Secure Systems Development

Assignment 3 (Option B)

28th March 2022
(Due for submission on **8th May 2022**)

The aim of this assignment is to develop a simple real-time *Bitcoin transaction viewer*. The aim is that the viewer will connect to the Bitcoin network, and then show:

- New transactions as they arrive into the Bitcoin network
- New blocks as they are mined and announced to the network

The viewer should show this information in real-time, as it is announced to the network.

How do I do this?

There are four main steps involved:

1. Study the types of messages that are transmitted between nodes in the Bitcoin network, and write code to parse them
2. Make your program connect to the Bitcoin network and start receiving broadcasts when new transactions are sent or new blocks are mined
3. Download details of the event as it occurs, and display some interesting information to the user (for example, you could display the value of a transaction, the hash and nonce of a block etc.)
4. Put this all in a loop to keep it running forever!

Bitcoin transactions

You can read about the Bitcoin transaction format here:

https://en.bitcoin.it/wiki/Protocol_documentation

The basic gist is all messages sent over the network use a specific datagram format, consisting of:

- The magic number **`0xD9B4BEF9`**, encoded in *little endian* format, i.e. as **`0xF9BEB4D9`**
- 12 bytes, indicating the *type* or (*command*) of transaction being send, with zeros as padding at the end if the type is less than 12 bytes long
- The *length* of the payload in bytes, encoded as a 32-bit value
- A *checksum*, the first four bytes of the *double SHA256 hash* of the payload (the double SHA256 of a value is the SHA256 of the SHA256 of this value!)
- The *payload* itself

The website contains detailed information about the encoding of each of the payloads that can be used in the payload field.

Connecting to the Bitcoin network

To make the initial connection, you first need to know the IP address of at least one existing Bitcoin node to connect to. If you don't know any, you can find them by doing a DNS lookup on a special Bitcoin lookup domain, which will get you list of reliable IP addresses for Bitcoin nodes. One example of such a domain is **`seed.bitcoin.sipa.be`**, and you can find others by googling.

Once you have an IP, connecting involves:

- Sending a **`version`** payload to the node
- The node will reply by sending back the **`version`** payload, and a **`verack`** payload
- Your program should send another **`verack`** back
- The node will then being sending you events as they occur!

Note that once you have a working connection, you can also send a **`getaddr`** payload to the node, and it should respond with an **`addr`** payload, telling you about all the other nodes it knows. You can store this information in a file, which allows you to build up a database of known IP addresses. Note that you can also use the **`ping`** payload to check if a node is still online. (Note: storing IP addresses like this i**sn't needed for the assignment**, I'm including the information for those who are interested! You can work on a hard-coded IP address for the assignment!)

Getting events

The node you connect to should send events to your machine as they occur, in real time. The events come in the form of the **`inv`** payload. Each inv contains a list of *inventory vectors*, each of which is basically an alert about an event that might interest you. Each inventory vector is 36 bytes long, and contains two things:

- A 32-bit value indicating the *type* of event (for example, 1 indicates a transaction, 2 a block)
- The *double hash* of the transaction or block, which will be 32 bytes long

You can ask for the full transaction or block by sending a **`getdata`** payload, containing the list of inventory vectors that interest you. You will get **`tx`** or **`block`** payload back in response.

These structures will contain detailed information about the transaction or block.

How do I submit?

The assignment should be submitted via Brightspace before 11:59PM on Sunday 8th May 2022.

Your submission should be either a *compressed file* (Zip/tar/etc.) containing the source code, or a *link to a git repository* containing your source code.

Note that this assignment is ***worth 30% of the total mark*** for this class.

How will the assignment be marked?

Marks will be awarded for implementing the following:

- Up to **10 marks** for connecting to the Bitcoin network, and receiving the inv payloads
- Up to **15 marks** for successfully printing details of the transactions and blocks as they arrive. The actual information you print is up to you, but your code should clearly be parsing the datagram arriving from the nodes

- Up to **5 marks** for clean code. This includes commenting, a README file explaining how to run the code, and a sane decomposition of the program into functions/classes/etc.

Good luck!