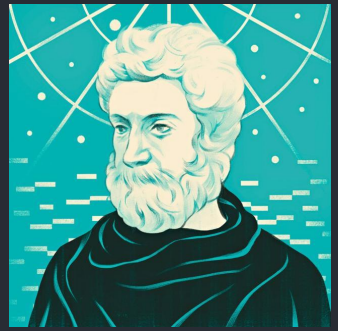
Mastering Ethereum CTFs

CONSENSYS
Diligence



Daniel Luca
@CleanUnicorn

Smart Contract Auditor @ ConsenSys Diligence

John Mardlin
@maurelian_

Co-founder @ ConsenSys Diligence

CONSENSYS
Diligence

Mastering Ethereum CTFs

- Let’s Play, Name That Vulnerability
  - Identify the vuln type
  - We’ll briefly explain the exploit
- Tips and tricks for faster CTF hacking
- Overall mindset and strategy

Let’s Play,

‘Name That Vulnerability’

ROUND 1

```
pragma solidity ^0.4.24;
contract DonationChallenge {
    struct Donation {
        uint256 timestamp;
        uint256 etherAmount;
    }
    Donation[] public donations;
    address public owner;

    function withdraw() public payable {
        require(msg.sender == owner);
        owner.transfer(this.balance);
    }

    function donate(uint256 etherAmount) public payable {
        Donation d;
        d.timestamp = now;
        d.etherAmount = etherAmount;
        donations.push(d);
    }
}
```

What's wrong with this version?

Was this initialized?

SWC-109: Uninitialized Storage Pointer

```
pragma solidity ^0.4.24;
contract DonationChallenge {
    struct Donation {
        uint256 timestamp;
        uint256 etherAmount;
    }
    Donation[] public donations;
    address public owner;

    function withdraw() public payable {
        require(msg.sender == owner);
        owner.transfer(this.balance);
    }

    function donate(uint256 etherAmount) public payable {
        Donation d;
        d.timestamp = now;
        d.etherAmount = etherAmount;
        donations.push(d);
    }
}
```

overwrite the owner with etherAmount

ROUND 2

```
pragma solidity ^0.4.21;
contract SecureWallet {
    address owner;

    function SecureWallet() public {
        owner = msg.sender;
    }

    function withdraw() public {
        require(msg.sender == owner);
        msg.sender.transfer(this.balance);
    }

    function () public payable {}
}
```

What is wrong with *this* version?

RU 1337?

Spell check

```
pragma solidity ^0.4.21;
contract SecureWallet {
    address owner;

    function SecureWallet() public {
        owner = msg.sender;
    }

    function withdraw() public {
        require(msg.sender == owner);
        msg.sender.transfer(this.balance);
    }

    function () public payable {}
}
```

Vulnerability:
Before 0.4.22, the constructor was just defined by having the same name as the contract.

Exploit:

1. call SecureWallet()
2. profit

SWC-118: Incorrect Constructor Name

ROUND 3

```
pragma solidity ^0.5.0;

contract GuessTheNumberChallenge {
    uint256 answer = uint256(blockhash(block.number - 1)) % 65535;

    constructor() public payable {}

    function guess(uint256 n) public payable {
        require(msg.value == 1 ether);

        if (n == answer) {
            msg.sender.transfer(address(this).balance);
        }
    }
}
```

Secret value

```
pragma solidity ^0.5.0;

contract GuessTheNumberChallenge {
    uint256 answer = uint256(blockhash(block.number - 1)) % 65535;

    constructor() public payable {}

    function guess(uint256 n) public payable {
        require(msg.value == 1 ether);

        if (n == answer) {
            msg.sender.transfer(address(this).balance);
        }
    }
}
```

Solution #1

Compute the answer

Solution #2

Use `web3.eth.getStorageAt()`

- Remix REPL
- Python script
- Theo
- Web3
- geth

SWC-120: Weak Sources of Randomness from Chain Attributes

Metamask

Environment: Injected Web3

Ropsten (3) network

Account: 0xc73...ef246 (3.614

Gas limit: 3000000

Value: 0 wei

GuessTheNumberChallenge - brows

Deploy

or

At Address 0x9A1237379d98EC9aF6A92cc18C89

Transactions recorded: 0

Deployed Contracts

GuessTheNumberChallenge at 0x9A1...52

```
pragma solidity ^0.5.0;

contract GuessTheNumberChallenge {
    uint256 answer;

    constructor() public payable {
        answer = uint256(1234);
    }

    function guess(uint256 n) public payable {
        require(msg.value == 1 ether);

        if (n == answer) {
            msg.sender.transfer(address(this).balance);
        }
    }
}
```

