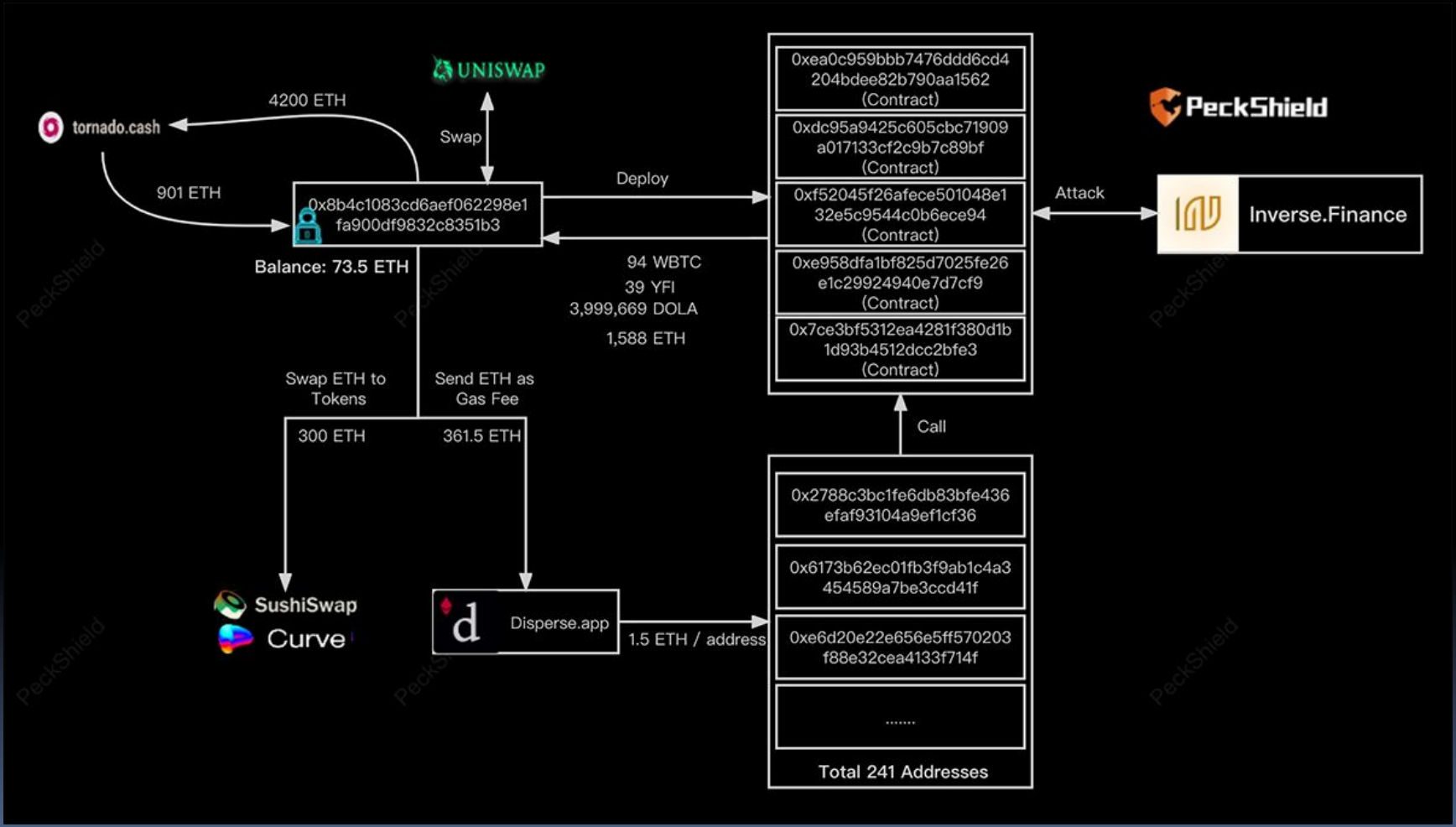


Image Credit: Peckshield

Inverse

Loss: US$15mm

Inverse have stated they want to set up a Chainlink feed to replace the TWAP oracle – but Chainlink have pretty onerous liquidity requirements for a feed, which is what necessitates TWAP oracles in the first place

Lesson: relying on a single thin-liquidity source for an oracle is *Not Good*, but small DeFi projects may have no choice

Wintermute

Root: Gnosis Safe address that Wintermute thought they owned on Optimism had actually been deployed by someone else

Possible because the mainnet Gnosis address was deployed with a version of the ProxyDeployer that used the CREATE opcode instead of CREATE2

Wintermute

Attacker replayed the deployment of ProxyFactory version 1.1.1 on Optimism (reliant only on address and nonce)

Possible because Optimism *didn't* enforce EIP-155 (replay attack protection)

Deployed over 9,000 proxies (), the one deployed at nonce 8,884 matched Wintermute, with the attacker as owner

Wintermute

Loss: US$27mm (1.5mm realised)

Attacker seized control of 20mm OP:
* Sold 1mm OP tokens for 720 ETH
* Sent 1mm to Vitalik (???)
* Sent 17mm *back* to Wintermute
* Has kept the remaining 1mm

Lesson: don't just eyeball addresses – confirm you have *ownership* of Safes you think you control, even if it seems evident

Conclusions

A lot of these attacks are very technically impressive (there's art in the details), but *please* just **claim the whitehat**

Audits aren't infallible: FV helps, but some of these were exogenous in nature

Each incident adds to the almanac of things we 'know' to avoid – whether we *do* avoid them subsequently is a different story

Them’s The Breaks