Distributed Operating System Principles (COP5615)

Project-1

Group Members

JAYACHANDRA GUNTURU (UFID: 56276000)

SATYA NAGA AKHILESH IRRINKI (UFID: 64730546)

SUMMARY

Implementation:

- Using F# and Actor model by AKKA actor library, implemented a solution that finds the SHA256 hash for a random string with the specified number of leading zeros given as input.
- Extended this solution to work with multiple clients and use more cores for mining.

Server:

- The server takes a number 'K' number of leading zeros as input for mining the required hash.
- The hash is generated by appending the UFID with a random string and encrypting with SHA 256 Algorithm.
- The random string is generated by taking 8 random characters from 'A-Z' and '0-9' and are formed into a string. The resultant string is added to the UFID.
- The server creates a pool of child actors and passes them the input value K. The child actors each then start working to find the desired hash value.
- The pool of actors generates SHA256 hash for the randomly generated string concatenated with UFID and checks them for the number of leading zeros.
- When the number of leading zeros values of hash matches with the K then, the hash value is and its token are stored.
- The server also checks any incoming request from the remote worker and sends the K value if there is a request.
- Finally when the RemoteWorker sends its generated hash value to the server, the server prints all its hash values if it finds any and the hash value from the RemoteWorker indicating that remote communication has been done successfully.

RemoteWorker:

- The remote worker takes the server's IP address as input and sends a request to the server for the K value so that it can also start mining.
- The remote worker then spawns a pool of child actors which work to find a hash with K leading zeroes.
- This pool of actors receives the K values and does the same process of generating SHA256 hash for the required string and check if the leading zeros match the value of K.
- Once the desired hash is found the remote worker sends the results and the completed message to the server and the server prints the required SHA256 hash with K leading zeros.

Execution:

- The Zip file consists of two F# script files named Server.fsx and RemoteWorker.fsx

Running Server.fsx

- Move to the directory in which the Server.fsx file is and change the IP address to current machine IP address.
- Run the following command with K (Required number of leading zeros in the hash value) as argument in the terminal.

'dotnet fsi Server.fsx K_Indicating_Leading_Zeroes'

- The server then starts for mining the coins with 4 leading zeros and looks for the request from any remote worker.
- Once the execution is completed the server prints the Hash value mined by the client and also all the hash values mined by the server.

Running RemoteWorker.fsx

- Move to the directory in which the RemoteWorker.fsx file is and change the IP in the configuration section and myIP at top to current machine IP.
- Run the following command int the terminal by giving the IP address of the server you want to communicate and get the K value from

'dotnet fsi RemoteWorker.fsx Server_IP_Adress'

- The RemoteWorker then tries to connect with the server and request for the K value.
- Once the RemoteWorker receives K value it also starts mining for the coin parallelly and send the found coin back to the server.
- Both the Server and RemoteWorker run on port 9090 in their respective machines.

RESULTS

1. Size of the work unit for best performance

- We have observed some latency across network due to which the real time is being increased thereby decreasing the CPU/Real Ratio. But it can be leveraged by the cores offered by the remote machine. As the number of leading zeroes increase there is increase in CPU/Real ratio. Even for the same K value the Time ratio is unpredictable when varying the actors. So it depends on the random string generation that can get the desired leading zeroes by applying SHA-256. But as the network latency decreases and powerful machines join the network, the desired hash value can be generated faster. But ultimately it is the randomness that cannot be predicted. However if we can generate strings sequentially that are closer to attain desired leading zeroes the performance would be a lot better.

2. Result for input as 4(leading Zeros)

- Following is the coin we found with 4 leading zeros and input string as UF Username+8char random string

  Input string: sirrinki;ZRQ8Y8ZQT

  Output hash value with 4 leading zeros:

  00004cbe0c189add824ac7b345ecc013ace97bd2ef3db9db87b17eafa375251e

```
Server worker started to mine hashes.....................
Remote Worker with IP 192.168.0.86 started mining for desired leading zero hashes
----------------Hashes mined by client----------------
IP Adress of client : 192.168.0.86
Token generated by client: sirrinki;ZRQ8Y8ZQT
Hash generated by client: 00004cbe0c189add824ac7b345ecc013ace97bd2ef3db9db87b17eafa375251e
```

