black hat ASIA 2021
May 6-7, 2021
BRIEFINGS

Disappeared Coins: Steal Hashrate in Stratum Secretly

Xin Liu, Rui Chong, Yuanyuan Huang, Yingli Zhang, Qingguo Zhou

Content

01

Mining and Pool

Mining pool

What's the proof of work(PoW)

Introduce the concept of transaction and coinbase in Blockchain

02

Stratum Protocol

Stratum V2 is rarely used so we still aim at Stratum V1

The features and communication process in Stratum V1

03

Steal Hashrate in Stratum Secretly

A direct job insertion attack model proposed by others before and the reason why the attack failed.

Purpose of our attack models

Job injection based on set_extranonce attack model

Time segment attack model

04

Proof of Concept

Tools and equipment we used for PoC

Shows the execution of attack scripts of the two model through

screenshots and videos

1 Mining and Pool

Minging Pool

It is already difficult for a single independent miner to stably mine a block to obtain revenue, so many miners combine their computing power in the form of cooperative mining to establish a mining pool.

Assign Jobs

According to the computing ability of miners, each miner is assigned a share target with different difficulty (which is far less than the bitcoin network difficulty)

Distribute Profit

Every miner who joins the mining pool can get profit according to the contribution to the mining pool.

Settlement Strategy

PPLNS (Pay Per Last N Shares), PPS (Pay Per Share), PPS+ (Pay Per Shares Plus), FPPS (Full Pay Per Shares)

1 Mining and Pool

Proof of Work

Nakamoto consensus:

every miner try to figure out the solution of the Bitcoin puzzle. If someone finds out the solution, the block he mined will be the next block in blockchain after other participants verify that his block is valid.

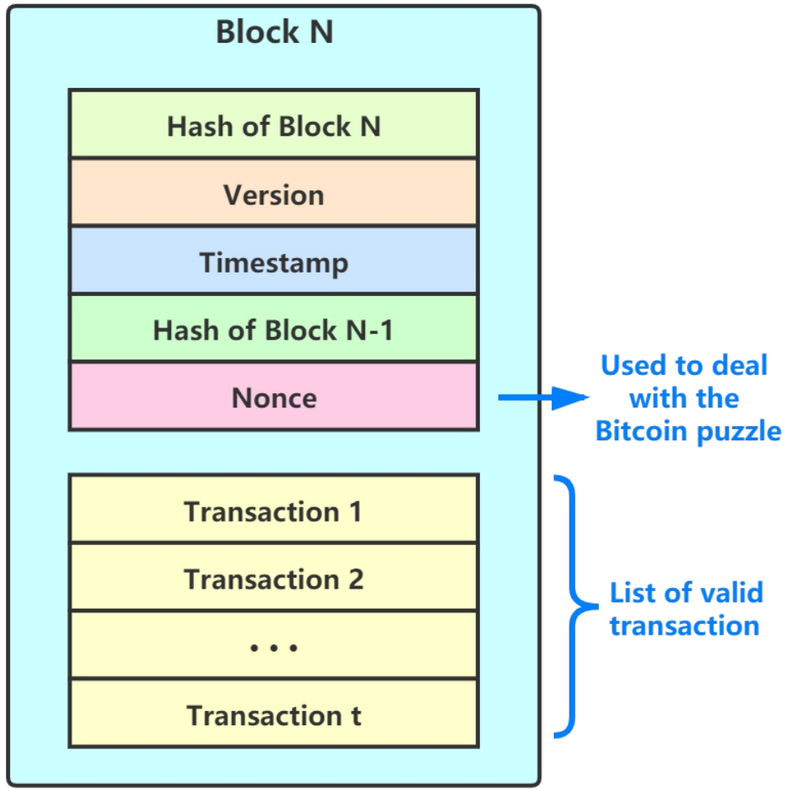
Proof of Work

1 Mining and Pool

Transaction

All transactions need to be verified whether they are valid by *miners*, then the valid transactions are packaged into a block.

And the new block is generated through proof of work(PoW) and added to the blockchain.

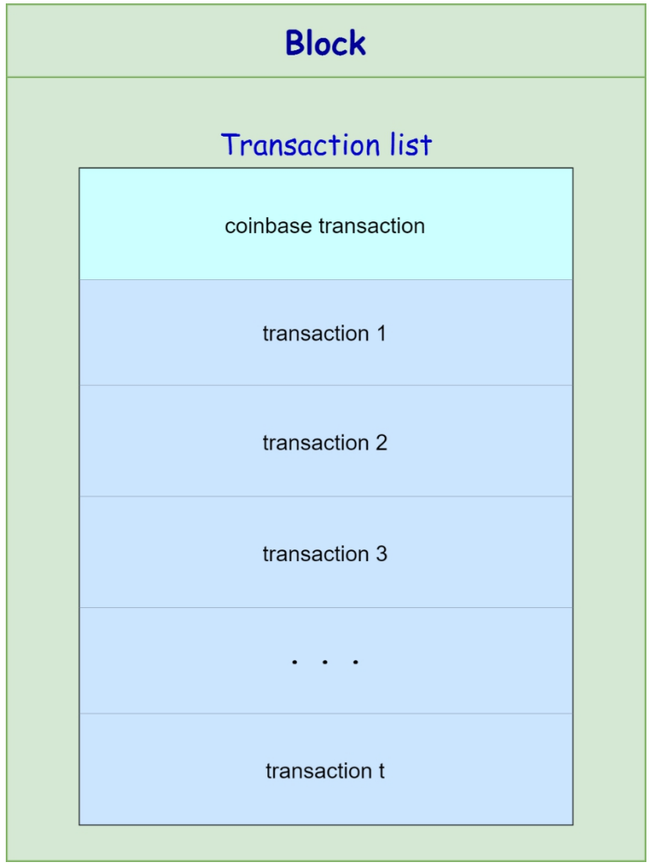


1 Mining and Pool

Transaction

Each block will collect a set of pending transactions in the Bitcoin network. But the first transaction in each block is special, called *coinbase* transaction.

It is used to specify that when the current block is mined through the proof of work, the miner will receive the corresponding block reward (currently 6.25 BTC) and fees.



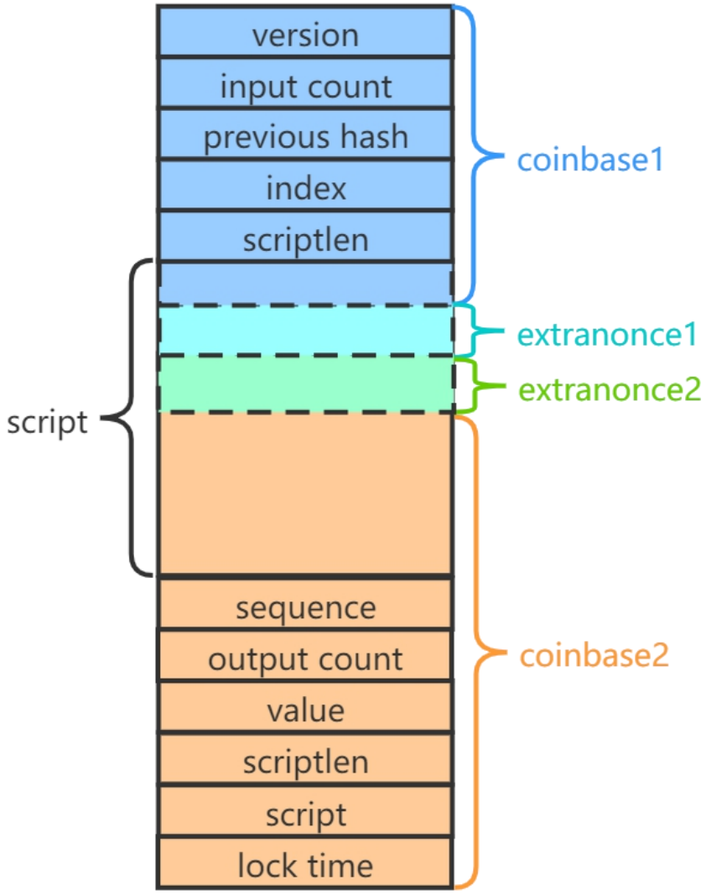
1 Mining and Pool

Coinbase Transaction

The coinbase transaction can be divided into 4 parts: coinbase1, extranonce1, extranonce2, coinbase2

Coinbase1: covers the first 5 fields of coinbase transaction and part of the script. Except for the version number, other fields are meaningless for all mining machines, because there is no input party for the coinbase transaction.

Extranonce1: covers a small part of the script in the coinbase transaction. It is specified by the mining pool and is unique for each connection between the mining pool and the mining machine.



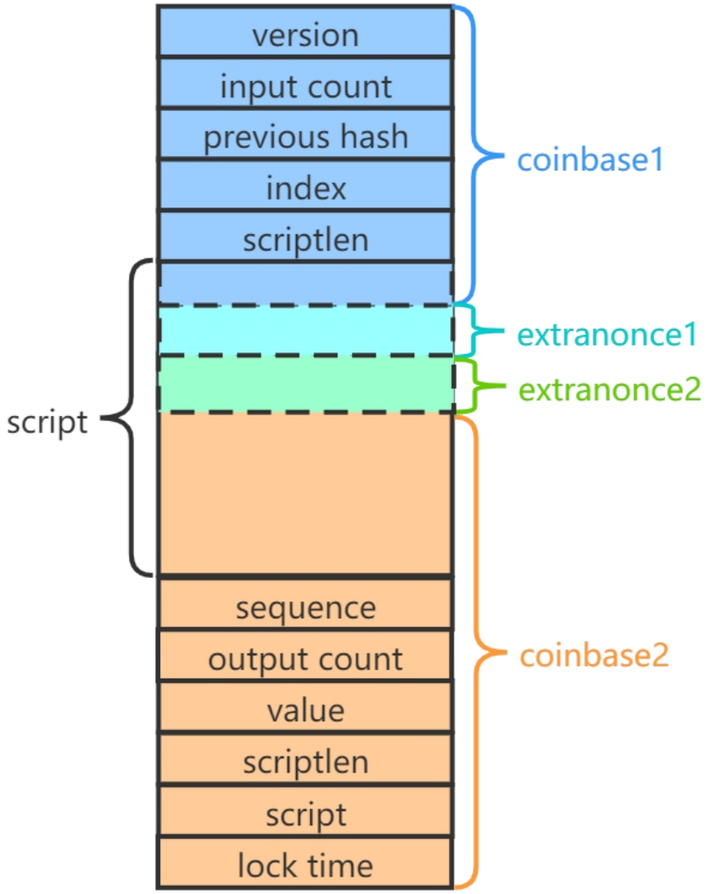
1 Mining and Pool

Coinbase Transaction

The coinbase transaction can be divided into 4 parts: coinbase1, extranonce1, extranonce2, coinbase2

Extranonce2: covers a small part of the script in the coinbase transaction, immediately after extranonce1. When miner solves Bitcoin puzzle with proof-of-work, if the *nonce* is exhausted, extranonce2 will be incremented to figure out the puzzle.

Coinbase2: covers the rest of coinbase transaction



2 Stratum Protocol

The communication between miners and mining pool needs to follow mining protocols, such as Stratum protocol, GetBlockTemplate protocol and GetWork protocol.

Stratum protocol is widely used, and Braiins put forward an upgraded version at the end of 2019, called Stratum V2.

Stratum protocol is based on TCP/IP plaintext transmission protocol, using JSON-RPC data format.

2 Stratum Protocol

Mining Pool	Stratum V1	Stratum V2
F2Pool	✓	×
Poolin	✓	×
Btc.com	✓	×
AntPool	✓	×
SlushPool	✓	✓

The Stratum V2 has not yet been widely used and we don't have the corresponding mining machine which supports Stratum V2. Mainstream mining machine (e.g. Antminer and Whatsminer) and mining software (e.g. Cgminer and Ccminer) currently do not support Stratum V2.

Since Stratum V2 has not yet been widely popularized, our attack models are only aimed at Stratum V1.

Next, the communication process of Stratum V1 will be introduced in detail.