0 listen on network Search with transaction hash or address

```
> web3.eth.getStorageAt("0x9A1237379d98EC9aF6A92cc18C89236D79D5206c", 0x0).then(console.log)

{
  "isFulfilled": false,
  "isRejected": false
}

0x0000000000000000000000000000000000000000000000000000000000000004d2

>
```

Use web3

Get real value

ROUND 4

```
pragma solidity ^0.5.0;

contract GuessTheNewNumberChallenge {
    constructor() public payable {}

    function guess(uint8 n) public payable {
        require(msg.value == 1 ether);
        uint8 answer = uint8(keccak256(block.blockhash(block.number - 1), now));

        if (n == answer) {
            msg.sender.transfer(address(this).balance);
        }
    }
}
```

Hash it out!

Predict the future?

Do it live!

```
pragma solidity ^0.5.0;

contract GuessTheNewNumberChallenge {
    constructor() public payable {}

    function guess(uint8 n) public payable {
        require(msg.value == 1 ether);
        uint8 answer =
uint8(keccak256(block.blockhash(block.number - 1), now));

        if (n == answer) {
            msg.sender.transfer(address(this).balance);
        }
    }
}
```

1. Copy and paste this line of code into your own contact.

2. Calculate `n`, and call `GuessTheNewNumberChallenge::guess(n)`.

SWC-120: Weak Sources of Randomness from Chain Attributes

ROUND 5

```
contract PredictTheFutureChallenge {
    address guesser; uint8 guess; uint256 settlementBlockNumber;

    function lockInGuess(uint8 n) public payable {
        require(guesser == 0); require(msg.value == 1 ether);
        guesser = msg.sender; guess = n;
        settlementBlockNumber = block.number + 1;
    }

    function settle() public {
        require(msg.sender == guesser);
        require(block.number > settlementBlockNumber);
        uint8 answer = uint8(keccak256(block.blockhash(settlementBlockNumber), now)) % 10;
        guesser = 0;
        if (guess == answer) {
            msg.sender.transfer(this.balance);
        }
    }
}
```

Future block

Blockhash again

```
contract PredictTheFutureChallenge {
    address guesser; uint8 guess; uint256 settlementBlockNumber;

    function lockInGuess(uint8 n) public payable {
        require(guesser == 0); require(msg.value == 1 ether);
        guesser = msg.sender; guess = n;
        settlementBlockNumber = block.number + 1;
    }

    function settle() public {
        require(msg.sender == guesser);
        require(block.number > settlementBlockNumber);
        uint8 answer = uint8(keccak256(block.blockhash(settlementBlockNumber), now)) % 10;
        guesser = 0;
        if (guess == answer) {
            msg.sender.transfer(this.balance);
        }
    }
}
```

1. Wait 256 blocks
2. EVM does not know older block hashes
3. Use 0 as the hash

ROUND 6

```
pragma solidity ^0.4.18;

contract Charity {
    using SafeMath for uint256;
    mapping(address => uint) public balances;

    function donate(address _to) public payable {
        balances[_to] = balances[_to].add(msg.value);
    }

    function withdraw(uint _amount) public {
        if(balances[msg.sender] >= _amount) {
            if(msg.sender.call.value(_amount)()) {
                _amount;
            }
            balances[msg.sender] -= _amount;
        }
    }
}
```

Come again?

Wait wait, DAOn't tell me…

Who ya gonna call?

```
pragma solidity ^0.4.18;

contract Charity {
    using SafeMath for uint256;
    mapping(address => uint) public balances;

    function donate(address _to) public payable {
        balances[_to] = balances[_to].add(msg.value);
    }

    function withdraw(uint _amount) public {
        if(balances[msg.sender] >= _amount) {
            if(msg.sender.call.value(_amount)()) {
                _amount;
            }
            balances[msg.sender] -= _amount;
        }
    }
}
```

Come again?

Who ya gonna call?

