Name: Omkar Joshi
Student number: 19200394

Password Management System

Introduction:

This project comprises of Password Management System done using Bash. In this project it acts as one place to save user all password regarding particular domain. As it's very tedious to remember each password.

What System Should Do?

- The system at the start should create folder according to user name and within that folder the user create different services as folder and within this folder user can store respective values.
- For example: John/Bank/aib
- As User folder can have multiple filename and values so data can be segregate according to that
- Moreover in this file it only contain login id and password
- The system should be synchronized as at one time one user can access its file.
- The system should contain proper error handling about the messages to let the user know if something went wrong.

Architecture Design:

- This project is comprised of total nine scripts like init.sh, insert.sh, ls.sh, rm.sh, show.sh Server.sh, client.sh, P.sh and V.sh
- The communication between client side and the server side are done via pipes

1) Init.sh

- In init.sh new user can be created via passing username as argument while running the script
- If the username is already present the system will return certain message to the user
- If the username is not present then the system will create directory as the argument name which pass through it.
- The system also check if more than one argument is pass to the system. If found it will return appropriate message.

2) Insert a service (insert.sh)

- In this project this script very vital role as when this script create a login id and password operation
- To run this script it requires 4 argument to pass to it
- The first argument contains username , second one is directory and file path , third argument is just empty string and last one is the payload

Name: Omkar Joshi
Student number: 19200394

- The script first check every command and argument and according to that it will run and perform the given operation
- Note this script is modified when you call from client side as it will ask user to fil it via prompt
- The script also check that if the user is exist then do further operation it will divide the structure with '/'
- This project the call to run this script is by first triggering client file that will trigger the script file through pipe and this script file is called the insert.sh file.

3) Update a service (same as insert.sh)
- To work this function, it will call the same script as insert sh
- As There is no new script for editing a service as it is implemented in insert.sh. Here if f is passed it will call the insert script
- This file check and if the required argument is passed it will update the services.

4) Remove a service (rm.sh)
- Services can be deleted using rm.sh which requires username and service name as mandatory
- Error is produced if service name is not present.
- The server.sh internally calls rm.sh with appropriate arguments to remove a service. Service is removed using rm -rf "servicename" command.

5) Show a service (show.sh)
- Login-password details can be shown using show.sh with username and service name as mandatory arguments.
- Error is produced if total number of parameters are not equal to 2.

6) List services (ls.sh)
- This file is checked and check if it contains desired parameter
- It uses tree command to print the output

7) Synchronization (P.sh and V.sh)
- If multiple clients are trying to access same file, only one should succeed in doing that. This is done by the means of semaphore. Here the lock is based on username.

8) Server.sh
- The server is continuously running by reading from the server pipe.
- The server.sh contains a switch case required
-. The server reads data from server.pipe and calls the scripts

9) client.sh :
- Client.sh uses a switch
- For the client-server communication, a pipe by the name of clientname is always created. The client side data is pushed onto the server.pipe. Then the client waits for response from the server side. After reading the output from server.pipe, it is shown on client end. Lastly, the pipe created by the client is deleted. case to identify the commands. Also, while invoking the script, the parameters length should

Name: Omkar Joshi
Student number: 19200394

Conclusion

The Password Management System stores login-password details. Overall it was a tough project to do but I learned a lot regarding how Bash works