AI-Enhanced Security in Decentralized Finance: Leveraging LLMs for Proactive Defense

Liyi Zhou
PhD student @Imperial College London

DeFi Attacks on Ethereum & BSC

>181 DeFi “attacks”

3.24B USD losses

2018-04 **2022-04**

IEEE S&P 2023

https://eprint.iacr.org/2022/1773.pdf

DeFi Attacks

DefiLlama

Total Value Hacked (USD)

$7b

Total Value Locked

$38.605b

18%!!!

Defence in Depth

Pre-deployment

Best practices

Pre-audit[1]

Audit[2]

Transfer limits

Pause button

Post-deployment

Intrusion Detection[3]

Intrusion Prevention[4]

Vulnerability Search[5]

Similarity Detection[6]

Post-incident

Fast Forensic[7]

Post Imitation

[1] Still need an audit? – 2023
[2] Securify – 2018
[3] BlockGPT – 2023
[4] Imitation Game – 2023
[5] DeFiPoser – 2021
[6] DeFi Attack SoK – 2023
[7] Fast Forensic – 2023

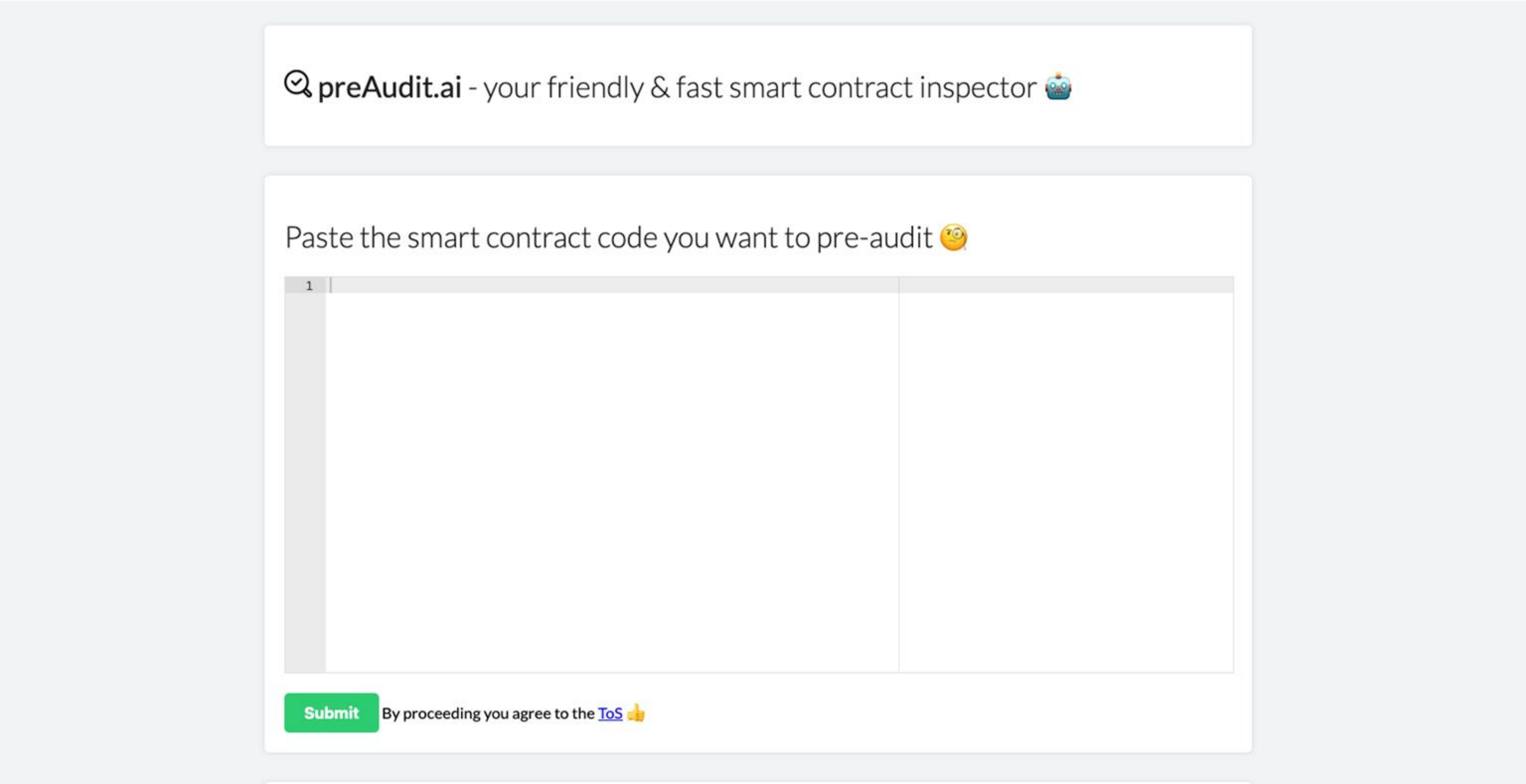
1) Pre-audit

2) BlockGPT

Pre-audit

Isaac David, Liyi Zhou, Kaihua Qin,
Dawn Song, Lorenzo Cavallaro, **Arthur Gervais**

