TON Smart Contracts Vulnerabilities

Matvey Mishuris

Smirnova Elizaveta

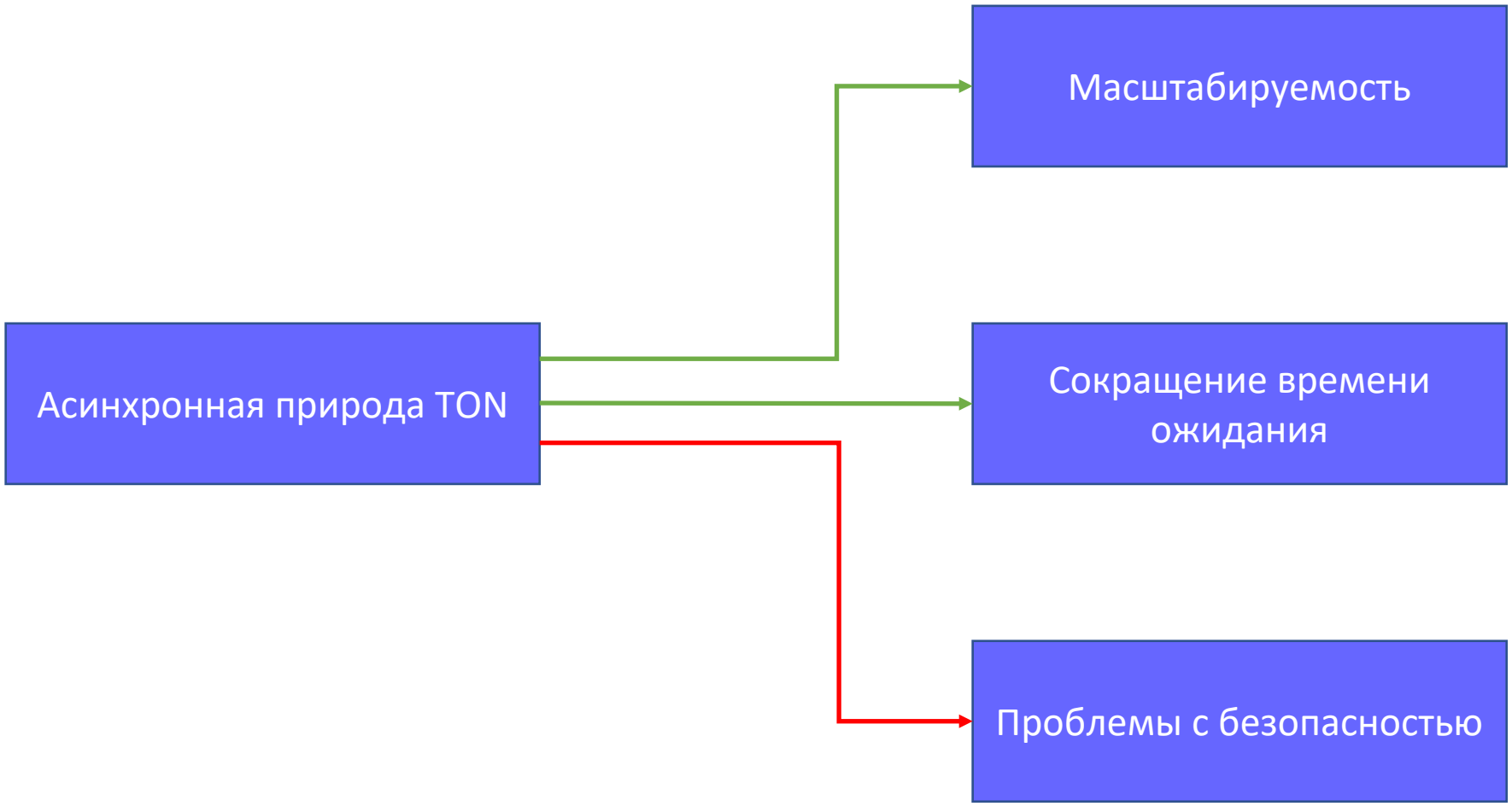
Yury Yanovich

Effectiveness Scalability Efficiency

The Open Network

Smart contracts Decentralized Finance Blockchain

TON


Асинхронная природа TON
Масштабируемость
Сокращение времени ожидания
Проблемы с безопасностью

