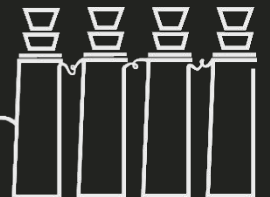
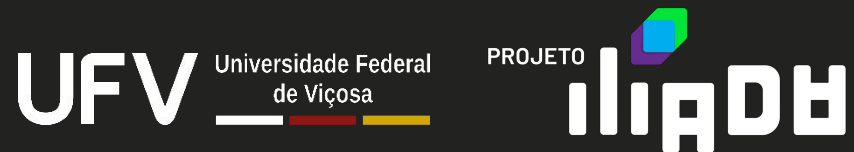


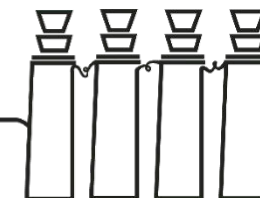
GT-SmartSeg: Grupo de Trabalho em Segurança de Contratos Inteligentes

Josué N. Campos, Luís H. S. de Carvalho, Isdael R. Oliveira,
Aline C. S. Silva, Iago G. Falcão, Matheus J. da Silva
Coordenador: José Augusto Miranda Nacif



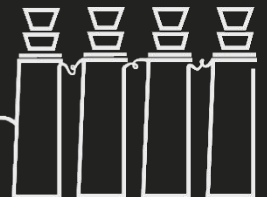
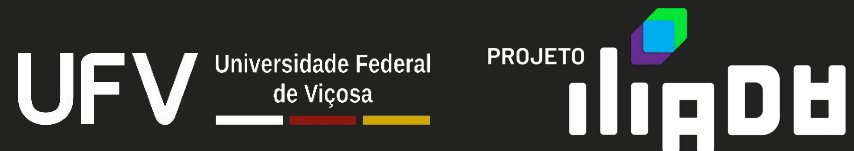
Agenda

- Cenário da segurança em contratos inteligentes
- Ferramentas de detecção automática de vulnerabilidades
- Objetivos do GT
- Resultados preliminares
- Próximos passos
- Sobre o GT-SmartSeg
- Agradecimentos



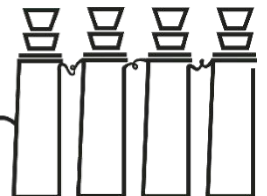
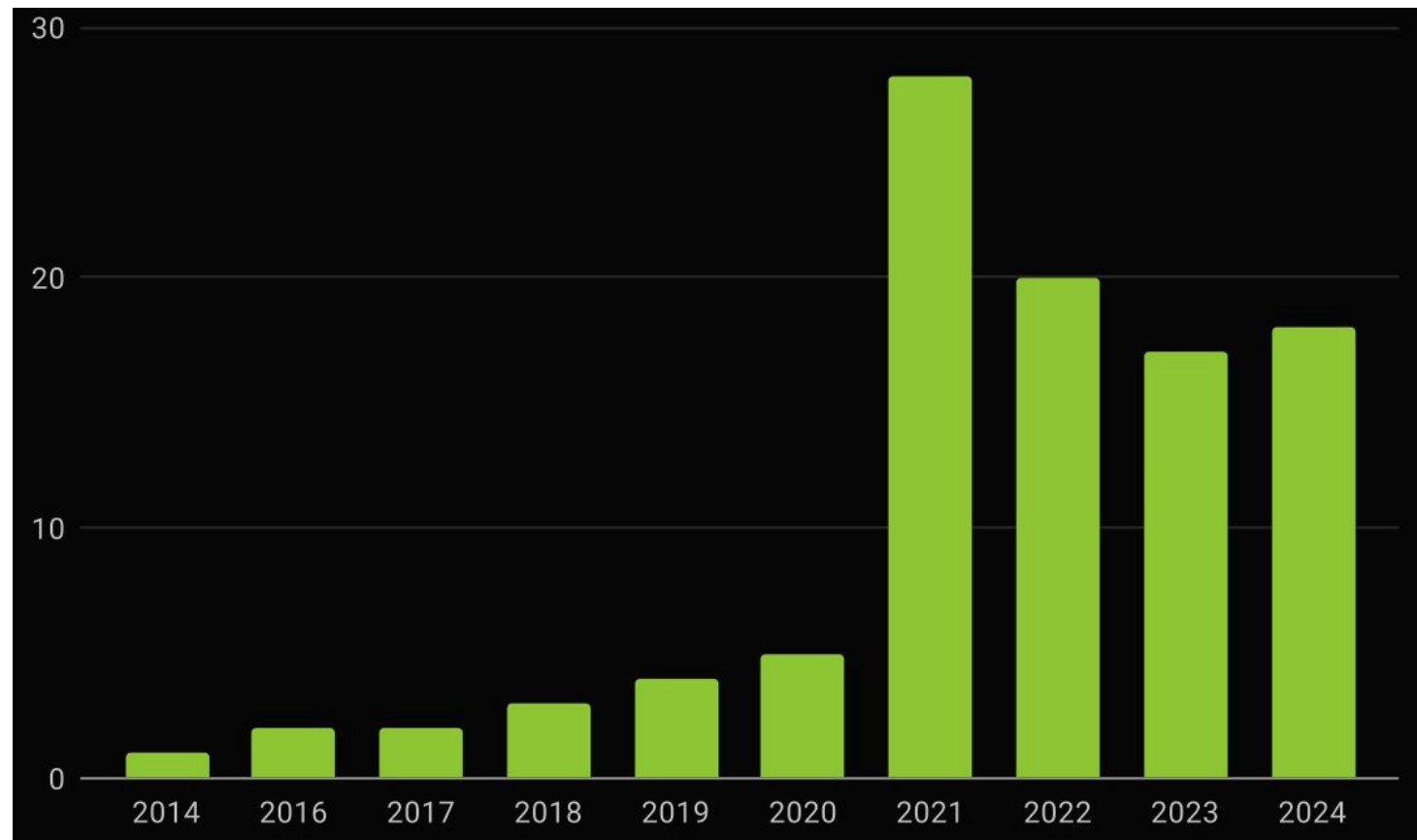
Cenário da segurança em contratos inteligentes

As auditorias de contratos inteligentes são importantes, mas enfrentam grandes desafios...



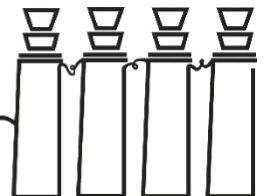
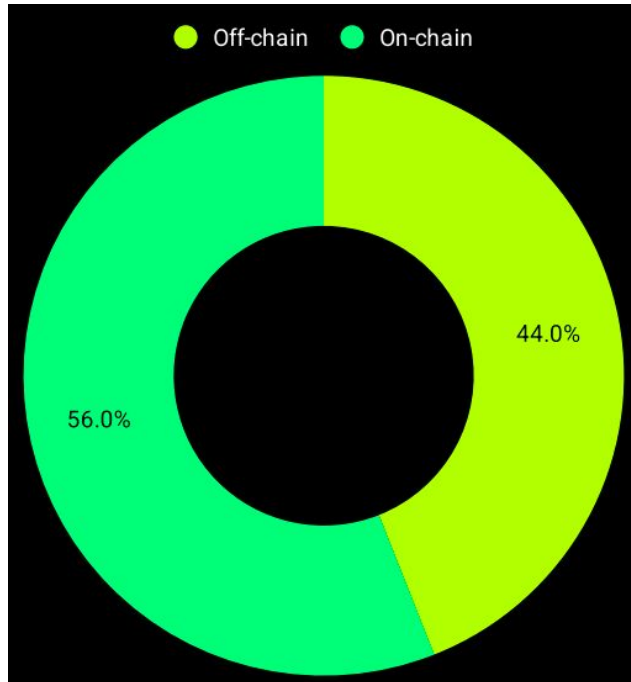
Ataques em Finanças Descentralizadas

