

*****Draft*****

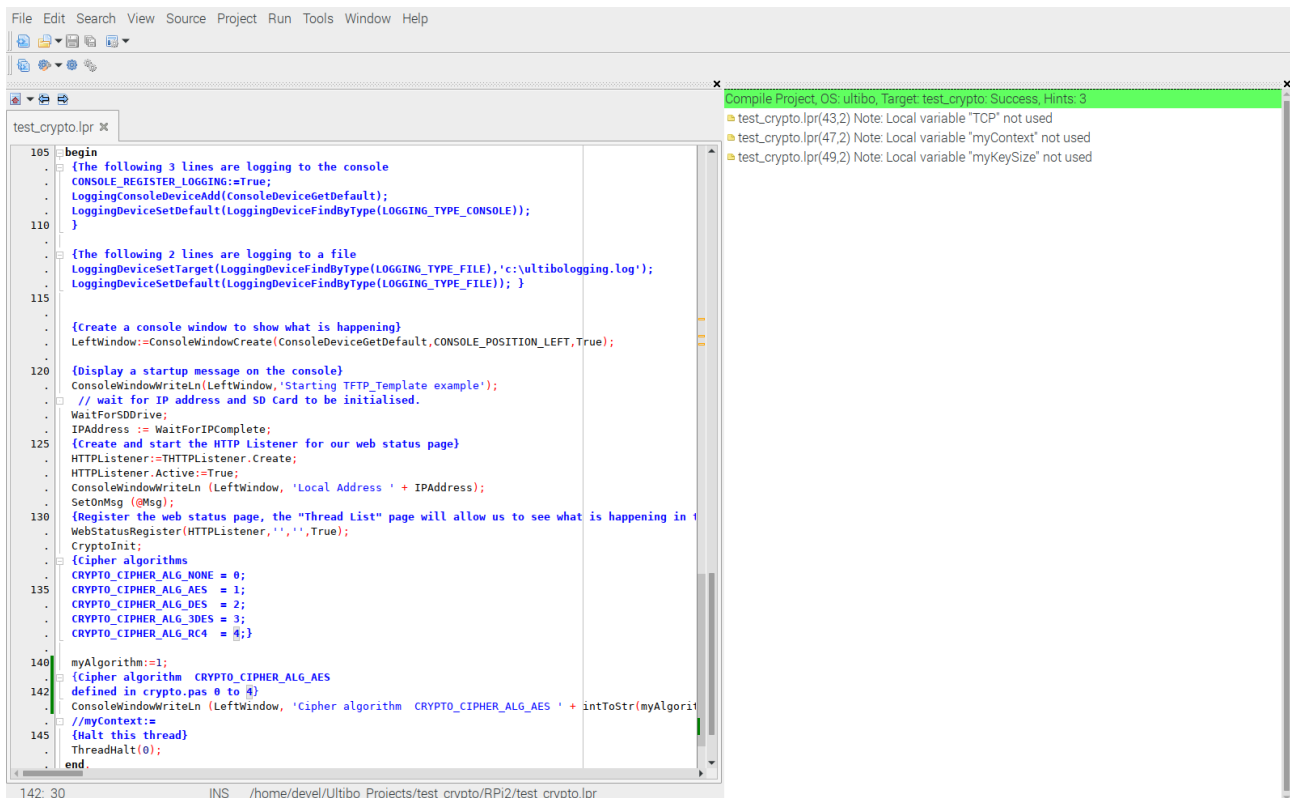
crypto notes 05/02/20

Starting with TFTP_Template

*****Draft*****

Started with the file from “TFTP_Template.lpr” to create “test_crypto.lpr” & “test_crypto.lpi”
In addition this needs uTFTP.pas, upker7.sh, and cmdstftp.

