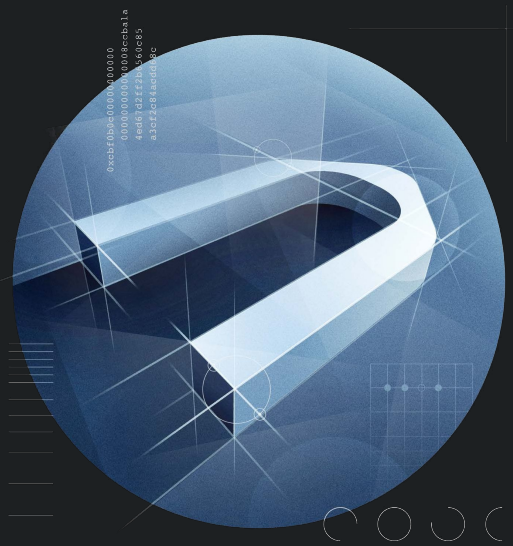
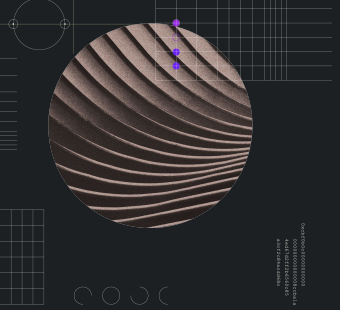


How to Predict the Impact of EVM Upgrades

Tony Valentine
DEDAUB

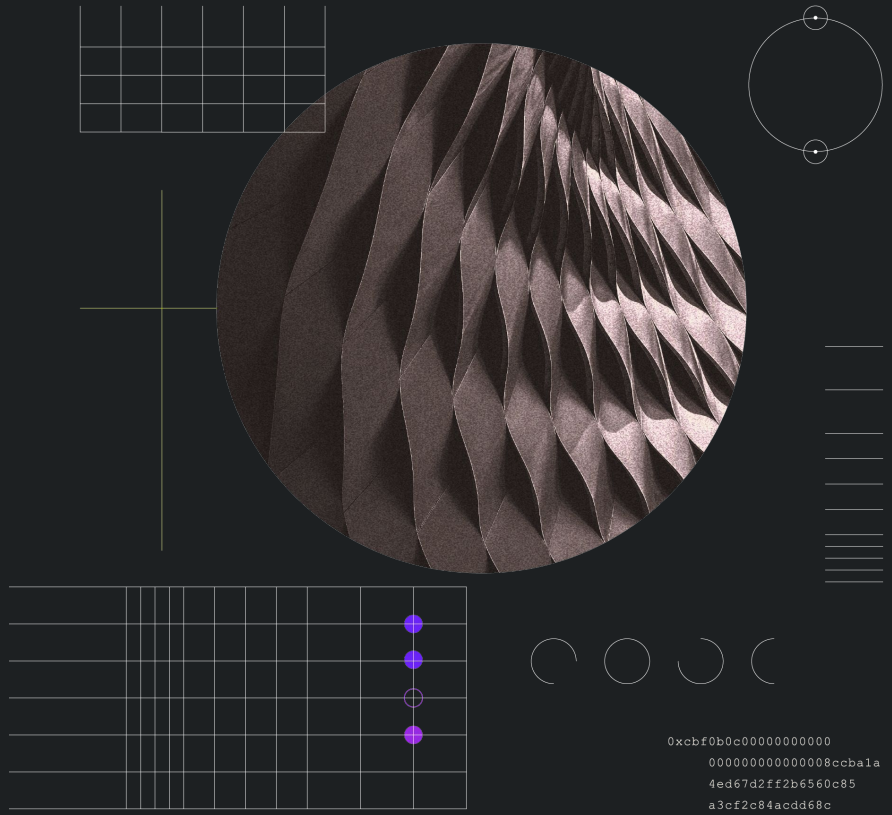


Overview

- Self Destruct Removal Study (EIP 4758/6780)
  - SELFDESTRUCT tl/dr
  - Why it needs to go
  - Analyzing current uses (former now)
  - Verdict
- Transaction/Receipt Root Serialization (EIP 6404/6466)
  - What's changing and why?
  - Finding Contracts performing RLP Decoding
  - Finding RLP blobs in Calldata
  - Verdict

