附录 A 每种 bug 的预定义正则表达式及检测方法

A.1 bug 检测方法

1. Authorization through tx.origin。 SolidityCheck 将匹配格式化代码中符合模式 3.1 但不符合模式 3.2 的语句（表 1 中展示了我们预定义的正则表达式）。
2. Block values as a proxy for time。 SolidityCheck 将匹配格式化代码中符合模式 3.3 的语句。
3. byte[]。 SolidityCheck 将匹配格式化代码中符合模式 3.4 的语句。
4. DoS with block gas limit。SolidityCheck 将匹配格式化代码中符合模式 3.5-3.10 的语句。模式 3.5-3.10 表示在条件判断部分中包含标识符或函数调用的 for 语句或 while 语句。
5. DoS with failed call。 SolidityCheck 将匹配格式化代码中符合模式 3.11 或 3.12 的语句。
6. Floating pragma。SolidityCheck 将匹配格式化代码中符合模式 3.13 的语句,或符合模式 3.14 但不包含“<”运算符的语句。对于该种 bug，为推广新版本的安全规范，SolidityCheck 将会检测合约是否包含符合模式 3.15 的语句。如果没有，则该合约将会被判定为包含该种 bug。
7. Implicit visibility level。SolidityCheck 将匹配格式化代码中符合模式 3.16-3.22 但不符合模式 3.23 的语句。
8. Incorrect constructor name。SolidityCheck 将检测格式化代码中是否包含符合模式 3.24 或 3.25 的语句。如果否，则 SolidityCheck 将报告合约包含该种 bug。
9. Integer division。SolidityCheck 将匹配格式化代码中符合模式 3.26 的语句。
10. Locked ethers。SolidityCheck 首先检测格式化代码中是否包含符合模式 3.27 的语句。如果包含，SolidityCheck 将在非汇编语句中检测符合模式 3.28 或 3.29 的语句，并在汇编语句中检测符合模式 3.30 的语句。如果格式化代码中不包含此类语句（3.28、3.29、3.30），SolidityCheck 将报告合约中包含该种 bug。
11. Outdated compiler version。 SolidityCheck 首先捕获格式化代码中符合模式 3.31 或 3.32 的代码语句，然后从捕获的语句中提取合约能够兼容的最旧 Solidity 版本（以下称捕获的最旧的 Solidity 版本为此版本）。如果此版本发布早于预定义的标准（现在设置为 0.5.0 版本），则 SolidityCheck 将会报告合约包含该种 bug。
12. Redundant refusal of payment。当使用 Solidity 0.4.0 或更新的 Solidity 版本开发合约时，符合模式 3.33 的语句表示合约中接收外部付款的 fallback 函数，SolidityCheck 将检测在此 fallback 函数体中是否使用了 revert 语句或 throw 语句。如果是，则 SolidityCheck 将报告合约中包含该种 bug。
13. Style guide violation。 SolidityCheck 将匹配格式化代码中符合模式 3.34-3.36 任意之一的语句。
14. Token API violation。首先，SolidityCheck 将会把格式化代码中符合模式 3.37 的代码语句认定为符合 ERC20、ERC721 或 ERC165 的代币合约标准的合约或接口。如果格式化代码中存在符合

模式 3.37 的语句，则该合约即被认为是遵循 ERC20、ERC721 或 ERC165 等代币合约标准之一开发的（下文称为该合约或接口为 ERC 合约）。然后，在 ERC 合约以及直接或间接继承自 ERC 合约的合约中，SolidityCheck 检测符合模式 3.38 所定义的函数中是否存在任何可能会引发异常的语句（例如，require 语句、assert 语句、throw 语句、revert 语句）。如果这些函数中包含会引发异常的语句，则 SolidityCheck 将报告该函数包含此种 bug。