Compile the project with “Run/Compile” or “Run/Clean up and Build”.



```
File Edit Search View Source Project Run Tools Window Help
test_crypto.lpr
105 begin
106 {The following 3 lines are logging to the console
107 .
108 .
109 .
110 .
111 }
112 {The following 2 lines are logging to a file
113 .
114 .
115 }
116 {Create a console window to show what is happening}
117 LeftWindow:=ConsoleWindowCreate(ConsoleDeviceGetDefault,CONSOLE_POSITION_LEFT,True);
118
119 {Display a startup message on the console}
120 ConsoleWindowWriteLn(LeftWindow,'Starting TFTP_Template example');
121 // wait for IP address and SD Card to be initialised.
122 WaitForSDDrive;
123 IPAddress := WaitForIPComplete;
124 {Create and start the HTTP Listener for our web status page}
125 HTTPListener:=THTTPListener.Create;
126 HTTPListener.Active:=True;
127 ConsoleWindowWriteLn (LeftWindow, 'Local Address ' + IPAddress);
128 SetOnMsg (Msg;
129 {Register the web status page, the "Thread List" page will allow us to see what is happening in t
130 WebStatusRegister(HTTPListener,'','',True);
131 CryptoInit;
132 {Cipher algorithms
133 CRYPTO_CIPHER_ALG_NONE = 0;
134 CRYPTO_CIPHER_ALG_AES = 1;
135 CRYPTO_CIPHER_ALG_DES = 2;
136 CRYPTO_CIPHER_ALG_3DES = 3;
137 CRYPTO_CIPHER_ALG_RC4 = 4;}
138
139 myAlgorithm:=1;
140 {Cipher algorithm CRYPTO_CIPHER_ALG_AES
141 defined in crypto.pas 0 to 4}
142 ConsoleWindowWriteLn (LeftWindow, 'Cipher algorithm CRYPTO_CIPHER_ALG_AES ' + IntToStr(myAlgorit
143 //myContext:=
144 {Halt this thread}
145 ThreadHalt(0);
146 end.
```

Compile Project OS: ultibo, Target: test_crypto, Success, Hints: 3

- test_crypto.lpr(43,2) Note: Local variable "TCP" not used
- test_crypto.lpr(47,2) Note: Local variable "myContext" not used
- test_crypto.lpr(49,2) Note: Local variable "myKeySize" not used

142: 30 INS /home/devel/Ultibo_Projects/test_crypto/RPi2/test_crypto.lpr

Once the Green bar is displayed it can be transfer to the Ultibo System.

After adding APICrypto.pas

In test_crypto.lpt in

var

AESECBKey:PByte;

AESECBData:PByte;

AESECBAESKey:TAESKey;

AESCBCKey:PByte;

AESCBData:PByte;

AESCBCVector:PByte;

//Context:PBigIntContext;

Cipher:PCipherContext;

key:String;

Data:String;

Actual:String;

PData:PString;

Datalen:LongWord;

With the addition of code below the hard coded matches APICrypto.pas

AESEncryptBlock (128bit)

Electronic Codebook (ECB)

AESEncryptBlock (192bit)

Electronic Codebook (ECB)

AESEncryptBlock (256bit)

Electronic Codebook (ECB)

AESDecryptBlock (128bit)

Electronic Codebook (ECB)

AESDecryptBlock (192bit)

Electronic Codebook (ECB)

AESDecryptBlock (256bit)

Electronic Codebook (ECB)

ConsoleWindowWriteLn (LeftWindow, '');