Understanding the Impact of EIP 4758 and 6780

Removing SELFDESTRUCT from the EVM



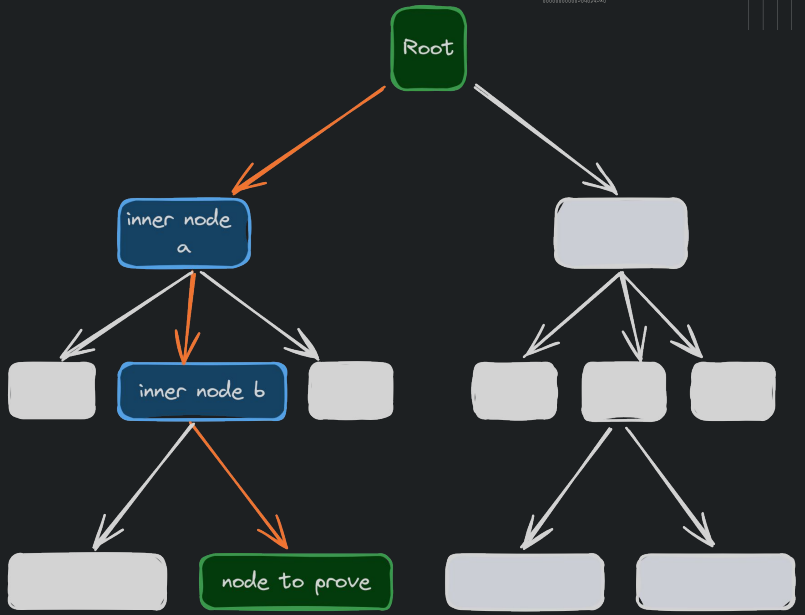
Understanding SELFDESTRUCT

- SELFDESTRUCT clears up a contract
  - resets nonce
  - clears code/storage
  - sends remaining Ether
- Abuses
  - Gas Token
    - Mitigated via EIP-3529
  - Mutable Code
    - EIP-6913 SETCODE (proposal)

Breakdown of EIP 4758 and 6780

- EIP-4758: Deactivating SELFDESTRUCT
  - Renames `SELFDESTRUCT` to `SENDALL`
  - Doesn't touch code or storage or nonce
  - Reduced data transmission and state updates for light clients

- EIP-6780: Restricting SELFDESTRUCT
  - Allows SELFDESTRUCT only within the same transaction as contract creation
  - Reduced impact over EIP-4758



Analyzing Current Uses

- Using Geths/Erigons tracing interface we can hook into the EVM during Execution
- Recorded all contract creations, and selfdestructs from 15m to 17.23m
  - 9.5m contract creations
    - 1m CREATE
    - 8.5m CREATE2
  - 420k SELFDESTRUCT

```
type EVMLogger interface {
    // Transaction level
    CaptureTxStart(gasLimit uint64)
    CaptureTxEnd(restGas uint64)
    // Top call frame
    CaptureStart(env *EVM, from common.Address, ...)
    CaptureEnd(output []byte, gasUsed uint64, ...)
    // Rest of call frames
    CaptureEnter(typ OpCode, from common.Address, ...)
    CaptureExit(output []byte, gasUsed uint64, ...)
    // Opcode level
    CaptureState(pc uint64, op OpCode, gas, ...)
    CaptureFault(pc uint64, op OpCode, gas, ...)
}
```

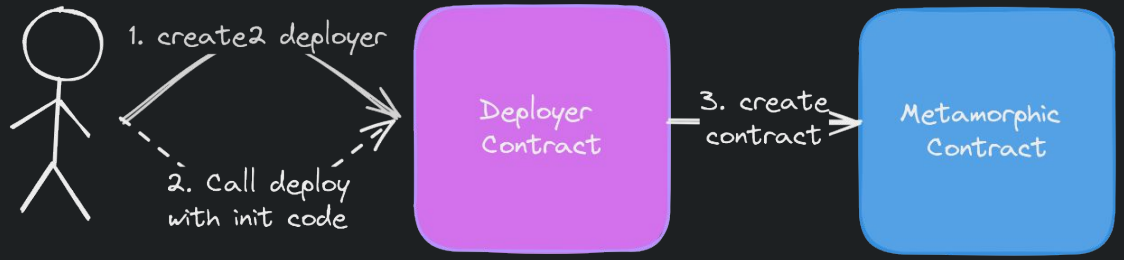
Proxies are for suckers

- CREATE2 + SELFDESTRUCT breaks Code Immutability
  - This is known as metamorphic contracts
  - Root cause of tornado cash governance exploit

