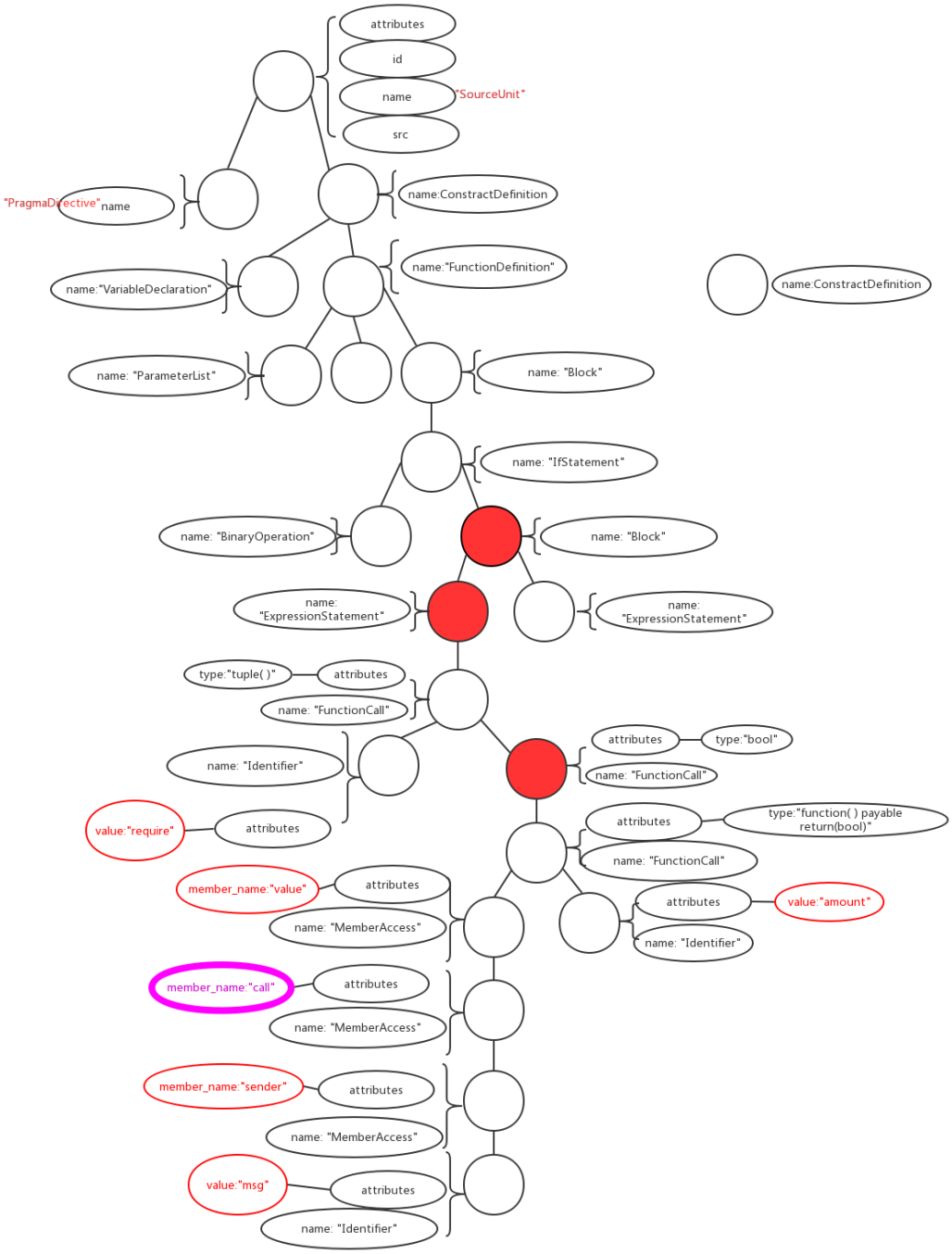
漏洞类型总结https://blog.sigmaprime.io/solidity-security.html#ether

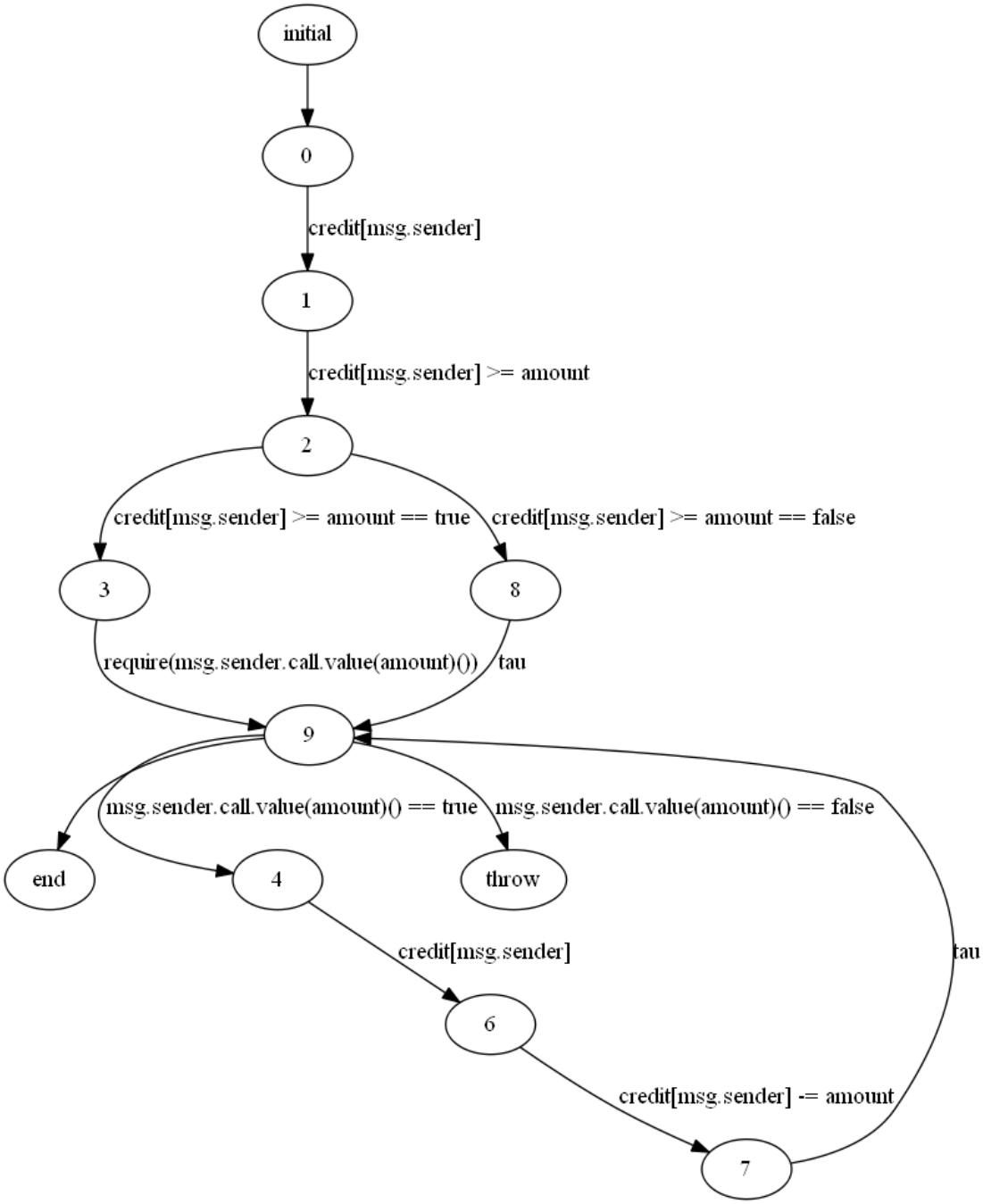
1. DAO

```
pragma solidity ^0.4.24;
contract SimpleDAO {
    mapping (address => uint) public credit;
    function donate(address to) payable public{
        credit[to] += msg.value;
    }
    function queryCredit(address to) view public returns (uint){
        return credit[to];
    }
    function withdraw(uint amount) public{
        if (credit[msg.sender]>= amount){
            require(msg.sender.call.value(amount)());
            credit[msg.sender]-=amount;
        }
    }
}
```

