【2019 Advanced Computer Networks Homework 6】

Rules:

1. Please do this homework in C language and run your program on Ubuntu 18.04.
2. Please provide Makefile to compile your homework; otherwise, you will get ZERO.
3. Do not copy homework from others (classmates, senior etc…). If this happened, you will get ZERO, whether you are either the owner of the homework or the copy-cat.
4. You have to deeply understand what your program do because TA will ask you about your program during demo.
5. If you have any question, please send email to (net_ta@net.nsysu.edu.tw) or come to EC5018, but TA will not help debugging.
6. No delay assignment is accepted. If you have trouble, please notify us in advance by email.
7. You can use C / C++ or Python and ZeroMQ .
8. You can use Mininet to do this homework .

Upload:

1. Please compress your homework into zip or tar archive.
2. Naming rules: “StudentID_TCPIP_HW6.zip”.
   For example: M063040001_TCPIP_HW6.zip
3. Upload your homework to National Sun Yat-sen Cyber University.
4. Deadline: 2019/12/16 (Mon.) 23:59

Homework 6:

One of the problems you will hit as you design larger distributed architectures is discovery. That is, how do pieces know about each other? It is especially difficult if pieces come and go; thus, we can call this the "*dynamic discovery problem*". There are several solutions to dynamic discovery. The simplest approach is: when you add a new piece, you reconfigure the network to know about it. In practice, this leads to increasingly fragile and unwieldy architectures. Let's say you have one publisher and a hundred subscribers. You connect each subscriber to the publisher by configuring a publisher endpoint in each subscriber. The very simplest is to add an intermediary; that is, a static point in the network to which all other nodes connect. In classic messaging, this is the job of the message broker. Adding a pub-sub proxy solves the dynamic discovery problem in our example. We set the proxy in the "middle" of the network. The proxy opens an XSUB socket and an XPUB socket, and binds each to well-known IP addresses and ports. This homework is to implement the following figure. Any publisher sends a message which must be received by all of its subscribers. Every subscriber can subscribe any number of publishers. So the proxy should manage each publisher's subscriber group. The it can forward messages correctly. You can use either C (C++) or Python programming language to implement this architecture.