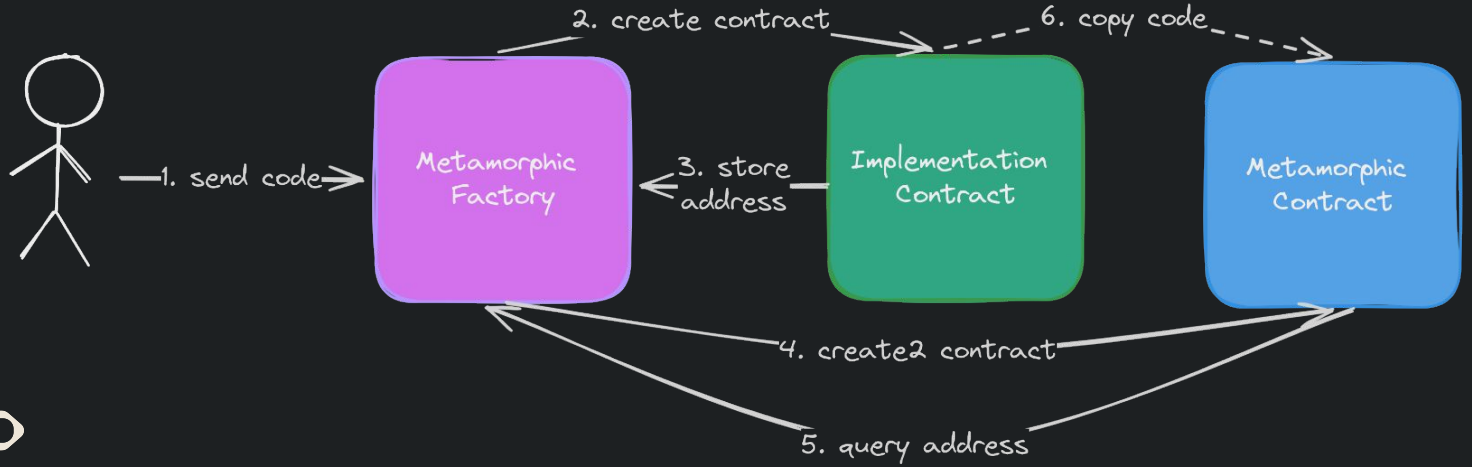
Address	Count
0xBA4A...8ECE	848
0x9C14...BFD6	843
0xA46E...A961	818
0x08BD...2C57	817
0x3E2D...F123E	723

```
def create(sender_address, sender_nonce):
    return keccak256(rlp([sender_address,sender_nonce]))[12:]

def create2(sender_address, salt, init_code):
    return keccak256(0xff + sender_address
                     + salt + keccak256(init_code))[12:]
```