Ataques por ano



Ataques em Finanças Descentralizadas

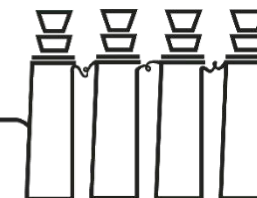
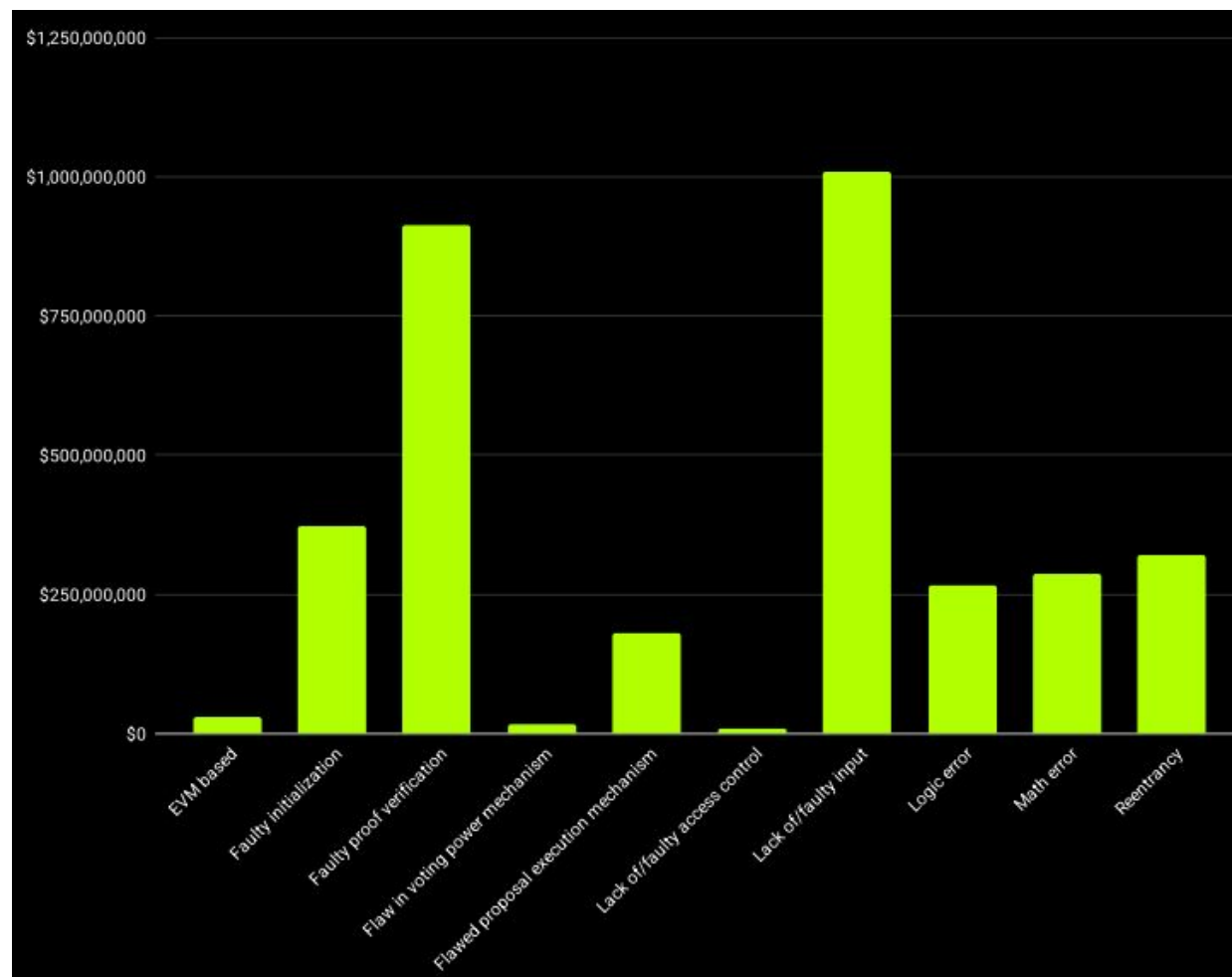
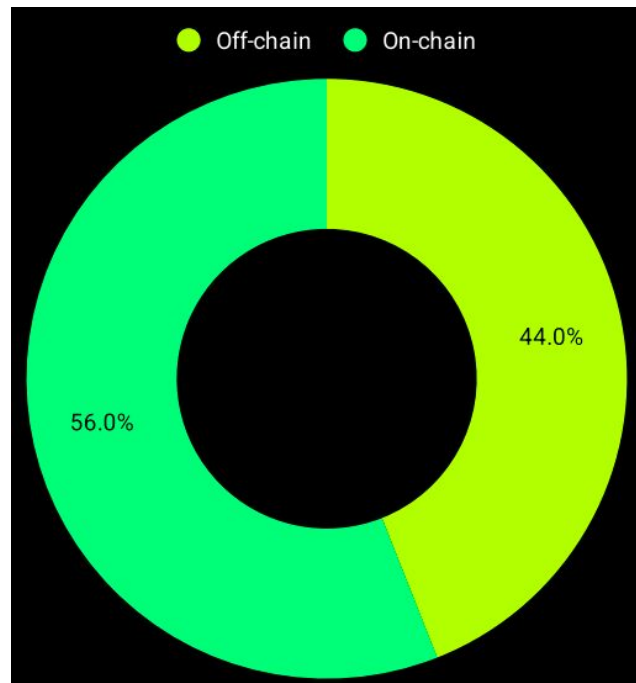
2014-2024



