**Lab - Encrypting and Decrypting Data Using OpenSSL**

**Ashkeyev Ali 18BD11079**

Objectives

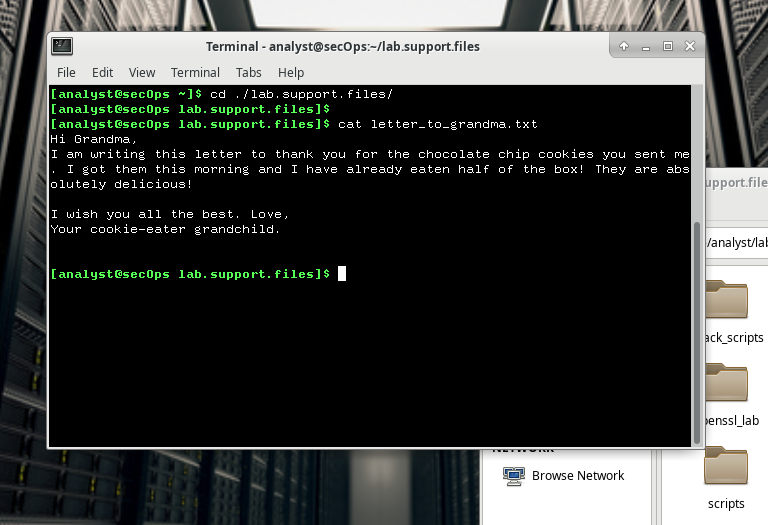
Part 1: Encrypting Messages with OpenSSL

Part 2: Decrypting Messages with OpenSSL

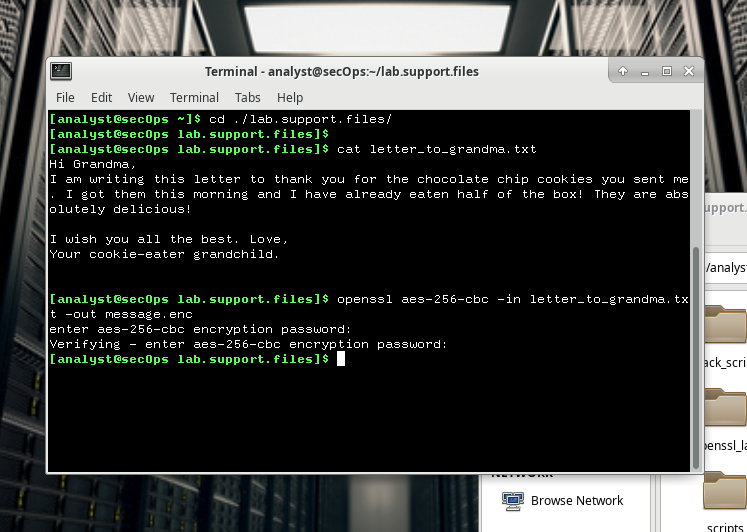
Part 1: Encrypting Messages with OpenSSL

OpenSSL can be used as a standalone tool for encryption. While many encryption algorithms can be used, this lab focuses on AES. To use AES to encrypt a text file directly from the command line using OpenSSL, follow the steps below:

**Step 1: Encrypting a Text File**

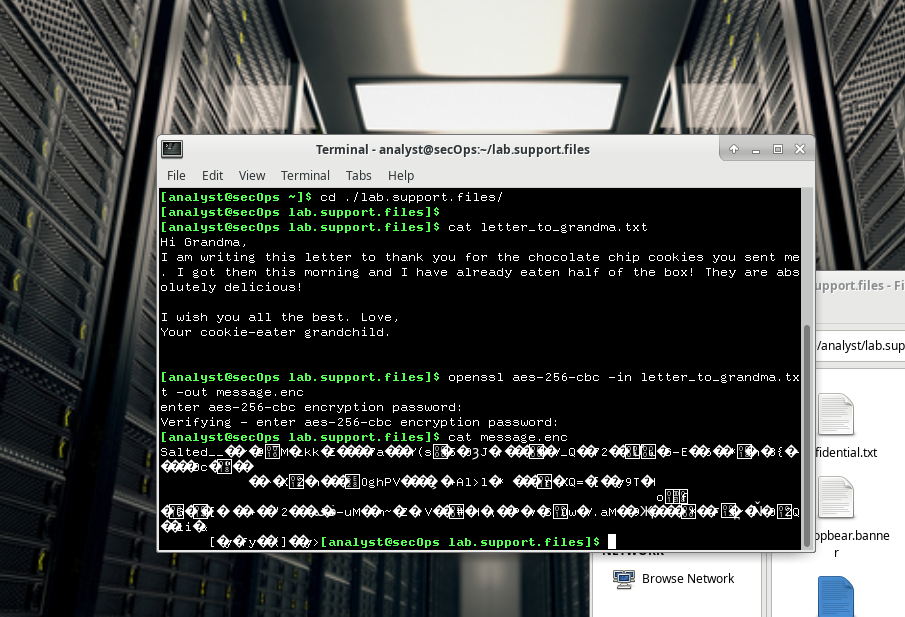


1. list the contents of the encrypted letter\_to\_grandma.txt text file on the screen
2. From the same terminal window, issue the command below to encrypt the text file. The command will use AES-256 to encrypt the text file and save the encrypted version as message.enc. OpenSSL will ask for a password and for password confirmation. Provide the password as requested and be sure to remember the password.



