EVM Wallet Fraud Detection

Rabby-style Approval Risk Scanner using Ethereum Logs

FRAUD DETECTION MVP

The Scam Doesn't Need Your Password

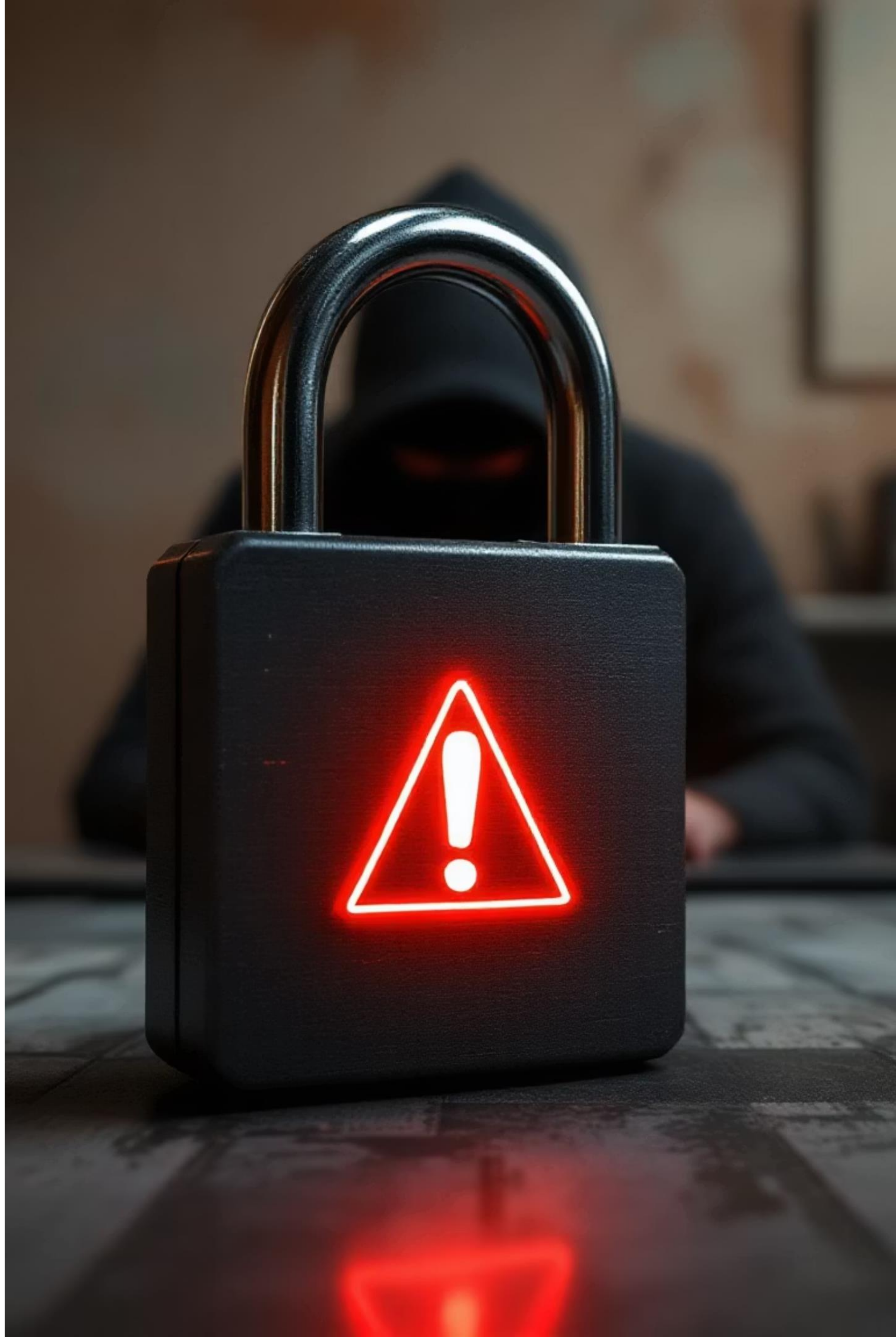
One wrong click on "Approve" can drain tokens later.