Ataques em Finanças Descentralizadas

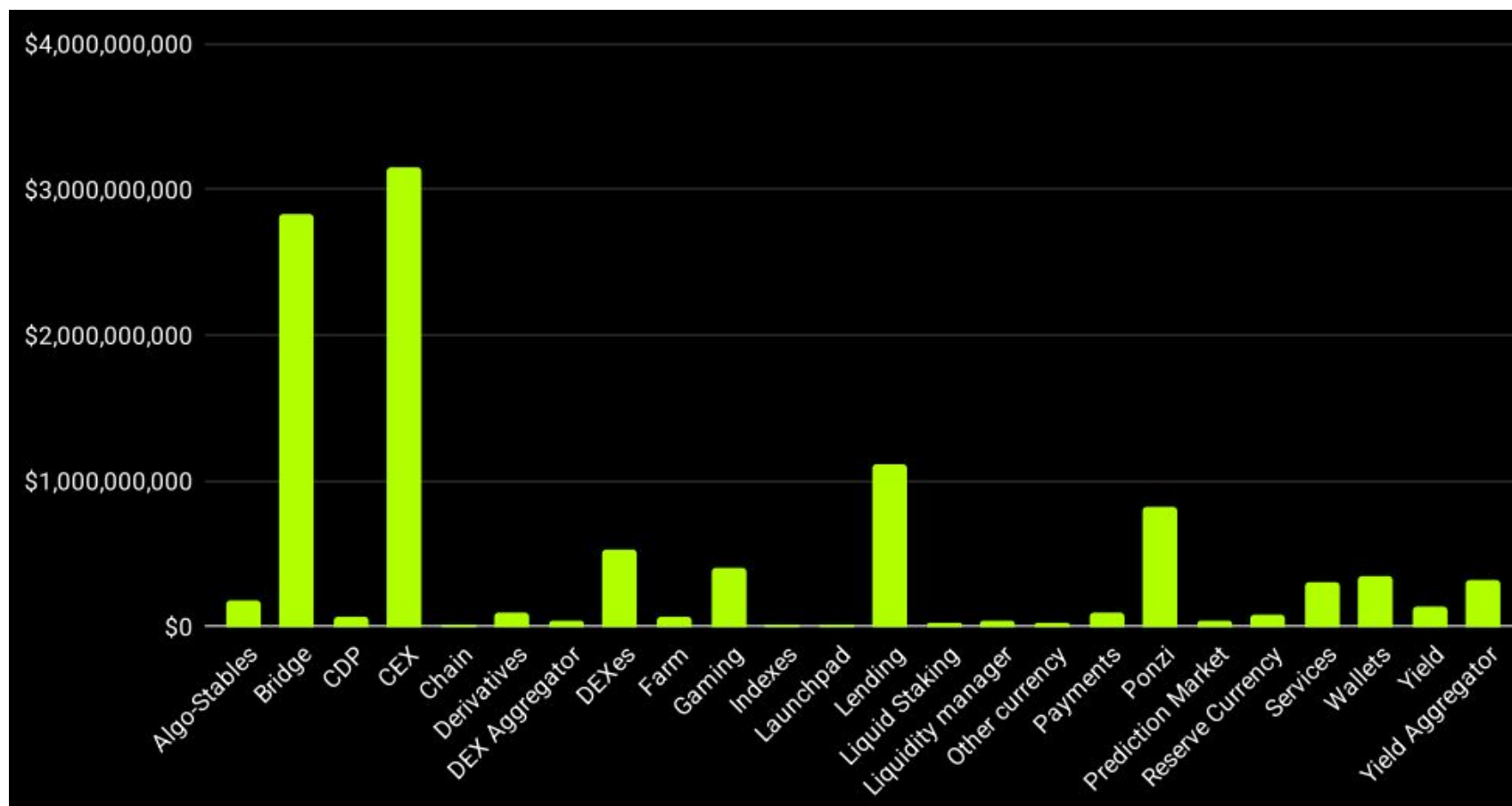
Impacto financeiro por tipo de ataque

2014-2024



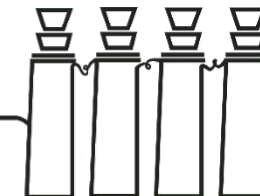
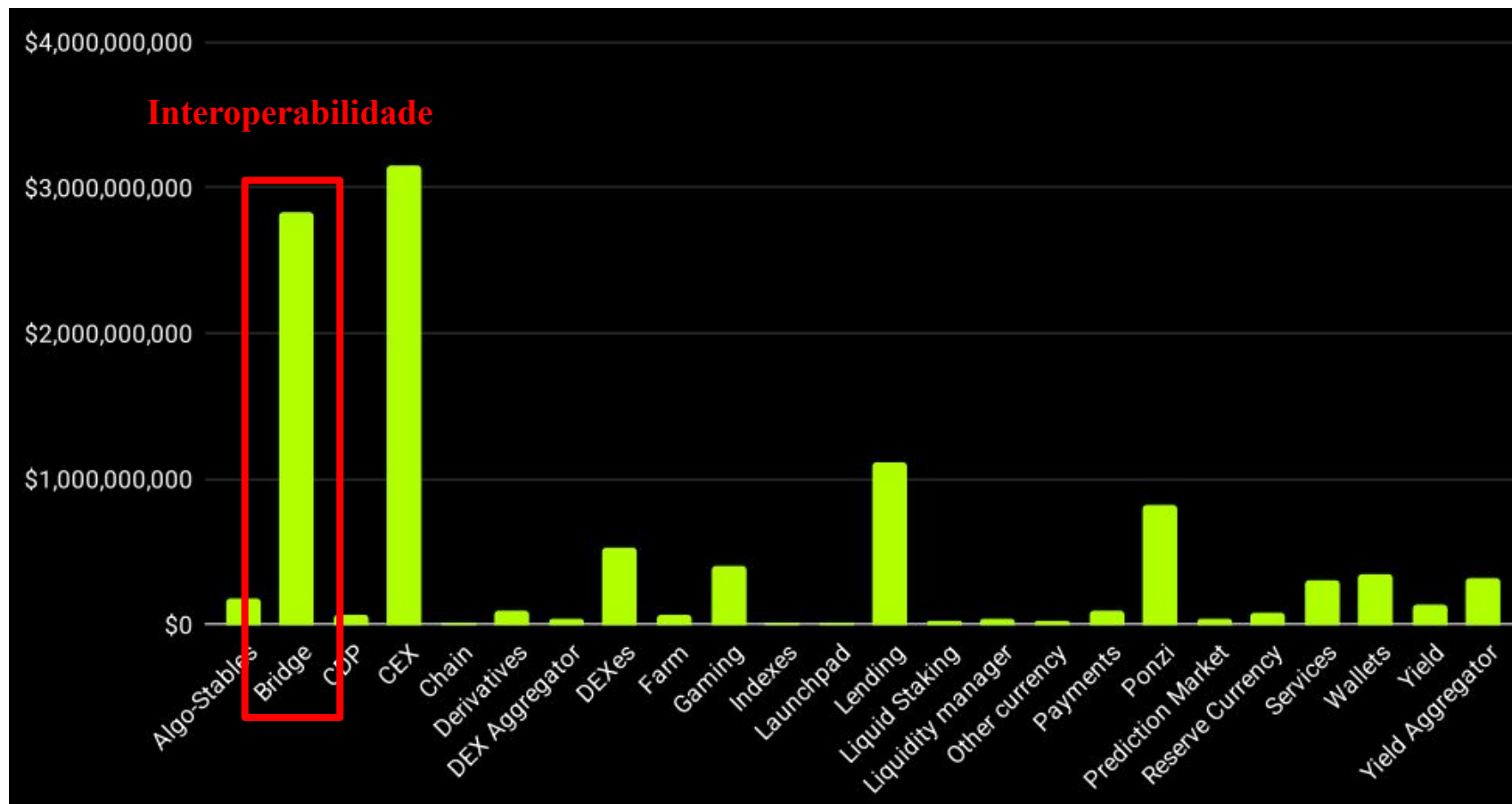
Ataques em Finanças Descentralizadas

