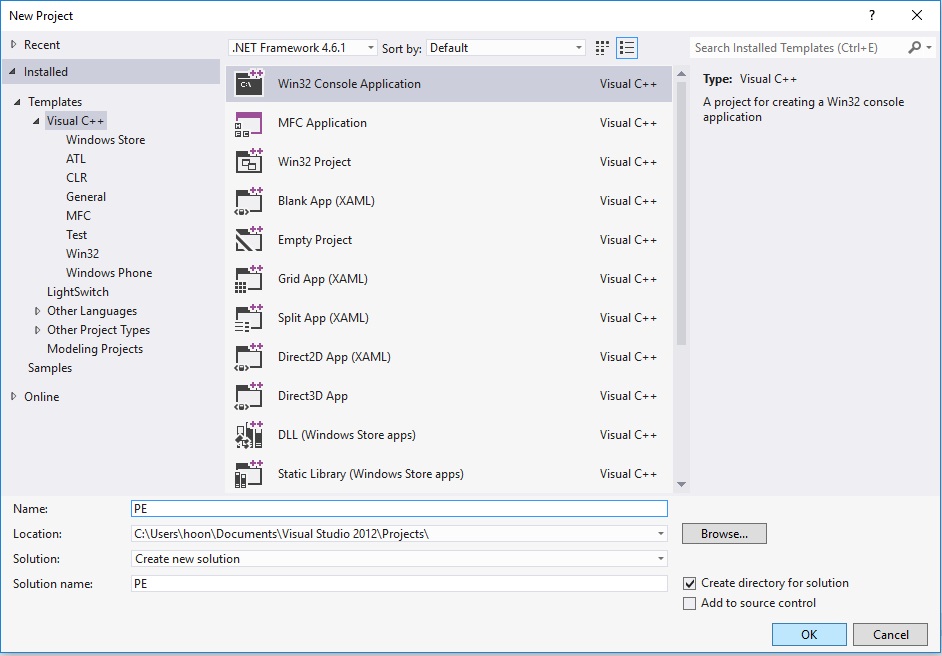
**Report for compiling the PE (Predicate Encryption) program:**

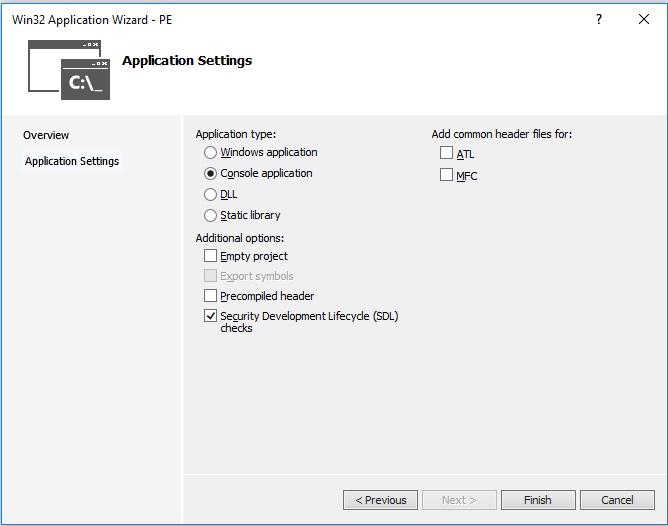
The program was compiled in Visual Studio 2012 and the operating system was Windows 10. In this report we will see how to compile the predicate encryption scheme. Which is attribute-hiding under the DBDH and P-DBDH assumptions. The present predicate encryption scheme should be allowed to encrypt a message as well as the attributes.

* Start with creating a new win 32 Console Application. Write project name “PE” and solution name “PE” and click ok.



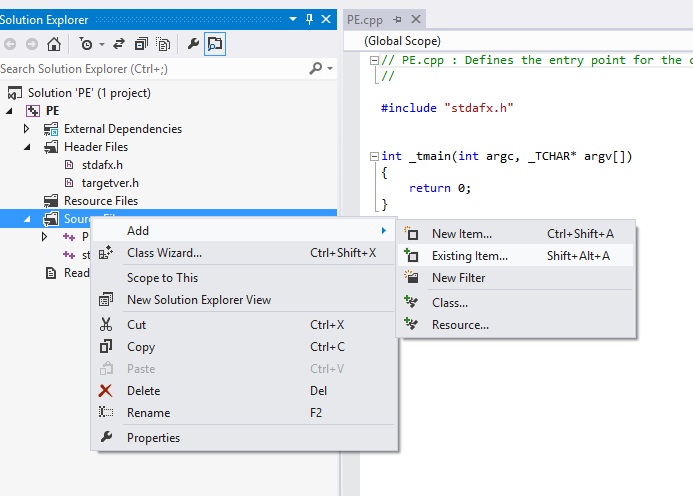
­­

* Now check Console application and Security Development Lifecycle, uncheck the “Precompiler header” in program settings and click finish. You can go to program setting from the left panel as show in the figure below.



