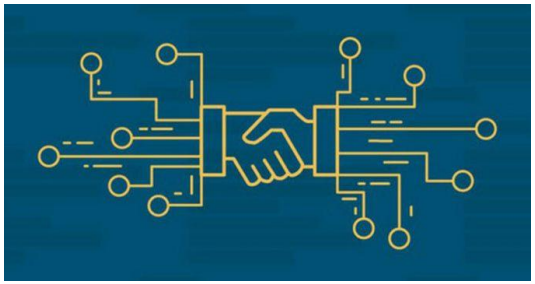
(In)Segurança em Smart Contracts

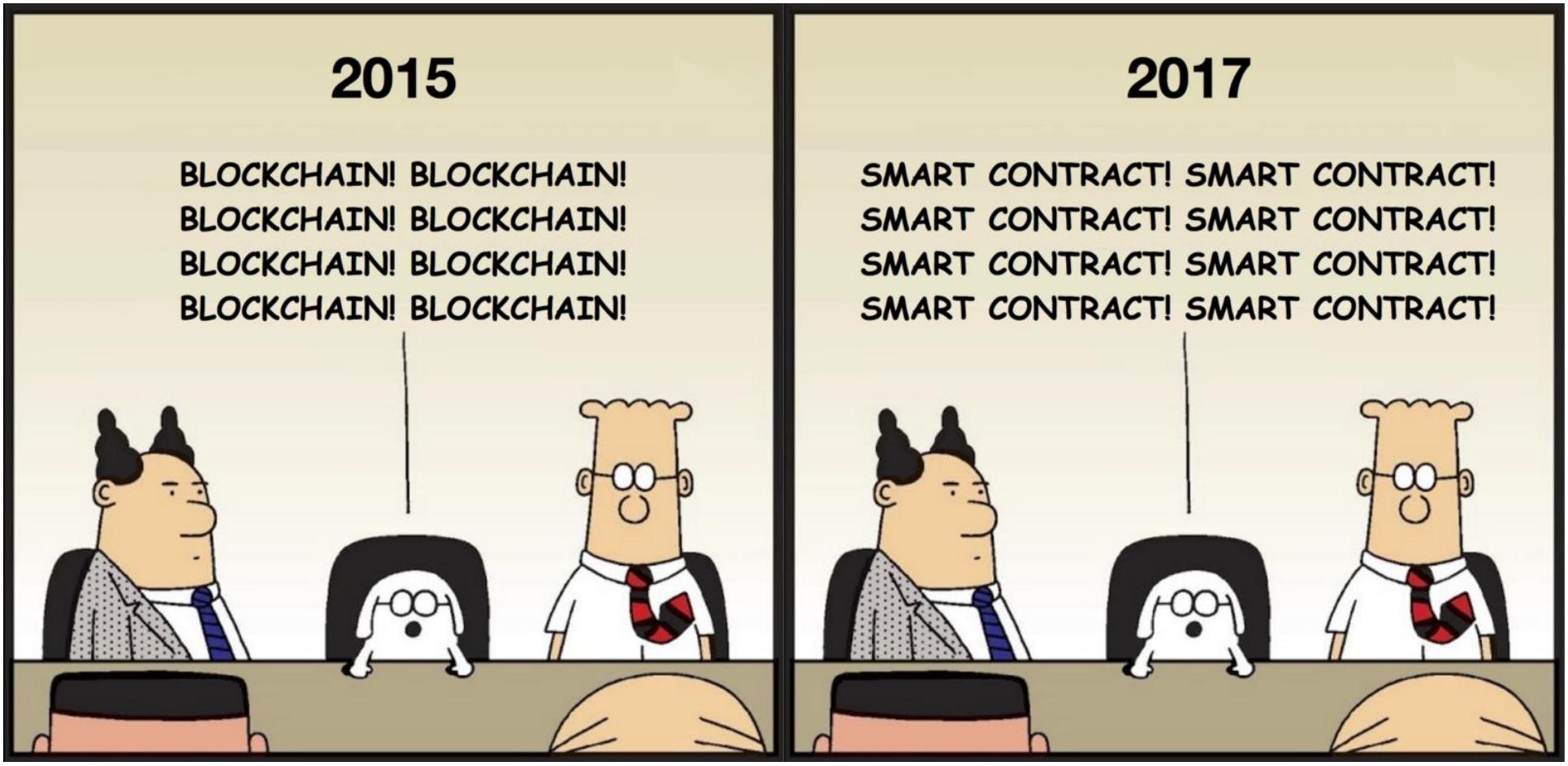
Luciano Porto Barreto

