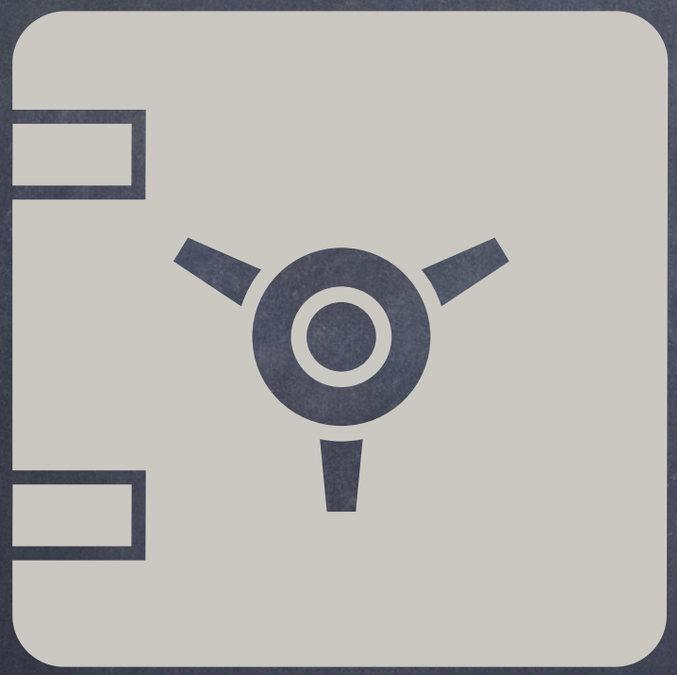
Security Pitfalls & Best Practices 101

Secureum Bootcamp

#1

Solidity



Old vs New Versions

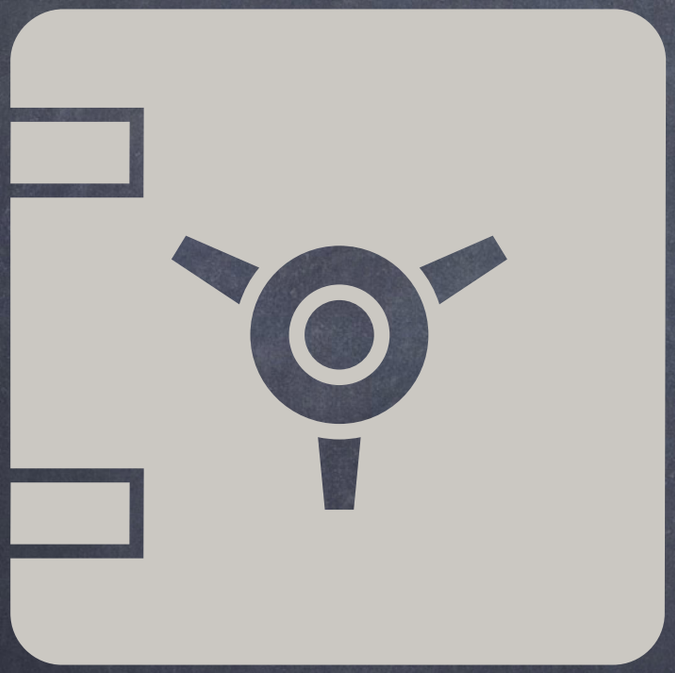
Old: Bugs, Time-Tested
New: Bug-fixes, New Bugs

Old: Less Features
New: More Features

Optimal Version
Security vs Functionality

#2

Unlocked Pragma



Solidity Pragma

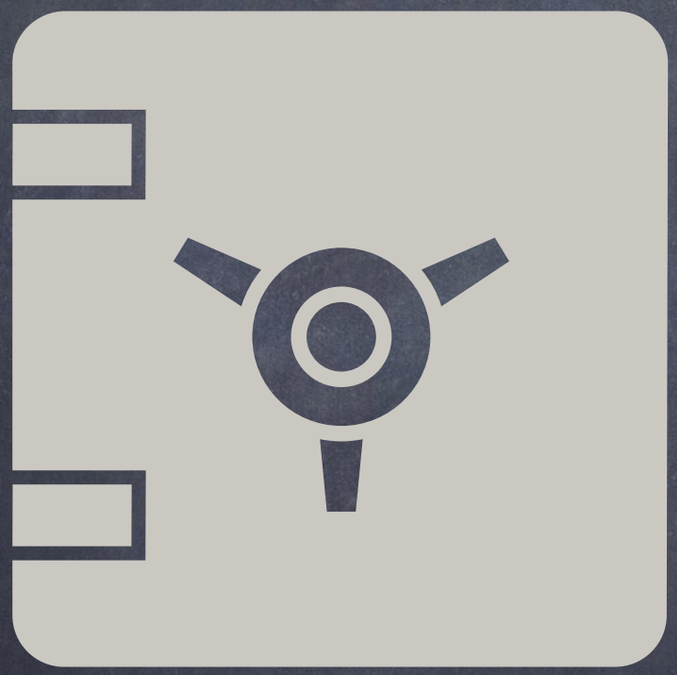
Use of ^ -> Unlocked

Testing vs Deployment
Different Versions

Lock Pragma
Test -> Deploy

#3

Multiple Pragma



Solidity Pragma

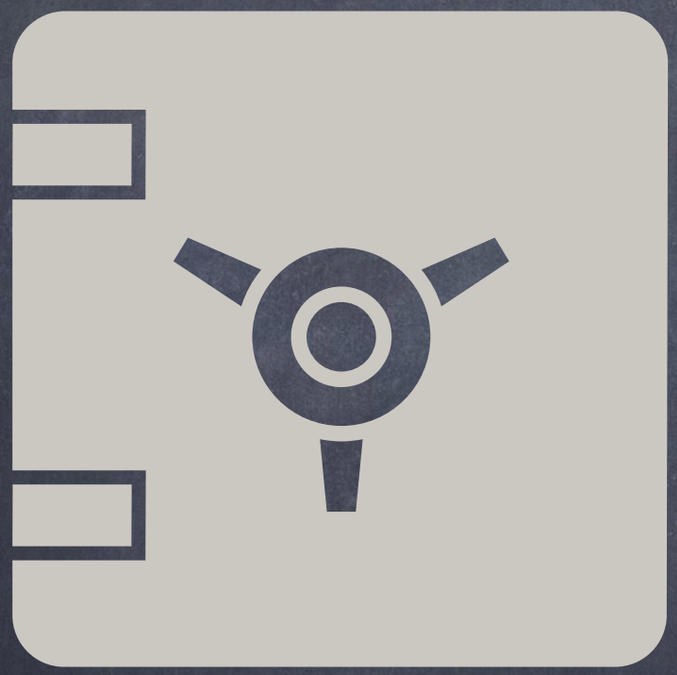
Different Pragmas
Different Contracts

Different Bugs
Different Features/Checks

Same Pragma
Same Bugs/Features/Checks

#4

Access Control



Access to Functions

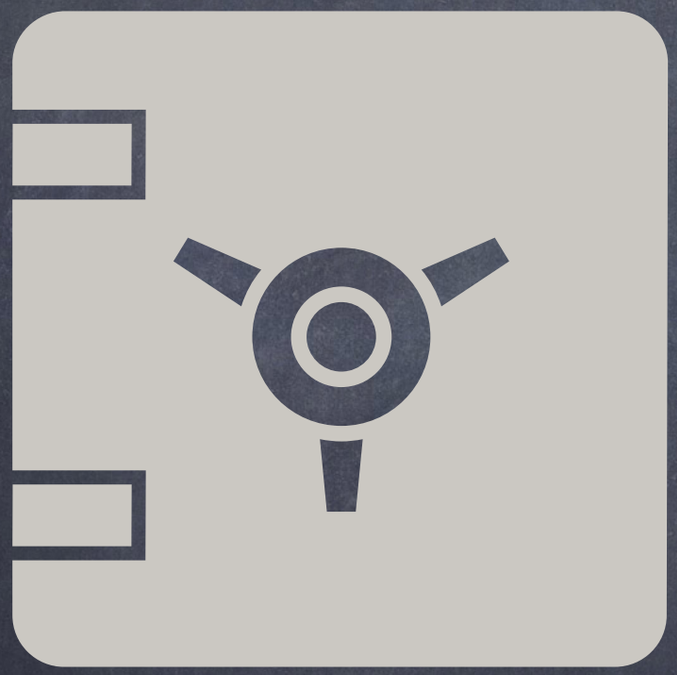
Public/External Functions

Addresses: Anyone/Owner/ RBAC

Correct Modifiers/ Addresses -> Enforce Access

#5

Withdraw Funds



Unprotected Withdraw Functions

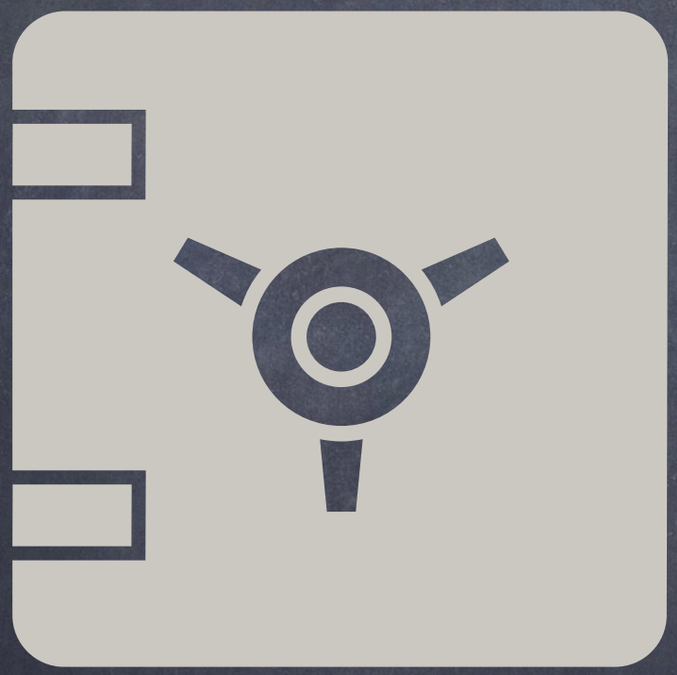
Public/External Withdraw ETH/Tokens

Unauthorized Withdraws
Loss of Funds

Access Control Withdraw Functions

#6

selfdestruct



selfdestruct()
Unprotected

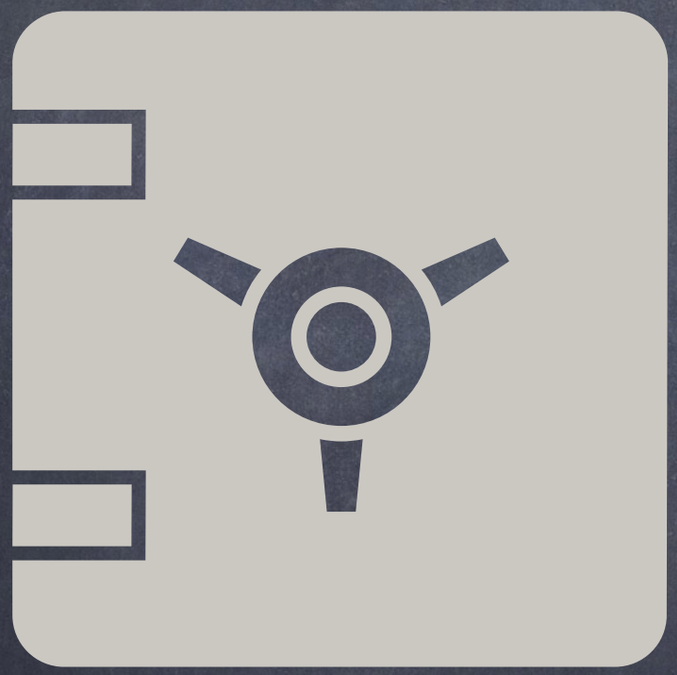
Destroy/Remove Contract

Unauthorized Call
Contract Killed

Access Control
Authorized Users

#7

Modifiers
Side-effects



Modifier Side-effects

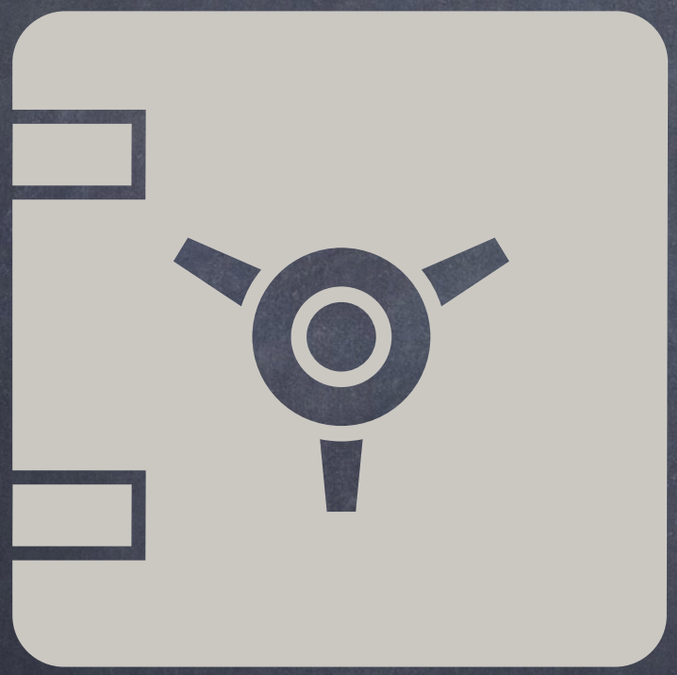
Modifiers -> Checks
No Side-effects

Unnoticed Side-effects
Readability & Auditability

No Side-effects
Only Checks

#8

Incorrect Modifier



Incorrect Modifier

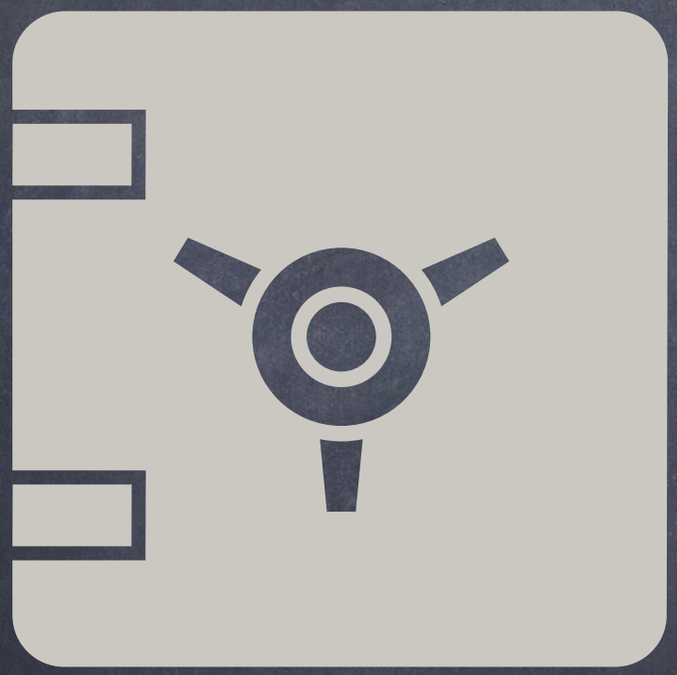
Checks
Execute _ or Revert

Default Value Returned
Unexpected

All Modifier Paths
Execute _ or Revert

#9

Constructor Names



Constructor Names

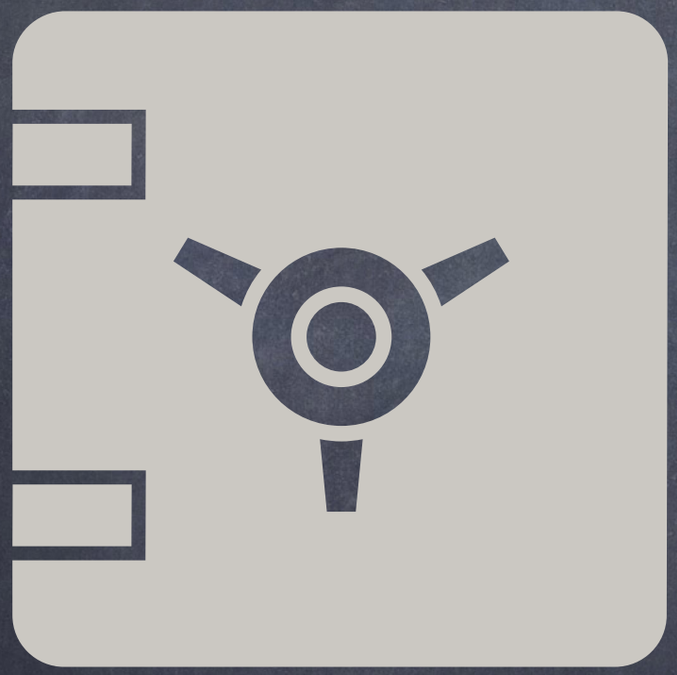
<0.4.22 -> Contract Name
<0.5.0 -> or constructor

>=0.5.0 -> Only constructor

Naming Confusion -> Bugs

#10

Void Constructor



Base Constructor

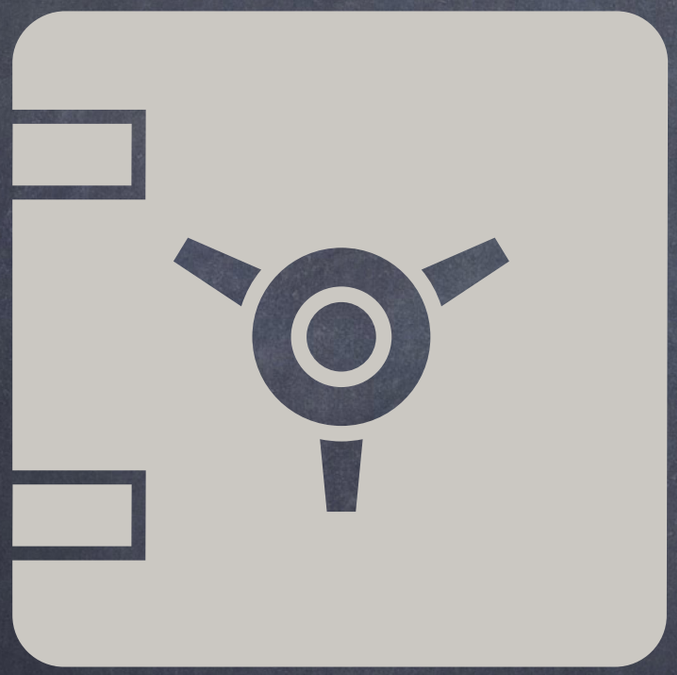
Calling Unimplemented Constructors

Misplaced Assumptions

Check Implementation Call or Not

#11

Constructor
callValue



Implicit Constructor
callValue Check

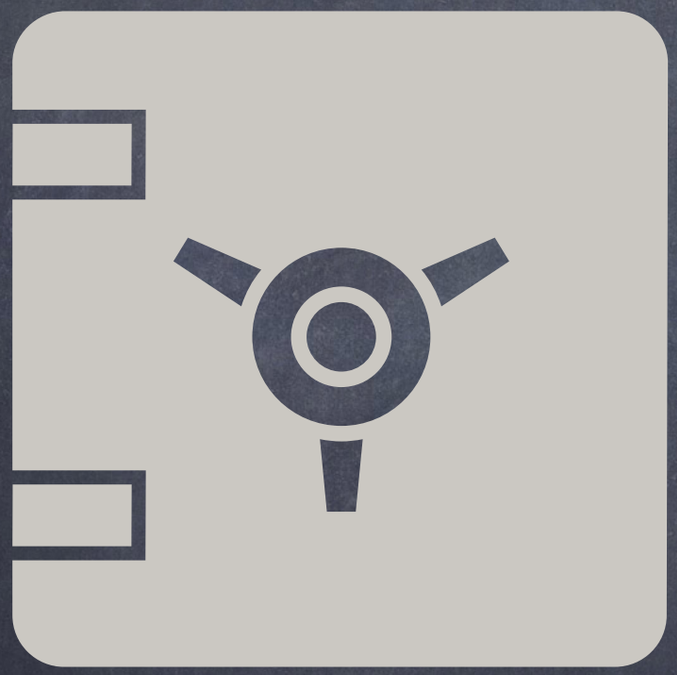
Not Explicitly Payable
Constructor -> Revert