2 Stratum Protocol

Step 1.1: First, miner initiates a subscription request to the mining pool to establish a connection through the *mining.subscribe* method

mining.subscribe("user agent/version", "extranonce1")

Step 1.2: After receiving the above subscription message, the mining pool will return the subscription_id, extranonce1 and extranonce2_size (unit: byte)

2 Stratum Protocol

Step 2.1: Miners use the *mining.authorize* method to send authorization request to the mining pool

mining.authorize("username", "password")

Step 2.2: mining pool returns true or false to notify miner whether authorization is successful

2 Stratum Protocol

Step 2.3: After that, the miner will use the *mining.extranonce.subscribe* method to inform the mining pool that it supports the *mining.set_extranonce* method

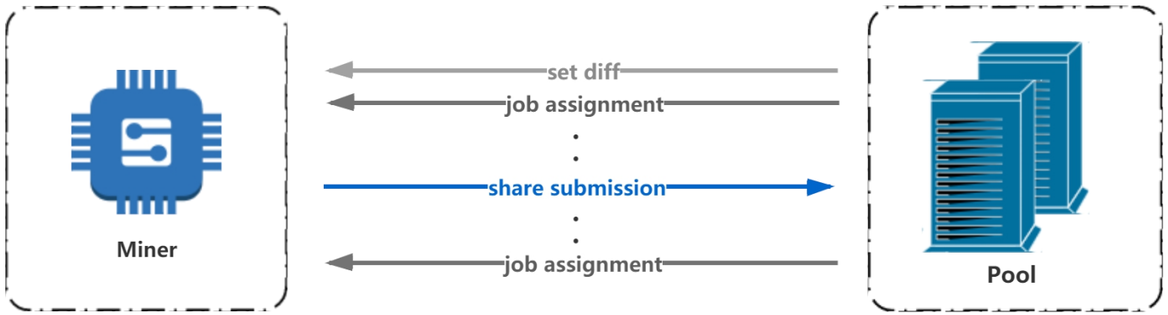
mining.extranonce.subscribe()

2 Stratum Protocol

Step 3.1: After successful subscription and authorization, the pool will negotiate the difficulty value with miner through *mining.set_difficulty* method.

The difficulty needs to be compatible with the computing power of mining machine.

Step 3.2: Mining Pool assigns jobs to miner through *mining.notify* method.



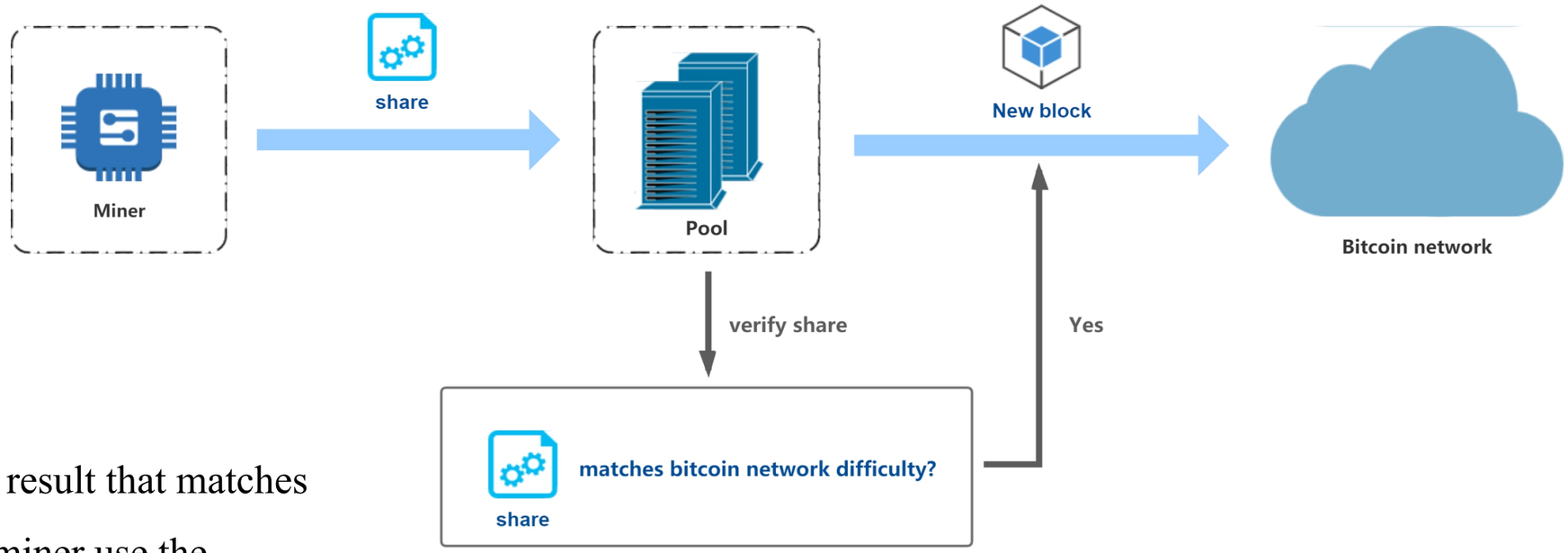
2 Stratum Protocol

mining.set_difficulty(difficulty)

mining.notify(Job ID, Hash of previous block, coinbase1, coinbase2, List of merkle branches, version, nBits, nTime, Clean Jobs)

- Job ID: the id number of job
- Hash of previous block: used to build the block header
- coinbase1, coinbase2: two unchangeable parts in coinbase transaction
- List of merkle branches: used to build the final merkle root
- Version: bitcoin block version
- nBits: the encoded bitcoin network difficulty
- nTime: the current time
- Clean Jobs: if true, it means that the miner needs to stop the current work and execute this new work

2 Stratum Protocol



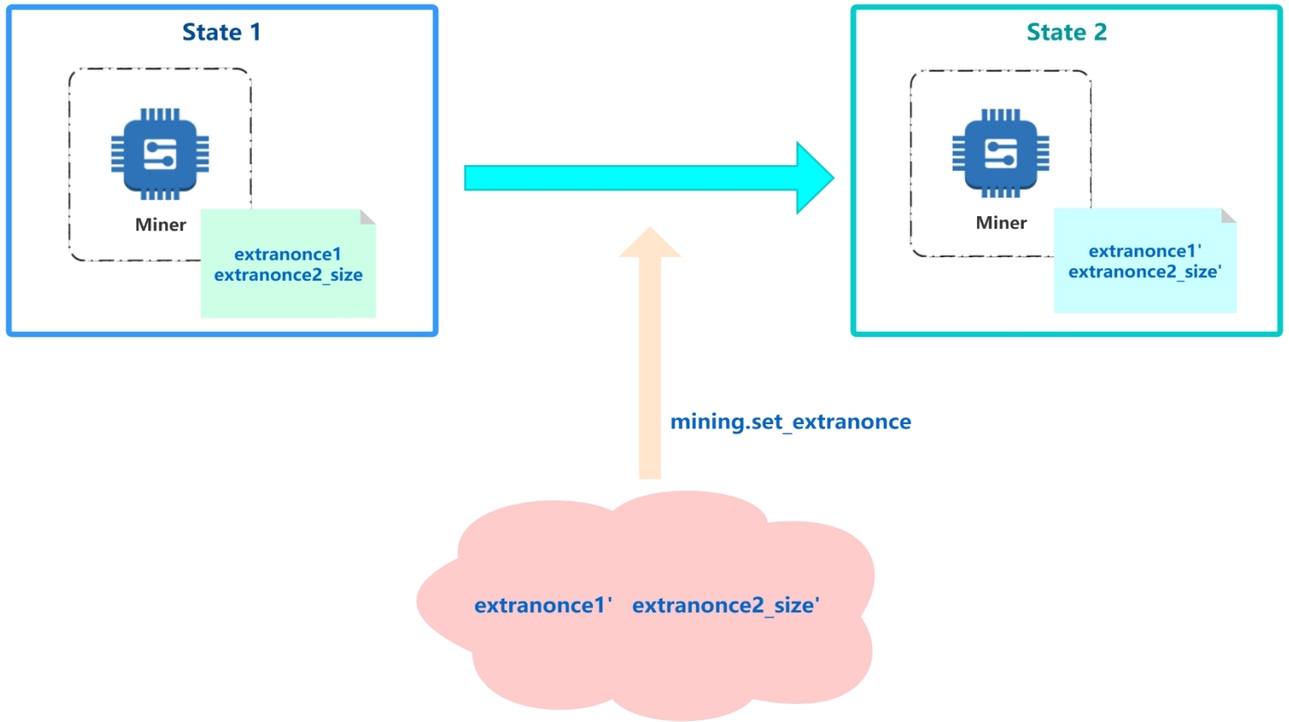
Step 4.1: When a miner finds out a result that matches the difficulty that mining pool set, miner use the *mining.submit* method to submit the result, which called *share*.

mining.submit("username", "Job ID", "extraNonce2", "nTime", "nonce")

2 Stratum Protocol

Step 5.1: using *mining.set_extranonce* method reset the value of extrance1 and bytes of extraonce2 saved by miner during the subscription phase (step 1.2)

mining.set_extranonce("extranonce1", extranonce2_size)

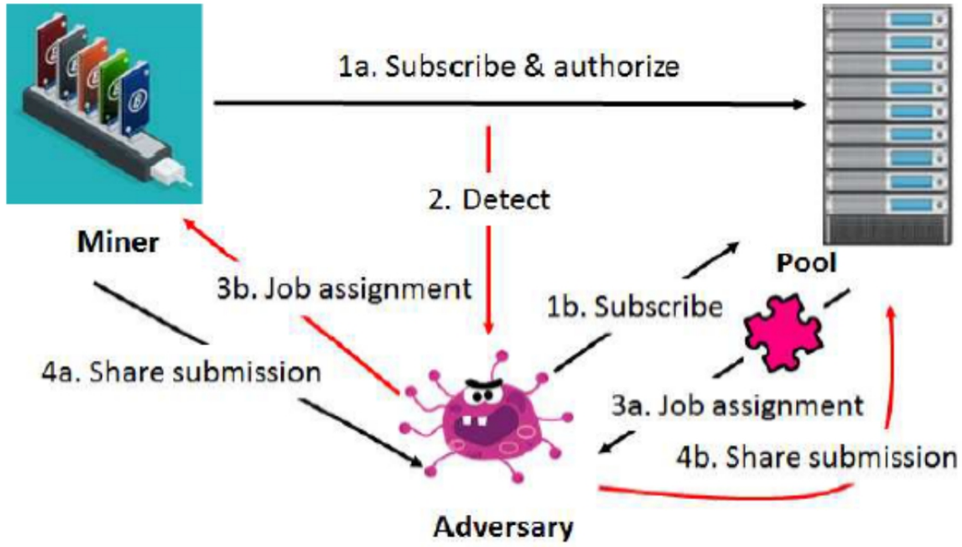


3 Steal Hashrate in Stratum Secretly

3.1 Why direct job insertion is not feasible

Direct job insertion based on TCP hijacking is not feasible in our situation.

The adversary hijacks the communication between the miner and the normal mining pool, and inserts the job of the malicious mining pool built by the adversary directly into the job flow between the normal mining pool and the miner.



Direct job insertion attack model proposed by others before

3 Steal Hashrate in Stratum Secretly

3.1 Why direct job insertion is not feasible

Miner will save extranonce1 specified by normal mining pool during subscription in Stratum. And extranonce1 is used to construct *coinbase* and calculate the *share*.

The extranonce1 specified by normal pool must be different from the one specified by malicious mining pool.

extranonce1_normal != extranonce1_malicious

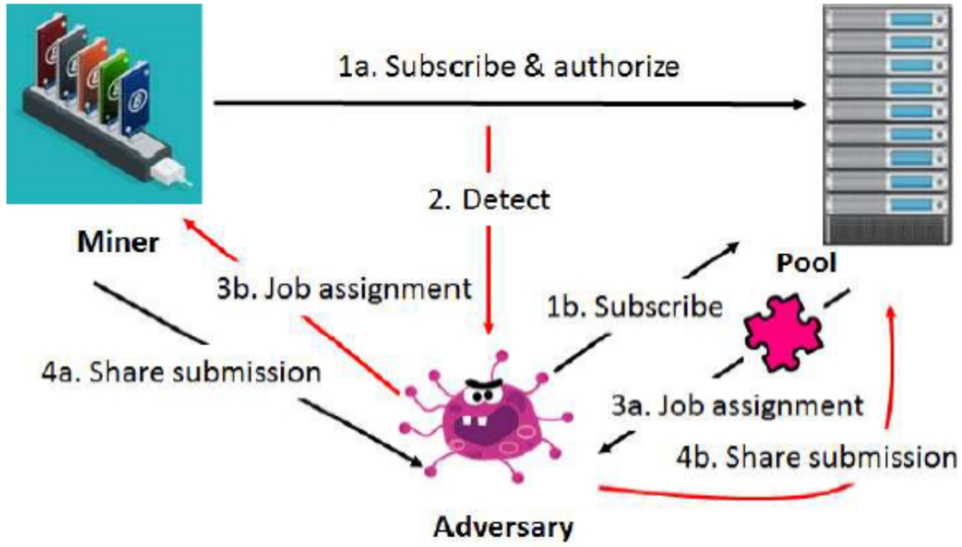
So that the *share* calculated by miner based on extranonce1_normal is different from the one calculated with extranonce1_malicious

3 Steal Hashrate in Stratum Secretly

3.1 Why direct job insertion is not feasible

Therefore, the *share* belonging to the malicious pool however calculated with extranonce1_normal will be rejected by the malicious pool in verification for *share* in Stratum.