source


attributes
id
name
"SourceUnit"
src
"PragmaDirective"
name
name:ContractDefinition
name:"VariableDeclaration"
name:"FunctionDefinition"
name:ContractDefinition
name: "ParameterList"
name: "Block"
name: "IfStatement"
name: "BinaryOperation"
name: "Block"
name:
"ExpressionStatement"
name:
"ExpressionStatement"
type:"tuple( )"
attributes
name: "FunctionCall"
name: "Identifier"
attributes
type:"bool"
name: "FunctionCall"
value:"require"
attributes
attributes
type:"function( ) payable
return(bool)"
name: "FunctionCall"
member_name:"value"
attributes
name: "MemberAccess"
attributes
value:"amount"
name: "Identifier"
member_name:"call"
attributes
name: "MemberAccess"
member_name:"sender"
attributes
name: "MemberAccess"
value:"msg"
attributes
name: "Identifier"

AST

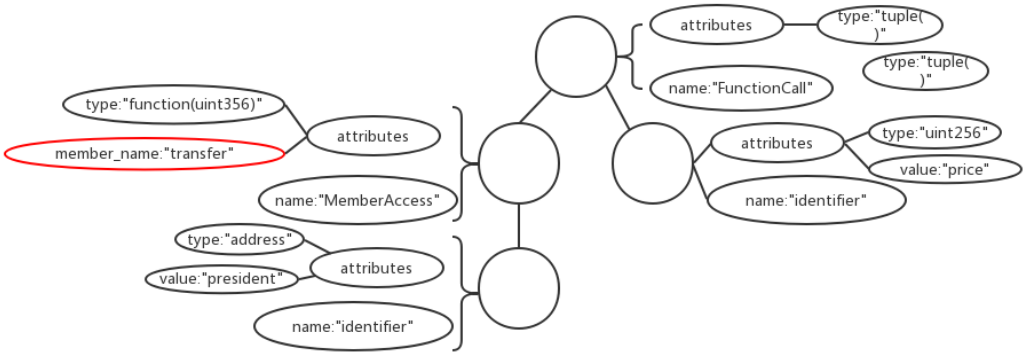

initial
0
credit[msg.sender]
1
credit[msg.sender] >= amount
2
credit[msg.sender] >= amount == true
credit[msg.sender] >= amount == false
3
8
require(msg.sender.call.value(amount)())
tau
9
msg.sender.call.value(amount)() == true
msg.sender.call.value(amount)() == false
end
4
throw
credit[msg.sender]
tau
6
credit[msg.sender] -= amount
7

CFG

2. transfer( ):

