Introduction to Modern Ciphers

Math 4175

§4.1.0. Product Cryptosystems

In order to study modern day cryptosystems, we need to understand the concept of combining two classical cryptosystems by forming their product.

For simplicity, we restrict ourselves to the cryptosystems where $\mathcal{P} = \mathcal{C}$.

A crypstosystem is called an endomorphic cryptosystem if $\mathcal{P} = \mathcal{C}$.

Definition: Let $S_1 = (\mathcal{P}, \mathcal{P}, \mathcal{K}_1, \mathcal{E}_1, \mathcal{D}_1)$ and $S_2 = (\mathcal{P}, \mathcal{P}, \mathcal{K}_2, \mathcal{E}_2, \mathcal{D}_2)$ be two endomorphic systems which have the same plaintext space $\mathcal{P}$ (and hence same cipher text space). We say that the systems S_1 and S_2 are equivalent ($S_1 \sim S_2$) iff $\{e_{k_1} : k_1 \in \mathcal{K}_1\} = \{e_{k_2} : k_2 \in \mathcal{K}_2\}$.

Definition: Let $S_1 = (\mathcal{P}, \mathcal{P}, \mathcal{K}_1, \mathcal{E}_1, \mathcal{D}_1)$ and $S_2 = (\mathcal{P}, \mathcal{P}, \mathcal{K}_2, \mathcal{E}_2, \mathcal{D}_2)$ be two endomorphic systems on the same set $\mathcal{P}$. The product cryptosystem of S_1 and S_2, denoted by $S_1 \times S_2$, is defined to be the cryptosystem:

$$S = (\mathcal{P}, \mathcal{P}, \mathcal{K}_1 \times \mathcal{K}_2, \mathcal{E}, \mathcal{D}).$$

§4.1.0. Product Cryptosystems

- So a key of the product cryptosystem has the form (k_1, k_2) where $k_1 \in \mathcal{K}_1$ and $k_2 \in \mathcal{K}_2$.
- For each key $k = (k_1, k_2)$, the encryption and decryption rules are defined by

$$e_{(k_1,k_2)}(x) = e_{k_2}(e_{k_1}(x)) \qquad \text{and} \qquad d_{(k_1,k_2)}(y) = d_{k_1}(d_{k_2}(y)).$$

- We have,

$$\begin{aligned} d_{(k_1,k_2)}(e_{k_2}(e_{k_1}(x))) &= d_{(k_1,k_2)}(e_{k_2}(e_{k_1}(x))) \\ &= d_{k_1}(d_{k_2}(e_{k_2}(e_{k_1}(x)))) \\ &= d_{k_1}(e_{k_1}(x)) = x \end{aligned}$$

§4.1.0. Example of Product Cryptosystems

Example 1: Consider the Vigenere cipher with the key $k_1 =$ **book** and the Permutation cipher with the key

$$k_2 = \begin{array}{|c|c|c|c|} \hline 1 & 2 & 3 & 4 \\ \hline 4 & 1 & 2 & 3 \\ \hline \end{array}$$

Let us encrypt **CRYPTOGRAPHY** in the product cipher $V \times P$ with the product key (k_1, k_2).

	C	R	Y	P	T	O	G	R	A	P	H	Y
	2	17	24	15	19	14	6	17	0	15	7	24
$+k_1$	1	14	14	10	1	14	14	10	1	14	14	10
	3	5	12	25	20	2	20	1	1	3	21	8
π_P :	25	3	5	12	1	20	2	20	8	1	3	21
	Z	D	F	M	B	U	C	U	I	B	D	V

For decryption, reverse the process.

§4.1.0. Example of Product Cryptosystems

Example 2: Now let us encrypt **CRYPTOGRAPHY** in the product cipher $P \times V$ with the product key (k_2, k_1).

	C	R	Y	P	T	O	G	R	A	P	H	Y
	2	17	24	15	19	14	6	17	0	15	7	24
π_P :	15	2	17	24	17	19	14	6	24	0	15	7
$+k_1$	1	14	14	10	1	14	14	10	1	14	14	10
	16	16	5	8	18	7	2	16	25	14	3	17
	Q	Q	F	I	S	H	C	Q	Z	O	D	R

Notice that we do not get same ciphertext as in example 1.
For decryption, again reverse the process.

