

Vulnerabilidades de Contratos Inteligentes

Joyce Quintino Alves

Agenda

- Blockchain
- Contratos Inteligentes
- Vulnerabilidades de Contratos Inteligentes
- Stack Blockchain
- Considerações Finais
- Referências

Agenda

- **Blockchain**
- Contratos Inteligentes
- Vulnerabilidades de Contratos Inteligentes
- Stack Blockchain
- Considerações Finais
- Referências



Blockchain

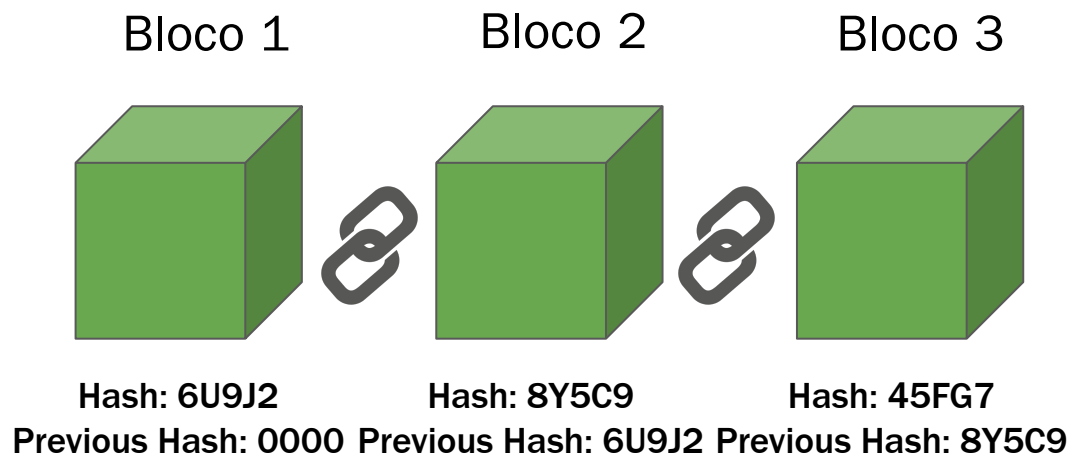
- É uma tecnologia de banco de dados distribuído criada por Satoshi Nakamoto em 2008

- Permite o registro de dados:

- **Descentralizada**
- **Criptografada**
- **Imutável**

- Divide-se em três principais tipos:

- **Pública**
- **Privada**
- **Consórcio**



Nakamoto, S., Bitcoin, A. (2008). A peer-to-peer electronic cash system. Bitcoin.—URL: <https://bitcoin.org/bitcoin.pdf>, v. 4, p. 2.

Agenda

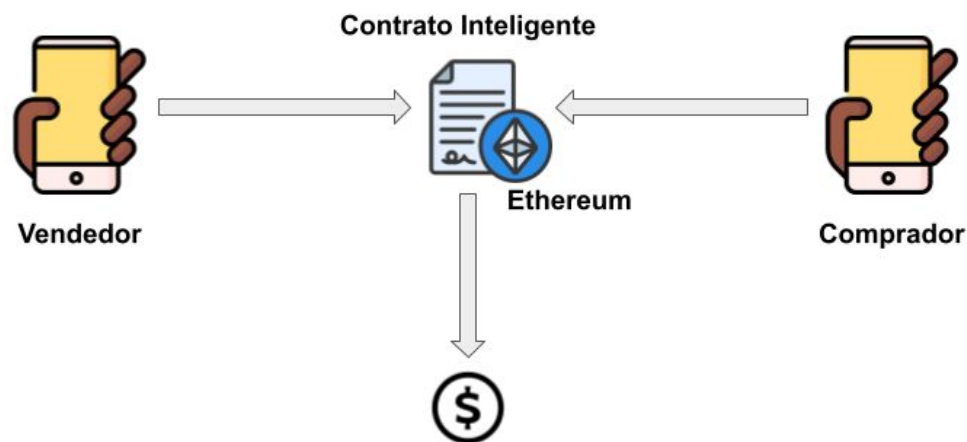
- Blockchain
- **Contratos Inteligentes**
- Vulnerabilidades de Contratos Inteligentes
- Stack Blockchain
- Considerações Finais
- Referências



Contratos Inteligentes

- Características:


- Autoexecução
- Imutabilidade
- Descentralização
- Autenticidade
- Integridade



Abijaude, J., Greve, F., and Sobreira, P. (2021). Blockchain e Contratos Inteligentes para Aplicações em IoT, Uma Abordagem Prática, pages 149–197.