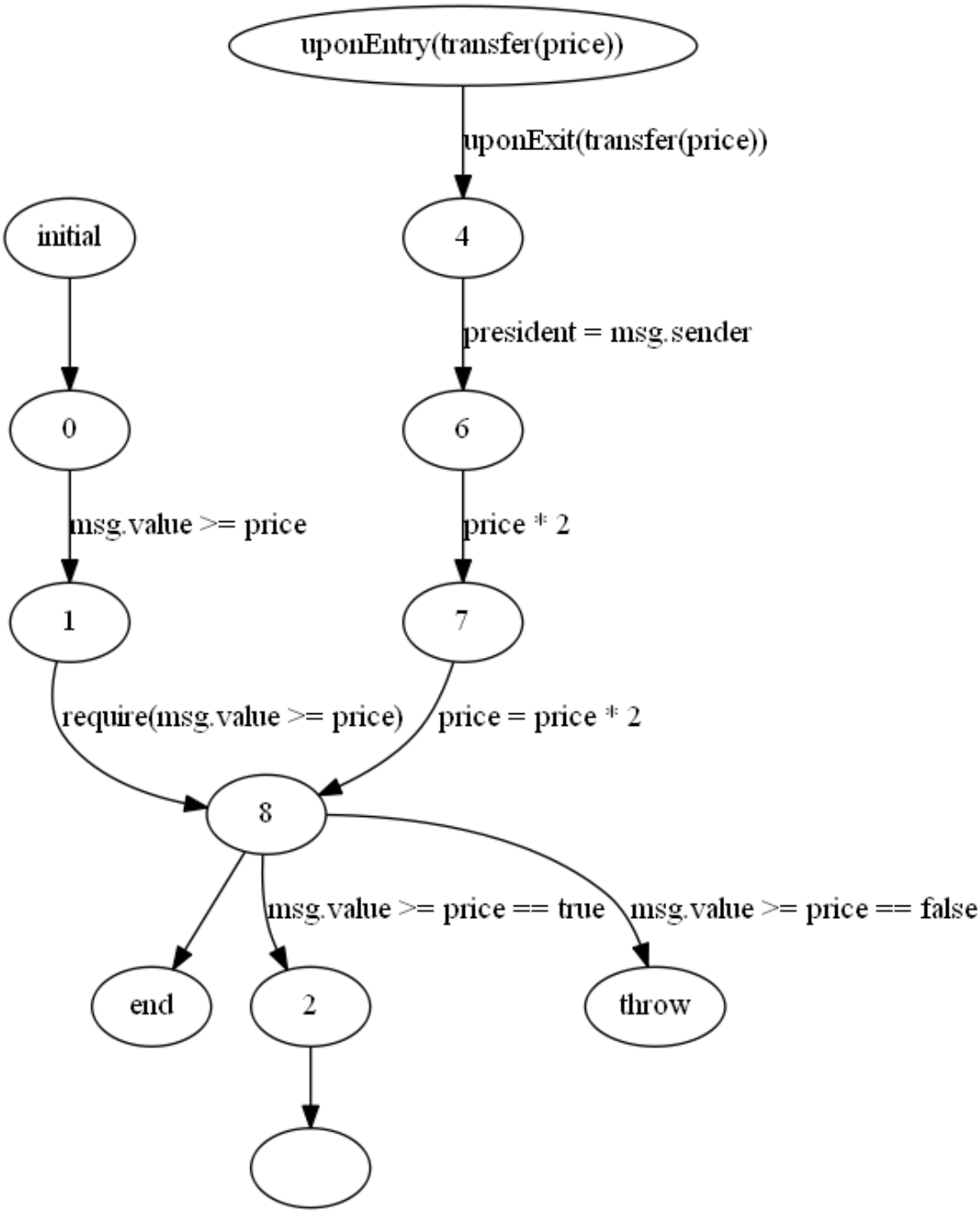
```
pragma solidity ^0.4.10;

contract PresidentOfCountry {
    address public president;
    uint256 price;

    function PresidentOfCountry(uint256 _price) {
        require(_price > 0);
        price = _price;
        president = msg.sender;
    }

    function becomePresident() payable {
        require(msg.value >= price);
        president.transfer(price);
        president = msg.sender;
        price = price * 2;
    }
}
```

source



president.transfer(price);

AST

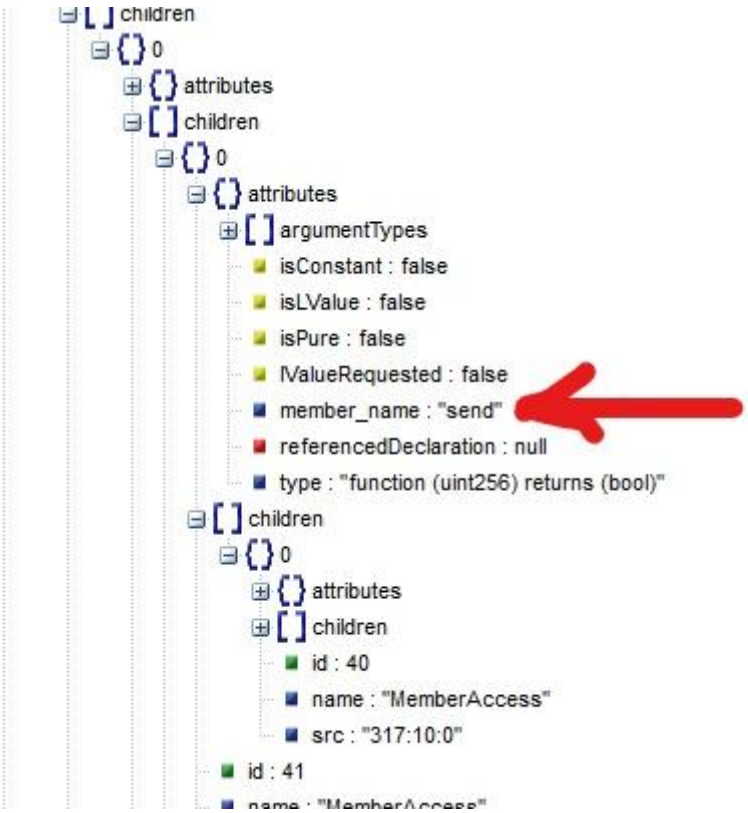

uponEntry(transfer(price))
uponExit(transfer(price))
initial
4
president = msg.sender
0
6
msg.value >= price
price * 2
1
7
require(msg.value >= price)
price = price * 2
8
msg.value >= price == true
msg.value >= price == false
end
2
throw

