Detecting Flash Loan Based Attacks in Ethereum

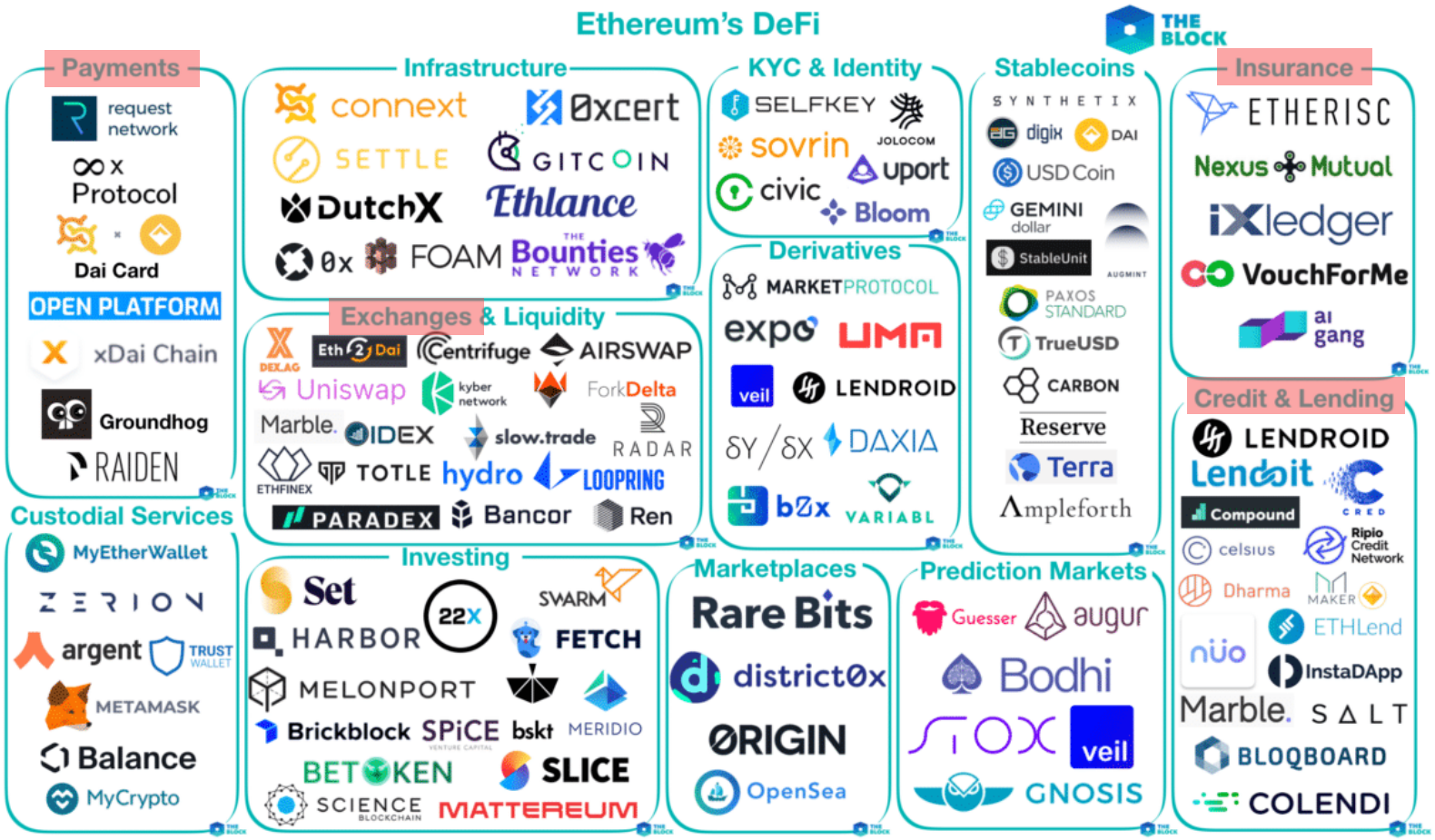
Qing Xia, Zhirong Huang, Wensheng Dou, Yafeng Zhang,
Fengjun Zhang, Geng Liang, Chun Zuo

Institute of Software, Chinese Academy of Sciences
Sinosoft Company Limited
(ICDCS 2023)

中科软科技
Sinosoft Co.,Ltd

DeFi ecosystem in Ethereum

- *DeFi (Decentralized Finance):* the blockchain-based form of finance that does not rely on centralized intermediaries.



DeFi ecosystem in Ethereum

- *DeFi (Decentralized Finance):* the blockchain-based form of finance that does not rely on centralized intermediaries.