3. CPU time/Real time for RemoteWorker and Client

- Below is the ratio of CPU time to REAL time for finding hash with 4 leading zeros by varying the number of worker actors on the **Remote Worker**

K Value – No of leading Zeros	No of Actors	CPU time	REAL time	CPU/Real Ratio
4	4	3.191 sec	6.421 sec	2.01
4	10	10.734 sec	4.498 sec	2.38

4	100	1 min 39.953 sec	41.983 sec	2.38
4	500	6 min 43.984 sec	4 min 11.876 sec	1.67
4	1000	12 min 04.031 sec	8 min 06.354 sec	1.5

- **For 4 leading zeros when No of actors = 4:**

Hash value generated by RemoteWorker :

```
Remote Worker with IP 192.168.0.86 started mining for desired leading zero hashes
---------------Hashes mined by client---------------
IP Adress of client : 192.168.0.86
Token generated by client: sirrinki;4A9CU1TWJ
Hash generated by client: 0000f9c83d77fa8c50af31609348f79095926a735c8dec1f75de855233b3103b
```

CPU, Real Time for RemoteWorker

```
Real: 00:00:00.000, CPU: 00:00:00.000, GC gen0: 0, gen1: 0, gen2: 0
Generating hash with 4 leading zeroes
Real: 00:00:03.191, CPU: 00:00:06.421, GC gen0: 356, gen1: 3, gen2: 0
PS D:\Sem-1\DOSP\RemoteProject>
```

- **For 4 leading zeros when No of actors = 10:**

Hash value generated by RemoteWorker

```
Remote Worker with IP 192.168.0.86 started mining for desired leading zero hashes
---------------Hashes mined by client---------------
IP Adress of client : 192.168.0.86
Token generated by client: sirrinki;JHMQ2JQYE
Hash generated by client: 0000937ac7de66b95f16068040258111a68027335c99fd32e0156ce5d7947f03
```

CPU, Real Time for RemoteWorker

```
Real: 00:00:00.000, CPU: 00:00:00.000, GC gen0: 0, gen1: 0, gen2: 0
Generating hash with 4 leading zeroes
Real: 00:00:04.498, CPU: 00:00:10.734, GC gen0: 613, gen1: 5, gen2: 0
PS D:\Sem-1\DOSP\RemoteProject>
```

- **For 4 leading zeros when No of actors = 100:**

Hash value generated by RemoteWorker

```
Remote Worker with IP 192.168.0.86 started mining for desired leading zero hashes
---------------Hashes mined by client---------------
IP Adress of client : 192.168.0.86
Token generated by client: sirrinki;VG0R6VKTU
Hash generated by client: 00001753601c372add49ba48836997bf6c3d2d89f57048b885dea8af0e080b4e
```

CPU, Real Time for RemoteWorker

```
PS D:\Sem-1\DOSP\RemoteProject> dotnet fsi RemoteWorker.fsx 192.168.0.118
Real: 00:00:00.000, CPU: 00:00:00.000, GC gen0: 0, gen1: 0, gen2: 0
Generating hash with 4 leading zeroes
Real: 00:00:41.983, CPU: 00:01:39.953, GC gen0: 10298, gen1: 9, gen2: 0
```

- **For 4 leading zeros when No of actors = 500:**

Hash value generated by RemoteWorker

```
Remote Worker with IP 192.168.0.86 started mining for desired leading zero hashes
----------------Hashes mined by client----------------
IP Adress of client : 192.168.0.86
Token generated by client: sirrinki;8LRHF27MQ
Hash generated by client: 00009678499a9d35cbc65e2a94ac941df315385471fdde74c7039882dfa749a4
```

CPU, Real Time for RemoteWorker

```
Real: 00:00:00.000, CPU: 00:00:00.000, GC gen0: 0, gen1: 0, gen2: 0
Generating hash with 4 leading zeroes
Real: 00:04:11.876, CPU: 00:06:43.984, GC gen0: 60707, gen1: 40, gen2: 2
PS D:\Sem-1\DOSP\RemoteProject>
```