Impacto financeiro por alvo do ataque



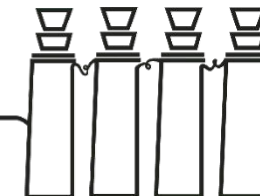
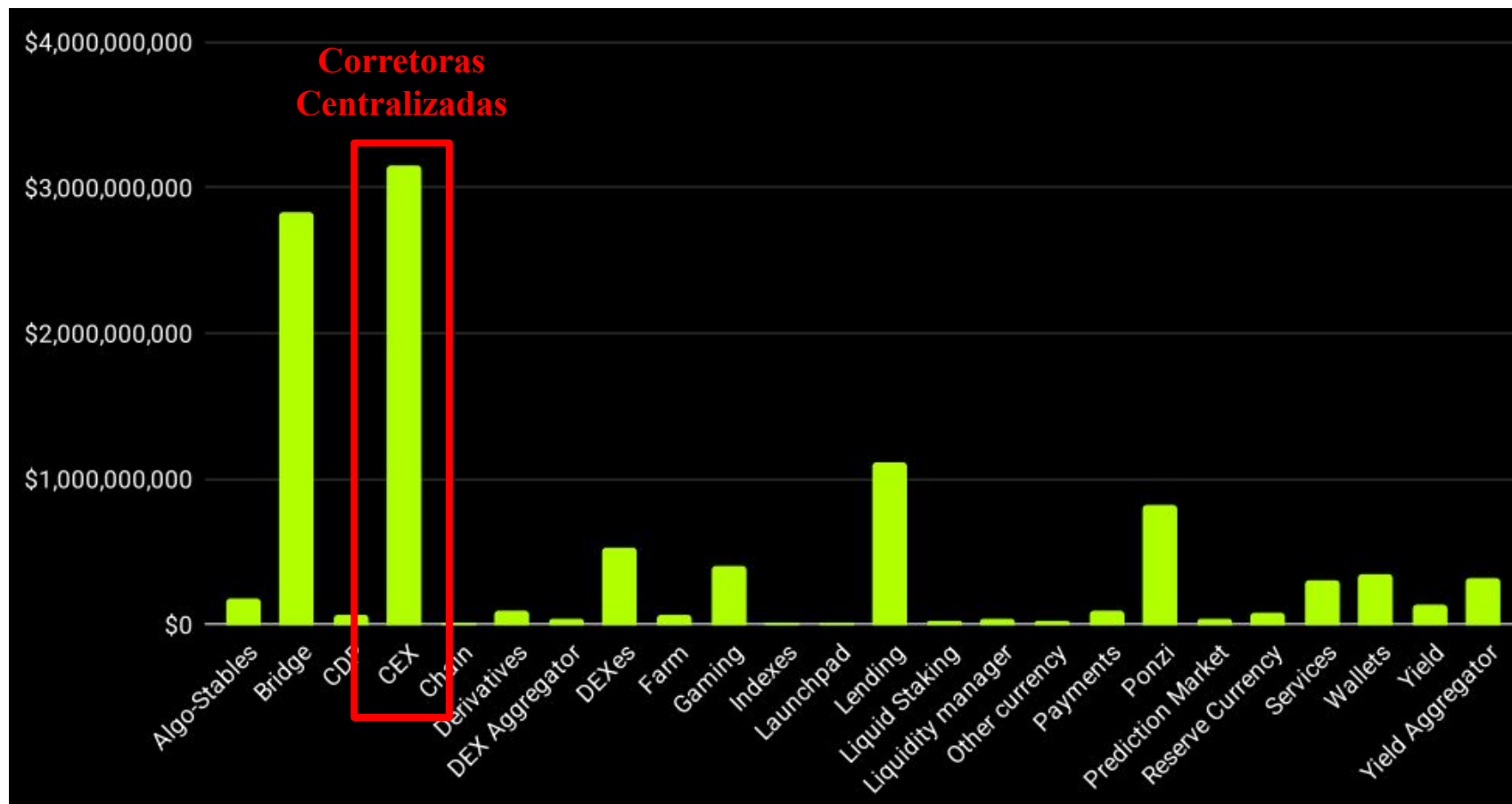
Ataques em Finanças Descentralizadas

Impacto financeiro por alvo do ataque

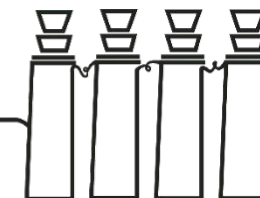
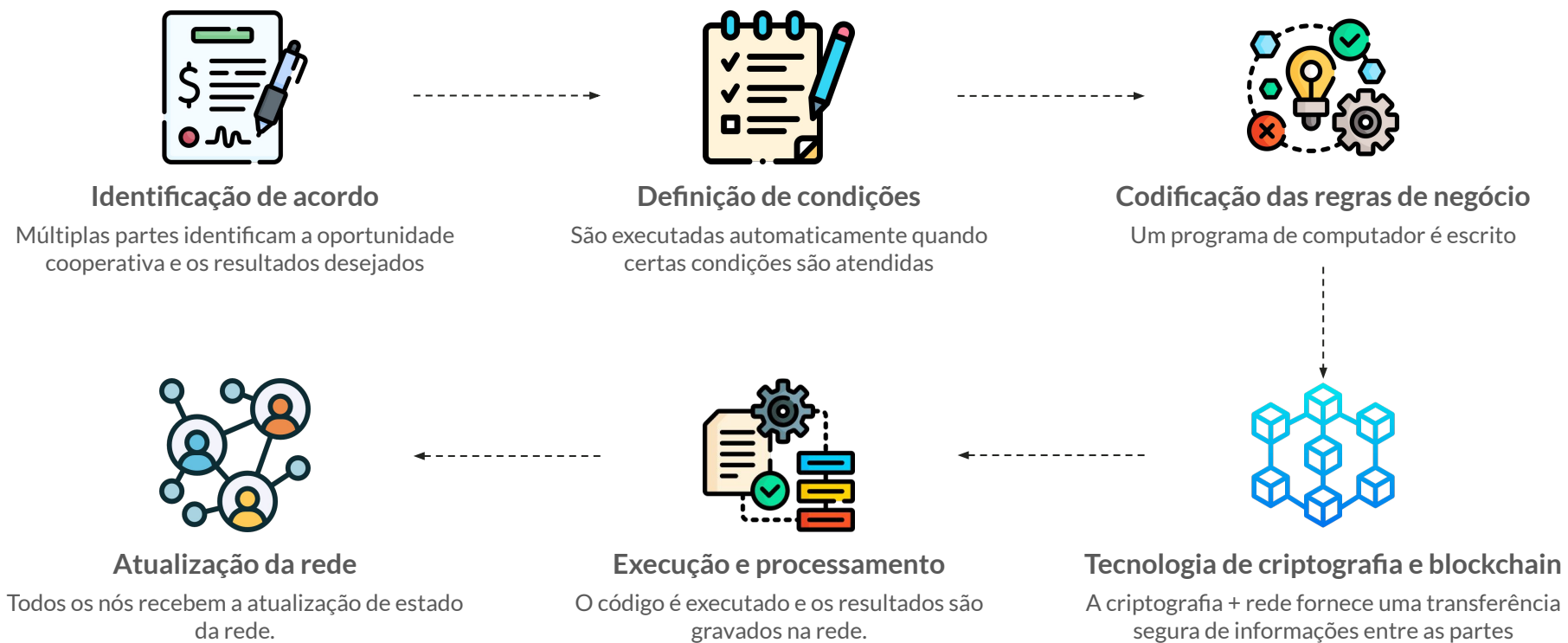