* Right Click the project “Source File” in the left panel and got to: Add🡪Existing Item.

Click “Existing Item”.



* + For MR\_PAIRING\_CP curve

cp\_pair.cpp zzn2.cpp big.cpp zzn.cpp ecn.cpp.

# For MR\_PAIRING\_MNT curve

mnt\_pair.cpp zzn6a.cpp ecn3.cpp zzn3.cpp zzn2.cpp big.cpp zzn.cpp ecn.cpp.

# For MR\_PAIRING\_BN curve

bn\_pair.cpp zzn12a.cpp ecn2.cpp zzn4.cpp zzn2.cpp big.cpp zzn.cpp ecn.cpp.

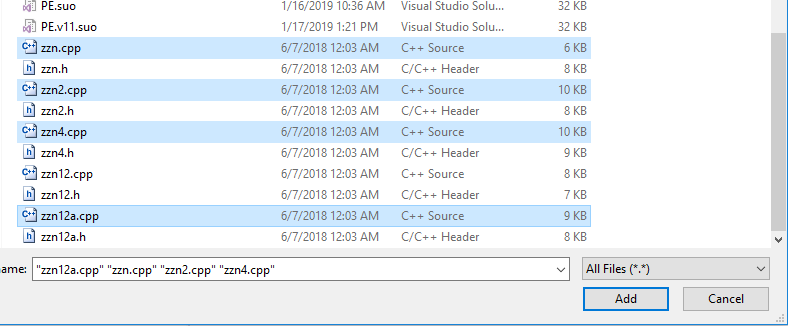
# For MR\_PAIRING\_BLS curve

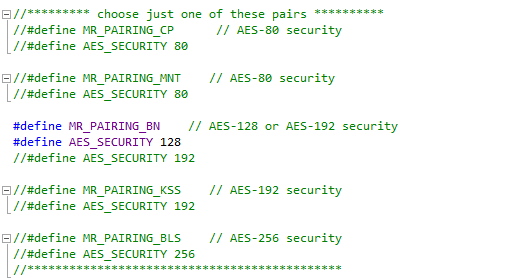
bls\_pair.cpp zzn24.cpp zzn8.cpp zzn4.cpp zzn2.cpp ecn4.cpp big.cpp zzn.cpp ecn.cpp.