CFG

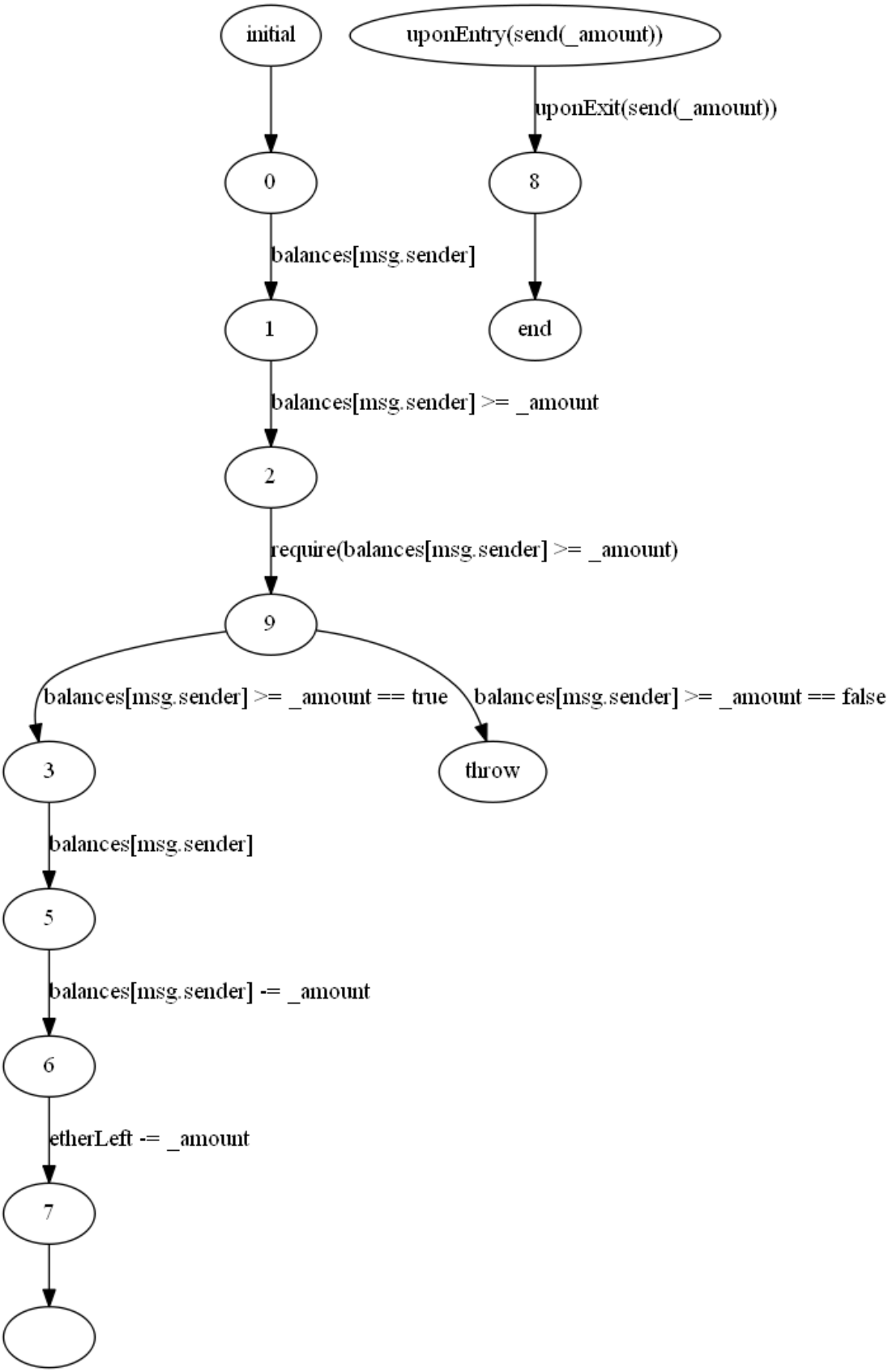
3. send( )

```
pragma solidity ^0.4.10;

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

    address owner;

    function KotET(){
        owner = msg.sender;

    }

    function withdraw(uint256 _amount) public {
        require(balances[msg.sender] >= _amount);
        balances[msg.sender] -= _amount;
        msg.sender.send(_amount);
    }
}
```

Source



AST


initial
uponEntry(send(_amount))
uponExit(send(_amount))
0
8
balances[msg.sender]
1
end
balances[msg.sender] >= _amount
2
require(balances[msg.sender] >= _amount)
9
balances[msg.sender] >= _amount == true
balances[msg.sender] >= _amount == false
3
throw
balances[msg.sender]
5
balances[msg.sender] -= _amount
6
etherLeft -= _amount
7

CFG

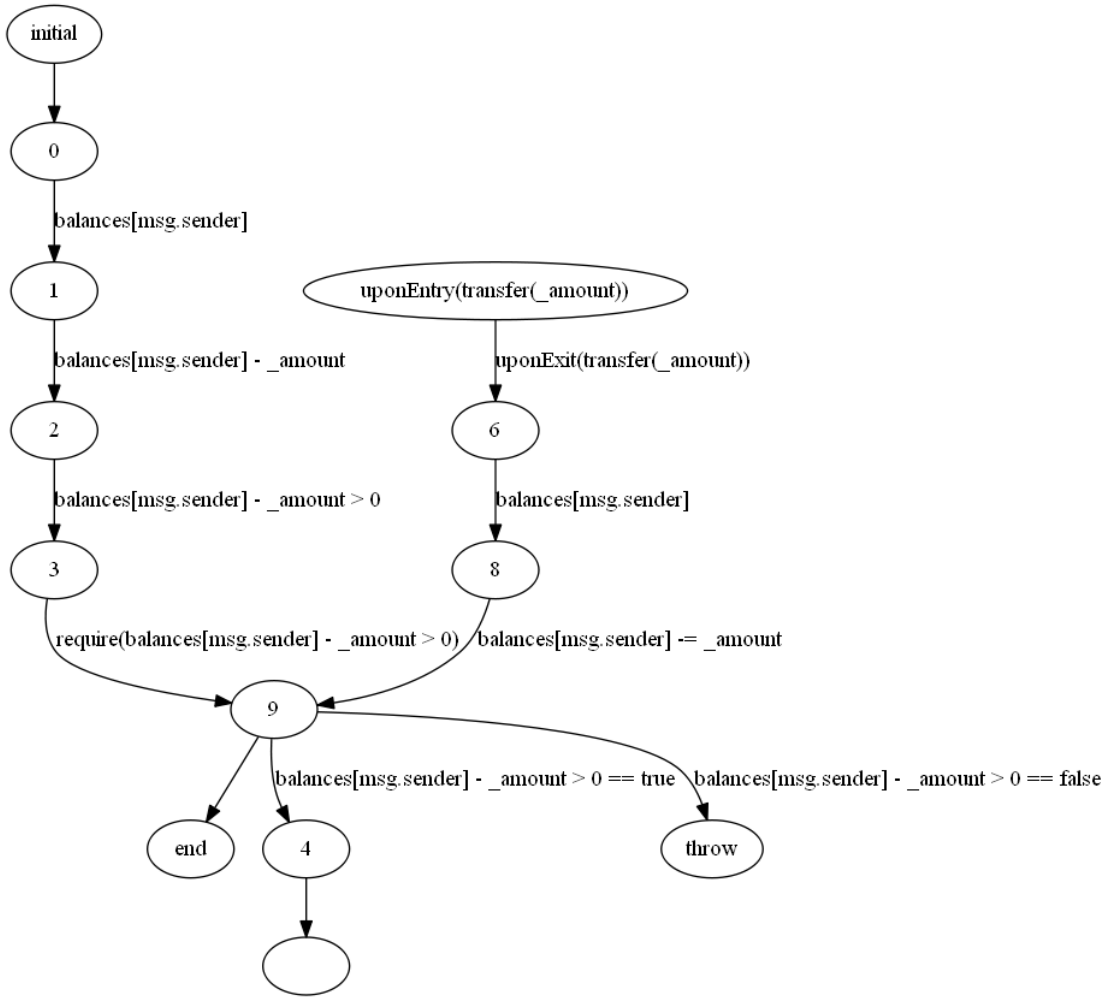
4. Overflow

```
pragma solidity ^0.4.10;

contract MyToken {
    mapping (address => uint) balances;
    function withdraw(uint  amount) {
        require(balances[msg.sender] - _amount > 0);  // 存在整数溢出
        msg.sender.transfer(_amount);
        balances[msg.sender] -= _amount;
    }
    function balanceOf(address _user) returns (uint) { return balances[_user]; }
    function deposit() payable { balances[msg.sender] += msg.value; }

}
```