What if..


preAudit.ai - your friendly & fast smart contract inspector
Paste the smart contract code you want to pre-audit
1
Submit
By proceeding you agree to the ToS

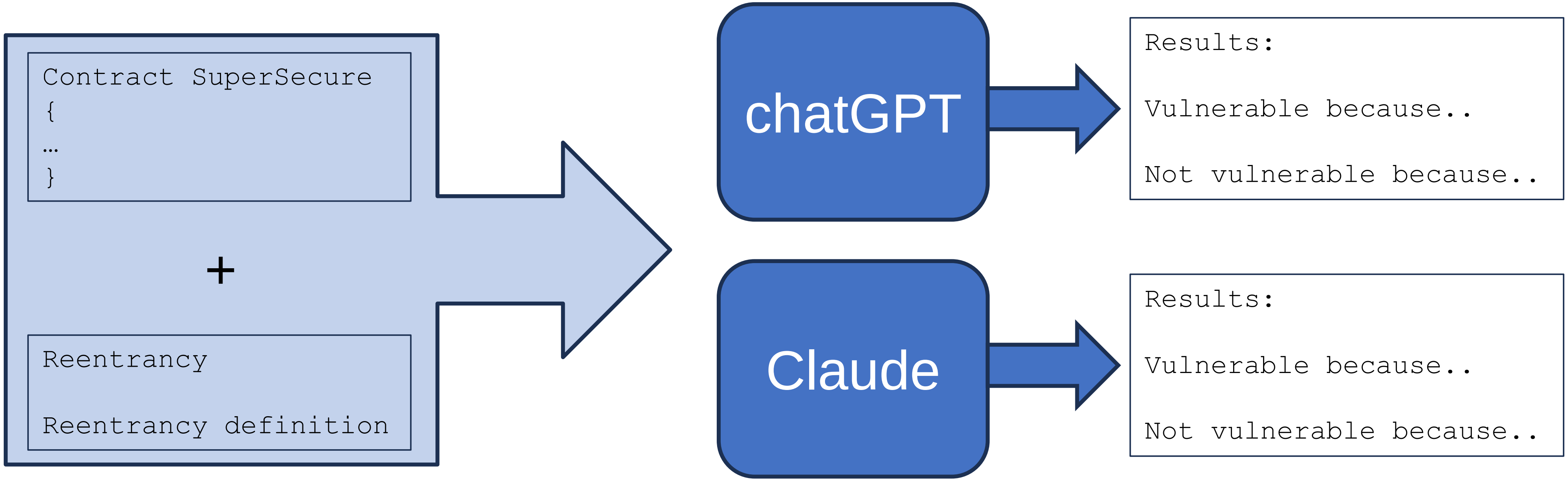
What if..

Thank you! The pre-audit evaluation started ✅.

In about 5 minutes, you will receive an email with the pre-audit ⏳.

Please keep in mind, a preAudit does not replace an audit and may contain many false positives, we're looking forward to your feedback!

Pre-audit



Contract SuperSecure
{
…
}
+
Reentrancy
Reentrancy definition
chatGPT
Claude
Results:
Vulnerable because..
Not vulnerable because..
Results:
Vulnerable because..
Not vulnerable because..

An automated Smart Contract Review

First reviewer:

List of evaluated attacks

- On-chain oracle manipulation
- Reentrancy
- Absence of code logic or sanity check
- Inconsistent access control

Vulnerabilities found

- Inconsistent access control

Explanations

On-chain oracle manipulation.

The provided smart contract is not vulnerable to 'On-chain oracle manipulation'. The reason is simple: this contract does not use an oracle. An oracle is an external data source, and this contract does not have any functionality that depends on any external data source. The contract merely stores a message and allows that message to be updated. Because an oracle is not involved, there's no opportunity for an attacker to manipulate an oracle to affect this contract's behavior.

Reentrancy.

The given smart contract is not vulnerable to 'Reentrancy' attack. In a 'Reentrancy' attack, an attacker takes advantage of the call to an external contract to take control and repeatedly call a function in the original contract before it has completed. This contract does not call any external contracts which could allow for control to be unjustifiably taken over by a malicious contract. Also, the function 'update'

Tested Vulnerabilities + Ground Truth

- 38 categories, e.g.
  - Reentrancy
  - On-chain oracle manipulation
  - Absence of code logic or sanity checks
- 51 vulnerable contracts
  - Vulnerabilities on 4 system layers

Results?