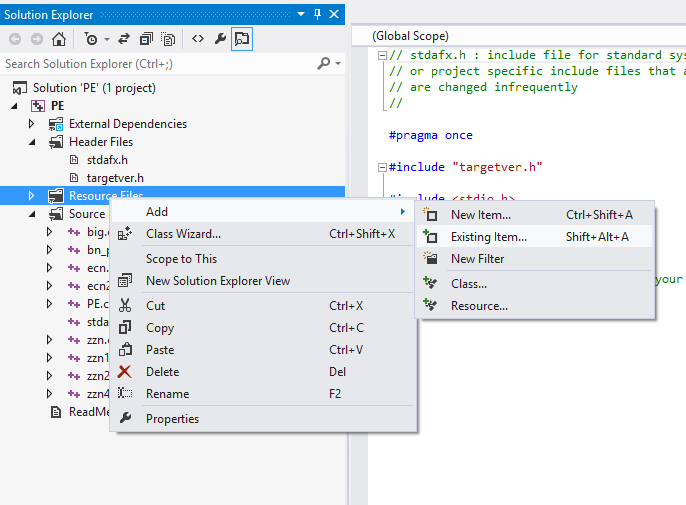
# For MR\_PAIRING\_KSS curve

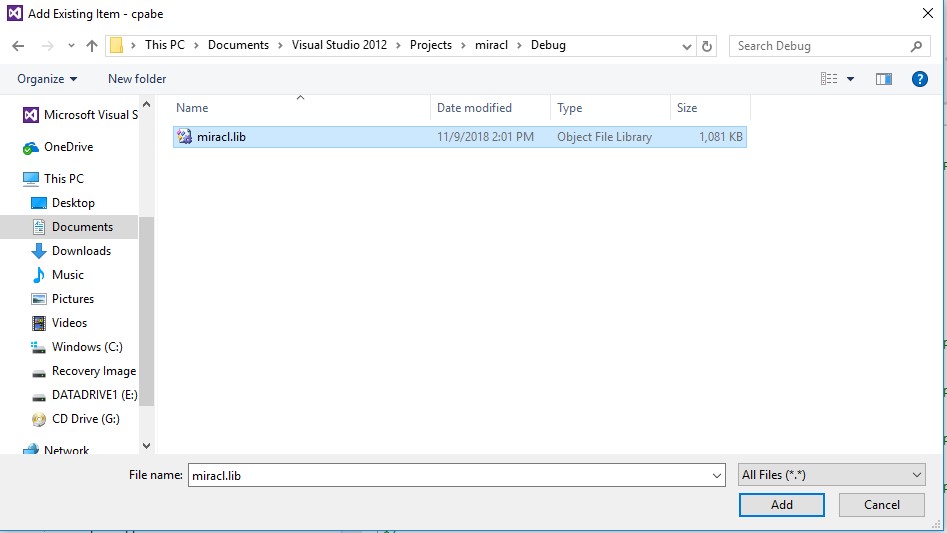
kss\_pair.cpp zzn18.cpp zzn6.cpp ecn3.cpp zzn3.cpp big.cpp zzn.cpp ecn.cpp.

Note: Code for PE.cpp file is not present in default miracl distribution. Write down all the code provided at the “program code” section of this report, in to the already existing PE.cpp file.

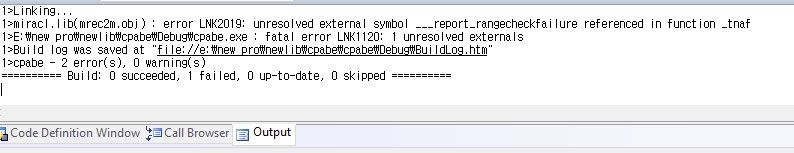


* To choose a pairing do the following:
  + Open the PE.cpp file. Go to the code section as shown in figure below. (Code has been copied in the PE.cpp file in above Note).
  + Just uncomment the type of pairing and the security you want to select.
  + Like in this case the pairing that is chosen is “MR\_PAIRING\_BN” and the security chosen is “AES Security 128”. As shown in the figure.
  + Note that “//” is used to comment in C++, to uncomment just remove the “//” from in front of the line you want to uncomment. Like in this case we have removed “//” at the starting of “#Define MR\_PAIRING\_BN” and “#Define AES\_SECURITY 128” 
* Add the library file "miracl.lib” in resource folder. Right Click on resource folder and go to ADD🡪Existing Item.



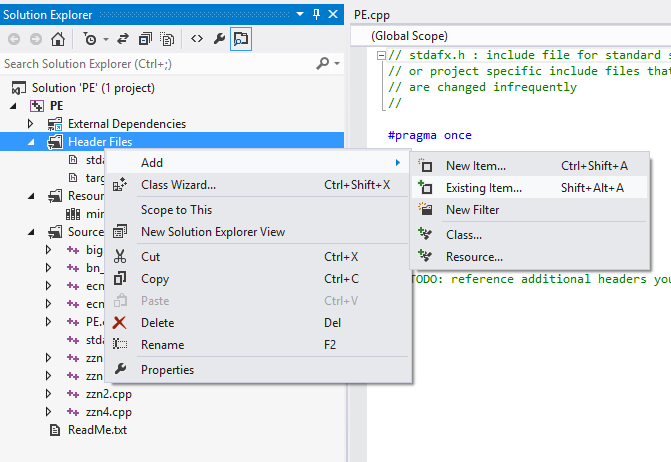


* We added “miracle.lib” to resource files to avoid the errors like error LNK2019, error LNK2001 as shown in figure below.

