ELEC 477

Kevin Yu: 20203451

Raatik Sharma: 20120770

Ainsley Taylor: 20210012

2024-02-05

Main.cpp

In main.cpp, a DNS server is initialized and configured with an address ("10.0.0.69"), port number (8081), and service name ("DNS"), followed by having its services started. The kVServer is initialized and configured with an address ("10.0.0.12"), port number (8080), service name ("email"), and DBM file name ("server") followed by having its services started. The kVServer2 is initialized and configured with an address ("10.0.0.10"), port number (8089), service name ("taxi"), and DBM file name ("server2") followed by having its services started. The first client (kvClient) is initialized and configured with an address ("10.0.0.14"), and it sets its service name to ("email") so it can talk to the email service server. The second client (kvClient2) is initialized and configured with an address ("10.0.0.11") and sets its service name ("taxi") to communicate with the taxi service server.

The importance of the main method is to test two separate clients talking to two different service servers by using a specified service name to find the corresponding IP address of the service server.

Kvclientstub.cpp

In kvClientstub.cpp, a method called dnsLookup() was added that takes a serviceName parameter. This parameter is used to translate the service name into the corresponding service server IP address and port number. The searchService() method from dirSvcClientStub takes the svcName parameter to search for the server information and saves it to searchResult. If the status for searchResult is true, the server information is available, and the server name and port number are updated. Lastly, the search result that has the server information is returned.

Additionally, a setSvcName method is added to allow the clientStub to store the service name that the client will talk to as an attribute of the class. This service name allows the client stub to pass this information to the DirSvcClientStub instance for DNS lookup.

Lastly, in the init() method, the dnsLook up method is callled immediately to configure the destination address for where the clientStub will be communicating to before it does any Key value commands RPCs.

KvService.cpp

In KvService.cpp, the only new feature added was a method called register service, which is used to register the service server's service name with its corresponding server address and port number in the DNS server.

This method is called at the beginning of the start method so the server can register its service to the DNS before listening for requests from clients.

DirSvcClientStub.cpp –

This class is a client stub class that will interact with the directory service server. It has a method called setScvName, which sets the service name. It then has a function called registerService, which registers a service with the DNS server. In this method, init is called if needed. It then gets the current value of serial for the request. The function then has a retry mechanism (at least once) for failures with communication with the socket, with a maximum number of retries being 5. A request is then created called dirSvcRequest, and the register request is also created. The message is then serialized and sent to the DNS server. The method handles send errors after this process. Next, the method receives a response; if it fails, the retry loop runs again. The received response is verified, and a success status is set if the verifications succeed. Otherwise, the retry loop runs again.

The next method in DirSvcClientStub is the searchService method. This method searches services based on name and sends a search request to the directory service to retrieve the server name and port. The method calls init if needed. It then creates a message containing a search request to send to the DNS server. If the message is too long, it returns an error. The message is then serialized and sent. The method receives a response from the DNS server in a loop that will retry for errors up to 5 times (at least once semantic). It then verifies the message and gets the information to return.

The next method implemented is an init method, which configures the directory service client stub to talk to the DNS server. It does so by configuring the destination IP address and port. The method configures the socket and then looks up the address by name. The method then performs DNS resolution and extracts the IP address. After, a socket file descriptor is created with a timeout set to 1 second if a reply is not received.

The method deleteService was then implemented, which is used for deleting a service from the DNS based on a provided service name. The init method is run if needed. Then, a message containing a delete request is made to send to the DNS server. If the message is too long, an error is returned. The message is serialized and sent. The method receives a response from the DNS server in a loop that will retry for errors up to 5 times (at least once semantic). It then verifies the message and extracts the delete status.

DirSvcService.cpp

The class DirSvcService is the server stub for the directory service server. It has a stop method which stops the server from processing requests. It then contains a setPort method that sets the port on which the dirSvcService will listen.

There is a start method where the socket information is configured and listens for incoming requests from the DirSvcClientStub. The method binds the socket with the server address and then waits for a message from a client. When a message is received, the method parses it and calls the corresponding method. Finally, the method returns the response to the client stub.

The next method implemented is a callMethodVersion1 method which determines what message was requested by the DirSvcClientStub (either search, register, or delete). The method then calls the appropriate method.

The following three methods are used to deal with the DirSvsClientStub request. The searchService method searches through the unordered map to find the corresponding server name and port based on the service name. The registerService method registers the service into the unordered map and stores the service name as the key and the server name and port as the value. The deleteService method deletes the service from the unordered map to remove the service from the DNS.

DirSvcServer.cpp

The DirSvcServer class deals with the directory service server tasks. It contains a start method, a setPort method, which sets the port for the directory service server, and a setSvcServer method, which sets the service name.

DIRSVC.proto

The proto file defines the new data serialization models for what data will be transferred over the network between DirSvcService.cpp and DirSvcClientStub.cpp. This file is very similar to the last protofile implementation for the KVservice serialization and deserialization. But the key differences are that the values being serialized and deserialized are a server name and a server port rather than arbitrary number of bytes (string). Also, the key is no longer an int but instead a string. Lastly, The proto-file has request and response data structures for search, register, and delete service.