Base Constructor
!=0 callValue -> No Revert

Compiler Bug
v0.4.5 -> v0.6.8

#12

delegateCall



Controlled delegateCall
User-controlled Address

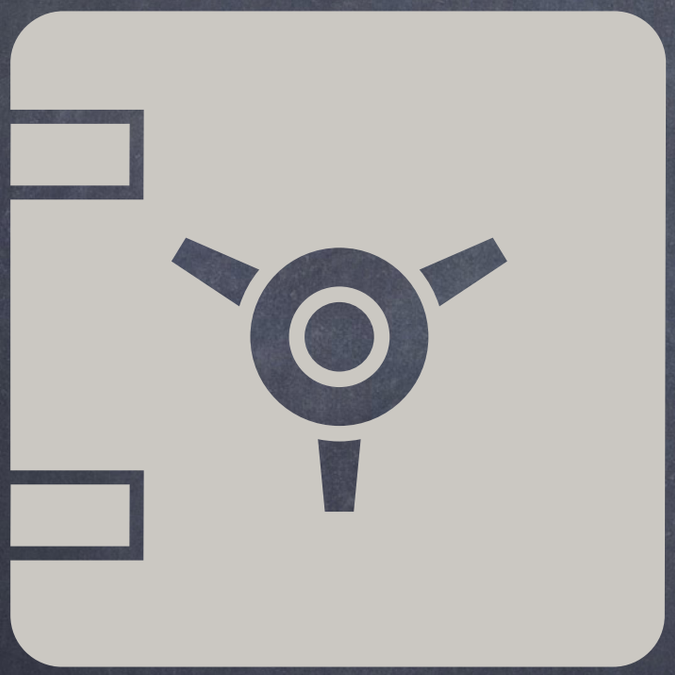
Malicious Contracts

Contract State ->
Unauthorized Modification

delegateCall -> Trusted
Addresses

#13

Reentrancy



External Contract Calls ->
Reentrancy

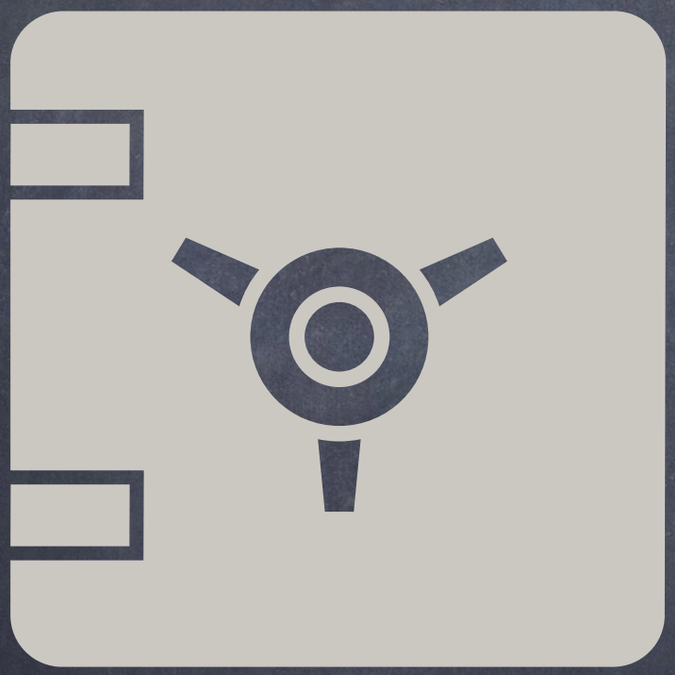
Contract Callbacks
C1 -> C2 -> C1

Multiple Withdrawals
000 Events

CEI Pattern
Reentrancy Guards

#14

ERC777



ERC777 Token Standard
Callbacks -> Reentrancy

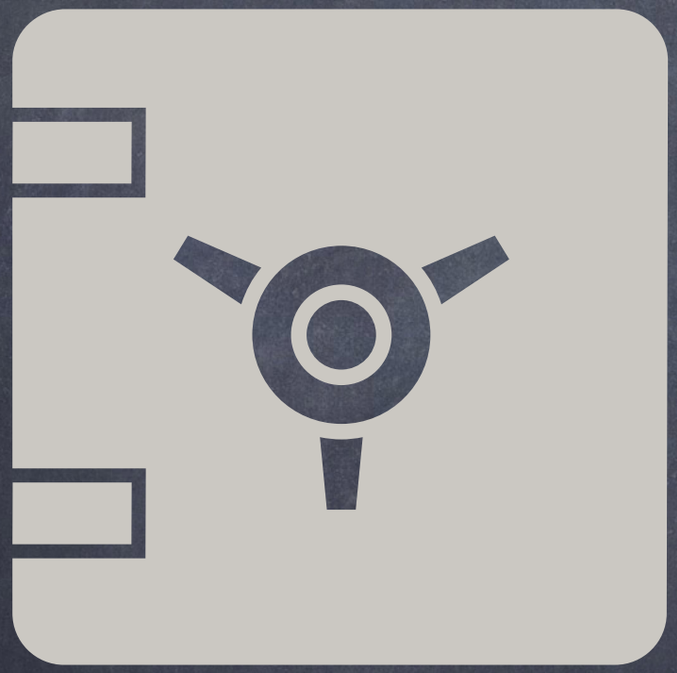
ERC20 Extension
Token Transfers -> Hooks

Hooks -> Reentrancy
C1->ERC777->Hook->C1

CEI Pattern
Reentrancy Guards

#15

transfer()
& send()



ETH transfer() & send()
Reentrancy Mitigations

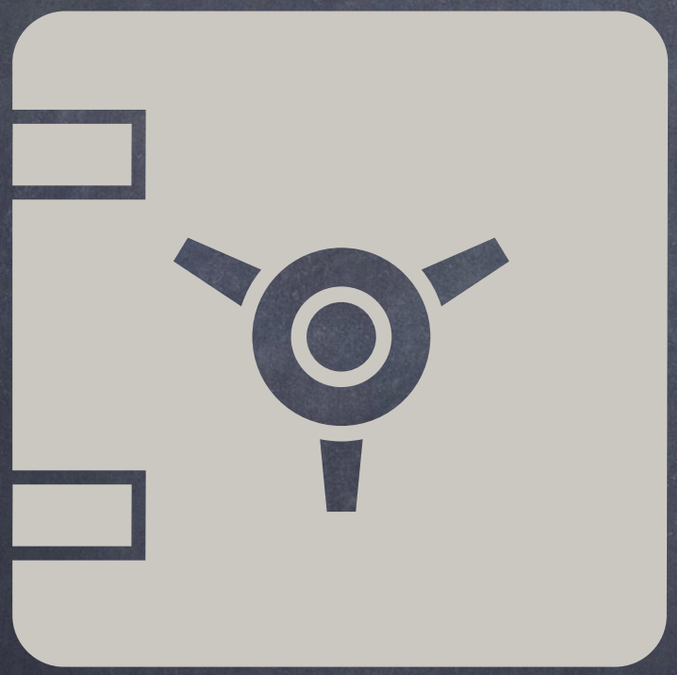
Gas Subsidy 2300
Reentrancy -> No Gas

Opcode Gas Repricing
Break Contracts

call() & CEI Pattern &
Reentrancy Guards

#16

Private Data



Privacy of On-chain Data

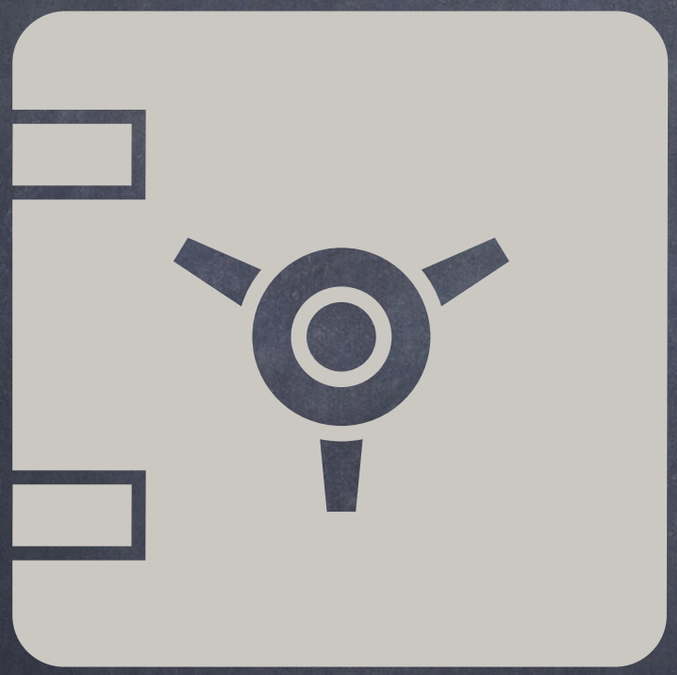
private Variables != Cannot Read

Blockchain Txs & State
On-chain vs Off-chain

Private Data
Encrypted Off-chain

#17

PRNG



PRNG in Contracts
Randomness Applications

block.timestamp
blockhash

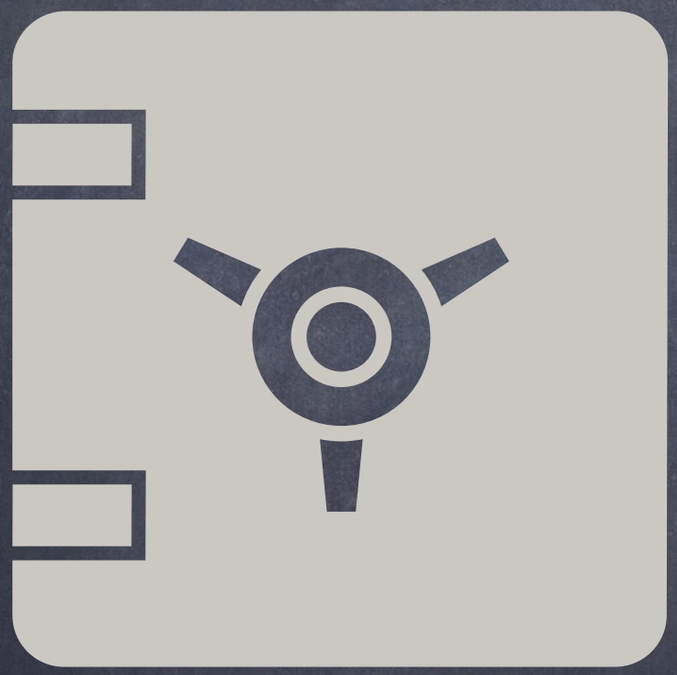
High Stakes
Miner Influence

Risk Awareness
Alternatives: VRF

#18

Time

On-chain Time



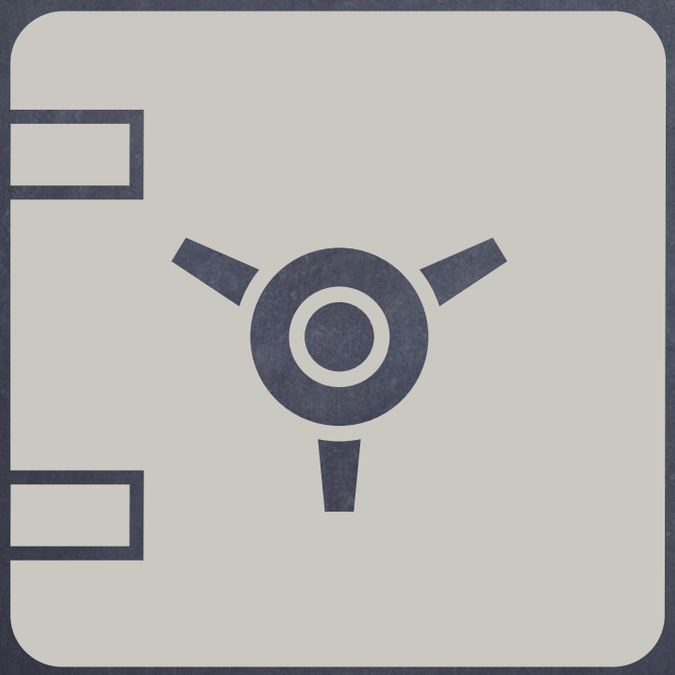
block.timestamp
block.number

Miner Influence &
Synchronization &
Unpredictable

Risk Awareness
Alternatives: Oracles

#19

Overflow/
Underflow



Integer Arithmetic
Overflows & Underflows

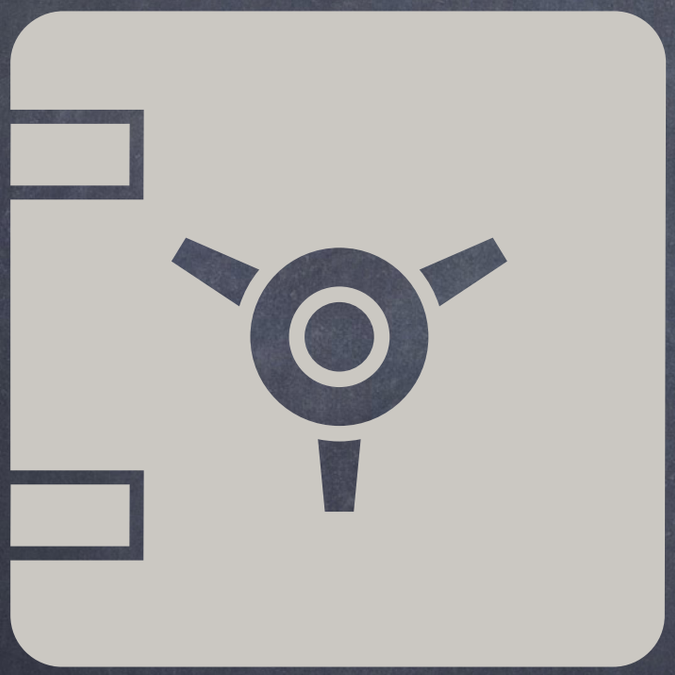
Wrapped Values -> Invalid
Data -> High/Low

Unexpected Behavior &
Vulnerabilities

OZ SafeMath
Default Checks -> >=0.8.0

#20

Divide -> Multiply



Integer Arithmetic
Divide Before Multiply

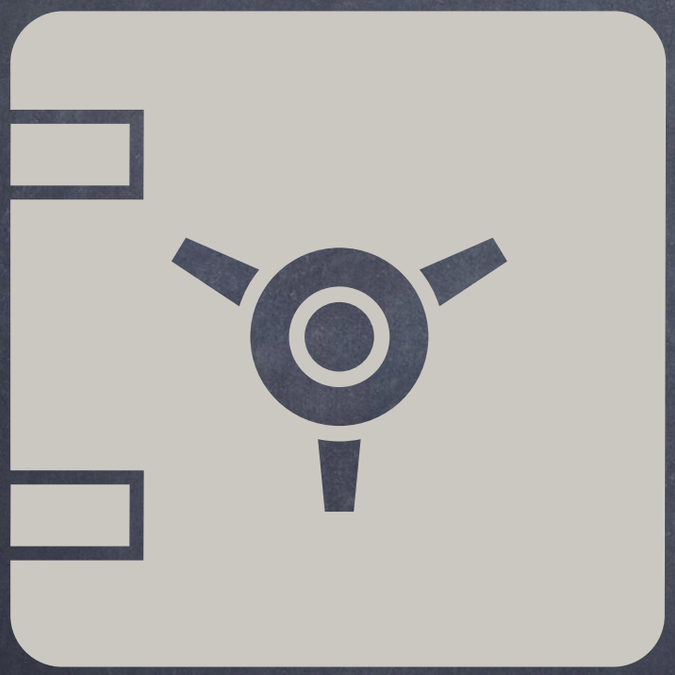
Solidity Integer Division
Truncation

Division -> Multiplication
Loss of Precision

Multiplication -> Division

#21

TOD



Transaction Order Dependence

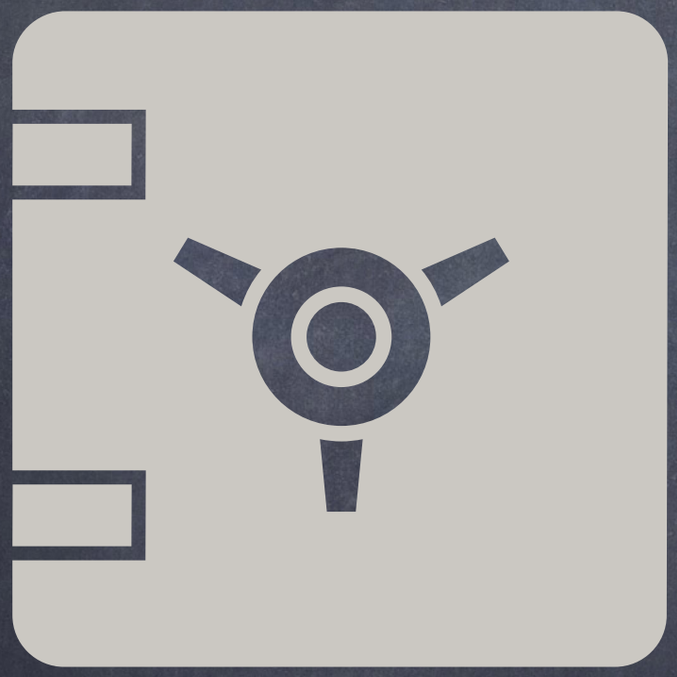
Mempool Transactions
Transaction Inclusion

Front-running &
Back-running

Don't Make Assumptions
Transaction Order

#22

ERC20
approve()



ERC20 approve()
Race-condition

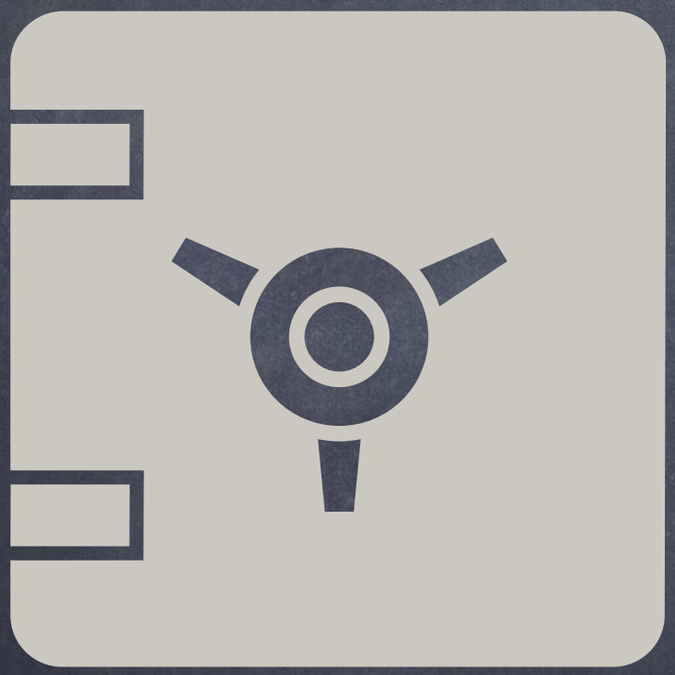
approve(100) -> approve(50)

Allowance Decrease
Front-run -> Spend 150

increaseAllowance()
decreaseAllowance()

#23

ecrecover



EVM ecrecover
Signature Malleability

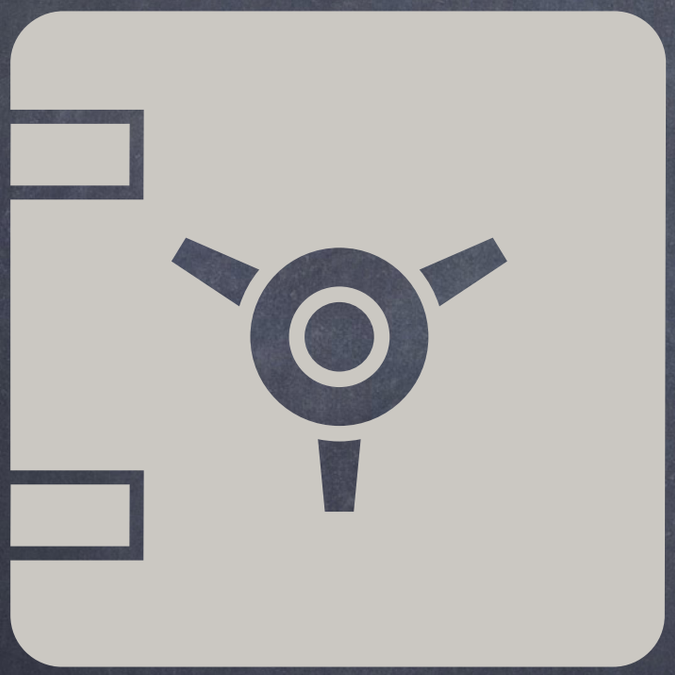
Sig -> (v, r, s)
Check s -> Lower Range

Replay Attacks

Use OZ ECDSA

#24

transfer()



ERC20 transfer()

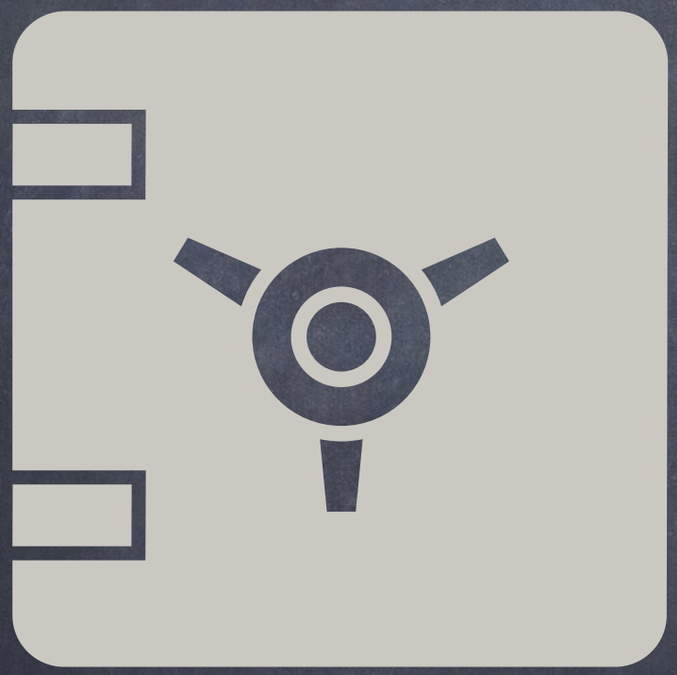
Specification -> Boolean Return Value

No Return Value -> Revert
solc >= 0.4.22

OZ SafeERC20 Wrappers

#25

ownerOf()



ERC721 ownerOf()