Tool Name	TP	FP	FN	TN	Precision	Recall	F1 Score	Accuracy
claude-v1.3-100k	28	576	44	1290	0.0464	0.389	0.0828	0.680
gpt-4-32k	15	216	57	1650	0.0649	0.208	0.0990	0.859
slither	5	142	68	1723	0.0340	0.0685	0.0455	0.892
oyente	0	38	73	1827	0.0000	0.0000	0.0000	0.943
confuzzius	0	4	73	1861	0.0000	0.0000	0.0000	0.960
mythril	1	48	72	1817	0.0204	0.0137	0.0164	0.938
solhint	5	109	68	1756	0.0439	0.0685	0.0535	0.909

Considerations

- Training data
- Reproducibility
- Binary or Non-binary classification
- False Positives
- Truncation
- Context length
- Model temperature

Do we still need a manual audit?

Yes, for now 🫢

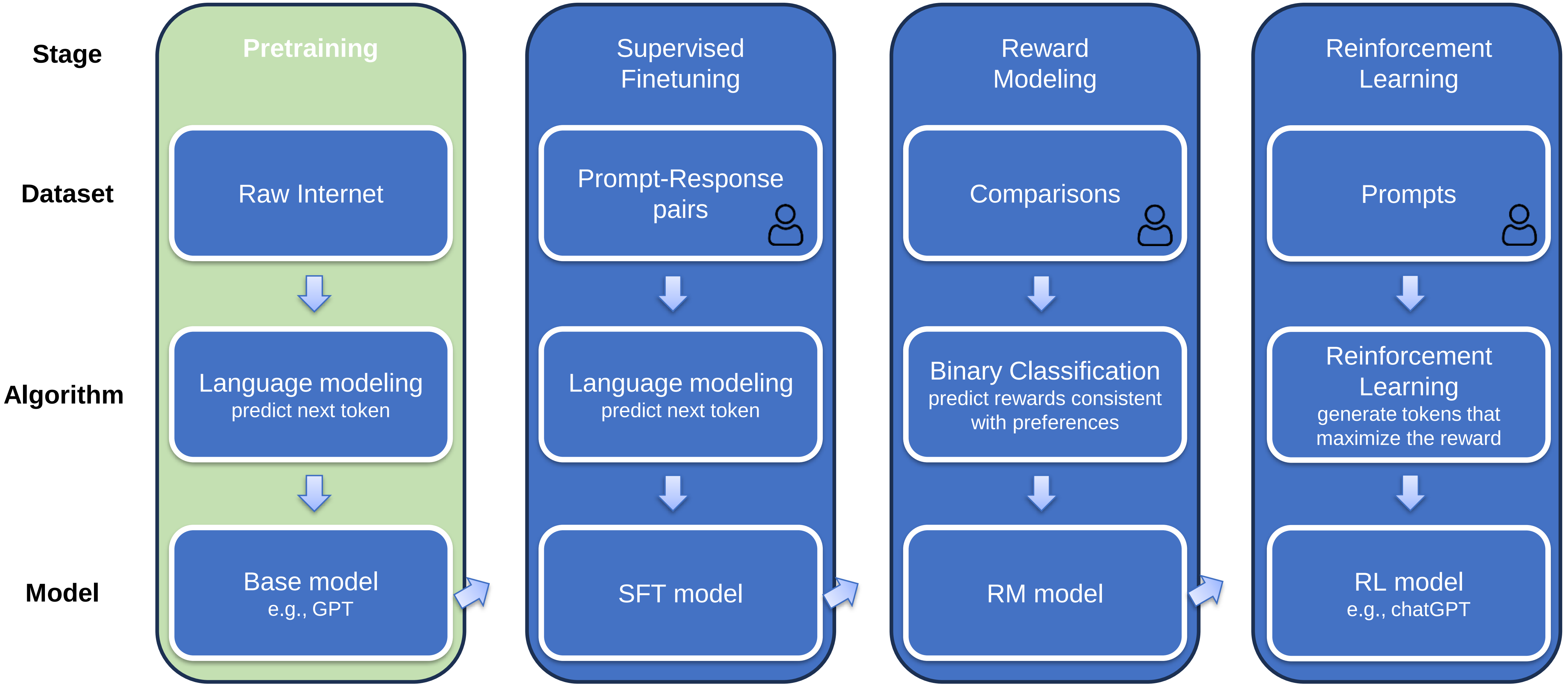
Are the LLMs better than existing tools?