What users do

- *Connect wallet to a website*
- *Click Approve to continue*
- *Assume it is safe*

What scammers exploit

- *Unlimited approvals*
- *Malicious spender contracts*
- *Delayed token drain using transferFrom*



Problem & Objective

Problem: Users approve contracts without understanding

- *Approvals grant spending permission*
- *Malicious spender can drain tokens later*

Objective: Build a Rabby-style approval risk scanner

- *Scan Approval logs from Ethereum*
- *Flag unlimited approvals with a risk score*

Blockchain Basics

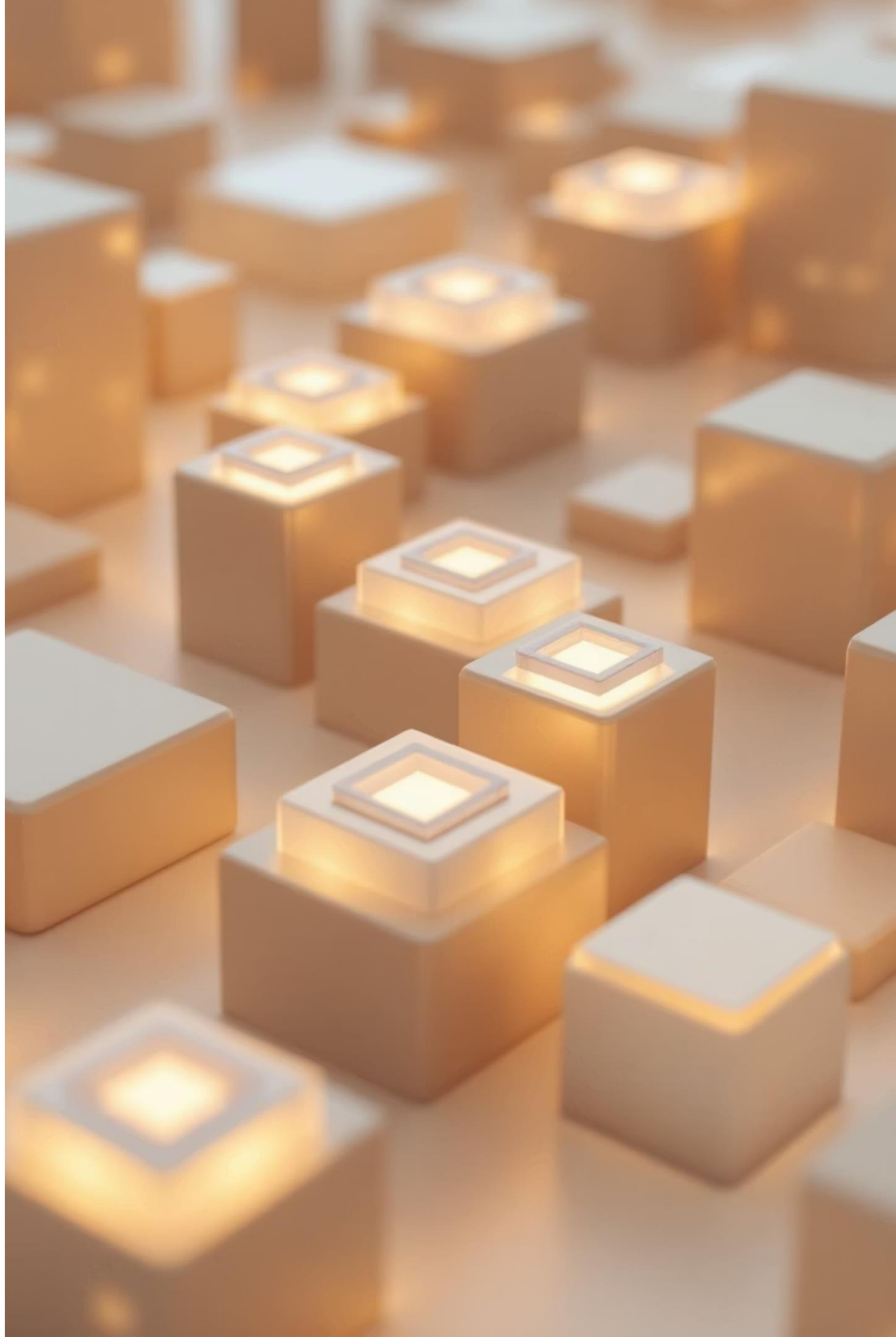
A public digital record book shared by everyone.