Specification -> Address Return Value

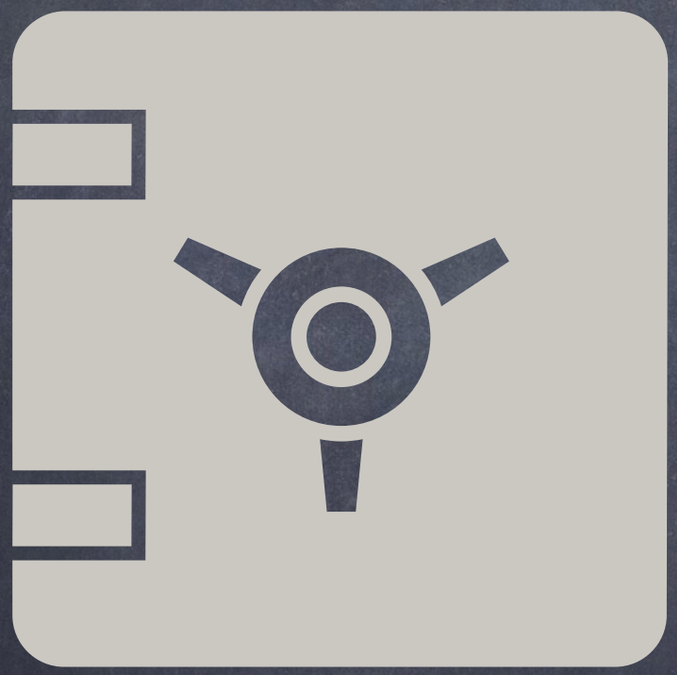
Bool Return Value -> Revert

solc >= 0.4.22

OZ ERC721 Contract

#26

Contract Balance



Unexpected Ether

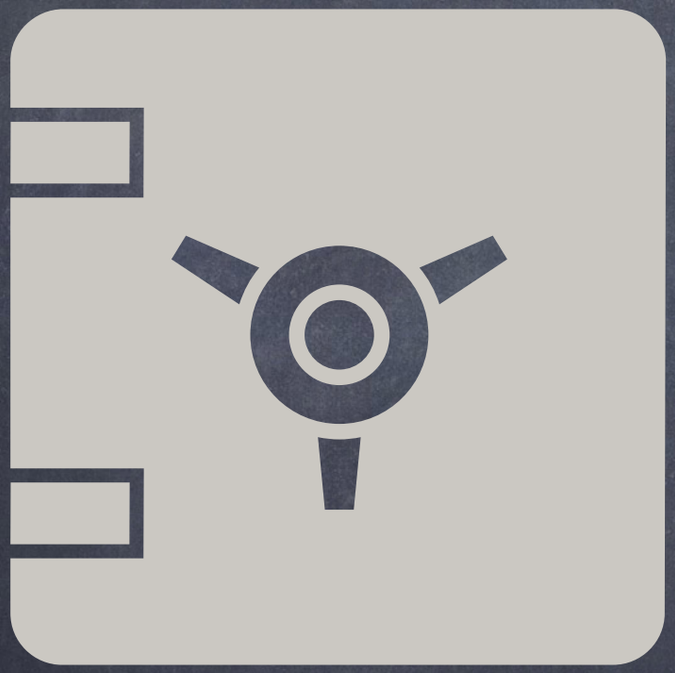
Creation
Payable Functions

Coinbase Tx
selfdestruct()

Assumptions on
this.balance

#27

fallback vs receive



fallback()
receive()

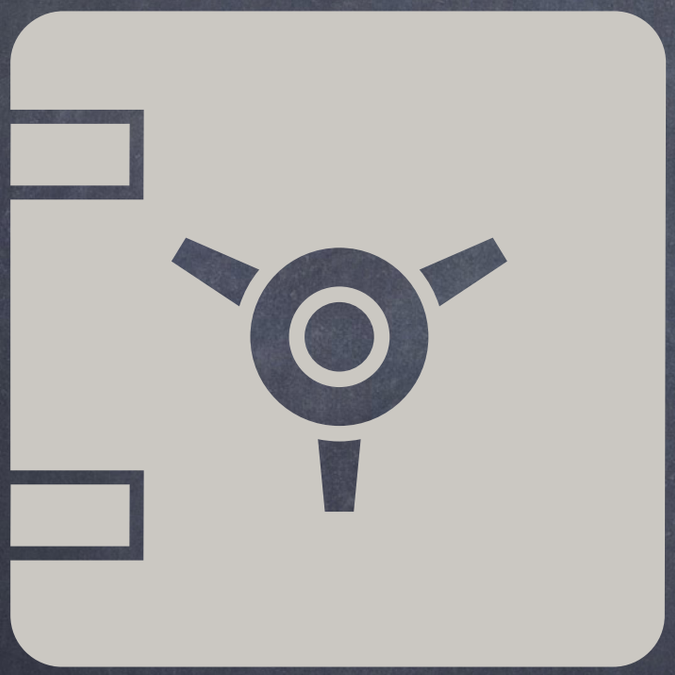
Differences
Similarities

Visibility, Mutability, Ether Transfers

Check Assumptions & Implications

#28

Strict Equalities



Strict Equalities
Dangerous

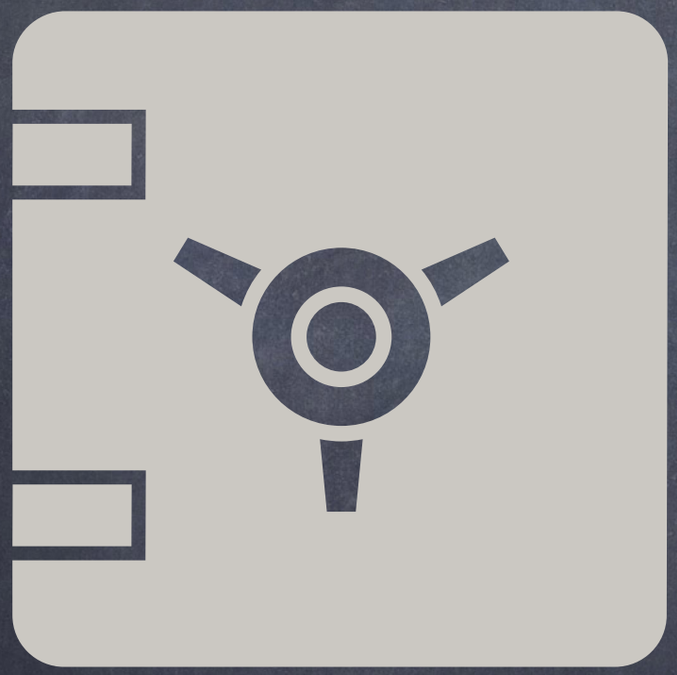
== Vs <= or >=

ETH/Token
Transfers & Balances

Check Constraints
Safe Defaults

#29

Locked Ether



Contract Balance
Locked Ether

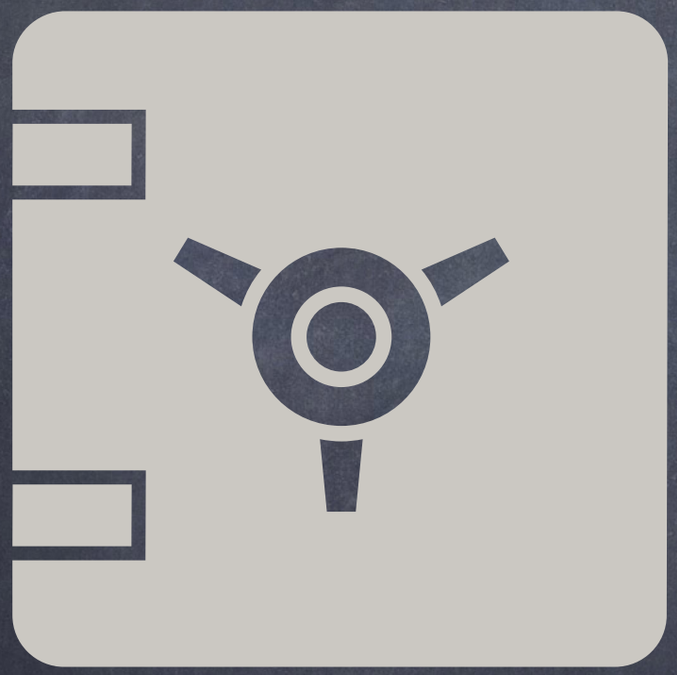
Payable Functions
No Withdraw

Can Deposit
Cannot Withdraw

Remove Payable
Add Withdraw

#30

tx.origin



Use of tx.origin
Dangerous

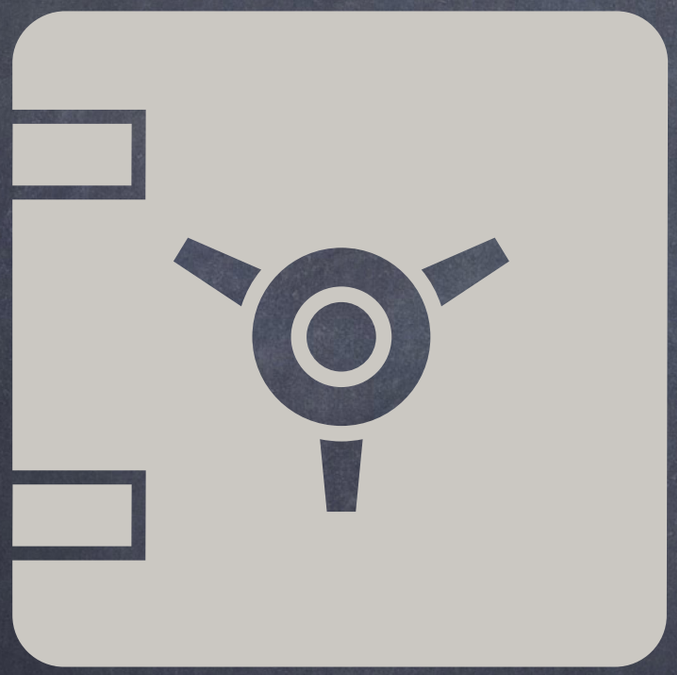
tx.origin -> EOA

Authorization
MITM Replay Attack

tx.origin -> msg.sender

#31

Contract?



Contract Vs EOA

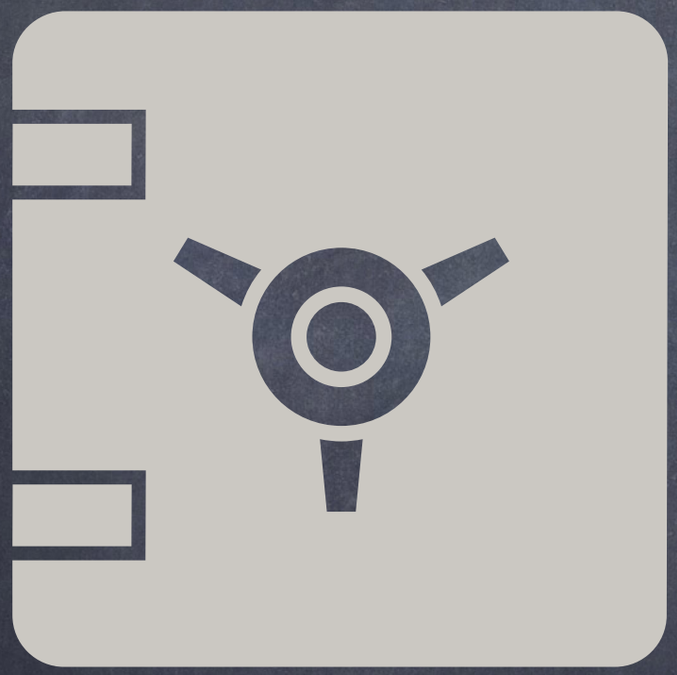
extcodesize > 0
msg.sender == tx.origin

Pros & Cons
Applications

Risk Awareness
Implications

#32

delete
Mapping



delete Mapping
Within Struct

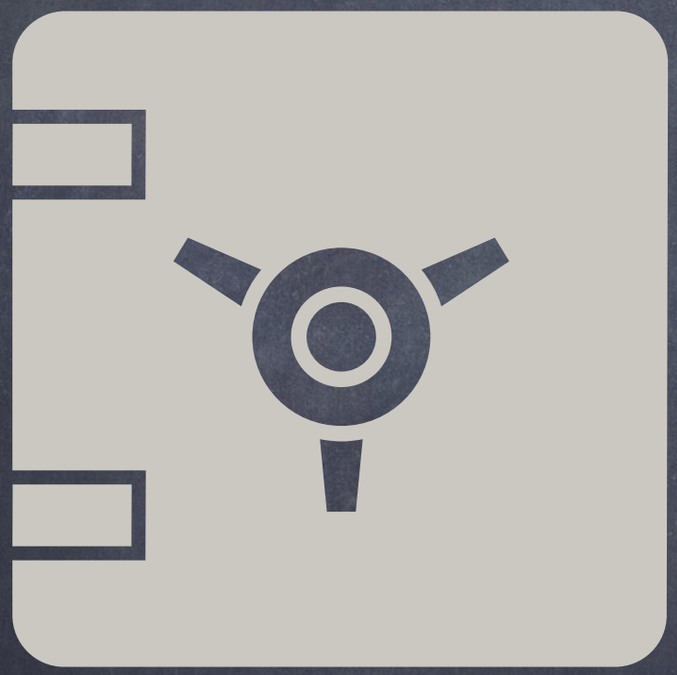
delete Struct
Mapping Intact

Unintended Consequences

Use Alternatives
Lock Vs Delete

#33

Tautology
Contradiction



Tautology -> Always True

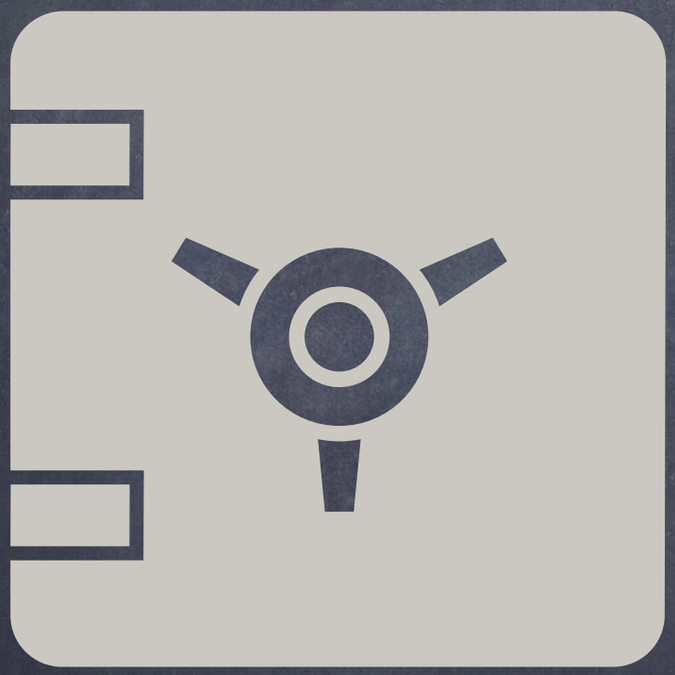
Contradiction -> Always False

x >= 0
uint x

Flawed Logic
Redundant Check

#34

Boolean Constant



Boolean Constants
true or false

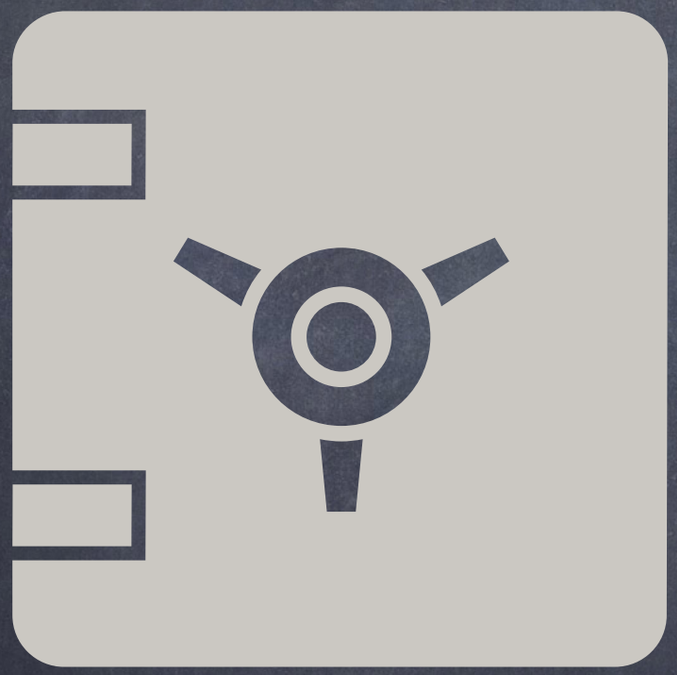
Use in Conditionals
Unnecessary

Flawed Logic
Redundant Use

Verify Usage
Remove -> Simplify

#35

Boolean Equality



Boolean Equality
true or false

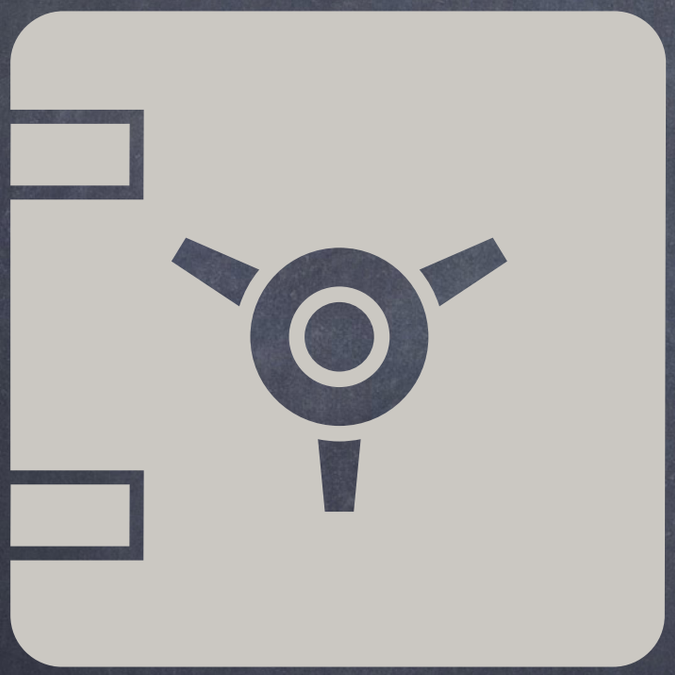
Use in Conditionals
if (x == true)

Redundant Use
Unnecessary

Use Directly
if (x)

#36

State Modification



Contract State Modifications

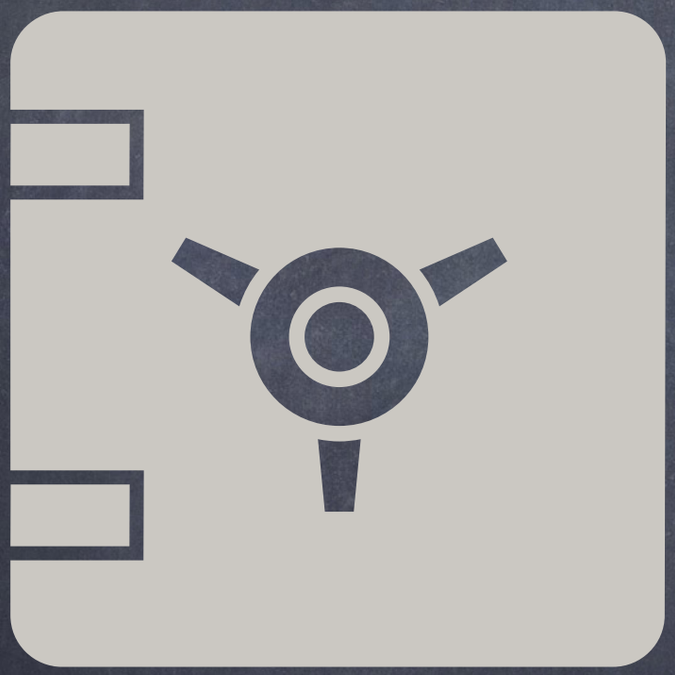
Functions: view or pure

solc >= 0.5.0
STATICCALL -> Revert

Function Mutability Check & Apply

#37

Return Values



Function Return Values
Low-level Calls

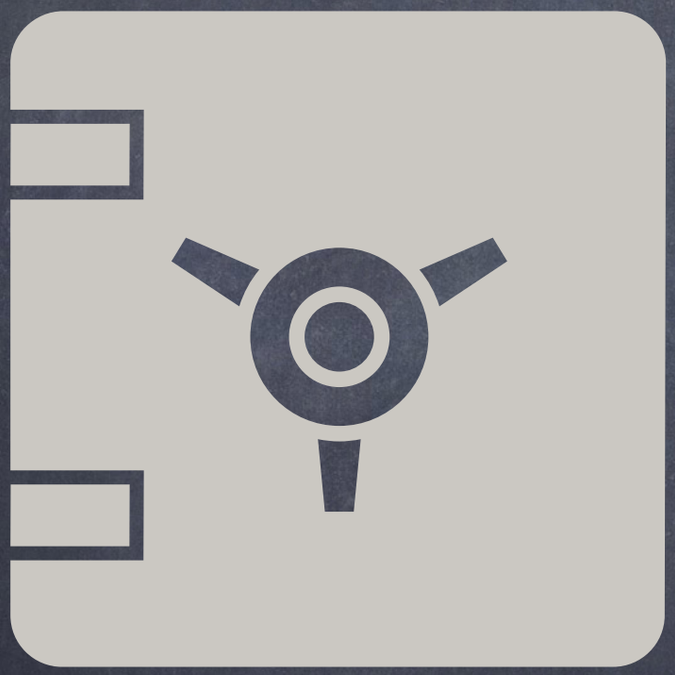
call/delegateCall/send

Check Return Value
Success/Failure

Unexpected Failures

#38

Account Existence



Account Existence
Low-level Calls

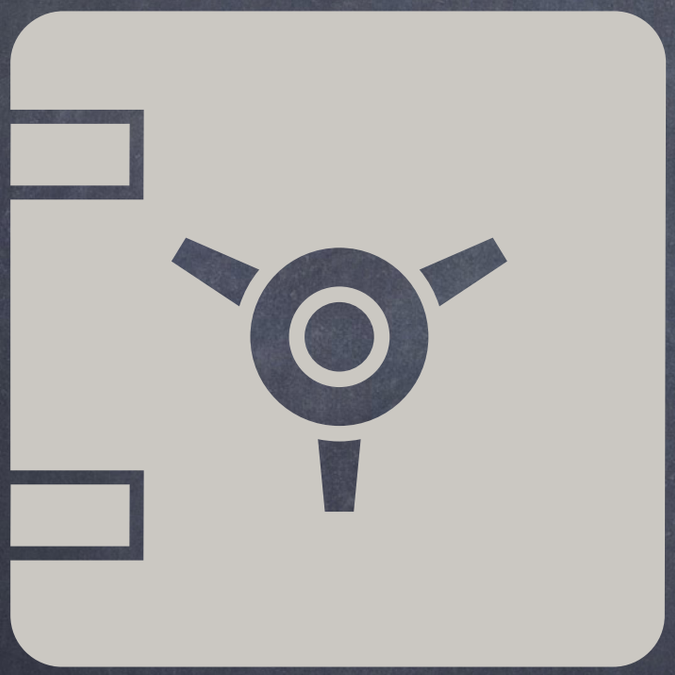
call/delegateCall/
staticCall

No Account -> Return true

Check Existence -> Before Call

#39

Shadowing



Solidity Built-in Shadowing

Built-in: now, assert etc.