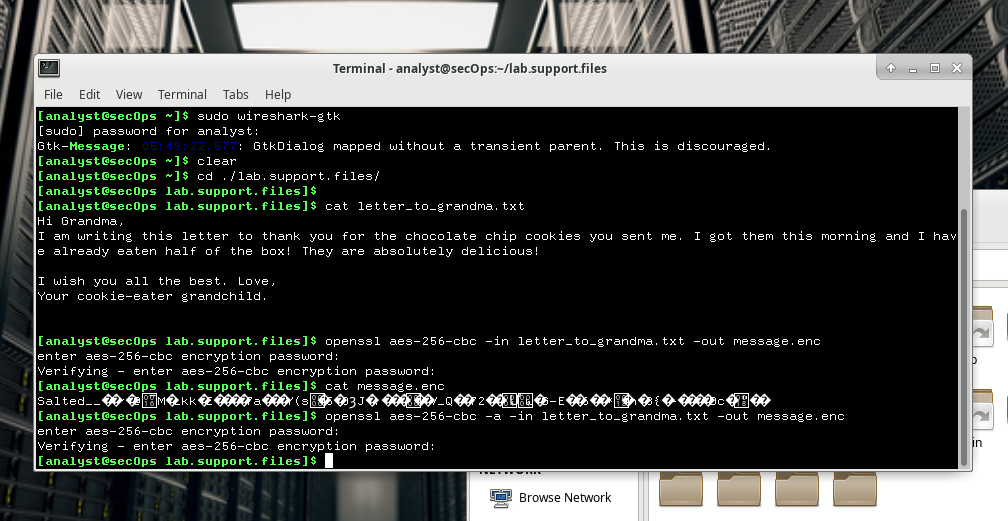
I set password cybercops

1. Did the contents of the message.enc file display correctly? What does it look like? Explain.



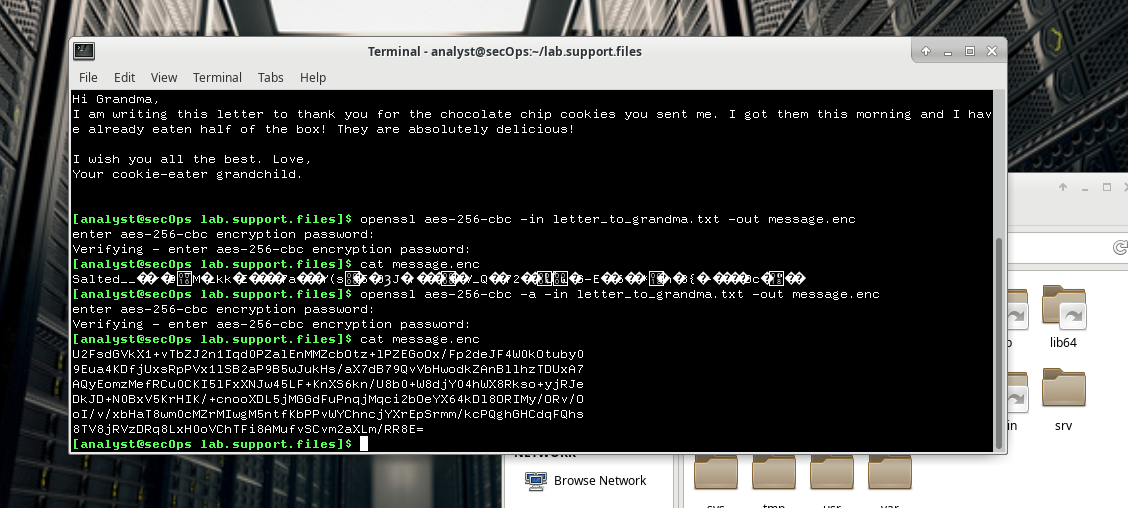
No, it is displayed symbols which is not readable for human. It looks like set of symbols. It happened because of file was encrypted and OpenSSL generated binary file.