Ataques em Finanças Descentralizadas

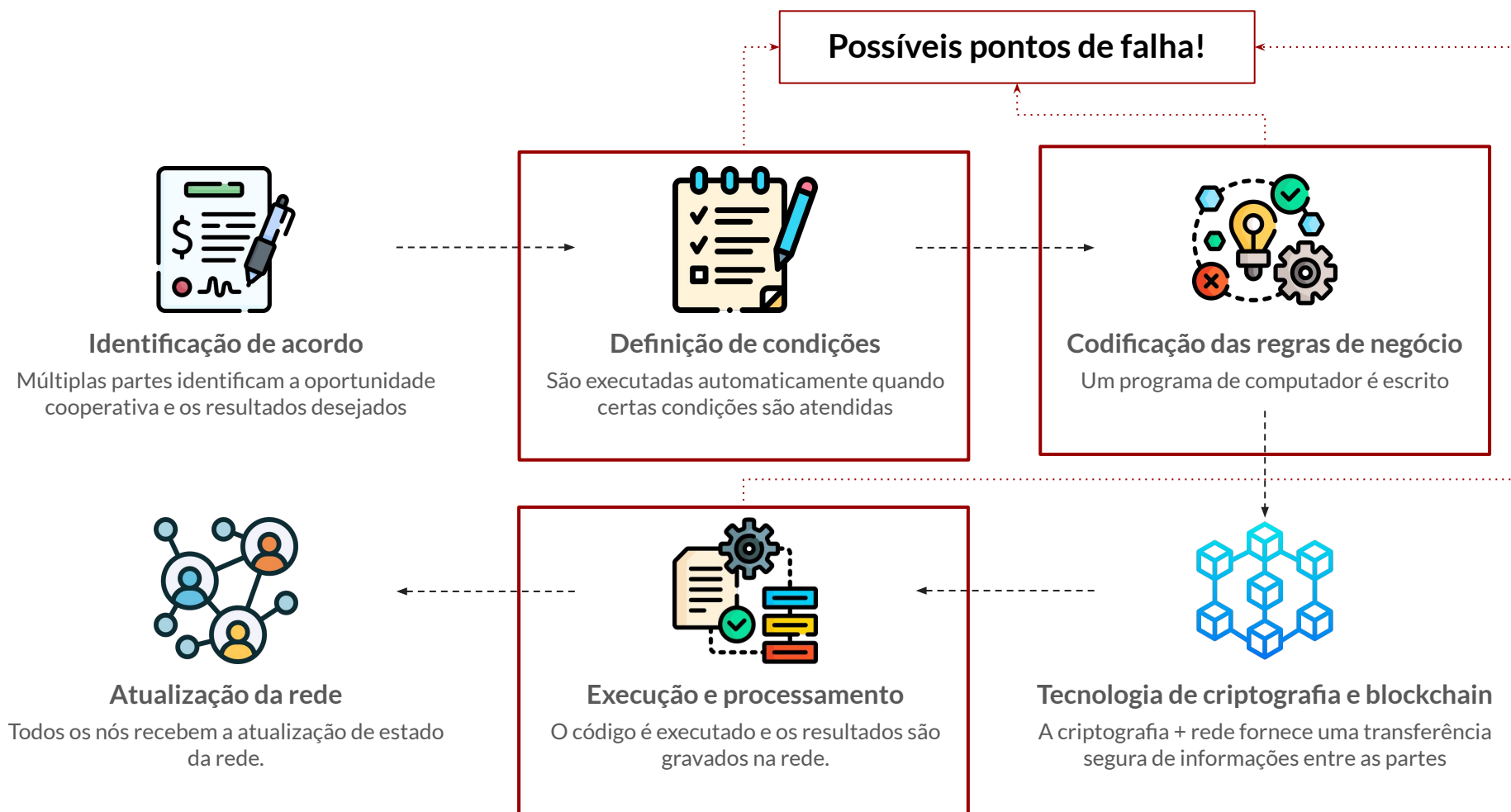
Impacto financeiro por alvo do ataque



Contratos Inteligentes

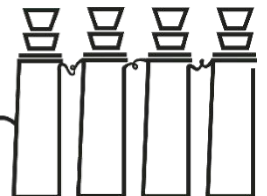


Contratos Inteligentes



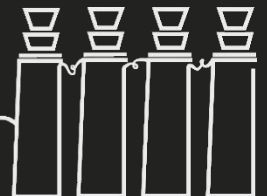
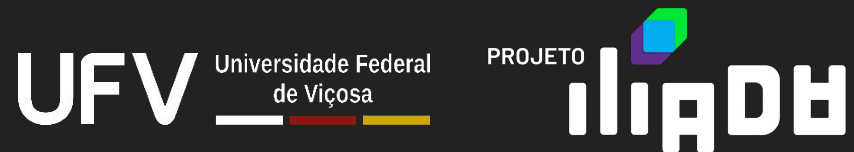
Segurança em Contratos Inteligentes

- A auditoria de segurança é uma das principais etapas no ciclo de desenvolvimento de contratos inteligentes
- A identificação de vulnerabilidades é feita **majoritariamente** por métodos de análise manuais
- Etapa **comumente** desvinculada da etapa de codificação do contrato inteligente
- **Desafios existentes:**
 - **Gargalos manuais:** A elaboração manual de relatórios pode ser lenta, cara e propensa à inconsistência, atrasando o tempo de colocação no mercado
 - **Vulnerabilidades escondidas:** Ataques novos e complexos podem escapar dos métodos tradicionais de auditoria



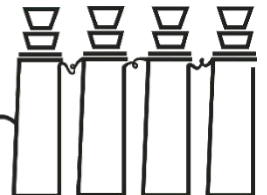
Ferramentas de detecção automática de vulnerabilidades

Mecanismos inovadores para auxiliar o processo de auditoria de contratos inteligentes...



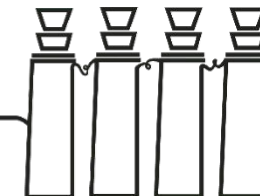
Ferramentas de Auditoria

- Mecanismos que empregam técnicas de análise de software para identificação de falhas
 - **Análise estática:** Verifica o código do contrato sem propriamente executá-lo
 - **Análise dinâmica:** Avalia o código do contrato utilizando informações de execução do ambiente utilizado
- Mostram-se como promissoras na literatura para reduzir o tempo das auditorias convencionais e possibilitar integrações da etapa de verificação do contrato durante a fase de codificação
- **Desafios existentes:**
 - **Fragmentação:** Ferramentas atuais são insuficientes e alguns bugs não podem ser detectados automaticamente
 - **Suporte:** Muitas acabam sendo descontinuadas ou passam a ser incorporadas em empresas de auditoria e deixam de receber suporte em código-aberto



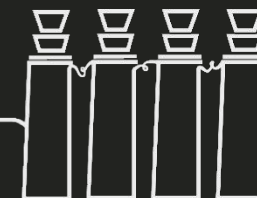
Ferramentas de Auditoria

- Ferramentas de detecção automática de vulnerabilidades disponíveis publicamente e que possuem suporte às versões atuais da linguagem Solidity:
 - **Slither:** Emprega análise estática por meio da conversão do código para uma Representação Intermediária (IR) e uso da técnica de execução simbólica
 - **Mythril:** Combina análise estática e dinâmica com a execução simbólica e resolução de teorema para explorar os estados possíveis de execução do contrato
 - **ConFuzzius:** Utiliza análise dinâmica com a técnica de Fuzzing evolutivo, testando o código com entradas aleatórias para identificar comportamentos inesperados



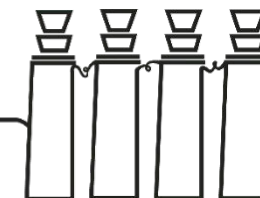
Objetivos e resultados preliminares

Nosso foco e contribuições científicas obtidas até o momento...



Objetivos

- **Desenvolver** um framework que facilite a execução automatizada de ferramentas de verificação em contratos inteligentes escritos na linguagem de programação Solidity
- **Avaliar** a capacidade de detecção de vulnerabilidades das principais ferramentas de código-aberto que analisam contratos inteligentes em Solidity utilizando um dataset de contratos inteligentes auditados manualmente como validação
- **Destacar** o estado-da-arte da análise automática de segurança de contratos inteligentes, identificar áreas de melhorias, potencializar o desenvolvimento de pesquisas futuras e novas ferramentas
- **Obs.:** Não queremos substituir a auditoria convencional, mas incorporar o ciclo de desenvolvimento do contrato inteligente com o ciclo de verificação de vulnerabilidades de segurança e auxiliar o desenvolvedor com práticas de códigos seguros

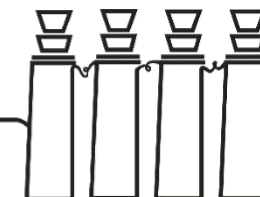


Resultados Preliminares

- Artigos científicos publicados e em revisão:

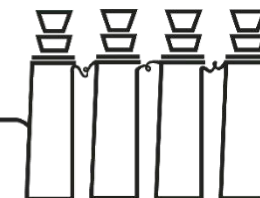


- “Análise das Ferramentas de Detecção de Vulnerabilidades para Contratos Inteligentes de Blockchains EVM” - publicado no VIII Workshop Blockchain: Teoria, Tecnologia e Aplicações (WBlockchain 2025) e ganhador do prêmio de *Best Paper* do Workshop
- “SCAO: Framework para Orquestração de Ferramentas de Auditoria para Contratos Inteligentes Solidity” - Submetido ao I Workshop Brasileiro de Sistemas Web3 (BrWeb3 2025)