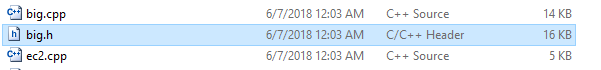


* Right Click the project “Header Files” in the left panel and got to: Add🡪Existing Item. Click

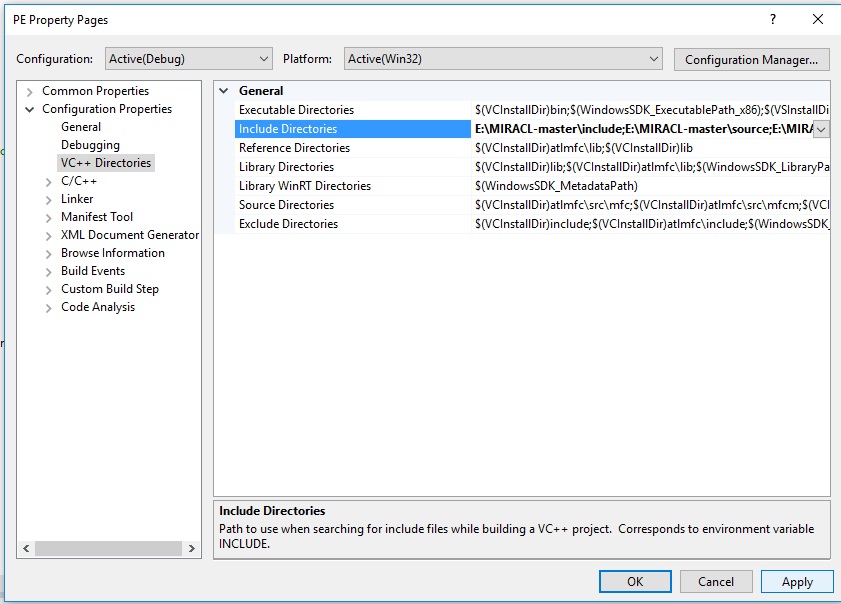
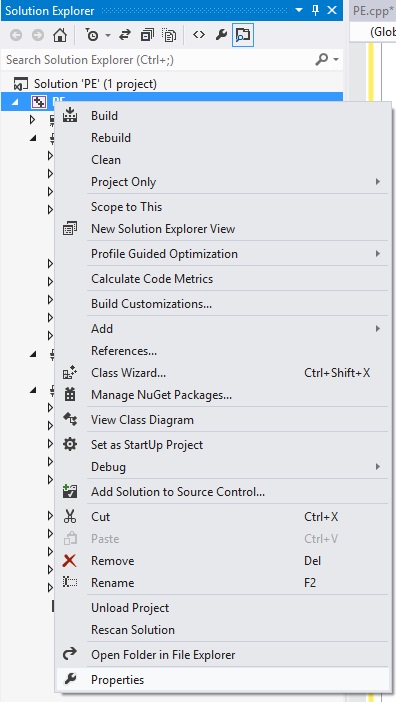
“Existing Item”.



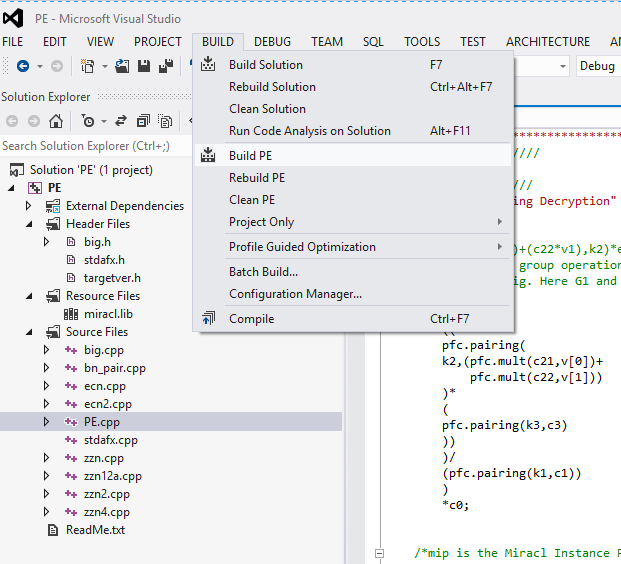
* Include the “big.h” file from miracl distribution.



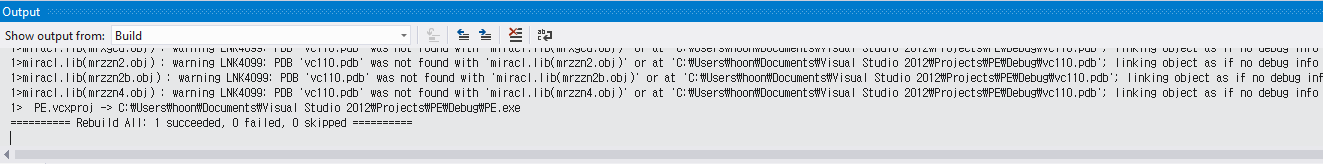
* Go to the properties🡪Configuration settings🡪VC++ directories🡪Include directories. Click on the drop down menu and select the “pairing” folder from miracl distribution.
* Go to the properties🡪Configuration settings🡪VC++ directories🡪Include directories. Click on the drop down menu and select the “include” folder from miracl distribution.
* Go to the properties🡪Configuration settings🡪VC++ directories🡪Include directories. Click on the drop down menu and select the “source” folder from miracl distribution. Then click on APPLY and OK.