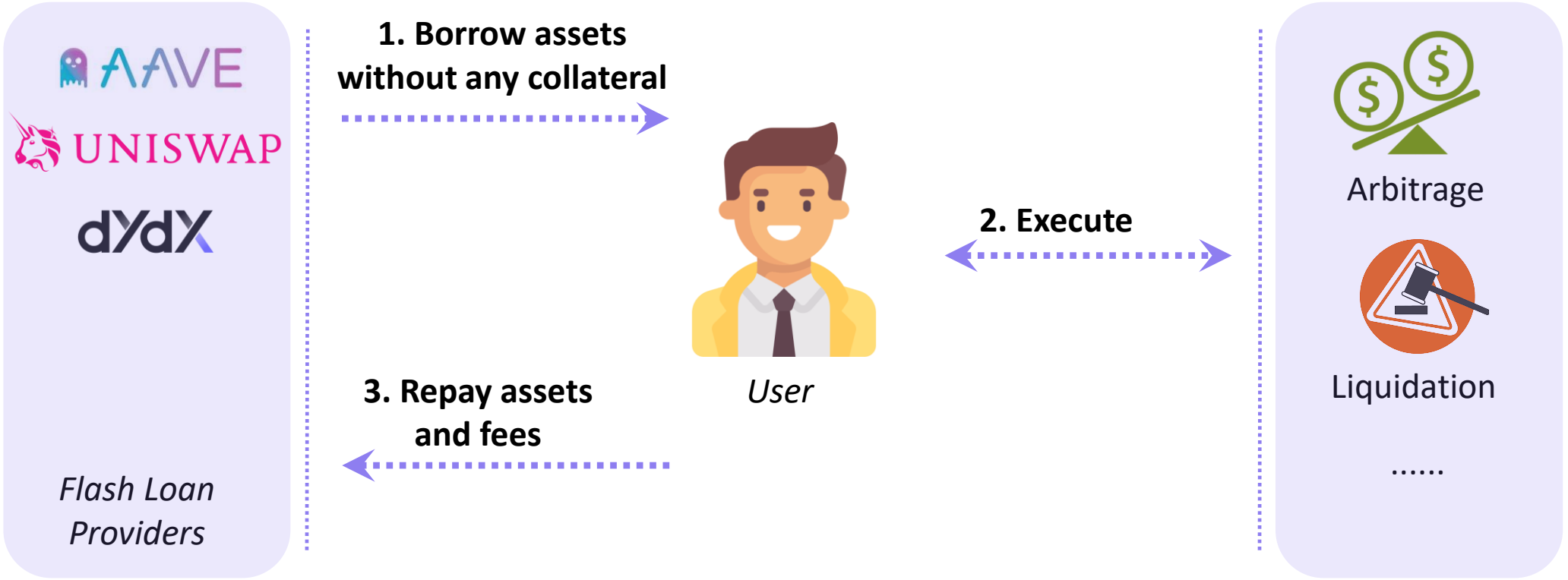
Total Value Locked

DeFi witnessed rapid growth in early 2021.
*Its market capitalization currently stabilizes at around **$40 billion**.*

https://defillama.com/

Flash loan

- Flash loan is an *uncollateralized loan* based on the *transaction atomicity*.



Flash loan

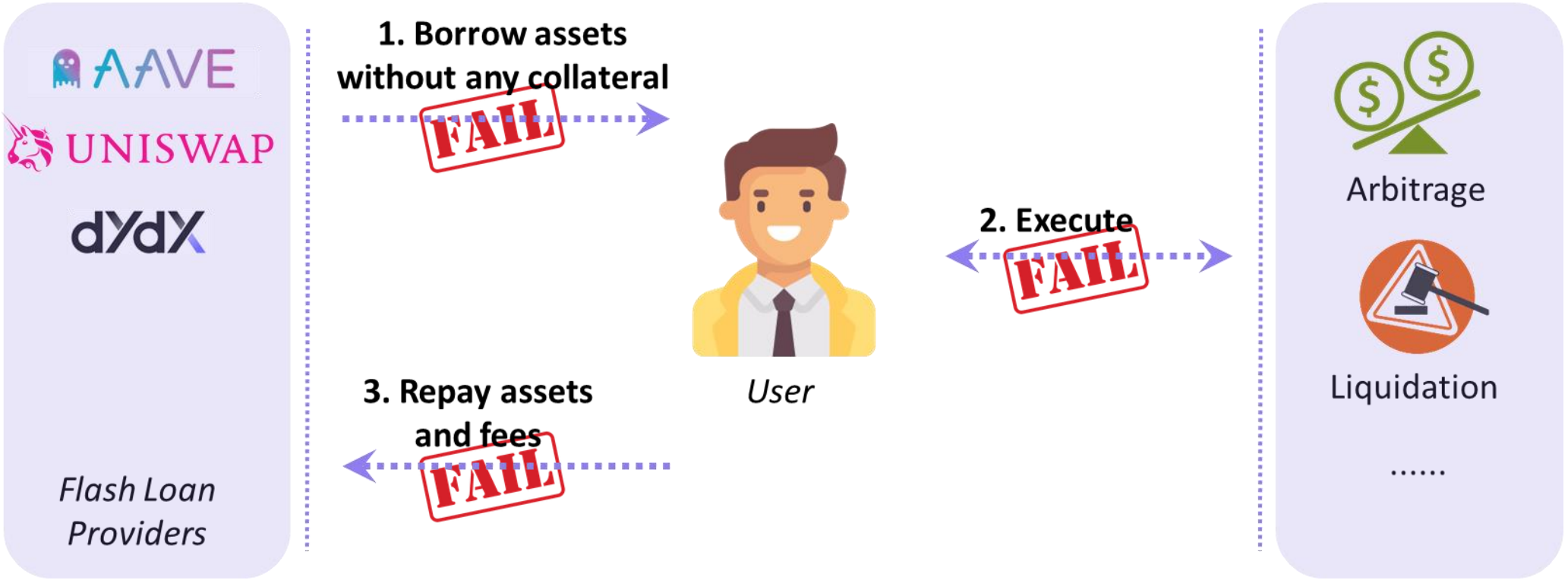
- Flash loan is an *uncollateralized loan* based on the *transaction atomicity*.