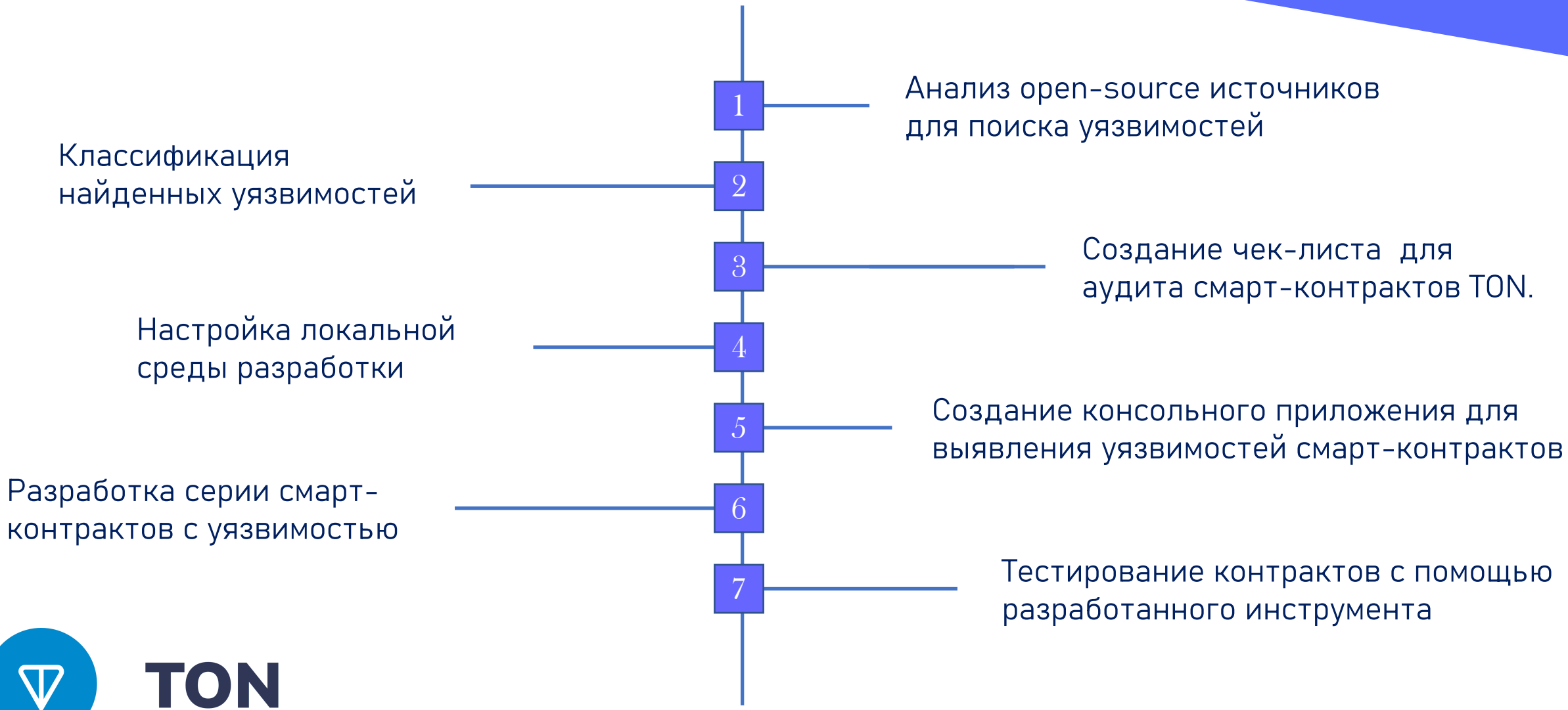
TON

Цель

Разработать инструмент для поиска и устранения уязвимостей в TON smart contracts

TON

План

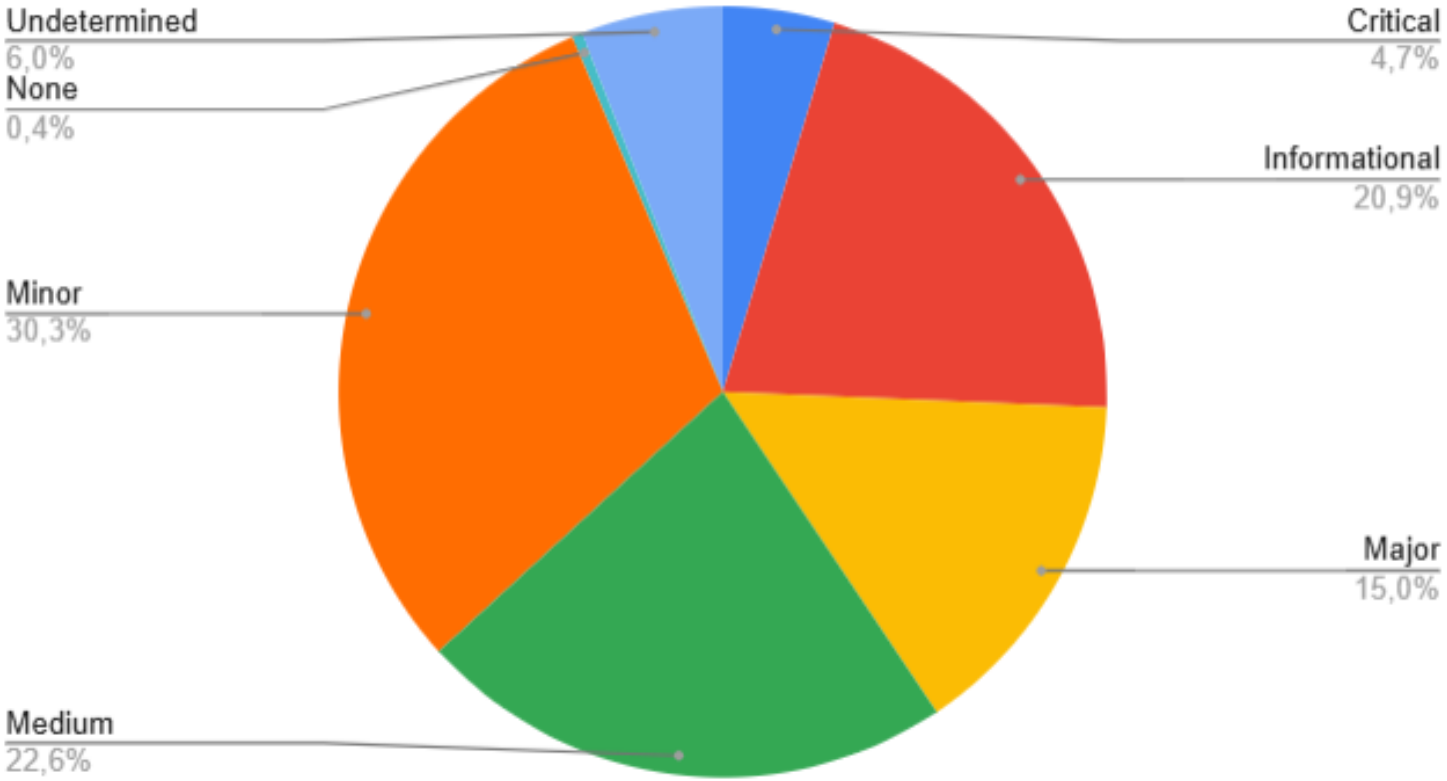

1
Анализ open-source источников для поиска уязвимостей
2
Классификация найденных уязвимостей
3
Создание чек-листа для аудита смарт-контрактов TON.
4
Настройка локальной среды разработки
5
Создание консольного приложения для выявления уязвимостей смарт-контрактов
6
Разработка серии смарт-контрактов с уязвимостью
7
Тестирование контрактов с помощью разработанного инструмента

TON

From Paradigm Shift to Audit Rift: Exploring Vulnerabilities and Audit Tips for TON Smart Contracts

TON

В результате обработки более 30 аудиторских отчётов в общей сложности было зарегистрировано более 230 уязвимостей.



Security Level	Vulnerability Count
Critical	11
Major	35
Medium	53
Low	71
Informational	49
Undetermined	14
Total	233