- **For 4 leading zeros when No of actors = 1000:**

Hash value generated by RemoteWorker

```
Remote Worker with IP 192.168.0.86 started mining for desired leading zero hashes
----------------Hashes mined by client----------------
IP Adress of client : 192.168.0.86
Token generated by client: sirrinki;EUBV4YYYM
Hash generated by client: 00000f74019113bf731222273c3e34d7b7b37a72b0bc0784aae206198898e7cb
```

CPU, Real Time for RemoteWorker

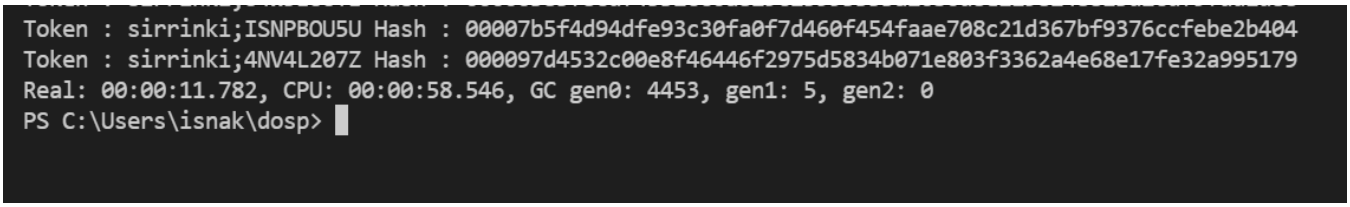
```
Real: 00:00:00.000, CPU: 00:00:00.000, GC gen0: 0, gen1: 0, gen2: 0
Generating hash with 4 leading zeroes
Real: 00:08:06.354, CPU: 00:12:04.031, GC gen0: 113581, gen1: 65, gen2: 4
PS D:\Sem-1\DOSP\RemoteProject>
```

- As the server keeps on mining till it gets a hash generated from the client, the CPU time is much higher than that of the client CPU time.
- Below is the ratio of CPU time to REAL time for finding hash with 4 leading zeros by varying the number of worker actors on the **Server.**

K Value – No of leading Zeros	No of Actors	CPU time	REAL time	CPU/Real Ratio
4	4	58.546 sec	11.782 sec	4.96
4	10	43.625 sec	8.96 sec	4.86
4	100	3 min 48.171 sec	44.082 sec	5.17
4	500	15 min 24.937 sec	4 min 20.237 sec	3.55
4	1000	22 min 11.796 sec	7 min 53.101 sec	2.81

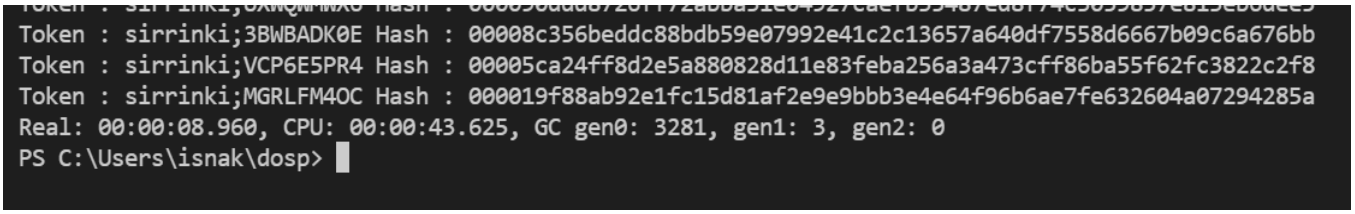
- **For 4 leading zeros when No of actors = 4:**

  As the server keeps on mining for Hash values till it gets response form client, there would be more hash values generated by the Server.



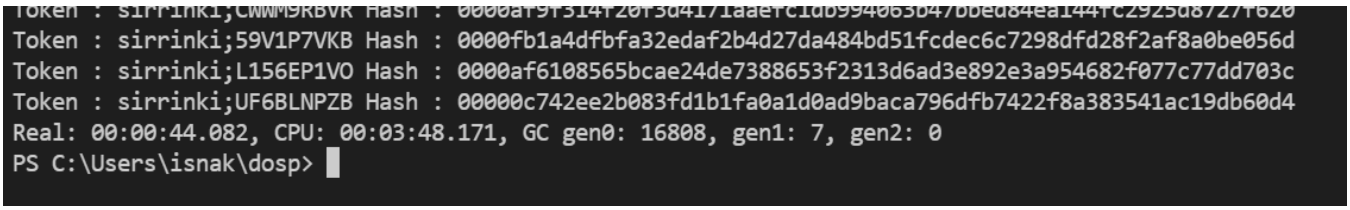
- **For 4 leading zeros when No of actors = 10:**