Padrão de Vulnerabilidades

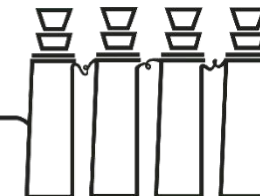
- Uma vulnerabilidade representa uma falha no contrato inteligente que pode ser explorada por um agente malicioso
- Solução não descontinuada de nomenclatura adotada neste trabalho:
 - Padrão OWASP (*Open Web Application Security Project*)
 - **Foco:** Ranque OWASP *Smart Contract Security Top 10* para 2025
- Analisamos como as ferramentas de detecção automática de vulnerabilidades cobrem os 10 grupos de vulnerabilidades mais relevantes para 2025, de acordo com a OWASP



Top-10 OWASP 2025

Código	Vulnerabilidade	Descrição
SC01	Controle de Acesso	Permite que usuários não autorizados modifiquem dados ou funções devido à falta de verificações de permissão.
SC02	Manipulação de Preço de Oráculo	Exploração de oráculos de preço para alterar a lógica do contrato, resultando em perdas financeiras.
SC03	Erros de Lógica	Erros na lógica do contrato que levam a comportamento inesperado, como distribuição incorreta de recompensas.
SC04	Falta de Validação de Entrada	Falta de validação de entrada permite que atacantes manipulem a execução do contrato.
SC05	Ataques de Reentrada	Permite múltiplas execuções de uma função antes de sua conclusão, podendo drenar fundos do contrato.
SC06	Chamadas Externas Não Verificadas	Falha ao verificar chamadas externas pode levar a execução incorreta do contrato.
SC07	Ataques de Empréstimos Rápidos	Uso de empréstimos rápidos para manipular protocolos, drenando liquidez ou alterando preços.
SC08	Overflow e Underflow de Inteiros	Erros aritméticos que podem levar a cálculos incorretos ou roubo de <i>tokens</i> .
SC09	Aleatoriedade Insegura	Falta de aleatoriedade segura pode permitir previsibilidade em sorteios e distribuições de <i>tokens</i> .
SC10	Ataques de Negação de Serviço (DoS)	Exploração de consumo excessivo de recursos para tornar o contrato inoperante.

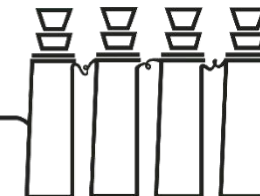
Impacto financeiro por alvo do ataque



Top-10 OWASP 2025

Código	Vulnerabilidade	Descrição
SC01	Controle de Acesso	Permite que usuários não autorizados modifiquem dados ou funções devido à falta de verificações de permissão.
SC02	Manipulação de Preço de Oráculo	Exploração de oráculos de preço para alterar a lógica do contrato, resultando em perdas financeiras.
SC03	Erros de Lógica	Erros na lógica do contrato que levam a comportamento inesperado, como distribuição incorreta de recompensas.
SC04	Falta de Validação de Entrada	Falta de validação de entrada permite que atacantes manipulem a execução do contrato.
SC05	Ataques de Reentrada	Permite múltiplas execuções de uma função antes de sua conclusão, podendo drenar fundos do contrato.
SC06	Chamadas Externas Não Verificadas	Falha ao verificar chamadas externas pode levar a execução incorreta do contrato.
SC07	Ataques de Empréstimos Rápidos	Uso de empréstimos rápidos para manipular protocolos, drenando liquidez ou alterando preços.
SC08	Overflow e Underflow de Inteiros	Erros aritméticos que podem levar a cálculos incorretos ou roubo de <i>tokens</i> .
SC09	Aleatoriedade Insegura	Falta de aleatoriedade segura pode permitir previsibilidade em sorteios e distribuições de <i>tokens</i> .
SC10	Ataques de Negação de Serviço (DoS)	Exploração de consumo excessivo de recursos para tornar o contrato inoperante.

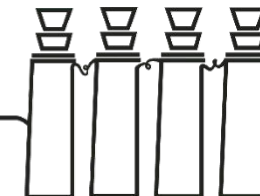
Vulnerabilidades relacionadas diretamente à codificação do contrato inteligente



Top-10 OWASP 2025

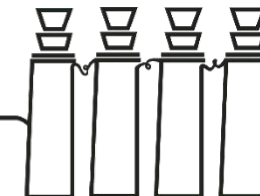
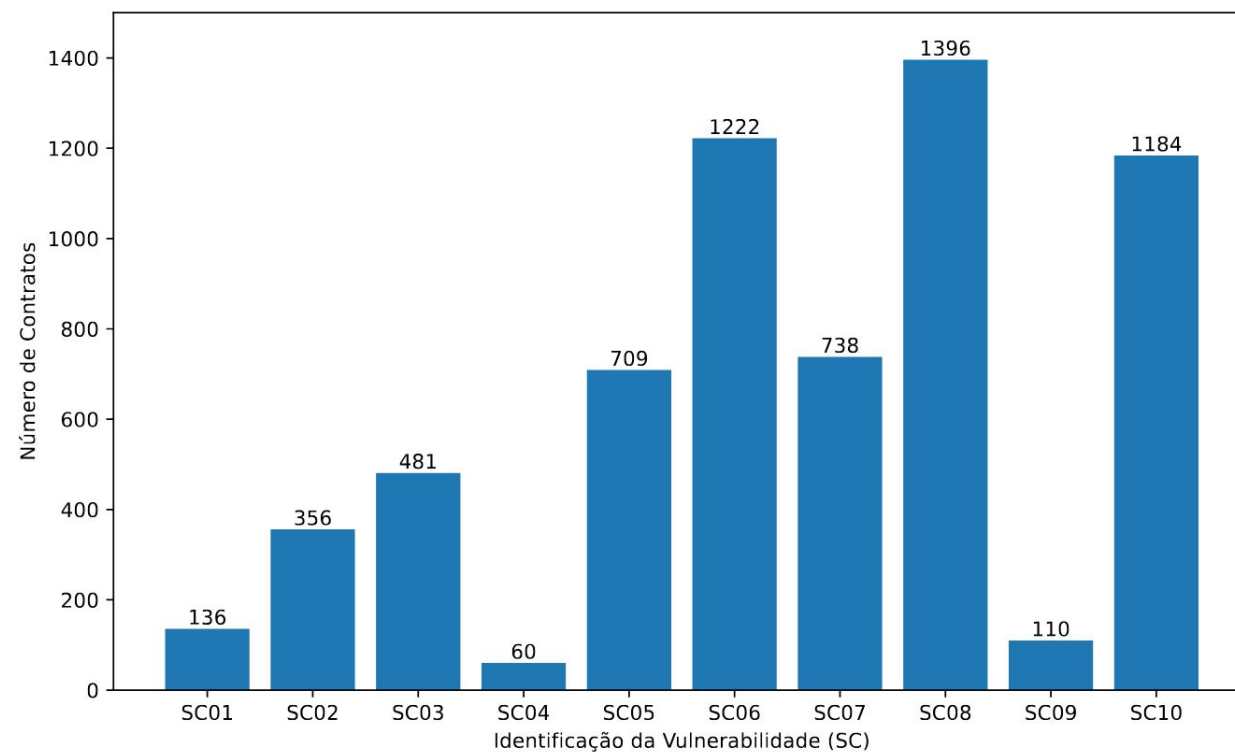
Código	Vulnerabilidade	Descrição
SC01	Controle de Acesso	Permite que usuários não autorizados modifiquem dados ou funções devido à falta de verificações de permissão.
SC02	Manipulação de Preço de Oráculo	Exploração de oráculos de preço para alterar a lógica do contrato, resultando em perdas financeiras.
SC03	Erros de Lógica	Erros na lógica do contrato que levam a comportamento inesperado, como distribuição incorreta de recompensas.
SC04	Falta de Validação de Entrada	Falta de validação de entrada permite que atacantes manipulem a execução do contrato.
SC05	Ataques de Reentrada	Permite múltiplas execuções de uma função antes de sua conclusão, podendo drenar fundos do contrato.
SC06	Chamadas Externas Não Verificadas	Falha ao verificar chamadas externas pode levar a execução incorreta do contrato.
SC07	Ataques de Empréstimos Rápidos	Uso de empréstimos rápidos para manipular protocolos, drenando liquidez ou alterando preços.
SC08	Overflow e Underflow de Inteiros	Erros aritméticos que podem levar a cálculos incorretos ou roubo de <i>tokens</i> .
SC09	Aleatoriedade Insegura	Falta de aleatoriedade segura pode permitir previsibilidade em sorteios e distribuições de <i>tokens</i> .
SC10	Ataques de Negação de Serviço (DoS)	Exploração de consumo excessivo de recursos para tornar o contrato inoperante.

