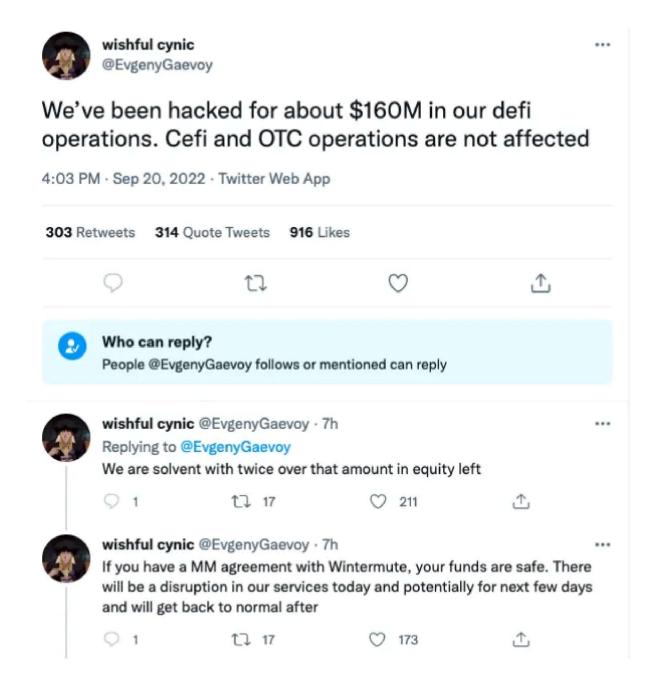
Beosin: \$160 Million Lost in Wintermute's Exploit from Using Profanity



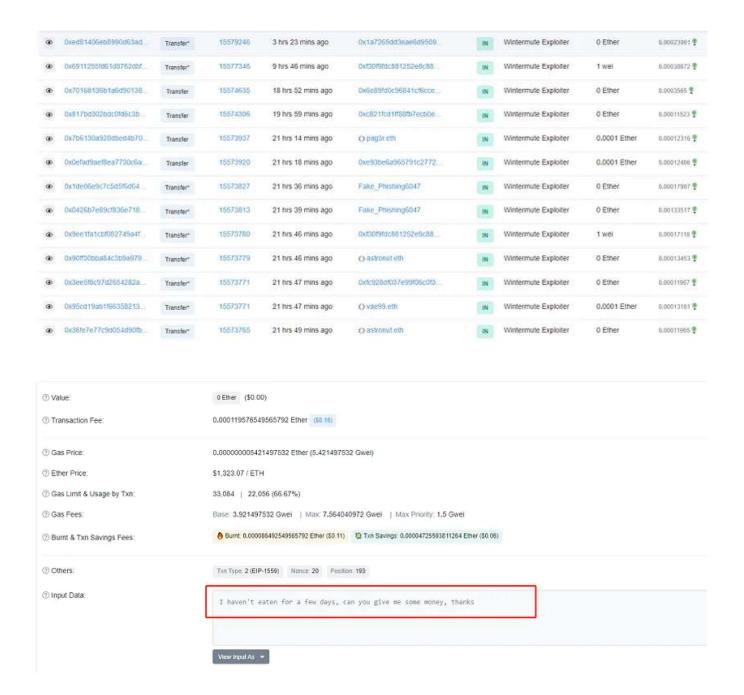
On Sep 20, 2022, Wintermute was exploited for about \$160M due to private key compromise. In June this year, Wintermute has already lost 20M Optimism (OP) tokens by an "internal mistake". Beosin security team analyzed the incident and here's what we found.

1. Incident Background

On Sep 20,2022, Wintermute's founder and CEO Evgeny Gaevoy tweeted that **Wintermute was hacked for about \$160M**.



Up to now, there are still a lot of addresses sending shitcoins or a small amount of Ether to the exploiter begging for some "relief".