Sometimes 🫢

BlockGPT

Yu Gai*, Liyi Zhou*, Kaihua Qin,
Dawn Song, **Arthur Gervais**

GPT Training Pipeline



Contributions

- Self-supervised learning for smart contract anomaly detection

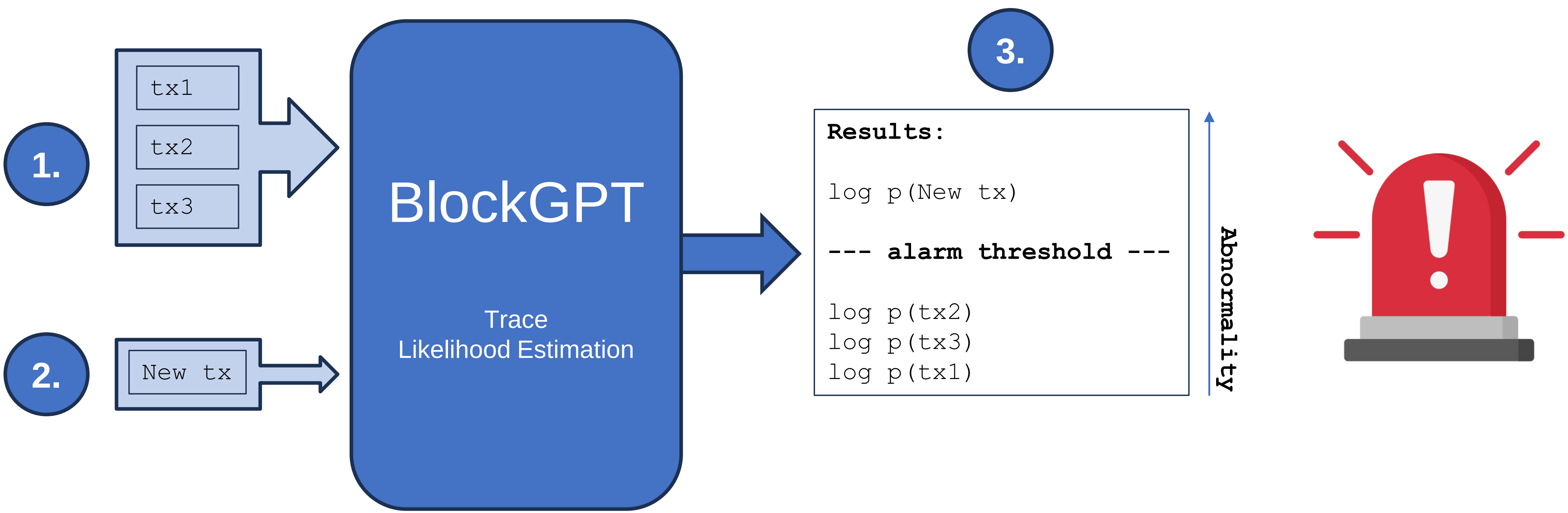
- BlockGPT ranks
  - 20/124 as most abnormal
  - 20/124 as second most abnormal
  - 7/124 as third most abnormal

- 2k transactions/second batched throughput
  can be used as Intrusion Detection System

Challenges of conventional ML-based IDS

- Binary classifier on labels: `f(tx)→{Attack, Benign}`
- Limited labelled attack data, attack patterns evolve
- Only <100 attacks/year

BlockGPT



1.
tx1
tx2
tx3
2.
New tx
BlockGPT
Trace
Likelihood Estimation
3.
Results:
log p(New tx)
--- alarm threshold ---
log p(tx2)
log p(tx3)
log p(tx1)
Abnormality

BlockGPT Advantages

- No engineered rules, data driven.
- Can detect new attacks not covered by known rules.
- Can detect non-profitable attack transactions!

Threat Model

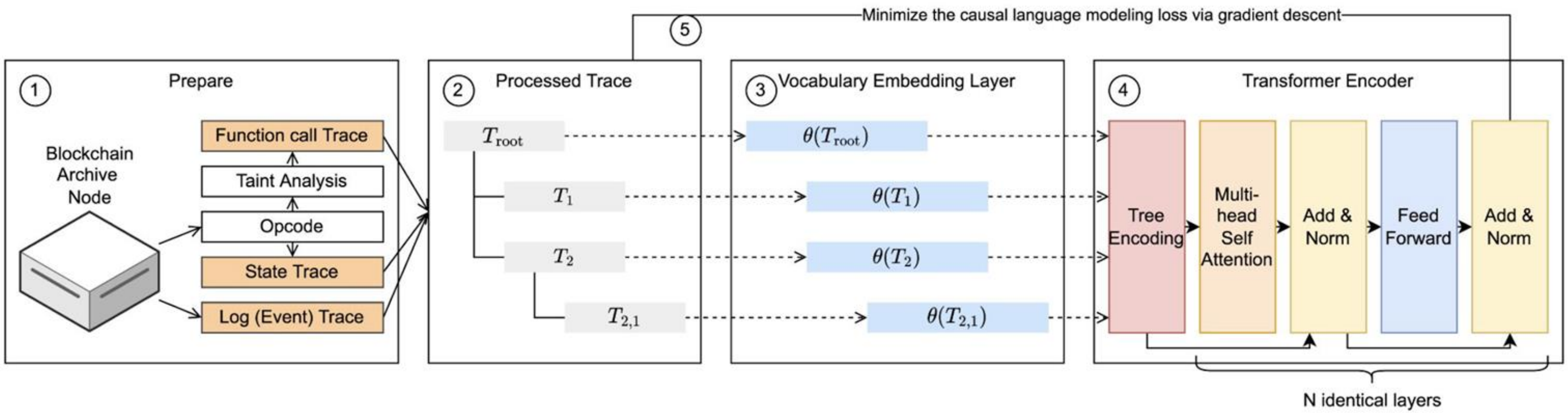
- Computationally bounded
- Money!
  - **Observable Adversary**: e.g., transactions propagate on a P2P
  - **Hidden Adversary**: e.g., colluding with a miner