Vulnerabilidades relacionadas à interação do contrato com a blockchain ou mundo externo

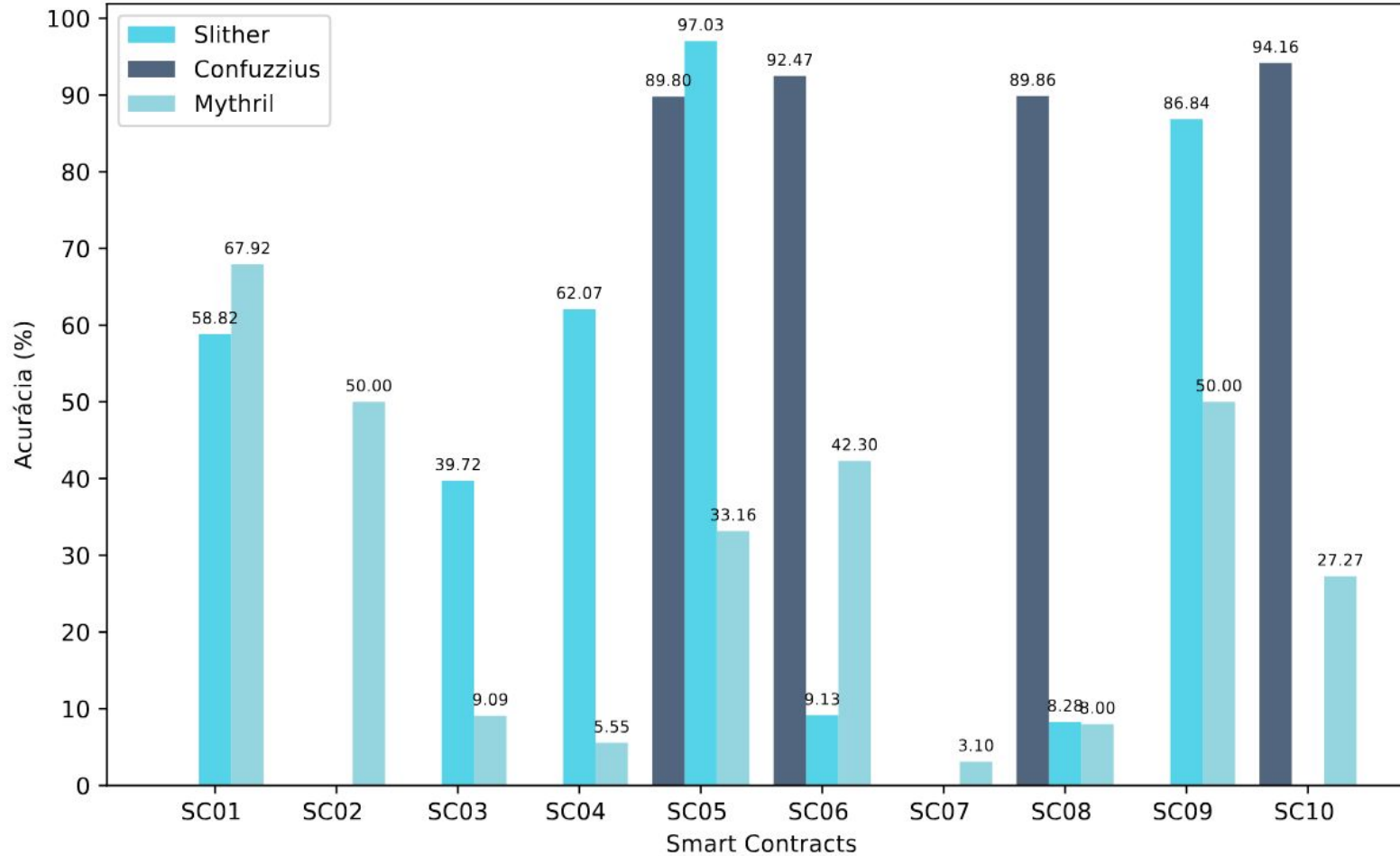


Conjunto de Dados

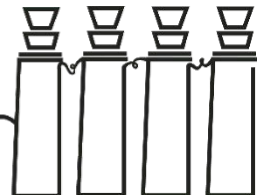
- [Di Angelo e Salzer *et al.* 2023] *Consolidation of ground truth sets for weakness detection in smart contracts*
- Seleção apenas das vulnerabilidades relacionadas ao ranque da OWASP
- Total de 6.072 contratos auditados manualmente



Resultados



SC	ConFuzzius	Slither	Mythril
01	-	1,52s	34,49h
02	-	-	5,52h
03	-	3,07s	7,45h
04	-	4,09s	5,35h
05	1,89s	2,44s	40,48h
06	1,9s	1,96s	7,58h
07	-	-	3,39h
08	1,94s	2,09s	9,18h
09	-	2,93s	8,24h
10	1,99s	-	8,33h

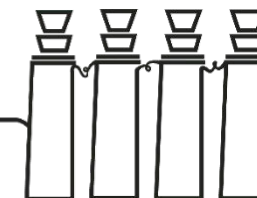
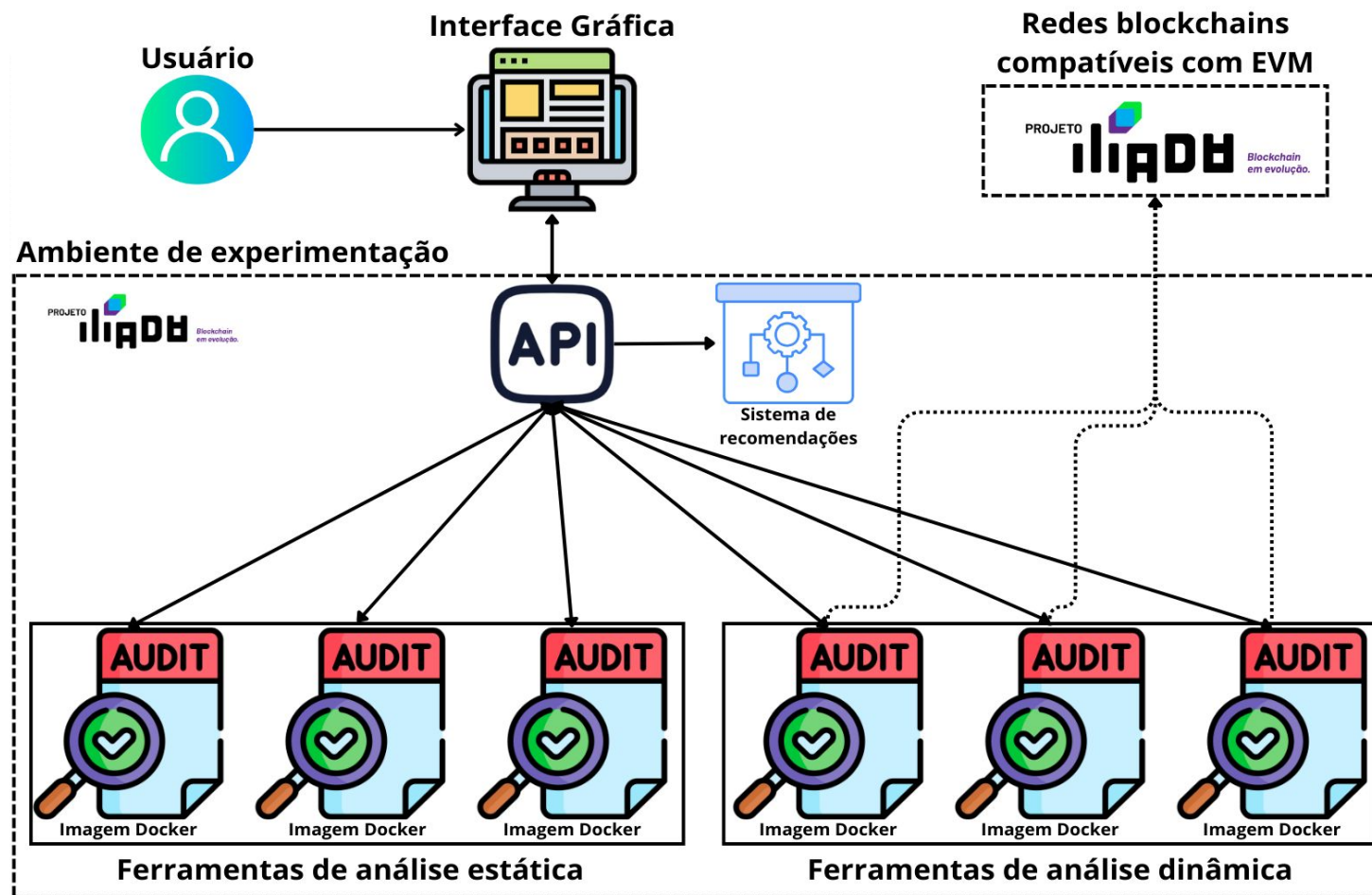


