Team Dijkstra

Crist – Ghodratnama – Howell – Sung

t12 – Progress Report D

Software Engineering II

Spring 2013

Completed Tasks

Documentation Redesign

The need to normalize the documentation became evident upon referencing the documentation. There appears to be too much information which is not used or not really actually needed in the documentation. Essentially the user needs to know the inputs and outputs of a module and not particularly the minute details of the code contained within. Change Logs and specific function information is not to be contained within the code file itself and the documentation is not done on a per file basis but rather how the module connects to the application and the flow of the data present. Input is generally XML on a specified port from a specified module or some particular syntax for XML that is defined by DTD's. Previous documentation did not emphasize this enough, thus a rewrite was necessary.

User Verification Module (Server Side)

User security was an issue on the server side that needed to be addressed. Because anyone could access any file given they know the user exists on the server, they could acquire their inbox directly calling port 20006 for email reception. While this is still in place, a user verification action on the user module was written to enforce this policy at a later time that the email transmission mechanics are in place. The server verifies the user and posts back a server message to the client and writes to the port that is listening for email reception, the location of the user on the network. When the user verification module is called by the client, it opens a listening port on its local self for the reception port and the server message port. If the message received by the server is ACCEPT, then the transmission process will begin. If the message is REJECT, then the port is shut down and the server does not even attempt the mail reception action on the server side. The client may attempt to force the reception of data or keep the data port open, but it will continue to listen and not receive any data and simple stay idle until a user is verified at that location.

This also required that password mechanics be implemented on the server side as well. Though these passwords are stored directly in the flat XML address-book file, security can be enforced based on the RWE properties of the UNIX file system to ensure invalid access does not happen. Since the server machine is reading this file in su mode (typically) we can enforce a "700" chmod property on the file and get the desired results.

Server Messaging (Server Side)

The need for a communication mechanism between the client and server was needed when the user verification module was developed. This would ensure that the client knew what action had occurred, on whether the process it was attempting was successfully on not. This also requires that the client be constantly listening (on port 20004) for messages from the server. After these message are received, then that information can determine what is the next step to take without assuming everything went smoothly, thus resulting in client side hangs because the server refuses to respond. This allows us to create error message dialogs on the client side based on server response and keep the client informed as a result. This was not determined to be in module form since every module would need to invoke it at some point. Thus is was pushed to the "include" folder on the server side.

Mailing List (Server Side)

The fundamental logic for the mailing list module was developed to allow for the storage of mailing lists and the verification of the administrator for that mailing list. This module will invoke user verification module and then the actions taken upon the module itself, thus creating the dependence on the user verification process. All actions taken on the mailing list have not been implemented yet, but the data structures and logic behind the module can be tested for logical soundness.

Current Tasks

Client Side GUI

The client side GUI, which will be the interfacing point for the email client, is currently in construction. The client has the ability to manage a simulation where the client can write and email and send that email, which is placed into an inbox after being processed by the client side ACL2 module. (Demo available) Functionality is kept simple at this point and the user does not have to invoke the verification process to send these emails. Eventually the client GUI will be plugged into the networking communication and the data will be processed on the server side and routed to the appropriate store folders after which the user will be able to request the reception of that information.

Network Communication

The transmission of the data across the network has begun. Machine identification appears to be an issue at the current point in time, but this will be overcome by implementing an isolated where network addresses are communicating on a short distance without the worry of firewalls and other security measures on the OU campus network. Thus far, communication has occurred successfully via desktop -> laptop communication on an isolated network, but we have been unable to transmit information across the OU network. This may be based on machine identification by the network itself.

Planned Tasks

User Verification

It is ideal that the client be verified anytime it needs to implement an action on the server side. As a result, most communication should be done though the user verification port and the server side ports should only be opened when the user is correctly verified, thus kicking off the module process. This security measure will ensure that the server cannot be exploited by information that is forced by the client side and we are “expecting” information to be received instead of assuming, at some point, the information will be received by “some” client.