**DLCT.cpp**

Main Process

**int \_tmain(int argc, \_TCHAR\* argv[]) {**

**loadProperties(); //in to PropertiesLoader.cpp**

**loading all configuration details from VCR.txt, which is placed in release folder.**

**Actually VCR.txt should be at the place from where final DLCT.exe is running, because our DLCT.exe(final executable) file look for VCR.txt in folder from where it is running and load the configuration details.**

This thread will die after transferring old log files

Initial FTP Thread

**\*\*\*\*\*\*\*\*\*\* Thread has been initiated \*\*\*\*\*\*\*\*\*\*\***

**hIOMutex = (HANDLE) \_beginthread(ftpLogInitial, 0, NULL); // ftpLogInitial(); // in to InitialFtpFunction.cpp**

**systemPropertiesStructureObj=getSystemProperties(); // in to SystemPropertiesResolver.cpp**

**to get local system information in to systemPropertiesStructureObj and this is copied in to registerRequestJsonStructureObj and this is passing parameter to**

**registerRequestJson=getRegisterRequestJSON(registerRequestJsonStructureObj );// to get JASON string in to JsonFunctions.cpp**

**“registerRequestJson” will be converted to STRING “registerRequestCstring “, this will be the passing parameter to**

registerWS() will create one thread “ftpThreadInitializer” to transfer log files for new user and It will wait for **ftpLogInitial() thread to complete.** if registering client will success.

**registerWS(registerRequestCstring);// in to RegisterClient.cpp**

Retrieve Content Thread, to receive Device and firewall commands

**\*\*\*\*\*\*\*\*\*\* Thread has been initiated \*\*\*\*\*\*\*\*\*\*\***

**\_beginthread( retrieveContentWS, 0 , NULL );// in to RetrieveContentClient.cpp**

MonitorFileChange() Thread will monitor Arc folder @ C:\Windows\security\logs\ **, any changes to file notification.log in to that will popup you notification balloon.**

**\*\*\*\*\*\*\*\*\*\* Thread has been initiated \*\*\*\*\*\*\*\*\*\*\***

**\_beginthread( MonitorFileChange, 0 , NULL );**

While loop to keep process Alive

**InitialFtpFunction.cpp**

**ftpLogInitial(); will be running in to thread**

**ftpLogInitial(); has been used to read data from register.txt file in order to send information to server is client is not running for the first time.**

**It will read information from the register.txt file and try to send the “log files”(example- DLCT2014110322.log). it will look for the files in to – “C:\Windows\security\logs” if any “DLCT\*\*\*\*\*.log” is present in that folder it will send it to server as per the information stored in to register.txt(information of last user to that system)**

**ftpLogInitial(){**

**registerResponseObjStuctureObject =extractRegistrationDetails(); // in to Storage.cpp**

**copy details from registerResponseObjStuctureObject to registerResponseObjStucturePtr**

**ftpNetworkDataIteratively(registerResponseObjStucturePtr); // in to FtpFunctions.cpp**

**}**

**RegisterClient.cpp**

**RegisterClient.cpp(){**

**//while loop until registering client get success**

**While(){**

**//webservice for specific register details to registering machine(client)**

**//if Success Storing register response to registerResponseObjStucturePtr and pass it to ftpThreadInitializer();**

**ftpThreadInitializer (); will be running in to thread**

**HANDLE handleValue = (HANDLE) \_beginthread( ftpThreadInitializer, false, (void\*) registerResponseObjStucturePtr);// ftpThreadInitializer() in to FtpFunctions.cpp**

**}**

**}**

**FtpFunctions.cpp**

ftpThreadInitializer(){

ftpNetworkDataIteratively(param);// Same function will be called by **ftpLogInitial();** But it will come out of while loop depends up on **“iterateTime” = “** **registerResponseObjStucturePtr->ftpInterval;”**

**The above veriable will be “-1” for ftpLogInitial(); ,** So we can differentiate when to be run in to loop and when only once.

}

**RetrieveContentClient.cpp**

processRetrieveContentWS(){

while(){

//Webservice is called @ specific Interval

// RetrieveContent @ specific Interval

//if any data receive will be run. Either Device or firewall command will be executed.

