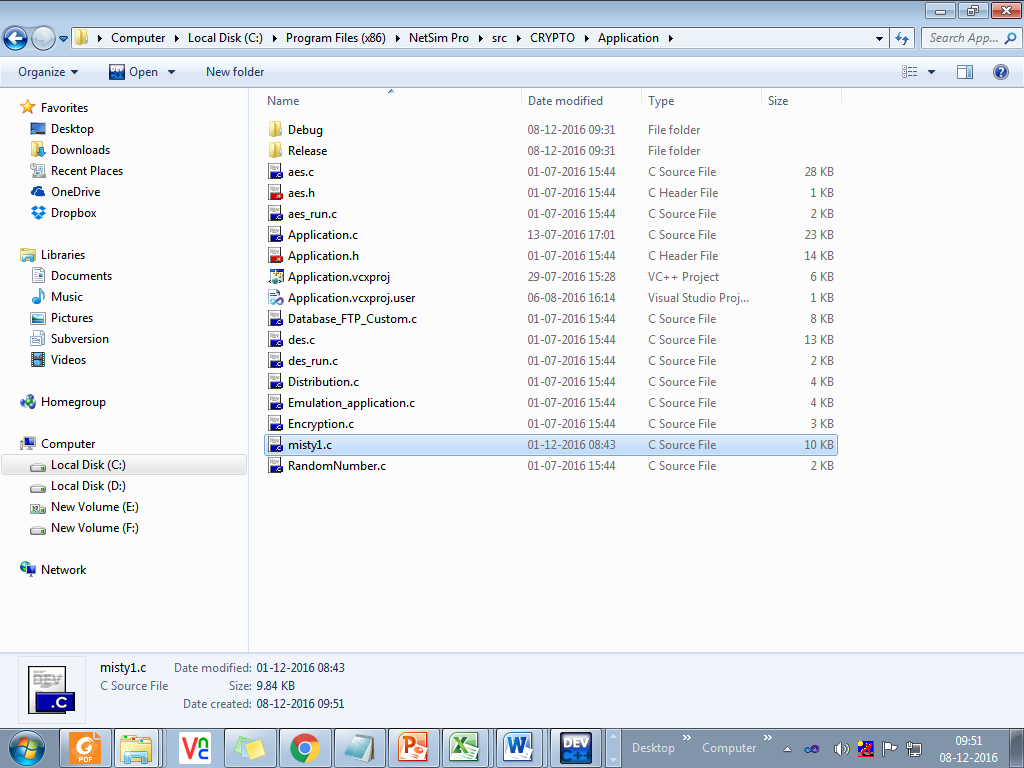
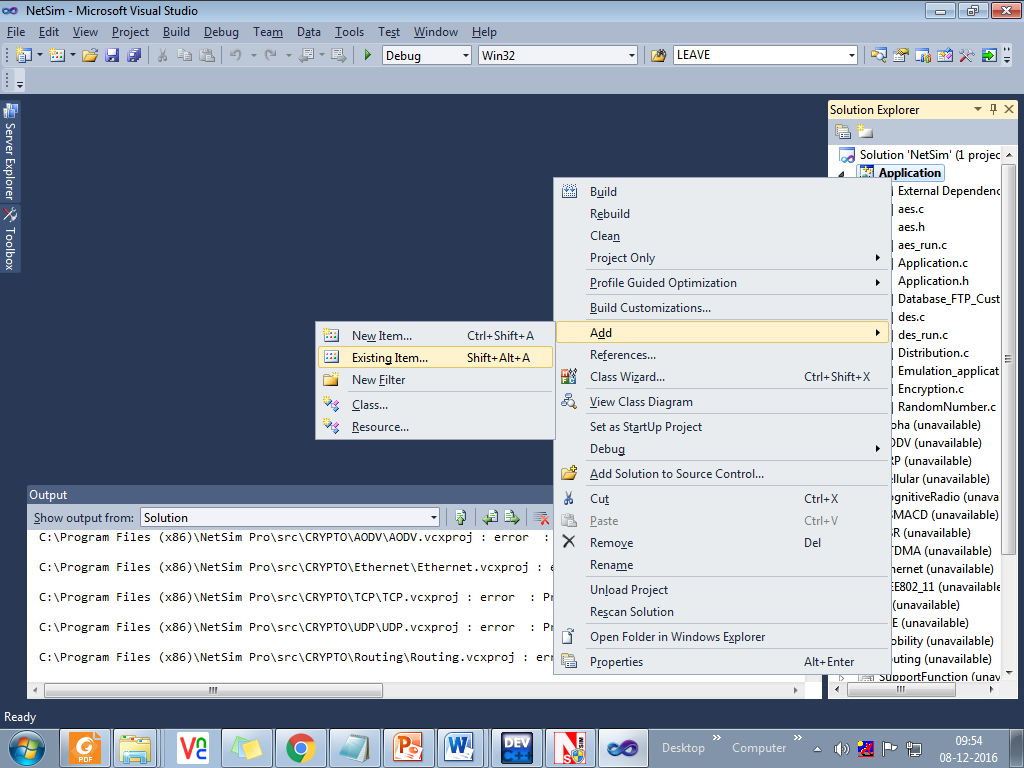
Implementing a new Crypto Algorithm – Mysty1