lbarreto@cvm.gov.br



maio/2019

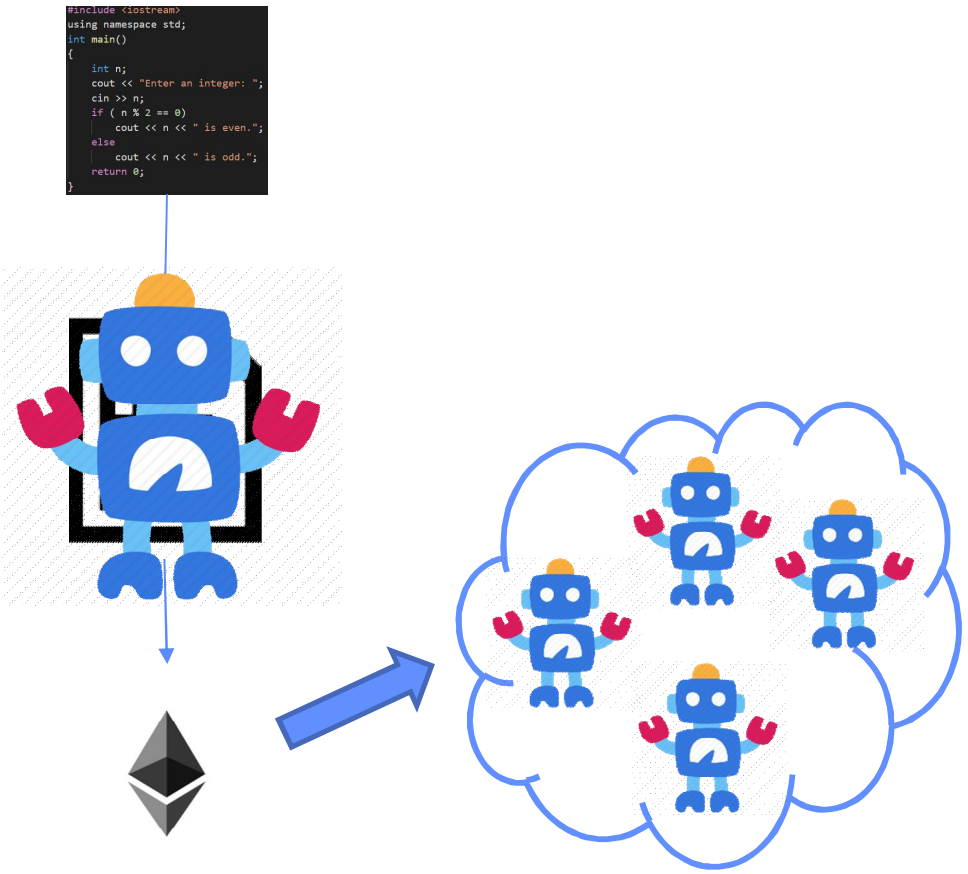
Dogbert, o consultor



"Smart Contracts"