§4.1.0. Example of Product Cryptosystems

A Multiplicative Cipher is a cryptosystem where

- $\mathcal{P} = \mathcal{C} = \mathbb{Z}_n$ and let $\mathcal{K} = \{\alpha \in \mathbb{Z}_n : \gcd(\alpha, n) = 1\}$.
- For each $\alpha \in \mathcal{K}$, define the encryption and decryption rules by $e_\alpha(x) = \alpha x$ and $d_\alpha(y) = \alpha^{-1} y$.

.

Example 3: Let M be the multiplicative cipher over $\mathbb{Z}_{26}$. For $7 \in \mathcal{K}$, we have $e_7(x) = 7x$ and $d_7(y) = 7^{-1}(y) = 15y$.

Example 4: Let S be the shift cipher on $\mathbb{Z}_{26}$ and consider the product cipher $M \times S$. Now $(7, 5)$ is a key in the product cipher $M \times S$ with the encryption rule:

$$e_{(7,5)}^{M \times S}(x) = e_5^S(e_7^M(x)) = e_5^S(7x) = 7x + 5 \mod 26$$

which is the encryption rule for Affine cipher with the key $(7, 5)$. So $M \times S$ is equivalent to the Affine cipher.

§4.1.0. Example of Product Cryptosystems

Example 5: Now consider the product cipher $S \times M$. Then $(5, 7)$ is a key in the product cipher $S \times M$ with the encryption rule:

$$e^{S\times M}_{(5,7)}(x) = e^M_7(e^S_5(x)) = e^M_7(x + 5) = 7x + 35 = 7x + 9 \mod 26$$

which is again an encryption rule for Affine cipher with the key $(7, 9)$ (Warning: notice NOT $(7, 5)$).

Therefore, $M \times S$ and $S \times M$ are both equivalent to the Affine Cipher over $\mathbb{Z}_{26}$, and conclude that $M \times S \sim S \times M \sim \mathcal{A}$, the Affine cipher.

But notice that the key $(5, 7) \in S \times M$ is not same as $(7, 5) \in M \times S$, but is equal to $(7, 9) \in M \times S$.

That is, the multiplicative and shift ciphers commute, since each product cryptosystem is equivalent to the affine cipher, despite the fact that individual keys do not commute.

§4.1.0. Product Cryptosystems

Definition: Let S_1, S_2 be two endomorphic cryptosystems over $\mathcal{P}$. If $S_1 \times S_2$ and $S_2 \times S_1$ yield equivalent cryptosystems, then we say that S_1 and S_2 commute.

We have to be careful, because what we are saying is that the set of encryption functions of commuting product cryptosystems are the same; i.e.,

$$\{e_{k_2} \circ e_{k_1} : (k_1, k_2) \in \mathcal{K}_1 \times \mathcal{K}_2\} = \{e_{k_1'} \circ e_{k_2'} : (k_2', k_1') \in \mathcal{K}_2 \times \mathcal{K}_1\}$$

but we are **NOT** saying that $e_{k_2} \circ e_{k_1} = e_{k_1} \circ e_{k_2}$ for each (k_1, k_2).

Product Operation is always Associative: That is

$$S_1 \times (S_2 \times S_3) = (S_1 \times S_2) \times S_3$$

§4.1.0. Product Cryptosystems

What if we take the product of the standard Affine Cipher with itself?

In other words, what is $\mathcal{A} \times \mathcal{A}$? This is called iterating the cipher.

For example, consider the keys $(3, 2)$ and (5,6) in the Affine cipher. Then $((3, 2), (5, 6))$ is a key in the product cipher $\mathcal{A} \times \mathcal{A}$ with the encryption rule:

$$\begin{aligned} e^{\mathcal{A}\times\mathcal{A}}_{((3,2),(5,6))}(x) &= e^{\mathcal{A}}_{(5,6)}(e^{\mathcal{A}}_{(3,2)}(x)) \\ &= e^{\mathcal{A}}_{(5,6)}(3x + 2) \\ &= 5(3x + 2) + 6 \\ &= 15x + 16 \\ &= e^{\mathcal{A}}_{(15,16)}(x) \end{aligned}$$

Indeed, $\mathcal{A} \times \mathcal{A} \sim \mathcal{A}$.

§4.1.0. Product Cryptosystems

We still get the same collection of encryption functions in $\mathcal{A} \times \mathcal{A}$ as the standard Affine Cipher $\mathcal{A}$, which means that there is no point in applying the Affine Cipher several times. It would just take more work to exchange/apply several keys, and does not add any security.