All operations are executed in a single transaction

“Flash loan transaction”

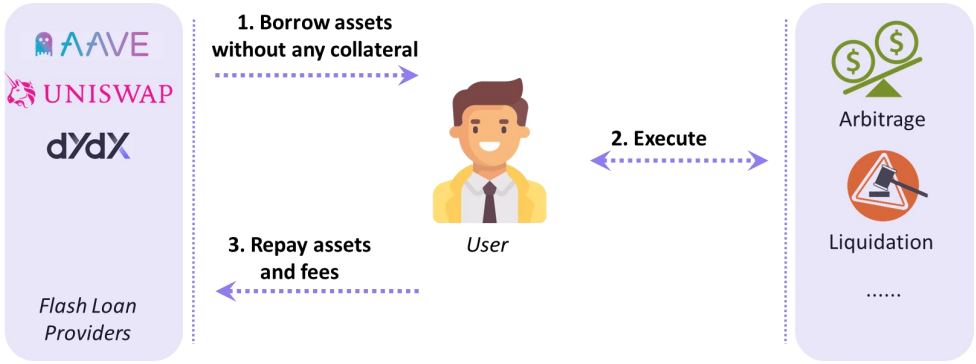
Flash loan

- Flash loan is an *uncollateralized loan* based on the *transaction atomicity*.



Flash loan

- Flash loan is an *uncollateralized loan* based on the *transaction atomicity*.

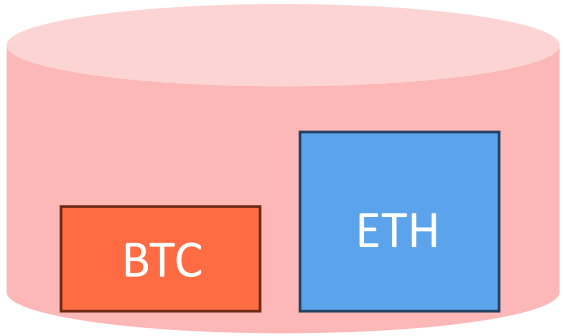


If the user repays, the transaction will be successful and the flash loan provider will earn a fee.

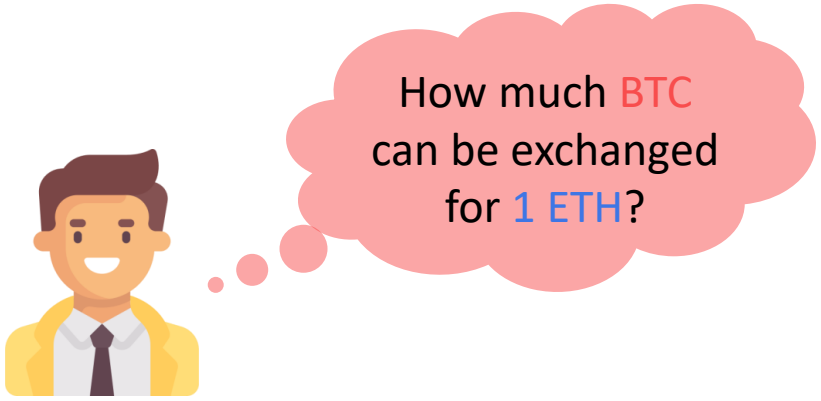
If the user fails to repay, the transaction will be aborted, but the flash loan provider will not lose anything.

Flash loan can be used to manipulate prices

- *Automated Market Makers (AMMs)* are widely used in DeFi protocols to trade assets, e.g., the constant product formula.



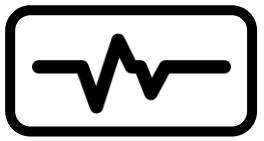
Liquidity pool



Reserve (BTC) * Reserve (ETH) = Constant

*e.g., 50 * 100 = 5000* ⇢ *(50-Δ) * (100+1) = 5000*

Δ =0.495



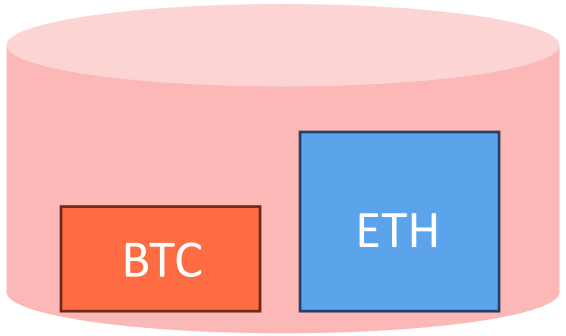
slight price slippage

Price (BTC) = Reserve (ETH) / Reserve (BTC)

e.g., price (BTC) = 2 ETH/BTC ⇢ Price (BTC) = 1/0.495 = 2.02 ETH/BTC

Flash loan can be used to manipulate prices

- *Automated Market Makers (AMMs)* are widely used in DeFi protocols to trade assets, e.g., the constant product formula.