Dataset

- Unlabeled (pretraining)
  - 68M txs/1523 days from victim dApps
- Labeled (evaluation only)
  - 124 DeFi attack
  - Possibly multiple attack transactions per dApp

BlockGPT Architecture

TX -> Tokenized Trace -> Trace Embedding -> Trace Likelihood



BlockGPT Results

Attacks ranked as most abnormal

Victim Name	Victim Contract	Application Categories	Damage (in USD)
Beanstalk	0xc1e0..24c5	Stablecoin	181,500,000
MonoX	0x66e7..ee63	DEX	31,133,333
PopsicleFinance	0xd63b..3546	Yield farming	20,700,000
PrimitiveFinance	0x9dae..f2f9	Derivatives	13,000,000
PunkProtocol	0x929c..49d6	Others	8,950,000
VisorFinance	0xc9f2..14ef	Others	8,200,000
DAOMaker	0xd6c8..b1ec	Others	4,000,000
DAOMaker	0x933f..2a13	Others	4,000,000
DODO	0x051e..a2b6	DEX	3,800,000
DODO	0x509e..41fb	DEX	3,800,000
CheeseBank	0x833e..743d	Digital Bank	3,300,000
dydx	0x5377..ba2c	Derivatives	2,211,000
RevestFinance	0xe952..1659	Others	2,005,000
BTFinance	0x3ec4..8af0	Yield farming	1,600,000
VisorFinance	0x65bc..054f	Others	975,720
WildCredit	0x7b3b..c6ca	Lending	650,000
SharedStake	0xa231..7ef5	Others	500,000
88mph	0x2165..b0a6	Lending	100,000
SanshuInu	0x35c6..7810	Others	100,000
KlondikeFinance	0xacbd..e747	Synthetic assets	22,116

Conclusions

- Self-supervised learning for anomaly detection
- Detects attacks *without* engineered rules
- High throughput

Paper: https://eprint.iacr.org/2023/592
Further details: https://rdi.berkeley.edu/


Thank you!

Transformer-based trace embedding

- Tokenization
  - Customized tokenization for DeFi (100k+ tokens)
    - 93233 Ethereum addresses
    - 6759 function signatures
  - Informative low-level instructions
    - EVM execution logs
    - EVM memory read/write
- Our transformer
  - 8 layers, each self-attention + position-wise feed-forward layer
  - About 1 Billion parameters

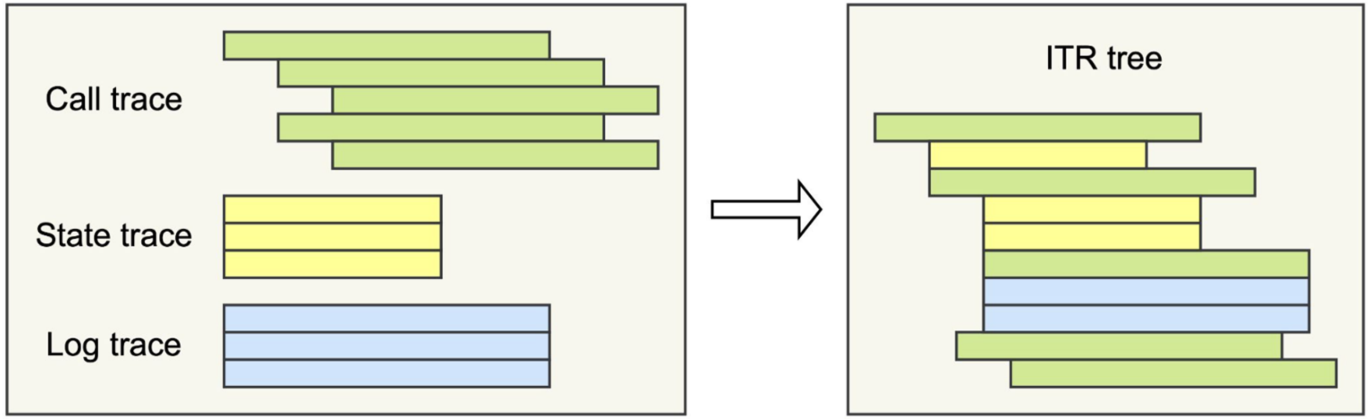
Tokenization Challenges

- Limited number of tokens (512, or 1024)
  - Traces can be large

Tokenization: from raw trace to tokens

- Raw trace as Intermediate Tree Representation (ITR)

```
CALL:
|    from: 0x99d...
|    to: 0xe59...
|    data: c4f...
|- DELEGATECALL:
 |   from: 0xe59...
 |   to: 0xe...
 |   data: f39...
 |- READ, 0x95c..., 0x67a
 |- LOG1, 0x0b8..., 0x699
```