15. Transaction order dependence。 SolidityCheck 将匹配格式化代码中符合模式 3.39 的语句。
16. Unchecked call return value。 SolidityCheck 将匹配格式化代码中符合模式 3.40 的语句。
17. Unencrypted private data on-chain。 SolidityCheck 将匹配格式化代码中符合模式 3.41 但不符合模式 3.42 的语句。
18. Unexpected ether balance。 SolidityCheck 将匹配格式化代码中符合模式 3.43 的语句。
19. Unsafe type inference。 SolidityCheck 将匹配格式化代码中符合模式 3.44 的语句，并要求“=”右侧的数字小于 2^{198} 且大于（-2）^{197}。这是因为超出这两个值的整数变量的类型被推断为 uint256 或 int256，它们是 Solidity 语言中存储范围最大的整数类型。
20. Using fixed point number type。 SolidityCheck 将匹配格式化代码中符合模式 3.45 的语句。

A.2 bug 修复方法

1. Integer overflow and underflow。 SolidityCheck 将格式化代码中符合模式 4.1 或 4.2 的代码语句识别为整数运算语句（表 2 中展示了我们预定义的正则表达式）。
2. Public function that could be declared as external。 SolidityCheck 将格式化代码中符合模式 4.3 的代码语句识别为声明为 public 可见性的函数。
3. Re-entrancy。SolidityCheck 将格式化代码中符合模式 4.4 或 4.5 的代码语句识识为危险语句。
4. Typographical error。SolidityCheck 将格式化代码中符合模式 4.6 或 4.7 的运算符识别为错误的运算符（即，=+或=-）。
5. Use of deprecated Solidity function。SolidityCheck 将格式化代码中符合模式 4.8-4.16 之一的内置符号识别为 Solidity 不推荐的功能，或将符合模式 4.17 但不符合模式 4.18 的内置符号识别为 Solidity 不推荐的功能

A.3 表 3 中的一些细节

在本节中，我们说明附录表 3 中的一些细节。在表 3 中，每个关键字后都跟随一个字符串， 该字符串的含义是：

- s。 当语句包含该字符串时，SolidityCheck 将使用相应的正则表达式来匹配该语句。
- uDigital。 仅当代码语句中包含所有 uDigital 字符串时，SolidityCheck 才会使用相应的正则表达式来匹配该语句。
- n。 如果该语句包含字符串，SolidityCheck 将不使用相应的正则表达式来匹配该语句。

表 1:

Bug name	Regular expressions	Number
Authorization through *tx.origin*	$\wedge(\backslash s) * ((require)\|(if))(\backslash s) * (\backslash()(.) * (tx \setminus .origin)(.) * (\backslash))$	(3.1)
	$((tx \setminus .origin)(\backslash s) * (==)(\backslash s) * (msg \setminus .sender)\|\|(msg \setminus .sender)(\backslash s) * (==)(\backslash s) * (tx \setminus .origin))$	(3.2)
Block values as a proxy for time	$(((\backslash b)(now)(\backslash b))\|((\backslash b)(block.timestamp)(\backslash b)))$	(3.3)
byte[ ]	$(\backslash s) * (byte)(\backslash s) * (\backslash[](\backslash s) * (\backslash])(\backslash s)$	(3.4)
DoS with block gas limit	$(\backslash b)(for)(\backslash s) * (\backslash()(.) * (;)(.) * (\backslash.)(.) * (;)(.) * (\backslash))$	(3.5)
	$(\backslash b)(while)(\backslash s) * (\backslash()(.) * (\backslash.)(.) * (\backslash))$	(3.6)
	$(\backslash b)(for)(\backslash s) * (\backslash()(.) * (;)(.) * (\backslash w) + (.) * (;)(.) * (\backslash))$	(3.7)
	$(\backslash b)(while)(\backslash s) * (\backslash()(.) * (\backslash w) + (.) * (\backslash))$	(3.8)
	$(\backslash b)(for)(\backslash s) * (\backslash()(.) * (;)(.) * (\backslash()(.) * (\backslash))(.) * (;)(.) * (\backslash))$	(3.9)
	$(\backslash b)(while)(\backslash s) * (\backslash()(.) * (\backslash()(.) * (\backslash))(.) * (\backslash))$	(3.10)
DoS with failed call	$((if)\|(require))(\backslash s) * (\backslash()(.) * (\backslash.)(\backslash w) + (\backslash()(.) * (\backslash))(.) * (\backslash))$	(3.11)
	$(for)(\backslash s) * (\backslash()(.) * (;)(.) + (\backslash.)(\backslash w) + (\backslash()(.) * (\backslash))(.) * (;)(.) * (\backslash))$	(3.12)
Floating pragma	$(\backslash s) * (pragma)(\backslash s) + (solidity)(\backslash s) + (\backslash\wedge)(\backslash d)(\backslash.)(\backslash d)(\backslash.)(\backslash d)(\backslash s) * (;)$	(3.13)
	$(\backslash s) * (pragma)(\backslash s) + (solidity)(\backslash s) + (\backslash >)(\backslash =)(\backslash d)(\backslash.)(\backslash d)(\backslash.)(\backslash d)$	(3.14)
	$(\backslash s) * (pragma)(\backslash s) + (experimental)(\backslash s)+$	(3.15)
Implicit visibility level	$\wedge(\backslash s) * ((uint)\|(int))(\backslash d)\{0,3\}(\backslash s) + (\backslash w)+$	(3.16)
	$\wedge(\backslash s) * ((ufixed)\|(fixed))((\backslash d)\{1,3\}(x)(\backslash d)\{0,2\})?(\backslash$	(3.17)
	$\wedge(\backslash s) * (bool)(\backslash s) + (\backslash w)+$	(3.18)
	$\wedge(\backslash s) * (address)(\backslash s) + (\backslash w)+$	(3.19)
	$\wedge(\backslash s) * (mapping)(\backslash s) * (\backslash()(\backslash s) * (\backslash w) + (\backslash$	(3.20)
	$\wedge(\backslash s) * (((bytes)\{0,2\})\|(byte))(\backslash s) + (\backslash w)+$	(3.21)
	$\wedge(\backslash s) * (string)(\backslash s)+$	(3.22)
	$\wedge(\backslash s) * (\backslash w) + (\backslash s) * (\backslash[](.) * (\backslash])(\backslash s)+$	(3.23)
Incorrect constructor name	$\wedge(\backslash s) * (constructor)(\backslash s) * (\backslash()$	(3.24)
	$\wedge(\backslash s) * (function)(\backslash s) * (contractName)(\backslash s) * (\backslash()$	(3.25)
Integer division	$(\backslash d) + (\backslash s) * (\backslash/)(\backslash s) * (\backslash d)+$	(3.26)