Liquidity pool

Reserve (BTC) * Reserve (ETH) = Constant

*e.g., 50 * 100 = 5000* ⇢ *(50-Δ) * (100+100) = 5000*

Δ =25

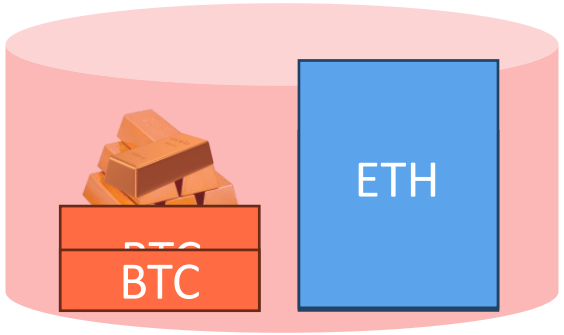
severe price slippage

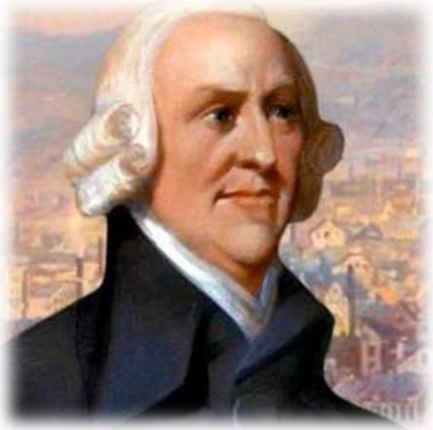
Price (BTC) = Reserve (ETH) / Reserve (BTC)

e.g., price (BTC) = 2 ETH/BTC ⇢ Price (BTC) = 100/25 = 4 ETH/BTC

Flash loan can be used to manipulate prices

- *Automated Market Makers (AMMs)* are widely used in DeFi protocols to trade assets, e.g., the constant product formula.



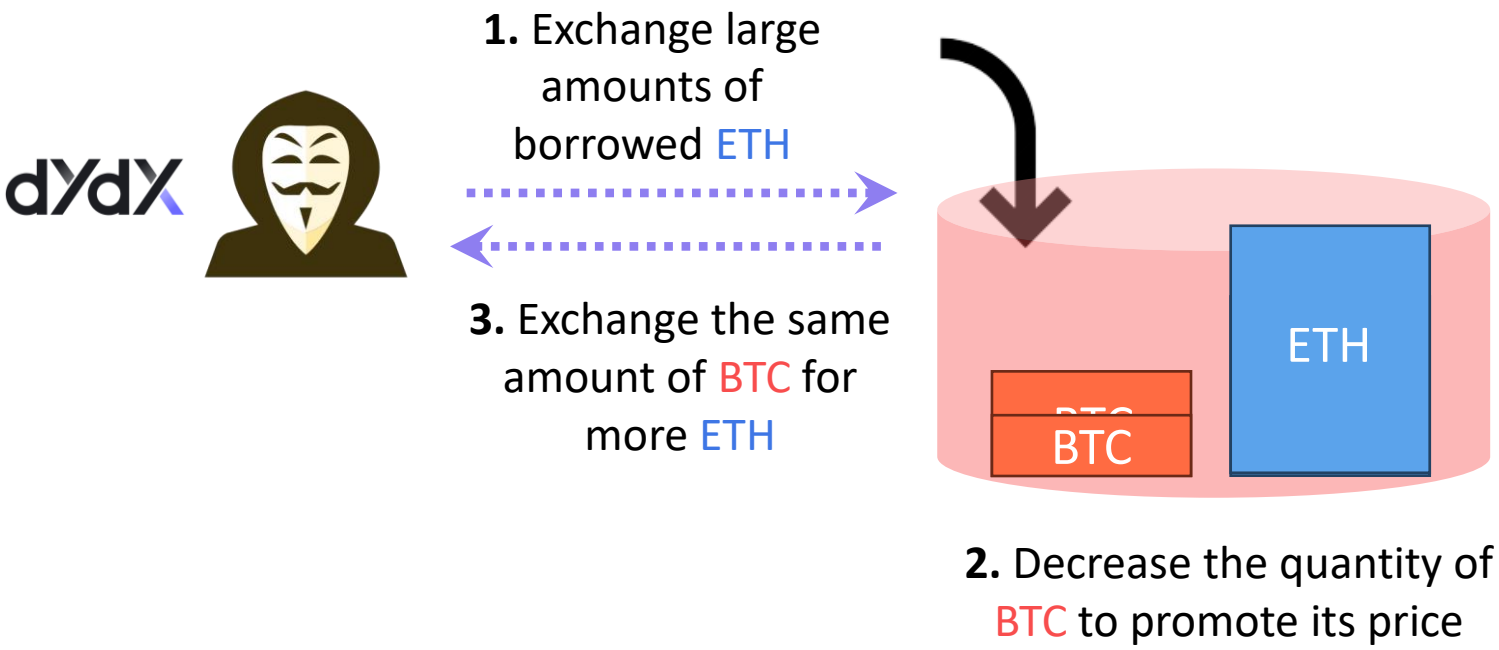


"The market price of every particular commodity is regulated by the proportion between the quantity which is actually brought to market"