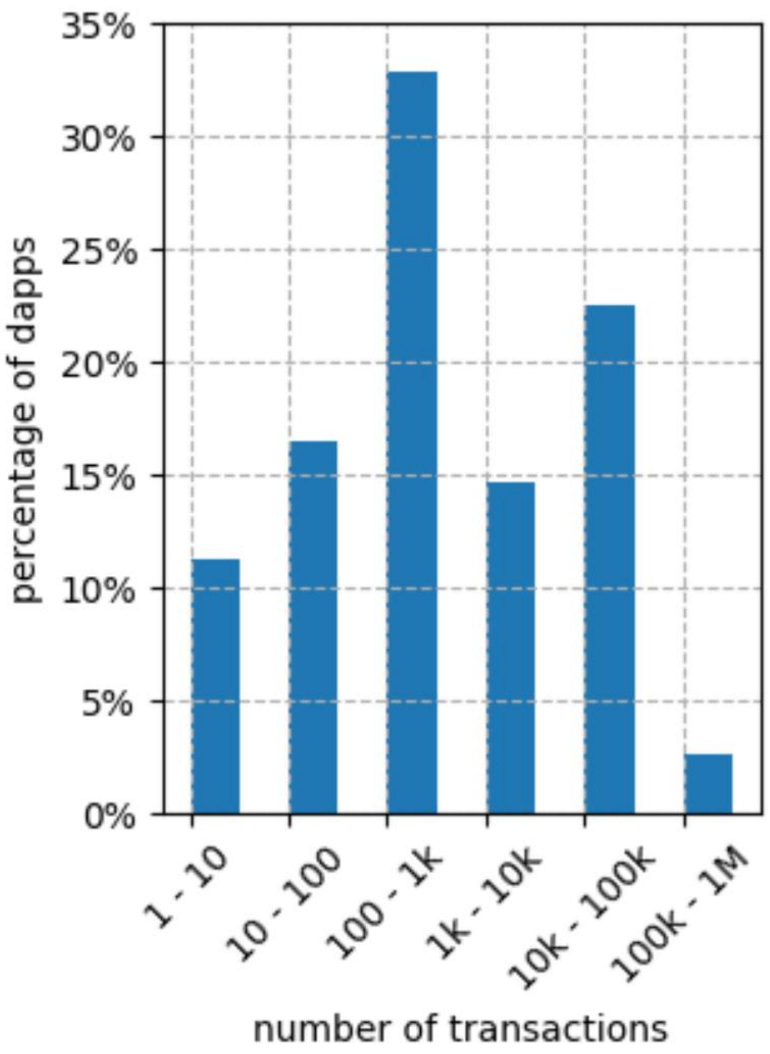
- Tokenized trace

```
CALL, from, 0x99d…, to, 0xe59…, data: c4f… DELEGATECALL, from, 0xe59…, to, 0xe…, data, f39…, READ, 0x95c…,
0x67a, LOG1, 0x0b8…, 0x699
```

Dataset

- Vulnerability layers
  - Smart Contract (42%)
  - Protocol (40%)
  - Auxiliary (30%)

- dApp transaction activity
  - Minimum: 4
  - Maximum: 0.6M



IDS based on estimated likelihood rank

- Given a DeFi app
  - BlockGPT estimates the log-likelihood of the traces of all transactions involving the app
  - Raises alarm for the k least likely, i.e. most abnormal transactions.

- k can be adjusted depending on the dApp & costs.

- No labeled data required for training.

Mutation Testing

```
Contract SuperSecure
{
…
}
```

Manually added vulnerability →

```
Contract SuperSecure
{
…
Vulnerability 1
…
}
```