Programa executável

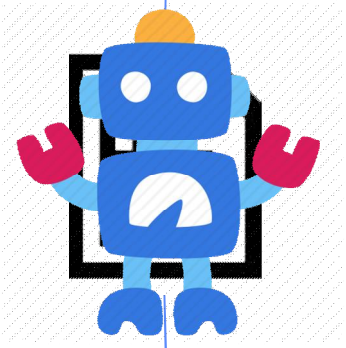
Smart contract

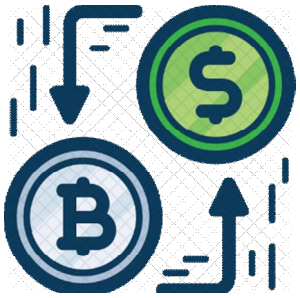


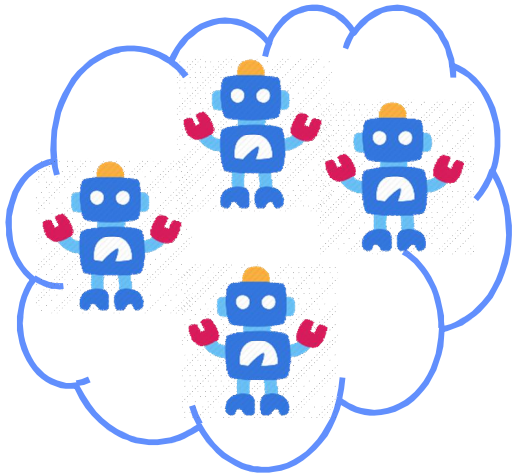
"Smart Contracts"

Smart Contract

```
#include <iostream>
using namespace std;
int main()
{
    int n;
    cout << "Enter an integer: ";
    cin >> n;
    if ( n % 2 == 0)
        cout << n << " is even.";
    else
        cout << n << " is odd.";
    return 0;
}
```







"Smart Contracts"

- **Conceito:**
  - São programas de computador que implementam acordos digitais autoexecutáveis entre terceiros.
  - Origem: Nick Szabo (1997)
  - Hoje: programa armazenado e executado em Blockchains
    - » "Programas/scripts persistentes"

ethereum

- **Uso principal:**
  - Oferta pública conhecida como ICO: *Initial Coin Offering*
    - » Criptoativos "pré-minerados"
    - » Clientes adquirem novo criptoativo (por troca ou compra)
    - » SC implementa as regras de aquisição
- **Outros:**
  - Automação contratual:
    - » Apólices de seguros, venda de ativos, crowdfunding, jogos

