Twitter Thread by FatMan

FatMan
@FatManTerra

■■ What if I told you that Mirror Protocol, up until 18 days ago, was susceptible to the one of the most profitable exploits of all time, allowing an attacker to generate $4.3m from $10k in a single transaction? Here's how I discovered this - by pure serendipity. ■■

Let's go back to May 9th, when a Mirror contract migration to fix short rewards locked people's funds by accident. We've discussed this before - that's not the point. But take a look at this thread. https://t.co/Qaw91D42dz (1/12)

It appears that OP is indeed correct - Mirror developers smuggled in a major bug fix without announcing it or telling anyone that this bug ever existed, which is slightly infuriating, but what can you do. So how exactly did this bug work? (2/12)

The Mirror Lock contract (that locks your collateral for 14 days when you short) lets you call an unlock function to unlock collateral via a list of position IDs. But they left out something crucial... A duplicate check. This fix was quietly smuggled in 18 days ago. (3/12)

```
- let unlock_amount: u128 = unlockable_positions
-     .iter()
-     .map(|valid_lock_info| {
-         // remove lock record
-         remove_position_lock_info(deps.storage, valid_lock_info.idx);
-         valid_lock_info.locked_amount.u128()
-     })
-     .sum();
```

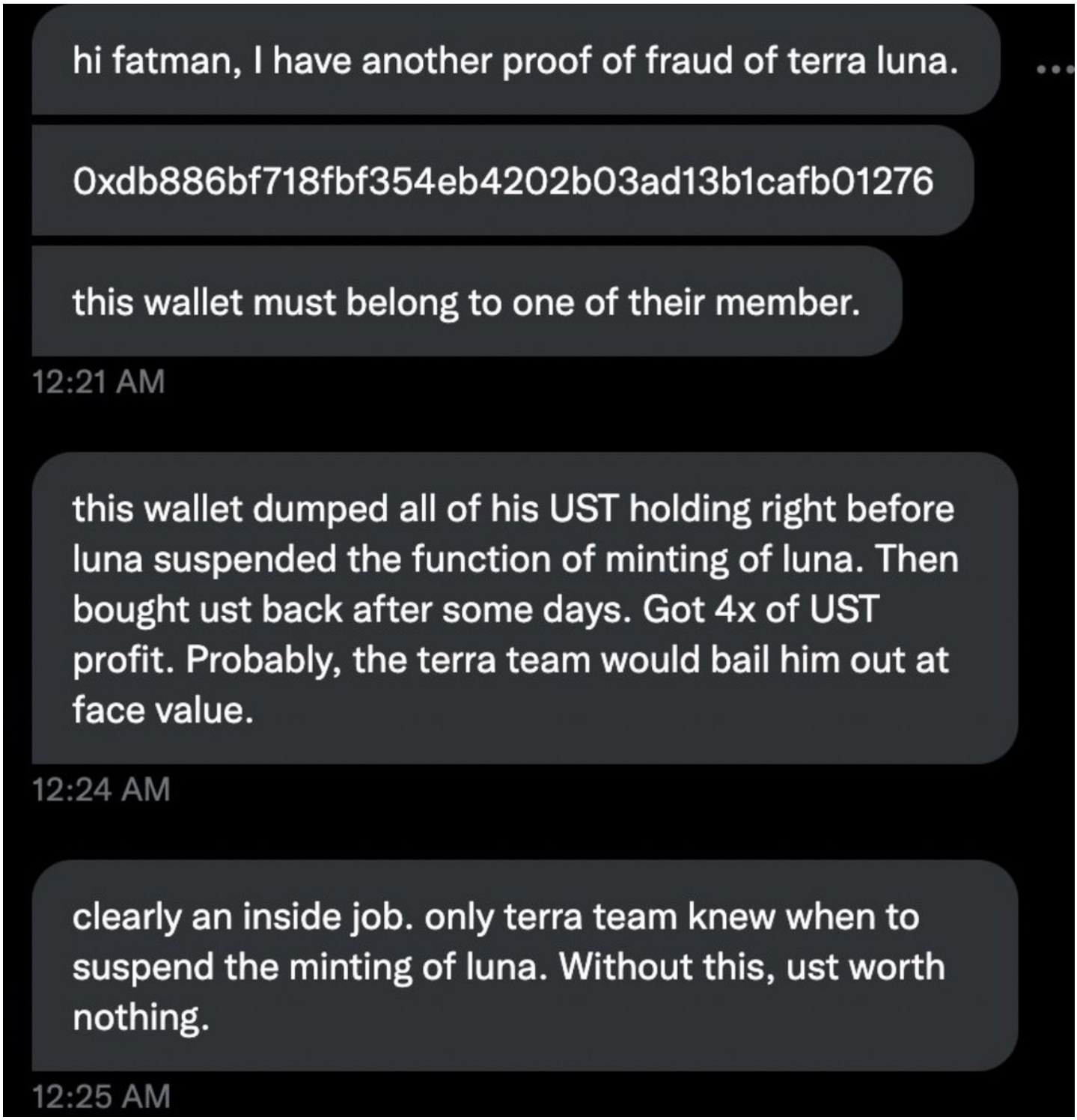
```
+ let mut unlocked_positions: Vec<Uint128> = vec![];
+ let mut unlock_amount = Uint128::zero();
+ for lock_info in unlockable_positions {
+     if unlocked_positions.contains(&lock_info.idx) {
+         return Err(StdError::generic_err("Duplicate position_idx"));
+     }
+     unlocked_positions.push(lock_info.idx);
+
+     // remove lock record
+     remove_position_lock_info(deps.storage, lock_info.idx);
+     unlock_amount = unlock_amount + lock_info.locked_amount
+ }
```

