VeriSmart: A highly precise safety verifier for Ethereum smart contracts

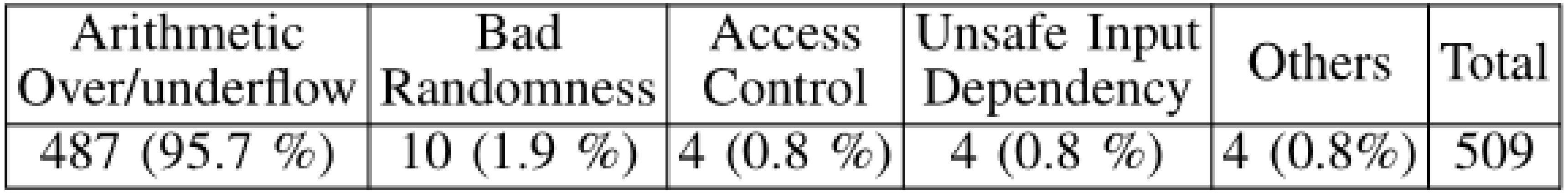
20204222 강우석

Introduction

Introduction

Introduction

- VeriSmart : very smart safety analyzer for verifying Ethereum smart contracts



Arithmetic Over/underflow	Bad Randomness	Access Control	Unsafe Input Dependency	Others	Total
487 (95.7 %)	10 (1.9 %)	4 (0.8 %)	4 (0.8 %)	4 (0.8%)	509

CVE-reported security vulnerabilities of Ethereum smart contracts (05/31/2019)

Introduction

- Important characteristic

1. Automatic
2. Exhaustive
3. Precise

Oyente

Mythril

manticore

Osiris

Using Solidity's SMTChecker

Leonardo Alt

Ethereum Foundation

leonardoalt

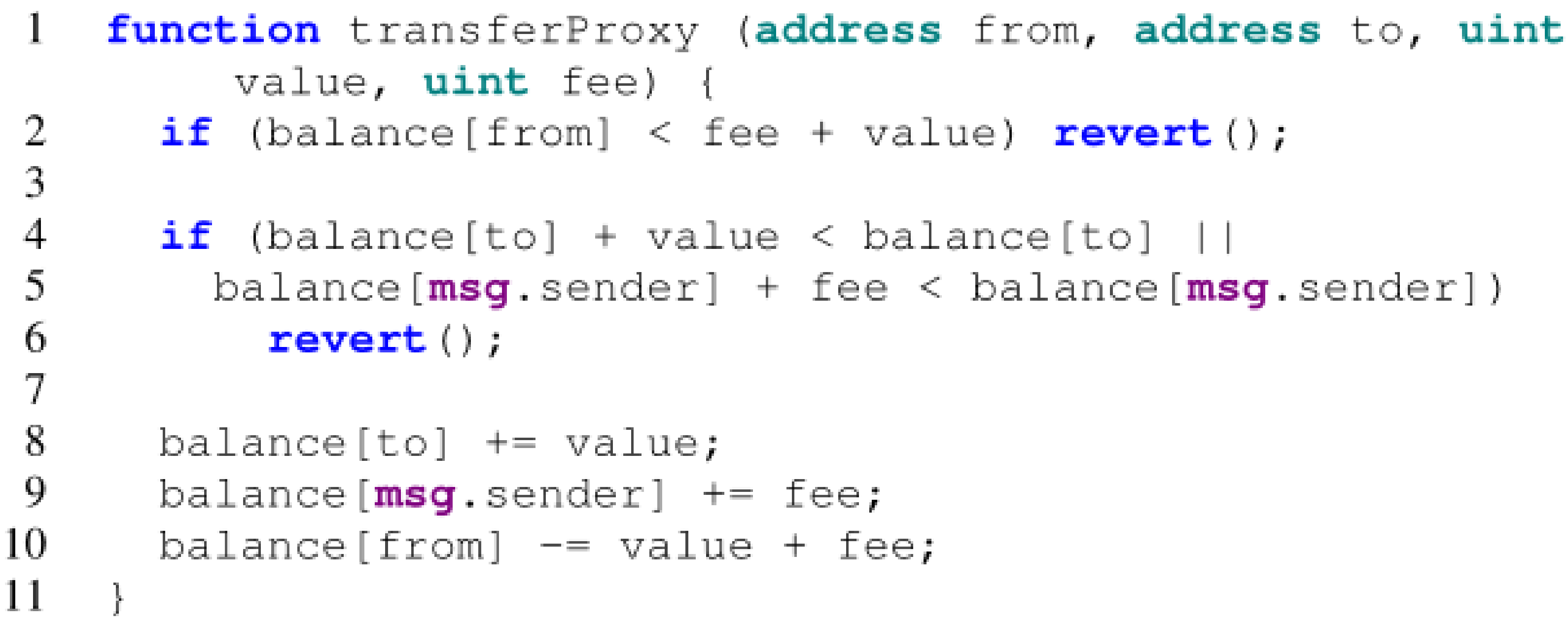
leo@ethereum.org

leonardoalt

Zeus
IBM
Policy
LLVM
21st Feb 2018
NDSS 2018
18

Motivating Examples

Example1



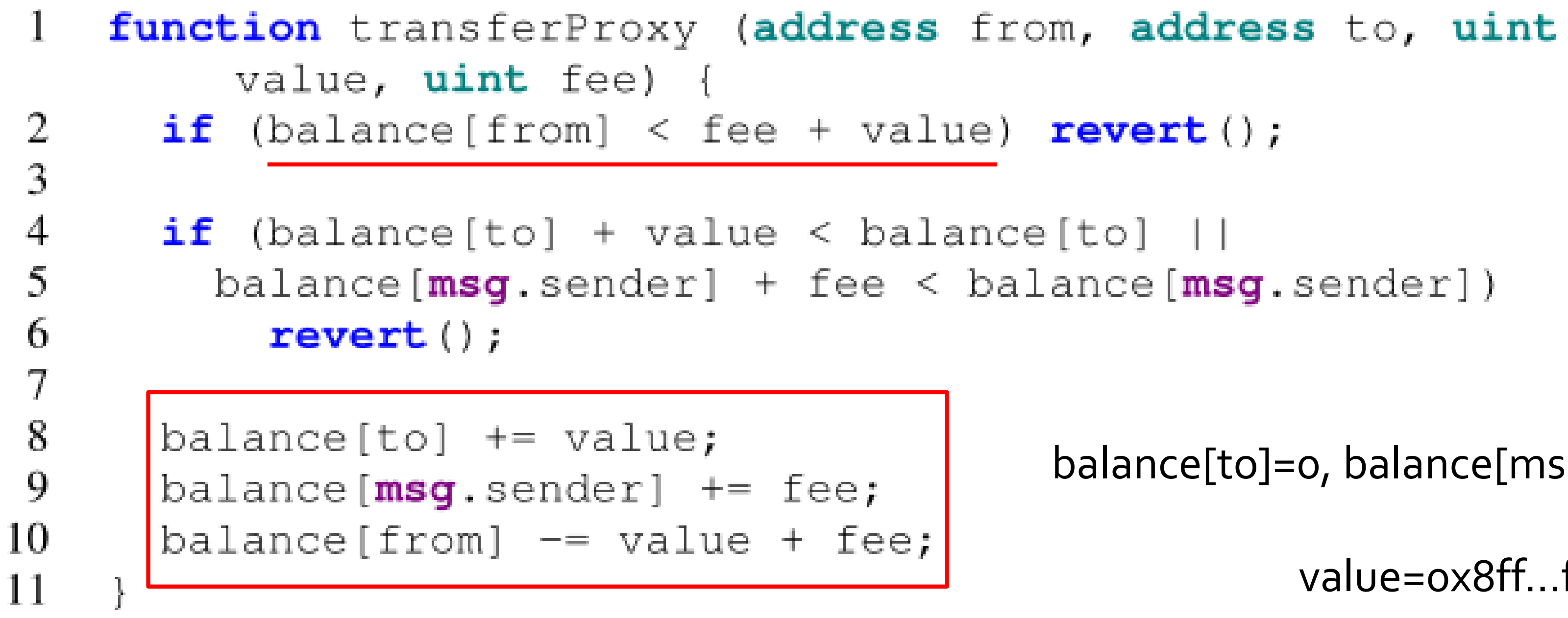
```
function transferProxy (address from, address to, uint
     value, uint fee) {
  if (balance[from] < fee + value) revert();

  if (balance[to] + value < balance[to] ||
    balance[msg.sender] + fee < balance[msg.sender])
      revert();

  balance[to] += value;
  balance[msg.sender] += fee;
  balance[from] -= value + fee;
}
```

A vulnerable function from SmartMesh (CVE-2018-10376)

Example1



```
function transferProxy (address from, address to, uint
      value, uint fee) {
  if (balance[from] < fee + value) revert();

  if (balance[to] + value < balance[to] ||
    balance[msg.sender] + fee < balance[msg.sender])
      revert();

  balance[to] += value;
  balance[msg.sender] += fee;
  balance[from] -= value + fee;
}
```