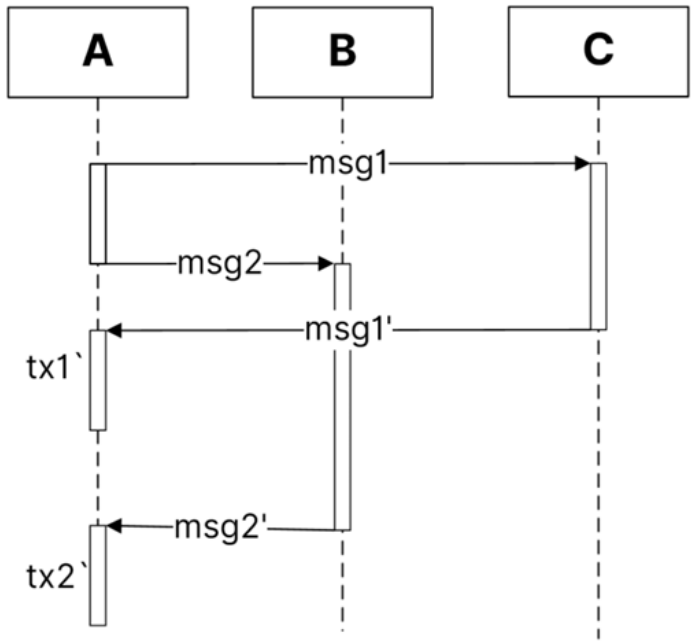
TON

Все уязвимости были представлены в виде таблицы для дальнейшего анализа

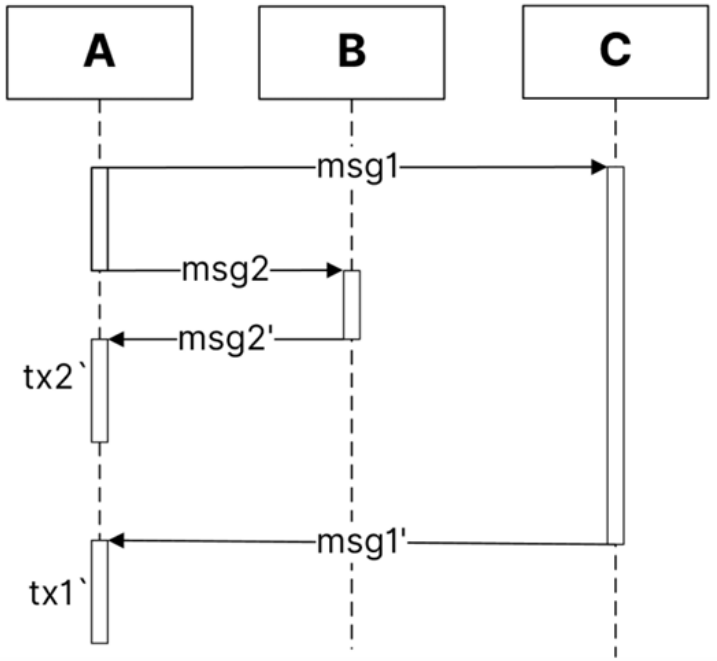
A	B	C	D	E	F	G	H	I	J	K	L
Project	Auditor	Date	Language	Security Level	Status	Vulnerability details	Type	Subtype	Subsubtype	Preliminary Type	Comment for Checklist
TonUP	TonBit	2023-05	Tact	Minor	Fixed	Incorrect event emit in SetTokenWalletAddress and SetUpWalletAddress functions.	Common Errors	Logical Errors	NA	Common Errors	Use correct event emitters to ensure accurate logging and transparency.
TON Locker Contract	Quantstamp	2023-07	FunC	Minor	Acknowledged	Missing validation for `vesting_total_duration` in `locker.load_data()`. Consider checking `vesting_total_duration > 0`.	Contract Design	Input Data Processing	Input Data Processing	Input Data Processing	Ensure proper validation of input data as per TON Audit Guide.
TON Locker Contract	Quantstamp	2023-07	FunC	Minor	Acknowledged	Time constraints on rewards and deposits can lead to unfair distribution. Consider implementing a shorter time limit for reward distribution.	Common Errors	Logical Errors	NA	Logical Errors	Check for logical errors and edge cases in time-based logic.
TON Locker Contract	Quantstamp	2023-07	FunC	Minor	Acknowledged	Potential storage fee insufficiency in `locker_bill` contract. Consider limiting `vesting_start_time + vesting_total_duration`.	Gas Control	Moderate Handling of	Calculation of Costs	Gas Control	Ensure accurate gas control and storage fee management.
TON Locker Contract	Quantstamp	2023-07	FunC	Minor	Acknowledged	Overflow and division by zero errors in edge cases. Consider adding validations for `unlock_period`	Common Errors	Logical Errors	NA	Logical Errors	`total_coins_locked`
TON Locker Contract	Quantstamp	2023-07	FunC	Minor	Acknowledged	Risk of capital inefficiency in reward-lacking deposits. Consider preventing deposits if no reward is added.	Common Errors	Logical Errors	NA	Logical Errors	Ensure proper handling of deposits and rewards.
TON Locker Contract	Quantstamp	2023-07	FunC	Informational	Acknowledged	Inefficiency in reward reservation within the `locker` contract. Consider limiting reserved TON by checking the balance first.	Gas Control	Moderate Handling of	Calculation of Costs	Gas Control	Optimize gas usage and resource management.
TON Locker Contract	Quantstamp	2023-07	FunC	Informational	Acknowledged	Redundant checks of immutable values in `load_data()`. Consider executing checks only once.	Best Practices	Code Review	NA	Code Efficiency	Improve code efficiency by reducing redundant operations.
TON Locker Contract	Quantstamp	2023-07	FunC	Informational	Acknowledged	Lack of documentation for minor business logic. Consider documenting reward and deposit handling in `README.md`.	Best Practices	Documentation	NA	Documentation	Ensure comprehensive documentation of business logic.