Source

require(balances[msg.sender] - _amount > 0); // 存在整数溢出

AST



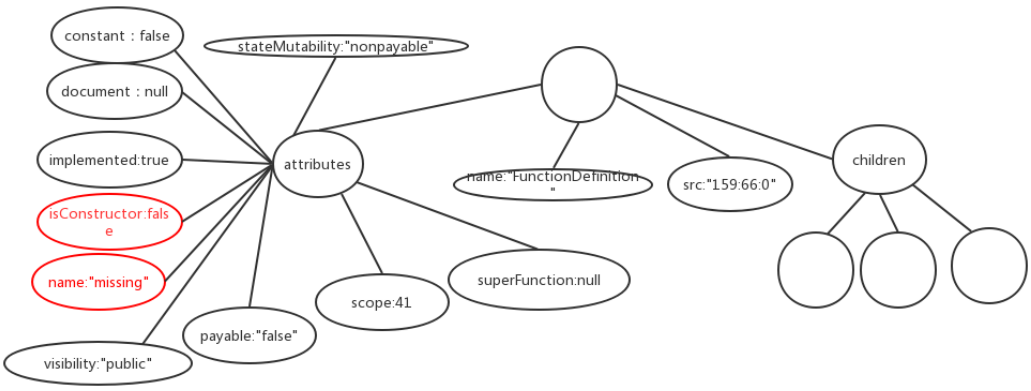

```
digraph withdraw{
initial -> "0";
"9" -> end;"0" -> "1" [label = "balances[msg.sender]"];
"1" -> "2" [label = "balances[msg.sender] - _amount"];
"2" -> "3" [label = "balances[msg.sender] - _amount > 0"];
"4" -> ""6" : transfer(_amount)" [label = "uponEntry(transfer(_amount))"];
""6" : transfer(_amount)" -> "6" [label = "uponExit(transfer(_amount))"];
"6" -> "8" [label = "balances[msg.sender]"];
"8" -> "9" [label = "balances[msg.sender] -= _amount"];
"9" -> throw [label = "balances[msg.sender] - _amount > 0 == false"];
"3" -> "9" [label = "require(balances[msg.sender] - _amount > 0)"];
"9" -> "4" [label = "balances[msg.sender] - _amount > 0 == true"];

}
```

CFG

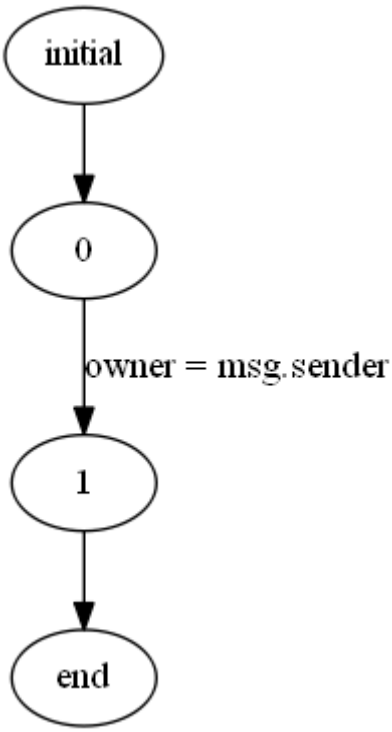
5. incorrect_constructor

```
pragma solidity ^0.4.10;

contract Missing{
    address private owner;

    modifier onlyowner {
        require(msg.sender==owner);
        _;
    }

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

    function () payable {}

    function withdraw() public onlyowner
    {
       owner.transfer(this.balance);
    }
}
```

source



AST



CFG

/***/

```
contract OddsAndEvens{
  struct Player { address addr; uint number;}
  Player[2] private players;
  uint8 tot = 0; address owner;

  function OddsAndEvens() {owner = msg.sender;}

  function play(uint number) {
    if (msg.value != 1 ether) throw;
    players[tot] = Player(msg.sender, number);
    tot++;
    if (tot==2) andTheWinnerIs();
  }

  function andTheWinnerIs() private {
    uint n = players[0].number
           + players[1].number;
    players[n%2].addr.send(1800 finney);
    delete players;
    tot=0;
  }

  function getProfit() {
    owner.send(this.balance);
  }
}
```

```
contract SetProvider {

  address setLibAddr;
  address owner;

  function SetProvider() {
    owner = msg.sender;
  }

  function updateLibrary(address arg) {
    if (msg.sender==owner)
      setLibAddr = arg;
  }

  function getSet() returns (address) {
    return setLibAddr;
  }
}
```

```
library Set {
  struct Data { mapping(uint => bool) flags; }

  function insert(Data storage self, uint value)
      returns (bool) {
    self.flags[value] = true;
    return true;
  }
  function remove(Data storage self, uint value)
      returns (bool) {
    self.flags[value] = false;
    return true;
  }
  function contains(Data storage self, uint value)
      returns (bool) {
    return self.flags[value];
  }
  function version() returns(uint) { return 1; }
}
```

p19

```
library Set { function version() returns (uint); }
contract Bob {
  SetProvider public provider;
  function Bob(address arg) { provider = SetProvider(addr); }
  function getSetVersion() returns (uint) {
    address setAddr = provider.getSet();
    return Set(setAddr).version();
}}
```

Now, assume that the owner of `setProvider` is also an adversary. She can attack `Bob` as follows, with the goal of stealing all his ether. In the first step of the attack, the adversary publishes a new library `MaliciousSet`, and then it invokes the function `updateLibrary` of `SetProvider` to make it point to `MaliciousSet`.

