Daniel Oliveros

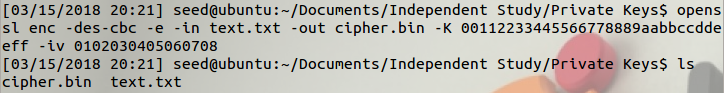
Garrett Bogart

Spring 2018 – Independent Study

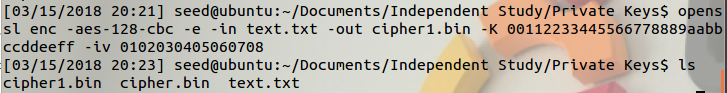
Secret-Key Encryption

**Task 1**

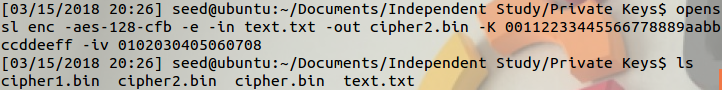
Des-cbc encryption

****

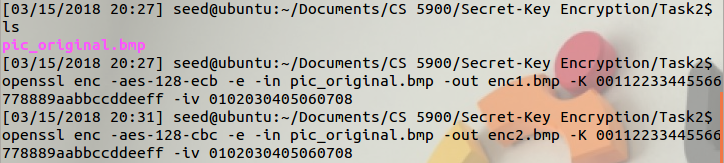
aes-128-cbc

****

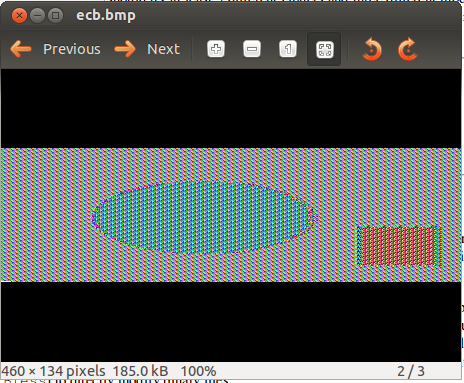
-aes-128-cfb

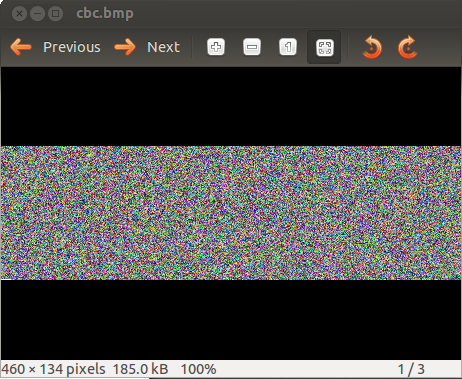
****

**Task 2**









**Task 3:**

****

****

**Dec1 - ecb**

****

**Dec2 - cbc**

****

**Dec3 - cfb**

****

**Dec4 - ofb**

****

**Task 4:**

**4.2:**



**Task 5**

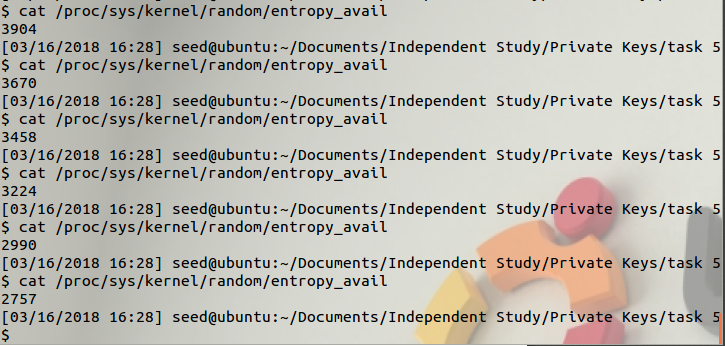
The program we wrote for this task goes through each word in the file “words.txt” and uses it as the encryption key for the aes-128-cbc algorithm. The resulting encrypted message is then compared to the one we were given in the problem statement, and if there is a match we print out the word used as a key that caused it.

This is what running this program looks like:

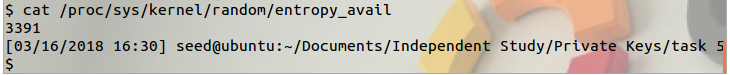


**Task 6**

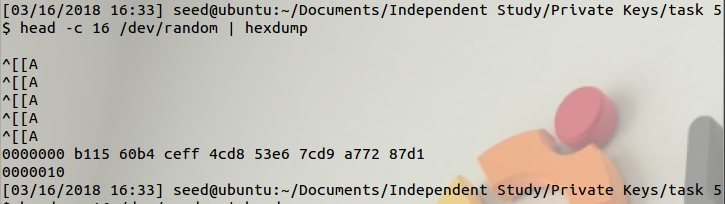
**6A:** After running the command in quick succession we can see the entropy decrease.

****

Then allowing some time to pass to build up some new randomness we can see the amount of available entropy increase

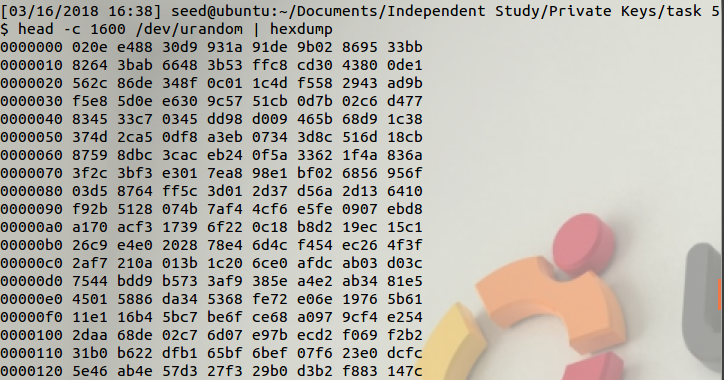


**6B:** After repeated use of the command a pause occured



It took a few seconds before the command completed. If you want to decrease the time of waiting a few seconds you can click around on your computer or type to a random file.

**6C:**  We ran the command multiple times and never ran out of pseudo random number



**Recommendations**

1. We included a outline.c file. We think that it provides a basic outline of how where the program needs to go as well as some tips on getting input and formatting strings.

**Ideas for Improvement**

1. A run through of Task 5 would be a great idea. In practice, there were many points where we got stuck but we could always move forward knowing what we needed to do next. It’s good to make sure students have an idea of what their process should look like