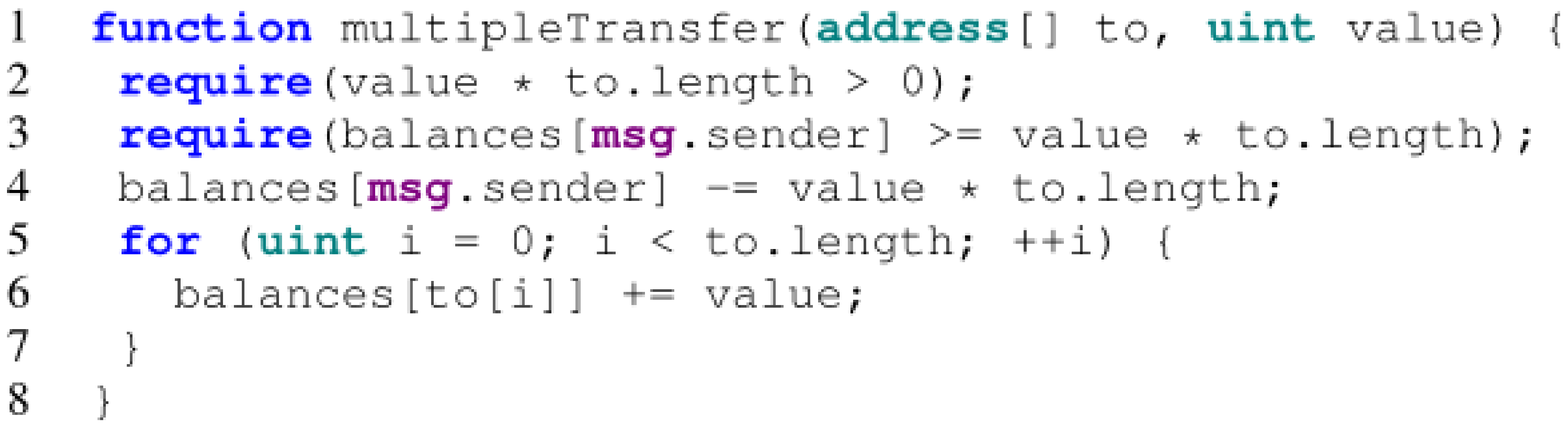
balance[to]=0, balance[msg.sender]=0, balance[from]=0

value=0x8ff…ff, fee=0x700…01

value + fee = 0 (overflow!)

balance[to] : 0x8ff…ff, balance[msg.sender] : 0x700…01

Example2

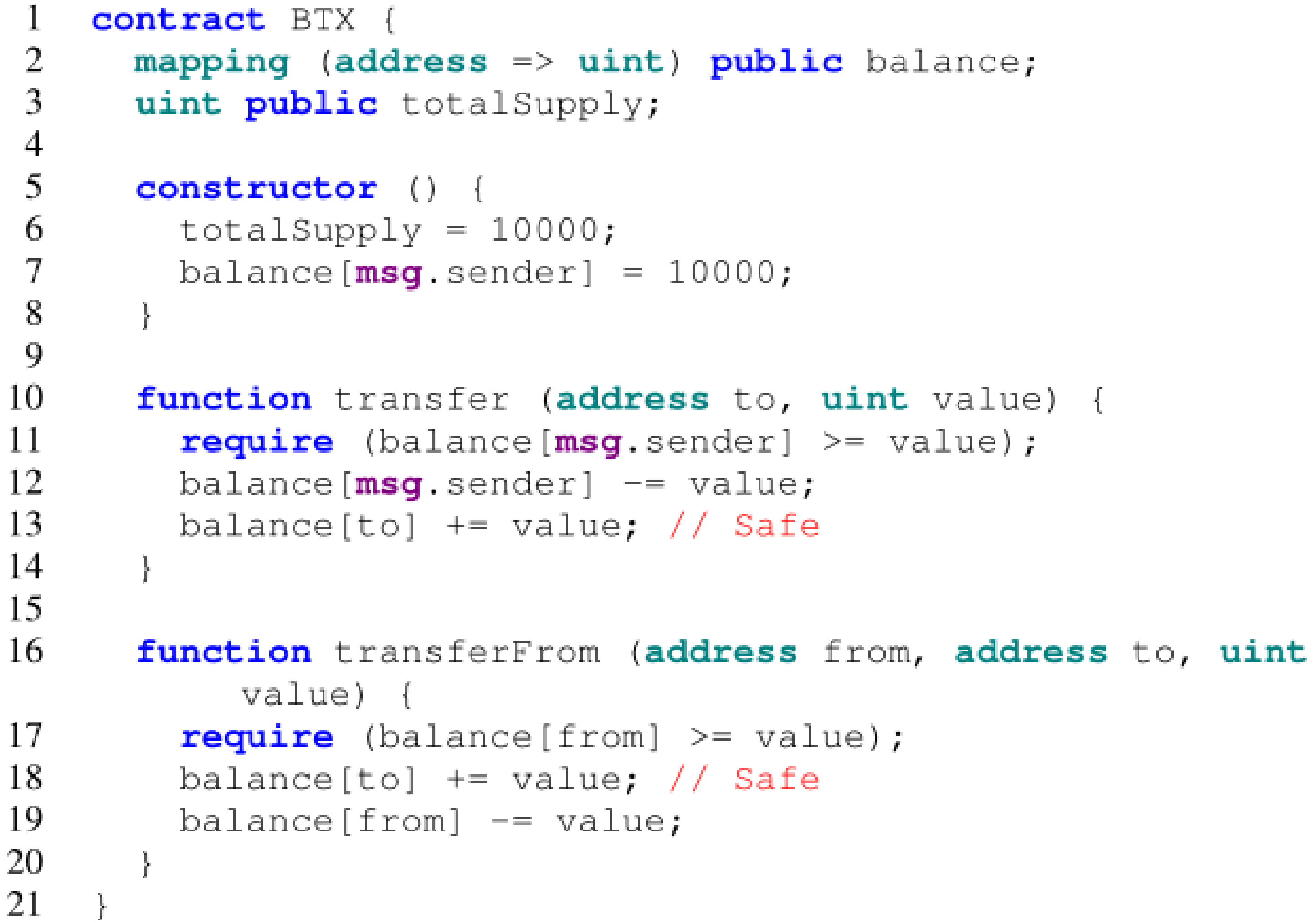


```
function multipleTransfer(address[] to, uint value) {
 require(value * to.length > 0);
 require(balances[msg.sender] >= value * to.length);
 balances[msg.sender] -= value * to.length;
 for (uint i = 0; i < to.length; ++i) {
   balances[to[i]] += value;
 }
}
```

A vulnerable function from Neo Genesis Token (CVE-2018-14006)

Example3

$$\sum_i \texttt{balance}[i] = 10000$$



```
contract BTX {
  mapping (address => uint) public balance;
  uint public totalSupply;

  constructor () {
    totalSupply = 10000;
    balance[msg.sender] = 10000;
  }

  function transfer (address to, uint value) {
    require (balance[msg.sender] >= value);
    balance[msg.sender] -= value;
    balance[to] += value; // Safe
  }

  function transferFrom (address from, address to, uint
       value) {
    require (balance[from] >= value);
    balance[to] += value; // Safe
    balance[from] -= value;
  }
}
```

Example contract simplified from CVE-2018-13326

Algorithm

Language

$$
\begin{array}{lcl}
c \in C & ::= & G^* \;\; F^*, \qquad\quad f \in F \;\; ::= \;\; x(y)\{S\} \\
a \in A & ::= & x := E \mid x[y] := E \mid \mathit{assume}(B) \mid \mathit{assert}(B) \\
s \in S & ::= & A \mid \mathit{if} \;\; B \;\; S_1 \;\; S_2 \mid \mathit{while}^l \;\; E \;\; S \mid S_1; S_2
\end{array}
$$

Subset of Solidity

Goal

- Proves or disproves every assertion in the contract

```
a + b

assert(a+b <= a)
```

```
a * b

assert(a == 0 ||
(a != 0 && (a*b)/a == b))
```