Names: Variables/ Functions/Modifiers

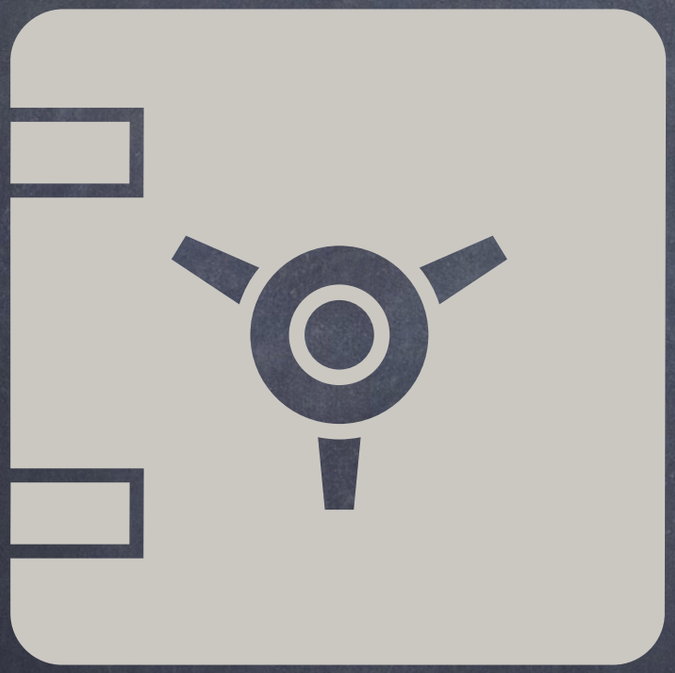
Override Behavior Dangerous/Unexpected

#40

Shadowing

State Variables
Shadowing

Contracts: Base Vs Derived

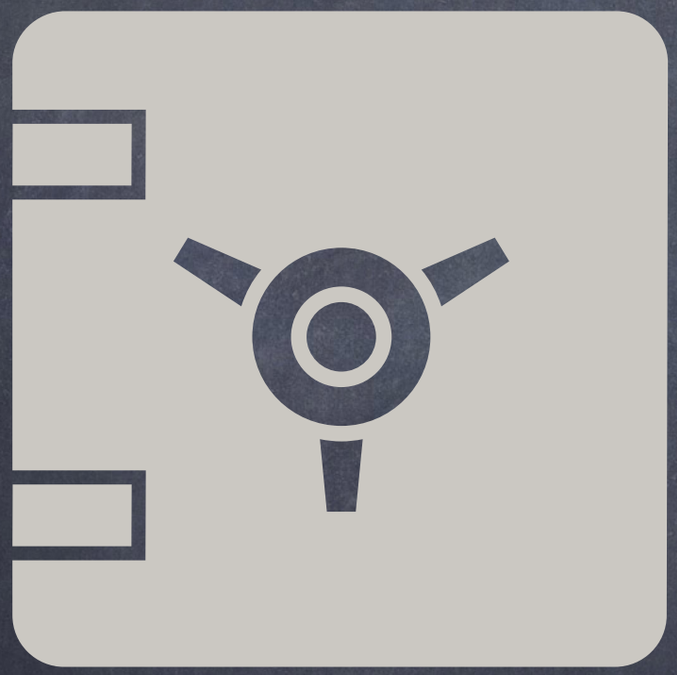


Base Variables
Shadowed Variables

Wrong Variable Usage
Dangerous/Unexpected

#41

Pre-declaration



Local Variables
Pre-declaration Usage

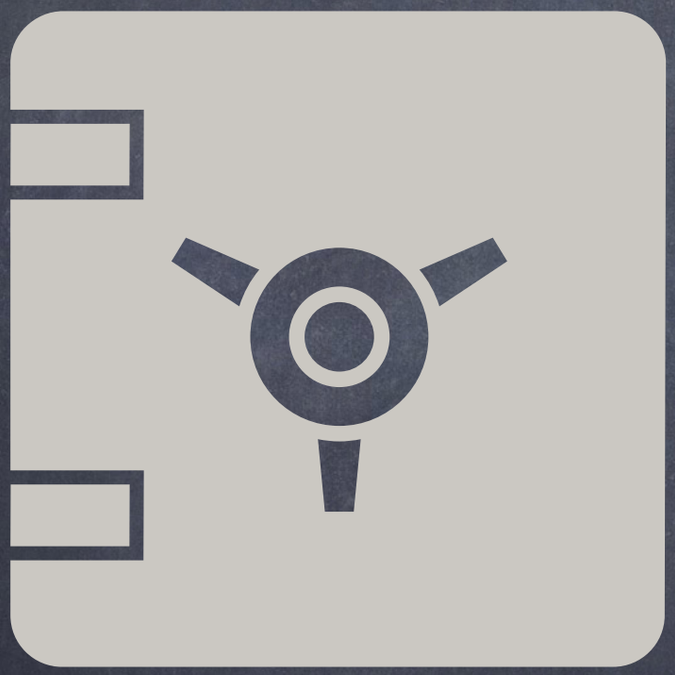
Declared Later
Declared Another Scope

solc < 0.5.0
Undefined Behavior

solc >= 0.5.0
C99-style Scoping Rules

#42

Costly Operations



Costly Operations Inside Loops

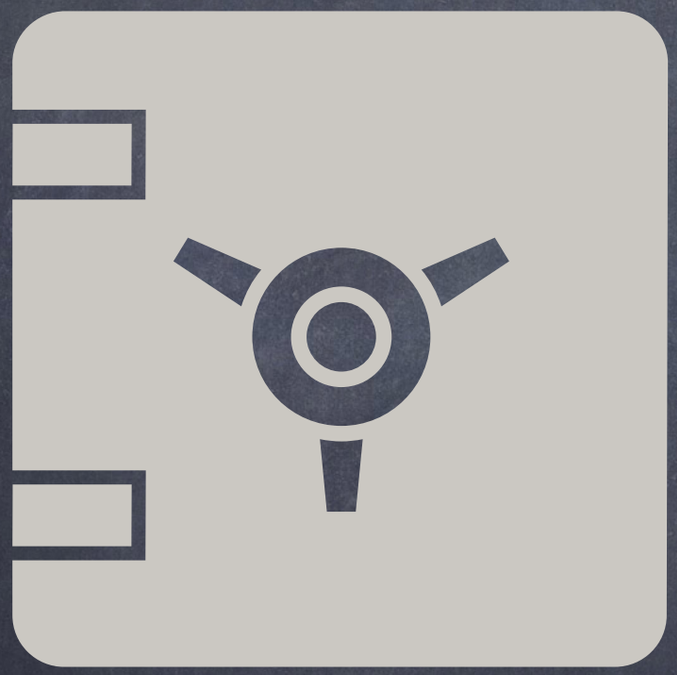
State Variable Updates

Out-of-Gas Error

Use Local Variables
Prevent DoS

#43

Costly Calls



External Calls
Inside Loops

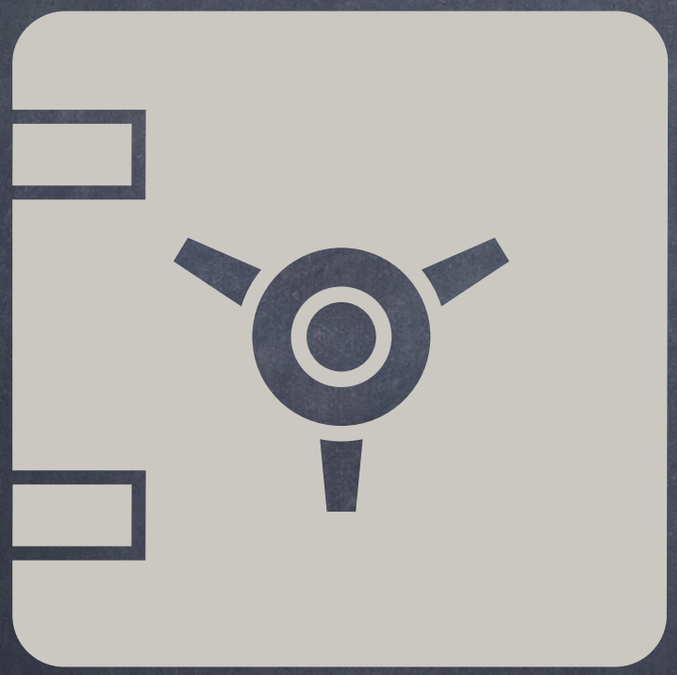
Loop Index
User Controlled

Reverts
Out-of-Gas Error

Avoid Calls in Loops
Prevent DoS

#44

Block Gas Limit



Block Gas Limit
15M Currently

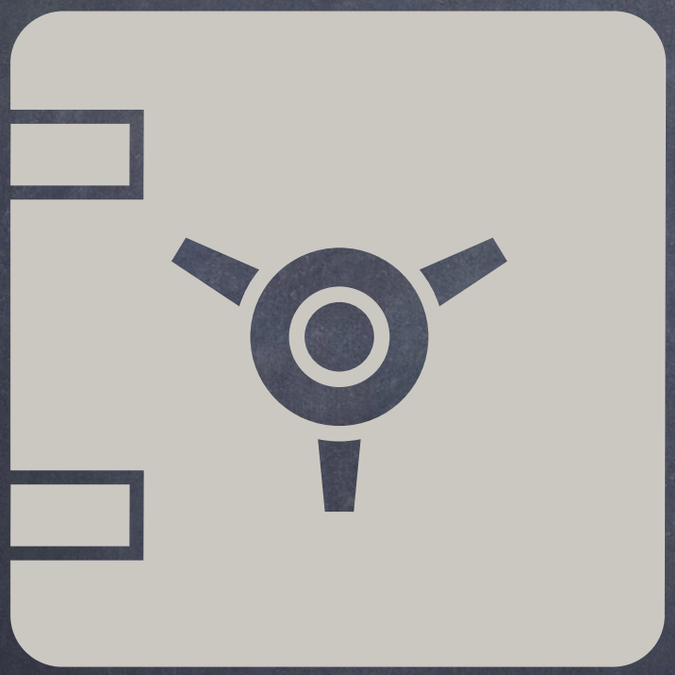
Loop Index
User Controlled

Expensive Operations
Out-of-Gas Error

Evaluate Loops
Prevent DoS

#45

Events



Missing Events
Critical Operations

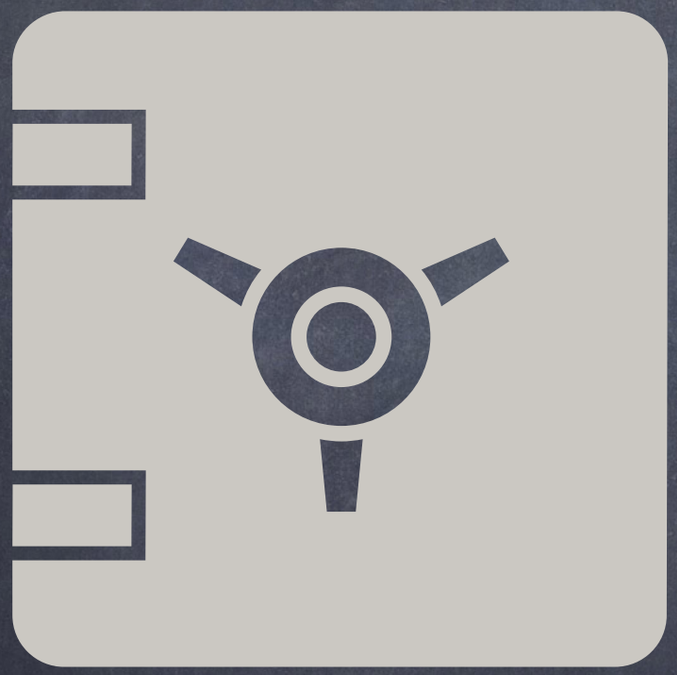
Off-Chain Monitoring

Transparency & UX

Add Events
Critical Operations

#46

Event Parameters



Event Parameters
Not Indexed

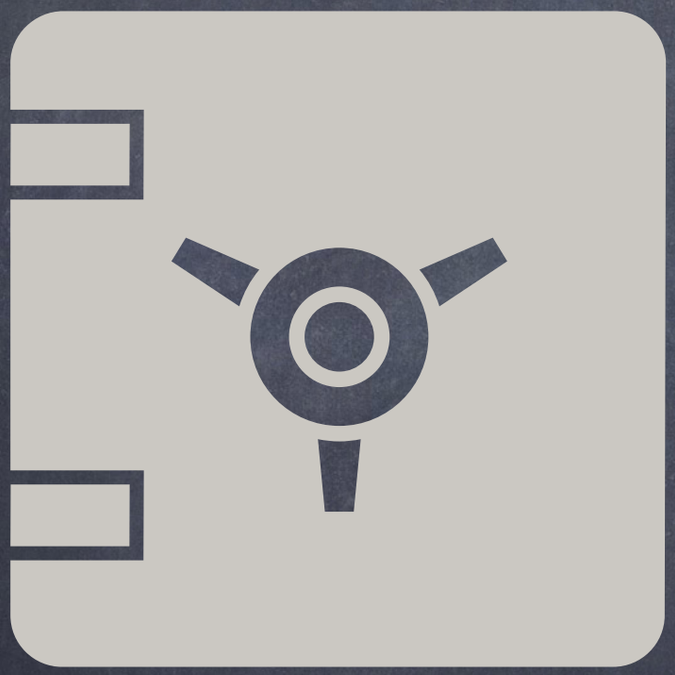
Indexed -> Topics
Not Indexed -> Data

ERC20 Specification
Faster Lookup

Add 'Indexed'
Critical Parameters

#47

Event Signatures



Event Signatures
Contract Type

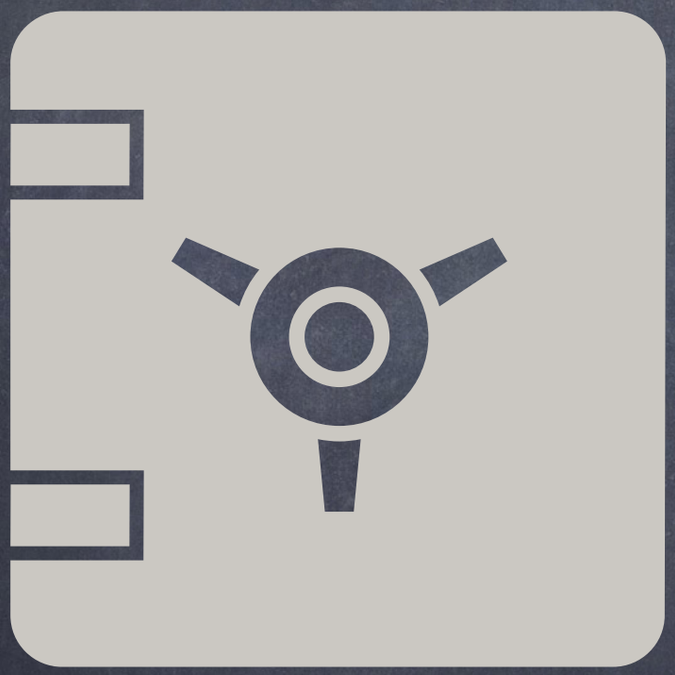
Contract Parameter
Address Vs Name

Wrong Hash
solc Bug: 0.5.0 -> 0.5.8

Use Address Type

#48

Unary Expression



Unary Expressions
Typo Susceptibility

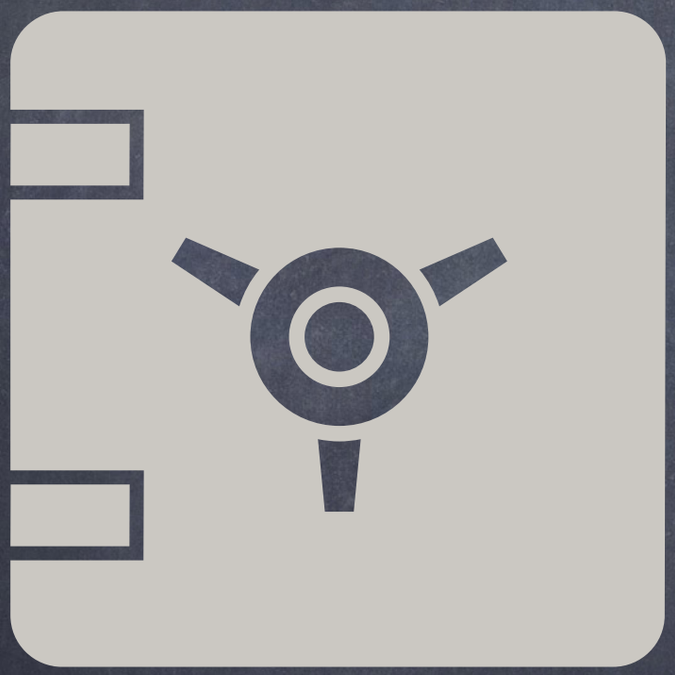
```
x += 1
x =+ 1
```

Increment vs Re-initialize

Unary + Deprecated
solc 0.5.0

#49

Zero Addresses



Zero Addresses
Special Consideration

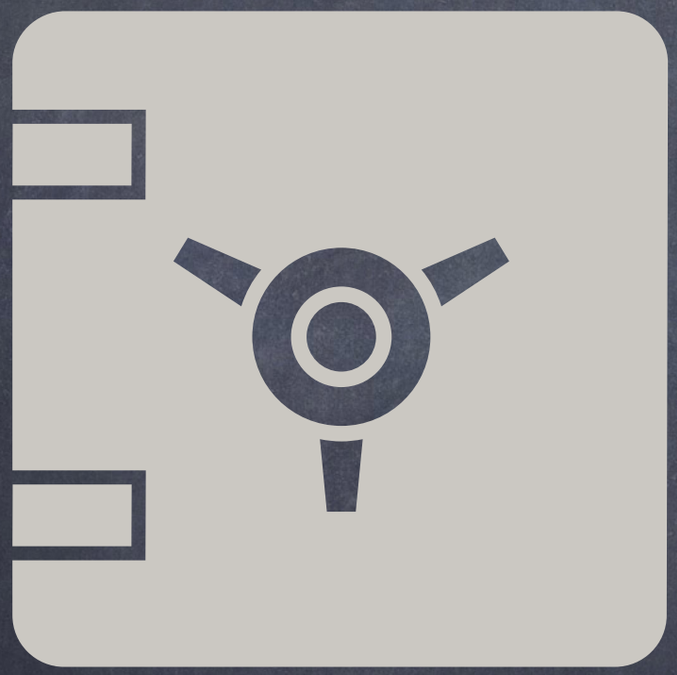
Default Value
Burn Address

Tokens Burnt
Contract Locked

Zero-address Checks
Address Parameters

#50

Critical Addresses



Critical Addresses
Change Value

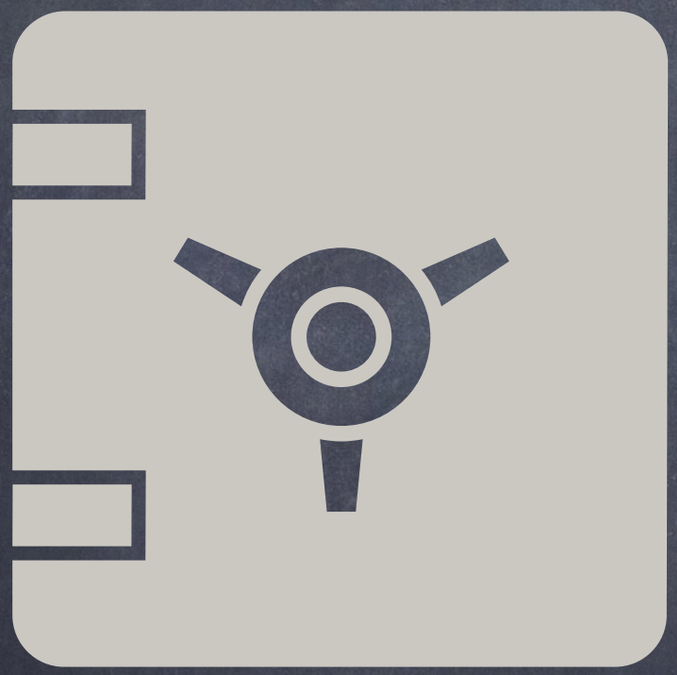
Single-step Change
Error -> Contract Locked