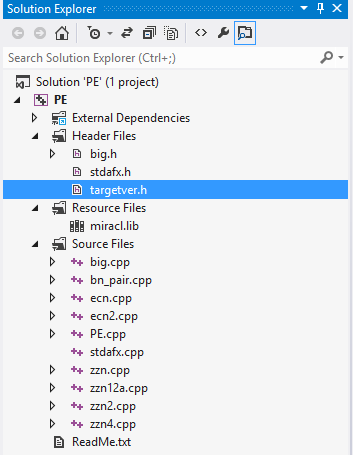
* Build the program from the Build Tab. Click Build PE.



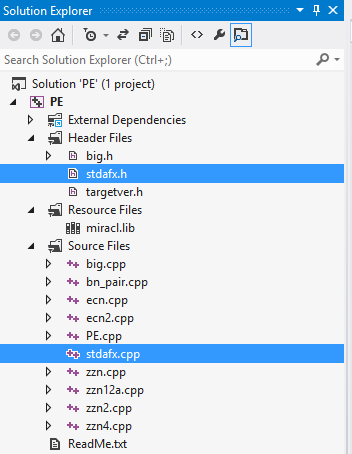
* The program should successfully build like below.



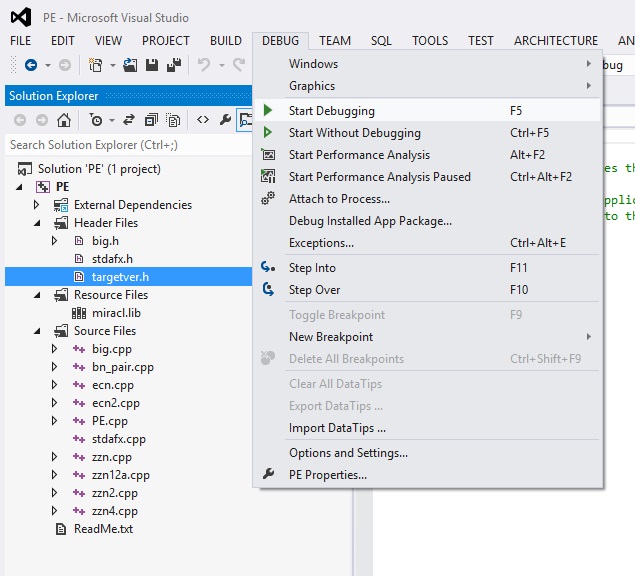
**Note**: From the solution explorer make sure that program has "targetver.h" in header files as show in below figure. As this file Include SDKDDKVer.h defines the highest available Windows platform. If you wish to build your application for a previous Windows platform, include WinSDKVer.h and set the \_WIN32\_WINNT macro to the platform you wish to support before including SDKDDKVer.h. “targetver.h” and “SDKDDKVer.h” are used to control what functions and constants are included into your code from the Windows headers, based on the OS. The “targetver.h” sets defaults to using the latest version of Windows unless thedefines are specified elsewhere. These two files (targetver.h and SDKDDKVer.h ) are auto generated when you create the project, you do not need to manually add them to the project.



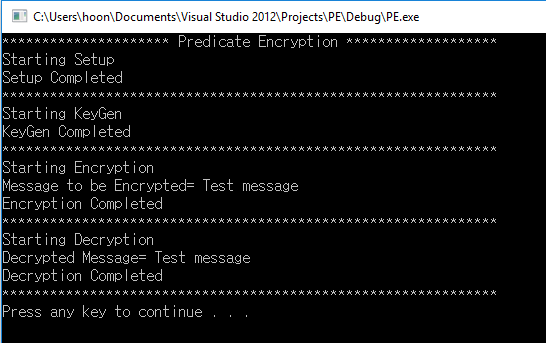
* Also make sure from the solution explorer that program has "stdafx.h" and "stdafx.cpp" as "stdafx.h" as show in figure below. It's a "precompiled header file" any headers you include in stdafx.h are pre-processed to save time during subsequent compilations. These two files (stdafx.cpp and stdafx.h) are auto generated when you create the project, you do not need to manually add them to the project or to create them.