BugMagnifier: TON Transaction Simulator to Reveal Smart Contract Vulnerabilities

TON: No Predefined Order



vs



TON

BugMagnifier разработан для анализа безопасности смарт-контрактов в TON. Основным элементом архитектуры BugMagnifier является класс TONDebugConsole.

TON

External interface

Настроена корректная инициализация среды (с автоматической компиляцией FunC контрактов в TVM-байткод и настройкой начального состояния и очереди сообщений через JSON-конфигурации)

```
                TON Debug Console
          Interactive debugger for TON smart contracts


Usage:
  tondebug --contract <path> [--init-state <path>] [--queue <path>] [--help]

Options:
  --contract    <path>  Path to FunC contract source file
  --init-state  <path>  Path to initial state JSON file
  --queue       <path>  Path to initial message queue JSON file
  --help                Show this help message

Example:
  tondebug --contract ./my-contract.fc --init-state ./state.json --queue ./messages.json
```

TON

Race condition

Был специально разработан наглядный FunC-контракт, реализующий механизм депозитного пула с преднамеренно внесенной уязвимостью. Такой подход позволяет на практике показать, как различные последовательности исполнения транзакций могут приводить к неожиданным результатам даже в относительно простых контрактах.


```
{
  "balance": "380543639199",
  "total": 4000000000,
  "owner_address": "EQB3G4DQvv-0L5onuadlo7wgOUH5dtBmUTyBiqyieUk-lAAY",
  "state": {
    "type": "active",
    "code": "b5ee9c7241010e0100b3000114ff00f4a413f4bcf2c80b0102012",
    "data": "b5ee9c724101010101002800004bee6b2800800ee3701a17dff685"
  }
}
```