Notation

- FOL : the set of first-order formulas in the combined theory

- e[y/x] : new expression where x gets replaced by y

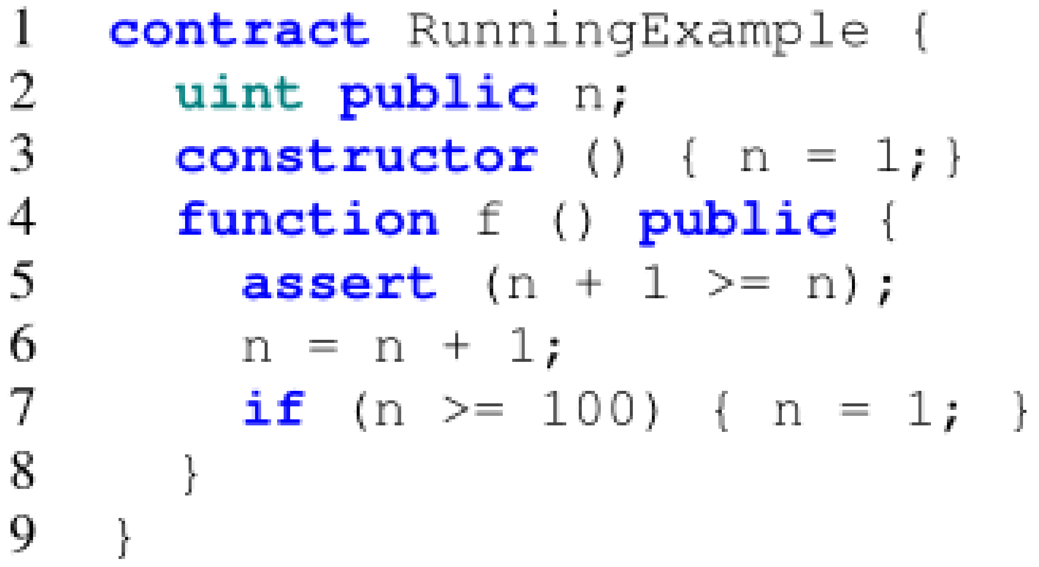
Overview

Transaction invariant $\psi \in \mathsf{FOL}$

Loop invariant $\mu \in \mathit{Label} \rightarrow \mathsf{FOL}$

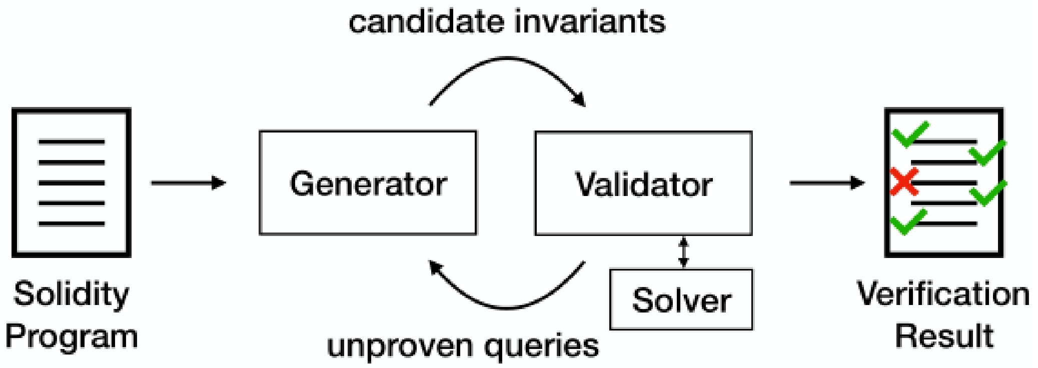
$$(\psi, \mu)$$

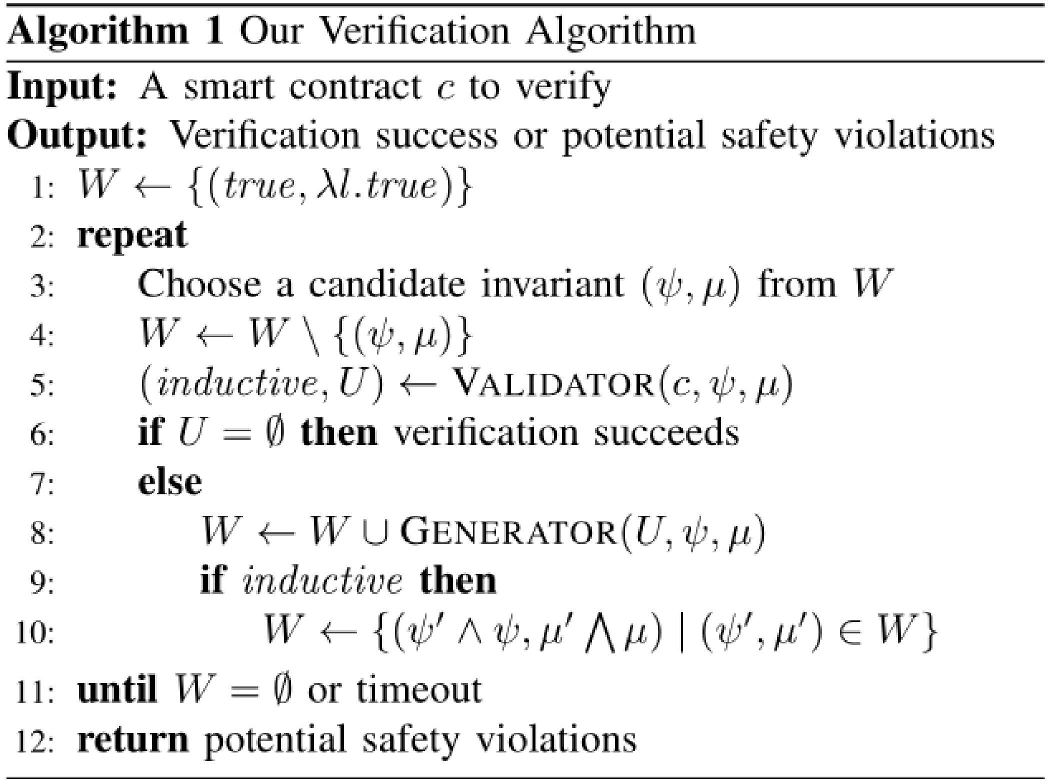
n <= 100 ⟶ n + 1 >= n



```
contract RunningExample {
  uint public n;
  constructor () { n = 1;}
  function f () public {
    assert (n + 1 >= n);
    n = n + 1;
    if (n >= 100) { n = 1; }
  }
}
```

Overview




Algorithm 1 Our Verification Algorithm

Input: A smart contract c to verify

Output: Verification success or potential safety violations

```
$W \leftarrow \{(true, \lambda l.true)\}$
repeat
    Choose a candidate invariant $(\psi, \mu)$ from $W$
    $W \leftarrow W \setminus \{(\psi, \mu)\}$
    $(inductive, U) \leftarrow \textsc{Validator}(c, \psi, \mu)$
    if $U = \emptyset$ then verification succeeds
    else
        $W \leftarrow W \cup \textsc{Generator}(U, \psi, \mu)$
        if $inductive$ then
            $W \leftarrow \{(\psi' \wedge \psi, \mu' \bigwedge \mu) \mid (\psi', \mu') \in W\}$
until $W = \emptyset$ or timeout
return potential safety violations
```

Validator

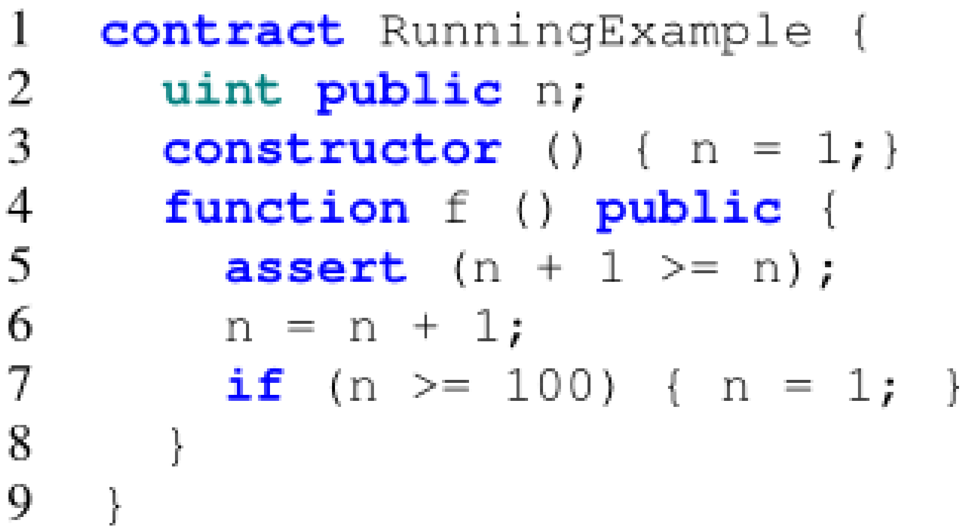
- Basic Path Construction

Break down the program into a finite set of basic paths $((l_1, \phi_1), a_1; \ldots; a_n, (l_2, \phi_2))$

$$\psi = n \leq 100$$

$$\begin{aligned} p_1 &: ((entry_0, true), n := 1, (exit_0, n \leq 100)) \\ p_2 &: ((entry_f, n \leq 100), a_1, (exit_f, n \leq 100)) \\ p_3 &: ((entry_f, n \leq 100), a_2, (exit_f, n \leq 100)) \end{aligned}$$

a1 = assert(n + 1 >= n); n = n + 1; assume(n >= 100)
a2 = assert(n + 1 >= n); n = n + 1; assume(n < 100)



```
contract RunningExample {
  uint public n;
  constructor () { n = 1;}
  function f () public {
    assert (n + 1 >= n);
    n = n + 1;
    if (n >= 100) { n = 1; }
  }
}
```