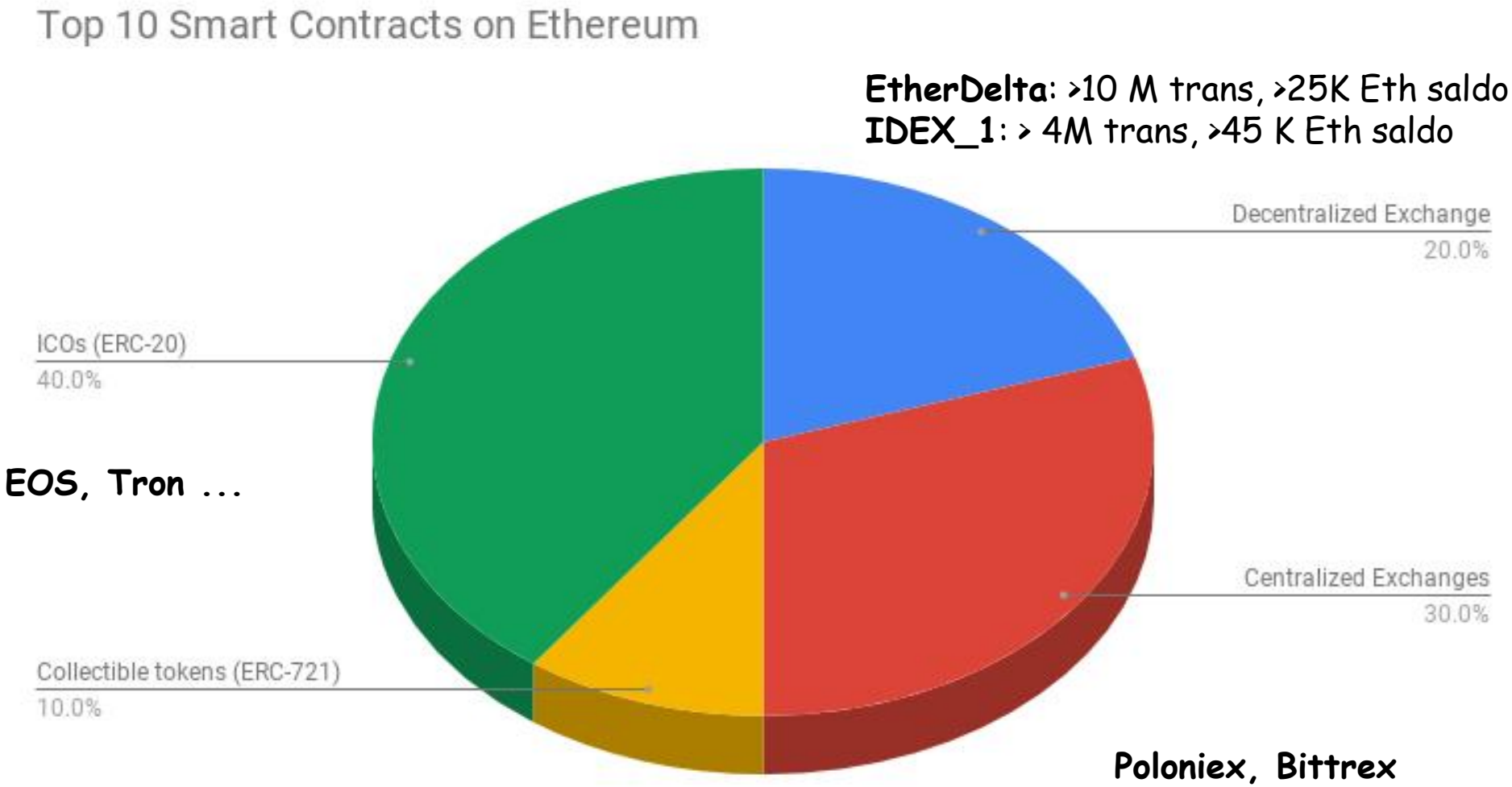
Características

- **Publicidade/Transparência:**
  - Contratos públicos, auditáveis, verificáveis
  - Em tese, qualquer um pode "ativar" um contrato
  - Terceiros deveriam entender as cláusulas contratuais (contratos negociados vs contratos de adesão)
- **Ubiquidade:**
  - Todos os nós participantes da rede devem executar operações enviadas para um contrato publicado
  - Maioria (consenso) segue o protocolo
- **Imutabilidade:**
  - Não podem ser modificados após sua publicação (como contratos após assinatura)
- **Dependências definidas:**
  - Referências externas devem ser hard-coded

Plataformas de desenvolvimento

- **Ethereum:**
  - Solidity (a la JavaScript) -> EVM (Ethereum Virtual Machine)
- **HyperLedger Fabric:**
  - Go
- **EOS:**
  - WASM (WebAssembly)
- **Stellar:**
  - Linguagem restrita
- **Cardano:**
  - Plutus (a la Haskell)
- **Outros:**
  - Azure+Ethereum

Dez maiores contratos por número de transações



EtherDelta: >10 M trans, >25K Eth saldo
IDEX_1: > 4M trans, >45 K Eth saldo

EOS, Tron ...

Poloniex, Bittrex

CryptoKitties

Última atualização: 24/09/2018

Exemplo: CriptoKitties

- **Jogo na Blockchain: (sc)**
  - Obj: Colecionar e criar gatos virtuais
    - » Contrato
  - Vendas: $ 27.419.101; Preço médio: $48;
  - Sucesso provocou congestão da rede Ethereum
    - » Questionamentos sobre escalabilidade
  - Aumento nas taxas

  - https://kittysales.herokuapp.com/
  - https://www.kittyexplorer.com/stats/

Como desenvolver Smart Contracts ?