Transactions grouped into blocks

Blocks linked in time order

Anyone can verify history

Transparent and tamper-resistant

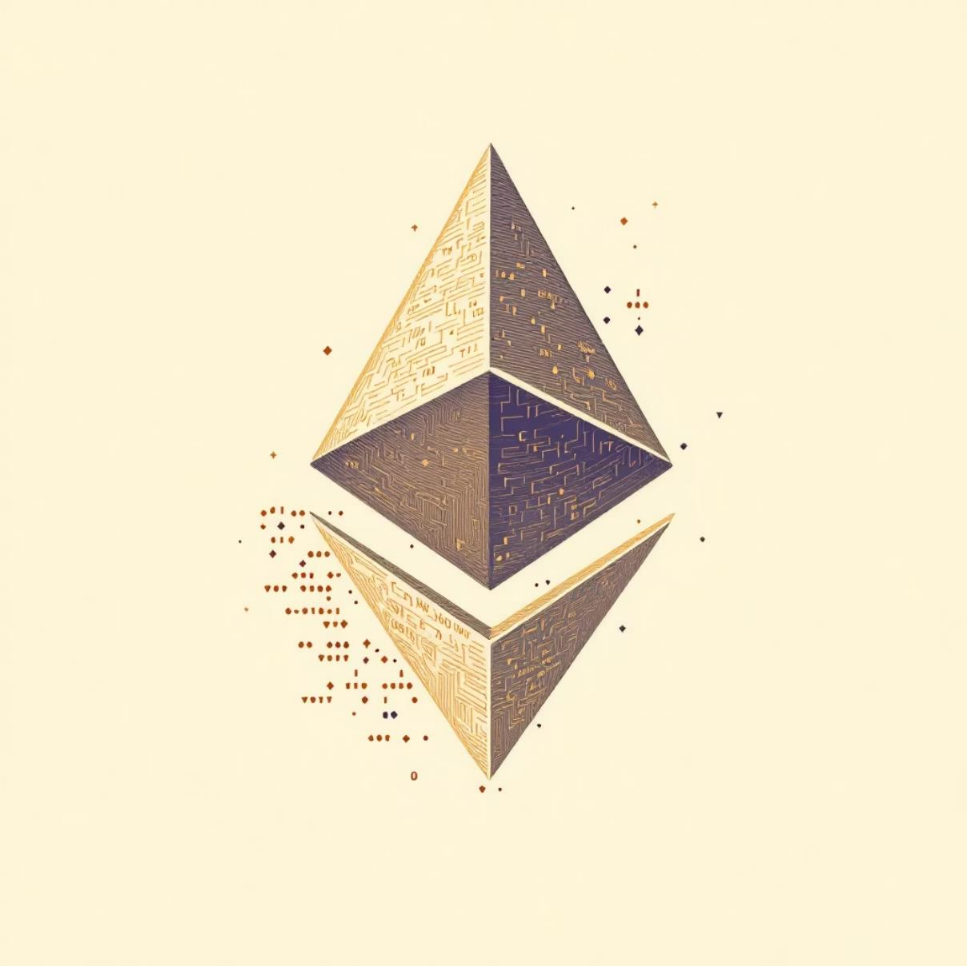


Why Ethereum / EVM?