Validator

- Generation of Verification Conditions

sp : stmt -> FOL x FOL -> FOL x FOL

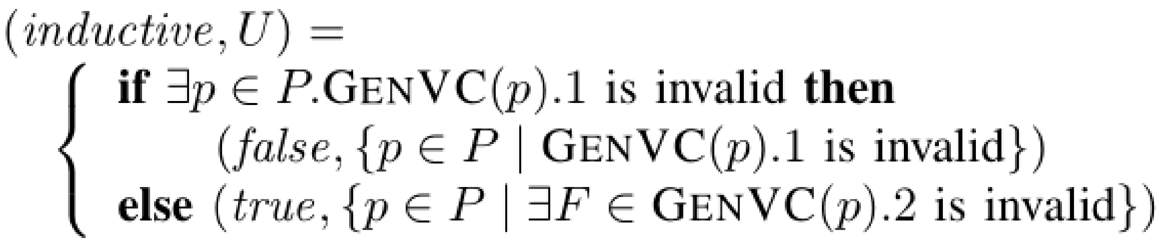
$$
\begin{aligned}
\mathsf{sp}(x := e)(\phi_1, \phi_2) &= (x = e[x'/x] \wedge \phi_1[x'/x], \phi_2) \\
\mathsf{sp}(x[y] := e)(\phi_1, \phi_2) &= (x = x'\langle y \lhd e[x'/x]\rangle \wedge \phi_1[x'/x], \phi_2) \\
\mathsf{sp}(\mathit{assume}(e))(\phi_1, \phi_2) &= (\phi_1 \wedge e, \phi_2) \\
\mathsf{sp}(\mathit{assert}(e))(\phi_1, \phi_2) &= (\phi_1, \phi_2 \wedge (\phi_1 \rightarrow e))
\end{aligned}
$$

$$\textsc{GenVC}(((l_1, \phi_1), a_1; \ldots; a_n, (l_2, \phi_2))) = (\underline{\phi_1' \rightarrow \phi_2, \phi_2'})$$

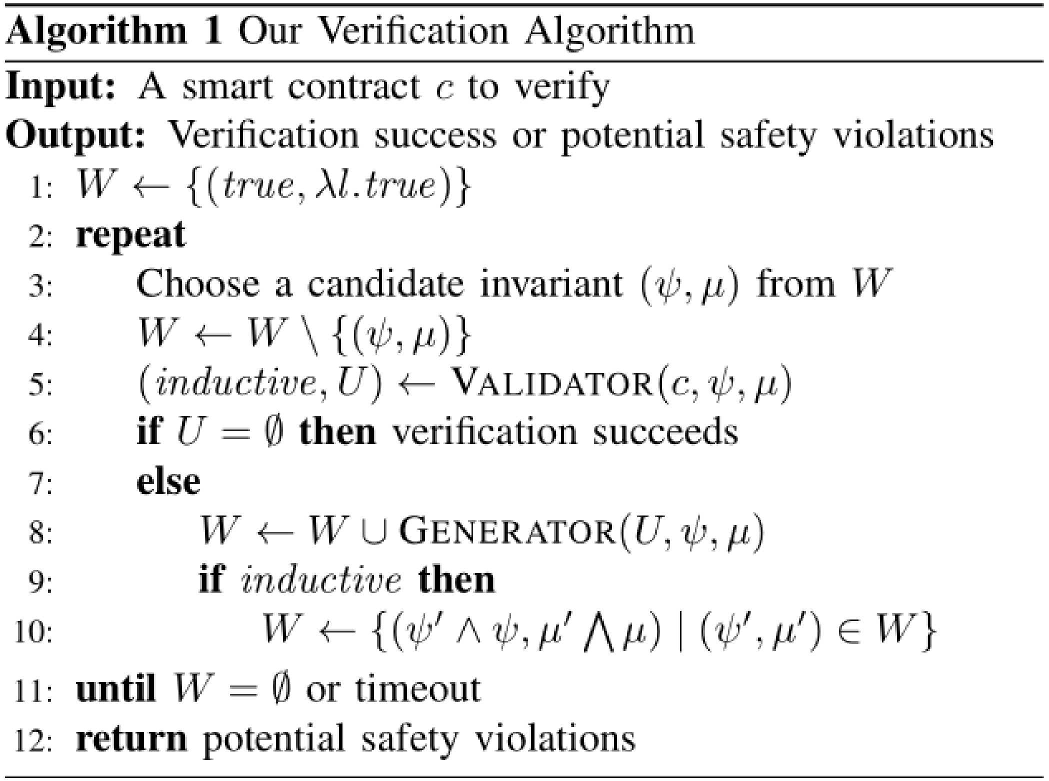
$$(\phi_1', \phi_2') = (\mathsf{sp}(a_n) \circ \cdots \circ \mathsf{sp}(a_2) \circ \mathsf{sp}(a_1))(\phi_1, \mathit{true})$$

Validator

- Collecting Unproven Paths



$$(inductive, U) = \begin{cases} \textbf{if } \exists p \in P.\textsc{GenVC}(p).1 \text{ is invalid } \textbf{then} \\ \qquad (false, \{p \in P \mid \textsc{GenVC}(p).1 \text{ is invalid}\}) \\ \textbf{else } (true, \{p \in P \mid \exists F \in \textsc{GenVC}(p).2 \text{ is invalid}\}) \end{cases}$$



Algorithm 1 Our Verification Algorithm

```
Input: A smart contract $c$ to verify
Output: Verification success or potential safety violations
$W \leftarrow \{(true, \lambda l.true)\}$
repeat
    Choose a candidate invariant $(\psi, \mu)$ from $W$
    $W \leftarrow W \setminus \{(\psi, \mu)\}$
    $(inductive, U) \leftarrow \textsc{Validator}(c, \psi, \mu)$
    if $U = \emptyset$ then verification succeeds
    else
        $W \leftarrow W \cup \textsc{Generator}(U, \psi, \mu)$
        if $inductive$ then
            $W \leftarrow \{(\psi' \wedge \psi, \mu' \bigwedge \mu) \mid (\psi', \mu') \in W\}$
until $W = \emptyset$ or timeout
return potential safety violations
```

Generator

$$\textsc{Generator}(U, \psi, \mu) \quad \longrightarrow \quad \{(\psi, \mu') \mid \mu' \in \textsc{Loop}(\mu, U)\} \cup \{(\psi', \mu) \mid \psi' \in \textsc{Tran}(\psi, U)\}$$

Generator

$\textsc{Loop}(\mu, U)$

$$\bigcup_{((l_1,\_),a,(l_2,\_))\in U} \{\mu[l_i \mapsto \phi_i] \mid i \in [1,2], \phi_i \in \textsc{RefineL}(\mu(l_i), a)\}$$

$\textsc{Tran}(\psi, U)$

$$\{\psi' \mid ((l_1,\_), a, (l_2,\_)) \in U, \psi' \in \textsc{RefineT}(\psi, a)\}$$

Generator

$$(\rightsquigarrow_{X,C}) \subseteq \mathsf{FOL} \times \mathsf{FOL}$$

$$\{\phi' \mid \phi \rightsquigarrow_{X,C} \phi'\}$$

1. Smart contracts often use loops in simple and restricted form

```
for(i = 0; i < x ; i++)
```

$x = y$, $x \geq y$, $x = n$, $x \geq n$, and $x \leq n$

2. It is important to capture the characteristic of mapping datatype

```
mapping (address => uint) public balance;
```

$\mathrm{sum}(\texttt{balance})$

3. Invariants are quantifier-free conjunctive formulas

Generator

$$\phi_1 \rightsquigarrow_{X,C} \phi_2 \iff \phi_2 = \phi_1 \wedge \varphi \text{ and } \varphi \in A$$

$x = y, x \geq y, x = n, x \geq n, x \leq n$, sum(x) = e where $x, y \in X, n \in C, e \in C \cup X$

$$\begin{array}{rcl} \textsc{RefineL}(\psi, a) & = & \{\psi' \mid \psi \rightsquigarrow_{vars(a), const(a)} \psi'\} \\ \textsc{RefineT}(\phi, a) & = & \{\phi' \mid \phi \rightsquigarrow_{globals, cnstr \cup const(a)} \phi'\} \end{array}$$

Solver

- Use SMT solver (in VeriSmart Z3)

- Preprocessing for uninterpretable symbols

$$F = \cdots \wedge \mathsf{sum}(x) = n \wedge x[i] = v_1 \wedge x[j] = v_2 \wedge \cdots$$

- Add domain-specific optimization to improve performance

$$\mathit{true} \rightarrow (a-b = 0) \vee (a-b \neq 0 \wedge ((a-b) * 255)/(a-b) = 255)$$