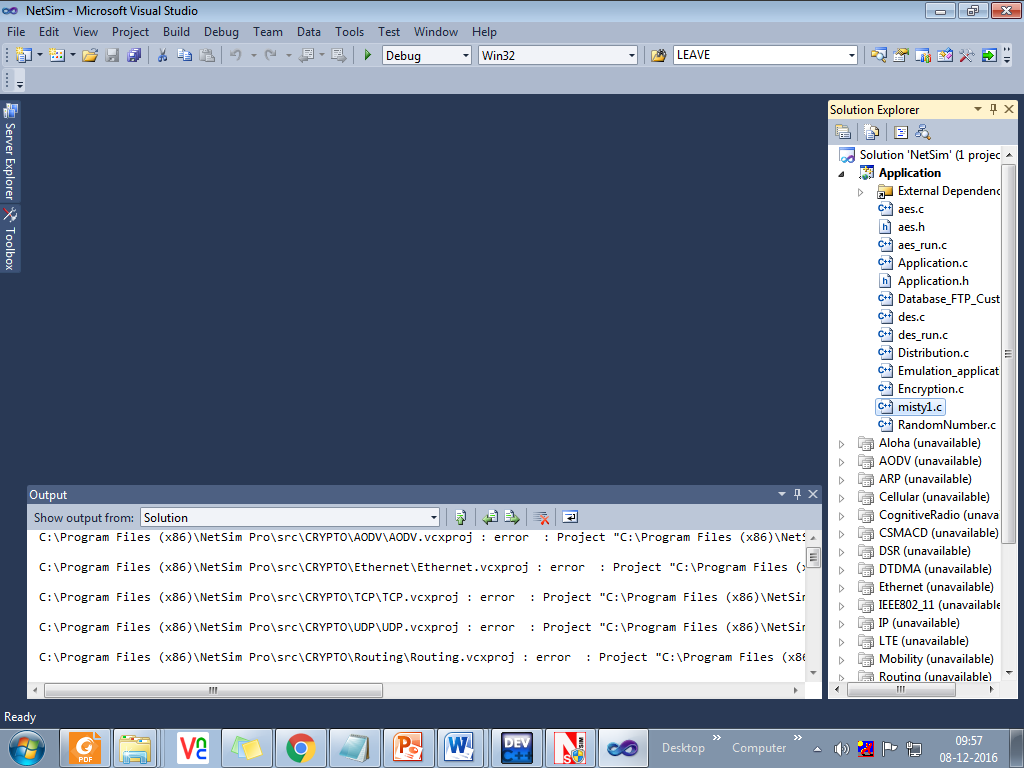
1. To add a new crypto algorithm for encryption, Copy the folders Application, lib, Include and the NetSim.sln file from NetSim’s src folder which is in <NetSim Installation Directory>.
2. Create a new folder with the name “Changing\_Crypto\_algorithm” and paste the copied content inside this folder.
3. Copy the mysty1.c file and paste it inside the Application folder which you currently have in the “Changing\_Crypto\_algorithm” folder.



