**Java实现Base64加密**

**1. 密码常用术语（一）**

**明文：待加密信息。**

**密文：经过加密后的明文。**

**加密：明文转换为密文的过程。**

**加密算法：明文转为密文的转换算法。**

**加密密钥：通过加密算法进行加密操作用的密钥。**

**解密：将密文转为明文的过程。**

**解密算法：密文转为明文的算法。**

**解密密钥：通过解密算法进行解密操作用的密钥。**

**密码常用术语（二）**

**密码分析：截获密文者试图通过分析截获的密文从而推断出原来的明文或密钥的过程。**

**主动攻击：攻击者非法入侵密码系统，采用伪造、修改、删除等手段向系统注入假消息进行欺骗。（对密文有破坏作用）**

**被动攻击：对一个保密系统采取截获密文并对其进行分析和攻击。（对密文没有破坏作用）**

**密码体制：由明文空间、密文空间、密钥空间、加密算法和解密算法五部分构成。**

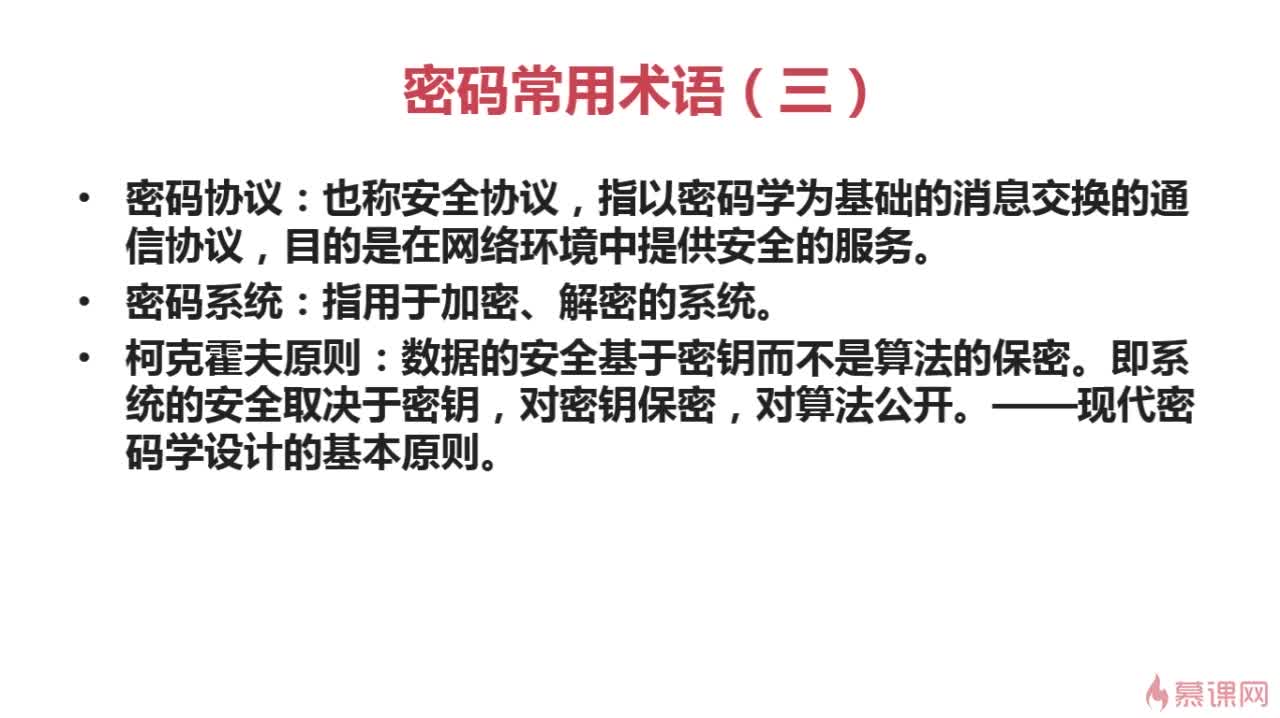
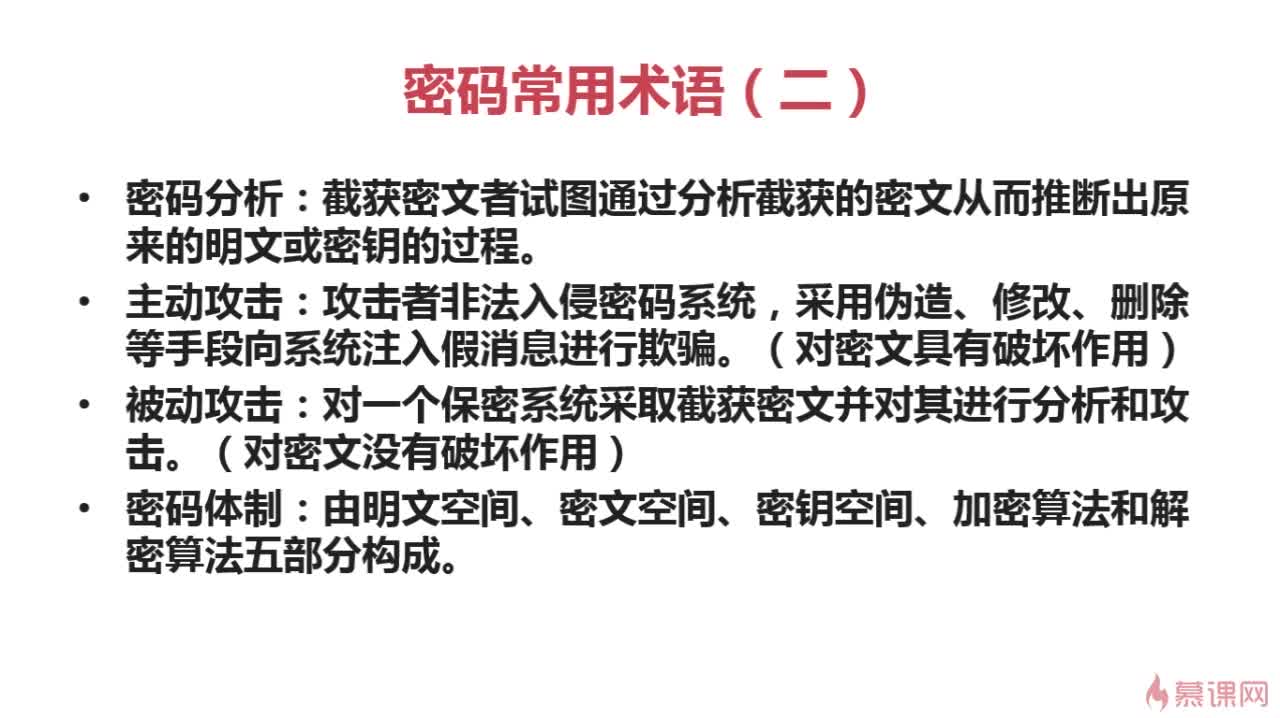
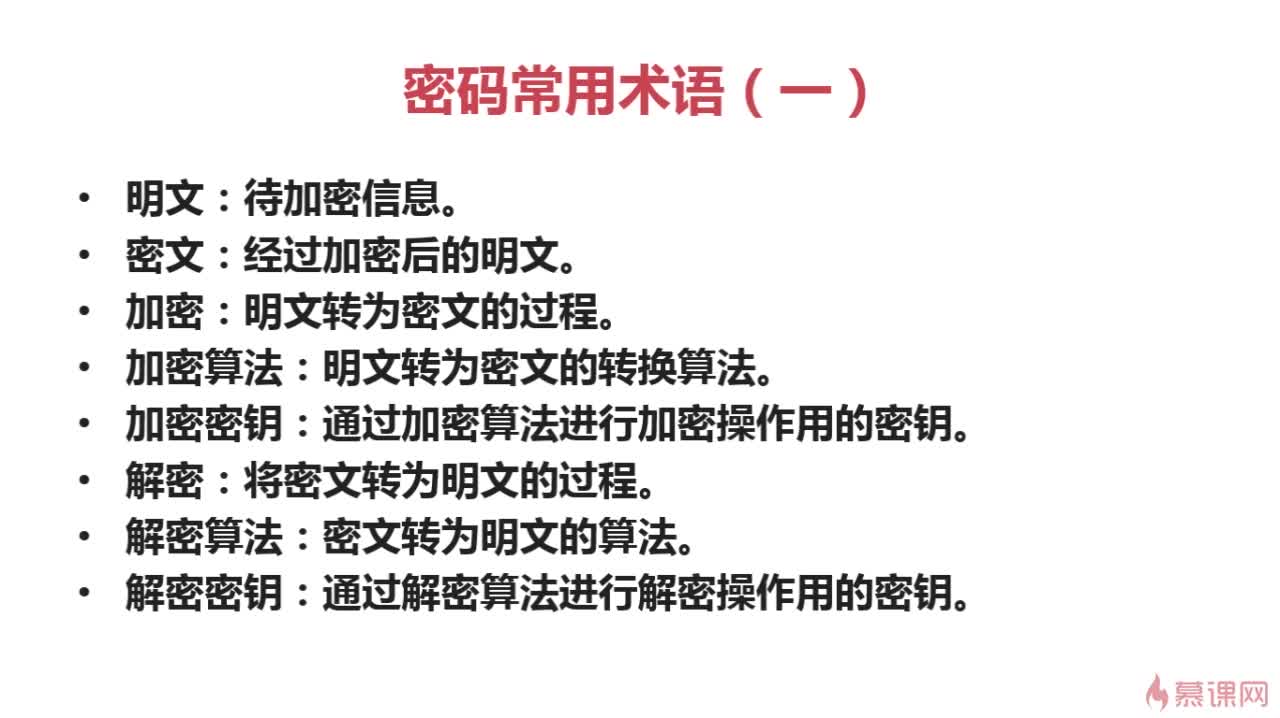
**密码常用术语（三）**