ConsoleWindowWriteLn (LeftWindow, 'AESEncryptBlock (128bit)');

```
ConsoleWindowWriteLn (LeftWindow, 'Electronic Codebook (ECB)');
AESECBKey:=AllocMem(AES_KEY_SIZE128);
```

```
StringToBytes('2b7e151628aed2a6abf7158809cf4f3c',PByte(AESECBKey),AES_
KEY_SIZE128);
AESECBData:=AllocMem(AES_BLOCK_SIZE);
```

```
StringToBytes('6bc1bee22e409f96e93d7e117393172a',PByte(AESECBData),AES
_BLOCK_SIZE);
AESKeySetup(AESECBKey,AES_KEY_SIZE128,@AESECBAESKey);
AESEncryptBlock(AESECBData,AESECBData,@AESECBAESKey);
Actual:=BytesToString(PByte(AESECBData),AES_BLOCK_SIZE);
ConsoleWindowWriteLn (LeftWindow, 'Key:  '
+'2b7e151628aed2a6abf7158809cf4f3c');
ConsoleWindowWriteLn (LeftWindow, 'Data:  '
+'6bc1bee22e409f96e93d7e117393172a');
ConsoleWindowWriteLn (LeftWindow, 'Actual: ' + Actual);
FreeMem(AESECBKey);
FreeMem(AESECBData);
```

```
ConsoleWindowWriteLn (LeftWindow, '');
ConsoleWindowWriteLn (LeftWindow, 'AESEncryptBlock (192bit)');
ConsoleWindowWriteLn (LeftWindow, 'Electronic Codebook (ECB)');
AESECBKey:=AllocMem(AES_KEY_SIZE192);
```

```
StringToBytes('8e73b0f7da0e6452c810f32b809079e562f8ead2522c6b7b',PByte(A
ESECBKey),AES_KEY_SIZE192);
AESECBData:=AllocMem(AES_BLOCK_SIZE);
```

```
StringToBytes('6bc1bee22e409f96e93d7e117393172a',PByte(AESECBData),AES
_BLOCK_SIZE);
AESKeySetup(AESECBKey,AES_KEY_SIZE192,@AESECBAESKey);
AESEncryptBlock(AESECBData,AESECBData,@AESECBAESKey);
Actual:=BytesToString(PByte(AESECBData),AES_BLOCK_SIZE);
ConsoleWindowWriteLn (LeftWindow, 'Key:  '
+'8e73b0f7da0e6452c810f32b809079e562f8ead2522c6b7b');
ConsoleWindowWriteLn (LeftWindow, 'Data:  '
+'6bc1bee22e409f96e93d7e117393172a');
ConsoleWindowWriteLn (LeftWindow, 'Actual: ' + Actual);
FreeMem(AESECBKey);
FreeMem(AESECBData);
```

```
ConsoleWindowWriteLn (LeftWindow, '');
ConsoleWindowWriteLn (LeftWindow, 'AESEncryptBlock (256bit)');
ConsoleWindowWriteLn (LeftWindow, 'Electronic Codebook (ECB)');
```

AESECBKey:=AllocMem(AES_KEY_SIZE256);

StringToBytes('603deb1015ca71be2b73aef0857d77811f352c073b6108d72d9810a30914dff4',PByte(AESECBKey),AES_KEY_SIZE256);
AESECBData:=AllocMem(AES_BLOCK_SIZE);

StringToBytes('6bc1bee22e409f96e93d7e117393172a',PByte(AESECBData),AES_BLOCK_SIZE);
AESKeySetup(AESECBKey,AES_KEY_SIZE256,@AESECBAESKey);
AESDecryptBlock(AESECBData,AESECBData,@AESECBAESKey);
Actual:=BytesToString(PByte(AESECBData),AES_BLOCK_SIZE);
ConsoleWindowWriteLn (LeftWindow, 'Key: ' +
'603deb1015ca71be2b73aef0857d77811f352c073b6108d72d9810a30914dff4');
ConsoleWindowWriteLn (LeftWindow, 'Data: ' +
'6bc1bee22e409f96e93d7e117393172a');
ConsoleWindowWriteLn (LeftWindow, 'Actual: ' + Actual);
FreeMem(AESECBKey);
FreeMem(AESECBData);
ConsoleWindowWriteLn (LeftWindow, '');
ConsoleWindowWriteLn (LeftWindow, 'AESDecryptBlock (128bit)');
ConsoleWindowWriteLn (LeftWindow, 'Electronic Codebook (ECB)');
AESECBKey:=AllocMem(AES_KEY_SIZE128);

StringToBytes('2b7e151628aed2a6abf7158809cf4f3c',PByte(AESECBKey),AES_KEY_SIZE128);
AESECBData:=AllocMem(AES_BLOCK_SIZE);