That's the reason why the attack cannot be successfully implemented.



Direct job insertion attack model proposed by others before

3 Steal Hashrate in Stratum Secretly

3.3 Preconditions of attack

The purpose of the two man-in-the-middle attacks we proposed is to steal the hashrate of the miners to work for the malicious mining while the miner and the normal mining pool are barely aware of it.

Key: FIX extranonce1

Malicious pool: the bitcoin mining pool built by the adversary

Normal pool: the bitcoin mining pools maintained by company

3 Steal Hashrate in Stratum Secretly

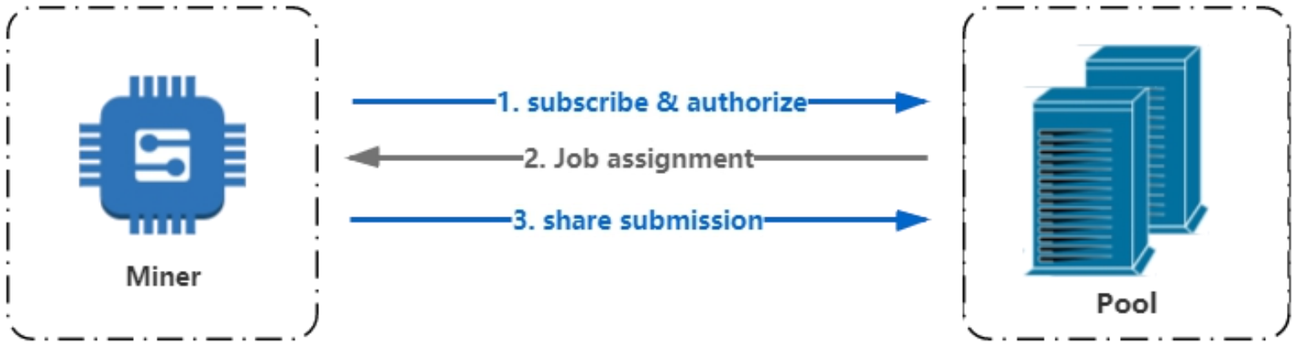
3.2 Preconditions of attack

Under normal conditions:

the miner sends subscription and authorization requests to the mining pool.

Once subscription and authorization is successful, the mining pool will allocate work to the miner.

When the miner finds out the *share*, it will submit it.

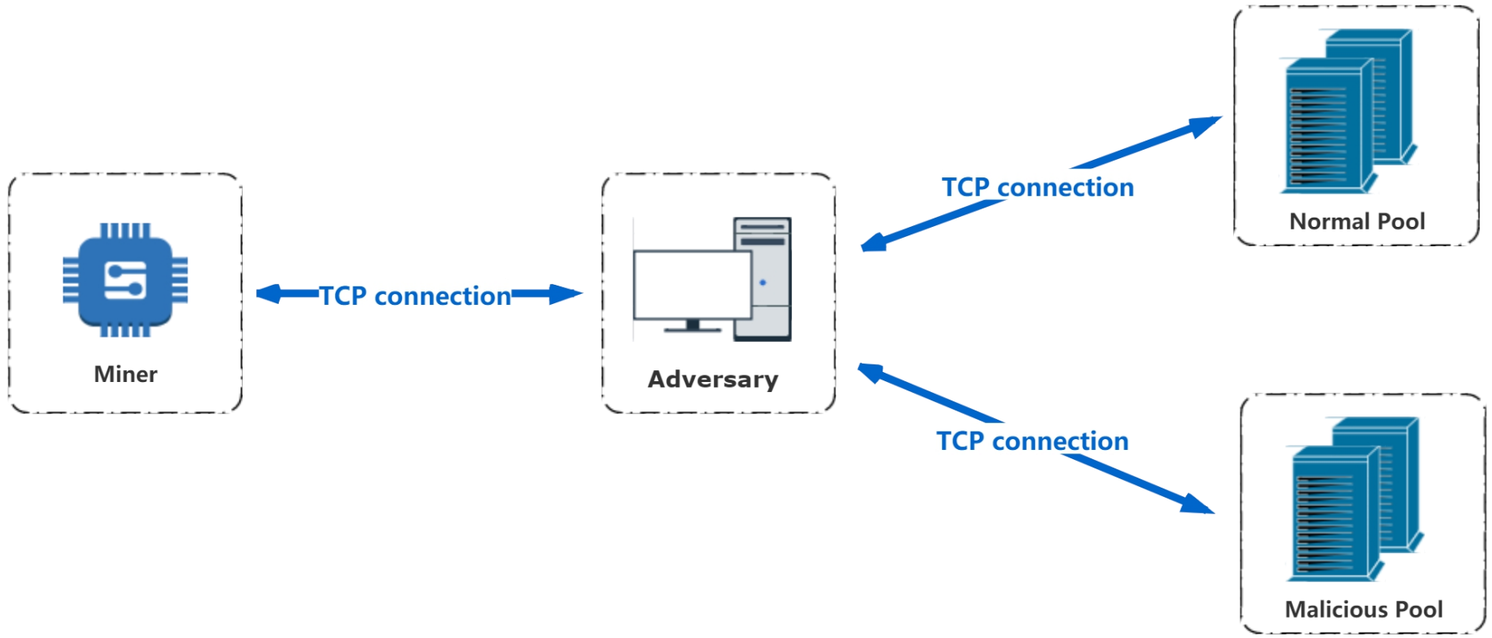


3 Steal Hashrate in Stratum Secretly

3.2 Preconditions of attack

Using man-in-the-middle attack to hijack the communication between miners and mining pools:

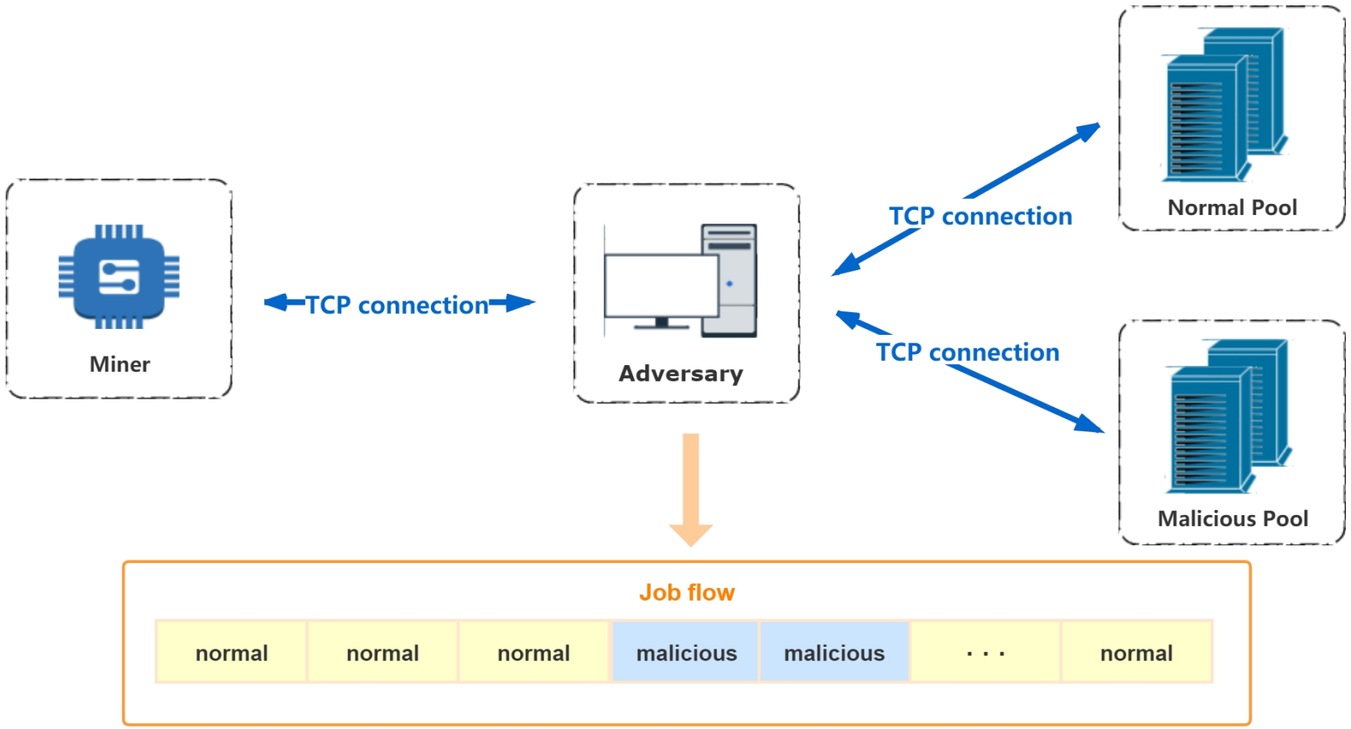
The adversary server hijacks the TCP communication between the miner and the mining pool and then maintains the TCP connections with the miner and the mining pools.



3 Steal Hashrate in Stratum Secretly

3.3 Job injection based on set_extranonce

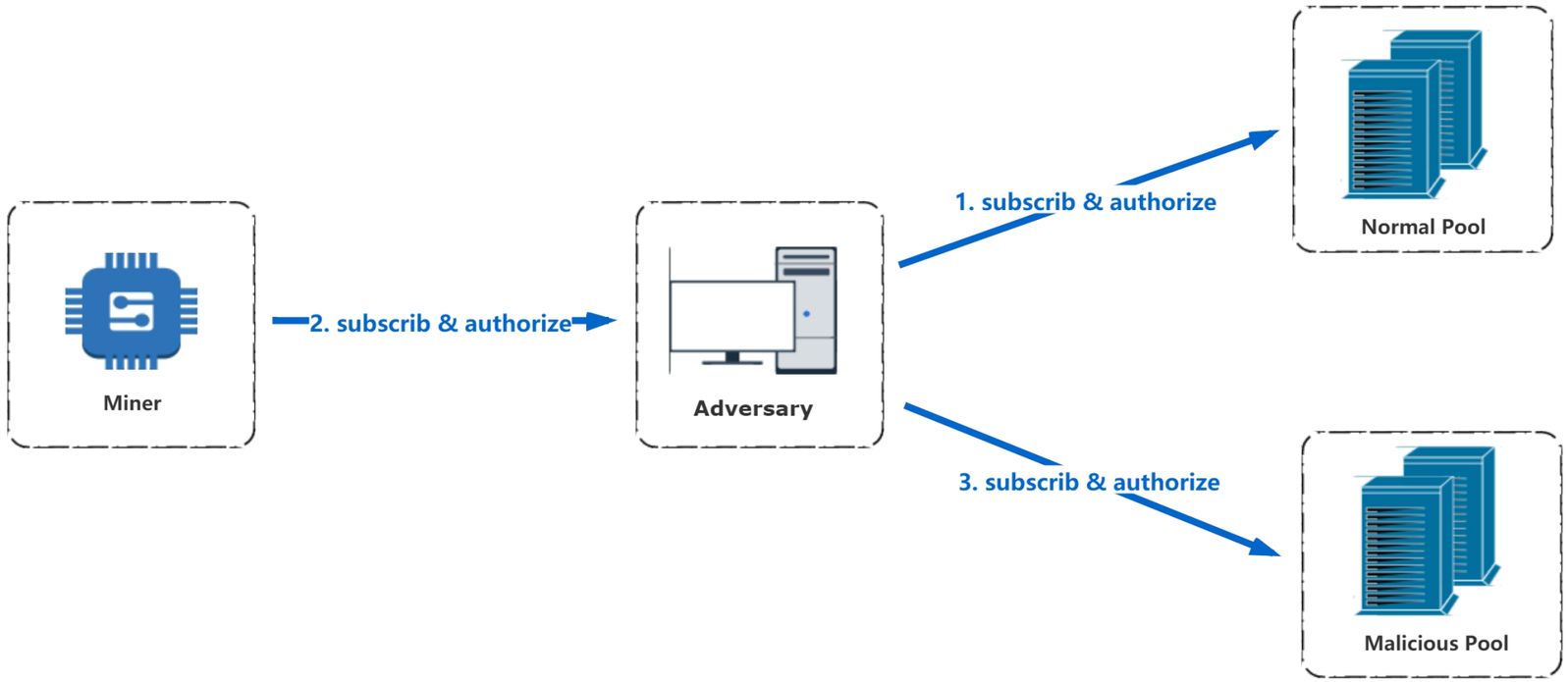
The adversary inserts the job from the malicious pool into the job flow of the normal pool.