----Adam Smith

Flash loan can be used to manipulate prices

- *Attackers can utilize borrowed assets from flash loans to disrupt asset prices, and make a profit.*

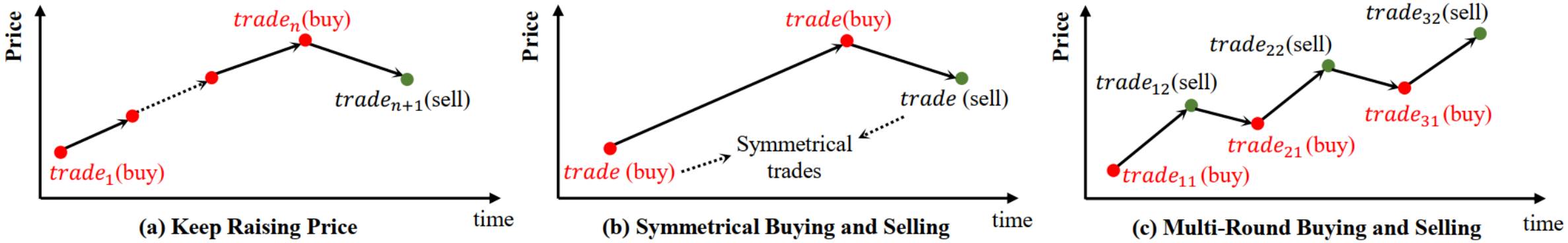


FlpAttack: flash loan based price manipulation attack

- *BZx attack on Feb 15, 2020.* https://peckshield.medium.com/bzx-hackfull-disclosure-with-detailed-profit-analysis-e6b1fa9b18fc

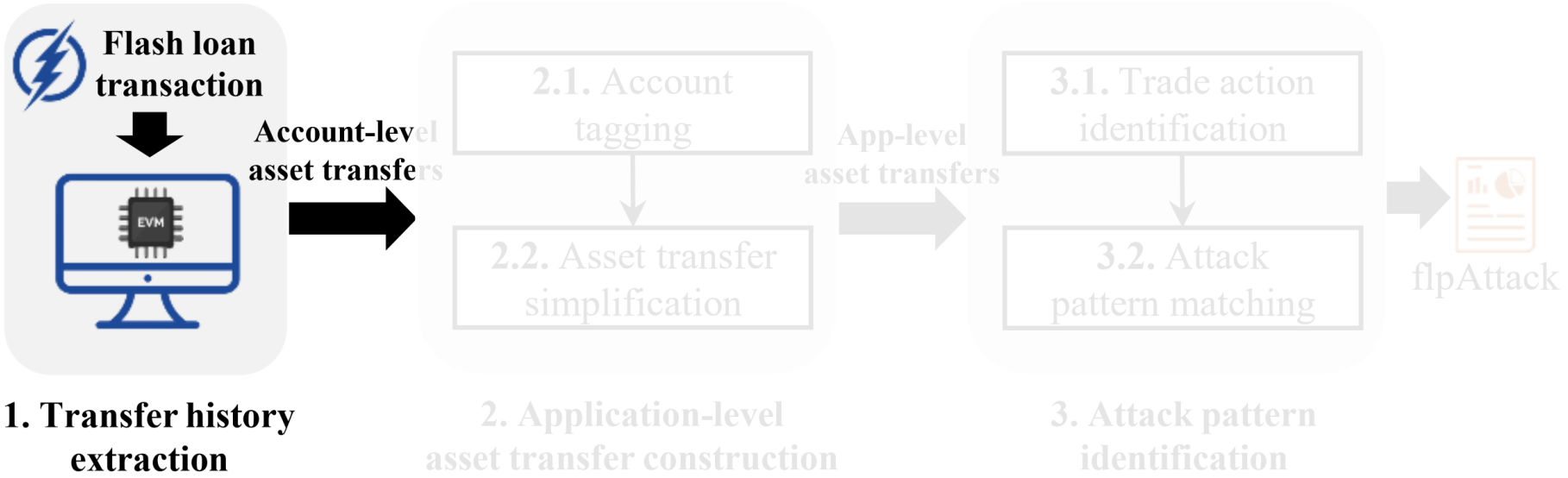
Empirical study on flash loan based attacks

- We collect *44* real-world flash loan based attacks in the past two years, including *23 attacks against Ethereum* and *21 attacks against BNB Smart Chain*.
- According to whether the attacker makes a profit from the deliberately-made price difference with flash loans, we divide these attacks into *22 price manipulation attacks* and 22 non-price manipulation attacks.
- We summarize *3 attack patterns* from 17 real-world flpAttacks.



