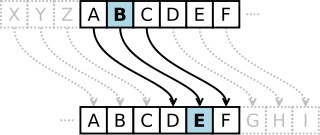
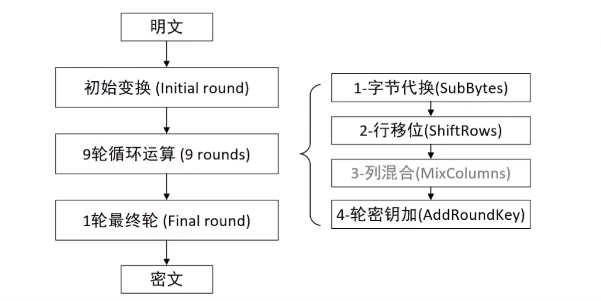
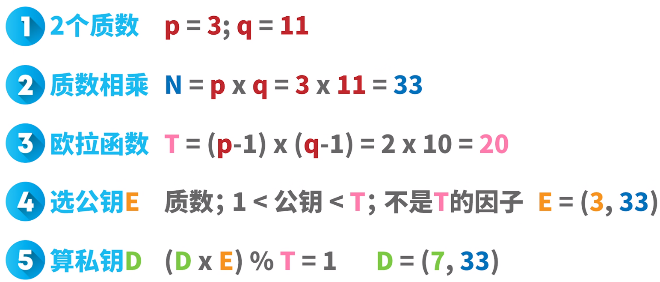
1. 对称加密：
2. 凯撒加密：以偏移量为密钥进行加密与解密，特点是非常简单，但是无法改变字母出现的频率，也导致可以通过分析字母出现频率快速破译。
3. AES（虽然看了俩小时b大和wiki，但是也没完全理解）：将明文划分为16Byte并组成4\*4的矩阵进行加密。在初始变换中，使用主密钥的子矩阵对明文进行异或加密，即轮密钥加。然后进行9轮循环运算（轮数由密钥长度决定，其中128bit密钥为9轮运算，而192bit和256bit密钥则会有更多轮次的加密）。每一环节通过固定的S-box进行字节代换，再对每一行执行不同大小的行移位，然后再将所得矩阵左乘一个给定的4\*4矩阵作列混合，并再次通过轮密钥加进行加密，原矩阵在轮密钥加环节中对回合密钥矩阵进行异或，除了初始变换使用主密钥以外，都是通过生成回合密钥进行异或，而回合密钥矩阵是密钥生成方案，即异或，轮替等运算生成的，可以延长主密钥。AES的加密强度高于已经被淘汰的DES和DES的升级版3DES，具有较高的防破解程度，被广泛使用，如WinRAR的加密方案中除了官方废弃的ZIP2.0以外就有128bit和256bit的AES。（上述加密可能有啥地方出错了，信安大佬可以随意指出）。
4. 非对称加密：
   1. RSA加密：RSA是一种非对称加密，存在公钥和私钥的概念，具体加密方法为将字符的unicode码作为底数，对公钥快速幂，得到密文。密文传输后再作私钥快速幂即可取得明文。而为了防范中间人攻击，会进一步应用哈希算法进行签名认证。这个签名的过程利用了公钥私钥可以互相加密解密的性质，即发送信息时，用自己的私钥给加密后的数据的哈希值加密，接收信息时，用公钥解密哈希值，与加密后数据的哈希值比对，如果相同，便可以确认公钥私钥匹配，即数据没有经过中间人的篡改。而其保密性与难破解性的数学基础是这条幂次取余运算中，已知b,e,m时可以快速得出c，而已知c,b,m时却很难通过运算快速得到幂次e，也确保了当公钥和私钥中E、D为大数时的安全性了。存在被破解的512bit和768bit的RSA算法，所以一般使用2048bit的RSA加密以确保安全性。
   2. 迪菲-赫尔曼密钥交换：虽然说我不太确定这是不是一种非对称加密，但是其他的我实在看不懂。迪菲-赫尔曼密钥交换是一种在公共信道里交换密钥的算法，主题算法基础就是，两人通过相互发送和的值来构成密钥。而当a，b，p都是很大的值（a,b各大于256bit，为通信者任意取的数，p为大于1024bit的质数，IETF RFC3526文档中遍提供了几个）时，便很难通过暴力破解密钥。而在交换了密钥之后，就可以通过对称加密算法进行交流了。

2、为什么非对称加密算法的密钥很难破解？密钥的安全性有哪些理论依据？

非对称加密算法的难破解性来自于大数难以质因数分解的特性，对一个大数质因数分解需要耗费大量的算力，而知道两个质数却能够很简单地计算出两数的乘积，无论这两个质数有多大。也就是说这是一个类单向的函数运算。

3、如果让你设计一个非对称加密算法，你会注意哪些方面？

设计时应当找到一个带后门的单向函数，即知道x时可以快速算出f(x)，而知道f(x)却不能在短时间内推出x的值，但是可以通过知道后门y和f(x)来计算出x。最常见的就是质数相乘和大数分解为两质数的过程。而在非对称加密的过程中，公钥即为x，私钥即为y，发送方通过x对明文进行加密并发送加密后的f(x)，而f(x)无法被中途破译。在收到f(x)后通过y对f(x)进行解密得到明文。

附加题1、

散列加密是一种将任意长度明文加密成固定长度密文的加密算法，也叫hash加密，如常用的MD5加密和SHA-1加密。一般可用于文件的校验，若校验码不同则可确定此校验码对应的文件并非拿到的文件。同时hash也可以用于密码储存，如互联网平台在储存用户名与密码时，可以选择将数据进行hash加密，在验证时只需要比对hash码即可非常大概率地确保安全，也能够做到在信息泄露时不至于将明文密码直接泄露出去。不过根据基本的数学映射思维，既然密文长度固定，也就代表了不同的明文可以对应相同的密文，也就证明存在极小概率使得错误密码的hash密文恰好为原密码的hash密文。

附加题2、

以下为WYL散列加密算法。

对于任意长度的明文M，首先将其补为8Byte的整数倍，不足处补00，再每64Byte为一组进行加密。加密过程如下：①将64byte变成8\*8的矩阵，再行列转置，即原a2,1的数据变到a1,2,。②对于初始密文00 01 02 03 04 05 06 07 08，先将密文整体右移a1,1位，如位移1位则是将全体00000000 00000001 00000010……变成00000000 00000000 10000001……，然后再由a1,1对密文第1 byte作异或运算，a1,2对密文第2 byte作异或运算……③对于每一行都作位移、异或的运算。④执行完一个8\*8的矩阵后将密文左右翻转，再次执行下一个矩阵。

最终对于任意长度的明文，都能得到一串8Byte大小的密文，虽然我不确定这个密文的抗碰撞性好不好，但是至少可以做到hash加密。