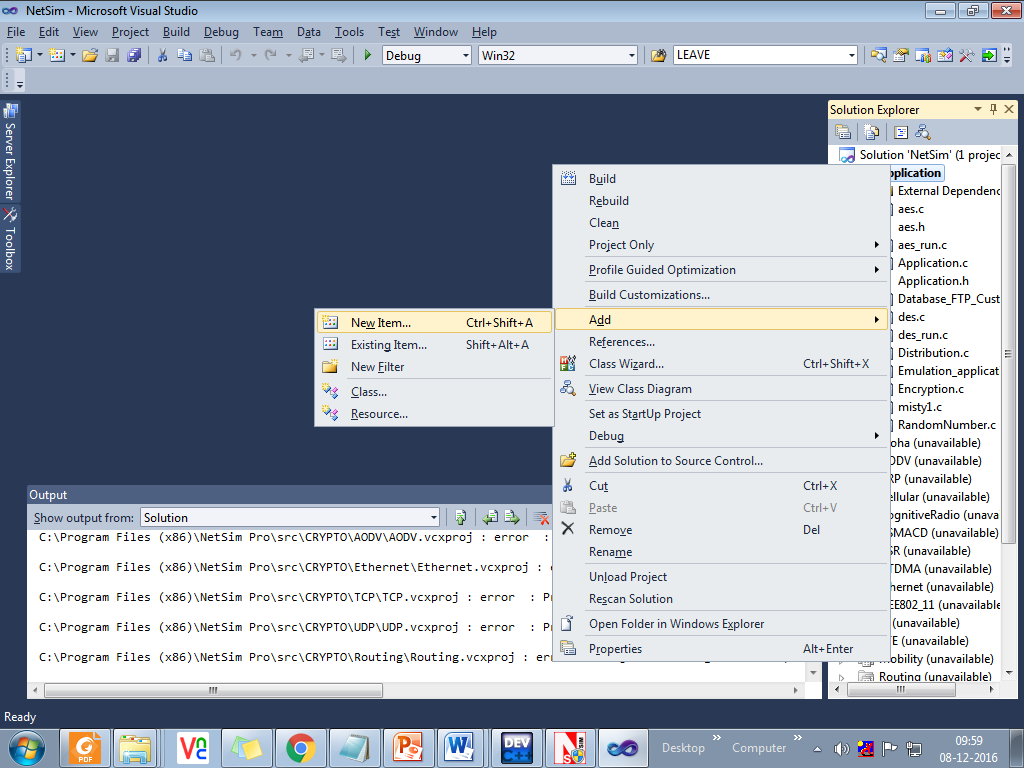
1. Double click on the NetSim.sln file to view the Application codes in visual studio.
2. To add the mysty1.c file to the Application project, right click on Application project, select Add->Existing Item.



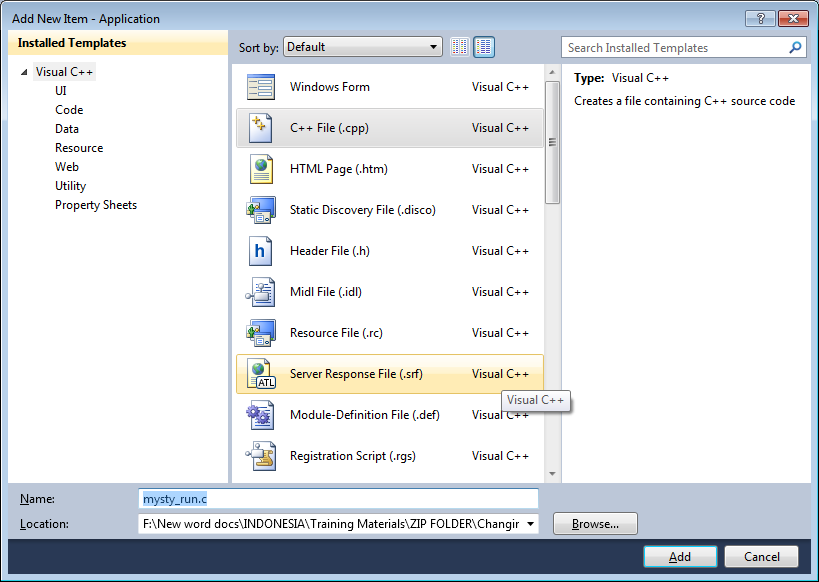
1. Now select mysty1.c file from the list of files and click on the add button.
2. You will find mysty1.c file displayed along with the other files under Application project, in the Solution Explorer.



1. Add a New c files by right clicking on the Application project and selecting Add->New Item.



1. Select C++ File(.cpp) from the list of file types and provide the file name as misty\_run.c



1. Now double click on the newly added mysty\_run.c file in the solution explorer and add the following lines of code to it.

#include <string.h>

#include <stdlib.h>

#include <stdio.h>

#include "application.h"

void misty\_run(char\* str,int\* len)

{

int n;

int l=\*len;

unsigned char buf[32];

unsigned char key[32];

for (n = 0; n < \*len; n+=16,str+=16,l-=16)

{

/\* Set the plain-text \*/

memcpy( buf, str, min(16,l));

misty1\_main(buf);

memcpy(str,buf,16);

}

}

In the mysty\_run() function inside the mysty\_run.c file we pass the plain text in parts of 16 bytes each time to get it encrypted. This is done because the crypto algorithm expects a 16 byte plaintext as input. Here the variable str contains the packet payload and len corresponds to the size of payload in bytes.

1. Now double click on the mysty1.c file in the solution explorer and make the following changes to it.
2. Add #include<application.h> and #define uint8 unsigned char to the beginning of the file(shown in red).