Approach

- We propose *LeiShen* to automatically detect flpAttacks, which takes a flash loan transaction as input and returns a detailed report regarding attack patterns as output.



Transfer history extraction

- We identify flash loan transactions from *3 popular flash loan providers, i.e., Uniswap, AAVE and dYdX*.
- A flash loan transaction can be identified by its *called functions or event logs*.

Provider	Function	Event
Uniswap	swap uniswapV2Call	
AAVE	flashLoan	FlashLoan
dYdX	Operate Withdraw callFunction Deposit	LogOperation LogWithdraw LogCall LogDeposit

- We replay each flash loan transaction in a modified Geth client to obtain the *transfer history of Ether and ERC20 tokens*.

Application-level asset transfer construction

T_1 = (“0xb017”, “0x65bf”, 5,637, WETH)
T_2 = (“0x65bf”, “0x57f8”, 5,637, WETH)
T_3 = (“0xc02a”, “0x57f8”, 5,637, ETH)
T_4 = (“0x57f8”, “0x65bf”, 5,637, ETH)
T_5 = (“0x65bf”, “0x31e0”, 5,637, ETH)
T_6 = (“0x31e0”, “0x4d2f”, 5,637, ETH)
T_7 = (“0x4d2f”, “0x31e0”, 51, WBTC)
T_8 = (“0x31e0”, “0x65bf”, 51, WBTC)
T_9 = (“0x65bf”, “0x8b3d”, 51, WBTC)

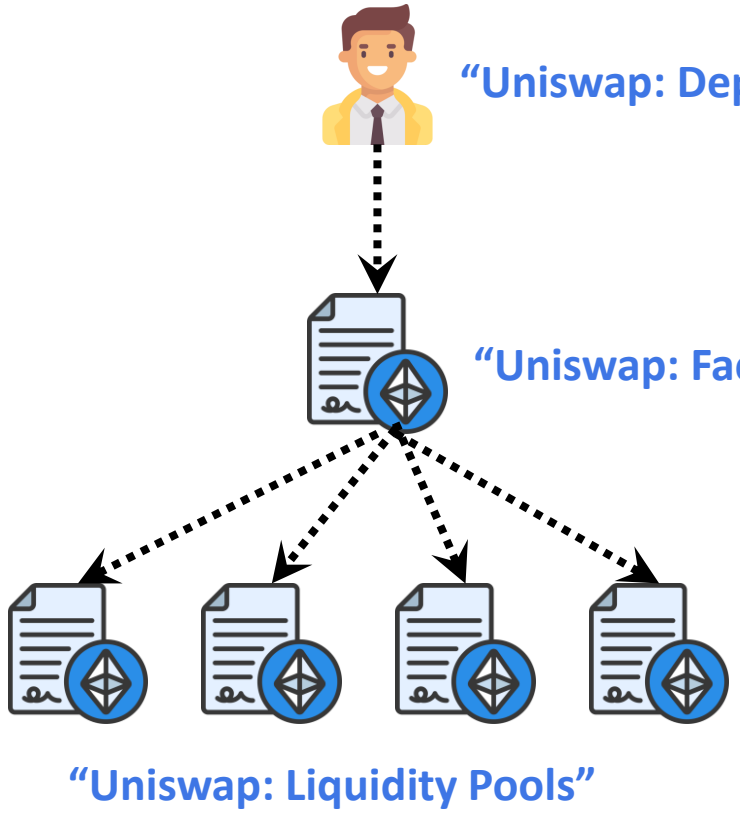
Account-level asset transfers in the bZx attack

Q1: *What is the identity of these 160-bit pseudo-anonymous address?*

Q2: *How to identify user’s trade actions and hence identify attack patterns from these complex transfers?*

Application-level asset transfer construction

- we collect *52,500 tagged accounts* of *119 DeFi applications* from Etherscan, and find that *52,482 (>99%)* tagged accounts follow the same tagging rule, i.e., *accounts with the creation relationship have the same application name tag*.



T_1 = ("0xb017", "0x65bf", 5,637, WETH)
T_2 = ("0x65bf", "0x57f8", 5,637, WETH)
T_3 = ("0xc02a", "0x57f8", 5,637, ETH)
T_4 = ("0x57f8", "0x65bf", 5,637, ETH)
T_5 = ("0x65bf", "0x31e0", 5,637, ETH)
T_6 = ("0x31e0", "0x4d2f", 5,637, ETH)
T_7 = ("0x4d2f", "0x31e0", 51, WBTC)
T_8 = ("0x31e0", "0x65bf", 51, WBTC)
T_9 = ("0x65bf", "0x8b3d", 51, WBTC)