Implementation

- Implemented in Ocaml

- Support the full solidity features

- Limited support for inline assembly

Evaluation

Evaluation

- 4 bug-finders

Oyente

Mythril

manticore

Osiris

- 2 verifiers

Using Solidity's SMTChecker

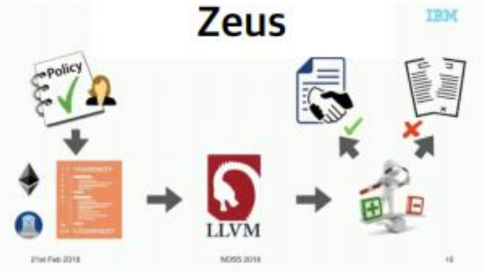
Leonardo Alt

Ethereum Foundation

leonardoalt

leo@ethereum.org

leonardoalt



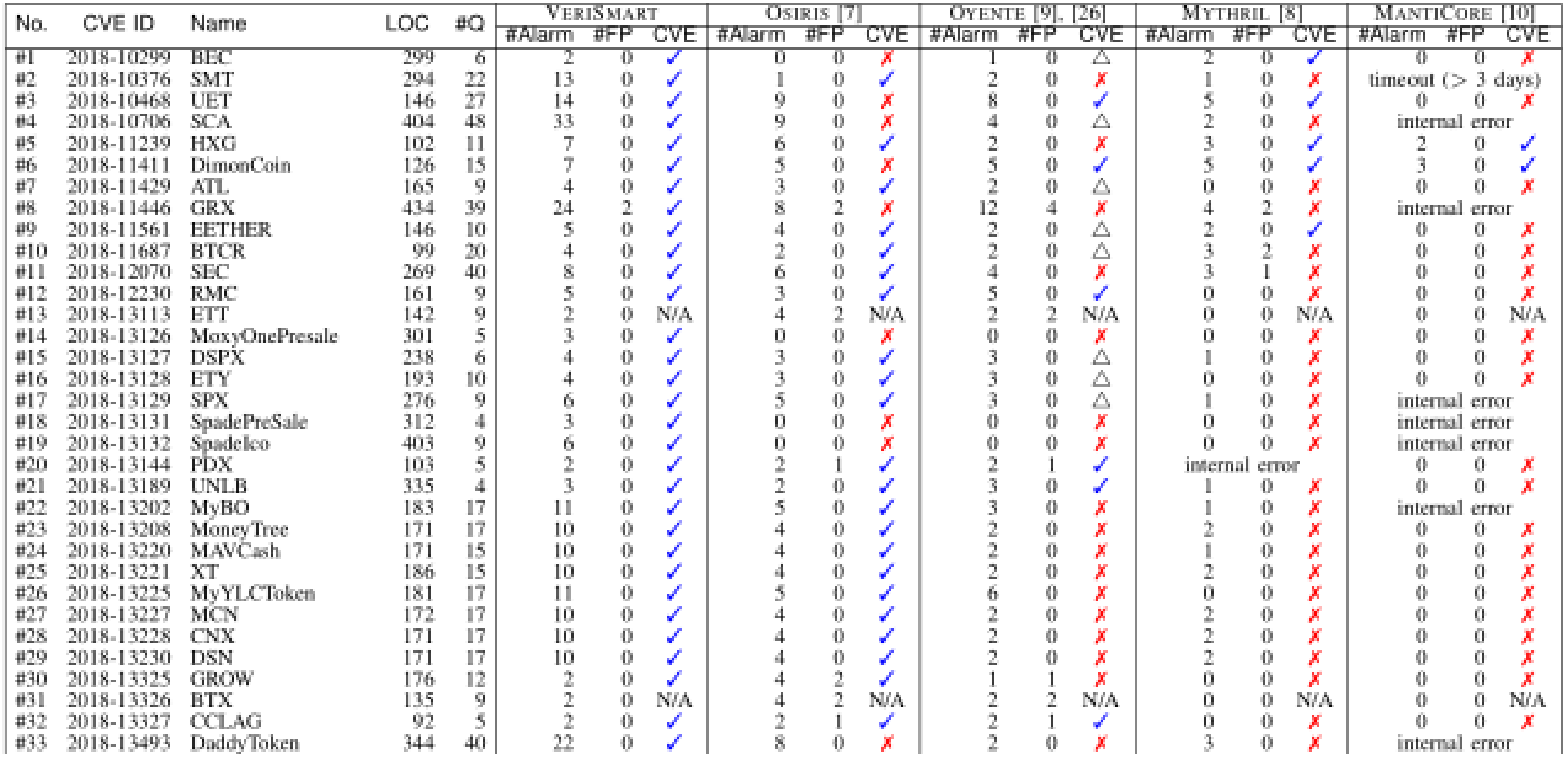
Zeus

21st Feb 2018

NDSS 2018

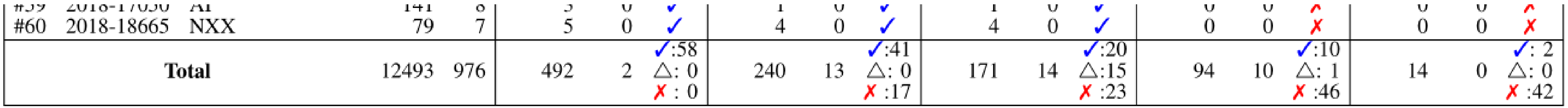
18

- Tested with Intel Core i7-9700K, 64GB RAM

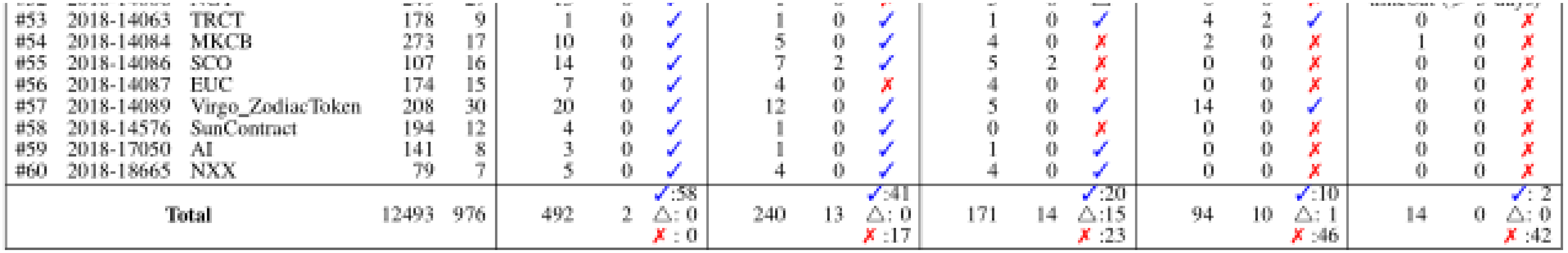


No.	CVE ID	Name	LOC	#Q	VeriSmart			Osiris [7]			Oyente [9], [26]			Mythril [8]			MantiCore [10]		
					#Alarm	#FP	CVE	#Alarm	#FP	CVE	#Alarm	#FP	CVE	#Alarm	#FP	CVE	#Alarm	#FP	CVE
#1	2018-10299	BEC	299	6	2	0	✓	0	0	✗	1	0	△	2	0	✓	0	0	✗
#2	2018-10376	SMT	294	22	13	0	✓	1	0	✓	2	0	✗	1	0	✗	timeout (> 3 days)		
#3	2018-10468	UET	146	27	14	0	✓	9	0	✗	8	0	✓	5	0	✓	0	0	✗
#4	2018-10706	SCA	404	48	33	0	✓	9	0	✗	4	0	△	2	0	✗	internal error		
#5	2018-11239	HXG	102	11	7	0	✓	6	0	✓	2	0	✗	3	0	✓	2	0	✓
#6	2018-11411	DimonCoin	126	15	7	0	✓	5	0	✗	5	0	✓	5	0	✓	3	0	✓
#7	2018-11429	ATL	165	9	4	0	✓	3	0	✓	2	0	△	0	0	✗	0	0	✗
#8	2018-11446	GRX	434	39	24	2	✓	8	2	✗	12	4	✗	4	2	✗	internal error		
#9	2018-11561	EETHER	146	10	5	0	✓	4	0	✓	2	0	△	2	0	✓	0	0	✗
#10	2018-11687	BTCR	99	20	4	0	✓	2	0	✓	2	0	△	3	2	✗	0	0	✗
#11	2018-12070	SEC	269	40	8	0	✓	6	0	✓	4	0	✗	3	1	✗	0	0	✗
#12	2018-12230	RMC	161	9	5	0	✓	3	0	✓	5	0	✓	0	0	✗	0	0	✗
#13	2018-13113	ETT	142	9	2	0	N/A	4	2	N/A	2	2	N/A	0	0	N/A	0	0	N/A
#14	2018-13126	MoxyOnePresale	301	5	3	0	✓	0	0	✗	0	0	✗	0	0	✗	0	0	✗
#15	2018-13127	DSPX	238	6	4	0	✓	3	0	✓	3	0	△	1	0	✗	0	0	✗
#16	2018-13128	ETY	193	10	4	0	✓	3	0	✓	3	0	△	0	0	✗	0	0	✗
#17	2018-13129	SPX	276	9	6	0	✓	5	0	✓	3	0	△	1	0	✗	internal error		
#18	2018-13131	SpadePreSale	312	4	3	0	✓	0	0	✗	0	0	✗	0	0	✗	internal error		
#19	2018-13132	SpadeIco	403	9	6	0	✓	0	0	✗	0	0	✗	0	0	✗	internal error		
#20	2018-13144	PDX	103	5	2	0	✓	2	1	✓	2	1	✓	internal error			0	0	✗
#21	2018-13189	UNLB	335	4	3	0	✓	2	0	✓	3	0	✓	1	0	✗	0	0	✗
#22	2018-13202	MyBO	183	17	11	0	✓	5	0	✓	3	0	✗	1	0	✗	internal error		
#23	2018-13208	MoneyTree	171	17	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#24	2018-13220	MAVCash	171	15	10	0	✓	4	0	✓	2	0	✗	1	0	✗	0	0	✗
#25	2018-13221	XT	186	15	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#26	2018-13225	MyYLCToken	181	17	11	0	✓	5	0	✓	6	0	✗	0	0	✗	0	0	✗
#27	2018-13227	MCN	172	17	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#28	2018-13228	CNX	171	17	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#29	2018-13230	DSN	171	17	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#30	2018-13325	GROW	176	12	2	0	✓	4	2	✓	1	1	✗	0	0	✗	0	0	✗
#31	2018-13326	BTX	135	9	2	0	N/A	4	2	N/A	2	2	N/A	0	0	N/A	0	0	N/A
#32	2018-13327	CCLAG	92	5	2	0	✓	2	1	✓	2	1	✓	0	0	✗	0	0	✗
#33	2018-13493	DaddyToken	344	40	22	0	✓	8	0	✗	2	0	✗	3	0	✗	internal error		