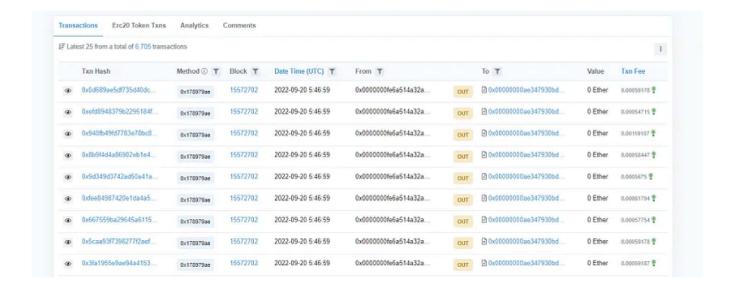
2. Incident Analysis

Attacker's address: 0xe74b28c2eae8679e3ccc3a94d5dode83ccb84705

Attack contract: 0x0248f752802b2cfb4373cc0c3bc3964429385c26

Victim contract: 0x00000000ae347930bd1e7bof35588b9228of9e75

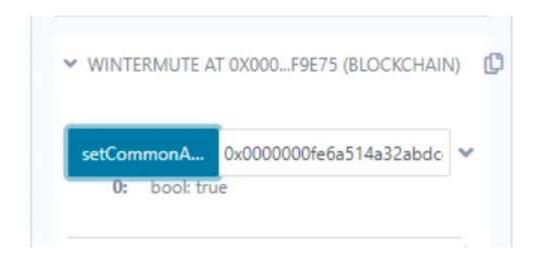
Beosin security team found that the attacker frequently used 0x000000066a... address to call the 0x178979ae function of project's 0x00000000ae34... contract to transfer money to address 0x0248 (the attacker's contract).



The project's contract is not open source. By decompiling the contract, we found that calling the 0x178979ae function requires permission checks.

From the above source code we can see that oxooooooofe6a address has Admin permission, so we deploy an contract to call setCommonAdmin() function for permission query, and we confirm that oxooooooofe6a address has setCommonAdmin permission. The address has normal interaction with the contract before the attack, then we can confirm that oxooooooofe6a's private key is compromised.

Combined with the address characteristics (0x0000000), it is suspected that the project used the Profanity tool to generate the address. The tool has been reported with a risk of brute force cracking of the private key in a previous article by researcher.



Wintermute's founder then confirmed on Twitter that Wintermute did use Profanity to generate addresses in June.

On September 15, <u>a report released by 1 inch Network</u> claimed that Profanity is at the risk of brute force. The Profanity tool uses a 32-bit random vector to generate a 256-bit private key, which may be a security risk in this way.

First, the algorithm of the tool to generate the private key is:

- 1) Profanity will randomly select a 32-bit seed private key in the key space of $2^{**}32 = 4294967296$.
- 2) It is then expanded to 2 million private keys using some deterministic key expansion algorithm.
- 3) Calculate the corresponding derived public key based on the derived private key.
- (4) Repeatedly "increment" the derived public key until the corresponding vanity address is calculated.

Note: The "increment" mentioned in the blog, most people think that if the address derived from the public key in the algorithm does not meet the conditions for a vanity number, each derived private key will be added by 1 and the calculation of steps 2–4 will be repeated.

Secondly, the blogpost mentions a brute-force cracking method to get the initial key.

1) Calculate all the key spaces in advance, i.e. the public keys corresponding to the 4294967296 private keys, and store them in the hash table.

- 2) Obtain the transaction signature from the blockchain explorer and recover the public key from the transaction signature R, S, V values.
- (3) Also extend the public key to 2 million public keys using deterministic key extension algorithm.
- (4) Repeatedly "decrement" the derived public key until the seed public key is obtained.
- 5) Find out the corresponding private key in the hash table according to the seed public key.

Note: Similarly, the "decrement" here, most people think that if the derived public key obtained in the algorithm is different from the seed public key stored in the hash table, then each public key will be decremented by 1 and the operation will be repeated in 3–4 steps.

So many people have some questions about this way mentioned in the blog. First of all, in the private key extension algorithm, since the private key is calculated using secp256k1 elliptic curve algorithm to calculate the public key, the operation is not linear making the public key obtained from two private keys with a value difference of 1 also have a very big difference. Therefore, the same extension algorithm cannot be used to extend the public key in theory.

Second, also due to the elliptic curve algorithm itself, when the private key takes the operation of adding 1 for the iterative calculation of the public key, the inverse operation of adding 1 for the public key, i.e., the operation of subtracting 1, which also cannot effectively improve the calculation speed.

The dev himself answered the above query. The Profanity tool key extension algorithm uses the formula:

Therefore the algorithm has an inverse operation:

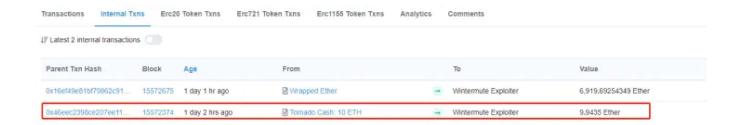
Also, since the key calculation is performed on a finite field, the corresponding "decrement" operation in the report is not a minus 1 operation, but a minus G point.