The problem with having no duplicate check is an attacker can create a short position, and after 14 days, they could call their position ID multiple times in a list. This would let them steal funds from the lock contract over and over at little cost and zero risk. (4/12)

So - this bug exists and was quietly patched up - but we don't know if anyone ever noticed it or exploited it before. It would be hard to check since you would need to sift through months of chain data and millions of transactions - the Mirror forum didn't bother. (5/12)

Call it luck, magic, or God's will - whatever you believe in - a source fell into my lap inadvertently revealing that this attack had indeed been executed hundreds of times since 2021. Before today, this was not known by anyone at all. Let's go meet the attacker, shall we? (6/12)

I happened to look at a DM (I can only read a fraction of my DMs!) and almost binned it, but something in me told me to look into the address. The man was right - the address indeed had eerily perfect timing, almost as if they had word directly from TFL. Besides the point. (7/12)



Here is the address for your perusal. https://t.co/7L9aeE38TF I was able to map this address to a Terra wallet via bridge tracing, and it had some large and interesting transactions, so I decided to dig in. Here's the Terra wallet. https://t.co/zAtn6GfVil (8/12)

Two coffees later, as I was about to give up, I found this. Hold on... What's going on here? A single transaction from October 2021 unlocking one position over and over again - and it actually executed. Here's the transaction: https://t.co/2pbiwqKWNT (9/12)



The lock contract didn't check that the funds were sent from the mint contract, so the attacker opened a position with $10 in collateral (!) and send $10k directly to the lock contract. They could then loop-unlock others' collateral over and over again from the contract. (10/12)

In one transaction, the attacker turned $10,000 into $4,300,000. This was actually done several times, generating a total of well over $30m. All of this went completely unnoticed by TFL and the Mirror team & community. This is the first time this attack has been revealed. (11/12)

And that's how with a little bit of luck and a lot of research, I found out about one of the greatest yet most simple smart contract exploits in blockchain history that went under the radar for almost a year. Who did this? I have no idea, but I'll try to find out. (12/12)