VeriSmart Osiris Oyente Mythril MantiCore



#59	2018-17050	AI	141	8	3	0	✓	1	0	✓	1	0	✓	0	0	✗	0	0	✗
#60	2018-18665	NXX	79	7	5	0	✓	4	0	✓	4	0	✓	0	0	✗	0	0	✗
Total			12493	976	492	2	✓:58 △: 0 ✗ : 0	240	13	✓:41 △: 0 ✗ :17	171	14	✓:20 △:15 ✗ :23	94	10	✓:10 △: 1 ✗ :46	14	0	✓: 2 △: 0 ✗ :42

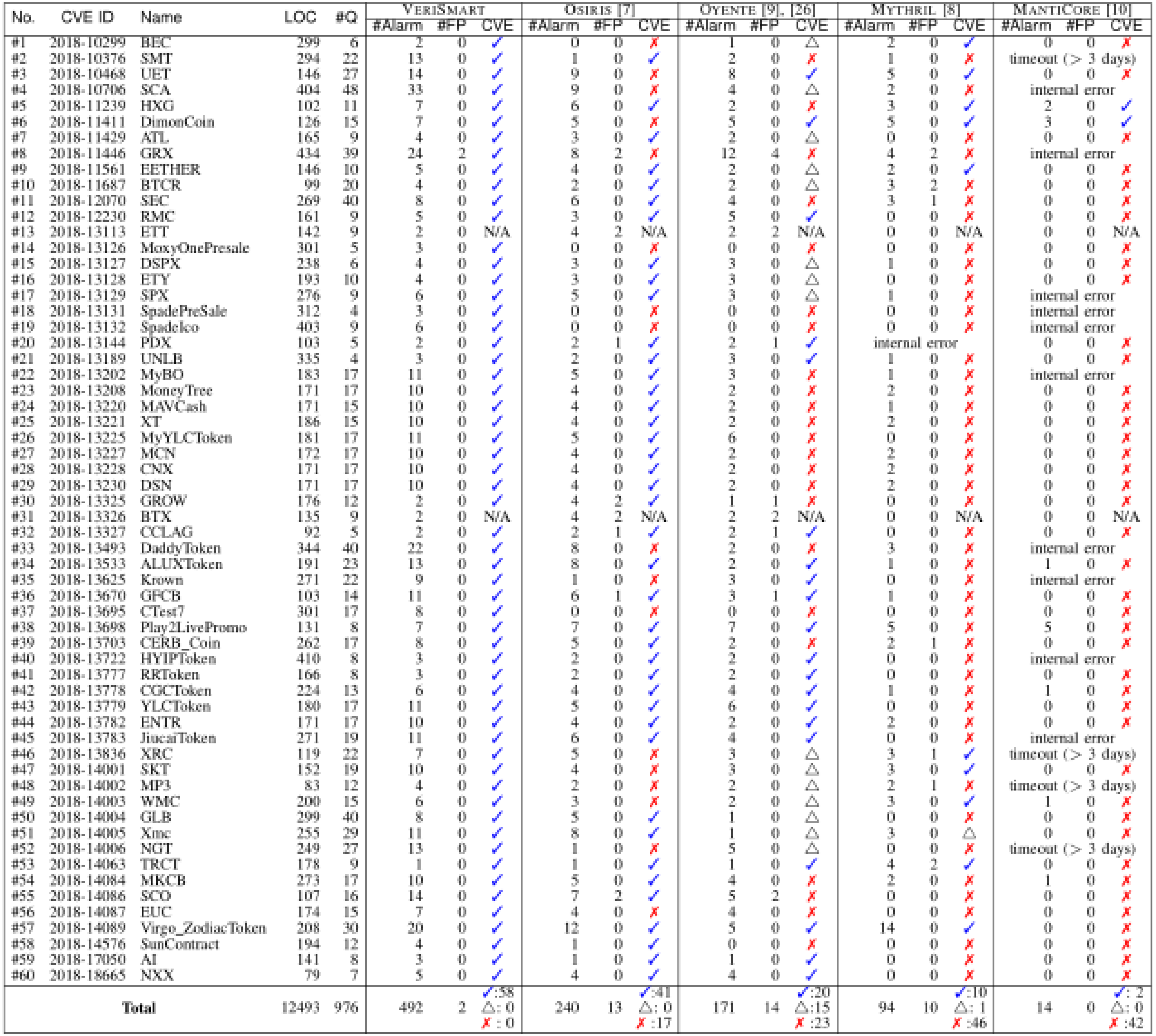


No.	CVE ID	Name	LOC	#Q	VeriSmart #Alarm	#FP	CVE	Osiris #Alarm	#FP	CVE	Oyente #Alarm	#FP	CVE	Mythril #Alarm	#FP	CVE	MantiCore #Alarm	#FP	CVE
#53	2018-14063	TRCT	178	9	1	0	✓	1	0	✓	1	0	✓	4	2	✓	0	0	✗
#54	2018-14084	MKCB	273	17	10	0	✓	5	0	✓	4	0	✗	2	0	✗	1	0	✗
#55	2018-14086	SCO	107	16	14	0	✓	7	2	✓	5	2	✗	0	0	✗	0	0	✗
#56	2018-14087	EUC	174	15	7	0	✓	4	0	✗	4	0	✗	0	0	✗	0	0	✗
#57	2018-14089	Virgo_ZodiacToken	208	30	20	0	✓	12	0	✓	5	0	✓	14	0	✓	0	0	✗
#58	2018-14576	SunContract	194	12	4	0	✓	1	0	✓	0	0	✗	0	0	✗	0	0	✗
#59	2018-17050	AI	141	8	3	0	✓	1	0	✓	1	0	✓	0	0	✗	0	0	✗
#60	2018-18665	NXX	79	7	5	0	✓	4	0	✓	4	0	✓	0	0	✗	0	0	✗
Total			12493	976	492	2	✓:58 △: 0 ✗ : 0	240	13	✓:41 △: 0 ✗ :17	171	14	✓:20 △:15 ✗ :23	94	10	✓:10 △: 1 ✗ :46	14	0	✓: 2 △: 0 ✗ :42