Two-step Change
Grant/Approve + Claim

Error Recovery
Risk Mitigation

#51

assert()



Use of assert()

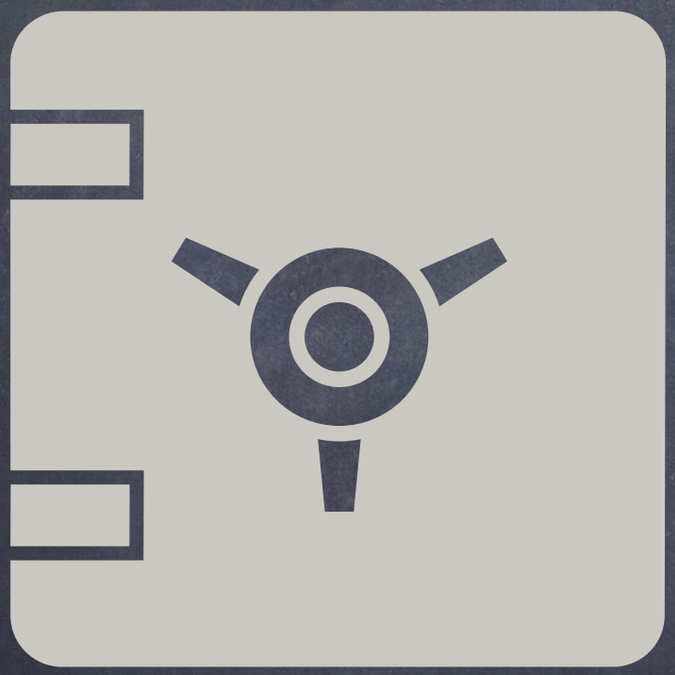
Assert Invariants

No State Change
Not Valiate Inputs

No assert() Failure
Normal Contract Function

#52

assert Vs require



assert() Vs require()
Different Usages

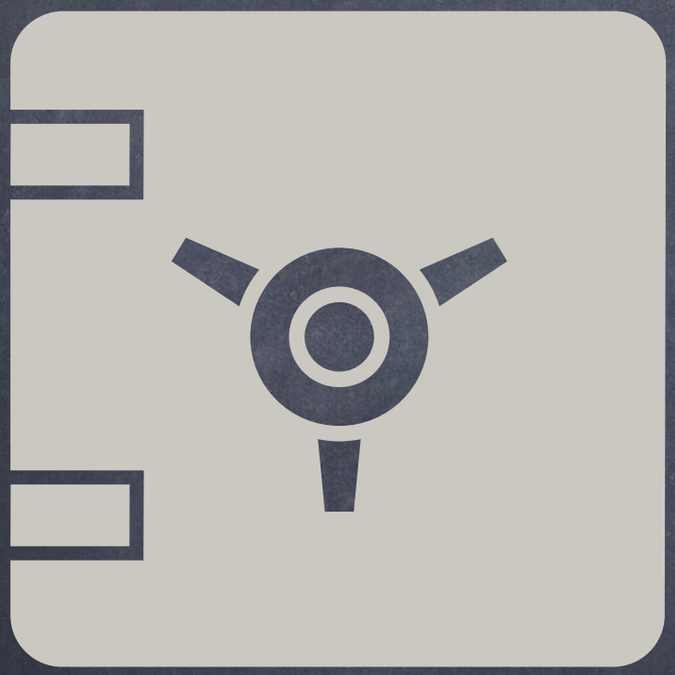
assert() -> Invariants
No Failures Expected

require() -> Validation
Failures Expected

Use Appropriately

#53

Keywords



Deprecated Keywords

E.g.: msg.gas, throw, sha3, constant, var etc.

Compiler Warnings -> Errors

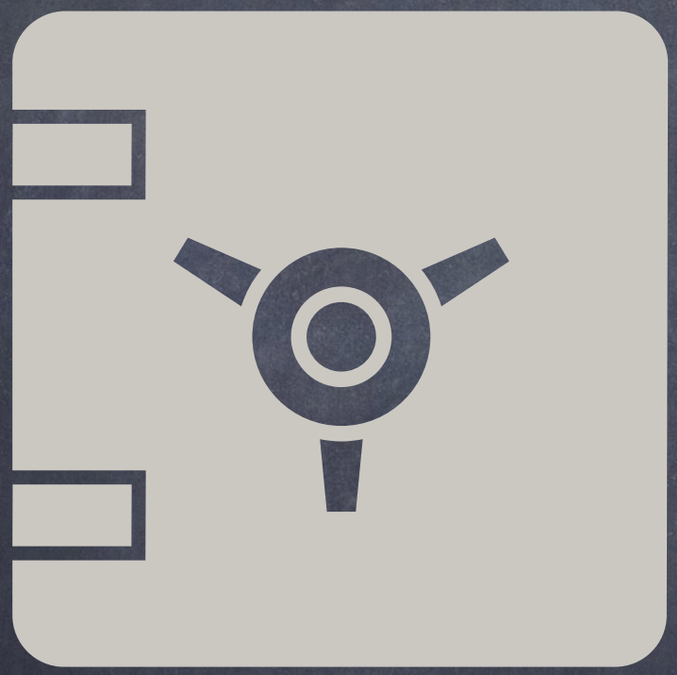
Avoid Deprecated Keywords

#54

Visibility

Functions: public/ external/internal/private

solc < 0.5.0
Default: public

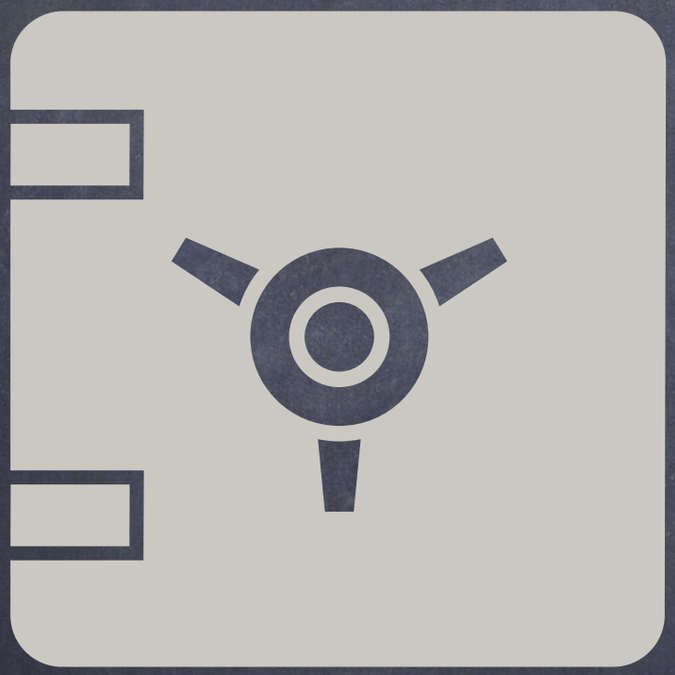


Unauthorized Access
Removed -> solc 0.5.0

Explicit Function Visibility

#55

Inheritance



Inheritance Order

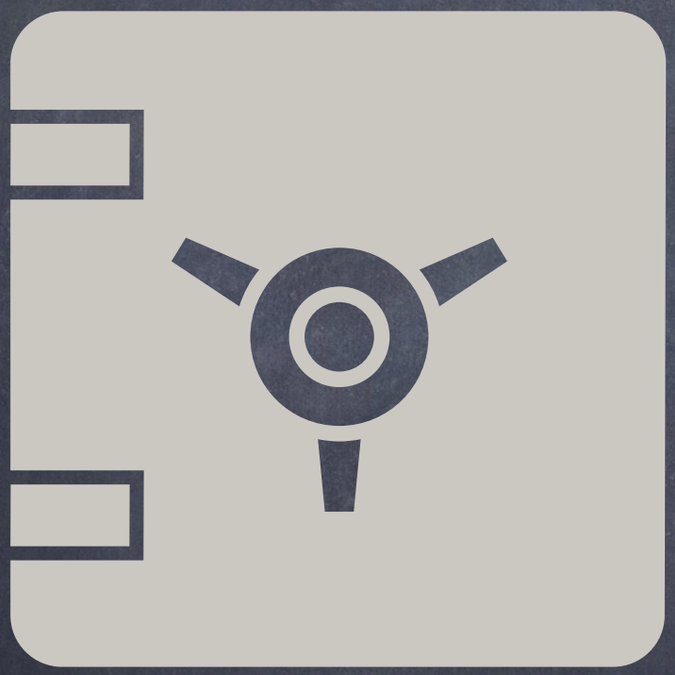
Multiple Contracts
Identical Function

Order -> Implementation

More General ->
More Specific

#56

Inheritance



Missing Inheritance

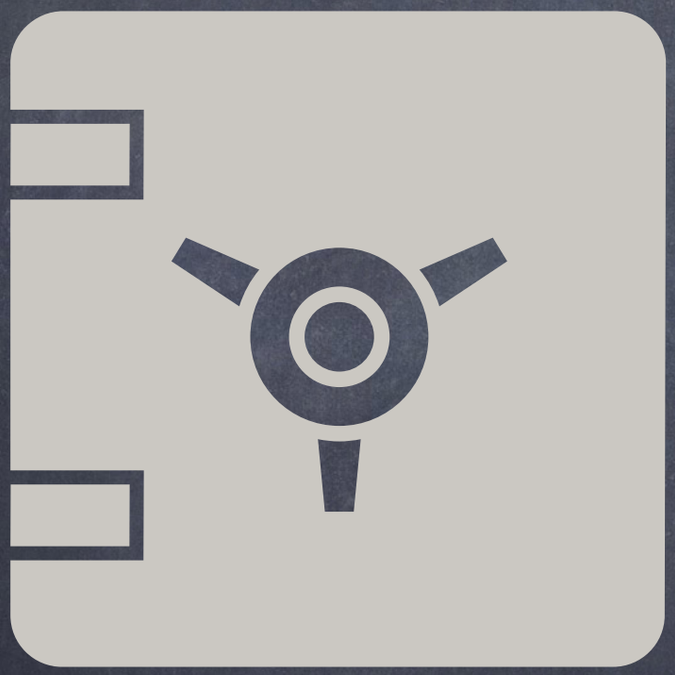
Contract Name
Contract Functions

Appear -> Interface or
Abstract Contract

Inherit Appropriately

#57

Gas Griefing



Transaction Relayers

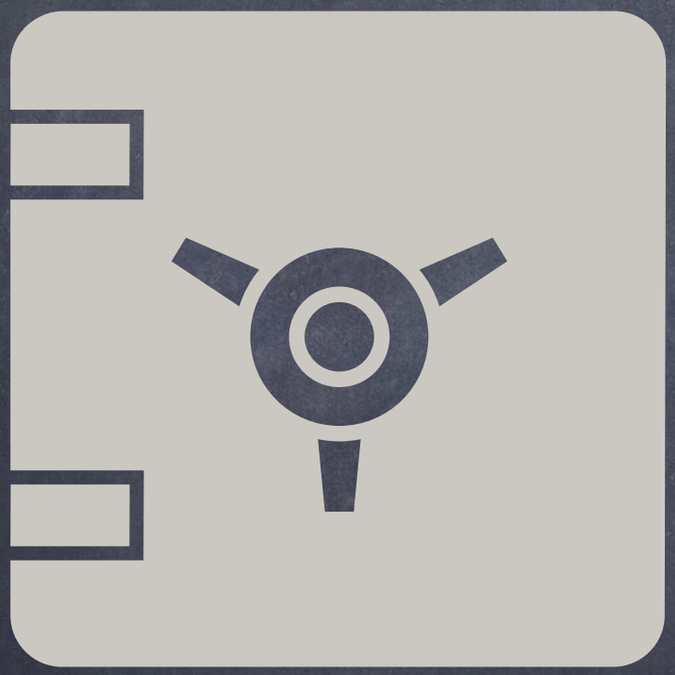
Users -> Forward Txs -> Blockchain

Tx -> Sufficient Gas
Tx Success/Fail

Trust in Relayers

#58

Reference Parameters



Function Parameters
Reference Types

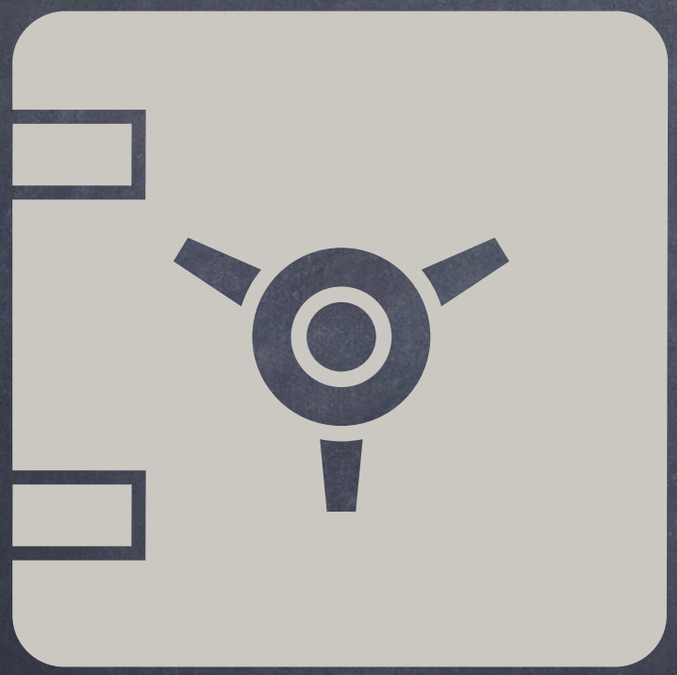
Pass by Value
Pass by Reference

Structs/Arrays/Mappings
Memory Vs Storage

Unexpected Changes
Aware: Value/Reference

#59

Arbitrary Jump



Function Type
Arbitrary Jump

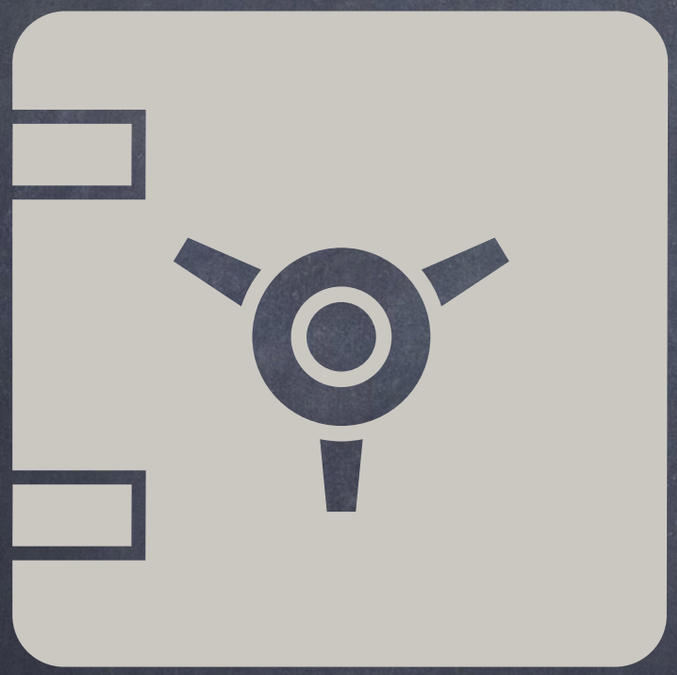
Variable Assignment
Assembly Code

Function Variable ->
Arbitrary Code

Avoid Assembly
Arbitrary Function Values

#60

Hash Collisions



abi.encodePacked()
Variable Length Arguments

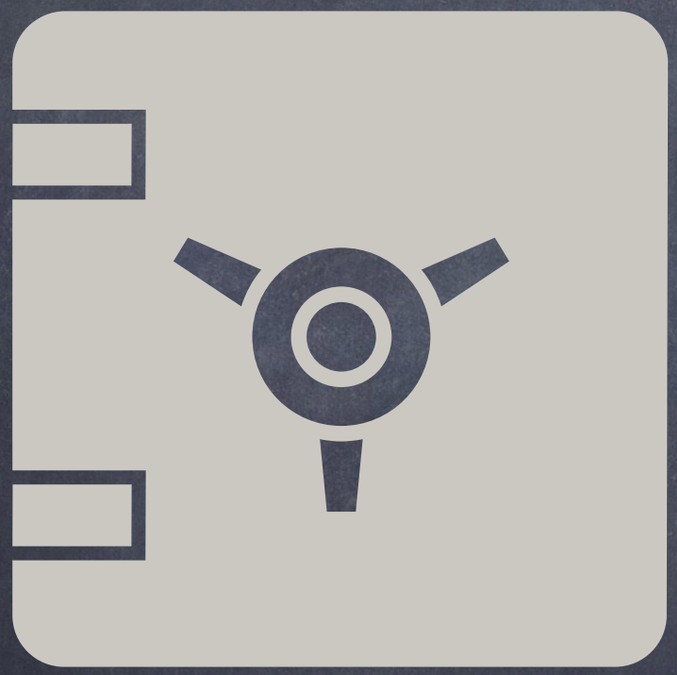
No Zero-padding
No Length

Packed Encoding ->
Collisions Possible

abi.encode()
No Dynamic/Taint

#61

Dirty Bits



Types < 32 Bytes

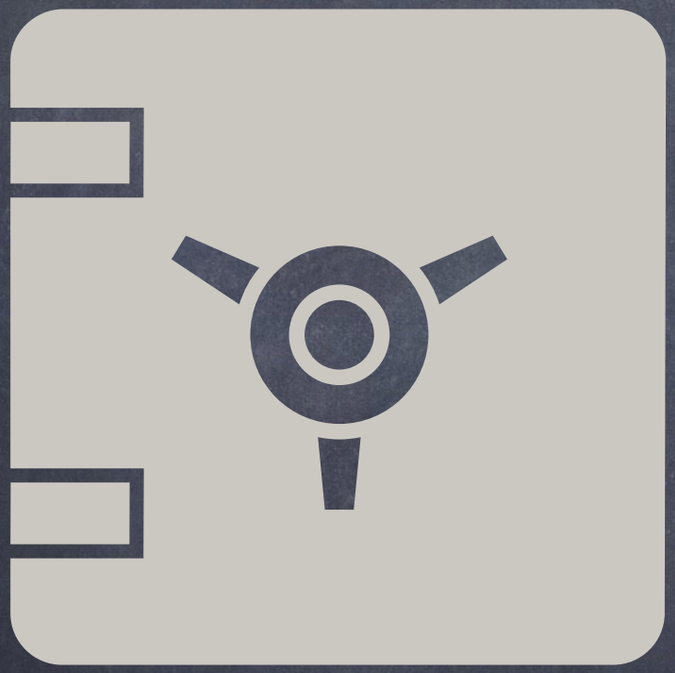
Dirty High Order Bits

Type Operations -> Ok
msg.data -> Not Ok

Risk Awareness

#62

Incorrect Shifts



Shift Operators

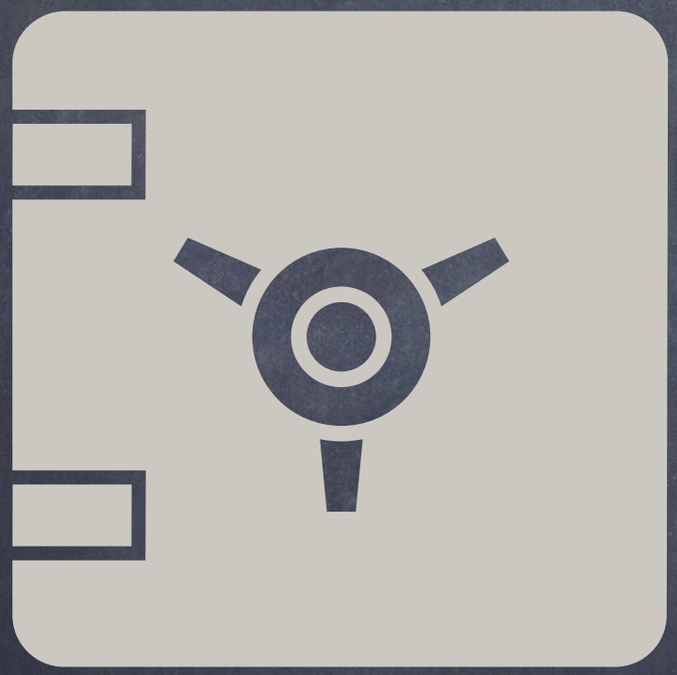
shl(x,y), shr(x,y), sar(x,y)