Locked ethers	$(\backslash s) * (function)(\backslash s)(.) + (\backslash))(.) * (\backslash s)(payable)((\backslash s)\|(\backslash\{)\|(;))$	(3.27)
	$(\backslash.)((transfer)\|(send))(\backslash s) * (\backslash()(.) + (\backslash))$	(3.28)
	$(\backslash.)(call)(\backslash.)(\backslash s) * ((value)\|(gas)(\backslash(.) + (\backslash))(\backslash.)(value)))(\backslash()$	(3.29)
	$(\backslash b)((delegatecall)\|(staticall)\|(callvalue)\|(call))(\backslash s) * (\backslash()$	(3.30)
Outdated compiler version	$(\backslash s) * (pragma)(\backslash s) + (solidity)(\backslash s) + (.) * (\backslash d)(\backslash.)(\backslash d)(\backslash.)(\backslash d) + (\backslash s) * (;)$	(3.31)
	$(\backslash s) * (pragma)(\backslash s) + (solidity)(\backslash s) + (\backslash >)(\backslash =)(\backslash d)(\backslash.)(\backslash d)(\backslash.)(\backslash d) + (\backslash s)+$	(3.32)
Redundant refusal of payment	$(\backslash b)(function)(\backslash s) * (\backslash()(\backslash s) * (\backslash))(.) * (\backslash s) + (payable)(\backslash s)*$	(3.33)
Style guide violation	$(\backslash b)(function)(\backslash s) + [\wedge a - z](\backslash w)+$	(3.34)
	$(\backslash s) * (event)(\backslash s) + [\wedge A - Z](\backslash w)+$	(3.35)
	$(\backslash b)(\backslash w) + (\backslash s) + (\backslash])(.) * (\backslash])$	(3.36)
Token API violation	$(\backslash s) * ((contract)\|(interface))(\backslash s) + (\backslash w) * ((ERC)\|(ERc)\|(eRC)\|(eRc)\|(ErC)\|(Erc)\|(erC)\|(erc))((20)\|$ $(721)\|(165))(\backslash w) * ((\backslash s)\|(\{)\|(;))$	(3.37)
	$(\backslash s) * (function)(\backslash s) + (\backslash b)((transfer)\|(transferFrom)\|(arrrove)\|(supportsInterface)\|(isApprovedFo rAll))(\backslash b)$	(3.38)
Transaction order dependence	$(\backslash s) * (function)(\backslash s) + (approve)(\backslash()(address)(\backslash s) + (\backslash w) + (\backslash s) * (\backslash,)(\backslash s) * ((uint256)\|(uint))(\backslash s) +$ $(\backslash w) + (\backslash))(\backslash s) + ((public)\|(external))(\backslash s) + (returns)(\backslash s) * (\backslash()(bool)(.) * (\backslash))$	(3.39)
Unchecked call return value	$(\wedge(\backslash s)*((if)\|(require))(\backslash s)*(\backslash()(.)*((\backslash w)\|(\backslash()\|(\backslash))\|(\backslash[]\|(\backslash])\|(.))((send)\|(delegatecall)\|(call)\|(callcode))(.)*$ $(\backslash()(.) * (\backslash))(.) * (\backslash)))$	(3.40)
Unencrypted private data on-chain	$(\backslash b)(private)(\backslash b)$	(3.41)
	$(\backslash b)(function)(\backslash b)$	(3.42)
Unexpected ether balance	$\wedge(\backslash s)((if)\|(while)\|(require))(\backslash s)*(\backslash()(.)*(((this.balance)(\backslash s)*(==)(\backslash s)*(\backslash d))+(ether))\|((\backslash d)+(\backslash s)*$ $(ether)(\backslash s) * (==)(\backslash s) * (this.balance)))(.) * (\backslash))(.) * \$$	(3.43)
Unsafe type inference	$(\backslash b)(var)(\backslash b)(\backslash s) + (\backslash w) + (\backslash s) * (=)(\backslash s) * (\backslash d) + (\backslash b)$	(3.44)
Using fixed point number type	$(\backslash b)((unfixed)(fixed))((\backslash d)\{1,3\}(x)(\backslash d)\{0,2\})?(\backslash s) + (\backslash w)+$	(3.45)