Evaluation

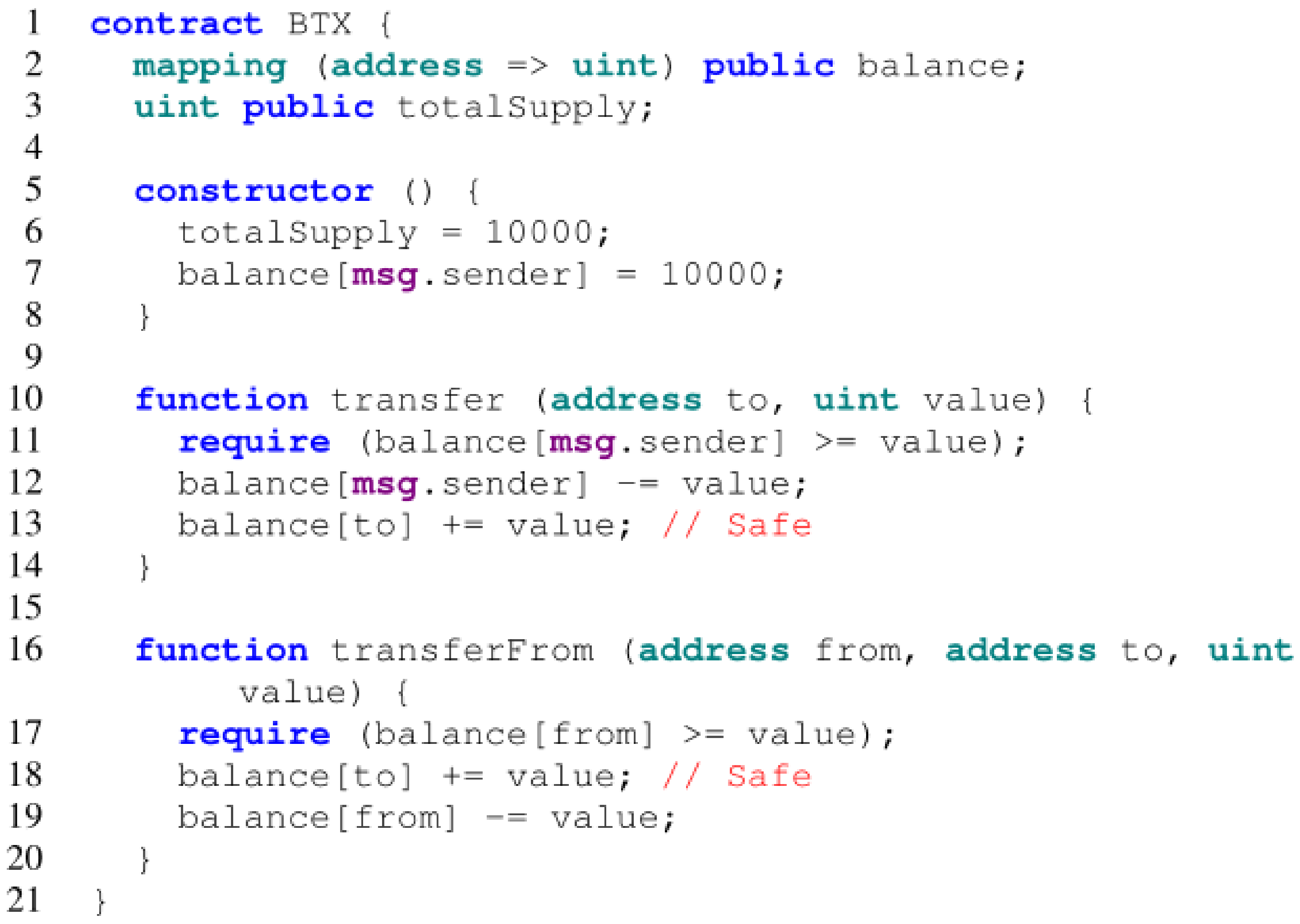
	VeriSmart	Osiris	Oyente	Mythril	MantiCore
Overall execution time (second)	3,807	14,942	840	49,680	112,920 (excluding timeout)
# of caught CVE (all 60)	58	41	20	10	2
$\frac{\#FP}{\#Alarm}$	0.41% (2/492)	5.42% (13/240)	8.19% (14/171)	10.64% (10/94)	0% (0/14)

$$\forall \texttt{x}.\texttt{totalLocked[x]} = \sum_{i} \texttt{locked[x][}i\texttt{]}$$



No.	CVE ID	Name	LOC	#Q	VeriSmart			Osiris [7]			Oyente [9], [26]			Mythril [8]			Manticore [10]		
					#Alarm	#FP	CVE	#Alarm	#FP	CVE	#Alarm	#FP	CVE	#Alarm	#FP	CVE	#Alarm	#FP	CVE
#1	2018-10299	BEC	299	6	2	0	✓	0	0	✗	1	0	△	2	0	✓	0	0	✗
#2	2018-10376	SMT	294	22	13	0	✓	1	0	✓	2	0	✗	1	0	✗	timeout (> 3 days)		
#3	2018-10468	UET	146	27	14	0	✓	9	0	✗	8	0	✓	5	0	✓	0	0	✗
#4	2018-10706	SCA	404	48	33	0	✓	9	0	✗	4	0	△	2	0	✗	internal error		
#5	2018-11239	HXG	102	11	7	0	✓	6	0	✓	2	0	✗	3	0	✓	2	0	✓
#6	2018-11411	DimonCoin	126	15	7	0	✓	5	0	✗	5	0	✓	5	0	✓	3	0	✓
#7	2018-11429	ATL	165	9	4	0	✓	3	0	✓	2	0	△	0	0	✗	0	0	✗
#8	2018-11446	GRX	434	39	24	2	✓	8	2	✗	12	4	✗	4	2	✗	internal error		
#9	2018-11561	EETHER	146	10	5	0	✓	4	0	✓	2	0	△	2	0	✓	0	0	✗
#10	2018-11687	BTCR	99	20	4	0	✓	2	0	✓	2	0	△	3	2	✗	0	0	✗
#11	2018-12070	SEC	269	40	8	0	✓	6	0	✓	4	0	✗	3	1	✗	0	0	✗
#12	2018-12230	RMC	161	9	5	0	✓	3	0	✓	5	0	✓	0	0	✗	0	0	✗
#13	2018-13113	ETT	142	9	2	0	N/A	4	2	N/A	2	2	N/A	0	0	N/A	0	0	N/A
#14	2018-13126	MoxyOnePresale	301	5	3	0	✓	0	0	✗	0	0	✗	0	0	✗	0	0	✗
#15	2018-13127	DSPX	238	6	4	0	✓	3	0	✓	3	0	△	1	0	✗	0	0	✗
#16	2018-13128	ETY	193	10	4	0	✓	3	0	✓	3	0	△	0	0	✗	0	0	✗
#17	2018-13129	SPX	276	9	6	0	✓	5	0	✓	3	0	△	1	0	✗	internal error		
#18	2018-13131	SpadePreSale	312	4	3	0	✓	0	0	✗	0	0	✗	0	0	✗	internal error		
#19	2018-13132	SpadeIco	403	9	6	0	✓	0	0	✗	0	0	✗	0	0	✗	internal error		
#20	2018-13144	PDX	103	5	2	0	✓	2	1	✓	2	1	✓	internal error			0	0	✗
#21	2018-13189	UNLB	335	4	3	0	✓	2	0	✓	3	0	✓	1	0	✗	0	0	✗
#22	2018-13202	MyBO	183	17	11	0	✓	5	0	✓	3	0	✗	1	0	✗	internal error		
#23	2018-13208	MoneyTree	171	17	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#24	2018-13220	MAVCash	171	15	10	0	✓	4	0	✓	2	0	✗	1	0	✗	0	0	✗
#25	2018-13221	XT	186	15	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#26	2018-13225	MyYLCToken	181	17	11	0	✓	5	0	✓	6	0	✗	0	0	✗	0	0	✗
#27	2018-13227	MCN	172	17	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#28	2018-13228	CNX	171	17	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#29	2018-13230	DSN	171	17	10	0	✓	4	0	✓	2	0	✗	2	0	✗	0	0	✗
#30	2018-13325	GROW	176	12	2	0	✓	4	2	✓	1	1	✗	0	0	✗	0	0	✗
#31	2018-13326	BTX	135	9	2	0	N/A	4	2	N/A	2	2	N/A	0	0	N/A	0	0	N/A
#32	2018-13327	CCLAG	92	5	2	0	✓	2	1	✓	2	1	✓	0	0	✗	0	0	✗
#33	2018-13493	DaddyToken	344	40	22	0	✓	8	0	✗	2	0	✗	3	0	✗	internal error		
#34	2018-13533	ALUXToken	191	23	13	0	✓	8	0	✓	2	0	✓	1	0	✗	1	0	✗
#35	2018-13625	Krown	271	22	9	0	✓	1	0	✗	3	0	✓	0	0	✗	internal error		
#36	2018-13670	GFCB	103	14	11	0	✓	6	1	✓	3	1	✓	1	0	✗	0	0	✗
#37	2018-13695	CTest7	301	17	8	0	✓	0	0	✗	0	0	✗	0	0	✗	0	0	✗
#38	2018-13698	Play2LivePromo	131	8	7	0	✓	7	0	✓	7	0	✓	5	0	✗	5	0	✗
#39	2018-13703	CERB_Coin	262	17	8	0	✓	5	0	✓	2	0	✗	2	1	✗	0	0	✗
#40	2018-13722	HYIPToken	410	8	3	0	✓	2	0	✓	2	0	✓	0	0	✗	internal error		
#41	2018-13777	RRToken	166	8	3	0	✓	2	0	✓	2	0	✓	0	0	✗	0	0	✗
#42	2018-13778	CGCToken	224	13	6	0	✓	4	0	✓	4	0	✓	1	0	✗	1	0	✗
#43	2018-13779	YLCToken	180	17	11	0	✓	5	0	✓	6	0	✓	0	0	✗	0	0	✗
#44	2018-13782	ENTR	171	17	10	0	✓	4	0	✓	2	0	✓	2	0	✗	0	0	✗
#45	2018-13783	JiucaiToken	271	19	11	0	✓	6	0	✓	4	0	✓	0	0	✗	internal error		
#46	2018-13836	XRC	119	22	7	0	✓	5	0	✗	3	0	△	3	1	✓	timeout (> 3 days)		
#47	2018-14001	SKT	152	19	10	0	✓	4	0	✗	3	0	△	3	0	✓	0	0	✗
#48	2018-14002	MP3	83	12	4	0	✓	2	0	✗	2	0	△	2	1	✗	timeout (> 3 days)		
#49	2018-14003	WMC	200	15	6	0	✓	3	0	✗	2	0	△	3	0	✓	1	0	✗
#50	2018-14004	GLB	299	40	8	0	✓	5	0	✓	1	0	△	0	0	✗	0	0	✗
#51	2018-14005	Xmc	255	29	11	0	✓	8	0	✓	1	0	△	3	0	△	0	0	✗
#52	2018-14006	NGT	249	27	13	0	✓	1	0	✗	5	0	△	0	0	✗	timeout (> 3 days)		
#53	2018-14063	TRCT	178	9	1	0	✓	1	0	✓	1	0	✓	4	2	✓	0	0	✗
#54	2018-14084	MKCB	273	17	10	0	✓	5	0	✓	4	0	✗	2	0	✗	1	0	✗
#55	2018-14086	SCO	107	16	14	0	✓	7	2	✓	5	2	✗	0	0	✗	0	0	✗
#56	2018-14087	EUC	174	15	7	0	✓	4	0	✗	4	0	✗	0	0	✗	0	0	✗
#57	2018-14089	Virgo_ZodiacToken	208	30	20	0	✓	12	0	✓	5	0	✓	14	0	✓	0	0	✗
#58	2018-14576	SunContract	194	12	4	0	✓	1	0	✓	0	0	✗	0	0	✗	0	0	✗
#59	2018-17050	AI	141	8	3	0	✓	1	0	✓	1	0	✓	0	0	✗	0	0	✗
#60	2018-18665	NXX	79	7	5	0	✓	4	0	✓	4	0	✓	0	0	✗	0	0	✗
Total			12493	976	492	2	✓:58 △: 0 ✗ : 0	240	13	✓:41 △: 0 ✗ :17	171	14	✓:20 △:15 ✗ :23	94	10	✓:10 △: 1 ✗ :46	14	0	✓: 2 △: 0 ✗ :42