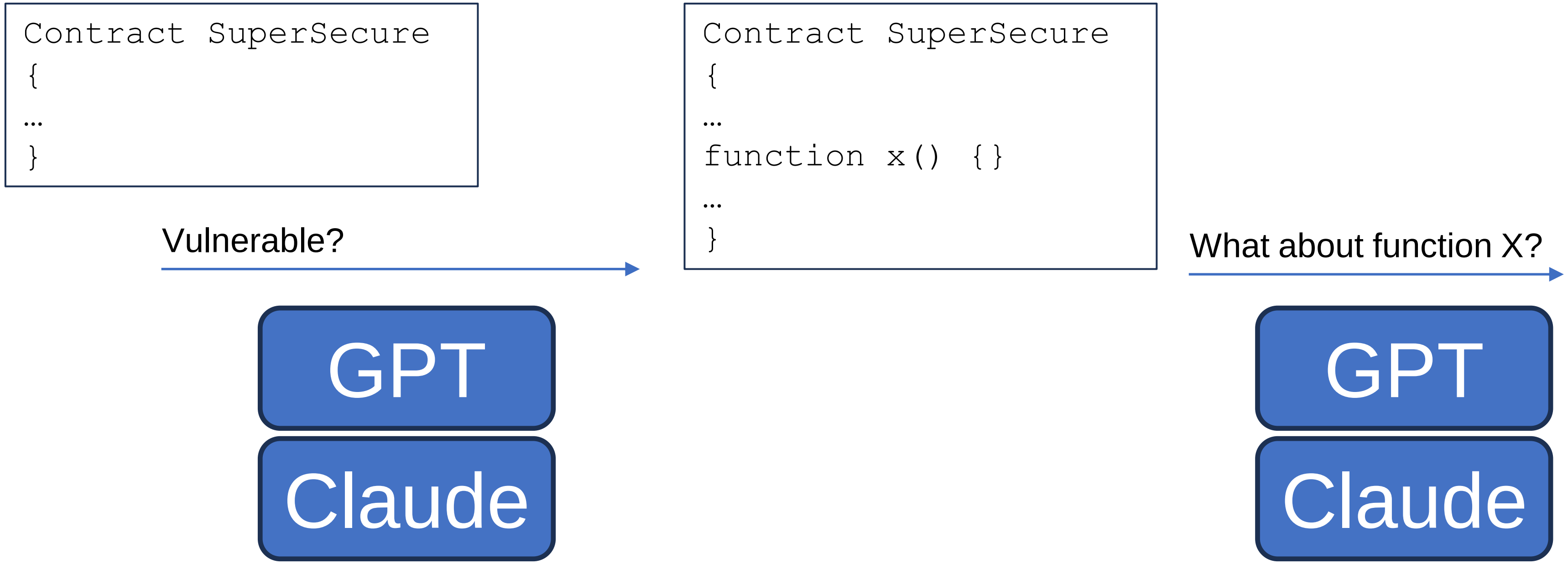
Ask the models →

GPT

Claude

GPT-4 non-binary:
78.7% true positives

Chain of Thought



Related Work Landscape

Technique	Assumed Prior Knowledge	Searchspace Unrestricted From Vulnerability Patterns	Real-Time Capable	Application Agnostic
Rank based – the goal is to find all unexpected execution patterns, implicitly capturing vulnerabilities				
BLOCKGPT (this paper)	All historical transactions	Unrestricted	●(0.16s)	●
Reward based – the goal is to extract financial revenue, implicitly capturing vulnerabilities				
APE [21]	N/A	Only profitable patterns	●(0.07s)	●
Naive Imitation [6]	N/A	Only profitable patterns	●(0.01s)	●
DeFiPoser [12]	DApp models	Only profitable patterns + Limited by the DApp models	●(5.93s)	○
Pattern based – the goal is to match / classify predefined known vulnerability patterns with rules (including machine learning methods)				
Pattern based dynamic analysis [19], [22], [23]	Rule	Limited by the rule	●	◐
Pattern based fuzzing [24]–[29]	Rule + ABI / DApp models	Limited by the rule	◐	◐
Pattern based symbolic execution [28], [30]–[40]	Rule + Source code / Bytecode	Limited by the rule	N/A	◐
Pattern based static analysis [22], [35], [41]–[48]	Rule + Source code / Bytecode	Limited by the rule	N/A	◐
Proof based – the goal is to prove that a set of smart contracts meet specific security properties				
Formal verification [28], [49]–[51]	Formal security properties + Source code / DApp models	Limited by the security properties	N/A	◐

Elastic Swap Attack (Dec-13-2022)

TX0 – “Attacker”
Function name: go()

TX1 – “Attacker”
Function name: go()
Propagated: P2P Network (detected at: 2022-12-13 02:32:43.238946+00)

TX2 – “Whitehat hacker”
Function name: NotYoink()
Built by: BeaverBuilder
Relayed by: BloXroute Max Profit (kudos to Toni Wahrstätter)

250 ms!

