Estifanos Abebe

CS 544 Computer Networks

Project 3 – Network Protocol Implementation

1. Summary

Translating the design specification to an actual implementation required introducing some changes to the original specification. However, the core features of the protocol remained unchanged. The following section explains the changes introduced and the reasoning behind them.

2. Changes Introduced

2.1. Change in the PDU

In the original specification, the datagram was designed to be a concatenation of the values of all the fields of the PDU. It was assumed that the fixed length of each field could be used to extract each field from the datagram. Even though this is possible, sending a byte-encoded JSON file proved to be very simple to implement and maintain.

Moreover, in the original specification, the PDU for different types of messages was assumed to be different in structure. However, during implementation, it became apparent that using a unified datagram for all message types minimizes bugs associated with incompatible messages moving across different complex state transitions. The unified PDU for the protocol is as below:

```
{
        “mtype” : int,
        “payload”: byte,
        “protocol_ver”: str,
        “firmware_ver”: str,
        “size”: int
}
```

Figure 1: Protocol Datagram Unit

2.2. State Transition

Even though it was mentioned on the implementation plan that servers could initiate software update of remote devices, it was decided not to implement this scenario. Although this has practical importance in the real world, it requires maintaining the IP address of the remote devices to establish a connection from the server. This would have required hard coding the implementation while not adding much value to the project. Moreover, this scenario could be assumed as a subset of a client-initiated update process. Therefore, it was sufficient to implement the client-initiated update process for demonstration purposes.

2. Overview of Implementation

The following shows how the implementation is structured.

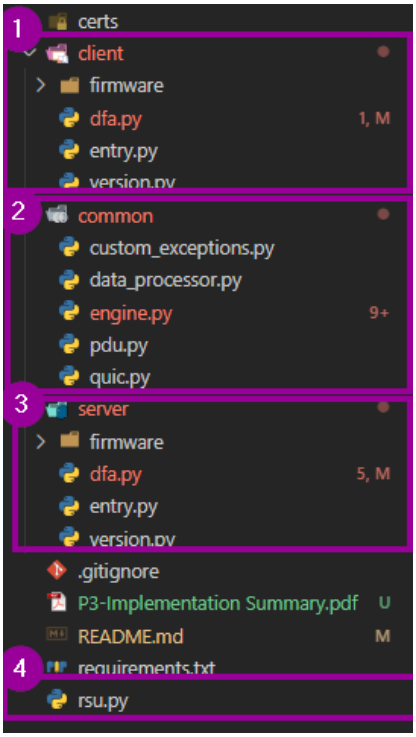

Figure 2: Project Structure

1. *Client*: Includes the implementation for the DFA of client and the versions it supports (protocol and firmware).
2. *Common*: Includes implementation of common features for both the server and client.
3. *Server*: Includes the implementation for the DFA of the server and the versions it supports.
4. *rsu.py*: Entry point for the cli based application.