Smart contracts + tokens → approvals become a major fraud surface.

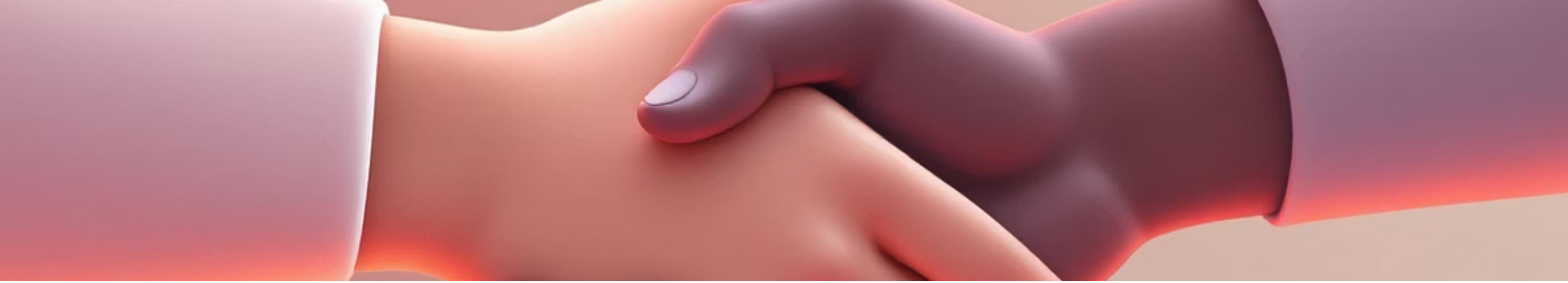
Bitcoin

- ***Mainly digital money***
- ***No smart-contract approvals***
- ***Different fraud patterns***



Ethereum / EVM

- ***ERC-20 tokens (USDT/USDC)***
- ***DeFi apps require approvals***
- ***Approvals can be abused by scammers***



What is Approval?

ERC-20 Token Approval

`Approval(owner, spender, value)`

1

Owner

Wallet address (user)

2

Spender

Contract/app address

3

Value

Amount allowed to spend

Unlimited approvals are the highest risk.

Fraud Pattern: Unlimited Approval

`value = 2^256 - 1 (MAX uint256)` → ***High risk***

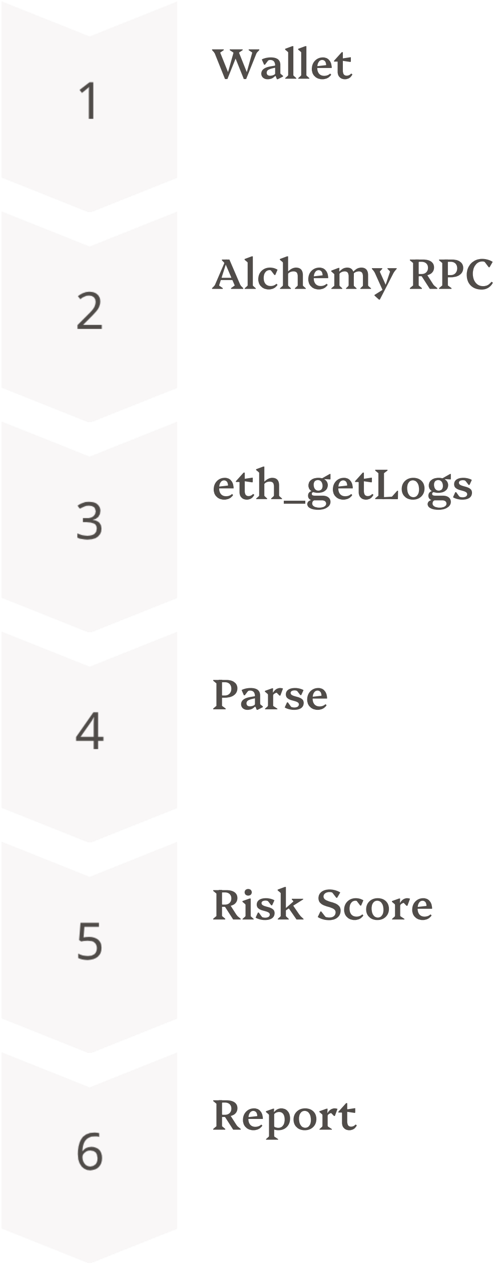
Unlimited approvals are convenient but dangerous