Shift y by x Bits

Confusing -> Exchanged Check Shifts

#63

Assembly



Solidity Assembly

Optimized -> Efficient
Gas Savings

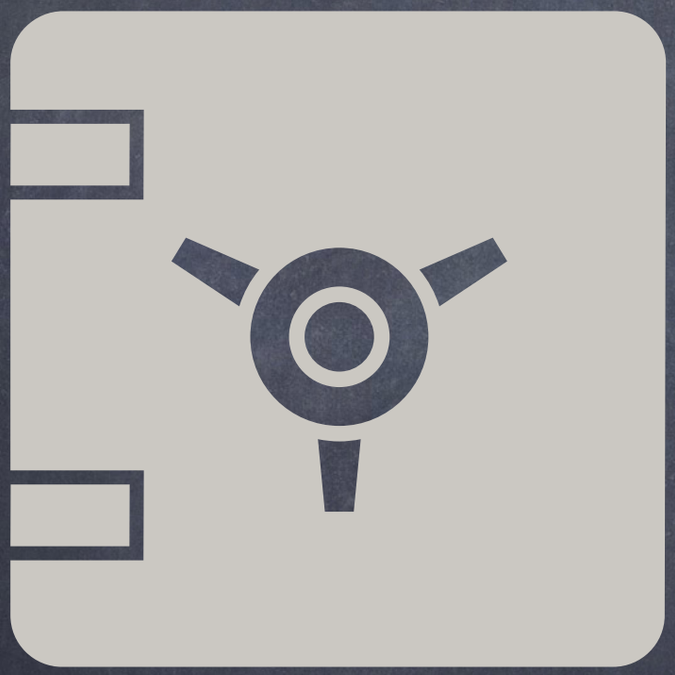
Error-prone
Semantics & Readability

Avoid or
Double-check

#64

RTLO

Right-to-Left-Override



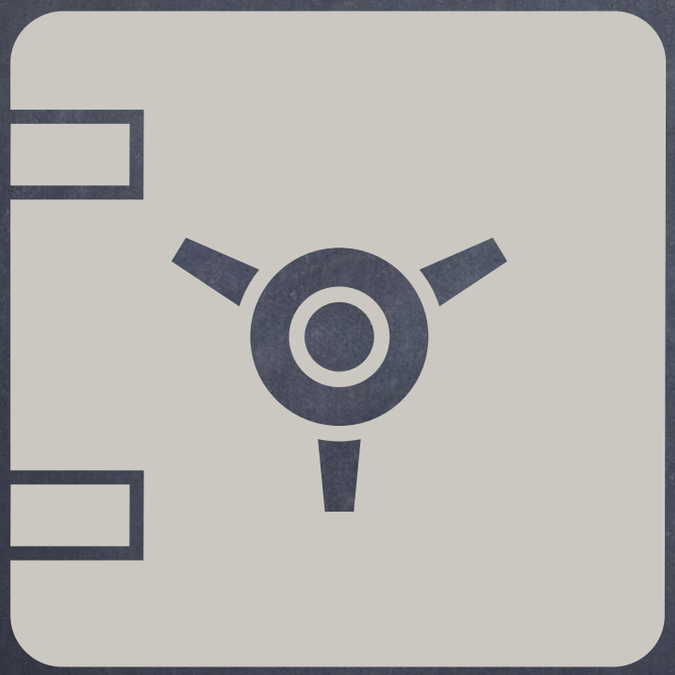
Unicode U+202E

RTL Text Rendering
Trick Users/Auditors

Ensure Absence of RTLO

#65

Constant



Constant State Variables

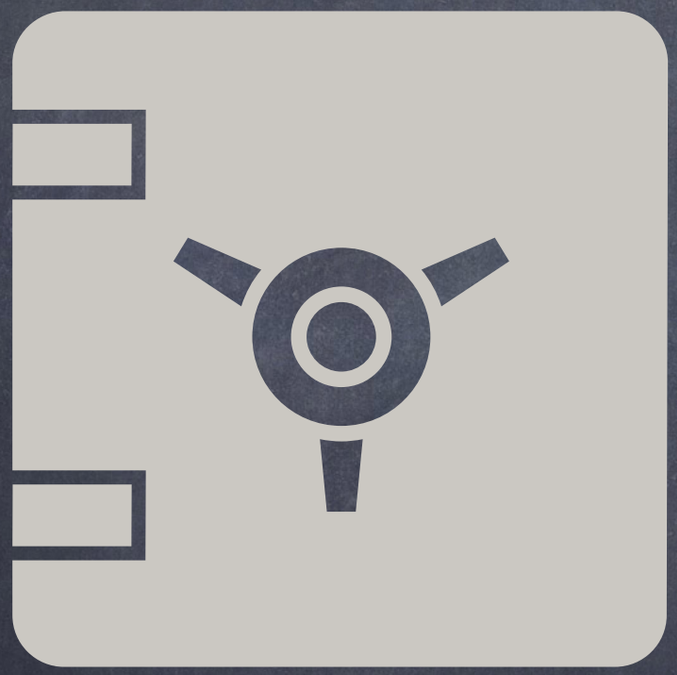
Declare Constant
Save Gas

Compile-time
Replacement -> No SLOADs

Identify & Declare

#66

Variables



Variable Names

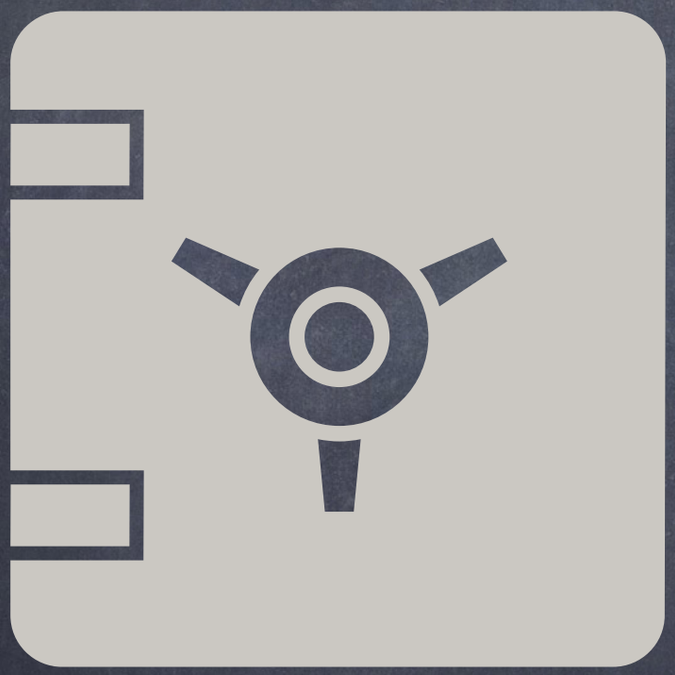
Similar -> Confusing

Replaced Usages
Readability

Distinct Names
Avoid Errors

#67

Variables



Uninitialized
State/Local Variables

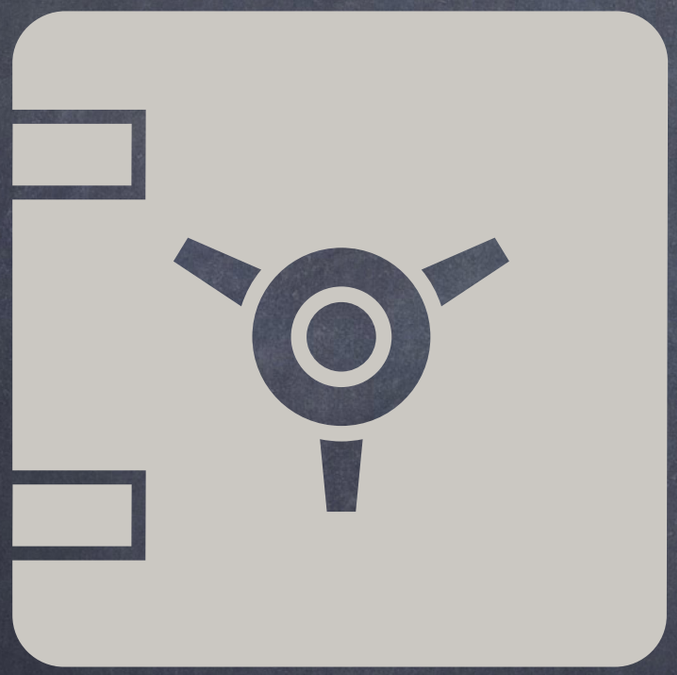
Default Values -> Zero

Address -> Zero
Bool -> False

Initialize Variables
Avoid Errors

#68

Storage Pointers



Uninitialized Storage Pointers

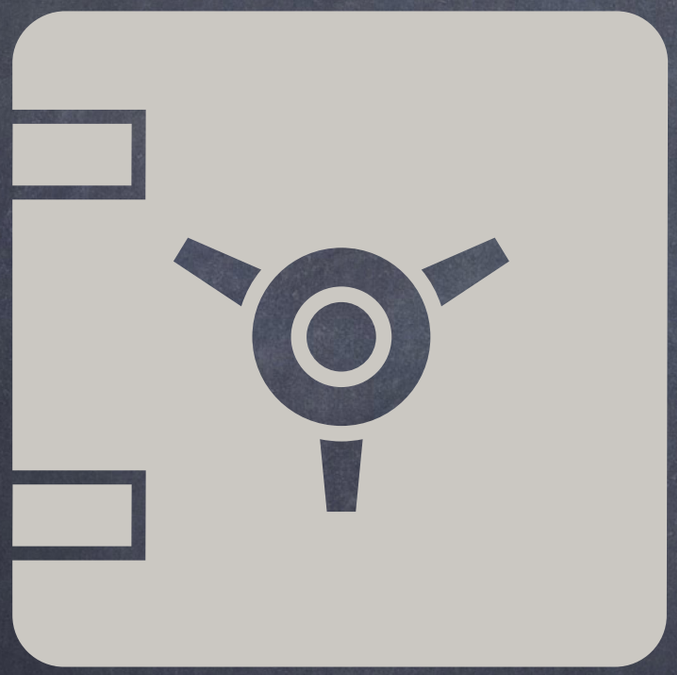
Local Storage Variables Uninitialized

Pointers -> Unexpected Storage Locations

Error-prone
solc >= 0.5.0 Disallowed

#69

Function Pointers



Uninitialized Function Pointers

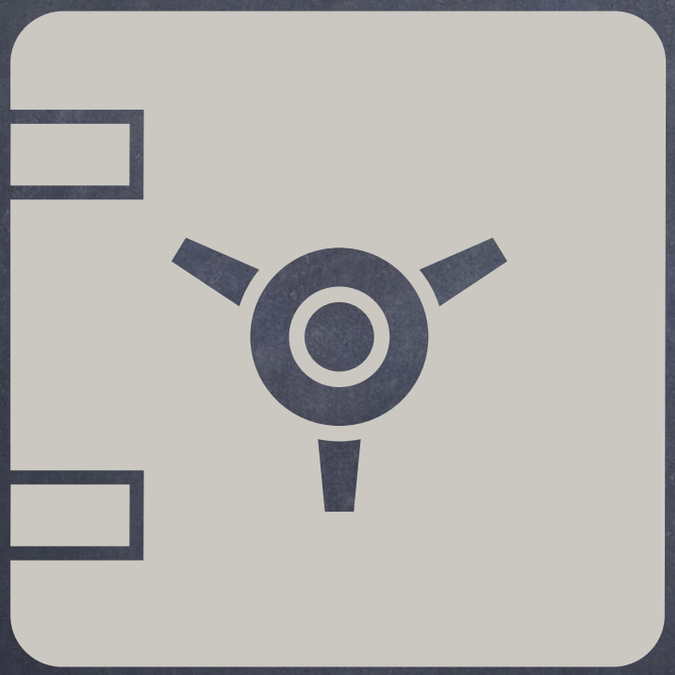
Constructors -> Unexpected Behavior

solc 0.4.5-0.4.26
solc 0.5.0-0.5.7

Compiler Bug -> Fixed

#70

Number Literals



Long Number Literals

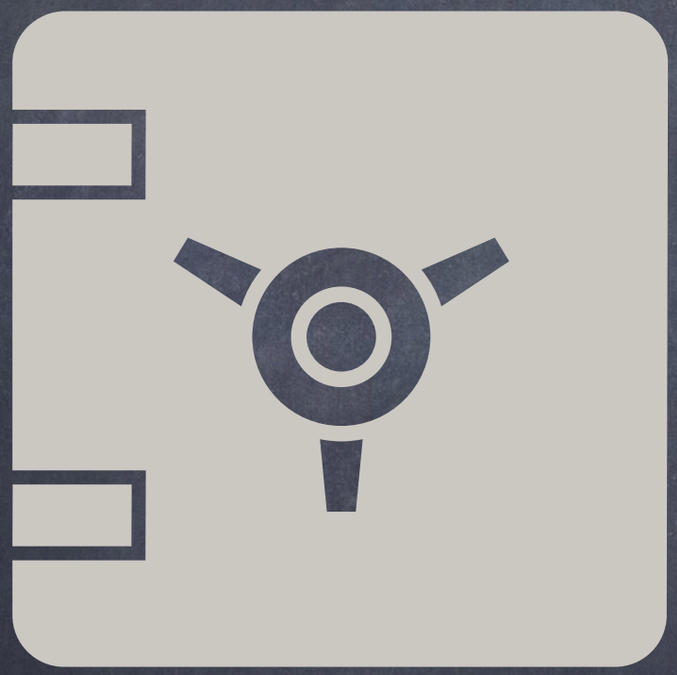
Too Many Digits -> Error-prone

E.g.: uint 1_ether = 10000000000000000000;

Use Ether/Time Suffix
Use Scientific Notation

#71

Enum



Out-of-range Enums

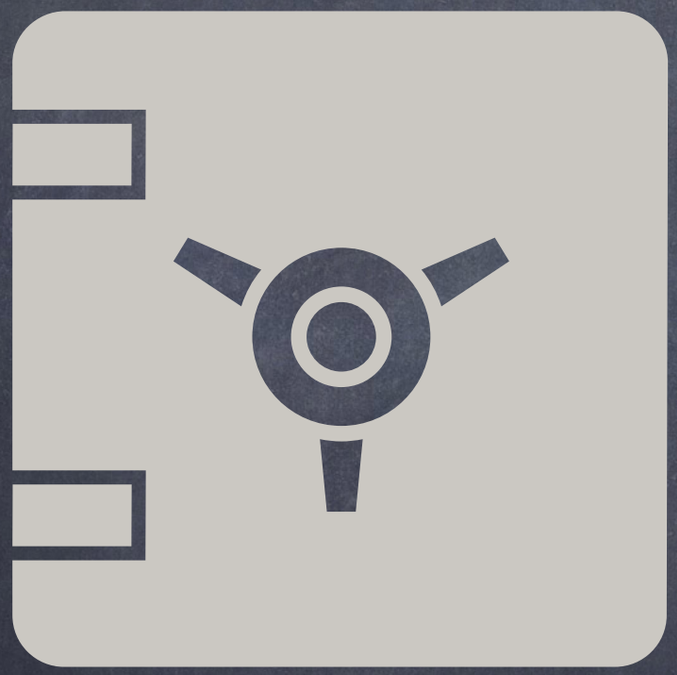
enum E{a}
E(1) -> Out-of-range

solc < 0.4.5
Unexpected Behavior

Compiler Bug -> Fixed
Check Enums

#72

Public Functions



Function Visibility
Public Vs External

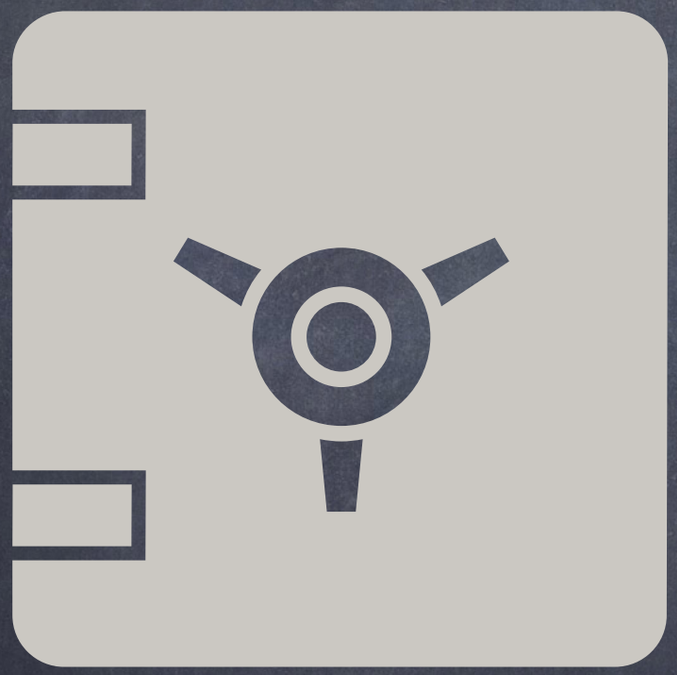
Public -> More Gas

Function -> No Calls
Within Contract

Public -> External
Save Gas

#73

Dead Code



Contract Code
Dead or Unreachable

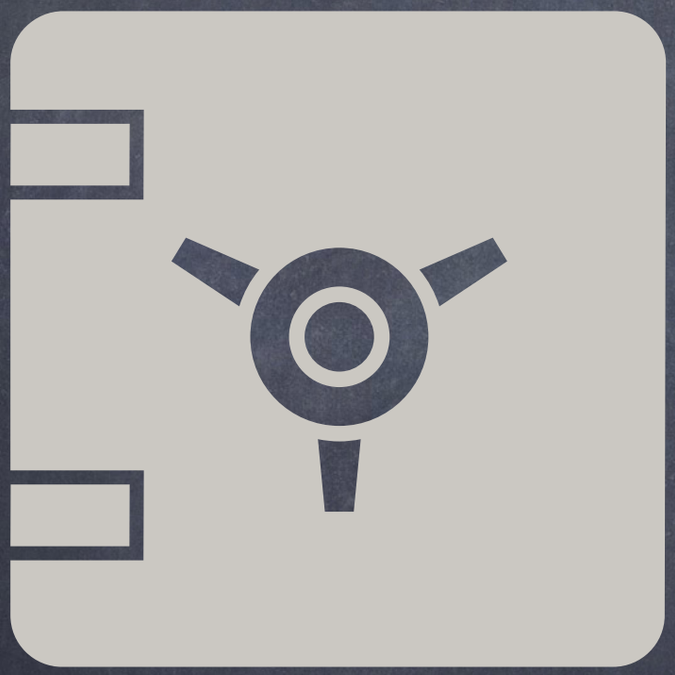
Programmer Error
Missing Logic

Optimization
Code Size -> Deploy Cost

Use Dead Code or
Remove Dead Code

#74

Return Value



Function Return Values

Call Sites -> Unused

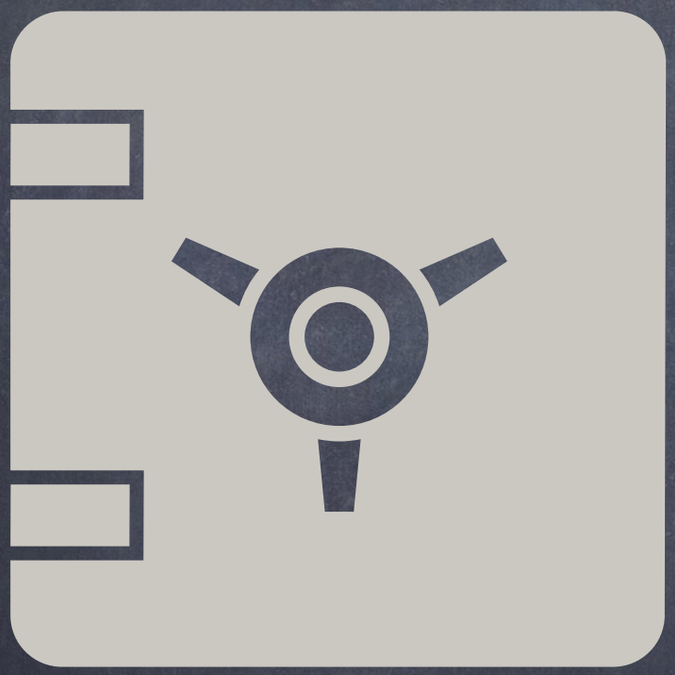
Missed Error Checking
Unexpected Behavior