3 Steal Hashrate in Stratum Secretly

3.3 Job injection based on set_extranonce

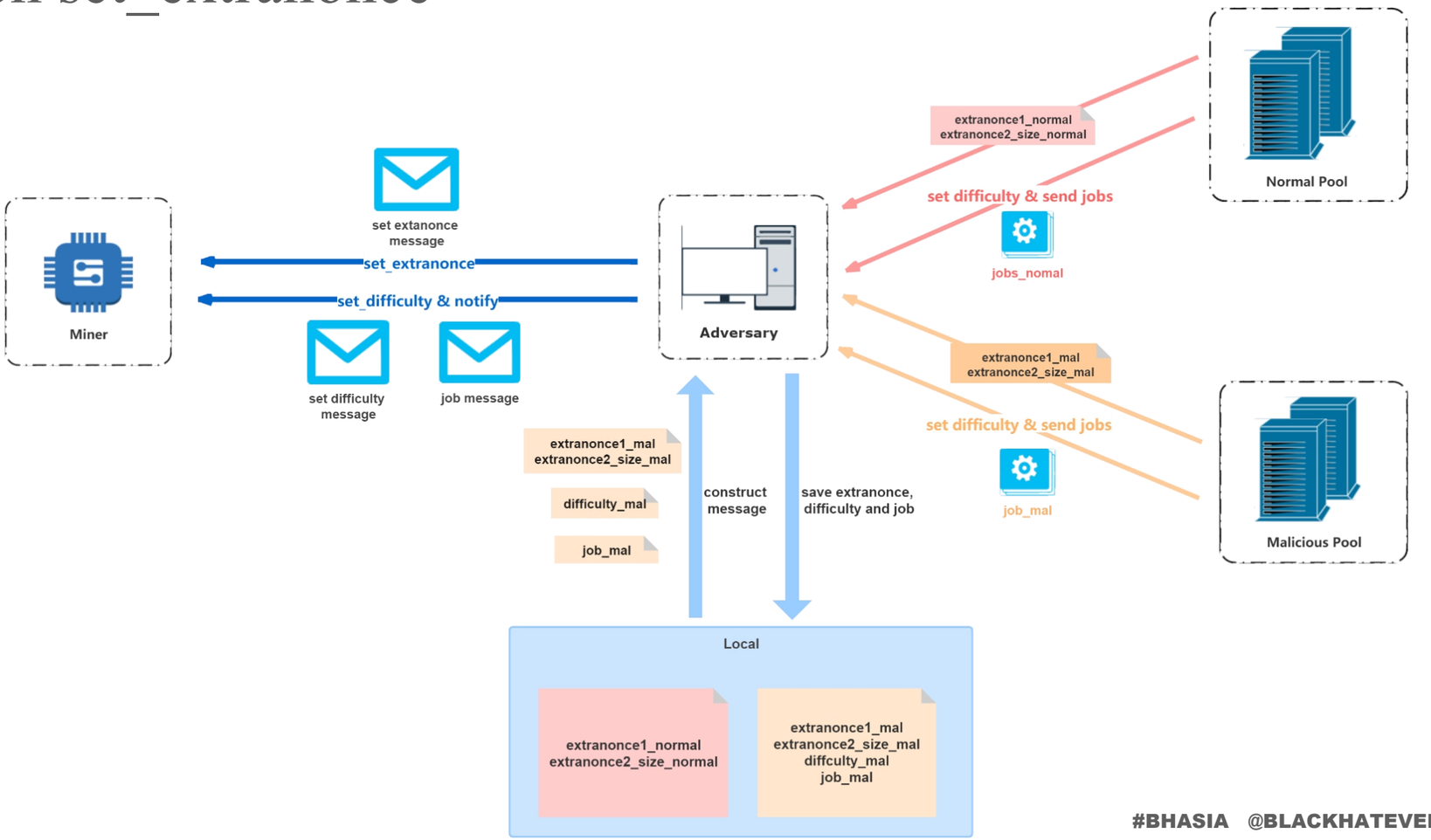
The adversary establishes a connection with the malicious mining pool, and then listens to the subscription and authorization message of the miner and forwards them to the normal mining pool.



3 Steal Hashrate in Stratum Secretly

3.3 Job injection based on set_extranonce

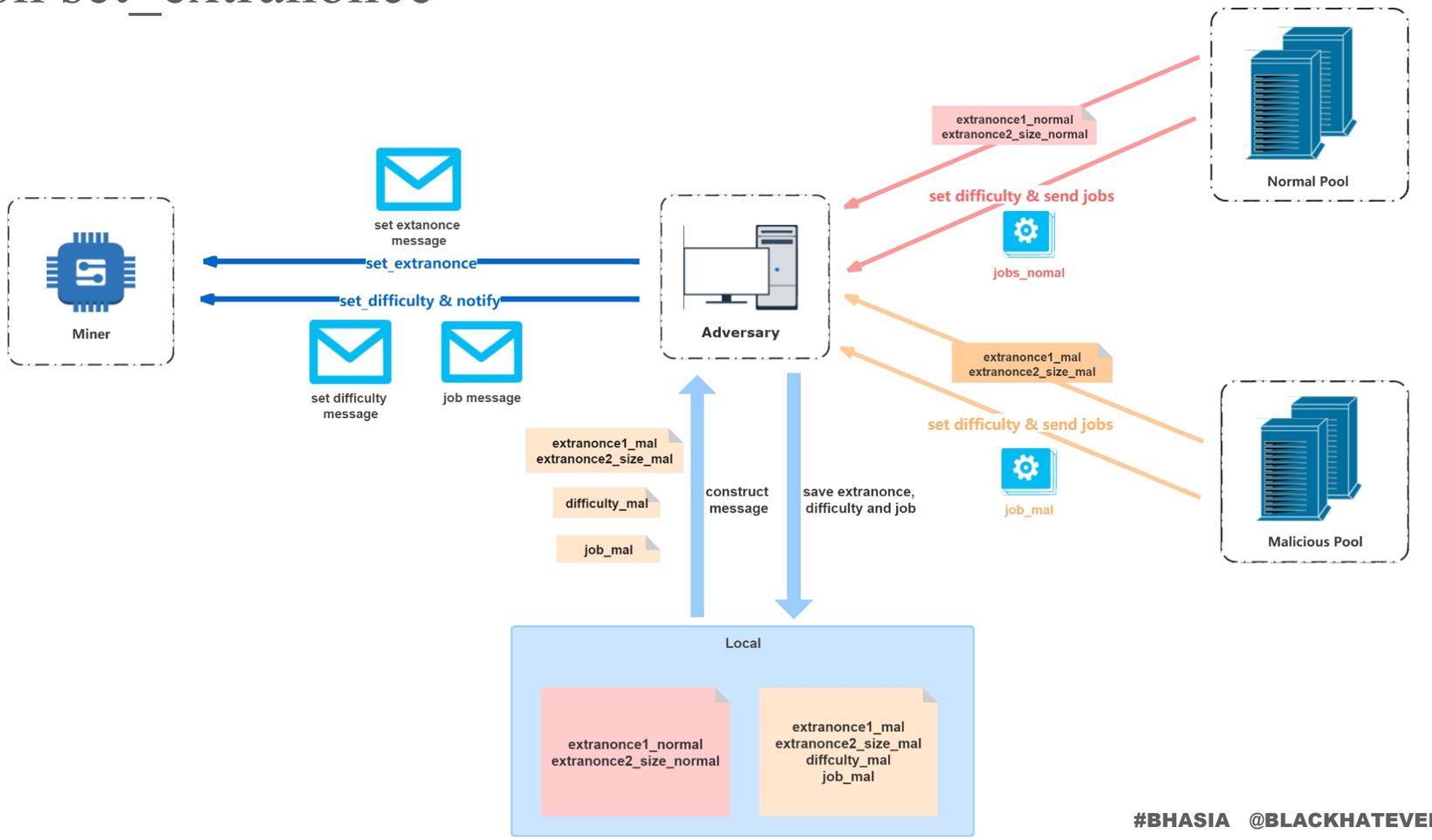
After receiving the subscription message, the mining pools will respond to miner with extranonce1 and extranonce2_size which will be saved by the adversary. What's more, the adversary will save the difficulty and job information sent by the mining pools.



3 Steal Hashrate in Stratum Secretly

3.3 Job injection based on set_extranonce

When the miner works for the normal pool for a period of time, the adversary will reset extranonce with extranonce1 and extranonce2_size of the malicious pool, and construct *set_difficulty* and *notify* message, and then sends them to miner to make miner work for malicious pool.