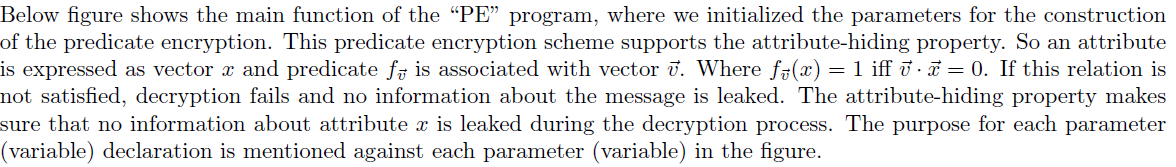
* Now click on Debug and click on Start debugging.



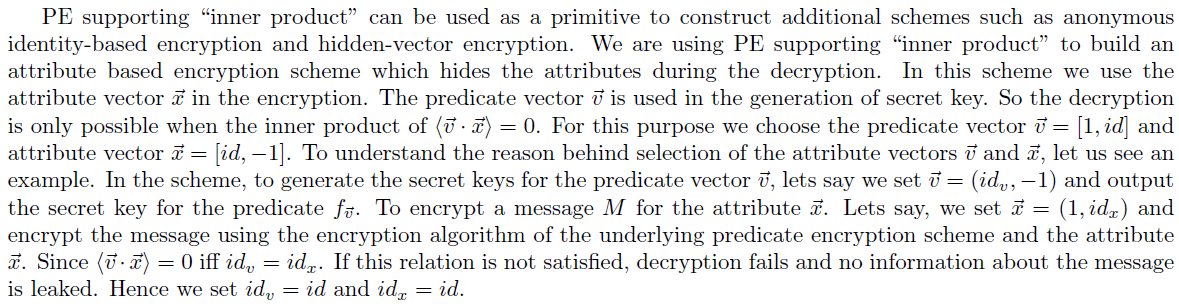
* When you will click on start debugging after successfully building the program. The program will output the following, as shown in figure below.