StringToBytes('3ad77bb40d7a3660a89ecaf32466ef97',PByte(AESECBData),AES_BLOCK_SIZE);
AESKeySetup(AESECBKey,AES_KEY_SIZE128,@AESECBAESKey);
AESDecryptBlock(AESECBData,AESECBData,@AESECBAESKey);
Actual:=BytesToString(PByte(AESECBData),AES_BLOCK_SIZE);
ConsoleWindowWriteLn (LeftWindow, 'Key: ' +
'2b7e151628aed2a6abf7158809cf4f3c');
ConsoleWindowWriteLn (LeftWindow, 'Data: ' +
'3ad77bb40d7a3660a89ecaf32466ef97');
ConsoleWindowWriteLn (LeftWindow, 'Actual: ' + Actual);
FreeMem(AESECBKey);
FreeMem(AESECBData);

ConsoleWindowWriteLn (LeftWindow, '');
ConsoleWindowWriteLn (LeftWindow, 'AESDecryptBlock (192bit)');
ConsoleWindowWriteLn (LeftWindow, 'Electronic Codebook (ECB)');
AESECBKey:=AllocMem(AES_KEY_SIZE192);

```
StringToBytes('8e73b0f7da0e6452c810f32b809079e562f8ead2522c6b7b',PByte(AESECBKey),AES_KEY_SIZE192);
AESECBData:=AllocMem(AES_BLOCK_SIZE);
```

```
StringToBytes('bd334f1d6e45f25ff712a214571fa5cc',PByte(AESECBData),AES_BLOCK_SIZE);
AESKeySetup(AESECBKey,AES_KEY_SIZE192,@AESECBAESKey);
AESDecryptBlock(AESECBData,AESECBData,@AESECBAESKey);
Actual:=BytesToString(PByte(AESECBData),AES_BLOCK_SIZE);
ConsoleWindowWriteLn (LeftWindow, 'Key: '
+'8e73b0f7da0e6452c810f32b809079e562f8ead2522c6b7b');
ConsoleWindowWriteLn (LeftWindow, 'Data: '
+'bd334f1d6e45f25ff712a214571fa5cc');
ConsoleWindowWriteLn (LeftWindow, 'Actual: ' + Actual);
FreeMem(AESECBKey);
FreeMem(AESECBData);
```

```
ConsoleWindowWriteLn (LeftWindow, '');
ConsoleWindowWriteLn (LeftWindow, 'AESDecryptBlock (256bit)');
ConsoleWindowWriteLn (LeftWindow, 'Electronic Codebook (ECB)');
AESECBKey:=AllocMem(AES_KEY_SIZE256);
```

```
StringToBytes('603deb1015ca71be2b73aef0857d77811f352c073b6108d72d9810a30914dff4',PByte(AESECBKey),AES_KEY_SIZE256);
AESECBData:=AllocMem(AES_BLOCK_SIZE);
```

```
StringToBytes('f3eed1bdb5d2a03c064b5a7e3db181f8',PByte(AESECBData),AES_BLOCK_SIZE);
AESKeySetup(AESECBKey,AES_KEY_SIZE256,@AESECBAESKey);
AESDecryptBlock(AESECBData,AESECBData,@AESECBAESKey);
Actual:=BytesToString(PByte(AESECBData),AES_BLOCK_SIZE);
ConsoleWindowWriteLn (LeftWindow, 'Key: '
+'603deb1015ca71be2b73aef0857d77811f352c073b6108d72d9810a30914dff4');
ConsoleWindowWriteLn (LeftWindow, 'Data: '
+'f3eed1bdb5d2a03c064b5a7e3db181f8');
ConsoleWindowWriteLn (LeftWindow, 'Actual: ' + Actual);
FreeMem(AESECBKey);
FreeMem(AESECBData);
```

./upker.sh

Starting TFTP_test_crypto example
Local Address 192.168.1.245
TFTP Ready.