- Curva de aprendizagem não é tão simples
  - Ambientes de programação

- Linguagem Solidity

- Crypto Zombies:

Causas dos Ataques

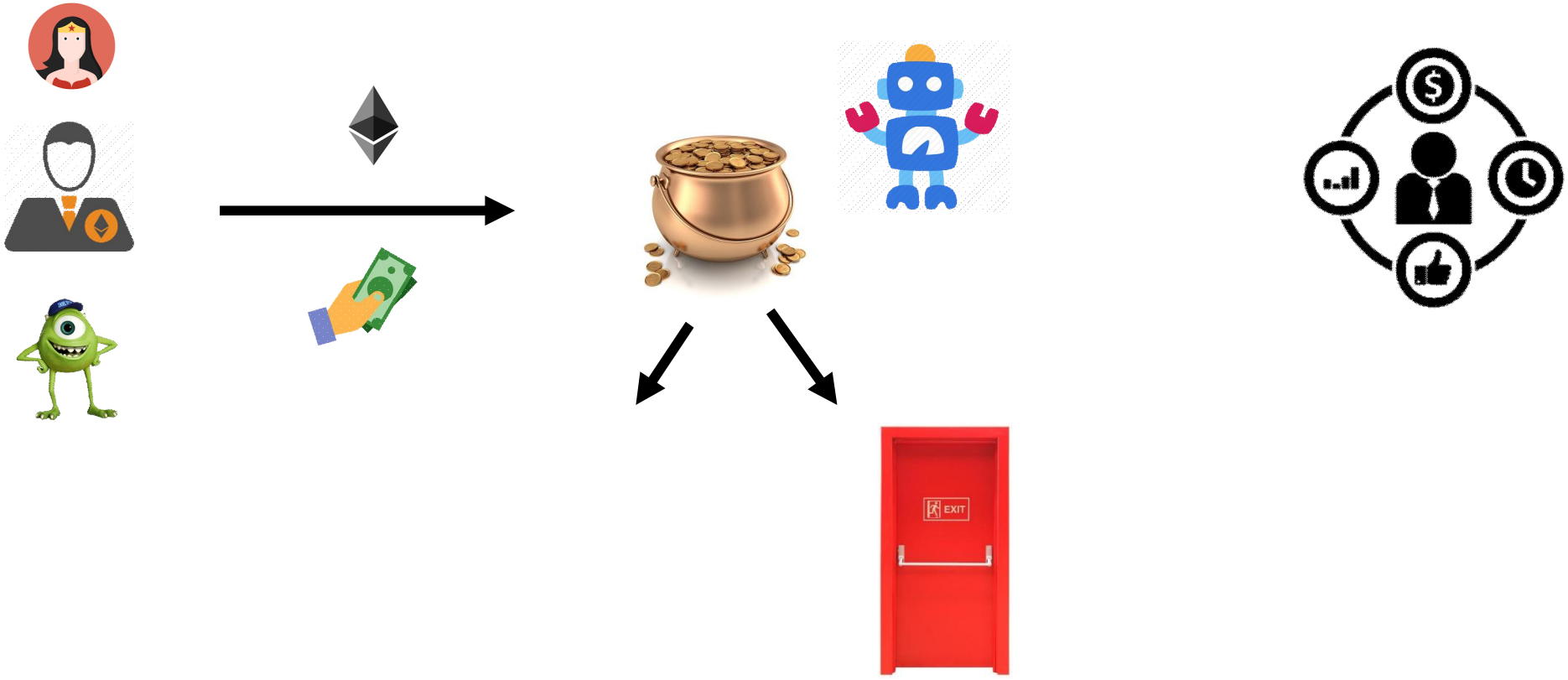
- **O problema de ser popular e público:**
  - Qualquer um pode testar off-line (rede de testes local)
  - Público não significa seguro: contratos maliciosos

- **Técnicas de exploração/ataque:**
  - Reentrância de código
  - Bibliotecas não-confiáveis
  - Ordem de operações
  - Overflow e underflow (transfer)