Contratos Inteligentes



```
1 // SPDX-License-Identifier: MIT
2
3 pragma solidity ^0.8.0;
4
5 // Declaração do contrato
6 contract MeuContrato {
7     // Mapeamento para armazenar o saldo de cada conta
8     mapping(address => uint256) public saldo;
9
10    // Evento para registrar transferências
11    event TransferenciaRealizada(address remetente, address destinatario, uint256 valor);
12
13    // Função para transferir tokens
14    function transferir(address destinatario, uint256 valor) public {  infinite gas
15        require(saldo[msg.sender] >= valor, "Saldo insuficiente");
16        saldo[msg.sender] -= valor;
17        saldo[destinatario] += valor;
18        emit TransferenciaRealizada(msg.sender, destinatario, valor);
19    }
20 }
```

Agenda

- Blockchain
- Contratos Inteligentes
- **Vulnerabilidades de Contratos Inteligentes**
- Stack Blockchain
- Considerações Finais
- Referências



Vulnerabilidades de Contratos Inteligentes

Vulnerabilidade	Descrição
Reentrancy	Isto acontece quando um atacante é capaz de chamar repetidamente uma função dentro de um contrato inteligente, explorando o estado do contrato não ter sido atualizado como esperado.
Access Control	Se um contrato inteligente não implementar corretamente o controlo de acesso, pode deixar funções críticas expostas. Isso pode permitir que usuários não autorizados realizem ações que devem ser restritas, como alterar o estado do contrato ou retirar valores.
Denial of Service (DoS)	Nos contratos inteligentes, isto pode ser conseguido consumindo todo o gás disponível ou fazendo com que as transacções falhem continuamente.

OWASP Smart Contract Top 10. <https://owasp.org/www-project-smart-contract-top-10/>

Reentrancy

```
1  // SPDX-License-Identifier: MIT
2  pragma solidity ^0.8.0;
3
4  contract VulneravelAreentrancy {
5      mapping(address => uint) balances;
6
7      function deposit() public payable {    ⚠ infinite gas
8          balances[msg.sender] += msg.value;
9      }
10
11     function withdraw(uint _amount) public {    ⚠ infinite gas
12         require(balances[msg.sender] >= _amount, "Saldo insuficiente");
13
14         // Reduz o saldo do usuário antes de enviar os fundos
15         balances[msg.sender] -= _amount;
16
17         // Envio de fundos
18         (bool success, ) = msg.sender.call{value: _amount}("");
19         require(success, "Falha na transferencia");
20
21         // WARNING: O código abaixo é suscetível a reentrancy
22         // Um ataque de reentrancy pode chamar a função withdraw repetidamente
23         // antes que o saldo seja atualizado
24     }
25 }
```

Access Control

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract VulnerableWithdraw {
5     mapping(address => uint256) public balances;
6
7     function deposit() public payable {    ⚠ infinite gas
8         balances[msg.sender] += msg.value;
9     }
10
11     function withdraw(uint256 amount) public {    ⚠ infinite gas
12         require(balances[msg.sender] >= amount, "Not enough funds");
13         payable(msg.sender).transfer(amount);
14         balances[msg.sender] -= amount;
15     }
16 }
```

Denial of Service (DoS)