Metamorphic Contract Patterns



"Create3" Pattern

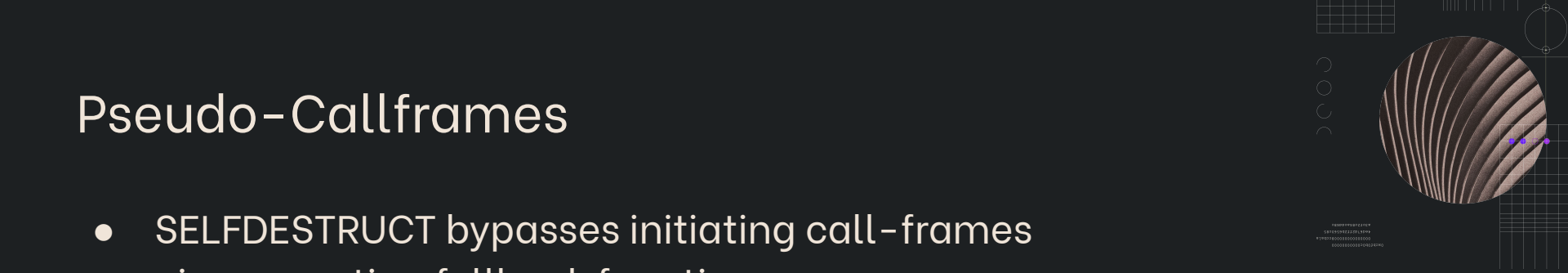


Metamorphic Factory Pattern

Pseudo-Callframes

- SELFDESTRUCT bypasses initiating call-frames circumventing fallback functions

```
contract WETH9 {
    ...
    function() public payable {
        deposit();
    }
    ...
    function totalSupply() public view returns (uint)  {
        return this.balance;
    }
```



I'll make my own `transfer`

```
CALL    [576869]ThreeFMutual.value(0.12518256258538227 ether).buy(_agent = 0x0000000000000000000000000000000000000000, _date = 100)
CALL      [2395]Vat.live()
RETURN      return (1)
S-CALL    [41959]Underwriter.mintShare(_curEth = 300507842912488517534, _newEth = 1943926234321902)
RETURN      return (2000000000000000152)
CREATE    [7720]0x7d01900B17865427192f322195f041A56C5058B8.value(0.012518256258538226 ether).create(0xd41d8cd98f00b204e9800998ecf8427e)
S-DESTR     selfdestruct(beneficiary = 0x9933AD4D38702cdC28C5DB2F421F1F02CF530780, value = 12518256258538226)
CREATE    [7720]0x0D757B4013D9e108E0f0766F00736f3418B11958.value(0.02503651251707645 ether).create(0xd41d8cd98f00b204e9800998ecf8427e)
S-DESTR     selfdestruct(beneficiary = 0x83D0D842e6DB3B020f384a2af11bD14787BEC8E7, value = 25036512517076452)
```

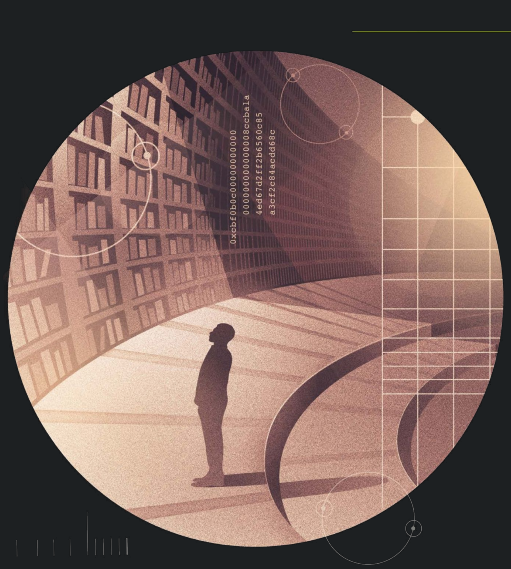
Initialization Code

```
function function_selector() public payable {
    MEM[64] = 128;
    MEM[MEM[64]:MEM[64] + this.code.size - 50] = this.code[50: this.code.size];
    MEM[64] += this.code.size - 50;
    require(this.code.size - 50 ≥ 32);
    selfdestruct(address(MEM[MEM[64]]));
}
```

Impact Breakdown

- 10 protocols deemed affected by EIP-4758
  - 3 high impact
- 0 protocols deemed affected by EIP-6780
- Both proposals fix metamorphic contracts
- EIP-6780 included in Dencun Hardfork (EIP-7569)

Pattern	EIP-4758	EIP-6780
Sending Remaining Ether from Contract	✓	✓
Short Lived Contracts (created now, cleared now)	✗	✓
Long Lived Contracts (created now, cleared later)	✗	✗