Contribuições

- Constatamos que as ferramentas possuem uma **alta variabilidade em identificar as vulnerabilidades de segurança, atingindo uma faixa de 0 a 97%**
- Observamos que os conjuntos de dados de contratos inteligentes vulneráveis da literatura possuem versões antigas da linguagem Solidity em sua maioria, com **70% dos contratos descontinuados pelas ferramentas**
- Nossas análises mostram que, enquanto a ferramenta **ConFuzzius** apresenta **altas taxas de detecção**, possui **suporte para poucas vulnerabilidades**. Já a **Mythril** consegue cobrir uma **maior quantidade de vulnerabilidades**, porém **não possui escalabilidade**. Por fim, a ferramenta **Slither** demonstra como uma **solução balanceada dentre as três**, mas possui **60% de taxa de falsos negativos**
- Direcionamos pesquisas futuras a partir da análise da acurácia das ferramentas e constatando que **novas técnicas e ferramentas precisam ser desenvolvidas** para cobrir todas as vulnerabilidades com **confiabilidade, eficiência e usabilidade**




Framework GT-SmartSeg




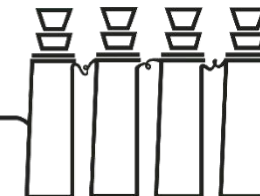
Framework Proposto

GT-SmartSeg

```
1 // SPDX-License-Identifier: MIT
2 pragma solidity 0.8.29;
3
4 contract EtherStore {
5     mapping(address => uint256) public balances;
6
7     function deposit() public payable {
8         balances[msg.sender] += msg.value;
9     }
10
11    function withdraw() public {
12        uint256 bal = balances[msg.sender];
13        require(bal > 0);
14
15        (bool sent,) = msg.sender.call{value: bal}("");
16        require(sent, "Failed to send Ether");
17
18        balances[msg.sender] = 0;
19    }
20
21    function getBalance() public view returns (uint256) {
22        return address(this).balance;
23    }
24 }
```

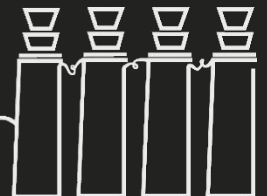
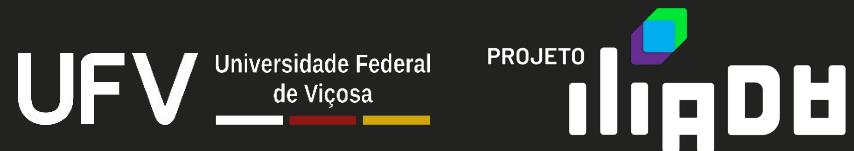
 Analisar contrato

 Baixar PDF



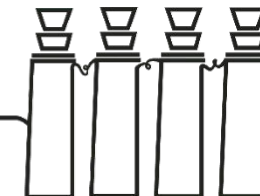
Próximos passos, conhecendo o GT e agradecimentos

Contribuição científica restante da nossa proposta, um pouco sobre o GT e agradecimentos...



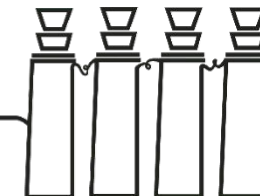
Próximos Passos

- Desenvolver nossa própria ferramenta combinando técnicas existentes para resolver as oportunidades identificadas (Em andamento)
- Consolidar os resultados experimentais dessa nova abordagem em uma nova contribuição científica
- Refinar os artefatos criados em uma Prova de Conceito voltada para resolução do problema de detecção automática de vulnerabilidades em contratos inteligentes



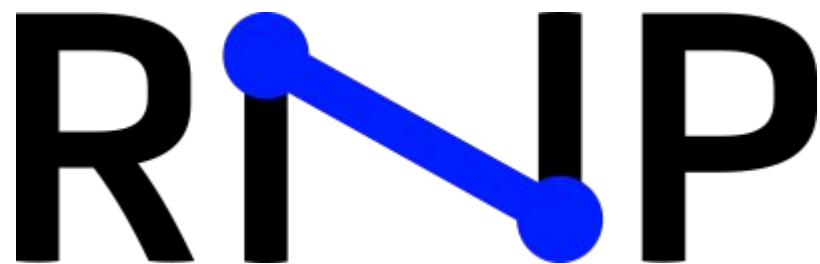
Sobre o Gt-SmartSeg

- Membros do curso de Ciência da Computação da UFV - Campus Florestal
- Equipe:
 - Josué N. Campos - Doutorando
 - Luiz H. S. de Carvalho - Mestrando
 - Isdael R. Oliveira - Mestrando
 - Aline C. S. Silva - Graduanda
 - Iago G. Falcão - Graduando
 - Matheus J. da Silva - Graduando
 - José A. M. Nacif - Coordenador

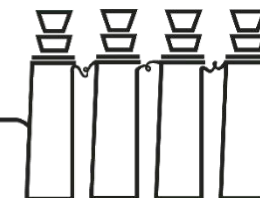


Agradecimentos

- Apoio da RNP e do Projeto ILIADA para realização dos trabalhos
- Oportunidade concedida pelo Observatório Nacional de Blockchain



**OBSERVATÓRIO
NACIONAL DE
BLOCKCHAIN**



Campus Viçosa:

Av. Peter Henry Rolfs, s/n, Campus Universitário

CEP: 36570-900

Viçosa - MG - Brasil | + 55 31 3899 - 2200

Campus Florestal:

Rodovia LMG 818, km 6

CEP: 35690-000

Florestal - MG - Brasil | + 55 31 3536 - 3300

Campus Rio Paranaíba:

Km 7 - Zona Rural, MG-230, Rodoviário

CEP: 38810-000

Rio Paranaíba - MG - Brasil | + 55 34 3855 - 9300



jnacid@ufv.br



josue.campos@ufv.br