Account-level asset transfers

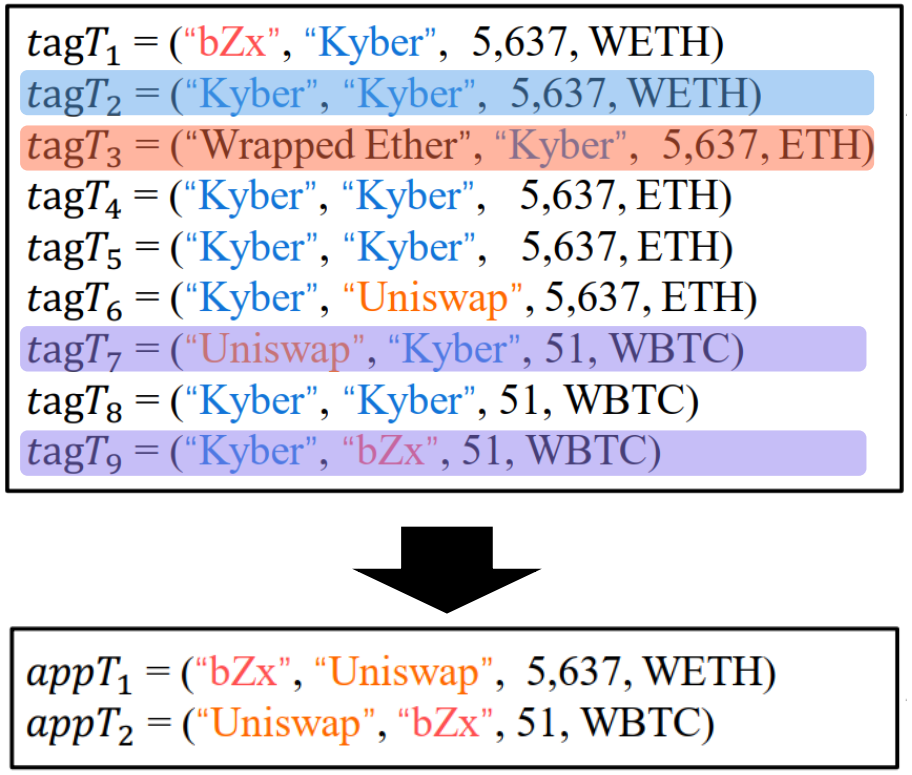
Clustering related accounts

$tagT_1$ = ("bZx", "Kyber", 5,637, WETH)
$tagT_2$ = ("Kyber", "Kyber", 5,637, WETH)
$tagT_3$ = ("Wrapped Ether", "Kyber", 5,637, ETH)
$tagT_4$ = ("Kyber", "Kyber", 5,637, ETH)
$tagT_5$ = ("Kyber", "Kyber", 5,637, ETH)
$tagT_6$ = ("Kyber", "Uniswap", 5,637, ETH)
$tagT_7$ = ("Uniswap", "Kyber", 51, WBTC)
$tagT_8$ = ("Kyber", "Kyber", 51, WBTC)
$tagT_9$ = ("Kyber", "bZx", 51, WBTC)

Application-level asset transfers

Application-level asset transfer construction

- Simplify application-level asset transfers with three heuristic rules, including *remove intra-app transfers*, *remove WETH related transfers* and *merge inter-app transfers*.



- ***Remove intra-app transfers*:** *the sender and the receiver have the same tag.*
- ***Remove WETH related transfers**: the sender or the receiver has the tag "Wrapped Ether".*
- ***Merge inter-app transfers**: in two consecutive transfers, the receiver in the first transfer and the sender in the second transfer have the same tag.*

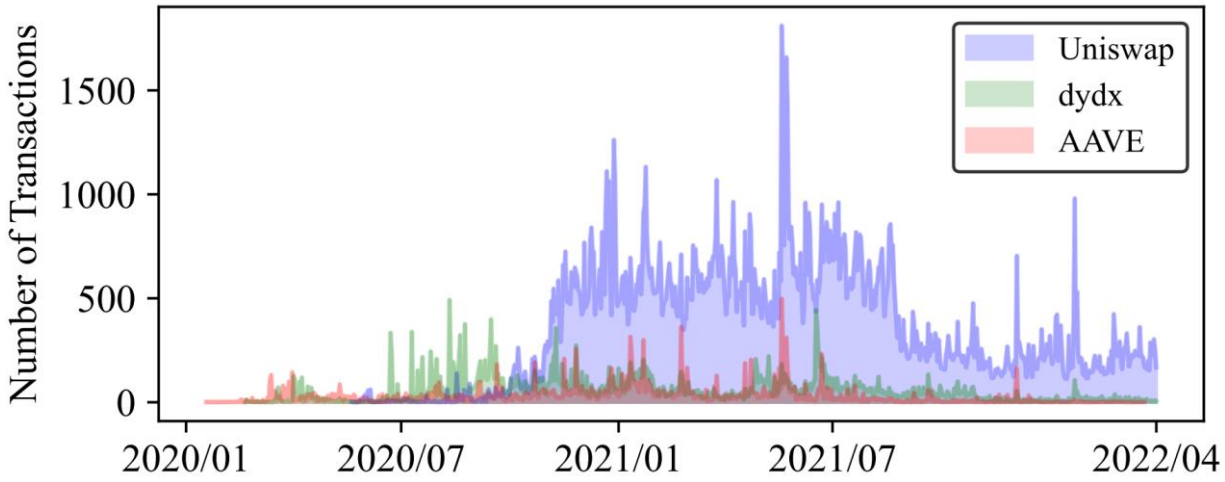
Attack pattern identification

- we identify *3 types of key trades* from application-level asset transfers.
  - In a ***swap***, a trader swaps an asset for another asset with a recipient.
  - In a ***mint liquidity***, a trader deposits assets to a DeFi application to mint new assets.
  - In a ***remove liquidity***, a trader returns minted assets to a DeFi application and takes previously deposited assets back.