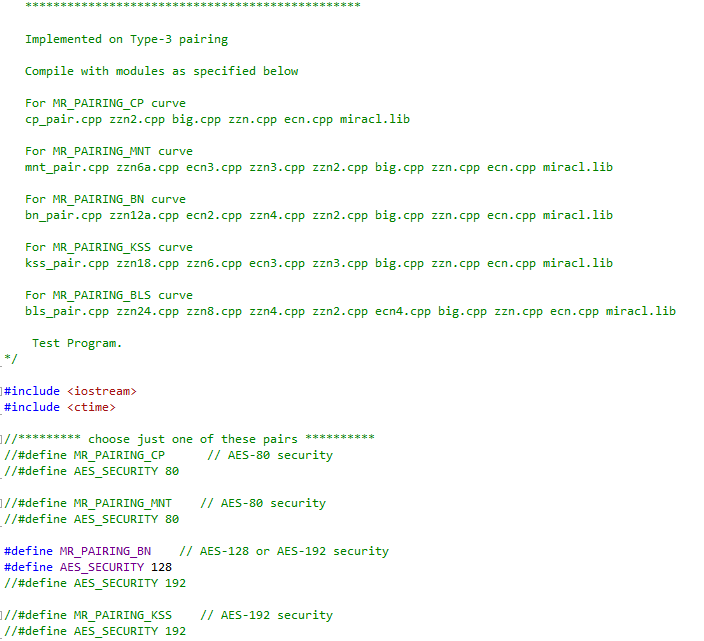


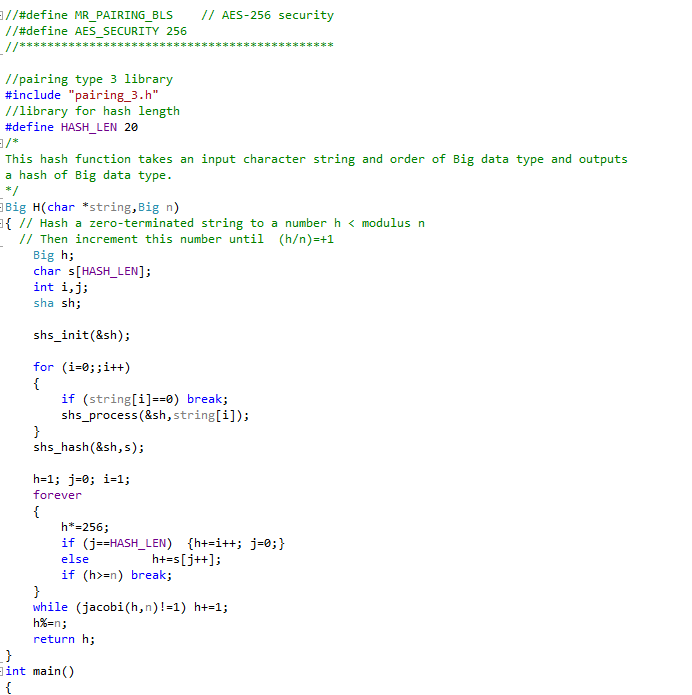
**Program Code**

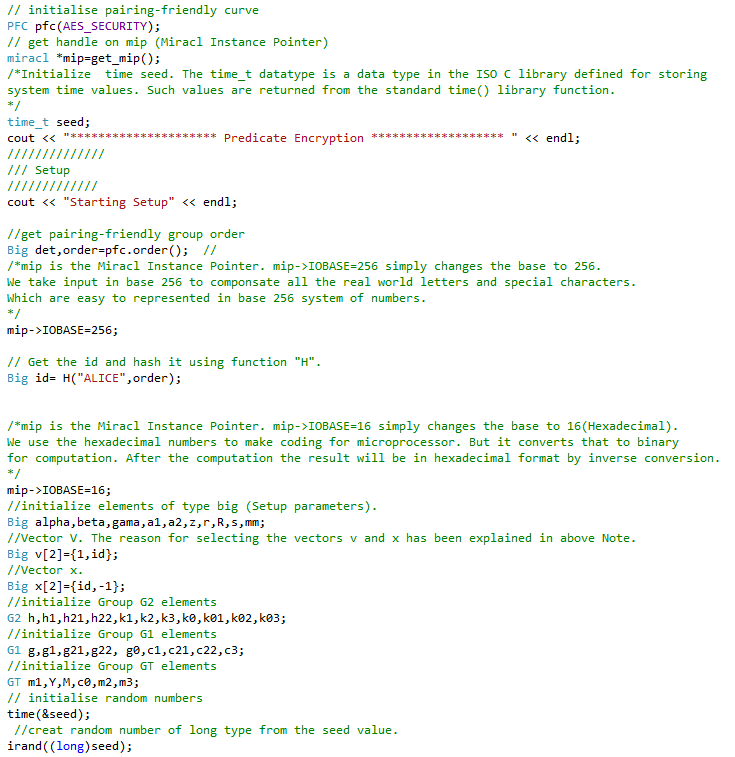


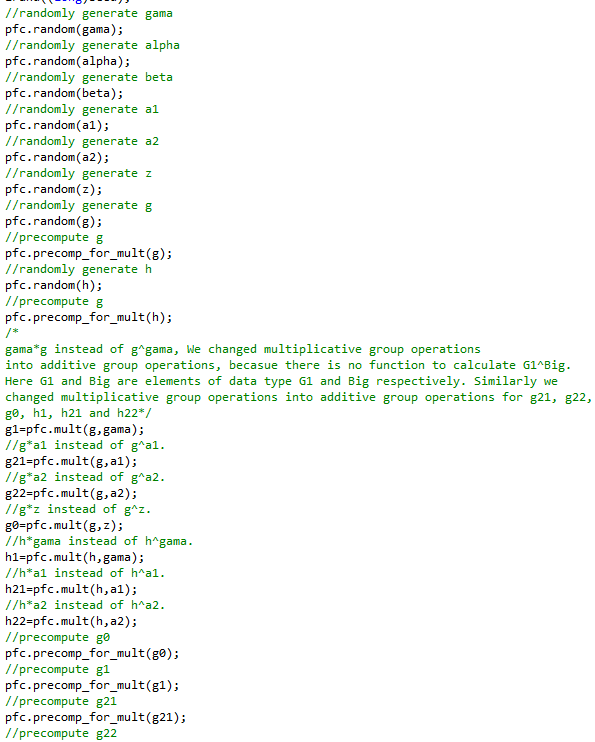
Note:

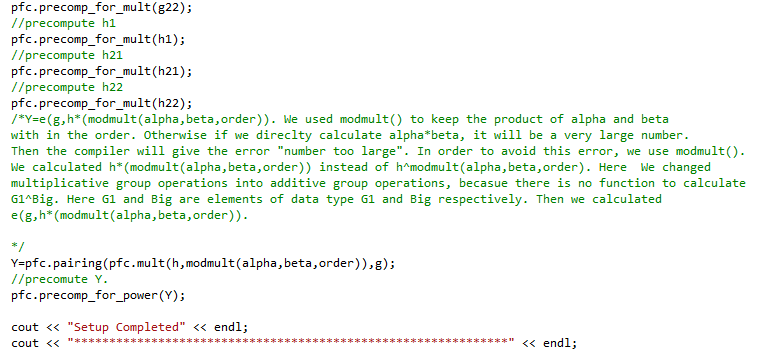




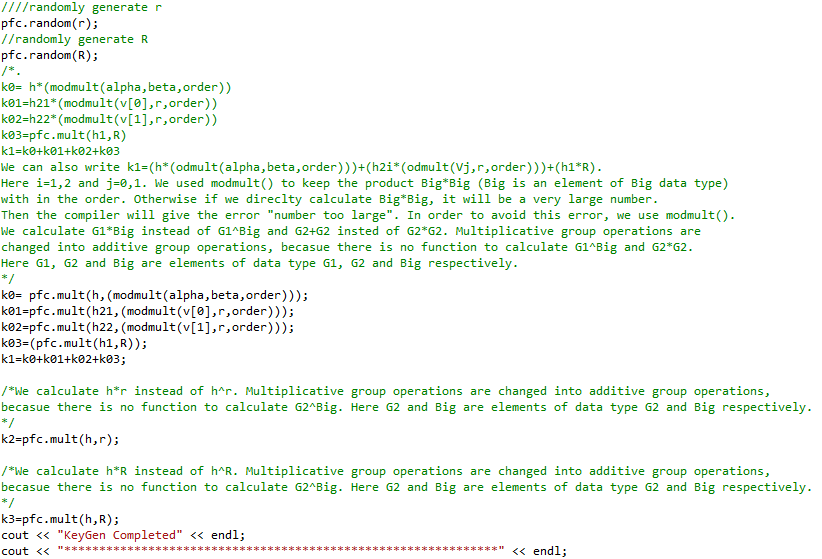




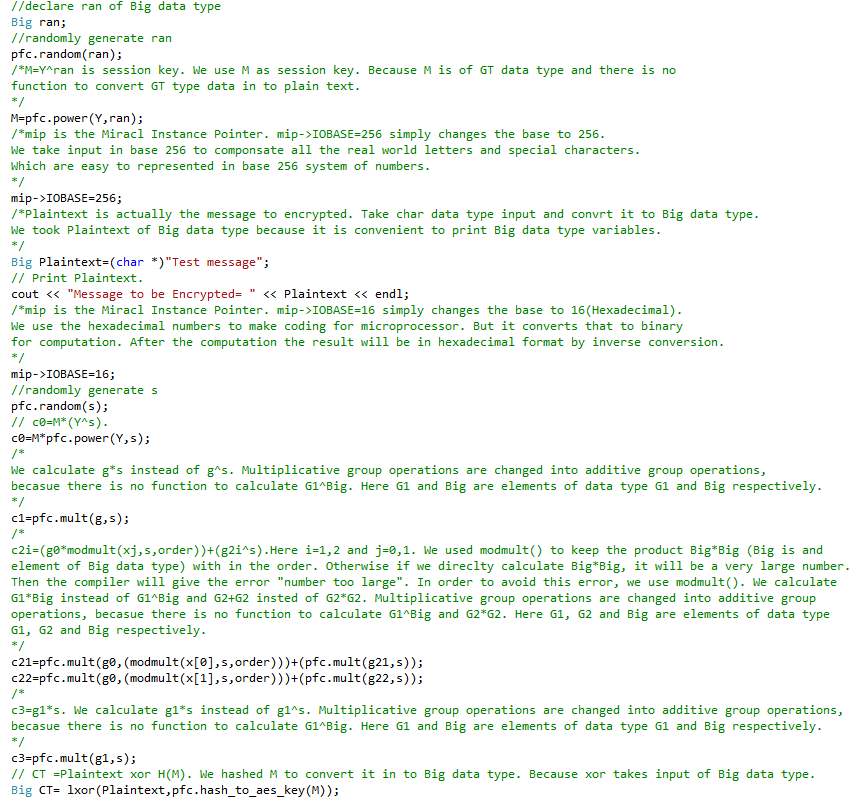




* **Key Gen:**



* **Encryption:**



* **Decryption:**  