Wait wait, DAOn't tell me…

SWC-107: Reentrancy

ROUND 7

```
pragma solidity ^0.5.0;

contract CastleKing {
    address public king;

    function gainControl() public {
        require(tx.origin != msg.sender, "Nope");
        king = msg.sender;
    }

    modifier onlyKing() {
        require(msg.sender == king);
        _;
    }
}
```

Use proxy contract

1. Deploy contract
2. Ask contract to call gainControl()
3. Profit!

```
contract Solution1 {
    function becomeKing(address _king) public {
        CastleKing(_king).gainControl();
    }
}
```

ROUND 8

```
pragma solidity ^0.5.0;

contract CastleKing {
    address public king;

    modifier enemyCheck() {
        uint256 size;
        address _local = msg.sender;
        assembly{ size := extcodesize(_local) }
        require(size == 0);
        _;
    }

    function gainControl() public enemyCheck {
        require(tx.origin != msg.sender, "Try again");
        king = msg.sender;
    }

    modifier onlyKing() { require(msg.sender == king); _; }
}
```

Checks if it's a contract.

Use proxy contract

1. Deploy contract and gain control in constructor
2. Profit!

```
contract Solution2 {
    constructor(address _king) public {
        CastleKing(_king).gainControl();
    }
}
```

1. Deploy contract and gain control in constructor
2. Profit!

```
contract Solution3 {
    constructor(address _king) public {
        CastleKing(_king).gainControl();
        require(CastleKing(_king).king() != address(0x0), "New king");
    }
}
```

Revert unless winning condition is not met.

ROUND 9

```
pragma solidity ^0.4.18;

contract Token {
    mapping(address => uint) balances;
    uint public totalSupply;

    function Token(uint _initialSupply) public {
        balances[msg.sender] = totalSupply = _initialSupply;
    }

    function transfer(address _to, uint _value) public returns (bool) {
        require(balances[msg.sender] - _value >= 0);
        balances[msg.sender] -= _value;
        balances[_to] += _value;
        return true;
    }
}
```

Go with the flow

Don't underestimate the danger.

You int gonna believe the one weird trick for minting infinite tokens!!!!

```
pragma solidity ^0.4.18;

contract Token {
    mapping(address => uint) balances;
    uint public totalSupply;

    function Token(uint _initialSupply) public {
        balances[msg.sender] = totalSupply = _initialSupply;
    }

    function transfer(address _to, uint _value) public returns (bool) {
        require(balances[msg.sender] - _value >= 0);
        balances[msg.sender] -= _value;
        balances[_to] += _value;
        return true;
    }
}
```

Vulnerability:
Integer Underflow (SWC-101)

Exploit:
Call transfer with `_value` greater than your balance.
It In the EVM:

1 - 2 = 2^256-1

Tips and tricks

Tips and Tricks

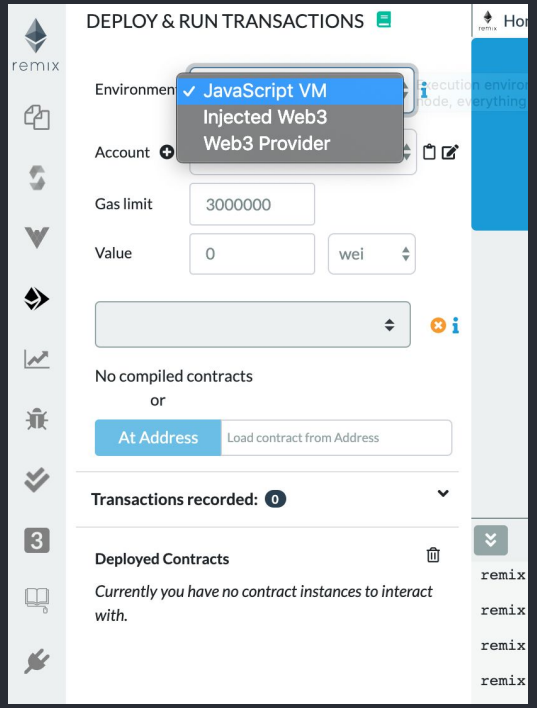
- Use a js vm to prototype
- Set up and run things in the constructor
- Add victory constraints in the constructor
- Read private variables with web3.eth.getStorageAt
- Obfuscating solutions…
- Google challenge type ie. (reentrancy ctfs)
- Reentrancy with Token.balance(victim)
  - (instead of using a counter/off by one errors)
- Use reversing libs

ToDo John:

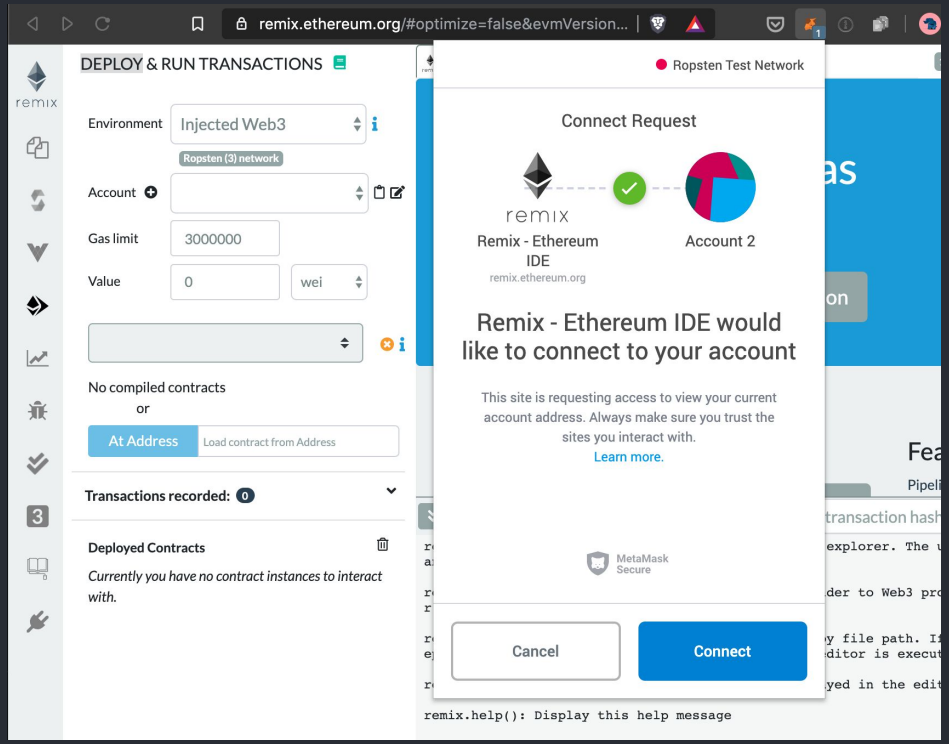
- Add more tips and tricks to the list
- Expand into their own slides

Tips and Tricks

Build in JS VM



Break with MM



Tips and Tricks

No Source Code? No problem.

```
6060604090815260028054 60a060020a60ff0219169055610 1c09051908101604090815 26
03c8252607860208301526101 2c908201526102586060820152610708608082015261 0e10
60a0820152611c2060c08201526138406 0e08201526170806101008201526 1e1006101208
2015262015180610140820152620 2a3006101608201526205460061018082015262093a80
6101a0820152620000a790600390600e620004e4565b50600f60055566071afd498d00006
00e553415620000c457600080fd5b600280546000805460016 0a060020a033316600160a0
60020a031991821681178355 60a060020a60ff0219909316740100000000000000000000
000000000000000000017169091179091556200012f90808060001981640100000006200
28f06200013682021704565b5062000649565b60008062000143620005875 65b600063fff
fffff8916891462000158576000 80fd5b63ffffffff881688146200016b57600080fd5b61
ffff871687146200017c57600080fd5b60028704925060 0d8361ffff1611156200019557 6
00d92505b6101006040519081016040908152878252600160406 0020a0342166020830152
6000908201819052 63ffffffff808c1660608401528a1660808301526 0a082015261ffff8
0851660c0830152881660e08201526006805491935060019180830162000201838262000 5
cb565b600092835260209092208591600202018151815560208201516 0
```