- With detected trades, we check whether they conform to an attack pattern.

Evaluation

- We implemented LeiShen with *Go* language with *~5,400 LOC*, and used modified Geth (v1.10.14) to synchronize Ethereum blockchain and replay transactions.
- We filtered *272,984 flash loan transactions* from the first *14,500,000 blocks*.
- Average detection time for a flash loan transaction: 10 milliseconds
- For 75% of the transactions: < 16 milliseconds



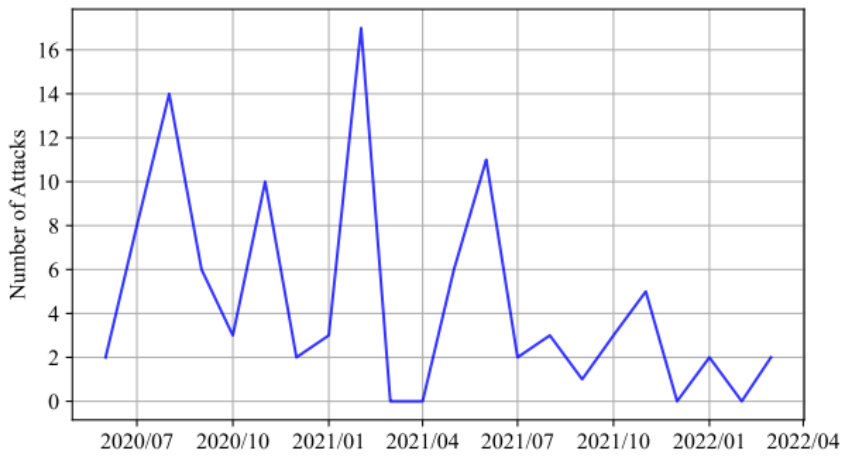
- ***Uniswap:*** *208,342*
- ***dydx:*** *41,741*
- *AAVE: 22,959*

Evaluation

- **Detecting known flpAttacks**
  - we detect *14 attacks out of 17 attacks* with patterns.
- **Detecting unknown flpAttacks**
  - among 180 detected “attacks”, *142 attacks* were verified as true positives, with an overall precision of *78.9%*.
  - 109 attacks are previously unknown.

ID	Attacks	DeFi-Ranger	Explorer+LeiShen	LeiShen
1	bZx-1			✓
2	bZx-2		✓	✓
3	Balancer		✓	✓
4	Eminence			✓
5	Harvest Finance	✓	✓	✓
6	Cheese Bank	✓		✓
7	Value DeFi	✓		
8	Yearn Finance	✓		✓
9	Spartan Protocol			✓
10	X'Token-1			
11	PancakeBunny			
12	JulSwap			
13	Belt Finance	✓		✓
14	xWin Finance	✓	✓	✓
15	Wault Finance			✓
16	Twindex			
17	AutoShark-2			✓
18	MY FARM PET			
19	PancakeHunny			
20	AutoShark-3	✓		✓
21	Ploutoz Finance	✓		✓
22	Saddle Finance	✓		✓

Attack pattern	N	TP	FP	$P(\%)$
KRP	21	21	0	100%
SBS	79	68	11	86.1%
MBS	107	60	47	56.1%



Analyzing unknown flpAttacks

- ***Attacked Applications***
  - Top3 attacked applications are decentralized exchanges.
- ***Attackers' behaviors***
  - After finishing an attack, some attackers call the self-destruct function to remove the storage from the Ethereum state.
  - Almost all attackers transfer their attack profit with the method of money laundering, such as multi-level intermediary accounts and mixing services.
- ***Profitability***
  - The maximum yield rate and net profit is $2.2 * 10^5\%$ and >$6M.
  - All attacks result in a total profit of ***>$21.8M***.

Attacked applications	Attacks	Attackers	Attack contracts	Attacked assets
Balancer	31	5	14	13
Uniswap	16	6	8	5
Yearn	11	1	1	1

	Yield rate (%)	Net profit ($)
Mean	0.3%	3,509
Min.	0.003%	23
Max.	$2.2 * 10^5\%$	6,102,198
TOP 10% in AVG	$5.7 * 10^4\%$	257,078
TOP 20% in AVG	$3.0 * 10^4\%$	135,522

Conclusion

- We conduct an empirical study on real-world flash loan attacks in the past two years and present three attack patterns to reveal attackers' behaviors in flpAttacks.
- We propose an *LeiShen* to automatically detect flpAttacks conforming to the three attack patterns by utilizing asset transfers of a flash loan transaction.
- We identify a total of 142 true attacks in the first 14,500,000 blocks in Ethereum, including 109 previously-unknown attacks.

LeiShen is the god in charge of thunder and lightning in ancient Chinese mythology.

(*https://github.com/tony4paper/LeiShen*)

Q&A

THANK YOU!