CRYPTOGRAPHY: SIMPLIFIED DES

Based on <u>Intro to Cryptography</u> (Trappe and Washington).

1. Facilitator Notes

1. Split students into 10 groups and send them to the 10 "Machines" for encryption. Give them a blank ENCRYPTION handouts sheet, so that they can fill out details as they go. Explain that every Machine is part of the whole system, whose goal is to encrypt a message (e.g. it takes `INPUT` and produces `OUTPUT` once you run through every machine).
2. Give them about 5 minutes at each Machine to work through a series of examples. I printed various inputs on different colors of paper for each round, so that no one gets confused on which input corresponds to which output (print 10 sets of inputs to do this 10 times). [Note: I have simplified the key K in the machines from the K_i from the text.]
3. After 5 minutes, have groups rotate through each Machine, so that they can understand how each one works.
4. Ask students to describe and discuss decryption in their groups and as a class. [Start with the 12-bit string LR and switch to get RL. Use the key K_i in reverse (if following the textbook), and decryption is basically the same as encryption.]
5. Discuss (next day) how the DES generalizes this.

Note. Additional things to do ahead of time:

- Make sure each Machine has a large and clear sign, so that groups can find each other easily.
- Make sure you get and bring in two pairs of scissors for SAW and SCISSORS.

2. Machines

The following pages are intended to be left at each Machine station as instructions. Sample `INPUT: 011100100110` should produce `OUTPUT: 100110011000` (this is the example in the textbook).

Other `INPUT` examples I will give my students include (10 examples total, so 10 colors of paper needed):

1. `101010101010`
2. `100100100100`
3. `010101010101`
4. `010010010010`
5. `000000000000`
6. `111111111111`
7. `110011101001`
8. `100011001011`
9. `001101010010`

See separate document for pages which are ready for printing.

Saw

Input(s): This is the first Machine in the system, and you have the `INPUT` for the whole system, which is a 12-bit string.

Output(s): Two 6-bit strings L and R (one copy of L and two copies of R).

Instructions:

1. Split `INPUT` in half into the left half L and the right half R. Make sure they are clearly labeled as such.
2. Give a copy of R to COPIER and another copy to EXPANDER.
3. Give L to LIGHTSWITCH.

Copier

Input(s): 6-bit string called R.

Output(s): 6-bit string called L_*.

Instructions:

1. You should get a 6-bit string from SAW called R.
2. Copy R exactly as it is but relabel it clearly as L_*.
3. Give L_* to SUPERGLUE.

Expander

Input(s): 6-bit string called R.

Output(s): 8-bit string called E (for "expansion").

Instructions:

1. You should get a 6-bit string from SAW called R. We can label the bits as: $b_1b_2b_3b_4b_5b_6$.
2. Write out E which uses the bits from R, but is $E = b_1b_2b_4b_3b_4b_3b_5b_6$.
3. Clearly label your output as E, and give it to KEY.

Key

Input(s): 8-bit string called E.

Output(s): 8-bit string called $E \oplus K$.

Instructions:

1. You are given that $K = \texttt{01100101}$.
2. You should receive an 8-bit string from EXPANDER called E.
3. Perform XOR (exclusive or) on E and K to yield $E \oplus K$.
4. Clearly label your output as $E \oplus K$, and give it to SCISSORS.

Scissors

Input(s): 8-bit string called $E \oplus K$.

Output(s): Two 4-bit strings called I and II.

Instructions:

1. You should receive an 8-bit string called $E \oplus K$ from KEY.
2. Split$E \oplus K$ in half into the left half I and the right half II. They are each 4-bits now. Make sure they are clearly labeled as I and II.
3. Give I to MAP I, and give II to MAP II.

Map I

Input(s): 4-bit string called I.

Output(s): 3-bit string called I_*.

Instructions:

1. You should receive a 4-bit string from SCISSORS called I.
2. The following matrix is MAP I.

   The first row is labeled as position `0` and the second is labeled as position `1`.

   The columns are labeled positions `000,001,010,011,100,101,110,111`.

$$\begin{array}{cc} & \begin{array}{cccccccc} \texttt{000} & \texttt{001} & \texttt{010} & \texttt{011} & \texttt{100} & \texttt{101} & \texttt{110} & \texttt{111} \end{array} \\ \begin{array}{c} \texttt{0} \\ \texttt{1} \end{array} & \begin{pmatrix} 101 & 010 & 001 & 110 & 011 & 100 & 111 & 000 \\ 001 & 100 & 110 & 010 & 000 & 111 & 101 & 011 \end{pmatrix} \end{array}$$

3. The first bit in I tells you which row you look at in the MAP. Find the appropriate row.
4. The next three bits in I tell you which column you look at in MAP. Find the appropriate column.
5. Write down the 3-bit output in the appropriate row and column.
6. Label your output as I_*, and give it to WELD.

Map II

Input(s): 4-bit string called II.

Output(s): 3-bit string called II_*.

Instructions:

1. You should receive a 4-bit string from SCISSORS called II.
2. The following matrix is MAP II.
   The first row is labeled as position `0` and the second is labeled as position `1`.
   The columns are labeled positions `000,001,010,011,100,101,110,111`.

$$\begin{array}{c c}
 & \begin{array}{cccccccc} \texttt{000} & \texttt{001} & \texttt{010} & \texttt{011} & \texttt{100} & \texttt{101} & \texttt{110} & \texttt{111} \end{array} \\
\begin{array}{c} \texttt{0} \\ \texttt{1} \end{array} & \begin{pmatrix} 100 & 000 & 110 & 101 & 111 & 001 & 011 & 010 \\ 101 & 011 & 000 & 111 & 110 & 010 & 001 & 100 \end{pmatrix}
\end{array}$$

3. The first bit in II tells you which row you look at in the MAP. Find the appropriate row.
4. The next three bits in II tell you which column you look at in MAP. Find the appropriate column.
5. Write down the 3-bit output in the appropriate row and column.
6. Label your output as II_*, and give it to WELD.

Weld

Input(s): Two 3-bit strings I_* and II_*.

Output(s): One 6-bit string called F.

Instructions:

1. You should receive a 3-bit string I_* from MAP I and a 3-bit string II_* from MAP II.
2. Create a 6-bit string by putting the two 3-bit strings together in the following order: $F = I_*II_*$.
3. Clearly label your output as F, and give it to LIGHTSWITCH.

Lightswitch

Input(s): A 6-bit string called L, and a 6-bit string called F.

Output(s): A 6-bit string called R_*.

Instructions:

1. You should get a 6-bit string from SAW called L, and a 6-bit string from WELD called F.
2. Perform XOR (exclusive or) on L and F to yield $L \oplus F = R_*$.
3. Clearly label your output as R_*, and give it to SUPERGLUE.

Superglue

Input(s): A 6-bit string called L_* and a 6-bit string called R_*.

Output(s): This is the last Machine in the whole system, and you have the final `OUTPUT`, which is a 12-bit string.

Instructions:

1. You should get a 6-bit string from COPIER called L_* and a 6-bit string from LIGHTSWITCH called R_*.
2. Create a 12-bit string by putting the two 6-bit strings together in the following order: $L_*R_* = \texttt{OUTPUT}$.