More generally, a cryptosytem is called idempotent if S is equivalent to $S \times S$.

The following are idempotent:

- Shift Ciphers, Affine Ciphers, Substitution Ciphers.
- Vigenere for fixed m.
- Hill Ciphers for fixed m.
- Transposition Cipher for fixed m.

§4.1.0. Product Cryptosystems

Remark: If S_1 and S_2 are both idempotent, and they commute, then $S_1 \times S_2$ will also be idempotent. This follows from the following:

$$\begin{aligned}
(S_1 \times S_2) \times (S_1 \times S_2) &= S_1 \times (S_2 \times S_1) \times S_2 \text{ (associativity)} \\
&= S_1 \times (S_1 \times S_2) \times S_2 \text{ (commutative)} \\
&= (S_1 \times S_1) \times (S_2 \times S_2) \text{ (associativity)} \\
&= S_1 \times S_2 \qquad\qquad \text{(idempotent)}
\end{aligned}$$

Of course, if a cryptosystem S is idempotent, then there is no point in using the product $S^2 = S \times S$, as it requires an extra key but provide no additional security.

§4.1.0. Product Cryptosystems

- So, if S_1 and S_2 are both idempotent, and we want $S_1 \times S_2$ to be non-idempotent, then it is necessary that S_1 and S_2 do not commute.
- If a cryptosystem is not idempotent, there is a potential increase in security by iterating it several times.
- This idea will be used later, for example, in Data Encryption Standard, which consists of 16 iterations.
- A common technique that will be used later is to take the product of substitution-type ciphers with permutation-type ciphers, because they do not commute (see an example in the next slide).
- In the next slide, we give an example of a non-idempotent cryptosystem, that is, $S \times S$ that is not equivalent to S.

§4.1.0. A Non-Idempotent Cryptosystem

- **Example:** In our example, both the plaintexts and ciphertexts are blocks of 4-bits strings consists of 0's and 1's. In other words, $\mathcal{P} = \mathcal{C} = (\mathbb{Z}_2)^4$. So $x \in \mathcal{P}$ iff x is a 4-bits string.

- Since we will use these 4-bits strings often in the next few sections, it is better to familiarize with them.

- There are sixteen of them starting from 0000 to 1111 corresponding to 0 to 15 in our decimal number system as listed in the table given in the next slide.

- For example, 1101 corresponds to $1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0 = 8 + 4 + 1 = 13$.

- But **13** may represent the single block 1101 or two blocks 0001 0011 (for 1 and 3). In order to avoid this confusion, we will use hexadecimal notation as indicated in the next table.

Hexadecimal	Decimal	Binary (strings of length 4)
0	0	0000
1	1	0001
2	2	0010
3	3	0011
4	4	0100
5	5	0101
6	6	0110
7	7	0111
8	8	1000
9	9	1001
A	10	1010
B	11	1011
C	12	1100
D	13	1101
E	14	1110
F	15	1111

§4.1.0. A Non-Idempotent Cryptosystem

- In our example, we will use two permutations, the substitution permutation denoted by π_S and the index permutation denoted by π_P as given below.
- Given a 4-bit string $x = x_1x_2x_3x_4$ we break this string into two 2-bits substrings; i.e., we regard x as

$$x = x_1x_2||x_3x_4$$

and then apply the substitution permutation π_S for each pair as defined below:

z	input	00	01	10	11
$\pi_S(z)$	output	01	11	10	00

- We apply the index permutation π_P to the entire 4-bits as below:

i	1	2	3	4
$\pi_P(i)$	1	3	2	4

This means that $v_1v_2||v_3v_4 \mapsto v_1v_3||v_2v_4$.

§4.1.0. A Non-Idempotent Cryptosystem

Encryption Rule for S:

Given a plaintext $x = x_1x_2||x_3x_4$ and a 4-bit string key $\mathbf{K} = k_1k_2||k_3k_4$:

- XOR (Add mod 2) $x_1x_2||x_3x_4$ and $k_1k_2||k_3k_4$ bitwise. Denote the resulting string by $u_1u_2||u_3u_4$.
- We apply π_S to each substring u_1u_2 and u_3u_4. Let $v_1v_2 = \pi_S(u_1u_2)$ and $v_3v_4 = \pi_S(u_3u_4)$.
- Then apply the index-permutation π_P to $v_1v_2||v_3v_4$. This yields, $v_1v_3||v_2v_4$ which we call as $y_1y_2||y_3y_4$. This resulting 4-bit string $y_1y_2||y_3y_4$ is the cipher text for $x = x_1x_2||x_3x_4$.