Return Values
Check or Remove

#75

Variables

State/Local Variables



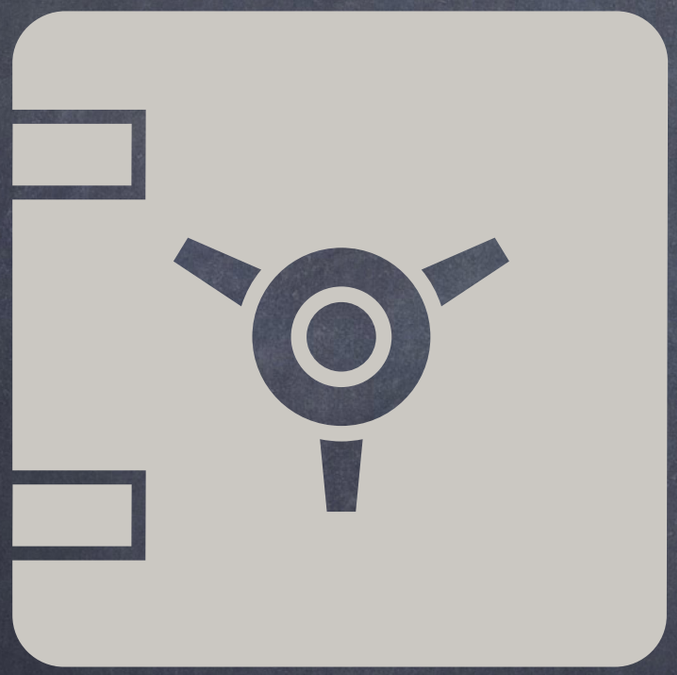
Declared & Unused

Error -> Missing Logic
or Optimization

Variables
Use or Remove

#76

Statements



Function Statements

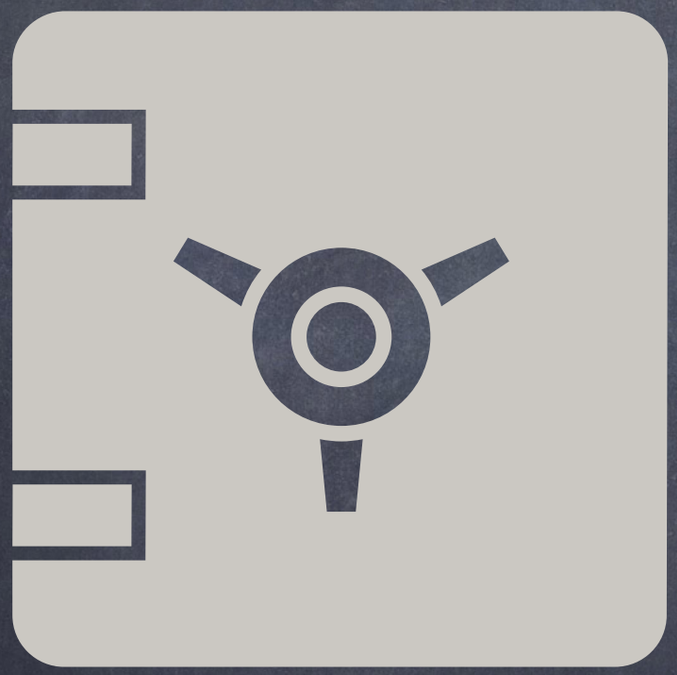
Present & No Effects

Error -> Missing Logic
or Optimization

Redundant Statements
Effect or Remove

#77

Storage Array



Storage Array
Signed Integers

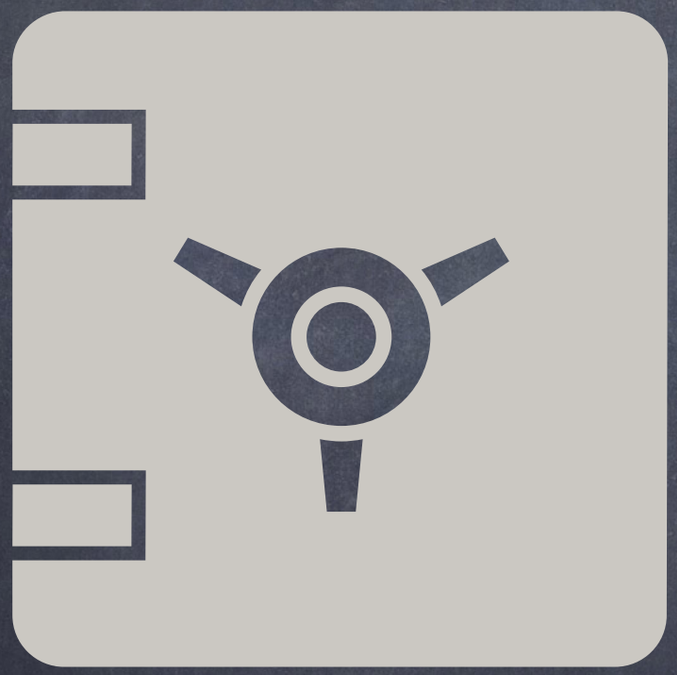
ABIEncoderV2
Type[] = int[]

Assignment -> Data Corruption

Compiler Bug -> Fixed
solc 0.4.7 -> 0.5.10

#78

Constructor Arguments



Constructor Dynamic Arguments

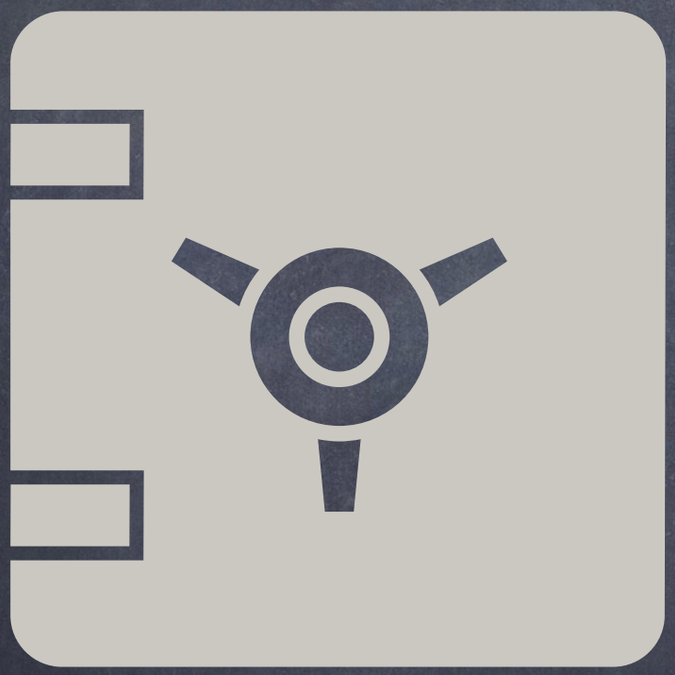
ABIEncoderV2 Dynamic Arrays

Revert or Invalid Decode

Compiler Bug -> Fixed
solc 0.4.16 -> 0.5.9

#79

Arrays



Storage Arrays

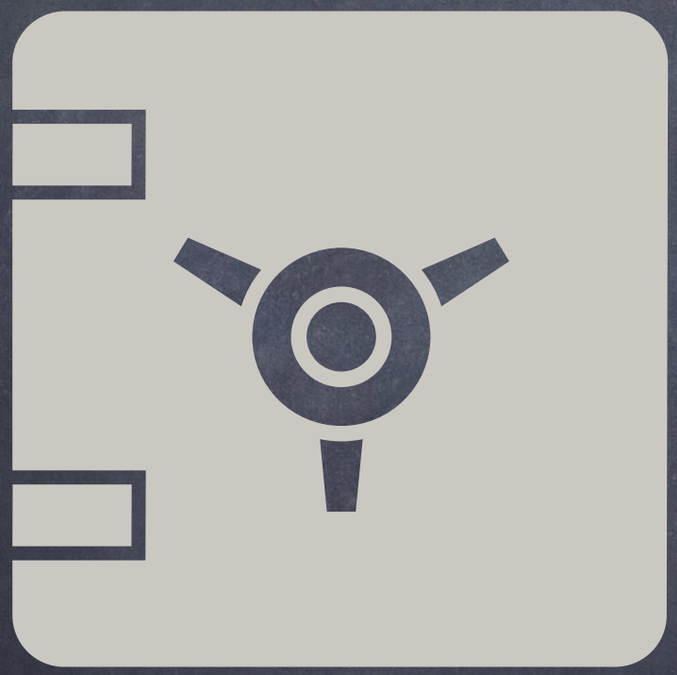
ABIEncoderV2
Structs or Static Arrays

Invalid Encode
External Fn/abi.encode()

Compiler Bug -> Fixed
solc 0.4.16 -> 0.5.10

#80

Structs



Calldata Structs

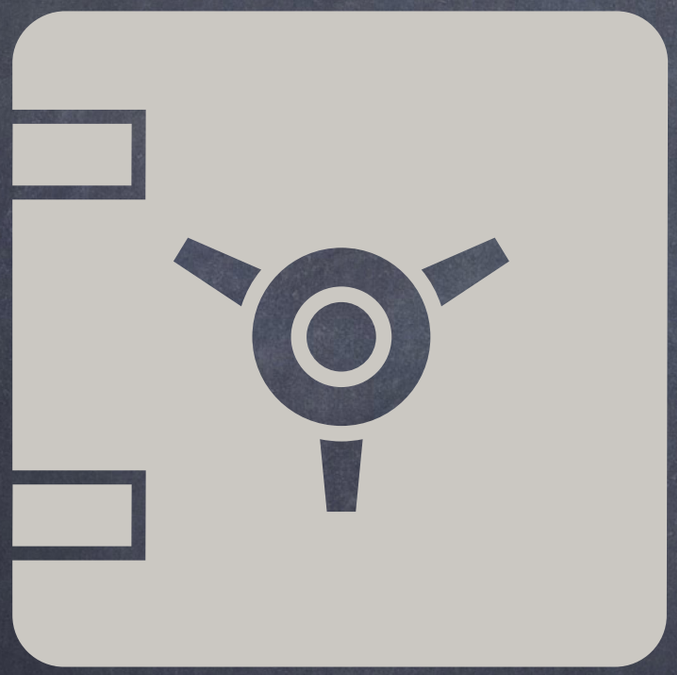
ABIEncoderV2
Static Size & Dynamic Encode

Read -> Incorrect Values

Compiler Bug -> Fixed
solc 0.5.6 -> 0.5.11

#81

Storage



Packed Storage

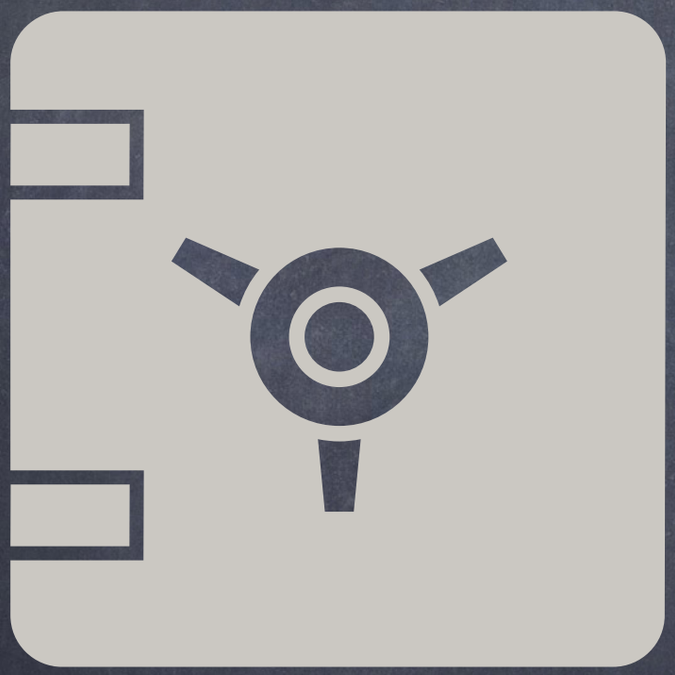
ABIEncoderV2
Structs/Arrays -> Type < 32B

Storage Encode -> Data Corruption

Compiler Bug -> Fixed
solc 0.5.0 -> 0.5.7

#82

Loads



Incorrect Loads
YUL Optimizer

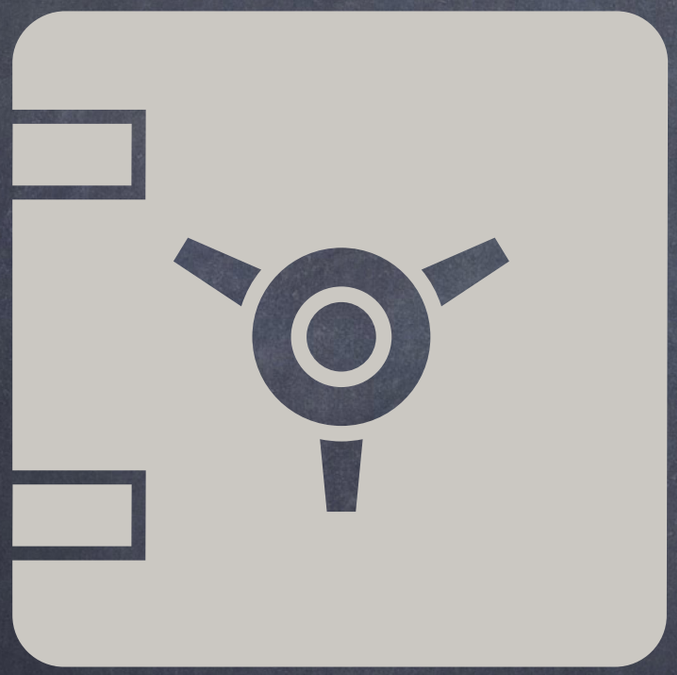
ABIEncoderV2 + YUL
MLOAD/SLOAD Calls

Calls Replaced -> Stale Values

Compiler Bug -> Fixed
solc 0.5.14 -> 0.5.15

#83

Arrays



Array Slices

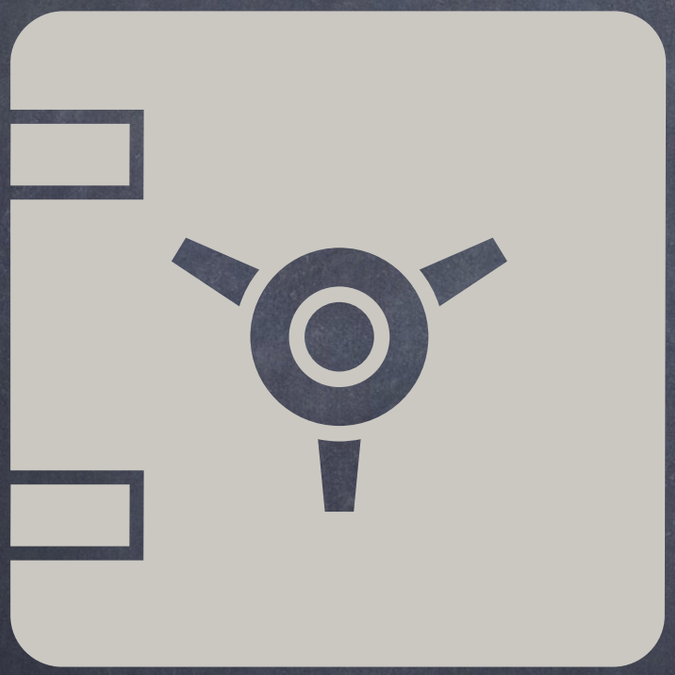
ABIEncoderV2
Dynamically Encoded Base Types

Read -> Invalid Data

Compiler Bug -> Fixed
solc 0.6.0 -> 0.6.8

#84

Escaping



Missed Escaping

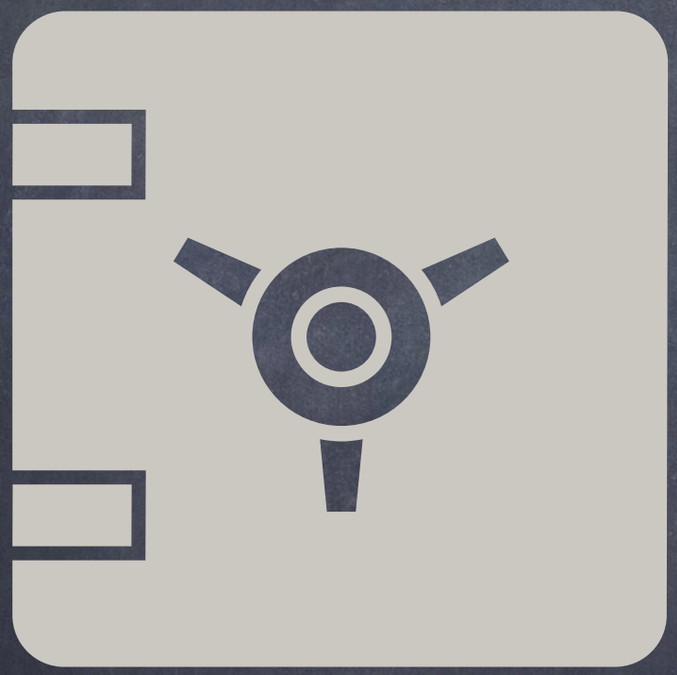
ABIEncoderV2
String Literals

\\ -> Different String

Compiler Bug -> Fixed
solc 0.5.14 -> 0.6.8

#85

Shift



Double Shift Size
Overflow

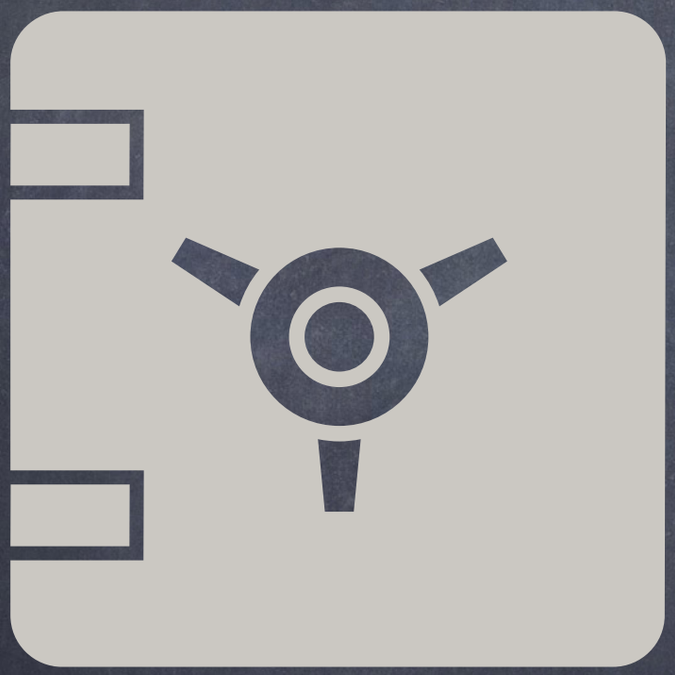
Optimizer
Double Bitwise Shifts

Shift > 2**256 -> Overflow

Compiler Bug -> Fixed
solc 0.5.5 -> 0.5.6

#86

byte
Instruction



byte Opcodes
Second Arg = 31

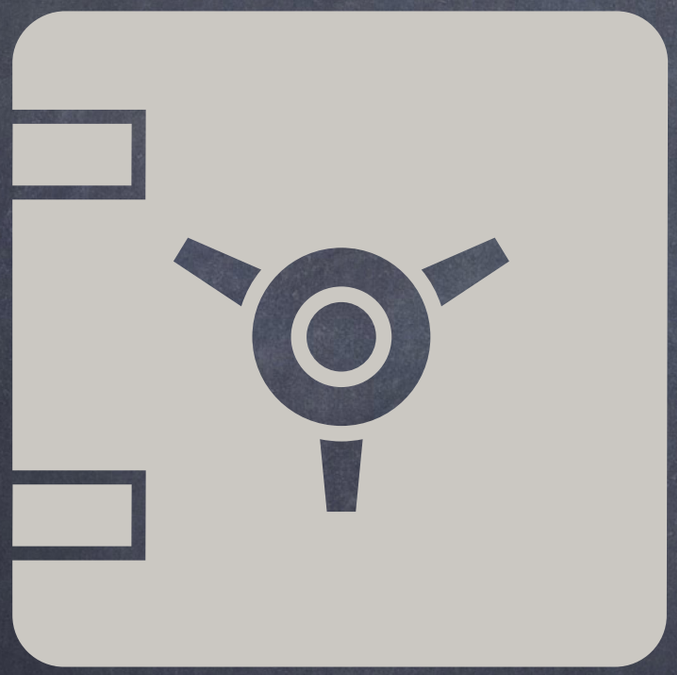
Optimizer
bytesNN Index or byte

Unexpected Value

Compiler Bug -> Fixed
solc 0.5.5 -> 0.5.7

#87

Assignment



Essential Assignments
Removed

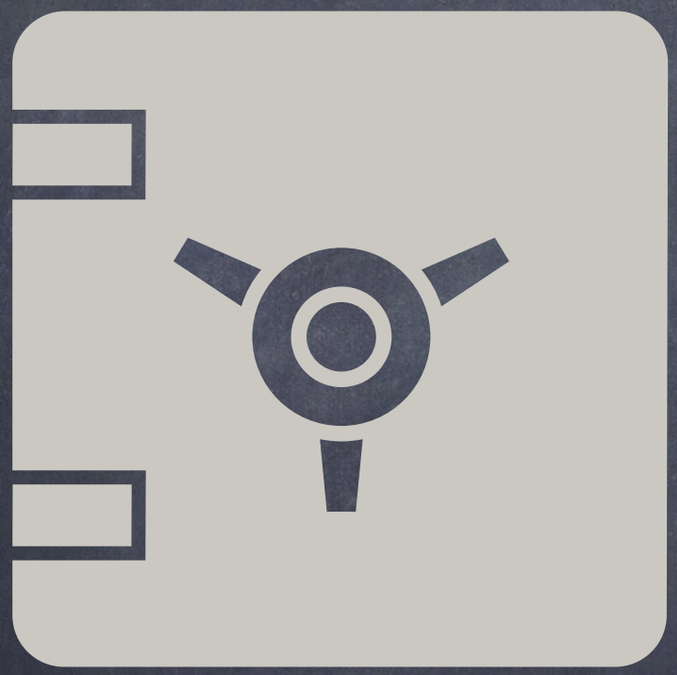
YUL Optimizer
Variables -> for Loops

continue/break
Assignments -> Removed

Compiler Bug -> Fixed
solc 0.5.8/0.6.0 -> 0.5.16/0.6.1

#88

Private Functions



Private Functions
Base -> Derived

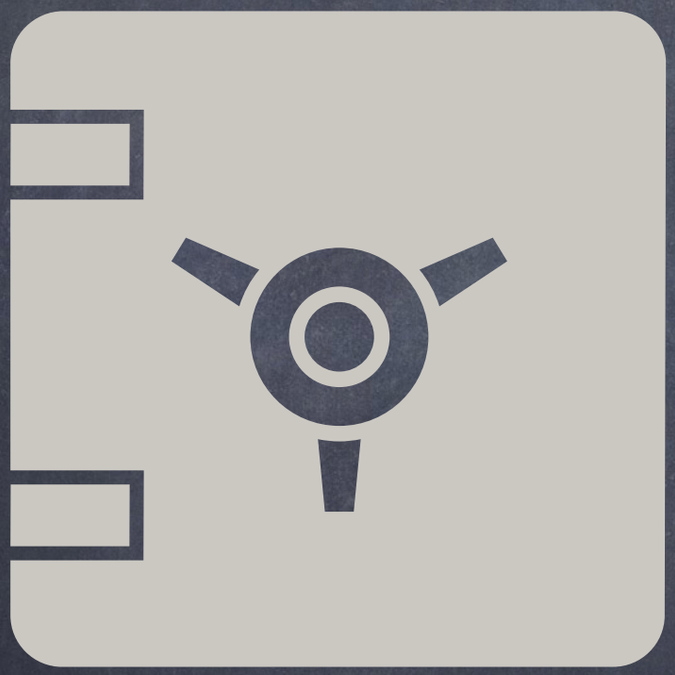
Not Visible -> Derived
Function -> Same Name/Type