  Hash values generated by Server



- **For 4 leading zeros when No of actors = 100:**

  Hash values generated by Server



- **For 4 leading zeros when No of actors = 500:**

  Hash values generated by Server

```
Token : sirrinki;6YUJ3DZYC Hash : 00008fa0e439e75f8f1c8bb1653063ffab823aff12b87b772f3cb96f46e26de3
Token : sirrinki;O38MZFJMQ Hash : 0000a13811d261b7a0870cdfa2cdc5aa3cd2978261aec0e8c4187c75b8f3adf0
Token : sirrinki;6U7UI2560 Hash : 000055f6c28077efe78010c78511342b2a2d1eb8922c58d160381008cbe3b3cd
Token : sirrinki;5XE7GL7V5 Hash : 0000ce352b7156a2fcf80ac48ae2a4731fd6fdf523626e3138693e6a9ca10f7b
Real: 00:04:20.237, CPU: 00:15:24.937, GC gen0: 87423, gen1: 29, gen2: 2
PS C:\Users\isnak\dosp>
```

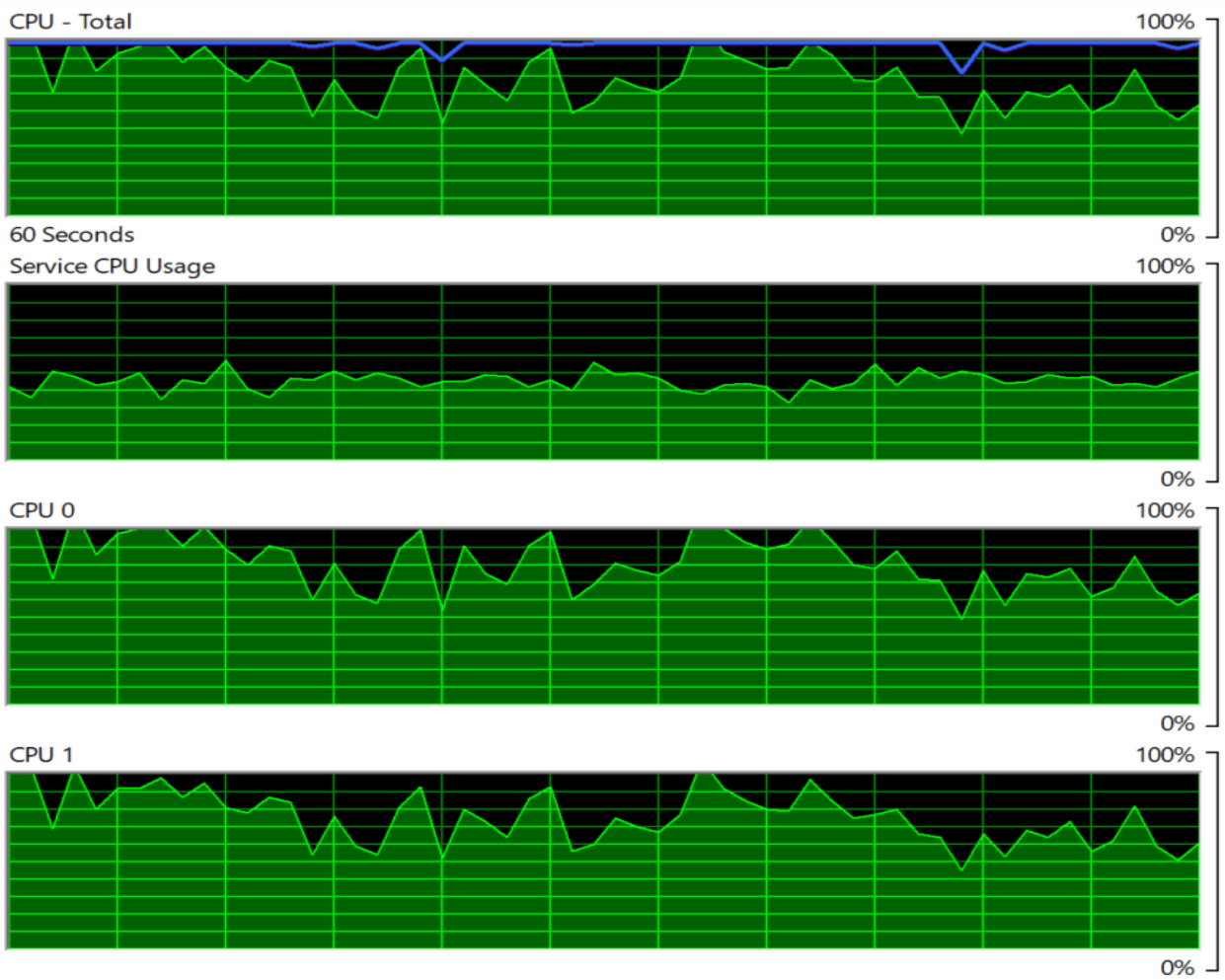
- **For 4 leading zeros when No of actors = 1000:**