Tips

No

Eveem 0x06012c8cf97BEaD5deAe237070F9587f8E7A266c show see random

Code

Json

Asm

Abi

Data

Meta

```
#
#  Eveem.org 26 Apr 2019
#  Decompiled source of 0x06012c8cf97BEaD5deAe237070F9587f8E7A266d
#
#  Let's make the world open source
#

const name = 'CryptoKitties'
const symbol = 'CK'
const GEN0_STARTING_PRICE = 10^16
const GEN0_AUCTION_DURATION = (24 * 3600)
const GEN0_CREATION_LIMIT = 45000
const PROMO_CREATION_LIMIT = 5000

def storage:
  ceoAddress is addr at storage 0
  cfoAddress is addr at storage 1
  cooAddress is addr at storage 2
  paused is uint8 at storage 2 offset 160
  cooldowns is array of uint32 at storage 3
  secondsPerBlock is uint256 at storage 5
  kitty is array of struct at storage 6
  kittyIndexToOwner is mapping of addr at storage 7
  balanceOf is mapping of uint256 at storage 8
  kittyIndexToApproved is mapping of addr at storage 9
  sireAllowedTo is mapping of addr at storage 10
  saleAuctionAddress is addr at storage 11
  siringAuctionAddress is addr at storage 12
  erc721MetadataAddress is addr at storage 13
  autoBirthFee is uint256 at storage 14
  pregnantKitties is uint256 at storage 15
  geneScienceAddress is addr at storage 16
  promoCreatedCount is uint256 at storage 17
  gen0CreatedCount is uint256 at storage 18
  newContractAddress is addr at storage 19

def cfoAddress(): # not payable
  return cfoAddress

def promoCreatedCount(): # not payable
  return promoCreatedCount

def ceoAddress(): # not payable
  return ceoAddress

def pregnantKitties(): # not payable
  return pregnantKitties

def isPregnant(uint256 _horseId): # not payable
```

60606 409081526
03c82 152610e10
60a08 006101208
20152 262093a80
6101a 498d00006
00e55 6600160a0
60020 000000000
00000 000006200
28f06 600063fff
fffff 080fd5b61
ffff8 000195576
00d92 020830152
60009 5261ffff8
08516 382620005
cb565

Eveem 0x06012c8cf97BEaD5deAe237070F9587f8E7A266c show see random

Code
Json
Asm
Abi
Data
Meta

```
#
#  Eveem.org 26 Apr 2019
#  Decompiled source of 0x06012c8cf97BEaD5deAe237070F9587f8E7A266d
#
#  Let's make the world open source
#

const name = 'CryptoKitties'
const symbol = 'CK'
const GEN0_STARTING_PRICE = 10^16
const GEN0_AUCTION_DURATION = (24 * 3600)
const GEN0_CREATION_LIMIT = 45000
const PROMO_CREATION_LIMIT = 5000

def storage:
  ceoAddress is addr at storage 0
  cfoAddress is addr at storage 1
  cooAddress is addr at storage 2
  paused is uint8 at storage 2 offset 160
  cooldowns is array of uint32 at storage 3
  secondsPerBlock is uint256 at storage 5
  kitty is array of struct at storage 6
  kittyIndexToOwner is mapping of addr at storage 7
  balanceOf is mapping of uint256 at storage 8
  kittyIndexToApproved is mapping of addr at storage 9
  sireAllowedTo is mapping of addr at storage 10
  saleAuctionAddress is addr at storage 11
  siringAuctionAddress is addr at storage 12
  erc721MetadataAddress is addr at storage 13
  autoBirthFee is uint256 at storage 14
  pregnantKitties is uint256 at storage 15
  geneScienceAddress is addr at storage 16
  promoCreatedCount is uint256 at storage 17
  gen0CreatedCount is uint256 at storage 18
  newContractAddress is addr at storage 19

def cfoAddress(): # not payable
  return cfoAddress

def promoCreatedCount(): # not payable
  return promoCreatedCount

def ceoAddress(): # not payable
  return ceoAddress

def pregnantKitties(): # not payable
  return pregnantKitties

def isPregnant(uint256 _horseId): # not payable
```

Tips and Tricks

No Source Code? No problem.

Handy decompilers

- eveem.org
- contract-library.com
- etherscan.io/bytecode-decompiler

Tips and Tricks

Use the Constructor

- Deploy and setup in the constructor

Tips and Tricks

Use these handy reversing tools

- Eveem.org
- Contract-library.com
- Even https://etherscan.io/bytecode-decompiler