```
1  // SPDX-License-Identifier: MIT
2  pragma solidity ^0.8.0;
3
4  //Auction susceptible to DoS attack
5  contract DosAuction {
6      address public currentFrontrunner;
7      uint public currentBid;
8
9      //Takes in bid, refunding the frontrunner if they are outbid
10     function bid() payable public {  ⚠ infinite gas
11         require(msg.value > currentBid, "O valor do lance deve ser maior que o lance atual");
12
13         //If the refund fails, the entire transaction reverts.
14         //Therefore a frontrunner who always fails will win
15         if (currentFrontrunner != address(0)) {
16             //E.g. if recipients fallback function is just revert()
17             require(payable(currentFrontrunner).send(currentBid), "Falha ao reembolsar o lance anterior");
18         }
19
20         currentFrontrunner = msg.sender;
21         currentBid = msg.value;
22     }
23 }
```

Agenda

- Blockchain
- Contratos Inteligentes
- Vulnerabilidades de Contratos Inteligentes
- **Stack Blockchain**
- Considerações Finais
- Referências



Stack Blockchain



Agenda

- Blockchain
- Contratos Inteligentes
- Vulnerabilidades de Contratos Inteligentes
- Stack Blockchain
- **Considerações Finais**
- Referências



Considerações Finais

- Estratégias de mitigação:
 - Boas práticas de desenvolvimento, por exemplo, utilização de modificadores de acesso (private, internal ou external) para controlar quem pode chamar as funções do contrato
 - Utilizar ferramentas de análise de código (**Análise Estática e Dinâmica**) para identificar vulnerabilidades antes da implantação do contrato na Blockchain



SLITHER



MythX

Agenda

- Blockchain
- Contratos Inteligentes
- Vulnerabilidades de Contratos Inteligentes
- Stack Blockchain
- Considerações Finais
- **Referências**



Referências

- A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari and M. Ayyash. (2015). Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications, in *IEEE Communications Surveys & Tutorials*, vol. 17, no. 4, pp. 2347-2376, Fourthquarter.
- Abijaude, J., Greve, F., and Sobreira, P. (2021). Blockchain e Contratos Inteligentes para Aplicações em IoT, Uma Abordagem Prática, pages 149–197.
- Atzori, L., Iera, A., Morabito, G. (2010). The Internet of Things: A survey. *Computer Networks*, Elsevier, v. 54, n. 15, p. 2787-2805, out.
- Fan, Y., Li, Y., Zhan, M., Cui, H., and Zhang, Y. (2020). Iotdefender: A federated transfer learning intrusion detection framework for 5g iot. In *2020 IEEE 14th International Conference on Big Data Science and Engineering (BigDataSE)*, pages 88–95.
- Hasan, Q. O. M. (2023). Machine Learning Based Framework for Smart Contract Vulnerability Detection in Ethereum Blockchain. PhD thesis, Rochester Institute of Technology.
- He, D., Wu, R., Li, X., Chan, S., and Guizani, M. (2023). Detection of vulnerabilities of blockchain smart contracts. *IEEE Internet of Things Journal*, 10(14):12178–12185.
- Li, J. (2023). Metamorphic testing for smart contract vulnerabilities detection.
- Mei, X., Ashraf, I., Jiang, B., and Chan, W. (2019). A fuzz testing service for assuring smart contracts. In *2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, pages 544–545.
- Minerva, R., Biru, A., and Rotondi, D. (2015). Towards a definition of the internet of things (iot). *IEEE Internet Initiative*, v. 1, 1–86.
- Nakamoto, S., Bitcoin, A. (2008). A peer-to-peer electronic cash system. Bitcoin.–URL: <https://bitcoin.org/bitcoin.pdf>, v. 4, p. 2.
- Patel, K. K.; Patel, S. M; Scholar, P. G. (2016). Internet of things-IOT: definition, characteristics, architecture, enabling technologies, application & future challenges. *International journal of engineering science and computing*, v. 6, n. 5.
- Stallings, W. (2015). *Criptografia e segurança de redes: princípios e práticas*. Pearson.
- Waheed, N., He, X., Ikram, M., Usman, M., Hashmi, S. S., and Usman, M. (2020). Security and privacy in iot using machine learning and blockchain: Threats and countermeasures. *ACM Comput. Surv.*, 53(6).



Mestrado e Doutorado em Ciência da Computação



GREat

Grupo de Redes de Computadores
Engenharia de Software
e Sistemas



**UNIVERSIDADE
FEDERAL DO CEARÁ**

Obrigada!