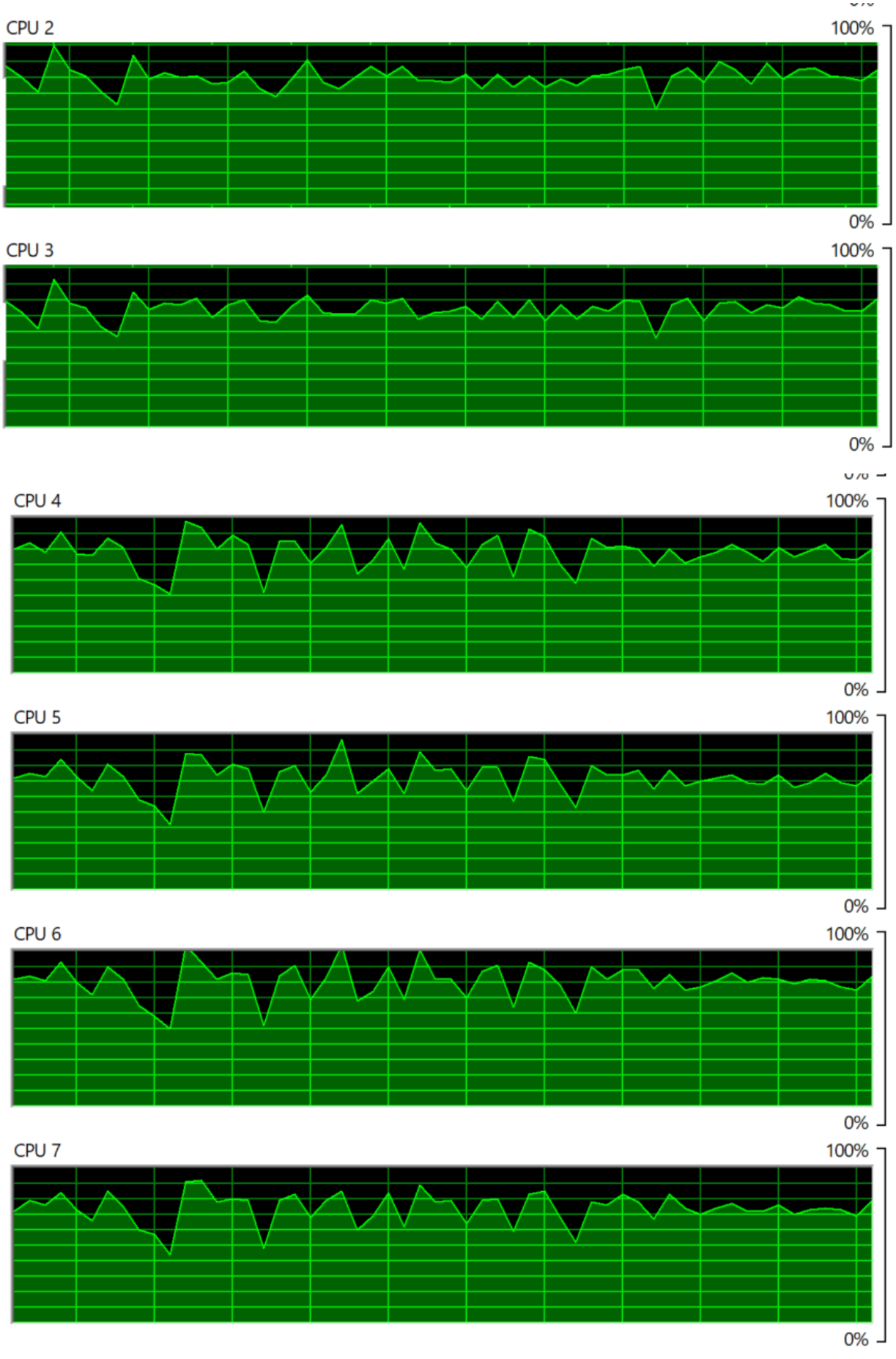
  Hash values generated by Server

```
Token : sirrinki;2PO7H48YR Hash : 0000b698d9e0527b35aea29e8946ccac1533b7e5a6c607eaf6db0173213bb92b
Token : sirrinki;NB88W5616 Hash : 0000e5a0e36c0b6c538bb66f5852ec076a65c1cf41e662c65db4ac26dc101eb0
Real: 00:07:53.101, CPU: 00:22:11.796, GC gen0: 120768, gen1: 51, gen2: 4
PS C:\Users\isnak\dosp>
```