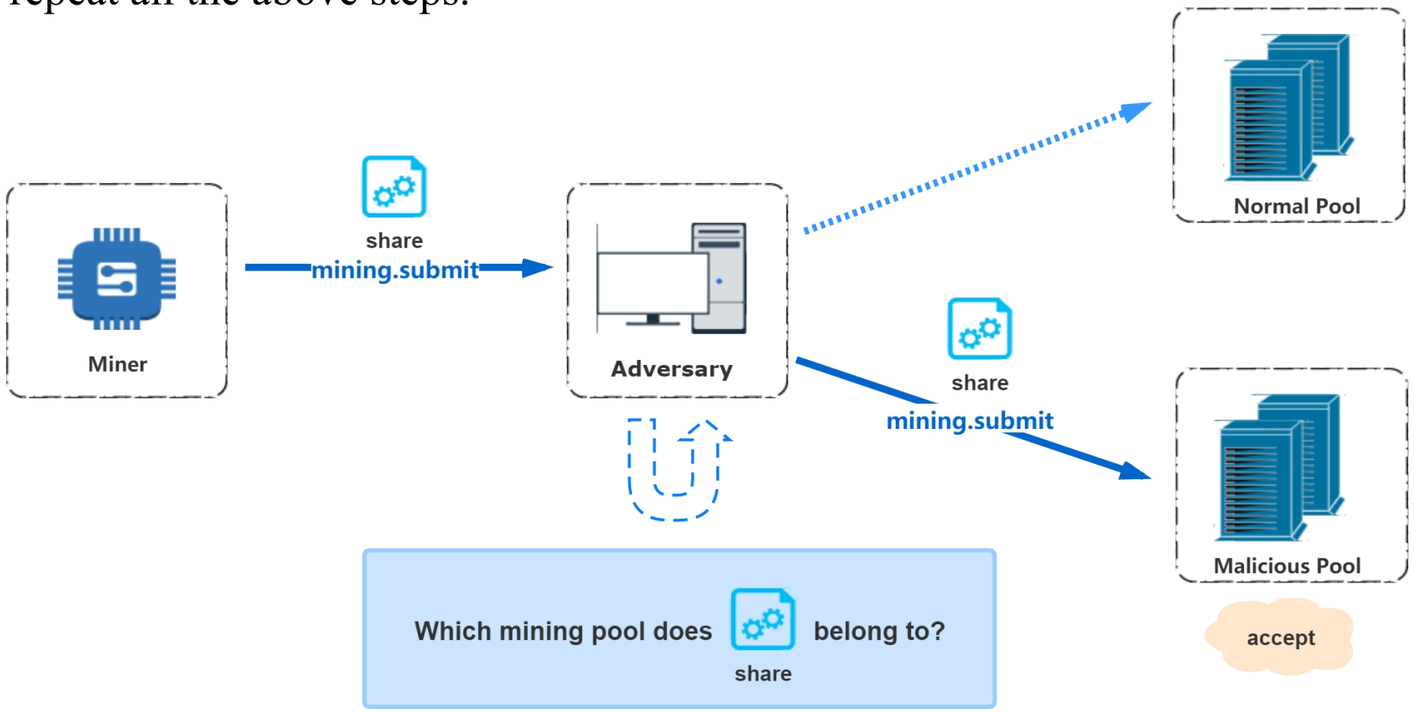


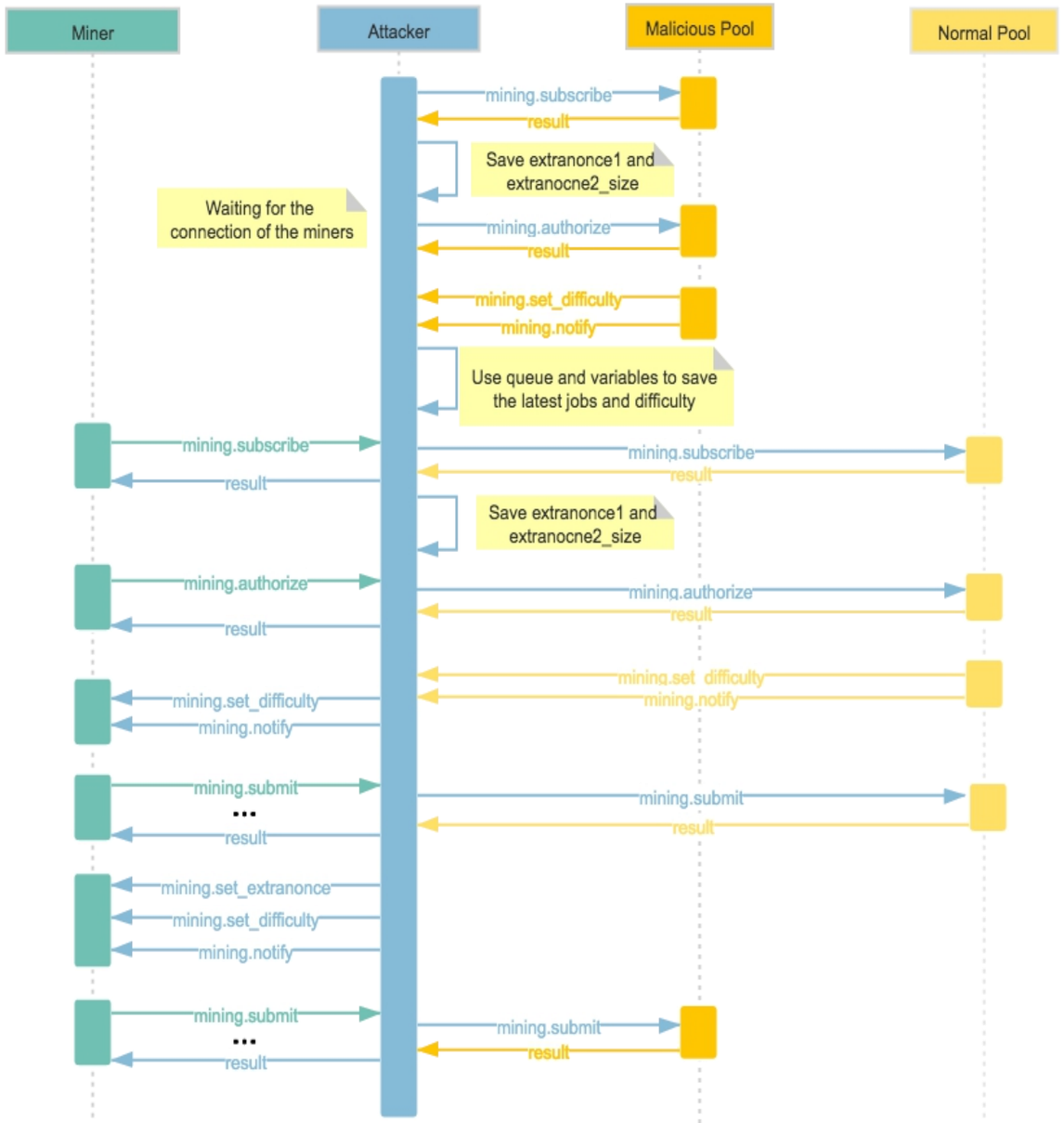
3 Steal Hashrate in Stratum Secretly

3.3 Job injection based on set_extranonce

Then, the adversary forwards *shares* to malicious pool.

After completing all the above steps, the *mining.set_extranonce* method will be used again to switch the miner to work for the normal mining pool, and repeat all the above steps.





Sequence diagram

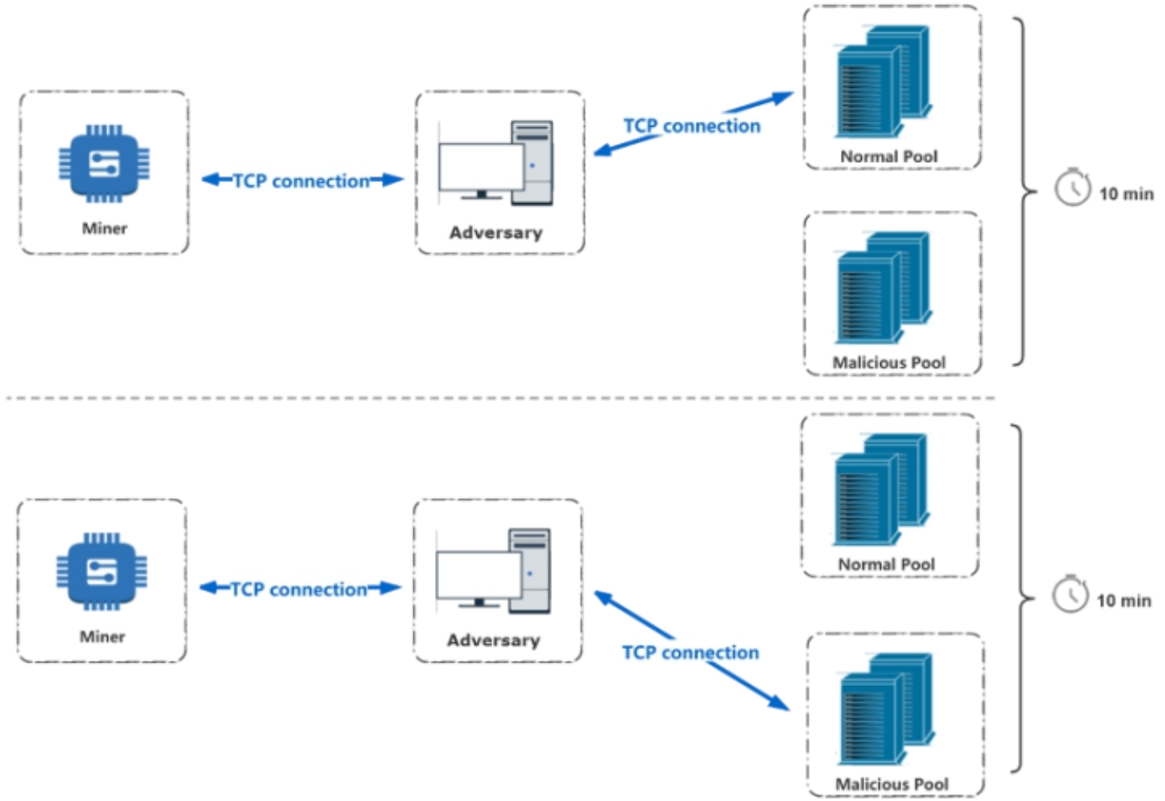
If attackers only steal 5% of the hashrate, it will be very unobvious. The miners and the pool can hardly find these attacks.

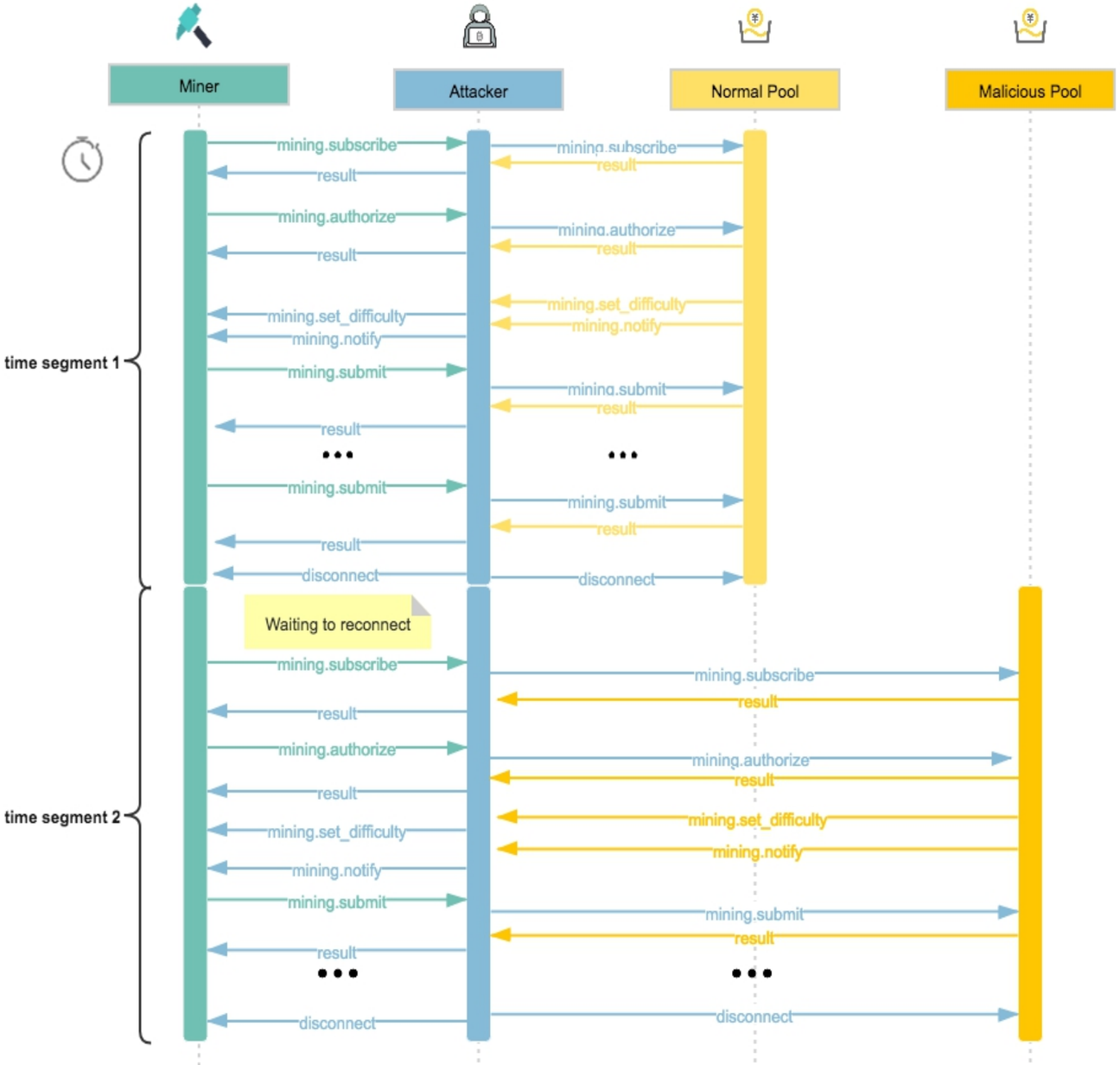
Besides, it has really good performance.

3 Steal Hashrate in Stratum Secretly

3.4 Time segment

The adversary hijacks the TCP communication between the miner and the normal pool, and make it work for two mining pool respectively at different time periods.





Sequence diagram

This method is based on the reconnection scheme.

At the end of the first time segment, the adversary disconnects from the miner and the normal pool.

In the second time segment, the adversary is waiting for reconnection from the miner. And then, make a new connection to the malicious pool.

The extranoce1 is refreshed.

4 Proof of Concept

4.1 BTCPool

BTCPool is backend system of https://pool.btc.com.

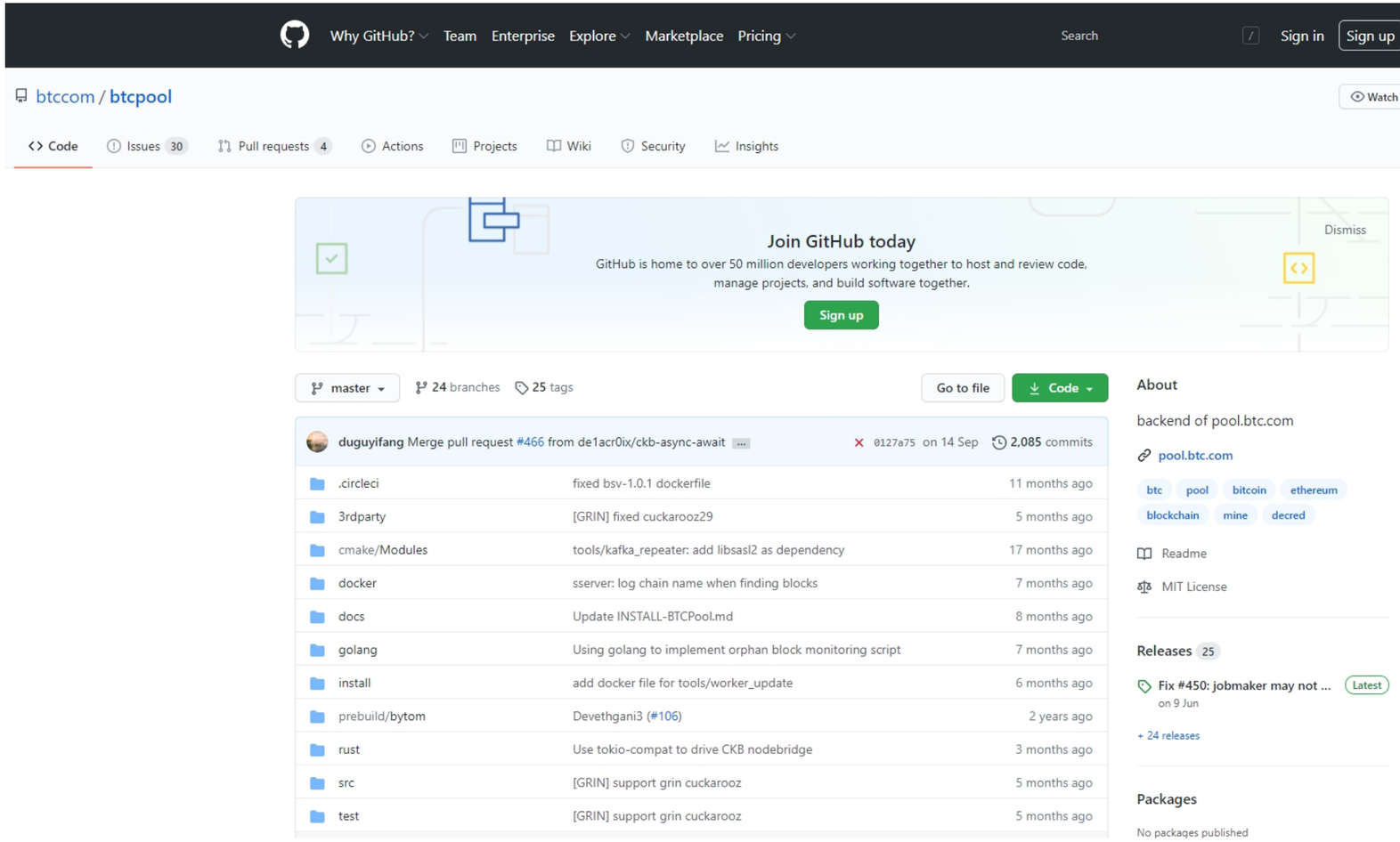
It’s an open source project on Github. https://github.com/btccom/btcpool

We built two mining pool to simulate the normal pool and the malicious pool using BTCPool on servers.