- **Objetivo principal:**
  - Subverter o funcionamento do contrato

Ataques relevantes: DAO

- **DAO** - Decentralized Autonomous Organization (mai/16):
  - Plataforma de crowdfunding: $150 ! 14% do Ether



  - **Perda de US$ 50 Mi : exploração de erro no smart contract**
  - Resolução controversa (hard-fork) provocou o surgimento do Ethereum Classic (dissidentes)

Ataques relevantes: DAO

- **DAO** - Decentralized Autonomous Organization (mai/16):
  - Plataforma de crowdfunding: $150 ! 14% do Ether
  - Empresa lança proposta (aprovada por um grupo seleto de curadores)
  - Usuários DAO votam. Se atingisse quorum, proposta receberia recursos
  - Investidor poderia sair através de uma "porta de saída" (split): revertia a operação de financiamento em um novo contrato "Child DAO" (recursos bloqueados por 28 dias)

  - Diversos avisos: Manifesto para Moratória
  - Ataque: envia recursos, depois atualiza o saldo
  - Recursos drenados para um Child DAO

Ataques relevantes: DoS Gas

- Transações recebem Gas - Ether:
- Operações/comandos custam Gas
- Transação é rejeitada caso o Gas esgote
  - Previne execuções infinitas do contrato -> possível roubo indevido dos recursos do chamador do contrato
- Mas, contratos requerem mais gas do que operações comuns
- Ex: Leilão de um bem

Lances futuros bloqueados.