- **<u>Performance of all the 8 cores in the CPU while mining for coins:</u>**

  The load is distributed to all the CPU's for processing the coins that generates desired hash value with required leading zeroes.




CPU 2
100%
0%
CPU 3
100%
0%
CPU 4
100%
0%
CPU 5
100%
0%
CPU 6
100%
0%
CPU 7
100%
0%

4. Most leading zeros coin found

- The coin with most number of leading zeros found is **8.**

  Input string:

  sirrinki;ithzw6mqzhs076ci0

  Output hash value with 4 leading zeros:

  00000000c4b1f8003887771388e5b8f4a5b5a167ce26c00d656837cca8367086

```
Token : sirrinki;ithzw6mqzhs076ci0
Hash Value   : 00000000c4b1f8003887771388e5b8f4a5b5a167ce26c00d656837cca8367086
192.168.0.118:8778
Real: 00:04:05.632, CPU: 00:14:38.015, GC gen0: 81836, gen1: 32, gen2: 2
```

5. Max number of working machines

- We were able to connect four machines where one machine acts as a server machine and the rest are remote workers.
- The server sends K(No of leading zeros) value when requested by the remote workers.
- The server prints the IP Address of the RemoteWorker whenever a remote machine connects with it for K.
- When one of the clients sends the desired Hash value the Server prints the IP address of the Remote worker that first sent the hash value, the token and hash value.
- The server then prints its generated hash values if any.

```
Server worker started to mine hashes.....................
Remote Worker with IP 192.168.0.118 started mining for desired leading zero hashes
Remote Worker with IP 192.168.0.245 started mining for desired leading zero hashes
Remote Worker with IP 192.168.0.100 started mining for desired leading zero hashes
----------------Hashes mined by client----------------
IP Adress of client : 192.168.0.100
Token generated by client: sirrinki;0P5H04VUO
Hash generated by client: 0000078063cb13ba8b3b758db6c83e3b66bb8e0fcb937d2f7e8d8216ce6b6062
----------------Hashes mined by server----------------
Token :  Hash :
Token : sirrinki;UTQ2GDVB3 Hash : 00000a09b07eab7b3bb210a549786bb48d05907909e132c9d47376bb3f1c26d2
Token : sirrinki;J2LIG7ZW1 Hash : 00000bffe005c3b3bd280b7ad88772eb4657712a8ce886093c81358b890f8d75
Token : sirrinki;9KH4SQYZV Hash : 00000ec9802a6fe6d88fcfa3dcc985768859cf57d964fcdd6c20b18691c16162
Token : sirrinki;0TBGKZ5YL Hash : 00000fbaeded63e8793c37541afff9a902ac9c92765df11846a0846585104e4e
```