OS: Ubuntu 16.04 64 bits

4 Proof of Concept

4.1 BTCPool



4 Proof of Concept

4.2 Ccminer

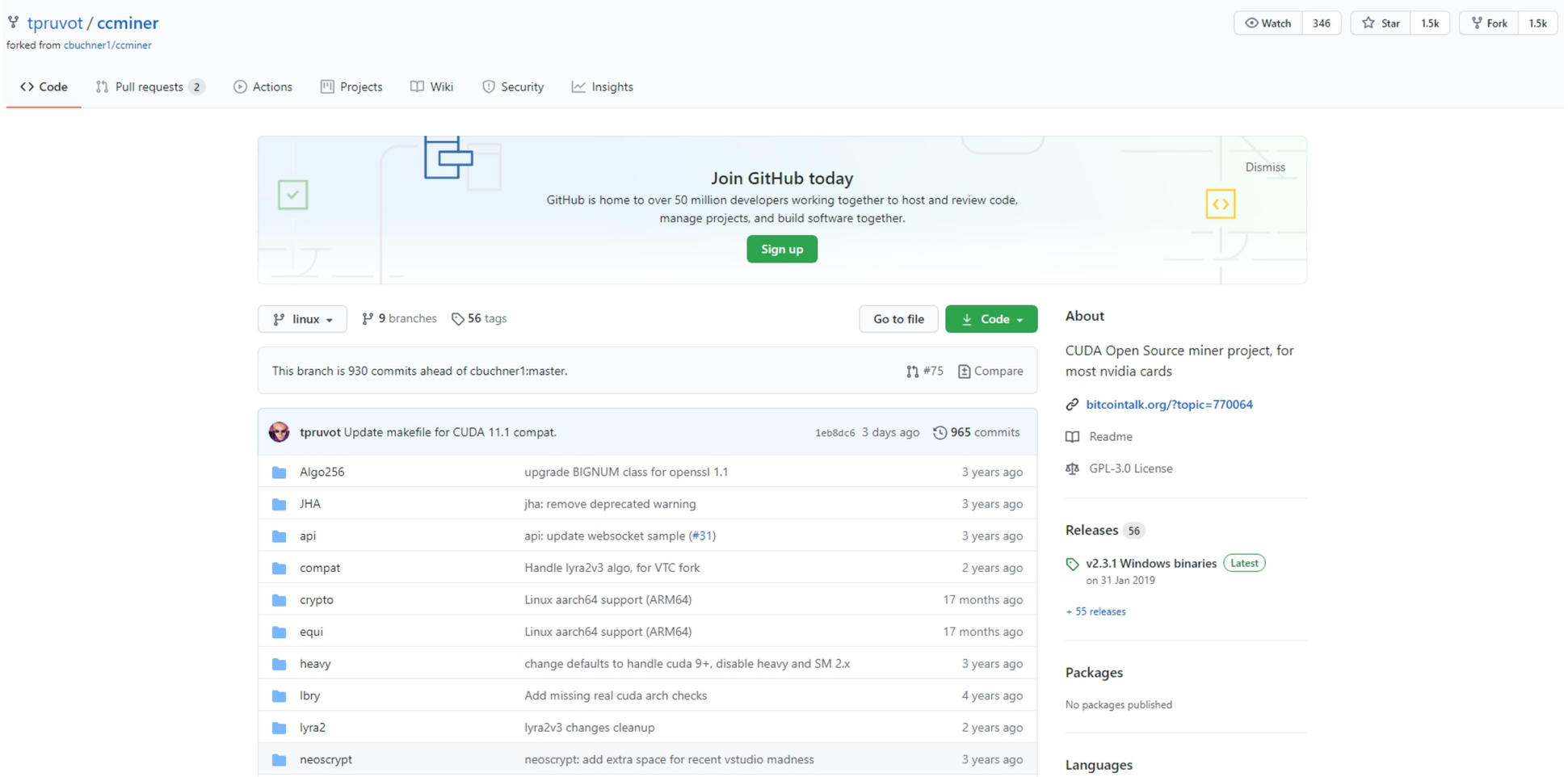
Ccminer is a kind of mining software that needs to be installed in miner's machine.

It's an open source software on Github. https://github.com/tpruvot/ccminer/tree/linux

Version: tpruvot linux

4 Proof of Concept

4.2 Ccminer



4 Proof of Concept

4.3 GPU

GeForce RTX 2080 Ti * 3 (ASIC Miners are SOLD OUT >.<)

```
[redacted]:~$ nvidia-smi
Wed Dec 16 16:59:34 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 440.33.01    Driver Version: 440.33.01    CUDA Version: 10.2     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  GeForce RTX 208...  On   | 00000000:19:00.0 Off |                  N/A |
| 30%   32C    P8    18W / 250W |      1MiB / 11019MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  GeForce RTX 208...  On   | 00000000:1B:00.0 Off |                  N/A |
|  0%   43C    P8     1W / 260W |      1MiB / 11019MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   2  GeForce RTX 208...  On   | 00000000:68:00.0 Off |                  N/A |
| 31%   36C    P8    20W / 250W |     59MiB / 11016MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|    2      1631      G   /usr/lib/xorg/Xorg                            39MiB |
|    2      1663      G   /usr/bin/gnome-shell                          17MiB |
+-----------------------------------------------------------------------------+
```

```
[2020-12-14 09:31:41] GPU #1: Intensity set to 25, 33554432 cuda threads
[2020-12-14 09:31:41] GPU #2: Intensity set to 25, 33554432 cuda threads
[2020-12-14 09:31:41] GPU #0: Intensity set to 25, 33554432 cuda threads
[2020-12-14 09:31:42] GPU #1: Gigabyte RTX 2080 Ti, 1007.25 MH/s
[2020-12-14 09:31:42] GPU #0: Zotac RTX 2080 Ti, 981.97 MH/s
[2020-12-14 09:31:42] GPU #2: Zotac RTX 2080 Ti, 979.54 MH/s
[2020-12-14 09:31:46] GPU #1: Gigabyte RTX 2080 Ti, 2849.73 MH/s
[2020-12-14 09:31:46] GPU #2: Zotac RTX 2080 Ti, 2869.03 MH/s
[2020-12-14 09:31:46] GPU #0: Zotac RTX 2080 Ti, 2855.05 MH/s
[2020-12-14 09:31:50] GPU #2: Zotac RTX 2080 Ti, 2838.45 MH/s
[2020-12-14 09:31:50] GPU #1: Gigabyte RTX 2080 Ti, 2841.18 MH/s
[2020-12-14 09:31:50] GPU #0: Zotac RTX 2080 Ti, 2829.64 MH/s
[2020-12-14 09:31:54] GPU #1: Gigabyte RTX 2080 Ti, 2561.23 MH/s
[2020-12-14 09:31:54] GPU #0: Zotac RTX 2080 Ti, 2530.67 MH/s
[2020-12-14 09:31:54] GPU #2: Zotac RTX 2080 Ti, 2842.83 MH/s
```

4 Proof of Concept

4.4 Job injection based on set_extranonce

During job injection based on set_extranonce attack, the adversary needs to keep three TCP connections to miner, the normal pool and the malicious pool at the same time. All of the messages between them will be forwarded by adversary.

- In the initial stage of the attack, the adversary will save extranonce1 and extranonce2_size from malicious pool. And the adversary will forward the subscription and authorization message to the normal pool.

- Next, the adversary will forward jobs sent by the normal pool to miner and send *shares* submitted by miner to the normal pool.

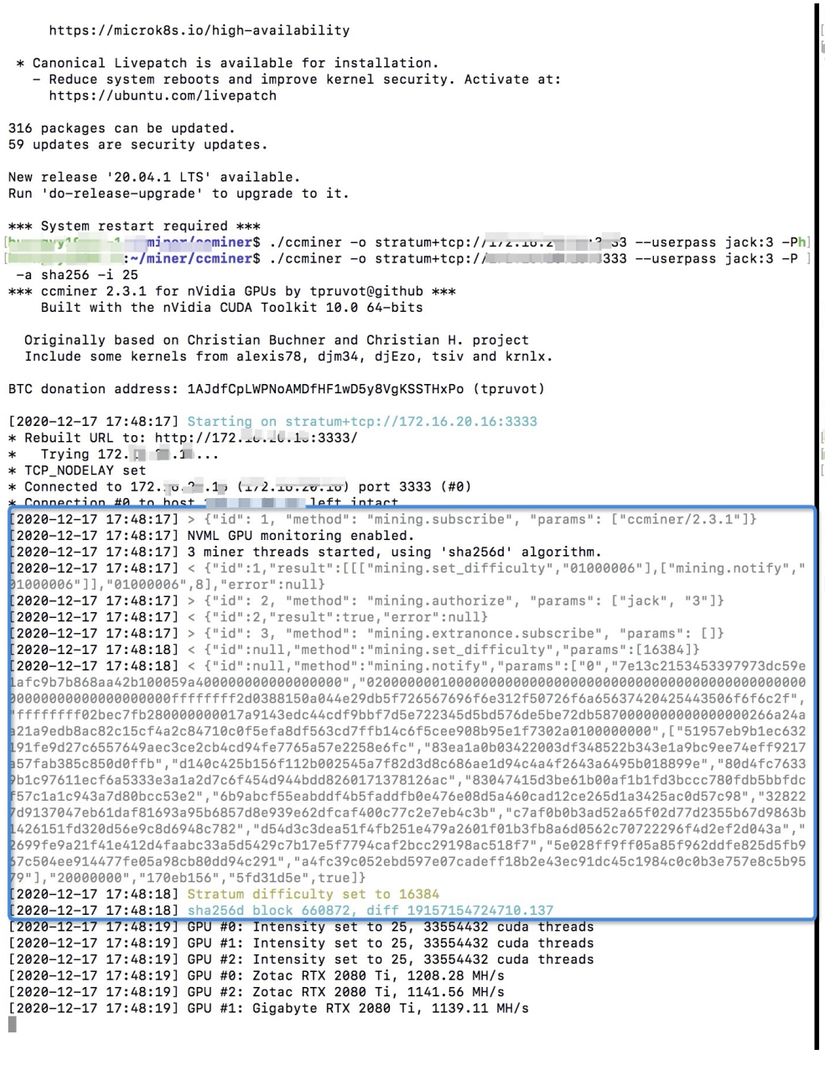
- After 10 *shares* being submitted to the normal pool, the adversary constructs fake *set_extranonce* message with extranonce1 and extranonce2_size from malicious pool and send it to miner. And then the adversary inserts the latest difficulty and job of the malicious pool into the job flow, and the miner will complete the job of malicious pool.

4.4 Job injection based on set_extranonce

The right half of the image is the output of the attack script.