In summary, there are two problems with the Profanity tool. Firstly, the key space is only 4 billion, which reduces the cost of blasting with the large amount of arithmetic power idle after the Ethereum merge. Secondly, the key expansion algorithm has the problem of inverse operations.

3. Fund Flow

Fund source

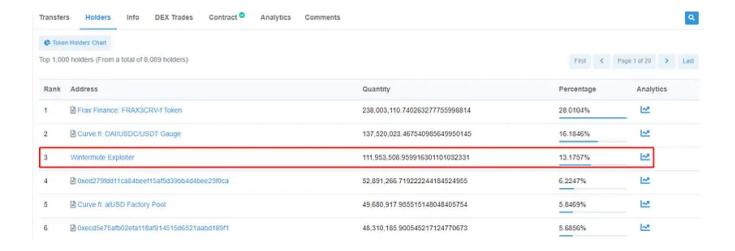
The attacker's funding source was the 10 ETH withdrawn from Tornado Cash 1 day before the attack.



Stolen funds breakdown

ttack address	TokenSymbol	value	Total stolen funds
	WETH	6,919.69	
	DAI	11,026,503.12	
	USDC	61, 350, 986. 34	
	USDT	29, 461, 553. 77	
	TUSD	3, 246, 604. 78	
	BUSD	9, 470, 755. 92	
	USDP	3,972,017.93	
	WBTC	671.25	
	CUBE	1, 789, 602. 46	
	MPL	59, 407. 37	
	eXRD	14, 160, 166. 98	
	SNX	261, 186. 40	
	FTX Token	26,852.69	
	MATIC	820,956.78	
	PRIMATE	30, 248, 063, 86	
	QNT	5, 222. 57	
	MULTI	113,669.18	
	ROUTE	179, 246.00	
	YGG	2, 174, 469. 13	
	CRV	1,019,428.38	
	PLA	1, 393, 842.96	
	NYM	1,673,197.78	
	LCX	8, 495, 859. 49	
	APE	80,966.18	
	REQ	4,922,375.39	
	RADAR	39, 463, 145. 11	
	SHIB	65, 563, 711, 793.06	
	LDO	408, 281. 12	
	AAVE	4,617.16	
	SUSHI	279, 275. 83	
	GODS	701, 554. 43	
	DYDX	429, 795. 70	
	DPI	7,009.58	
	1INCH	414, 495. 78	
	ELON	1, 127, 528, 944, 057. 05	
	CUDOS	36, 276, 385. 41	
	ALI	15, 914, 254. 19	
	ERN	218,005.75	\$159, 417, 832. 71
	FTM	871,944.63	
	CTX	63,500.51	
	ANGLE	4,647,645.54	
	NEXO	203,574.00	
	RAD	82, 110. 43	
	LOOT	388, 208. 78	
	GRT	2,827,278.56	
	PUSH	609, 149. 68	
	LON	268,842.68	
	COMP	3, 283. 18	
	IMX	298, 149. 40	
	SAND	177, 452. 43	
	UNI	26, 720. 31	
	OMG	74,846.71	
	ZRX	417, 169. 67	
	MANA	204,829.52	
	LINK	13,897.07	
	KP3R	930.28	
	CRO	848, 288. 37	
	INDEX	36,857.36	
	SRM	115, 749. 74	
	YFI	10.08	
	SD	226, 615. 86	
	PSP	3,197,983.06	
	BOND	13,952.08	
	REVV	4, 211, 473.00	
	AMP	12, 566, 172. 58	
	RLC	51, 745. 51	
	SKL		
		1,527,805.40	
	REN	547, 177. 29	
	BAT	74, 550. 76	
	BAND	40, 313. 51	
	RNDR	79,088.59	
	GALA	792, 368. 10	
	LRC	99, 448. 28	
	AUDIO	117,661.52	
	MKR	282. 48	

According to Beosin Trace statistics, 75 types of assets are involved in the stolen funds, with a total value of about \$160 million. About 110 million of DAI/USDC/USDT were deposited into the Curve.fi protocol, gaining 110 million LP tokens, ranking the #3 holder in the 3Crv pool.



All the stolen funds are currently held at the attacker's address.

In response to this incident, Beosin security team recommends that:

- 1. Wintermute needs to remove administrative privileges such as setCommonAdmin/owner for 0x0000000fe6a address and other vanity addresses, and replace them with secure wallet addresses.
- 2. Other projects or users who use Profanity tool to generate vanity addresses should transfer assets ASAP.

Contact

If you have need any blockchain security services, please contact us:

Website Email Official Twitter Alert Telegram LinkedIn