TX3 – “Whitehat hacker”
Function name: yoink()
Propagated: P2P Network (detected at: 2022-12-13 02:32:43.481679+00)

time

Elastic Swap Attack (Dec-13-2022)

Whitehat hacker capabilities

Bilingual

- “yoink” contract for transactions on the P2P network
- “No Yoink” for transactions through relayers

Generalized? Front-Running

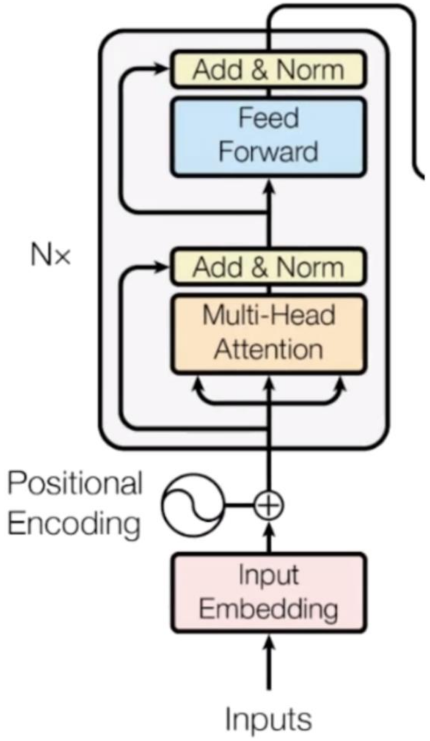
- Mimic & front-run in 250 ms!

Bribe genius

- Vulnerable 523.55 ETH
    - – 78.53 ETH (15% Bribe)
    - - **44.50 ETH (10% bounty)**

Transformer and Language Models

- LLM
  - given a sequence of tokens $x_1, \ldots, x_n$, find its likelihood:
  - $p(x_1, \ldots, x_n) = ?$
- Transformer
  - Multi-layer neural network with self-attention
  - given $x_1, \ldots, x_n$ generates a sequence of vectors, from which we compute $\log p(x_1, \ldots, x_n)$
- Pretraining
  - Maximize the log-likelihood of observed sequences of tokens: $\max \log p(x1, \ldots, xn)$



Intrusion Detection with BlockGPT

- Percentage ranking
  - Flag the least likely α%
- Absolute ranking
  - Flag the least likely k%

BlockGPT IDS Performance

Dataset Size (the total number of transactions interacting with the vulnerable smart contract)	Percentage Ranking Alarm Threshold (%) ≤ 0.01%	≤ 0.1%	≤ 0.5%	≤ 1%	≤ 10%	Absolute Ranking Alarm Threshold top-1	top-2	top-3
0 - 99 txs (32 attacks, 28% of dataset)	-	-	-	-	5 (16%)	7 (22%)	20 (63%)	23 (72%)
Average false positive rate	-	-	-	-	8.18%	0%	14.8%	28.3%
Average number of false positives	-	-	-	-	5.1	0	1	2
100 - 999 txs (38 attacks, 33% of dataset)	-	-	8 (21%)	12 (32%)	28 (74%)	7 (18%)	12 (32%)	15 (39%)
Average false positive rate	-	-	0.24%	0.71%	9.65%	0%	0.46%	0.81%
Average number of false positives	-	-	1.5	3.5	39.4	0	1	2
1000 - 9999 txs (17 attacks, 15% of dataset)	-	6 (35%)	9 (53%)	11 (65%)	13 (76%)	4 (24%)	7 (41%)	7 (41%)
Average false positive rate	-	0.054%	0.45%	0.95%	9.96%	0%	0.049%	0.098%
Average number of false positives	-	1.4	11.5	23.7	324.5	0	1	2
10000 + txs (29 attacks, 25% of dataset)	2 (7%)	7 (24%)	16 (55%)	18 (62%)	21 (72%)	2 (7%)	3 (10%)	4 (14%)
Average false positive rate	0.007%	0.097%	0.50%	1%	10%	0%	0.004%	0.008%
Average number of false positives	2.5	120.1	429.9	819.6	7302.1	0	1	2
Overall	2 (2%)	13 (11%)	33 (28%)	41 (35%)	67 (58%)	20 (17%)	42 (36%)	49 (42%)
Average false positive rate	0.007%	0.077%	0.42%	0.90%	9.71%	0%	7.19%	13.5%
Average number of false positives	2.5	65.3	211.9	367.2	2368.5	0	1	2

Case Study #1: Beanstalk (Observable Adv)

- April 2022
  - Adversary borrows 1B USD
  - Exchange proceeds for 67% stake in Beanstalks
  - Passes vote to withdraw treasury
- Observable Adversary
  - Etherscan observed the transaction 30 seconds before being mined.
- BlockGPT
  - Ranks the transaction as most abnormal among all beanstalk txs

Case Study #2: Revest (Hidden Adv)

- March 2022
  - 4 adversarial transactions over 17 minutes
  - 2M USD lost
- Hidden Adversary
  - Mined through FaaS (Flashbots)
- BlockGPT
  - Can only act as retrospective tool
  - Once the first adversarial transaction is mined
  - Could have prevented 3 out of the 4 transactions


Are attacks similar?

Bytecode Similarity Analysis 🤨

Bytecode Similarity → Ngram → Embedding → 🤨

Victim Contracts

- 100% similarity among 38
- 80% similarity among 85

Attacker Contracts

- 100% similarity among 29
- 80% similarity among 73

Adversarial and vulnerable contracts are detectable.