AESEncryptBlock (128bit)
Electronic Codebook (ECB)
Key: 2b7e151628aed2a6abf7158809cf4f3c
Data: 6bc1bee22e409f96e93d7e117393172a
Actual: 3ad77bb40d7a3660a89ecaf32466ef97

AESEncryptBlock (192bit)
Electronic Codebook (ECB)
Key: 8e73b0f7da0e6452c810f32b809079e562f8ead2522c6b7b
Data: 6bc1bee22e409f96e93d7e117393172a
Actual: bd334f1d6e45f25ff712a214571fa5cc

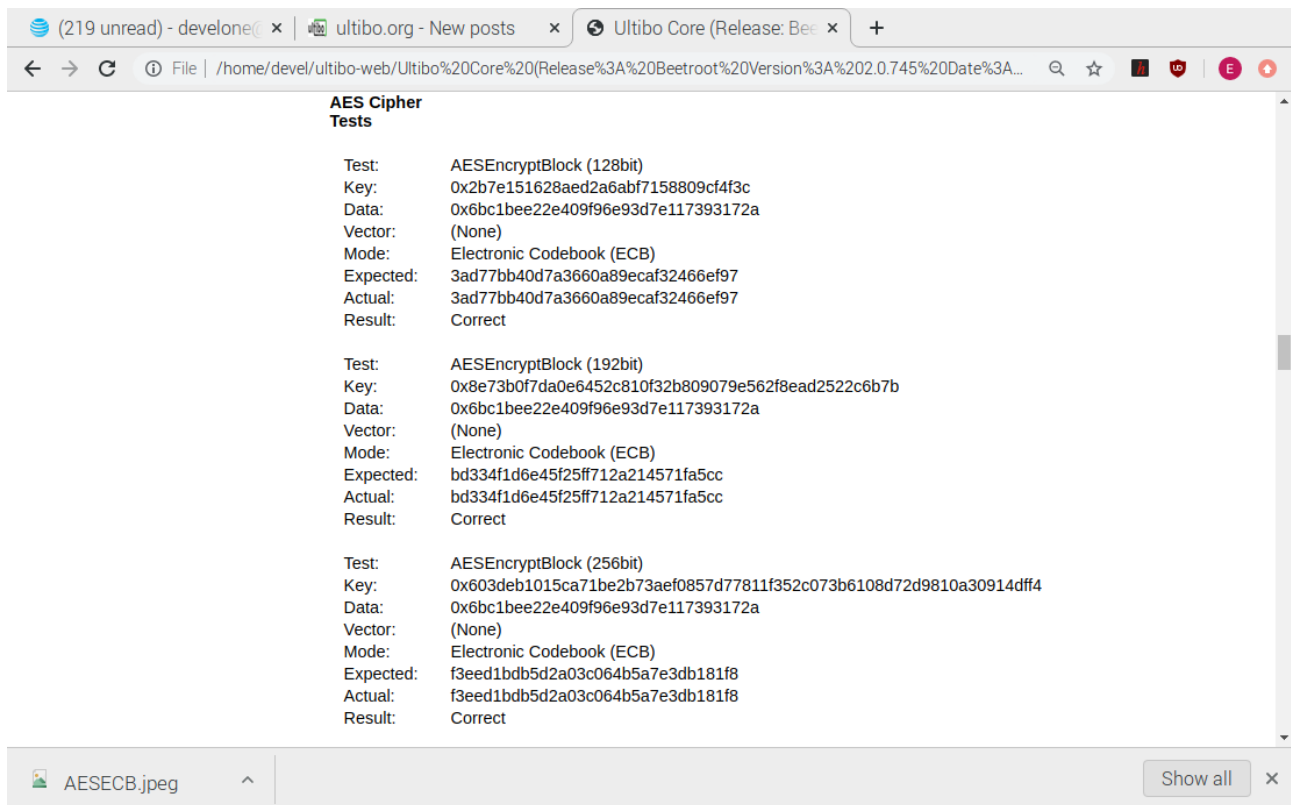
AESEncryptBlock (256bit)
Electronic Codebook (ECB)
Key: 603deb1015ca71be2b73aef0857d77811f352c073b6108d72d9810a30914dff4
Data: 6bc1bee22e409f96e93d7e117393172a
Actual: f3eed1bdb5d2a03c064b5a7e3db181f8