Evaluation



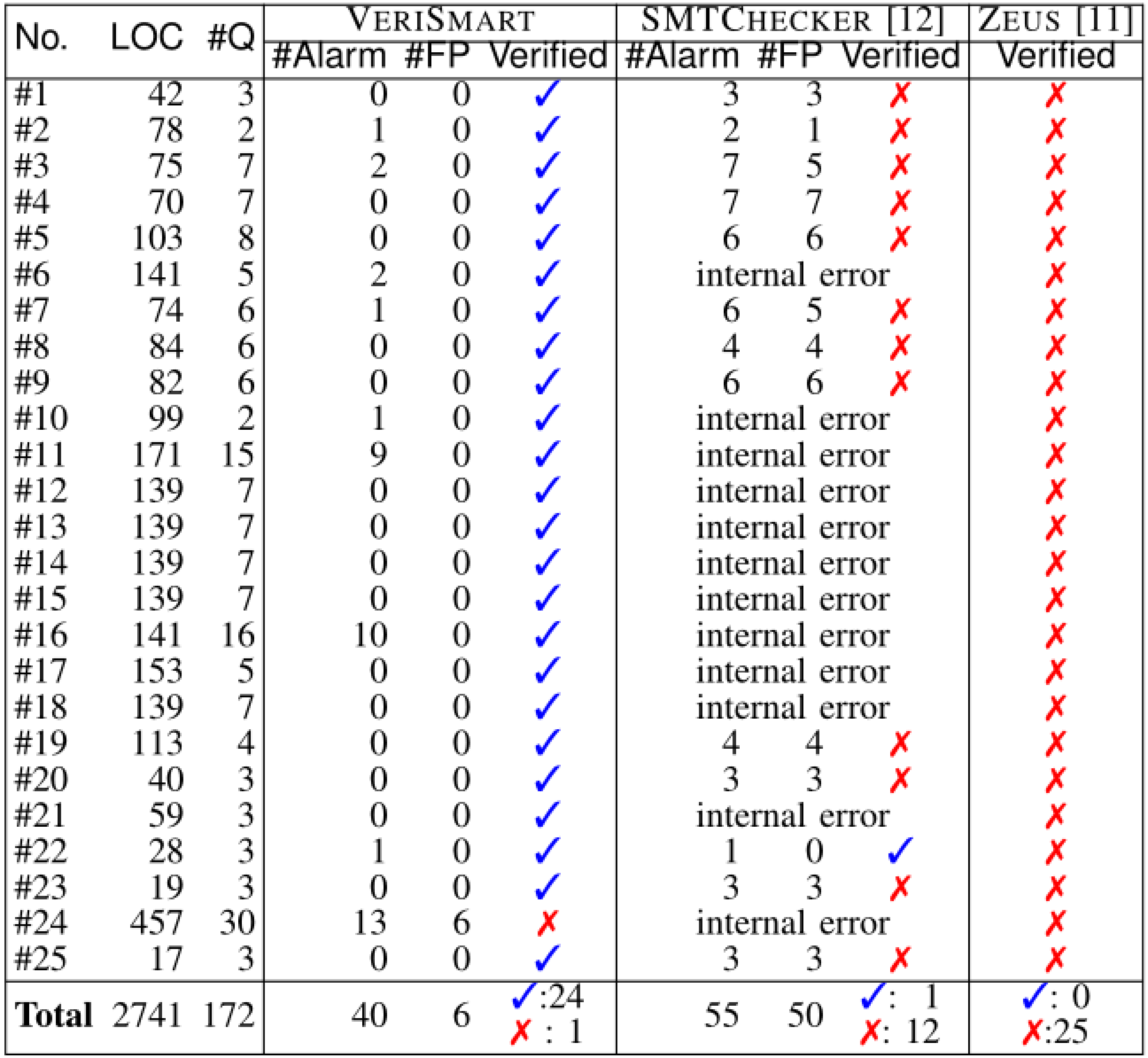
```
contract BTX {
  mapping (address => uint) public balance;
  uint public totalSupply;

  constructor () {
    totalSupply = 10000;
    balance[msg.sender] = 10000;
  }

  function transfer (address to, uint value) {
    require (balance[msg.sender] >= value);
    balance[msg.sender] -= value;
    balance[to] += value; // Safe
  }

  function transferFrom (address from, address to, uint
      value) {
    require (balance[from] >= value);
    balance[to] += value; // Safe
    balance[from] -= value;
  }
}
```

Example contract simplified from CVE-2018-13326



No.	LOC	#Q	VeriSmart			SMTChecker [12]			Zeus [11]
			#Alarm	#FP	Verified	#Alarm	#FP	Verified	Verified
#1	42	3	0	0	✓	3	3	✗	✗
#2	78	2	1	0	✓	2	1	✗	✗
#3	75	7	2	0	✓	7	5	✗	✗
#4	70	7	0	0	✓	7	7	✗	✗
#5	103	8	0	0	✓	6	6	✗	✗
#6	141	5	2	0	✓	internal error			✗
#7	74	6	1	0	✓	6	5	✗	✗
#8	84	6	0	0	✓	4	4	✗	✗
#9	82	6	0	0	✓	6	6	✗	✗
#10	99	2	1	0	✓	internal error			✗
#11	171	15	9	0	✓	internal error			✗
#12	139	7	0	0	✓	internal error			✗
#13	139	7	0	0	✓	internal error			✗
#14	139	7	0	0	✓	internal error			✗
#15	139	7	0	0	✓	internal error			✗
#16	141	16	10	0	✓	internal error			✗
#17	153	5	0	0	✓	internal error			✗
#18	139	7	0	0	✓	internal error			✗
#19	113	4	0	0	✓	4	4	✗	✗
#20	40	3	0	0	✓	3	3	✗	✗
#21	59	3	0	0	✓	internal error			✗
#22	28	3	1	0	✓	1	0	✓	✗
#23	19	3	0	0	✓	3	3	✗	✗
#24	457	30	13	6	✗	internal error			✗
#25	17	3	0	0	✓	3	3	✗	✗
Total	2741	172	40	6	✓:24 ✗ : 1	55	50	✓: 1 ✗: 12	✓: 0 ✗:25

Conclusion

- Transaction Invariants is important

- VeriSmart is powerful verification tool for real-world smart contract

Thanks!