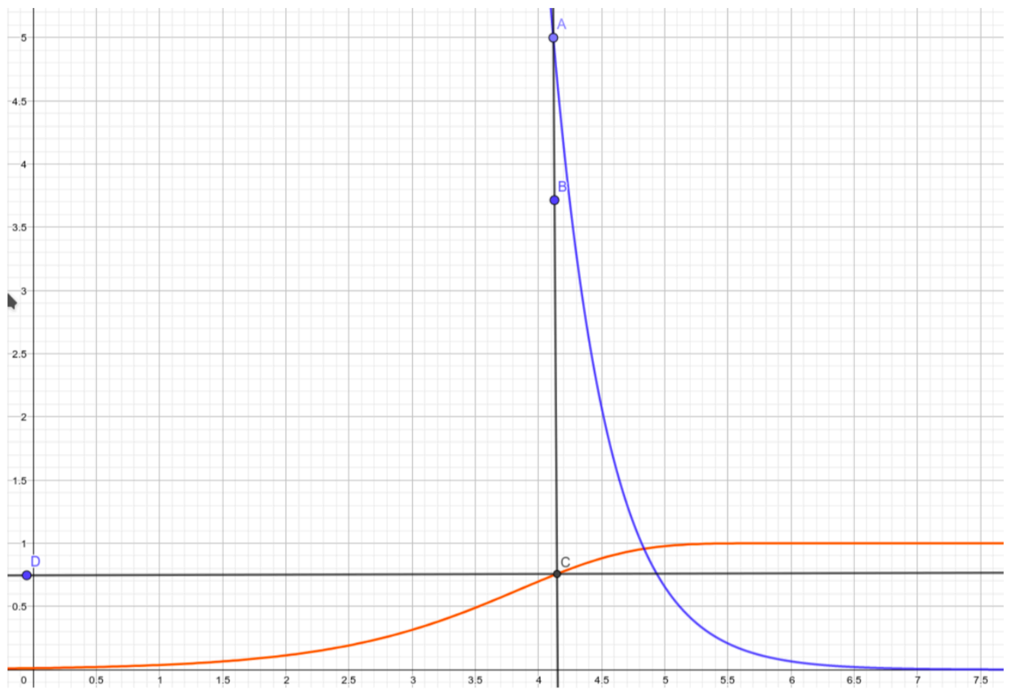
Faris Zahrah
Practical Cryptology
Hellman and Rainbow Tables
November 14, 2018

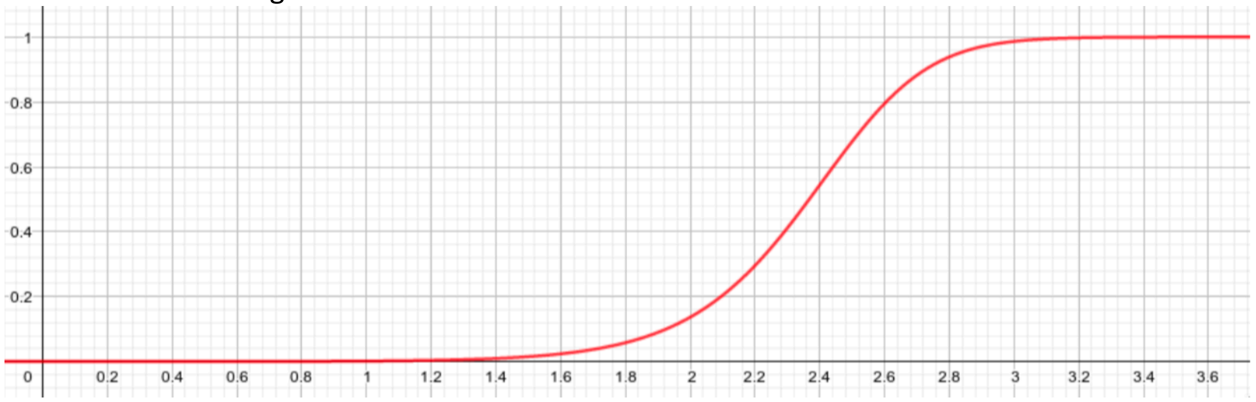
Hellman and rainbow tables are means of precomputing as much information for an attack as possible. The trade-off then comes between time and memory in both the precomputation period and after the cipher text has been observed. Precomputing allows an attacker to decrypt intercepted messages in much less time once they are observed.

Problem 1 of the Hellman table exercise dealt with the idea of how many iterations (columns per table) and how many different random inputs (rows per table) should be used. We were given the number of table set to 256. When the function was plotted with m * t^2 = l, I obtained using roughly 14k rows and between 4 and 5 rows. So, I used these as a starting off point for parameters going forward with the code.

The Hellman table code is relatively simple, the most finicky part was that I implemented it in python. It performs an AES encryption for every column, then adds the reduction function which in this case is adding in the table number, and at the end of each row it adds the result to a list of endpoints. The list of end points is only based on the last 8 bits of each result; if the result is already in the list then it is not counted twice in the total coverage.



The rainbow table reduces the number of tables to one. The trade-off is that this table is much larger. Based on the formula a d the graph below, we chose to use 312 columns, and 312*2^8 rows. The scale is logarithmic



We chose to use just over 80k rows with 316 columns. This gives the probability of coverage close to 75%. I received a coverage of about 16,000,000 although our expected results were much closer to 12,000,000.