TON

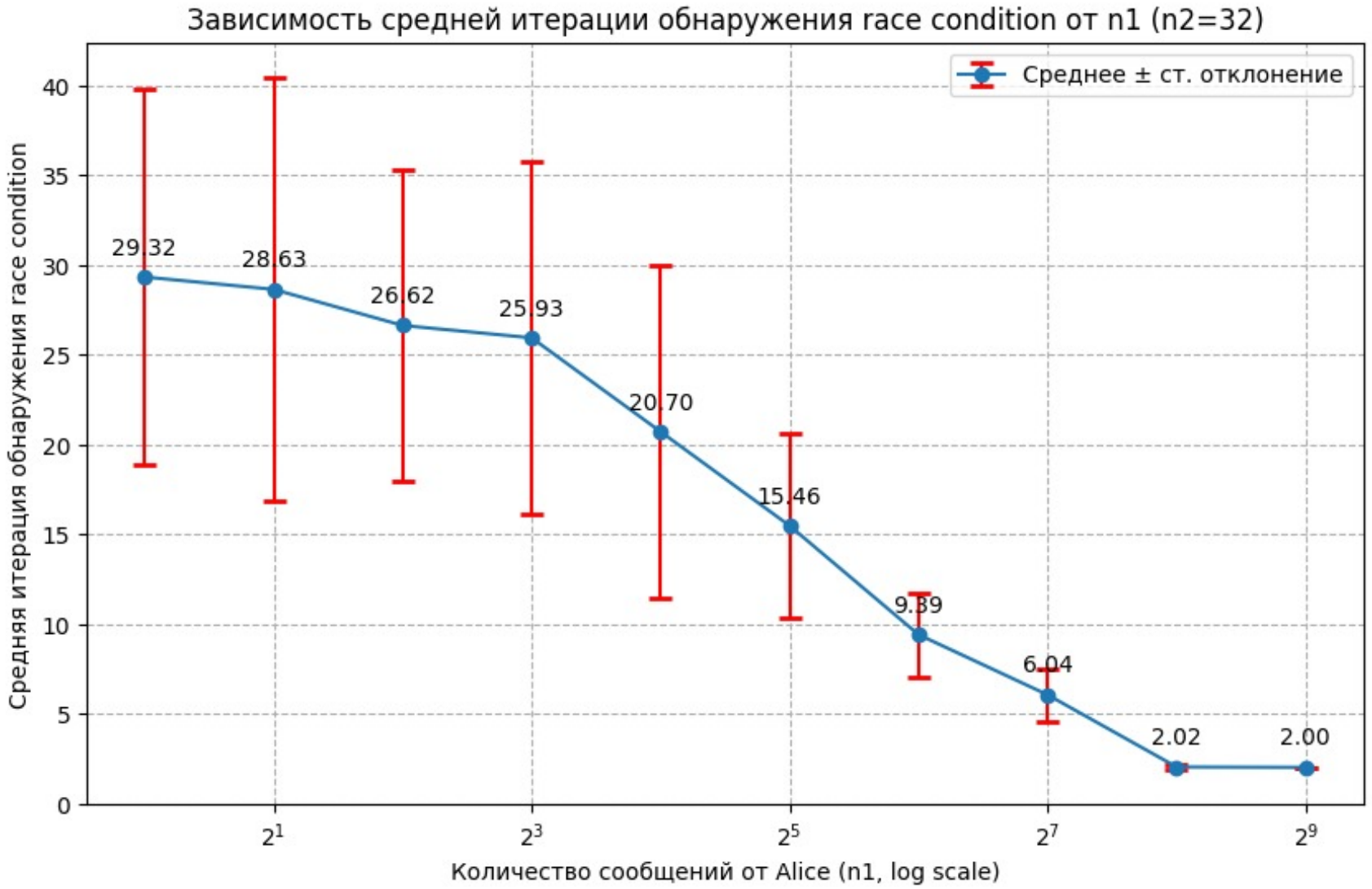
Численные эксперименты

Численный эксперимент позволил оценить сложность обнаружения условий гонки в смарт-контрактах TON в завимисти от исполняемого порядка сообщений очереди.

n1 (Alice)	n2 (Bob)	Total Sum Iterations	Average Iteration	Standard deviation
1	32	2932	29.32	10.45131087
2	32	2863	28.63	11.80451463
4	32	2662	26.62	8.643068105
8	32	2593	25.93	9.851682931
16	32	2070	20.70	9.226115766
32	32	1546	15.46	5.164799272
64	32	939	9.39	2.335042664
128	32	604	6.04	1.469693846
256	32	202	2.02	0.1407052941
512	32	200	2.00	0.00

TON

При фиксированном количестве сообщений от Bob и варьируемом количестве сообщений от Alice для каждой комбинации (n1, n2) генерируется очередь сообщений, после чего выполняется 100 независимых прогонов.


Зависимость средней итерации обнаружения race condition от n1 (n2=32)
Среднее ± ст. отклонение
Средняя итерация обнаружения race condition
Количество сообщений от Alice (n1, log scale)
29.32
28.63
26.62
25.93
20.70
15.46
9.39
6.04
2.02
2.00
0
5
10
15
20
25
30
35
40
2^1
2^3
2^5
2^7
2^9

TON

Технологический стек
Языки программирования

TypeScript

- Статическая типизация снижает риск ошибок при работе со сложными структурами TON (ячейки, транзакции).
- Упрощает рефакторинг и поддержку кода.
- Позволяет писать кастомные скрипты на TS, которые транспилируются в JS для выполнения.

FunC

- Официальный язык для разработки смарт-контрактов в TON.
- Компилируется в байт-код Fift, который исполняется в TVM (TON Virtual Machine).

TON

Результаты

Достигнутые:

- Разработан чек-лист для аудита смарт-контрактов на основе составленной классификации
- Оформленная статья отправлена в научный журнал "BCRA" (Blockchain: Research and Applications)
- Разработан инструмент в виде интерактивной консоли для аудита смарт-контрактов
- Разработана и протестирована при помощи BugMagnifier серия контрактов с Race Condition

Будущие:

- В процессе финальной обработки находится статья по второму блоку работы

Ссылки:

- https://docs.ton.org/v3/documentation/smart-contracts/func/docs/functions
- https://github.com/ton-org/sandbox
- https://test.ton.org/tblkch.pdf
- https://github.com/ton-org/blueprint?tab=readme-ov-file#contract-development
- https://tonbit.xyz/reports/TonUP-Smart-Contract-Final-Audit-Report.pdf
- https://certificate.quantstamp.com/full/ton-locker-contract/6872997f-1110-45cc-b70f-2a4cd639da1f/index.html
- Разбор конкретных кейсов
- Тестирование
- Submit первой статьи

TON