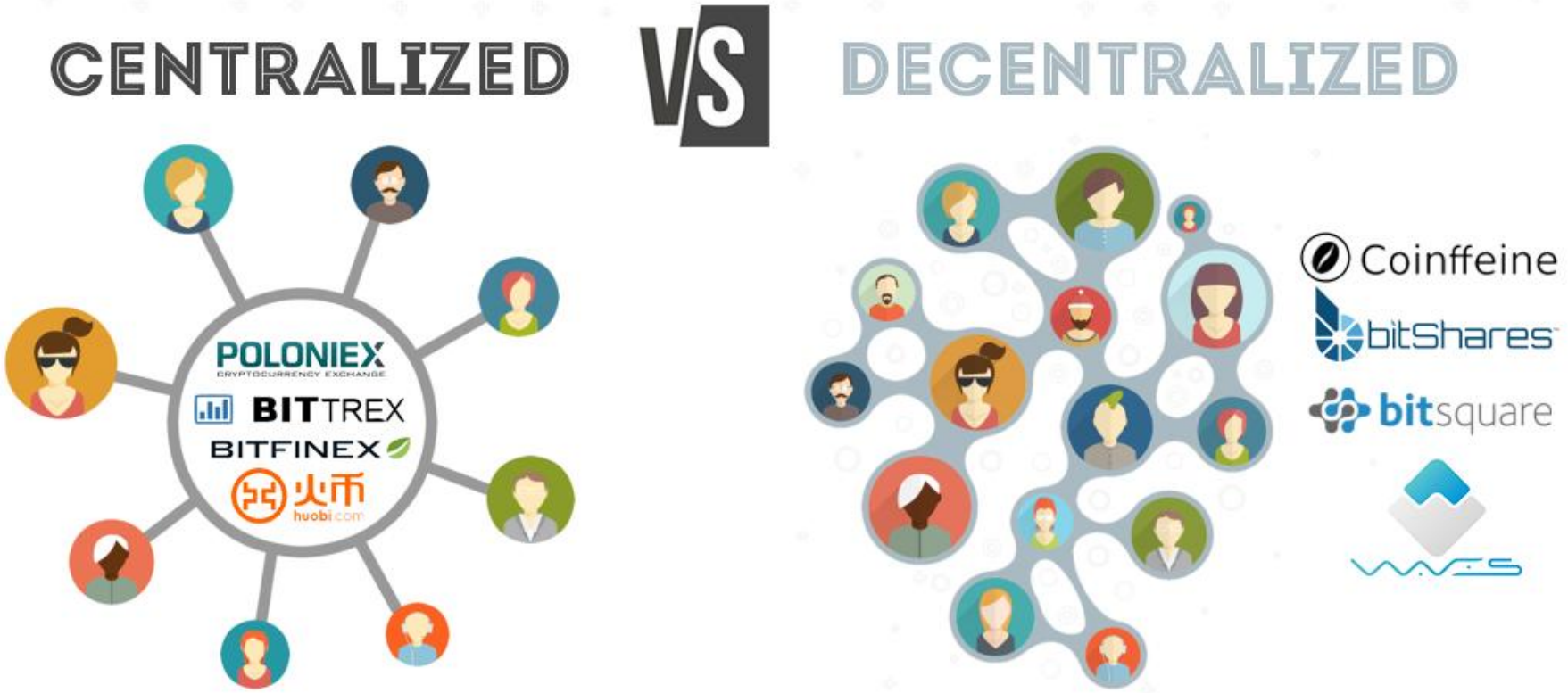
Ataques relevantes: Parity

- **Parity**: whitepaper
  - Contrato usava biblioteca de outro contrato (dono não inicializado) – comum no Ethereum: preço do gas
  - Atacante tornou-se dono do contrato usando a função InitWallet() – funções são públicas por padrão – depois encerrou o contrato
  - Corrida entre black e white hats

```
function() payable {
  // just being sent some cash?
  if (msg.value > 0)
    Deposit(msg.sender, msg.value);
  else if (msg.data.length > 0)
    _walletLibrary.delegatecall(msg.data);
}
```

- **Resultado**: (jul e nov/17): whitepaper
  - 1º ataque: **furto de 32 mi em Ether** . White Hats recuperaram parte do dinheiro. Correção em 20/jul. Porém...
  - **2º ataque conseguiu congelar 500.000 Ether** devido a um erro no smart contract: carteira com multi signature (alerta oficial)
  - Dúvida: erro ou hacker (DevOps199 ?) tornou-se dono do contrato

Corretoras Descentralizadas



Corretoras Descentralizadas

- **Compra e venda de criptoativos:**
  - Volume diário > $10 B
  - Maior parte ocorrem em exchanges centralizadas
    - » Custodia ativos e fecham negócios
  - Alternativa: descentralizar (DEX)
    - » SC controla a operação da exchange (s/ custódia)
    - » Prós: preços eficientes, equidade negocial, fundos livres de roubo pelo operador da exchange, transparência...

- **Problema: lentidão**
  - Efetivação de operação é lenta
  - Atacantes podem antecipar suas ordens pagando mais aos mineradores (front-running)
- **Outros:**
  - Automação contratual:
    - » Apólices de seguros, venda de ativos, crowdfunding, jogos

Limitações, desafios e oportunidades

- **Dificuldades:**
  - Imutabilidade: acertar de primeira...
  - Modelo não convencional: comportamento dos contratos
  - Ciclo de vida: manutenção/atualizações

- **Boas práticas de programação – contratos seguros:**
  - Guia de boas práticas para programação segura
  - Testar contratos
  - Novas linguagens e abstrações (não Turing completa)
  - Usar bibliotecas confiáveis
  - Compromissos: "Blind commitments"

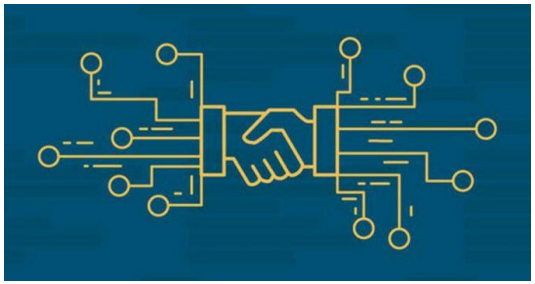
- **Necessidade de auditoria e verificação de código**
  - Consultoria e ferramentas especializadas

- **Comunicação com o mundo exterior:**
  - Oráculos

(In)Segurança em Smart Contracts

Luciano Porto Barreto

lbarreto@cvm.gov.br



maio/2019