Malicious spender can drain tokens anytime

Rabby warns users — we detect the same signal

System Architecture

End-to-end pipeline



1 Wallet

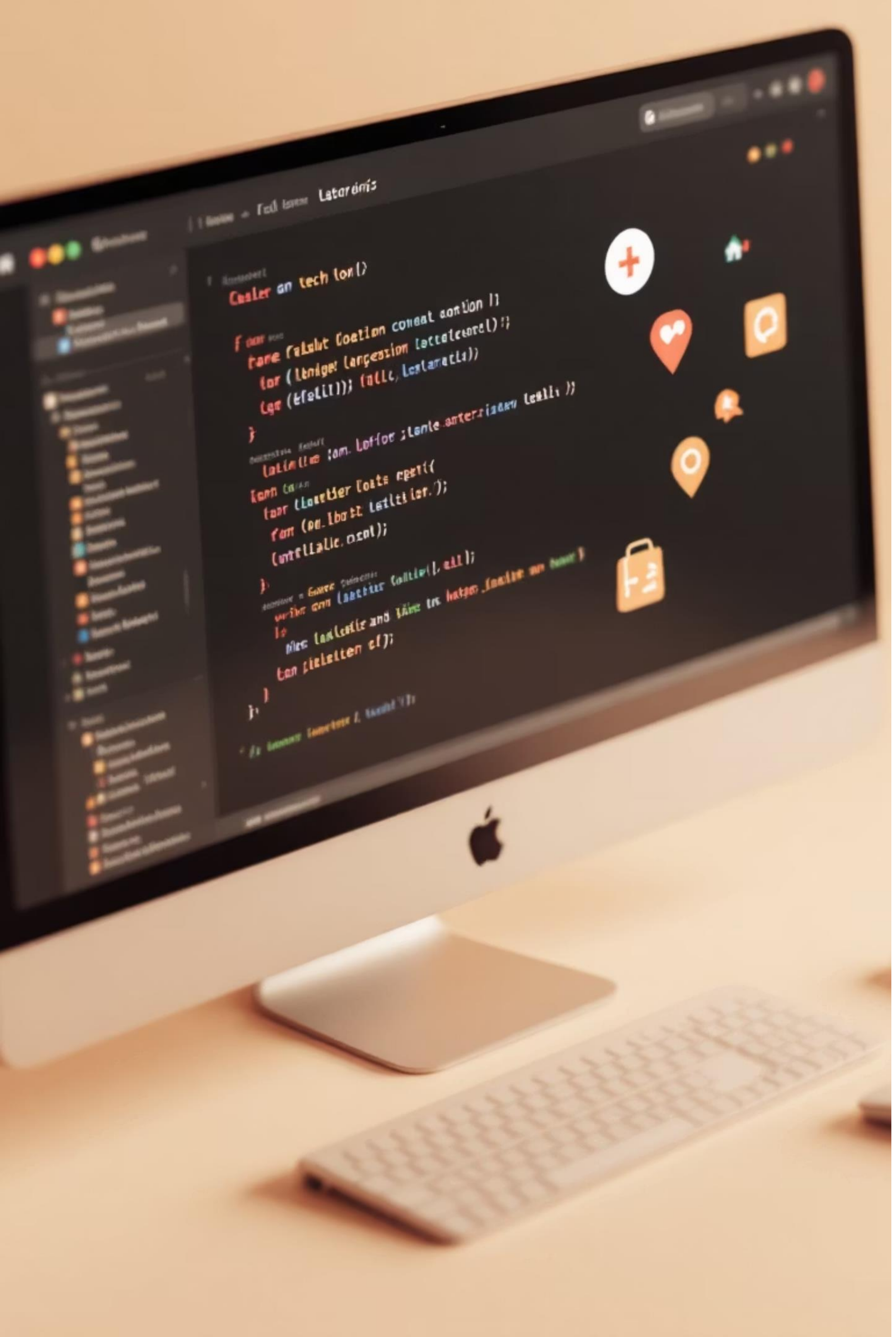
2 Alchemy RPC

3 eth_getLogs

4 Parse

5 Risk Score

6 Report



Tech Stack

Lightweight MVP implementation

Python

Core logic

Requests

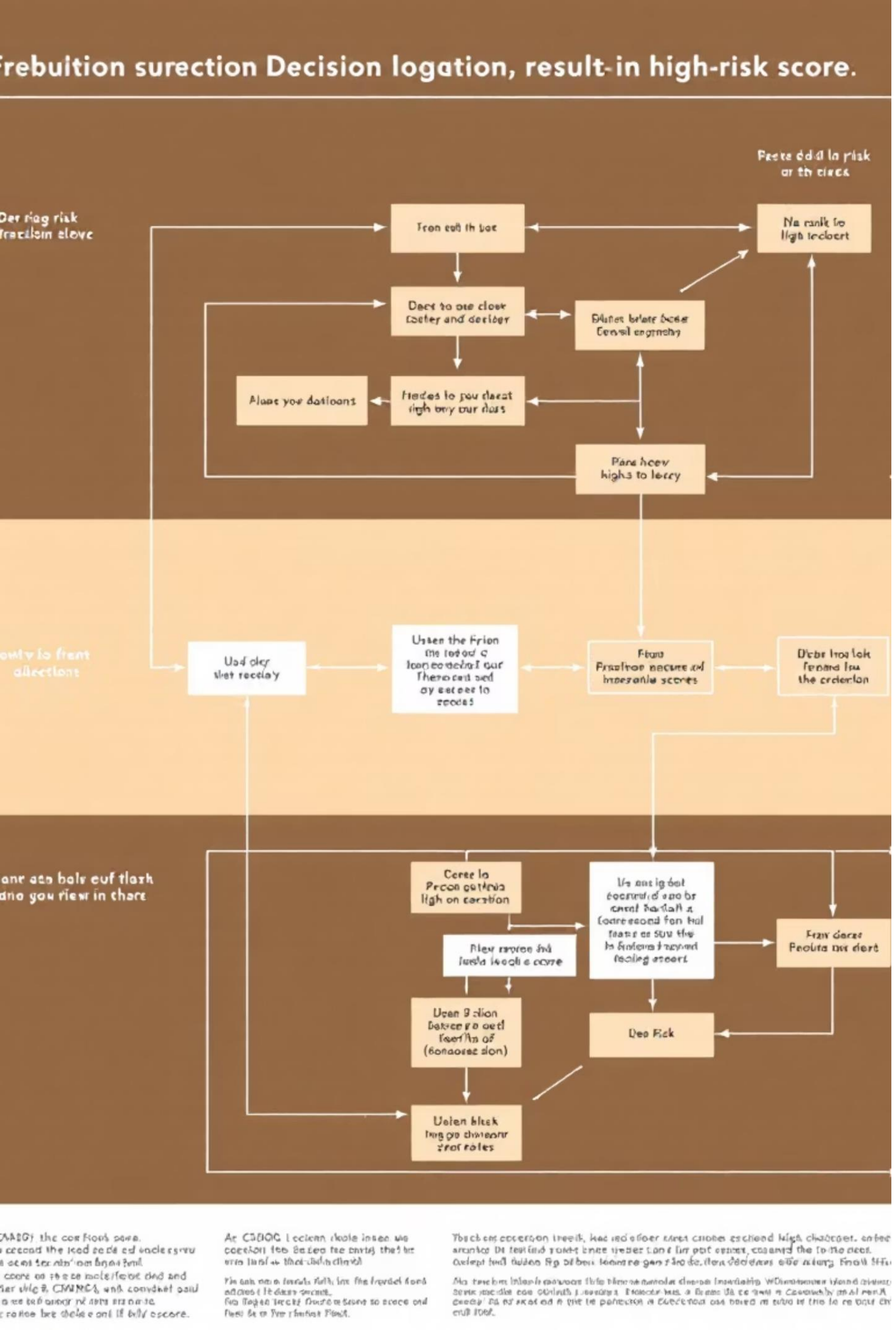
JSON-RPC calls

Pandas

Report table

Alchemy RPC

Ethereum endpoint


rebuition surection Decision logation, result-in high-risk score.
Peete édd la risk or th cirea
Der rieg risk traclom elove
Tron eull th bae
Ne ranik lo ligh teclaert
Dacs to ore clowr cotey and decloer
Blunes brlate boae Cervil engrnchy
Alune yoe datloons
Hudes to jou daest ligh bey our llars
Pare heev highs to leccy
only to frent allections
Ud dey thet reday
Useen the Frien
Feuro Prasfron necure of Inoesonle scores
Debe Ince loit Ireane lou the crderlon
onr aea bole euf tlash dno you riesr in charc
Cerre lo Preon gettrea ligh on carstion
Nlev rsyrce ihd Iuela iseoli a ccyre
Uean 9 llon Datcere o oetl Ioerlin of (Sonaoree slon)
Frav deres Peoura nor dert
Deo Fick
Uslen blesk
Ihg go thweanr Irot rales

Detection Logic

Rule-based scoring for clear alerts

1

MVP Rule

If allowance == MAX_UINT256 → risk_score = 60 (High)