恶意的

```
library MaliciousSet {
  address constant attackerAddr = 0x42;
  function version() returns(uint) {
    attackerAddr.send(this.balance);
    return 1;
}}
```

p20

```
pragma solidity ^0.4.10;

contract ICoin {
    address owner;
    mapping (address => uint256) public balances;

    modifier OwnerOnly() { require(msg.sender == owner); _; }

    function ICoin() { owner = msg.sender; }
    function approve(address _to, uint256 _amount) OwnerOnly { balances[_to] += _amount; }
    function transfer(address _to, uint256 _amount) {
        require(balances[msg.sender] > _amount);
        balances[msg.sender] -= _amount;
        balances[_to] += _amount;
    }
}
```

具体代币功能的合约 ICoin，当 A 账户向 B 账户转代币时调用 `transfer()` 函数，例如 A 账户（0x14723a09acff6d2a60dcdf7aa4aff308fddc160c） 向 B 账户（0x4b0897b0513fdc7c541b6d9d7e929c4e5364d2db）转 8 个 ICoin，`msg.data` 数据为：

```
0xa9059cbb  -> bytes4(keccak256("transfer(address,uint256)")) 函数签名
0000000000000000000000004b0897b0513fdc7c541b6d9d7e929c4e5364d2db  -> B 账户地址（前补 0 补齐 32 字
0000000000000000000000000000000000000000000000000000000000000008  -> 0x8（前补 0 补齐 32 字节）
```

那么短地址攻击是怎么做的呢，攻击者找到一个末尾是 `00` 账户地址，假设为 `0x4b0897b0513fdc7c541b6d9d7e929c4e5364d200`，那么正常情况下整个调用的 `msg.data` 应该为：

```
0xa9059cbb  -> bytes4(keccak256("transfer(address,uint256)")) 函数签名
0000000000000000000000004b0897b0513fdc7c541b6d9d7e929c4e5364d200  -> B 账户地址（注意末尾 00）
0000000000000000000000000000000000000000000000000000000000000008  -> 0x8（前补 0 补齐 32 字节）
```

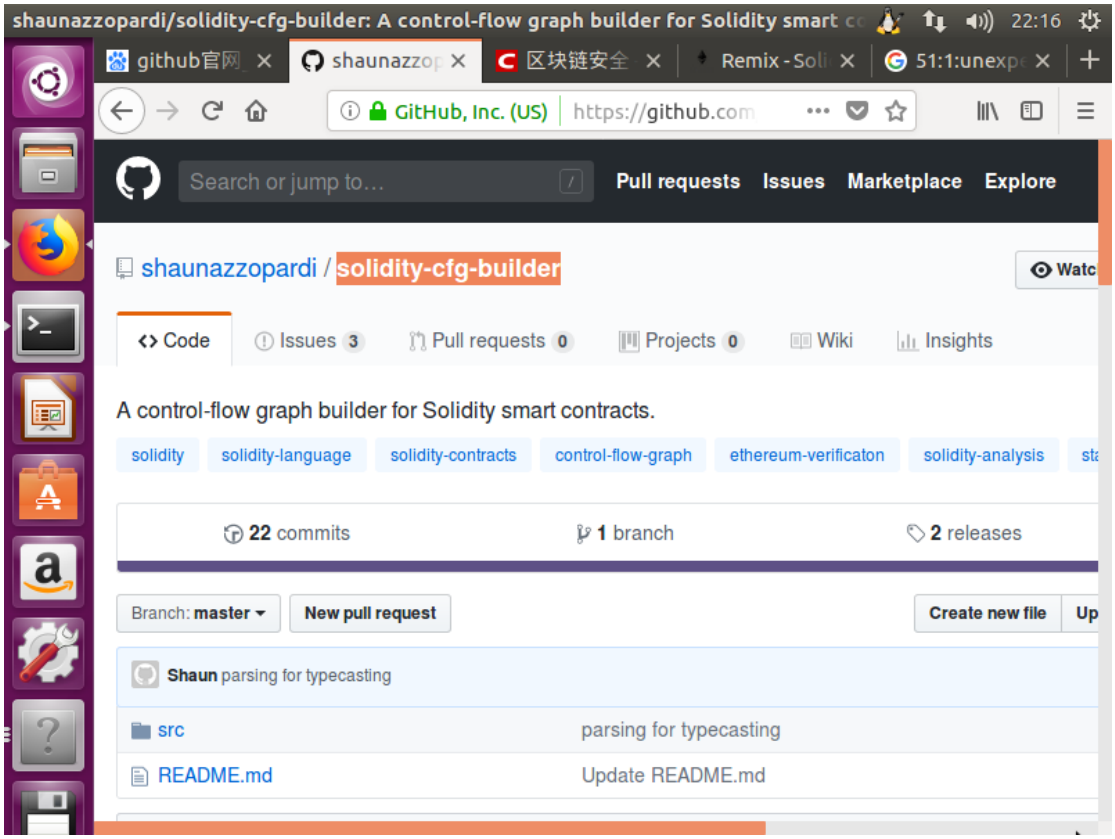
但是如果我们将 B 地址的 `00` 吃掉，不进行传递，也就是说我们少传递 1 个字节变成 `4+31+32`：

```
0xa9059cbb  -> bytes4(keccak256("transfer(address,uint256)")) 函数签名
0000000000000000000000004b0897b0513fdc7c541b6d9d7e929c4e5364d2  -> B 地址（31 字节）
0000000000000000000000000000000000000000000000000000000000000008  -> 0x8（前补 0 补齐 32 字节）
```

当上面数据进入 EVM 进行处理时，会犹豫参数对齐的问题后补 `00` 变为：

```
0xa9059cbb
0000000000000000000000004b0897b0513fdc7c541b6d9d7e929c4e5364d200
0000000000000000000000000000000000000000000000000000000000000800
```