AESDecryptBlock (128bit)
Electronic Codebook (ECB)
Key: 2b7e151628aed2a6abf7158809cf4f3c
Data: 3ad77bb40d7a3660a89ecaf32466ef97
Actual: 6bc1bee22e409f96e93d7e117393172a

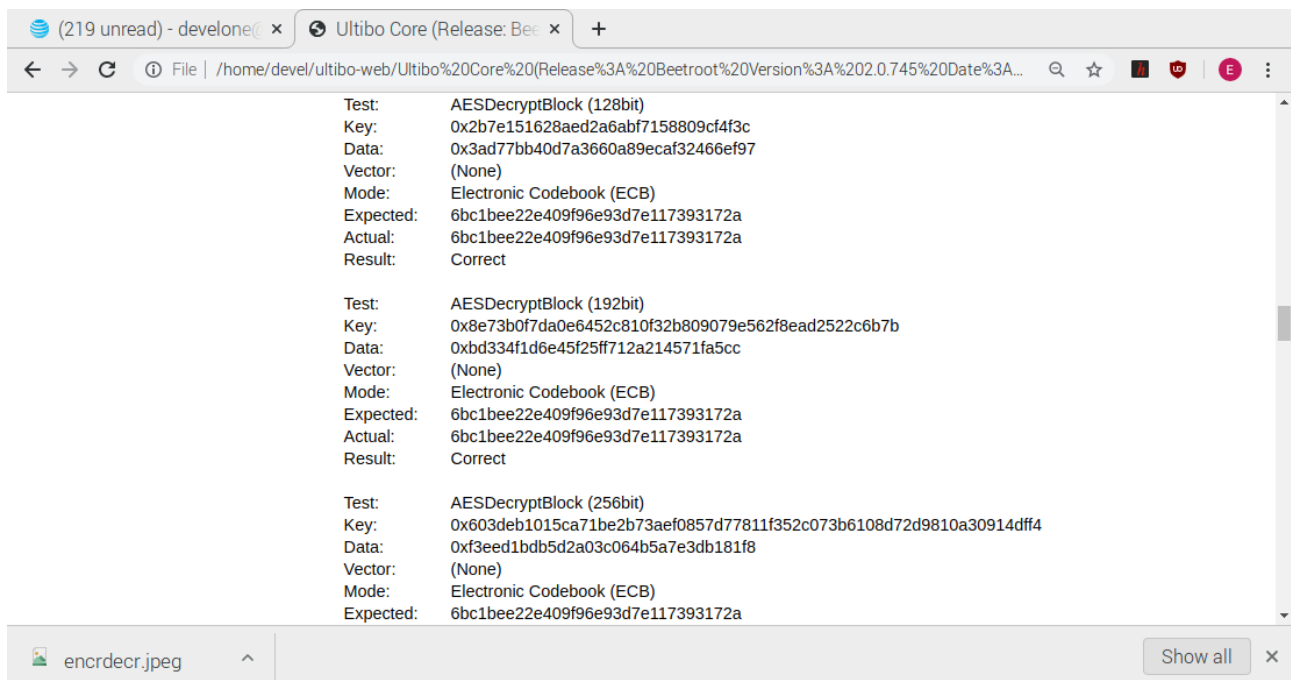
AESDecryptBlock (192bit)
Electronic Codebook (ECB)
Key: 8e73b0f7da0e6452c810f32b809079e562f8ead2522c6b7b
Data: bd334f1d6e45f25ff712a214571fa5cc
Actual: 6bc1bee22e409f96e93d7e117393172a

AESDecryptBlock (256bit)
Electronic Codebook (ECB)
Key: 603deb1015ca71be2b73aef0857d77811f352c073b6108d72d9810a30914dff4
Data: f3eed1bdb5d2a03c064b5a7e3db181f8
Actual: 6bc1bee22e409f96e93d7e117393172a
Transfer for "kernel7.img" started.
Transfer for kernel7.img complete.