Migrating Root Serialization from RLP to SSZ

EIP-6404/6466 Explained

- ETH2 uses a new serialization format Simple SerialiZe (SSZ) instead of Recursive Length Prefix (RLP)
  - Simple, Bijective (only 1 representation), Compact, Merkle-first, Indexable
- EIP-6404: transition `transactions_root` from a Merkle Patricia Trie (MPT) root to SSZ root on Execution Layer
  - This change brings it inline with consensus
- EIP-6466: transition `reciepts_root` from MPT to SSZ
  - Execution Layer and Consensus Layer

RLP (En)/(De)coding

1. Single Byte (`0x00-0x7f`)
   - Direct representation; `0x05` ➜ `0x05`
2. String of Bytes (`0x80-0xb7`)
   - First byte indicates length + data; `0x83 0x63 0x61 0x74` ➜ cat
3. Long String (`0xb8-0xbf`)
   - Length specified after prefix; `0xb8 0x03 0x64 0x6f 0x67` ➜ dog
4. List of Items (`0xc0-0xf7`)
   - First byte indicates total length + items; `0xc8 ... 0x31 0x32` ➜ List(cat, 12)
5. Long List (0xf8-0xff)
   - Length of list specified after prefix;: `0xf8 0x07 ... 0x31 0x32` ➜ List(cat, 12)

```python
def decode_rlp(data):
    if (first_byte := data[0]) ≤ 0x7f:
        return first_byte
    elif first_byte ≤ 0xb7:
        length = first_byte - 0x80
        return data[1 : 1 + length]
    elif first_byte ≤ 0xbf:
        ll = first_byte - 0xb7
        dataLength = int(data[1 : 1 + ll])
        return data[1 + ll : 1 + ll + dataLength]
    elif first_byte ≤ 0xf7:
        length = first_byte - 0xC0
        return decode_list(data[1 : 1 + length])
    else:
        ll = first_byte - 0xf7
        l = int(data[1 : 1 + ll])
        return decode_list(data[1 + ll : 1 + ll + l])
```

```python
def decode_rlp(data):
    if (first_byte := data[0]) ≤ 0x7f:
        return first_byte
    elif first_byte ≤ 0xb7:
        length = first_byte - 0x80
        return data[1 : 1 + length]
    elif first_byte ≤ 0xbf:
        ll = first_byte - 0xb7
        dataLength = int(data[1 : 1 + ll])
        return data[1 + ll : 1 + ll + dataLength]
    elif first_byte ≤ 0xf7:
        length = first_byte - 0xC0
        return decode_list(data[1 : 1 + length])
    else:
        ll = first_byte - 0xf7
        l = int(data[1 : 1 + ll])
        return decode_list(data[1 + ll : 1 + ll + l])
```

Constants Used
in Comparison Operations

```
0x11eb: PUSH1     0xc0
0x11ed: DUP3
0x11ee: LT
0x11ef: ISZERO
0x11f0: PUSH2     0x11fe
0x11f3: JUMPI
```

```
Begin block 0x11df
prev=[0x11ce], succ=[0x11f4, 0x11fe]
================================
0x11e0: v11e0(0x20) = CONST
0x11e3: v11e3 = ADD v11cearg0, v11e0(0x20)
0x11e4: v11e4 = MLOAD v11e3
0x11e6: v11e6 = MLOAD v11e4
0x11e7: v11e7(0x0) = CONST
0x11e9: v11e9 = BYTE v11e7(0x0), v11e6
0x11eb: v11eb(0xc0) = CONST
0x11ee: v11ee = LT v11e9, v11eb(0xc0)
0x11ef: v11ef = ISZERO v11ee
0x11f0: v11f0(0x11fe) = CONST
0x11f3: JUMPI v11f0(0x11fe), v11ef
```

```
Begin block 0x11df
prev=[0x11ce], succ=[0x11f4, 0x11fe]
================================
0x11e0: v11e0(0x20) = CONST
0x11e3: v11e3 = ADD v11cearg0, v11e0(0x20)
0x11e4: v11e4 = MLOAD v11e3
0x11e6: v11e6 = MLOAD v11e4
0x11e7: v11e7(0x0) = CONST
0x11e9: v11e9 = BYTE v11e7(0x0), v11e6
0x11eb: v11eb(0xc0) = CONST
0x11ee: v11ee = LT v11e9, v11eb(0xc0)
0x11ef: v11ef = ISZERO v11ee
0x11f0: v11f0(0x11fe) = CONST
0x11f3: JUMPI v11f0(0x11fe), v11ef
```

```
select md5_bytecode
from decompiled_code
where tac_level similar to '[variable] = [variable] < 0x80'
  and tac_level similar to '[variable] = [variable] < 0xb8'
  and tac_level similar to '[variable] = [variable] < 0xc0'
  and tac_level similar to '[variable] = [variable] < 0xf8'
```