CFG:工具


shaunazzopardi/solidity-cfg-builder: A control-flow graph builder for Solidity smart
22:16
github官网
区块链安全
Remix - Soli
GitHub, Inc. (US)
https://github.com
Search or jump to…
Pull requests
Issues
Marketplace
Explore
shaunazzopardi / solidity-cfg-builder
Code
Issues 3
Pull requests 0
Projects 0
Wiki
Insights
A control-flow graph builder for Solidity smart contracts.
solidity
solidity-language
solidity-contracts
control-flow-graph
ethereum-verificaton
solidity-analysis
22 commits
1 branch
2 releases
Branch: master
New pull request
Create new file
Shaun parsing for typecasting
src
parsing for typecasting
README.md
Update README.md

Building the tool

Requirements: GHC (e.g. install the full Haskell Platform)

Compilation: Run the following command in the src folder:

```
ghc -o solidity-cfg-builder Main.hs
```

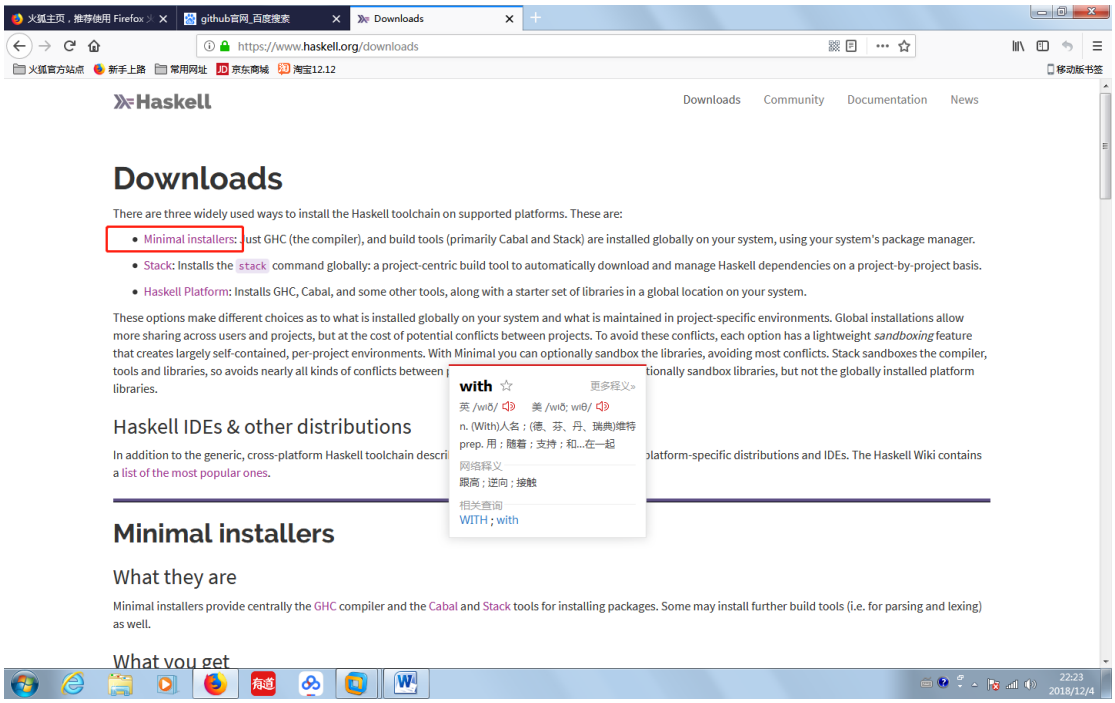
Tool usage:

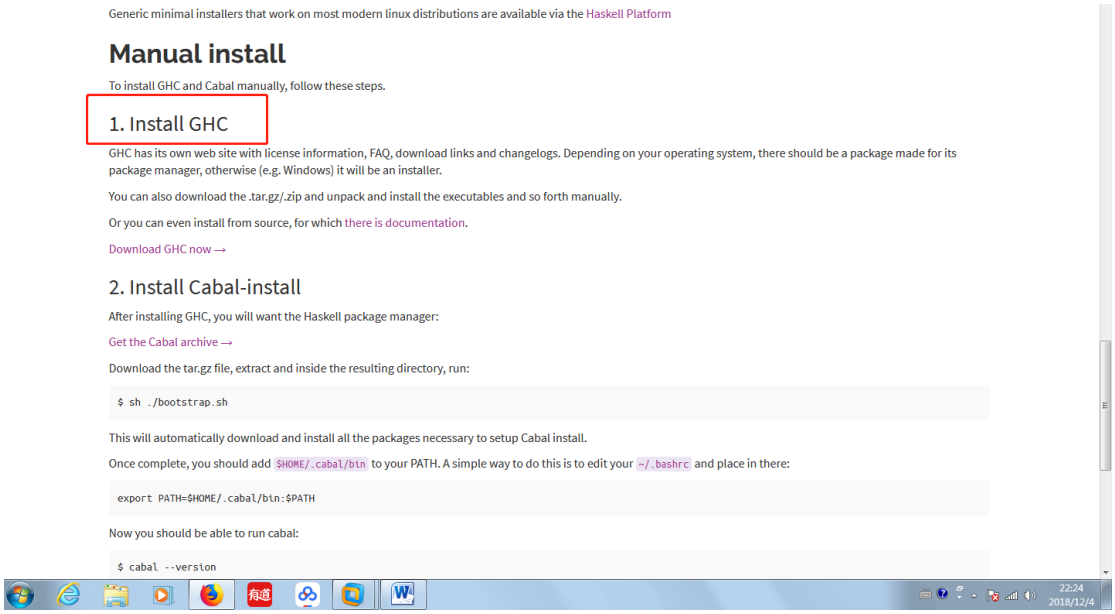
For correct results always make sure that the Solidity code compiles with a Solidity compiler.