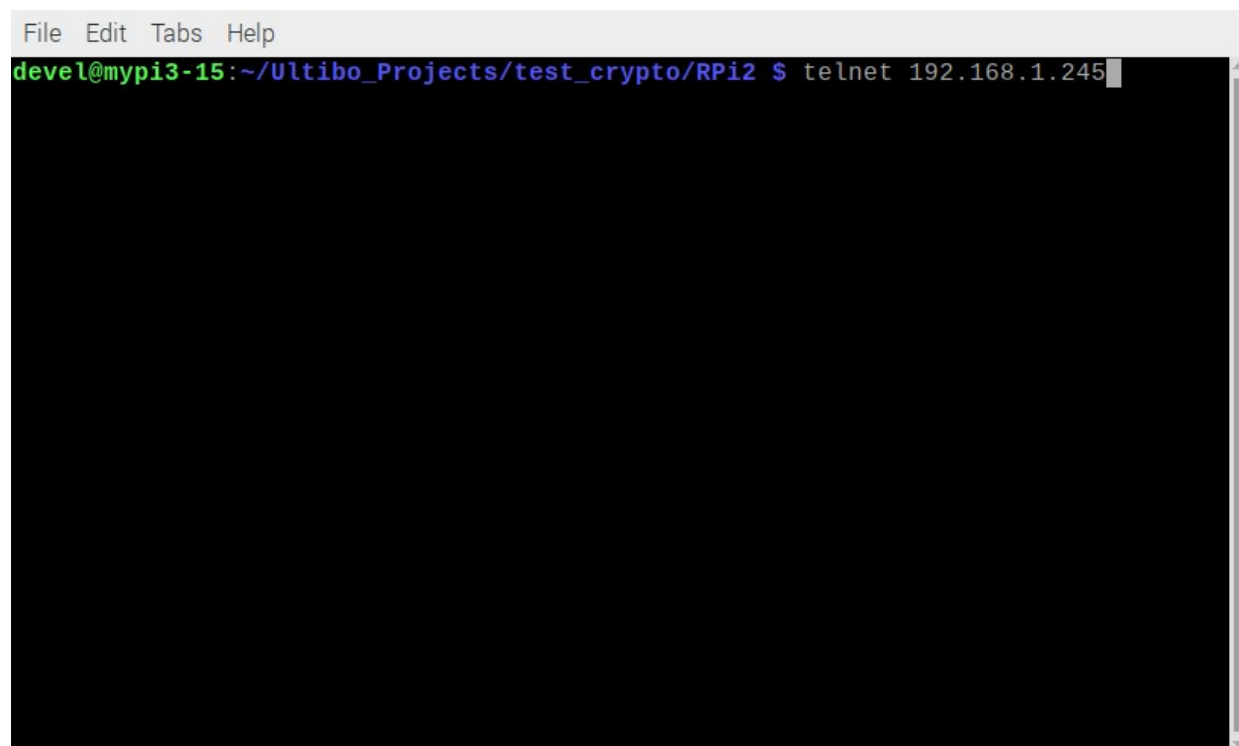
Now the results match the results on <http://192.168.1.245/status/cryptoapi/>



Dencryption APICrypto.pas



shell1

A terminal window with a light gray title bar containing the menu items 'File', 'Edit', 'Tabs', and 'Help'. The terminal's background is black, and the text is in a monospaced font. The prompt 'devel@mypi3-15:~/Ultibo_Projects/test_crypto/RPi2 \$' is shown in green and blue. The command 'telnet 192.168.1.245' is entered in white, followed by a white cursor. A vertical scrollbar is visible on the right side of the terminal area.

```
File Edit Tabs Help
devel@mypi3-15:~/Ultibo_Projects/test_crypto/RPi2 $ telnet 192.168.1.245
```

shell2