Decryption Rule: Reverse the above procedure.

§4.1.0. A Non-Idempotent Cryptosystem

Step 1: Encrypt **01** $= 0000\ 0001$ on the cipher $S \times S$ (using two rounds) with keys $\mathbf{k}_1 = 0000$ and $\mathbf{k}_2 = 0000$ to show that the corresponding cipher text is $0010\ 1010 = \mathbf{2A}$.

$$
\begin{array}{rcc}
P: & 0000 & 0001 \\
 & \oplus & \oplus \\
k_1: & 0000 & 0000 \\
 & 0000 & 0001 \\
\pi_S: & 0101 & 0111 \\
\pi_P: & 0011 & 0111 \\
 & \oplus & \oplus \\
k_2: & 0000 & 0000 \\
 & 0011 & 0111 \\
\pi_S: & 0100 & 1100 \\
\pi_P: & 0010 & 1010
\end{array}
$$

§4.1.0. A Non-Idempotent Cryptosystem

Step 2: In order to verify $S \times S$ is not equivalent to S, we have to show that none of the keys takes the plaintext **01** to the cipher text **2A** in S (in one round).

$$
\begin{array}{rcc}
P: & 0000 & 0001 \\
 & \oplus & \oplus \\
k: & ???? & ???? \\
 & 0011 & 0111 \\
\pi_S: & 0100 & 1100 \\
\pi_P: & 0010 & 1010
\end{array}
$$

Is $k = 0011$ or $k = 0110$? So no single key will work. Therefore $S \times S$ is not equivalent to S or in other words, S is a non-idempotent cipher.

§4.1.1. Modern Block Ciphers

- Most modern day block ciphers are product ciphers. Recall that block ciphers transform a fixed size of blocks of plaintext into same size blocks of cipher text.
- As we saw in the previous example of non-idempotent cryptosystem, product ciphers frequently incorporate a sequence of permutation and substitution operations.
- A commonly used design is that of an iterated cipher.
- In an iterated cipher, we apply a simple encryption function iteratively a number of times (called rounds), say N. So, in an iterated cipher, we first fix the number of rounds N.
- The encryption function applied above is called the round function, say g.

§4.1.1. Modern Block Ciphers

- From a chosen key K of specific length, a set of N subkeys, also called round keys, are derived and denoted by $K^1, \cdots, K^N$. Any algorithm that is used to generate the round keys is called the key schedule.
- The round function g takes two inputs: a current state, denoted by w^{r-1}, and a round key K^r and produces the next state as an output, that is, $w^r = g(w^{r-1}, K^r)$.
- In order to have a possible decryption, the function $g(., K^r)$ must be invertible for each fixed K^r.
- The initial state w^0 is defined to be the plaintext x.
- The cipher text y is defined to be the final state after all N rounds have been performed.

§4.1.1. Modern Block Ciphers

Hence the encryption of an iterated cipher is carried out as follows:

$$
\begin{aligned}
w^0 &\longleftarrow x \\
w^1 &\longleftarrow g(w^0, K^1) \\
w^2 &\longleftarrow g(w^1, K^2) \\
&\vdots \\
w^{N-1} &\longleftarrow g(w^{N-2}, K^{N-1}) \\
w^N &\longleftarrow g(w^{N-1}, K^N) \\
y &\longleftarrow w^N
\end{aligned}
$$

The decryption is accomplished by reversing the operations, which is possible since g is invertible for each fixed round key.

§4.1.1. Modern Block Ciphers

The decryption of an iterated cipher can be carried out as follows:

$$
\begin{aligned}
w^N &\longleftarrow y \\
w^{N-1} &\longleftarrow g^{-1}(w^N, K^N) \\
w^{N-2} &\longleftarrow g^{-1}(w^{N-1}, K^{N-1}) \\
&\vdots \\
w^1 &\longleftarrow g^{-1}(w^2, K^2) \\
w^0 &\longleftarrow g^{-1}(w^1, K^1) \\
x &\longleftarrow w^0
\end{aligned}
$$

We will discuss in the next few sections some of the iterated ciphers and their cryptanalysis.