To use the tool pass the location of a solidity file and the preferred location of the output to the e

```
"./solidity-cfg-builder" <solidity-code.sol> <cfg.gv>
```

在 linux 上跑。



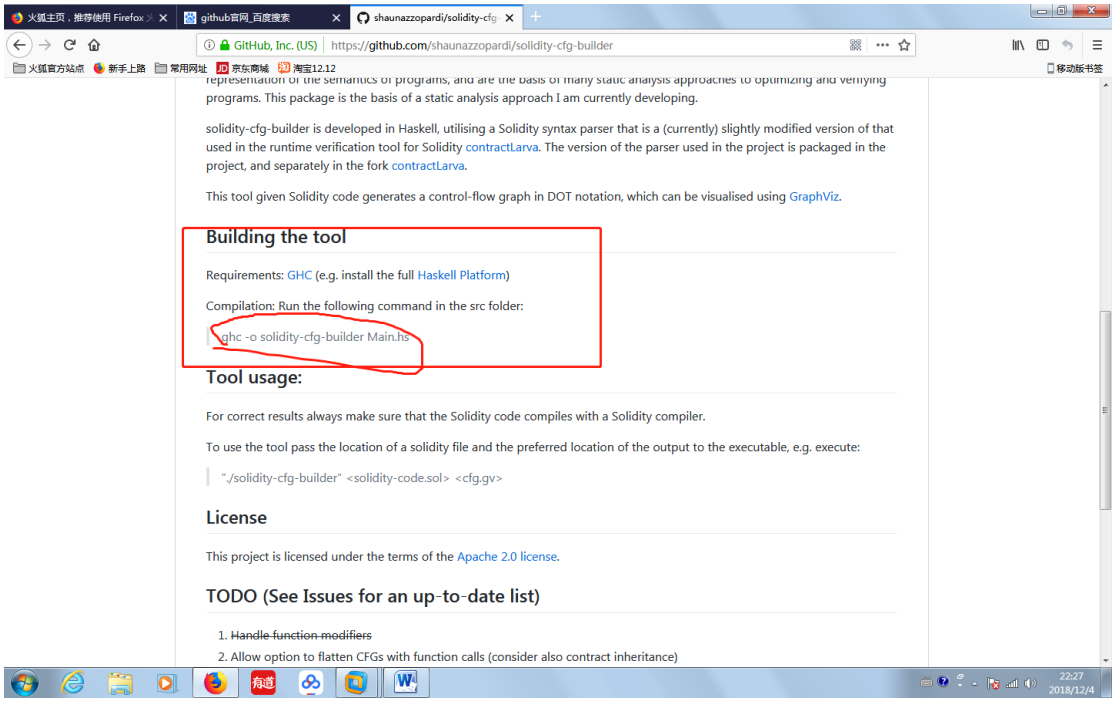


Sudo install cabal-install

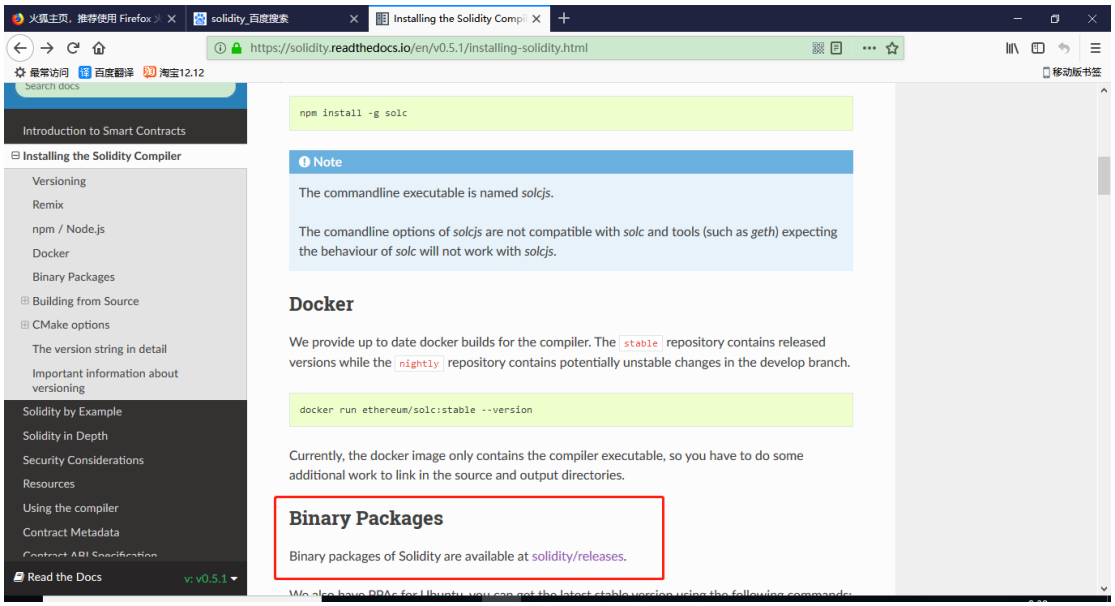
Could not find module ‘Text.Parsec’.

Text.Parsec.Numbe 出错那行删除。

最后


programs. This package is the basis of a static analysis approach I am currently developing.
solidity-cfg-builder is developed in Haskell, utilising a Solidity syntax parser that is a (currently) slightly modified version of that used in the runtime verification tool for Solidity contractLarva. The version of the parser used in the project is packaged in the project, and separately in the fork contractLarva.
This tool given Solidity code generates a control-flow graph in DOT notation, which can be visualised using GraphViz.
Building the tool
Requirements: GHC (e.g. install the full Haskell Platform)
Compilation: Run the following command in the src folder:
ghc -o solidity-cfg-builder Main.hs
Tool usage:
For correct results always make sure that the Solidity code compiles with a Solidity compiler.
To use the tool pass the location of a solidity file and the preferred location of the output to the executable, e.g. execute:
"./solidity-cfg-builder" <solidity-code.sol> <cfg.gv>
License
This project is licensed under the terms of the Apache 2.0 license.
TODO (See Issues for an up-to-date list)
1. Handle function modifiers
2. Allow option to flatten CFGs with function calls (consider also contract inheritance)

AST 工具


Introduction to Smart Contracts
Installing the Solidity Compiler
Versioning
Remix
npm / Node.js
Docker
Binary Packages
Building from Source
CMake options
The version string in detail
Important information about versioning
Solidity by Example
Solidity in Depth
Security Considerations
Resources
Using the compiler
Contract Metadata
Read the Docs
npm install -g solc
Note
The commandline executable is named solcjs.
The comandline options of solcjs are not compatible with solc and tools (such as geth) expecting the behaviour of solc will not work with solcjs.
Docker
We provide up to date docker builds for the compiler. The stable repository contains released versions while the nightly repository contains potentially unstable changes in the develop branch.
docker run ethereum/solc:stable --version
Currently, the docker image only contains the compiler executable, so you have to do some additional work to link in the source and output directories.
Binary Packages
Binary packages of Solidity are available at solidity/releases.