表 2:

Bug name	Regular expressions	Number
Integer overflow and underflow	$\wedge(\backslash s)*(\backslash w)*(\backslash s)+((\backslash w)\|(\backslash()\|(\backslash))\|(\backslash[)\|(\backslash])\|(\backslash.))+(\backslash s)*(=)(\backslash s)*((\backslash w)\|(\backslash()\|(\backslash))\|(\backslash[)\|(\backslash])\|(\backslash.))+(\backslash s)*((\backslash+)\|(\backslash-)\|(\backslash*)\|(\backslash/)\|(\backslash\%))(\backslash s)*((\backslash w)\|(\backslash()\|(\backslash))\|(\backslash[)\|(\backslash])\|(\backslash.))+(\backslash s)*(;)\$$ $\wedge(\backslash s)*(\backslash w)*(\backslash s)+((\backslash w)\|(\backslash()\|(\backslash))\|(\backslash[)\|(\backslash])\|(\backslash.))+(\backslash s)*(((\backslash+)(\backslash\ =))\|((\backslash-)(\backslash\ =))\|(\backslash*)(\backslash\ =)\|(\backslash/)(\backslash\ =)\|(\backslash\%)(\backslash\ =))(\backslash s)*((\backslash w)\|(\backslash()\|(\backslash))\|(\backslash[)\|(\backslash])\|(\backslash.))+(\backslash s)*(;)\$$	(4.1) (4.2)
Public function that could be declared as external	$\wedge(\backslash s)*(function)(\backslash s)+(\backslash w)*(\backslash()(.)*(\backslash))(\backslash s)+(.)*(\backslash b)(public)(\backslash b)(.)*(\{)\$$	(4.3)
Reentrancy	$(.)+(\backslash.)(call)(\backslash.)(value)(\backslash s)*(\backslash()(.)+(\backslash))(\backslash()(\backslash))$ $(.)+(\backslash.)(call)(\backslash.)(value)(\backslash s)*(\backslash()(.)+(\backslash))(\backslash()"")(\backslash))$	(4.4) (4.5)
Typographical error	$(\backslash\ =)(\backslash s)*(\backslash+)$ $(\backslash\ =)(\backslash s)*(\backslash-)$	(4.6) (4.7)
Use of deprecated Solidity functions	$(\backslash s)(suicide)(\backslash s)*(\backslash()$ $(\backslash s)(block)(\backslash s)*(\backslash.)(\backslash s)*(blockhash)(\backslash()$ $(\backslash s)(sha3)(\backslash s)*(\backslash()$ $(\backslash.)(\backslash s)*(callcode)(\backslash s)*(\backslash()$ $(\backslash s)(throw)(\backslash b)$ $(\backslash s)(msg)(\backslash s)*(\backslash.)(\backslash s)*(gas)(\backslash b)$ $(\backslash s)(var)(\backslash b)$ $(\backslash s)(constant)(\backslash b)$ $(\backslash s)(function)(\backslash s)$	(4.8) (4.9) (4.10) (4.11) (4.12) (4.13) (4.14) (4.15) (4.16)

表 3:

Bug name	Keywords	Bug name	Keywords
Authorization through *tx.origin*	tx.origin(s)	Block values as a proxy for time	now(s), block.number(s), block.coinbase(s), block.difficulty(s), blockhash(s)
byte[ ]	byte(s)	DoS with block gas limit	for(s) , for((s), while(s) , while((s)
DoS with failed call	if(s) , if((s), for(s) , for((s), require(s) , require((s), while(s) , while((s)	Floating pragma	pragma(u1), solidity(u1)
Implicit visibility level	function(s), constructor(s), modifier(s), mapping(s), [(u1),](u1)	Incorrect constructor name	contract(s), constructor(s), function(s)
Integer division	/(s)	Locked ethers	payable(s), .send(s), .transfer(s), .call(s)
Outdated compiler version	pragma(u1), solidity(u1), ;(u1)	Redundant refusal of payment	pragma(u1), solidity(u1), function(s)
Style guide violation	function(s), event(s), [(u1),](u1)	Token API violation	contract(s), interface(s), ERC20(s), ERC721(s), ERC165(s), function(u1,u2, u3, u4, u5), approve(u1), transfer(u2), transferFrom(u3), isApprovedForAll(u4), supportsInterface(u5)
Transaction order dependence	approve(s)	Unchecked call return value	.send(s), .call(s), .delegatecall(s), callcode(s)
Unencrypted private data on-chain	private(u1, u2), function(un1), modifier(un2)	Unexpected ether balance	.balance(s)
Unsafe type inference	var(s)	Using fixed point number type	fixed(s)
Integer overflow and underflow	library(u1), contract(u2), safemath(u1, u2), +(s), -(s), *(s), /(s), %(s)	Public function that could be declared as external	function(u1), public(u1)
Reentrancy	.call.value(s)	Typographical error	=(u1, u2), +(u1), -(u2)
Use of deprecated Solidity functions	suicide(s), block(u1), .(u1, u2), blockhash(u1), sha3(s), callcode(s), throw(s), msg(u2), gas(u2), constant(s), var(s)		