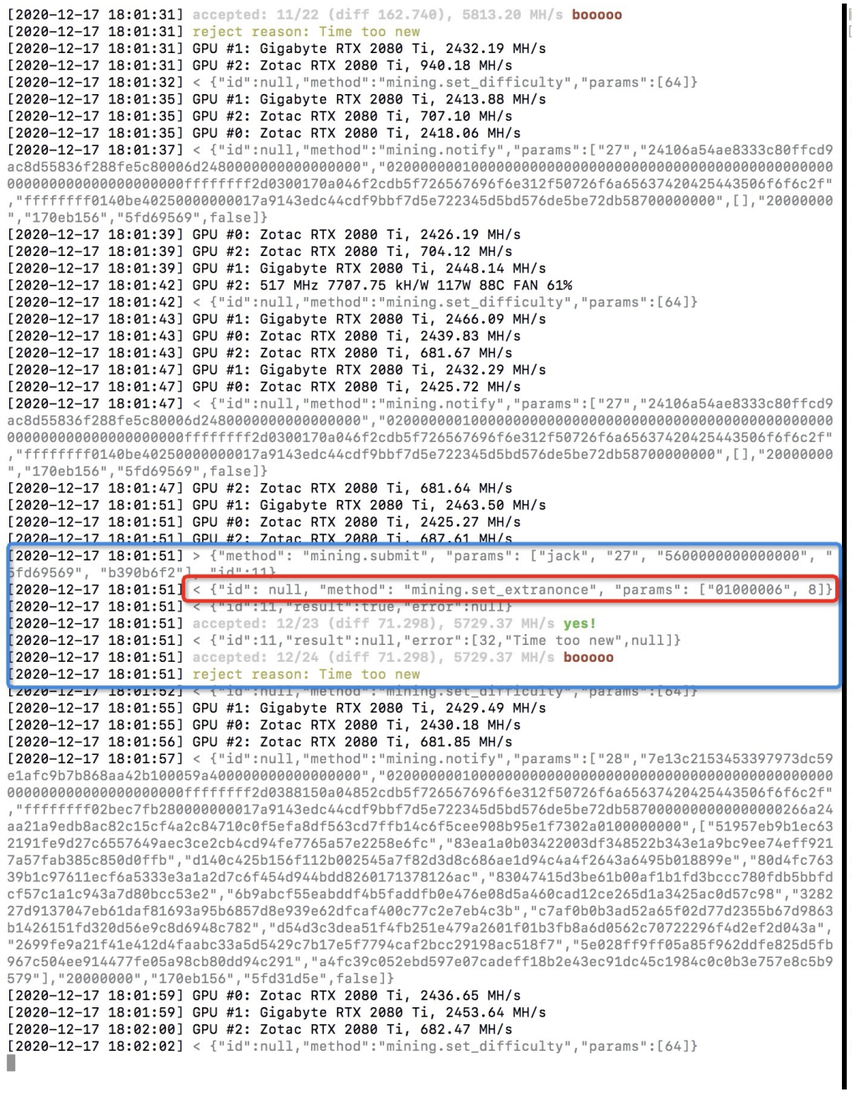
The left part is the output of the ccminer.

The miner received the job message and difficulty negotiation message hijacked and forwarded by the adversary successfully.



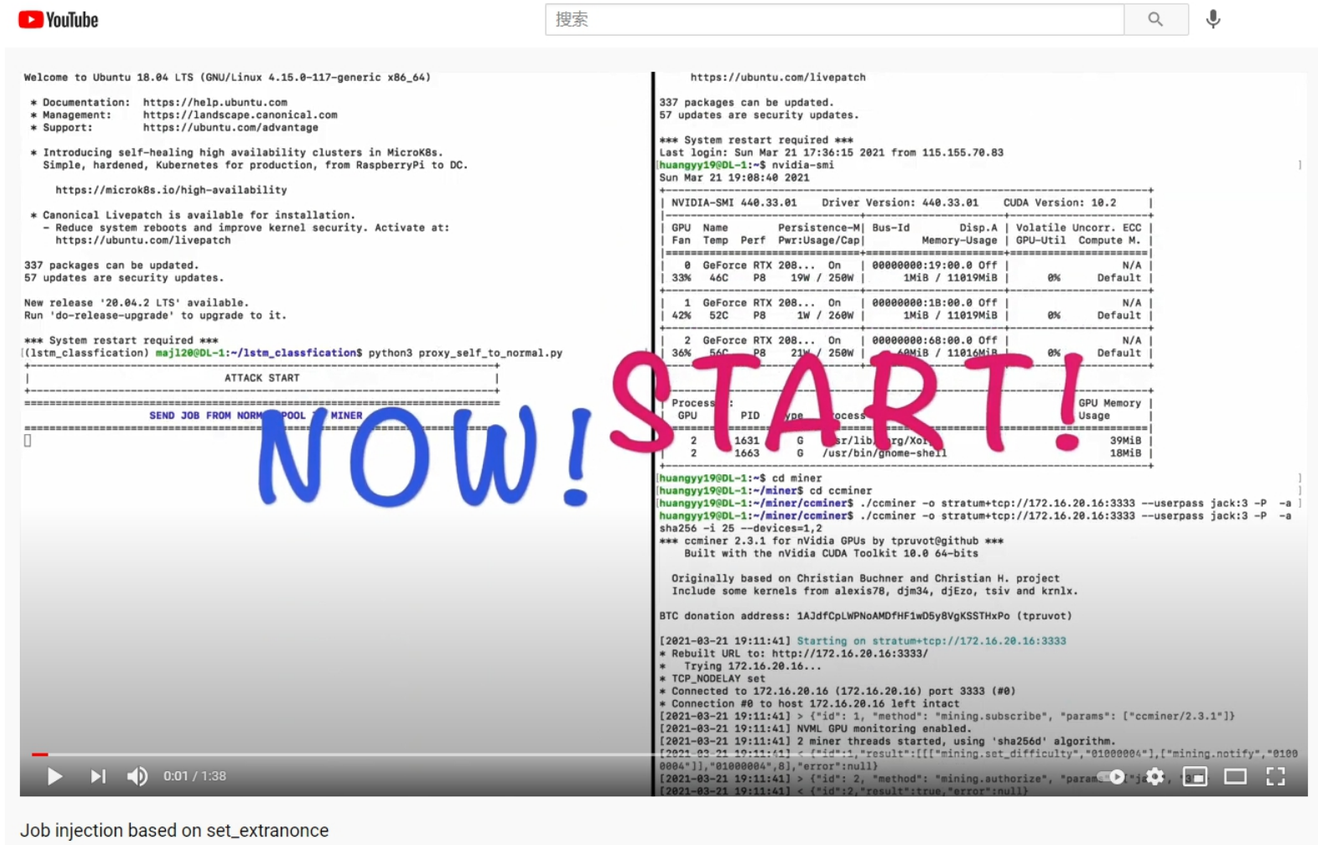
4.4 Job injection based on set_extranonce

After the miner submitted several *shares* to the normal mining pool, it received the *mining.set_extranonce* message constructed by the adversary, and it can be seen that the *share* submitted by the miner for the malicious pool was accepted by the malicious pool.



4.4 Job injection based on set_extranonce

https://www.youtube.com/watch?v=ZvpdOj6U0vM



Job injection based on set_extranonce

4 Proof of Concept

4.5 Time segment

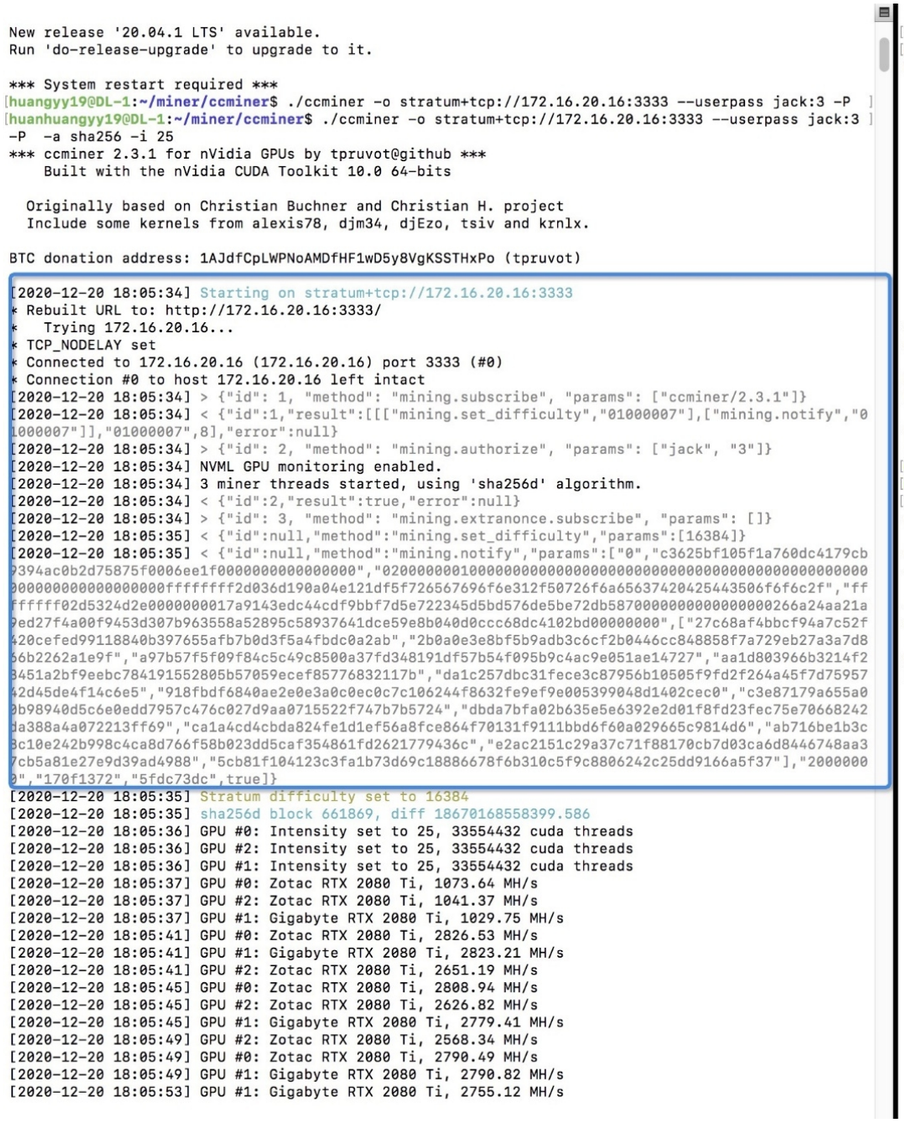
During the time segment attack, the adversary needs to switch the connection to different pools at a fixed time segment. We set the time segment to 10 minutes.

- In the first 10 minutes, the adversary will forward the subscription and authorization message to the normal pool. The miner will only work for the normal pool in the remaining time. At the end of the ten minutes, the adversary will disconnect from the normal pool.

- In the next 10 minutes, the miner will reconnect to the pool, and the adversary will hijack the TCP connection to the malicious pool. The miner will only work for the malicious pool in the remaining time until the time segment ends.