Base Private Function
Change Behavior

Compiler Bug -> Fixed
solc 0.3.0 -> 0.5.17

#89

Tuples



Tuple Assignments

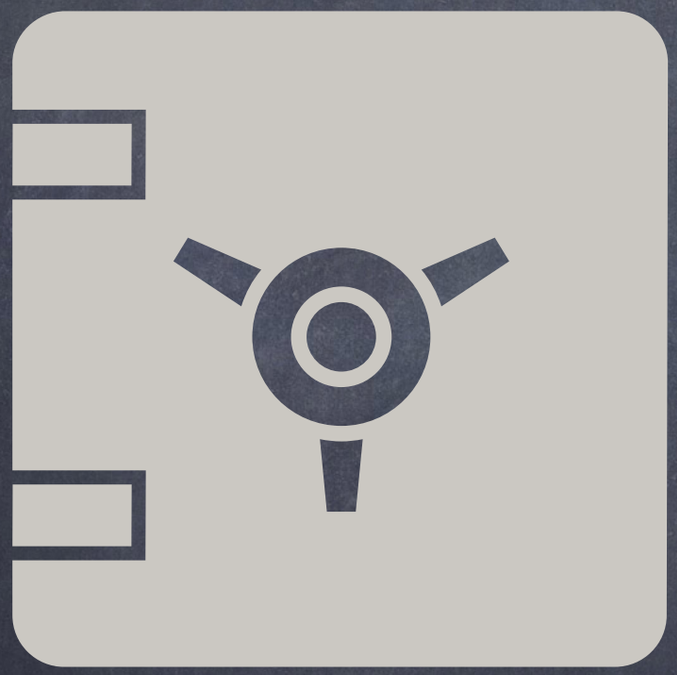
Multiple Stack Slots
E.g.: Nested Tuples

Invalid Values

Compiler Bug -> Fixed
solc 0.1.6-> 0.6.6

#90

Arrays



Dynamic Arrays
Cleanup

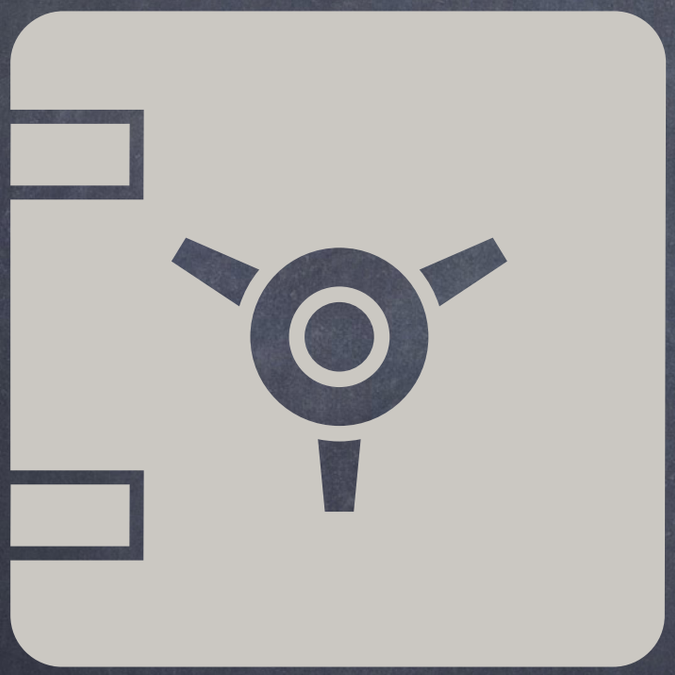
Dynamic Arrays
Types <= 16 Bytes

Assigned Array Shrinks
Deleted Slots -> Not Cleaned

Compiler Bug
Fixed -> 0.7.3

#91

byte Array



Empty Byte Array Copy

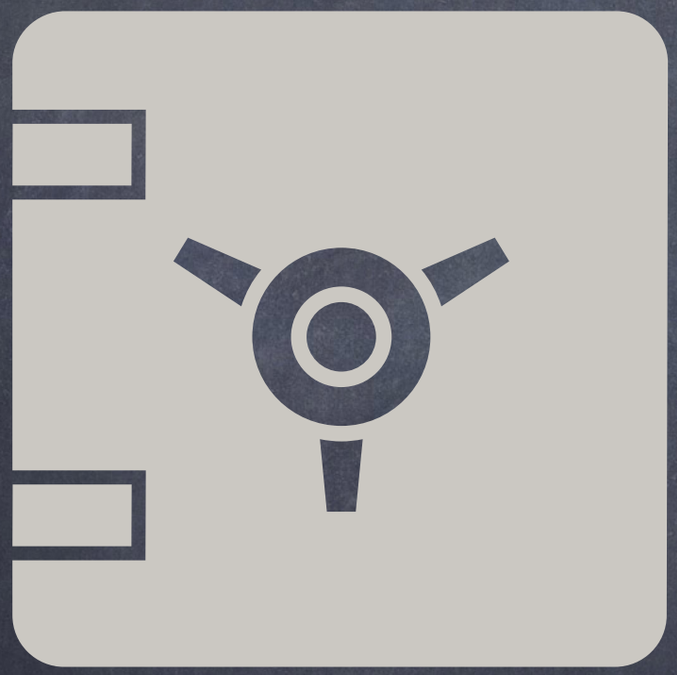
Memory/Calldata -> Storage

Increase Length -> No New Data -> Corruption

Compiler Bug
Fixed -> 0.7.4

#92

Memory Array



Memory Array Creation Overflow

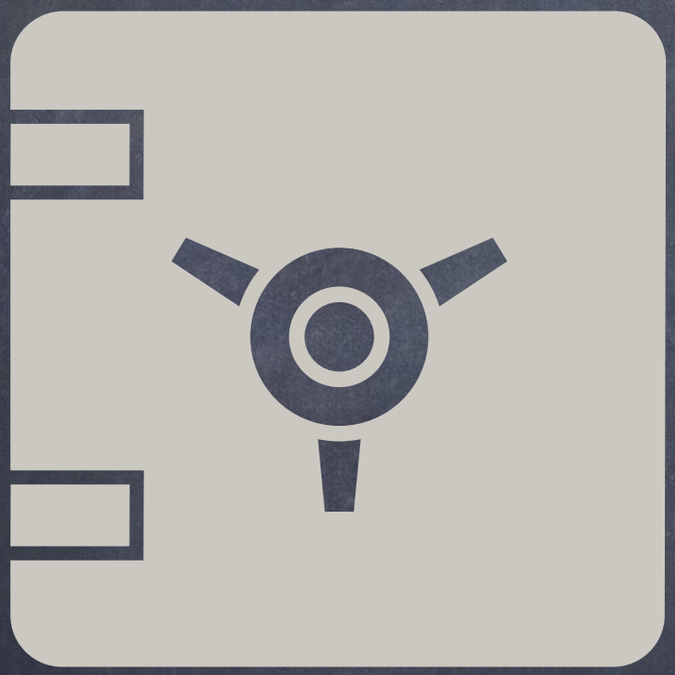
Very Large Memory Arrays

Overlapping Memory -> Corruption

Compiler Bug -> Fixed
solc 0.2.0 -> solc 0.6.5

#93

using for



Calldata Parameters
using for

Function Calls -> Internal
Library Functions

Read -> Invalid Data

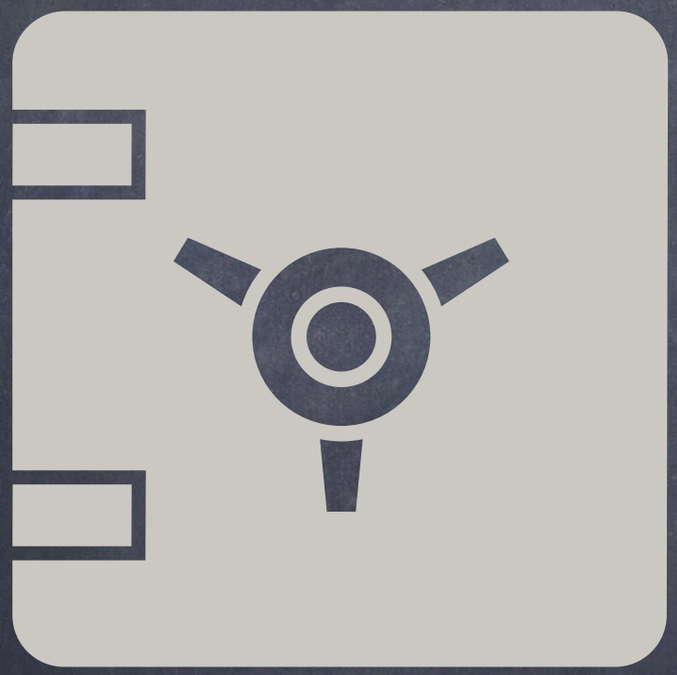
Compiler Bug -> Fixed
solc 0.6.9 -> solc 0.6.10

#94

Free Functions

Free Functions Redefinition

Free Functions -> Outside Contract

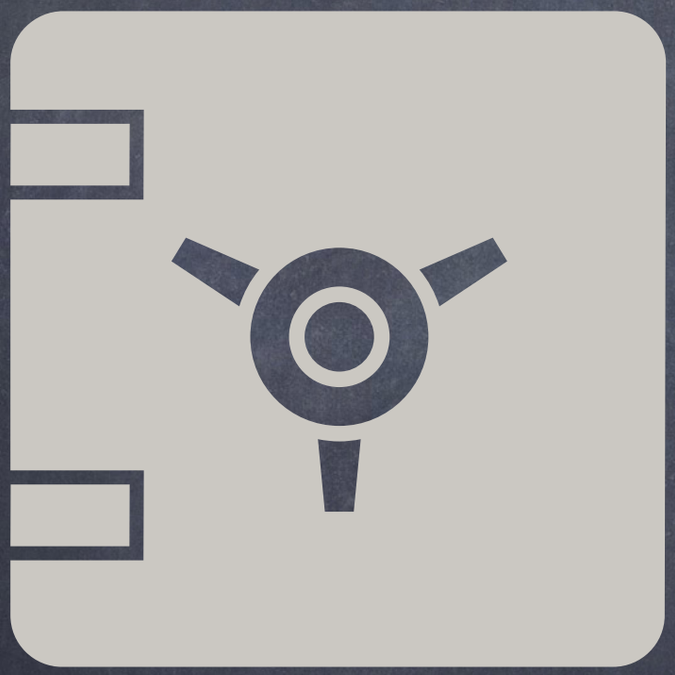


Same Name/Parameters -> No Compiler Error

Compiler Bug -> Fixed
solc 0.7.1 -> solc 0.7.2

#95

Initializers



Unprotected Initializers

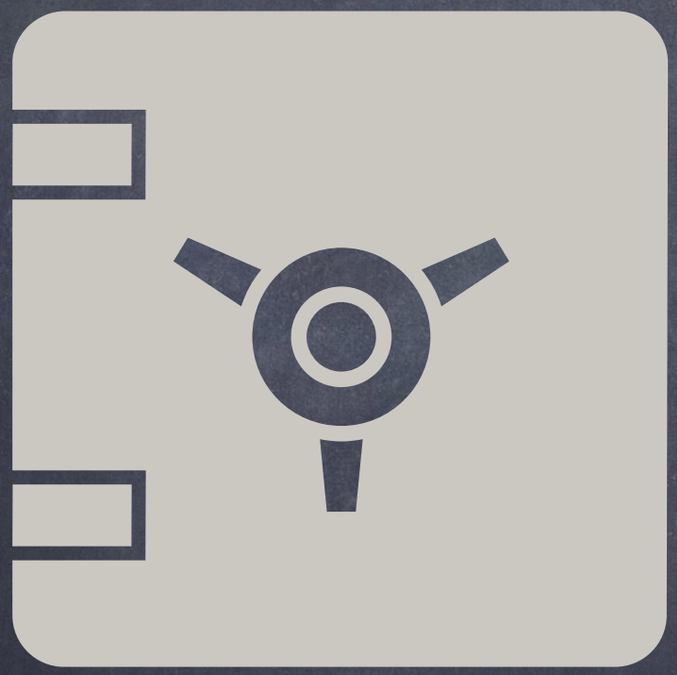
Proxy-based Contracts
Constructors -> Initialize()

Multiple Invocations ->
Reinitialize Contract

OZ Initializable Library
Initializer Modifier

#96

State Variables



State Variables Initialized

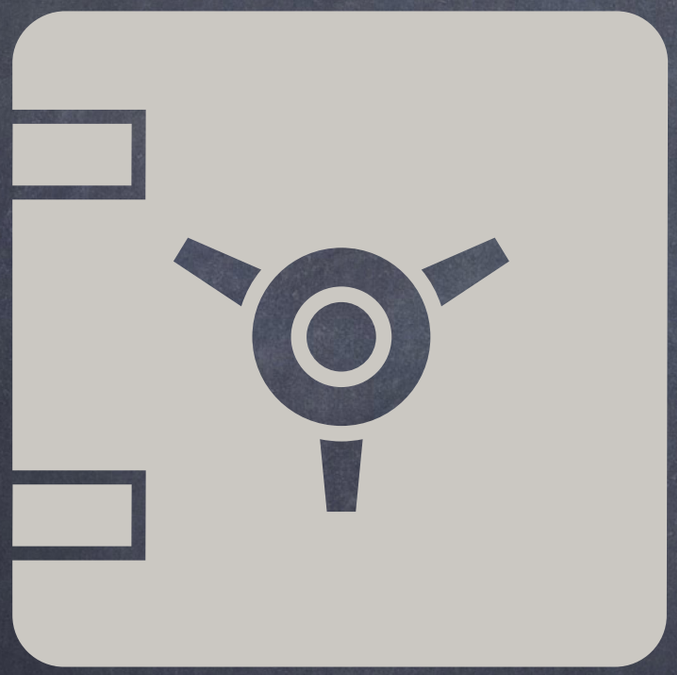
Proxy-based Contracts
Constructors -> Initialize()

State Variables
Declare -> Initialize

Initialize -> Initialize()
Not Set Otherwise

#97

Import Contracts



Imported Contracts

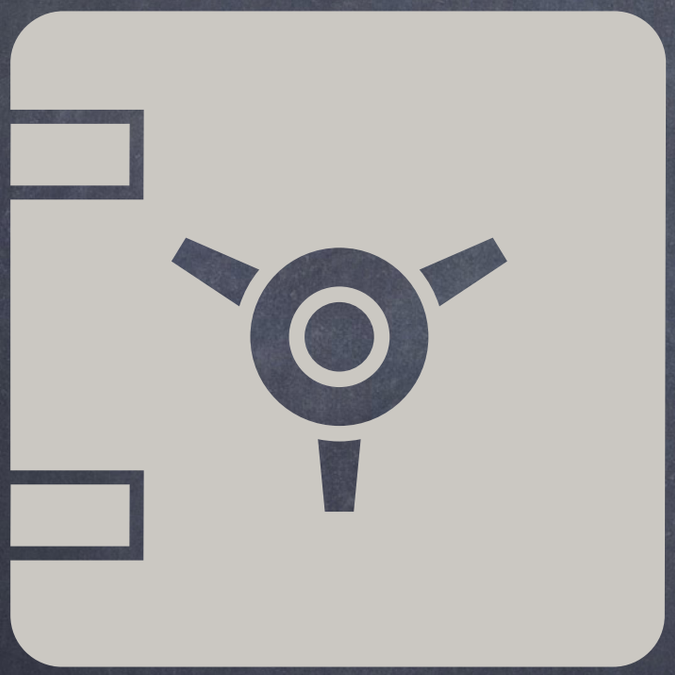
Proxy-based Contracts -> Proxy-based Imports

Constructor -> Initialize()

Ensure Imports -> Proxy-based

#98

selfdestruct



selfdestruct()
delegateCall()

Proxy-based Contracts
Data Proxy -> Logic Impl

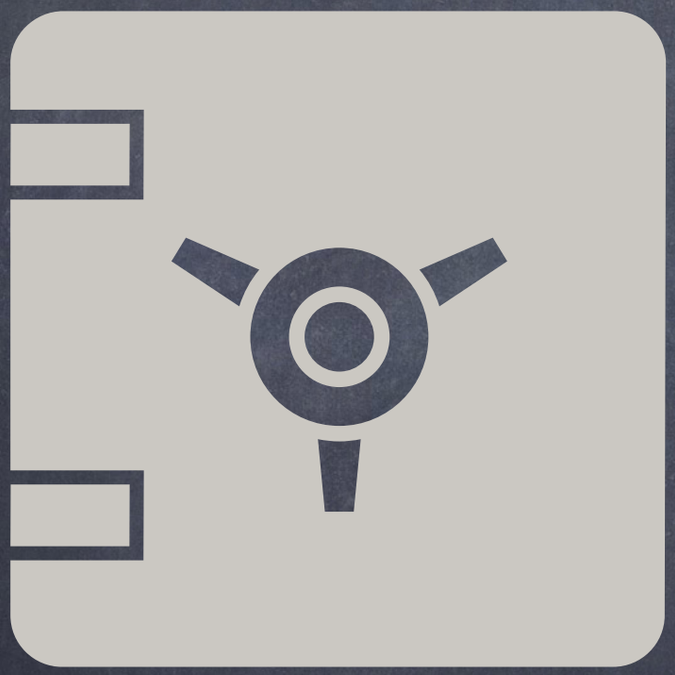
Logic Contract Destructed
Address -> No Contract

Avoid selfdestruct()
Avoid delegateCall()

#99

State Variables

State Variables Order/ Layout/Type/Mutability



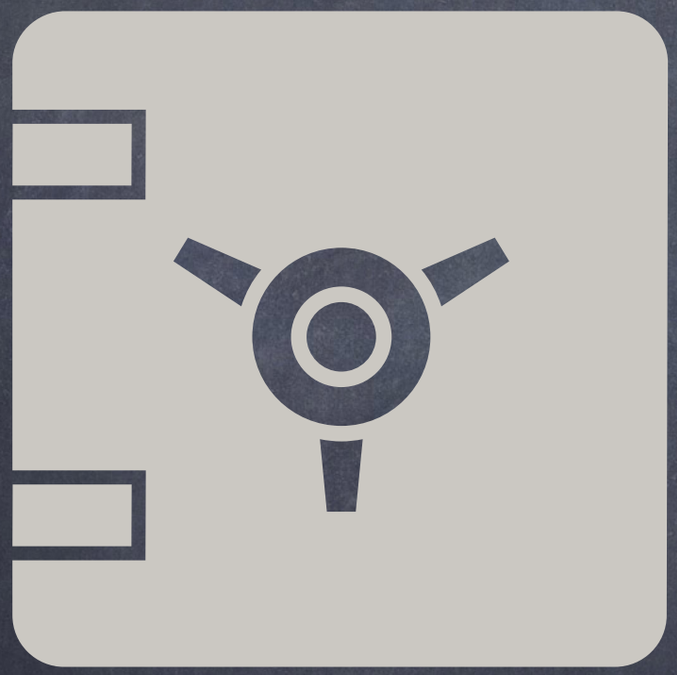
Proxy-based Contracts
Data Proxy -> Logic Impl

Proxy <-> Impl
Not Same -> Critical Errors

Same -> Order/Layout/Type/ Mutability

#100

Function ID



Function ID Collision

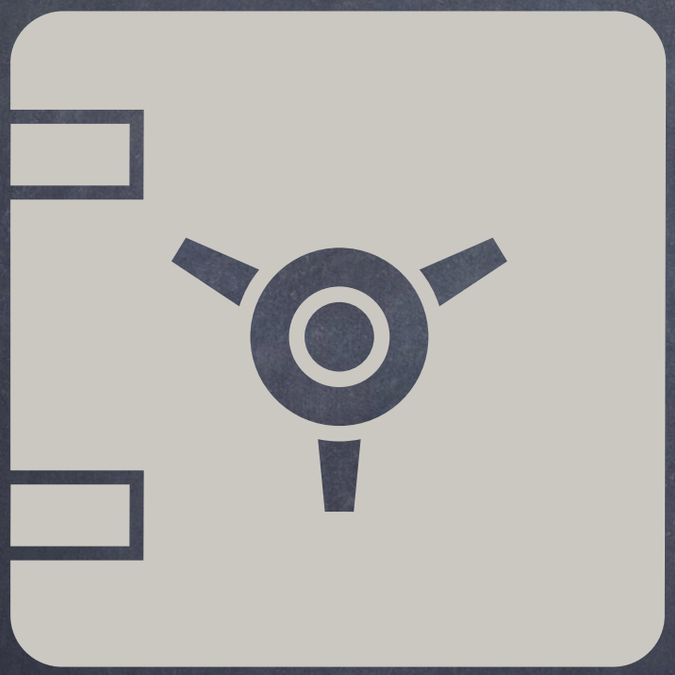
Proxy-based Contracts
Data Proxy -> Logic Impl

Malicious Proxy -> Same
Function ID -> Hijack Call

Untrusted Proxy
Function ID Collision

#101

Shadowing



Function Shadowing

Proxy-based Contracts
Data Proxy -> Logic Impl

Proxy -> Function Shadow
-> Intercept Call

Proxy Functions
Names & Shadowing