第五次作业

1. 工作模式基础

	ECB	CBC	CFB	OFB	CTR
加密递推式	$C_i = E(k, P_i)$	$C_1 = E(k, (P_1 \oplus IV))$ $C_i = E((P_i \oplus C_{i-1}), k)(i \neq 1)$	$I_1 = IV$ $I_i = LSB_{b-s}(I_{i-1}) \mid\mid C_{i-1}$ $j = 2, \dots N$ $O_i = E(K, I_i), j = 1, \dots N,$ $C_i = P_i \oplus MSB_s(O_i), j = 1, \dots, N$	$I_i = E(K, I_{i-1}),$ $O_i = E(I_i, K),$ $C_i = O_i \oplus P_i$	$C_i = P_i \oplus E(K, Counter)$
解密递推式	$P_i = D(k, C_i)$	$P_1 = IV \oplus D(K, C_1)$ $P_i = C_{i-1} \oplus D(k, C_i)(i \neq 1)$	$I_1 = IV$ $I_i = LSB_{b-s}(I_{i-1}) \mid\mid C_{i-1}$ $j = 2, \dots N$ $O_i = E(K, I_i), j = 1, \dots N,$ $P_i = C_i \oplus MSB_s(O_i), j = 1, \dots, N$	$I_i = E(K, I_{i-1}),$ $O_i = E(I_i, K),$ $P_i = O_i \oplus C_i$	$P_i = C_i \oplus D(K, Counter)$
初始向量（时变值，计数器等）（Y/N）	N	Y	Y	N	N
加密并行性（Y/N）	Y	N	N	N	Y
解密并行性（Y/N）	Y	Y	Y	N	Y
需要分组密码解密模块（Y/N）	N	Y	Y	Y	N
含有反馈（Y/N）	N	Y	Y	Y	N

2. 错误恢复能力

（1）错误传播

- **ECB**：对后续无影响，只影响自身；解密时，由于压根没有反馈，所以除了出错的那一组，其余解密均正确
- **CBC**：影响后面所有的密文分组，解密时，$E((P_{i+1} \oplus C_i'), K) = C_{i+1}'$，$D(C_{i+1}', K) \oplus C_i' = P_{i+1}$,解得的明文无误，出错的明文只有第i个和$i+1$个。
- **CFB**：影响后续$\lceil \frac{64}{j} \rceil$个消息，因为要将出错的$s$比特的内容移出移位寄存器时（每次左移$j$位）才能保证解密所得的明文正确，即会影响$\lceil \frac{64}{j} \rceil$个消息。但解密时，除了$i$个会出现问题，其他的都不会出错（因为从第$i+1$个开始，参与加密的是$C_{i+1}$ 等，加密结果也是P_{i+1}'等，解密是加密的逆运算，自然能还原C_{i+1}不会出错
- **OFB**：对后续无影响，只影响自身，解密时第i个也会发生错误，其余的不会发生错误
- **CTR**：对后续无影响，只影响自身，解密时第i个也会发生错误，其余的不会发生错误

（2）传输错误

- **ECB**：只有第i个会出错，因为解密都是分组独立的，第i个密文错误不会影响到其余分组
- **CBC**：会影响到第i和$i+1$个，因为P_i作为解密的第i个输入，会导致第i个解密错误，同样错误的C_i'会影响到第$i+1$组的异或运算部分，造成错误
- **CFB**：第i个消息后面的都有可能出错，因为P_i的错误会导致C_i的错误，传导到$i+1$个，以此类推，后面的消息解密都有可能出错
- **OFB**：只会影响第i个消息，因为从$i+1$个开始，从第i个传导进去的是密钥操作运算，并没有错误
- **CTR**：只会影响第i个消息，因为每个分组加密解密都是独立的，错误不会影响

(3)

- 对于**OFB**来说，如在密文中取1比特的补，那么在恢复的明文中相应位置的比特也为原比特的补。因此使得敌手有可能通过对消息校验部分的篡改和对数据部分的篡改来实现攻击；并且无法进行并行运算
- 对于**CTR**来说，不存在通过多次错误注入来破解的可能；对并行运算的支持也很好

3.工作模式的填充

`PKCS 7`是一种密码学中的标准，它规定了数字证书的格式、数字签名和加密信息的语法以及加密的填充方式。

`PKCS 7`的填充方式是一种基于块的填充方式。其基本原理是在明文数据的末尾添加一些填充字节，使其长度达到块长度的整数倍。数据长度不足数据块长度时,缺几位补几个几。

例如：对于`AES128`算法其数据块为`16Byte`（数据长度需要为`16Byte`的倍数）,如果数据为`00112233445566778899AA`一共`11个Byte`,缺了5位,采用`PKCS 7`方式填充之后的数据为`00112233445566778899AA0505050505`。

如果明文长度刚好是块长度的整数倍，那么需要添加一个块长度的填充字节，这样做的目的是为了区分有效数据和补齐数据，这样牺牲了数据长度的做法可以更为灵活透明的去解包数据，发送端和接收端不需要约定好`blockSize`，接收端总能通过数据包的最后一个字符得到填充的数据长度。

4.CBC Padding攻击

(1)

16 c'_{i-1} 15 $O(y_1) = 1$ 256

(2)

15 66 异或 15

(3)

01 14 14 $O(y_2) = 1$ 256 14 66 14

(4)

$\ldots t_{15}$ 倒序 c_i p_i