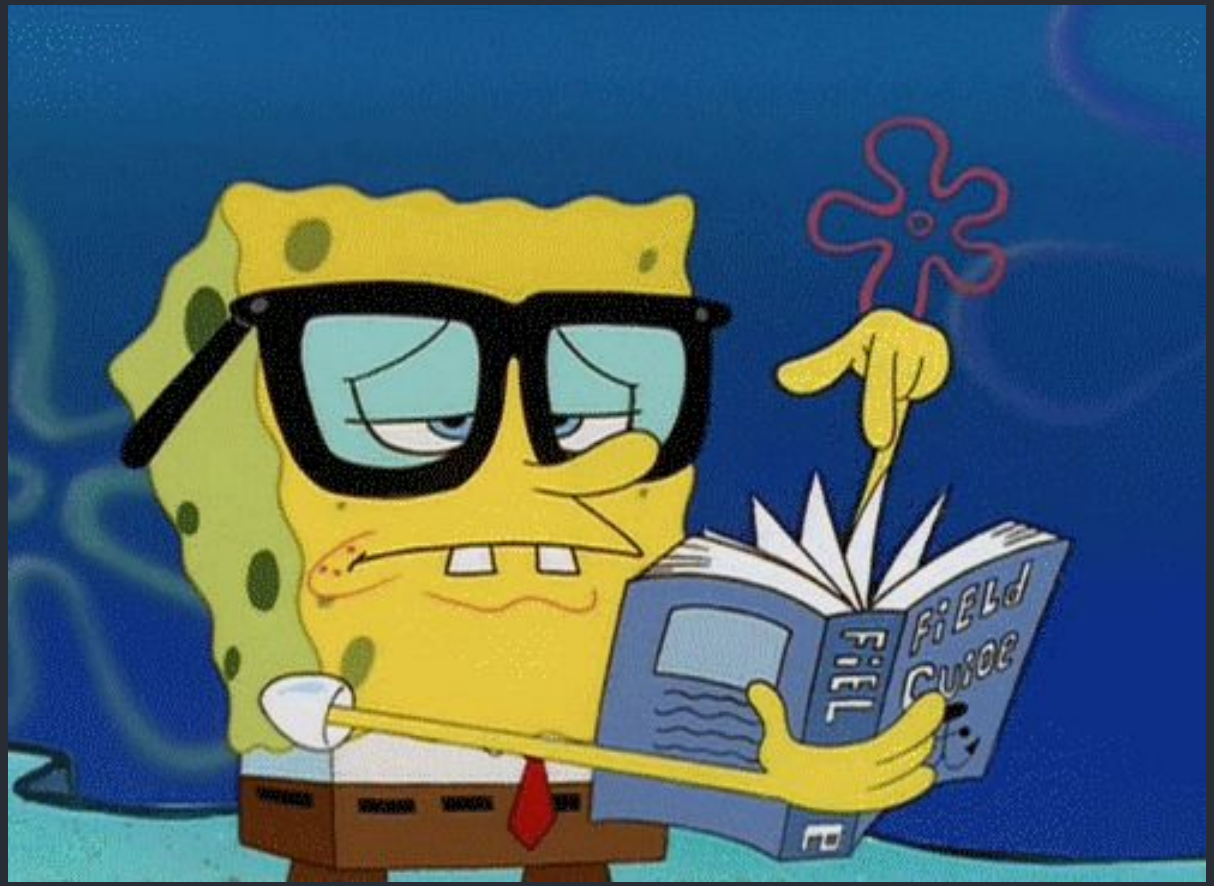
Tips and Tricks

Inherit these handy utilities

```
pragma solidity ^0.5.0;

contract CtfUtils {

    event log(string, uint256);
    event log(string, bytes32);
    event log(string, address);

    function getEthBalance() view public returns (uint256) {
        return address(this).balance;
    }

    function getEthBalanceAt(address _at) view public returns (uint256) {
        return _at.balance;
    }

}
```