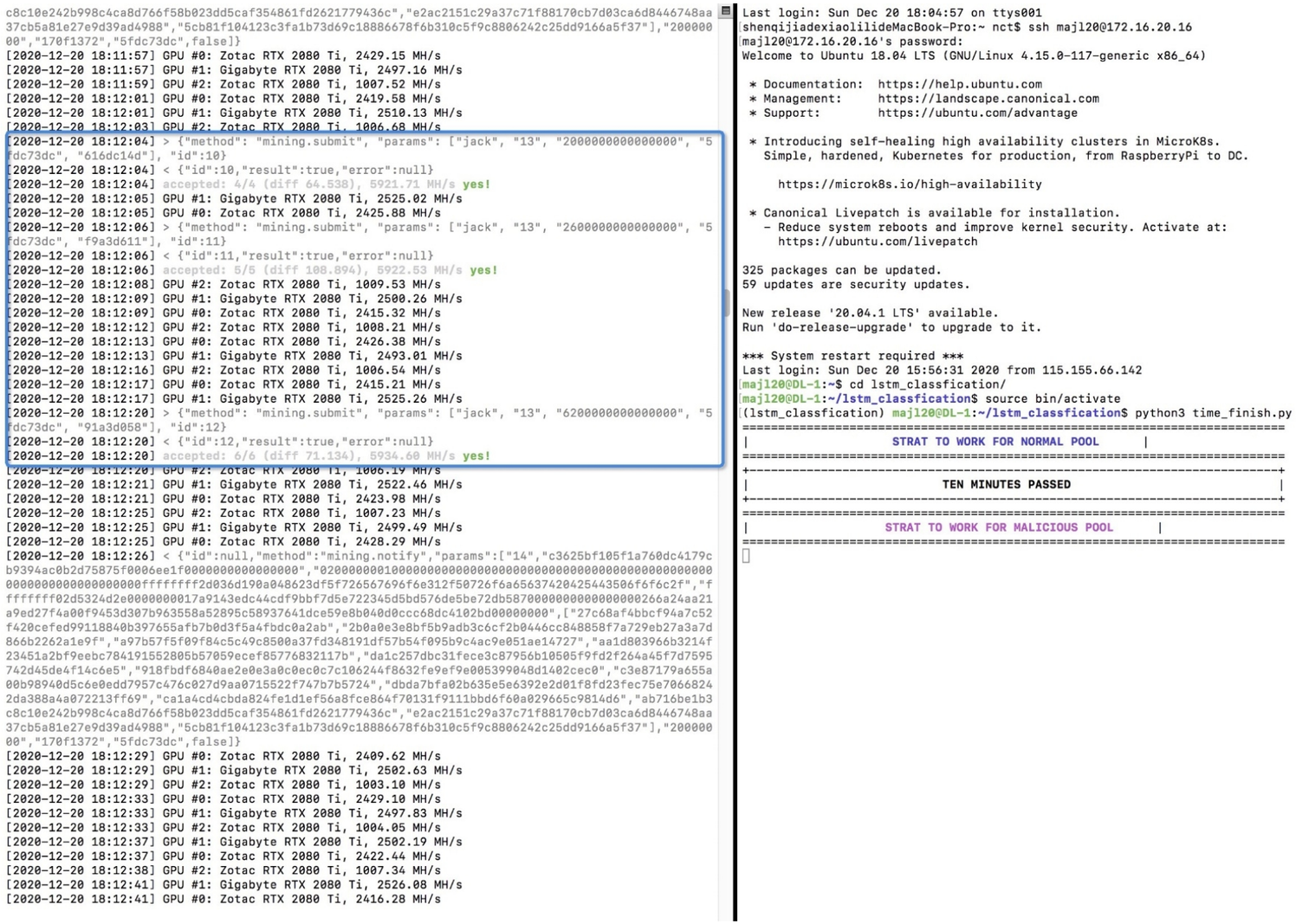
4.5 Time segment

The miner received the job message and difficulty negotiation message hijacked and forwarded by the adversary successfully.



4.5 Time segment

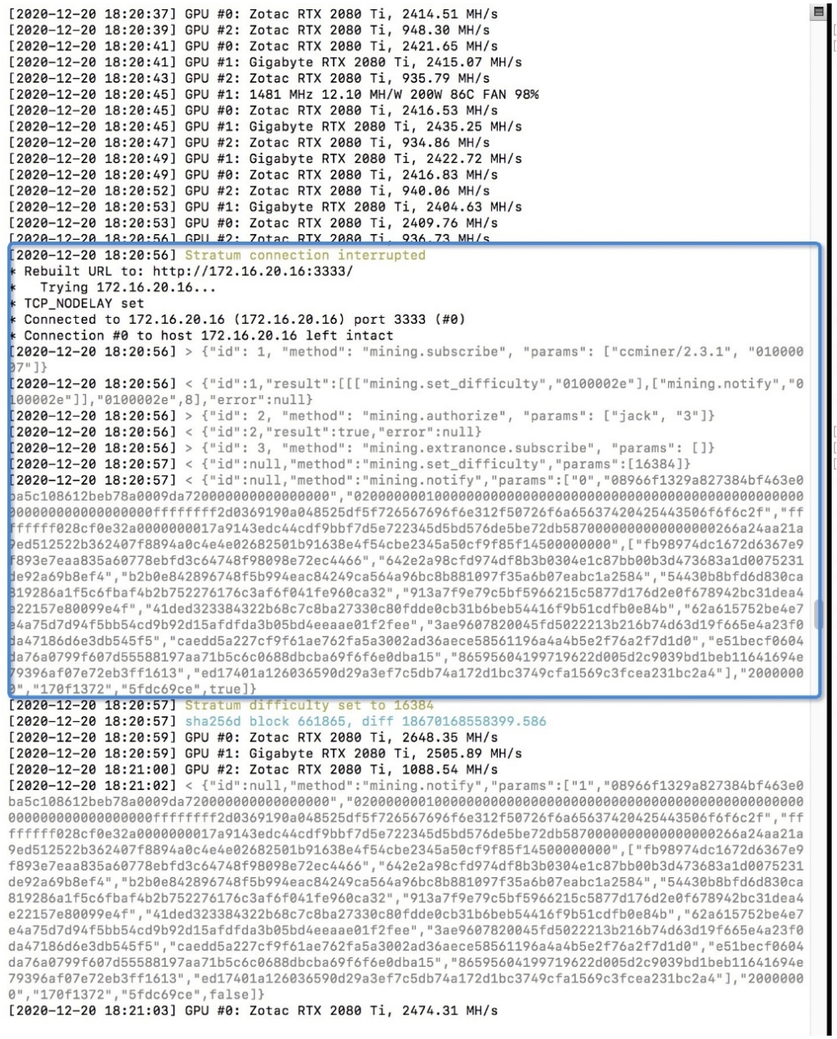
The adversary successfully forwarded the *shares* submitted by the miner to the normal mining pool.



4.5 Time segment

At the end of the first 10 minutes, the Stratum connection interrupted because the adversary disconnect from the miner and the normal pool.

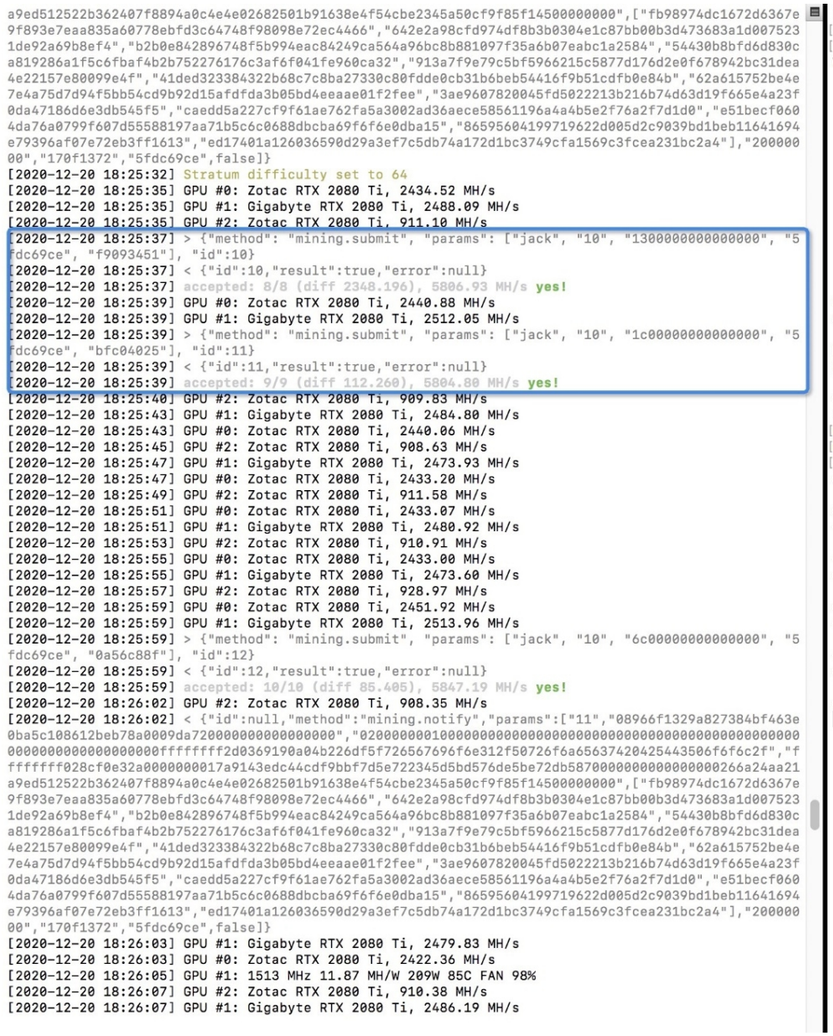
And then the miner reconnect to the pool, and the adversary hijacks the connection to the malicious pool.



4.5 Time segment

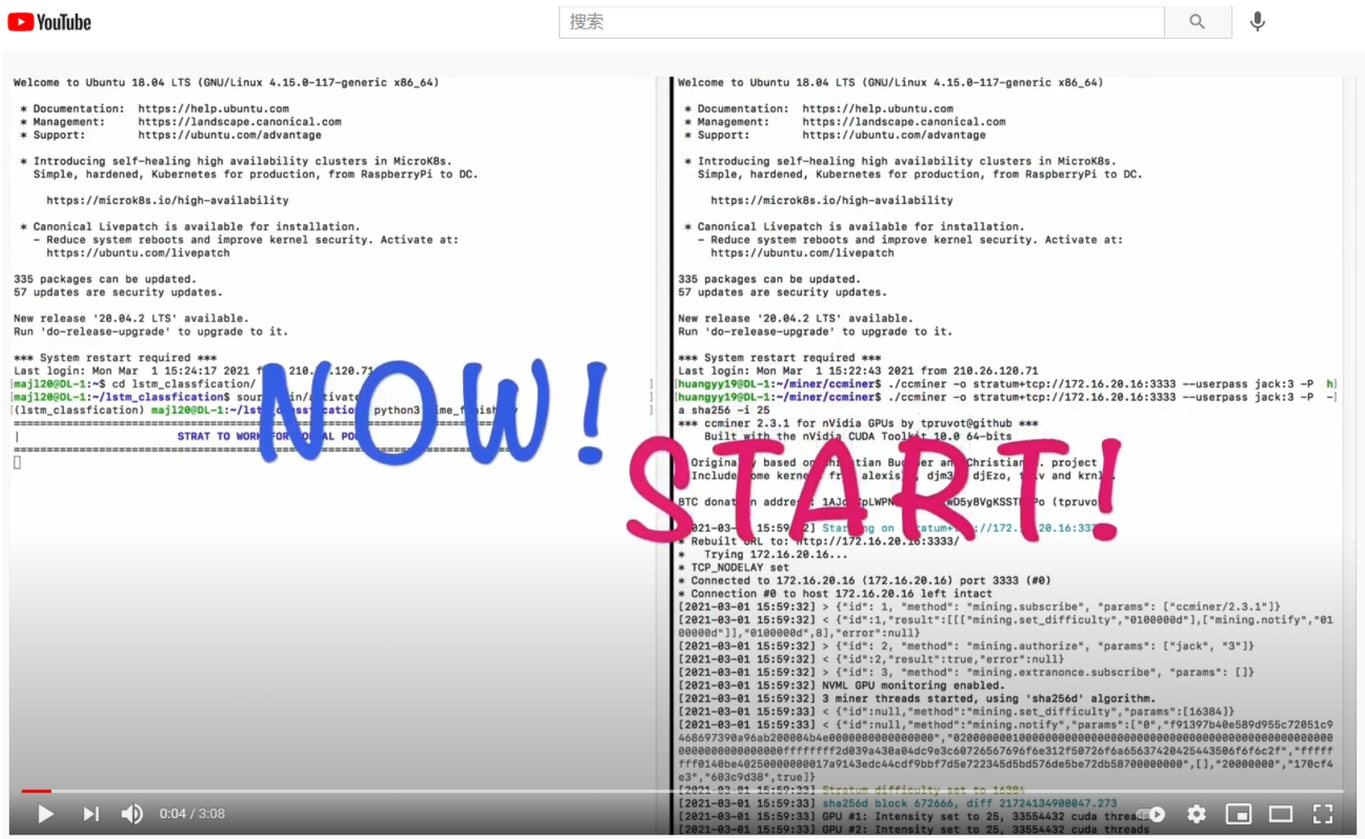
The malicious pool has received and accepted the *share* submitted by the miner.

As the second ten minutes ends, the adversary disconnects from the miner and the malicious pool, and will be waiting for the reconnection of miner.



4.5 Time segment

https://www.youtube.com/watch?v=OIS4TRMgAJs



4 Proof of Concept

4.6 Summary

Job injection based on set_extranonce attack model has better concealment, because it inserts a small amount of malicious pool job to miner in the job flow at a low frequency, making it difficult for the pool administrator to detect.

The second attack model is to switch the connection between the normal pool and the malicious pool in a fixed time segment. Therefore, the mining pool administrator may observe the fluctuation of the computing power.

Both of our proposed attack schemes can achieve the purpose of stealing hashrate for malicious mining pool.

4 Proof of Concept

4.6 Summary


Clarivate™

MarkMonitor™ Why Domain Management Products &

BLOG | UNCATEGORIZED JANUARY 25, 2019 3 MINUTE READ

China cybersecurity update: DNS hijacking and IoT crimes

BRIAN KING
Director of Internet Policy and Industry Affairs
Clarivate

Share this article

Many ISPs are using traffic hijacking for illegal profit-making activities such as pop-up advertisements.

If they change the target of hijacking to Stratum, it will greatly harm the interests of miners.

It’s dangerous to blockchain community.

Thanks