#include <stdlib.h>

#include <string.h>

#include "application.h"

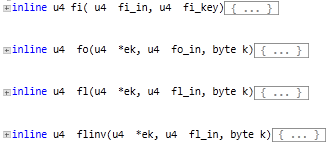
typedef unsigned long u4;

typedef unsigned char byte;

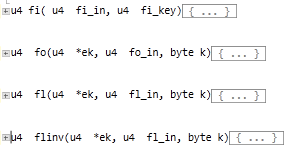
#define MISTY1\_KEYSIZE 32

#define uint8 unsigned char

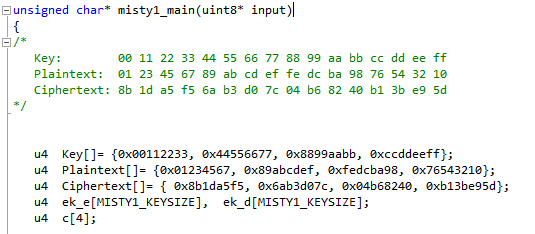
1. Remove **inline** keyword that is present before the functions fi(), fo(), fl() and flinv().



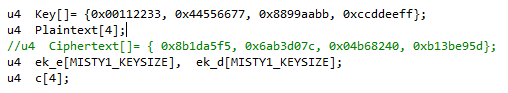
To



1. Now go to the main() function in the file and remove or comment the line **#ifdef TESTMAIN** before the main() function and also the associated **#endif** at the end of the main() function.
2. Now rename the function **main()** to **unsigned char\* misty1\_main(uint8\* input)**



1. Comment the declaration of Ciphertext, Modify the declaration of Plaintext variable, as shown below:



1. Now comment the lines starting from misty1\_keyinit() to misty1\_key\_destroy() as shown below:



1. Add the following lines of code just above the misty1\_key\_destroy(ek\_e); statement as shown below:

// Memcpy is used to equate input which is Char to Plaintext

// which is Unsigned Long

memcpy(Plaintext,input,2\*sizeof(u4));

memcpy(&Plaintext[2],&input[8],2\*sizeof(u4));

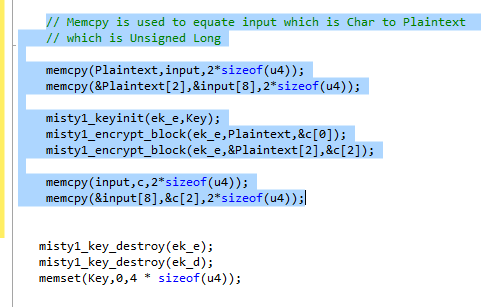
misty1\_keyinit(ek\_e,Key);

misty1\_encrypt\_block(ek\_e,Plaintext,&c[0]);

misty1\_encrypt\_block(ek\_e,&Plaintext[2],&c[2]);

memcpy(input,c,2\*sizeof(u4));

memcpy(&input[8],&c[2],2\*sizeof(u4));



1. Inside the mysty1\_main function the above codes were modified to ensure that the plaintext is properly initialized with the 16 bytes of payload received, for the encryption to happen.
2. Here, memcpy() is done initially to equate input received as which is char, to the plain text which is unsigned long.

memcpy(Plaintext,input,2\*sizeof(u4));

memcpy(&Plaintext[2],&input[8],2\*sizeof(u4));

1. After the calls to misty1\_encrypt\_block() memcpy() is done to equate the encrypted cipher text back to the input.

memcpy(input,c,2\*sizeof(u4));

memcpy(&input[8],&c[2],2\*sizeof(u4));

1. Now double click on the application.c file and make a call to mysty\_run() function instead of the call to aes256, inside the copy\_payload() function as shown below (changes are marked in red):

if(info->encryption==Encryption\_TEA)

encryptBlock(real,payload,&key);

else if(info->encryption==Encryption\_AES)

{

misty\_run(real,payload);

//aes256(real,payload);

}

else if(info->encryption==Encryption\_DES)

des(real,payload);

1. Rebuild Application project and replace the libApplication.dll file in NetSim bin folder.
2. Open Configuration.xml file from the zip and make sure that AES encryption is selected in the application properties.
3. Also Wireshark option has to be set to either Online or Offline in any of the nodes where AES256 encryption is enabled.
4. Now mysty1 codes will be running instead of AES256.
5. You can see the encrypted payload in Wireshark either during simulation if online is set or after the simulation if offline is set.
6. Setting Wireshark to either online or offline will give you Packet Capture metrics where links to .pcap files are provided. The number of links available depends on the number of nodes in which Wireshark is enabled.