Overall strategy

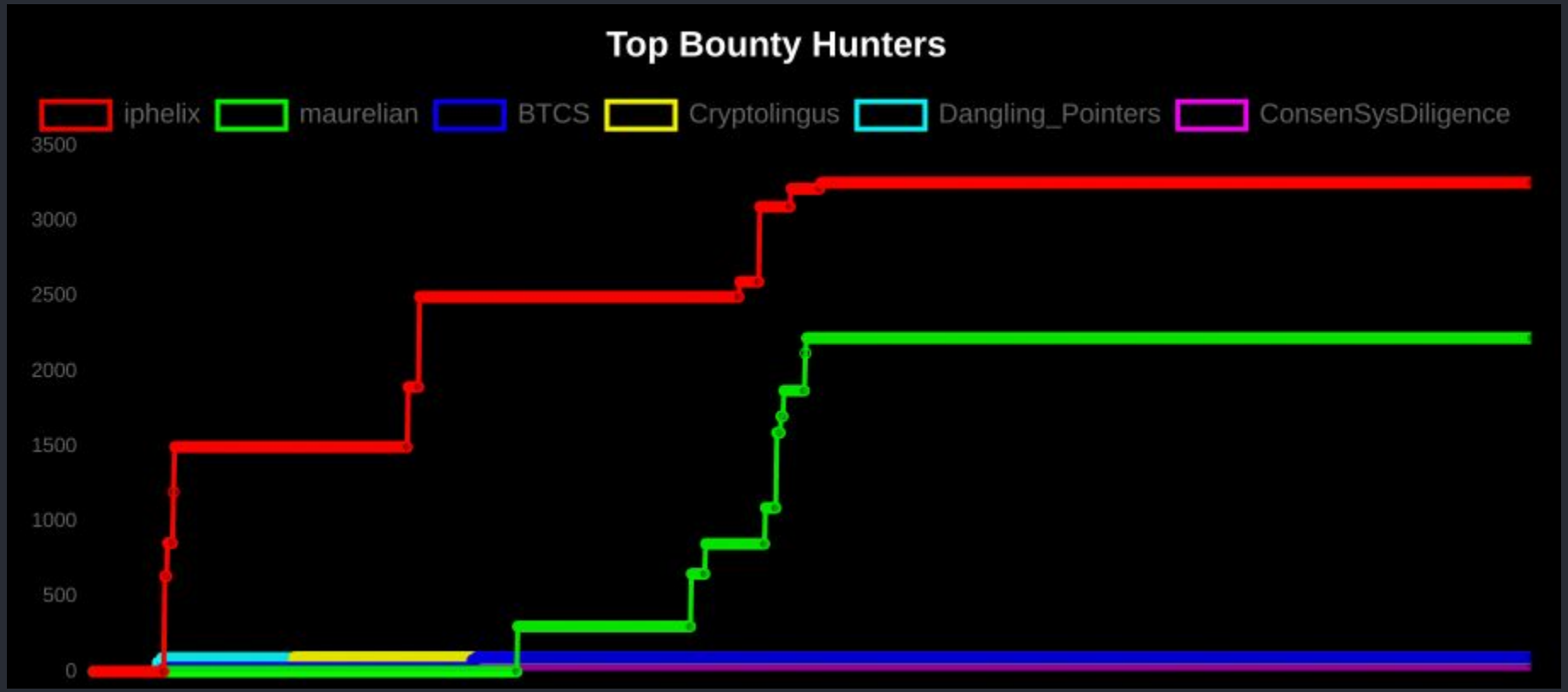
Scan challenges



Know your team

Start with the easy ones

Start with the easy ones



Ask for help

Delete this slide

Notes / Ideas

This talk is going to be so much fun!

The vuln ID section, and the tips and tricks section have some overlap… (ie. use “web3.eth.getStorageAt” use a “decompiler”), could it make sense just to combine them?

Encourage the pros to give other’s a chance.

ToDos

- Add references to SWC
- Pump Visual Auditor much more
- What if the code is too small?
  - Make a public deck/pdf with just the code samples in it?
-

Delete this slide

Take breaks

ROUND 10

```
contract MetaToken {
    mapping(address => uint256) public balanceOf;
    // Imagine typical ERC20 stuff here ...

    mapping(uint256 => bool) usedNonces;
    // Meta transactions rule! Anyone can just sign a list of recipients and
    // amounts, and then a relayer can call metaBatchTransfer to make it happen.
    function metaBatchTransfer(
          bytes calldata signature, address[] calldata recipients,
           uint160[] calldata amounts, uint160 nonce)
     external {
        require(!usedNonces[nonce], "nonce reuse");
        usedNonces[nonce] = true;
        bytes32 hash = keccak256(abi.encodePacked(recipients, amounts, nonce));
        address signer = hash.recover(signature);
        for (uint256 i = 0; i < amounts.length; i++) {
            uint256 amount = amounts[i];
            address recipient = recipients[i];
            balanceOf[signer] = balanceOf[signer].sub(amount);
            balanceOf[recipient] = balanceOf[recipient].add(amount);
        }
```

Is it just me, or is it crowded in here?

uint160?

uint160?

```
keccak256(abi.encodePacked([0x01, 0x02, 0x03], [10,100,1000], 123));
```

Is equivalent to:

```
keccak256(abi.encodePacked([0x01, 0x02, 0x03, 10, 100], [1000], 123));
```