2

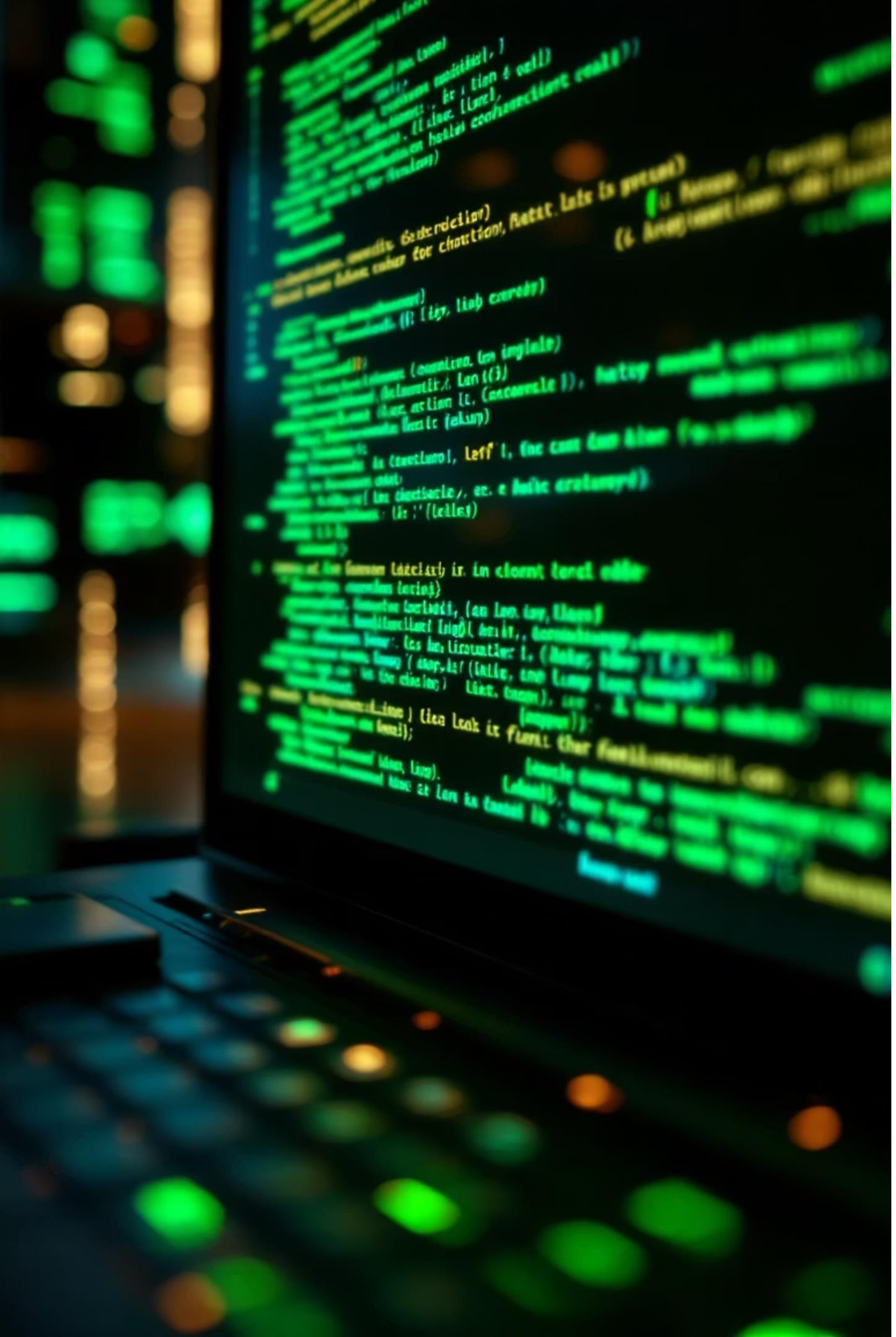
Reason Shown

"UNLIMITED approval"

3

Report

Shows latest approval per token + spender



Demo: Command-Line Wallet Scan

Our demo showcases a command-line tool for scanning wallet approvals.

Command

```
py evm_approvals_scan.py --rpc "<ALCHEMY_URL>" --address "" --lookback_blocks 20000 --chunk 10
```

Output & Goal

Generates a risk-ranked report of approvals to warn users before potential token drains.

Results: What We Achieved

Our Minimum Viable Product (MVP) successfully delivered key functionalities for approval scanning.

Fetched Approval Logs

Successfully retrieved approval logs directly from the Ethereum blockchain.

Filtered Wallet Approvals

Efficiently filtered logs to display approvals relevant to a specific wallet address.

Risk Report Generation

Detected unlimited approvals and generated a comprehensive risk report for user review.

Limitations: Current Constraints

Despite our achievements, certain limitations currently impact the system's full potential.

Alchemy Free Tier

`eth_getLogs` ***is limited to 10 blocks per request, affecting data retrieval speed.***

No Token Decoding

Lacks functionality for decoding token symbols or names from contract addresses.

No Spender Reputation

Absence of a spender reputation or blacklist database for enhanced risk assessment.

Future Enhancements: How We Can Improve

Our roadmap includes several key enhancements to elevate the system's capabilities and user experience.

Token Metadata

Integrate ERC-20 calls for comprehensive token symbol and name decoding.

Spender Reputation

Develop a database for spender reputation and known scam addresses.

Drain Pattern Detection

Implement detection for `approve` → `transferFrom` ***drain patterns.***

Web Dashboard

Create an intuitive web interface for non-technical users to access features.

Questions & Answers

We welcome your questions and feedback on our project.

Engage with Us

This is an opportunity to discuss the demo, results, and future plans.

Your Insights

We value your perspective and any suggestions you may have.