1. To make the file readable, run the OpenSSL command again, but this time add the -a option. The -a option tells OpenSSL to encode the encrypted message using a different encoding method of Base64 before storing the results in a file. Note: Base64 is a group of similar binary-to-text encoding schemes used to represent binary data in an ASCII string format.



We encoded file by using Base64 to transfer binary data in an ASCII string format.

1. Once again, use the cat command to display the contents of the, now re-generated, message.enc file: Note: The contents of message.enc will vary.



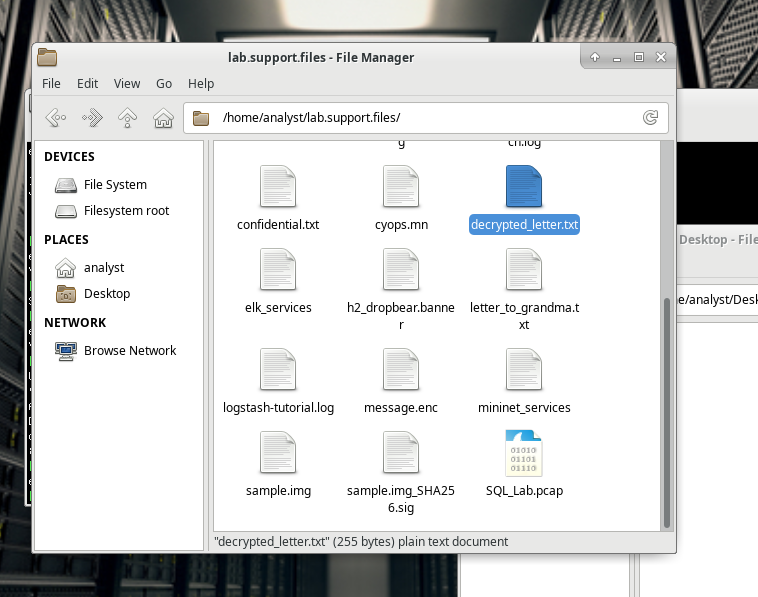
**Is message.enc displayed correctly now? Explain**

Yes. Now it has text format, not symbols as in previous step

**Can you think of a benefit of having message.enc Base64-encoded?**

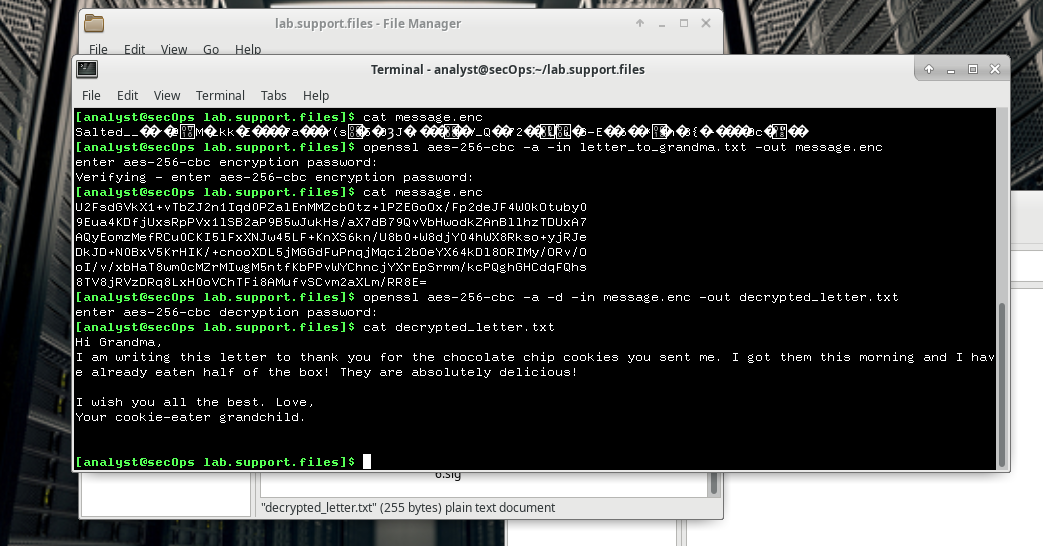
The primary use case of base64 encoding is when you want to store or transfer data with a restricted set of characters; i.e. when you can't pass an arbitrary value in each byte. Some forms of communication uses special meanings for certain characters. So sending binary means the communication method could interpret it wrongly. Email is an extreme case, it’s because messages are supposed to work no matter what’s on the sender’s side or the receiver.

**Part 2: Decrypting Messages with OpenSSL**



Creating decryoted letter text file.

**Was the letter decrypted correctly?**



Yes, it is decrypted correctly, and we can see normal letter.

**The command used to decrypt also contains -a option. Can you explain?**

In previous step we encoded it to Base64, so it should be decoded to base64 before it will be decrypted by the OpenSSL.