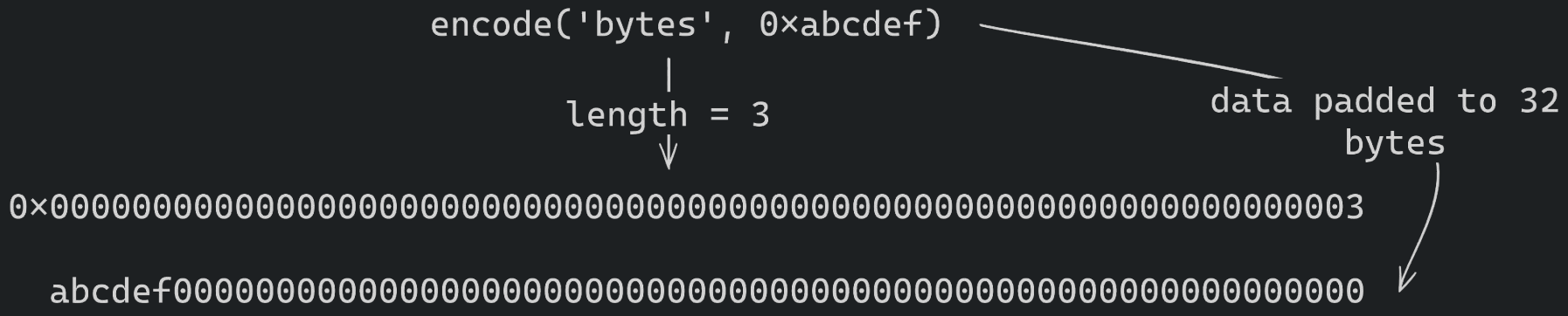
We look for comparison operations in the TAC

The result of the comparison operation flows into the JUMPI, which dictates control flow

Solidity ABI Encoding Overview

- Fixed-Size Types
  - Encoded as 32 bytes. Right-padded with zeros.
  - Example: uint256, address, booleans
- Dynamic-Size Types
  - Offset to data location, followed by length and actual data.
  - Example: string, bytes
- Arrays
  - Fixed-Size: Sequential encoding.
  - Dynamic-Size: Offset, length, and sequential elements.
- Structs/Tuple
  - Fields encoded in order, respecting each field’s type.
- Enums
  - Encoded as underlying integer, following fixed-size rules

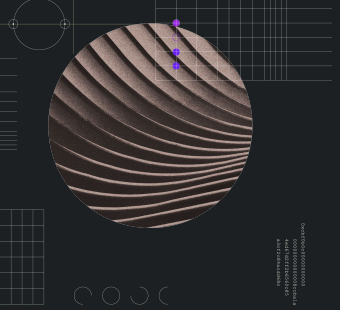
Encoding Bytes Objects



Dynamically Detecting RLP Blobs

```
0000000000000000000000000000000000000000000000000000000000002540

0000000000000000000000000000000000000000000000000000000000000000

                                ...

x    0000000000000000000000000000000000000000000000000000000000000217
y    f90214a0a7004b14f447caaa621e840a1270e6ec6f89b45312198af3ca176eaa
```

```
                                  217          f9
if is_valid_uint64(x) and y[0] > 0xf7:
    ll = y[0] - 0xf7
    l = uint64(y[1:1+ll])

                          217      214    2
if uint64(x) == l + ll + 1:
    return true
```



Impact Breakdown

- Most bridge applications on Ethereum were unaffected
  - RLP proofs were done from L2s to L1s
- 3 Bridges were moderately affected
  - LayerZero
  - zkBridge
  - Telepathy
- EIP-6466 (receipts) had a higher impact than EIP-6404 (transactions root)

Scan the QR code for slides and resources