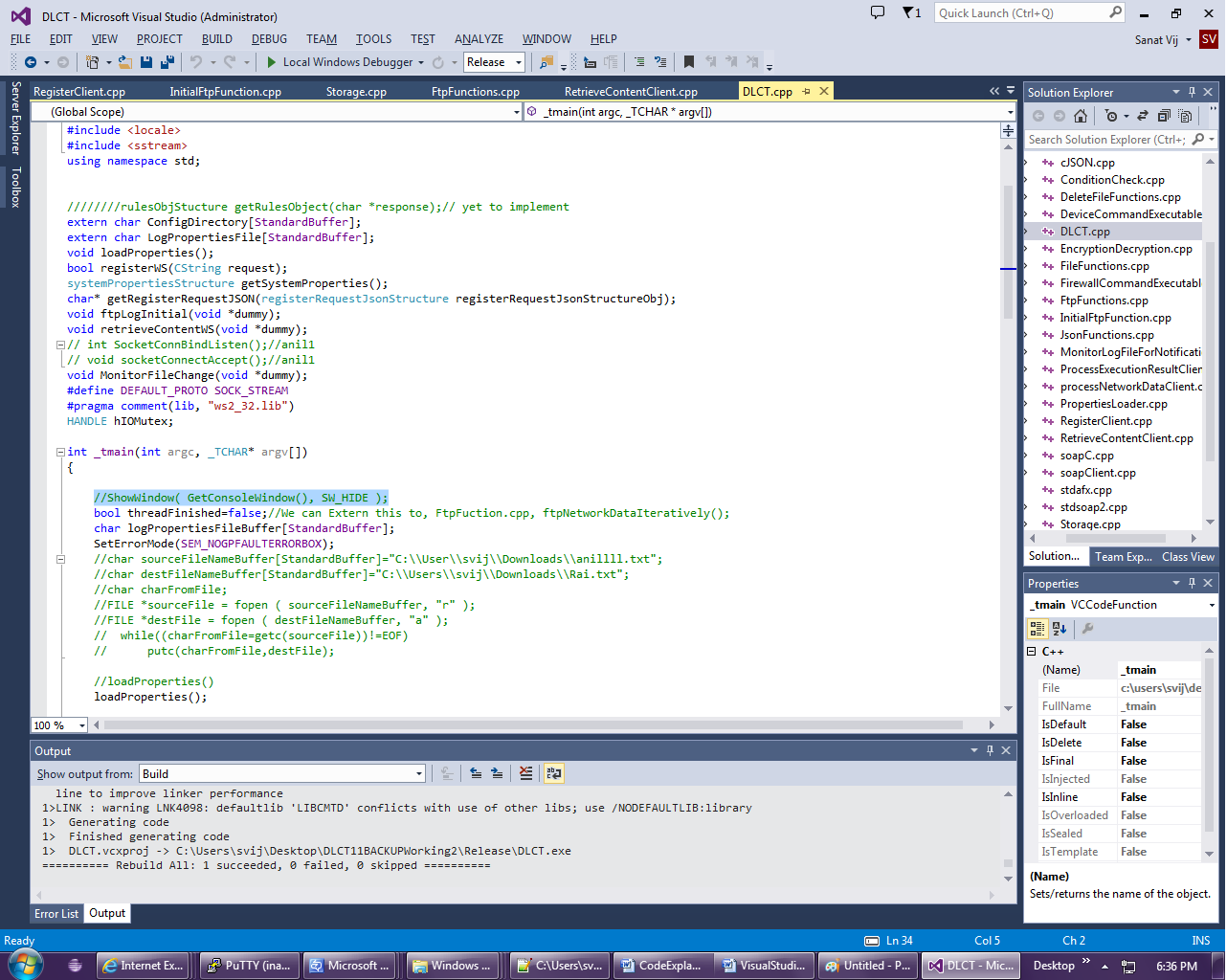
//if no command to be execute and received string is “SYNCHRONISED” it will wait for specific amount of time and again request for RetrieveContent.

}

}

TO See Executable window of DLCT.exe(To Enable and Disable)

Just comment or uncomment “//ShowWindow( GetConsoleWindow(), SW\_HIDE );” line in to DLCT.cpp. As shown below….



## DLCT file list written by developer

1. Conditioncheck.cpp (40)
2. DeleteFileFunction.cpp (63)
3. DeviceCommandExcutable.cpp (109)
4. DLCT.cpp (135)
5. Encryptiondecryption.cpp (110)
6. FileFunction.cpp (150)
7. FirewallCommandExcutable.cpp (74)
8. FtpFunctions.cpp (327)
9. InitialFtpFunction.cpp (54)
10. JsonFunctions.cpp (366)
11. MonitorLogFileForNotification.cpp (150)
12. ProcessExecutionResultClient.cpp (95)
13. processNetworkDataClient.cpp (150)
14. PropertiesLoader.cpp (280)
15. RegisterClient.cpp (197)
16. RetrieveContentClient.cpp (344)
17. Storage.cpp (64)
18. StoredCommandsProcessor.cpp (117)
19. SystemPropertiesResolver.cpp (77)

**Header file**

1. commandLinkedList.h (5)
2. SubsequenteRequestStructure.h (7)