**密码协议：也称安全协议，指以密码学为基础的消息交换的通信协议，目的是在网络环境中提供安全的服务。**

**密码系统：指用于加密、解密的系统。**

**柯克霍夫原则：数据的安全基于密钥而不是算法的保密。即系统的安全取决于密钥，对密钥保密，对算法公开。--现代密码学设计的基本原则。**

**Ps：柯克霍夫原则公开算法是为了让这算法有更好的发展或是否存有一些BUG，而对密钥保密是为了保护已加密的信息的安全。**

****

**2. 密码分类--时间**

**古典密码：以字符为基本加密单元。**

**现代密码：以信息块为基本加密单元。**

**保密内容分类：**

**受限制算法：算法的保密性基于保持算法的秘密**

**基于密钥算法：算法的保密性基于对密钥的保密**

**密码体制：**

**对称密码:指加密密钥与解密密钥相同**

**非对称密码：加密密钥与解密密钥不同，密钥分为公钥、私钥**

**分组密码:指加密时将明文分为固定长度的组，用同一密钥和算法对每一块加密，输出也是固定长度的密文，多用于网络加密**

**流密码：也称序列密码，指加密时每次加密一位或者一个字节明文**

**散列函数又称，哈希函数、消息摘要函数（单向函数）：用于验证数据的完整性**

**特点：长度不受限制、哈希值容易计算、散列函数过程不可逆**

**散列函数的相关算法（MD5、SHA、MAC）**

**数字签名：主要针对以数字的形式存储的信息进行的处理，会产生一种带有操作者身份信息的代码**

**执行签名数字的实体被称为签名者，执行签名当中的算法被称为签名算法。**

**散列函数（哈希函数或单向函数）**

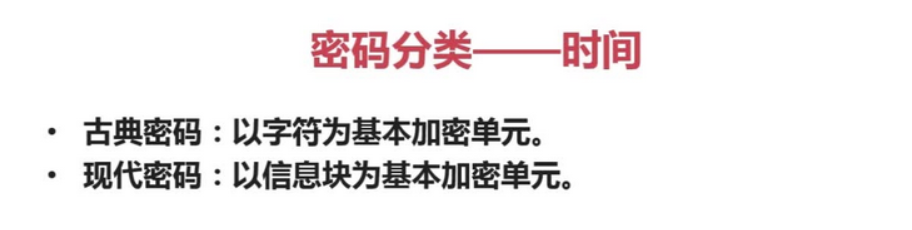
**-作用：用来验证数据的完整性。**

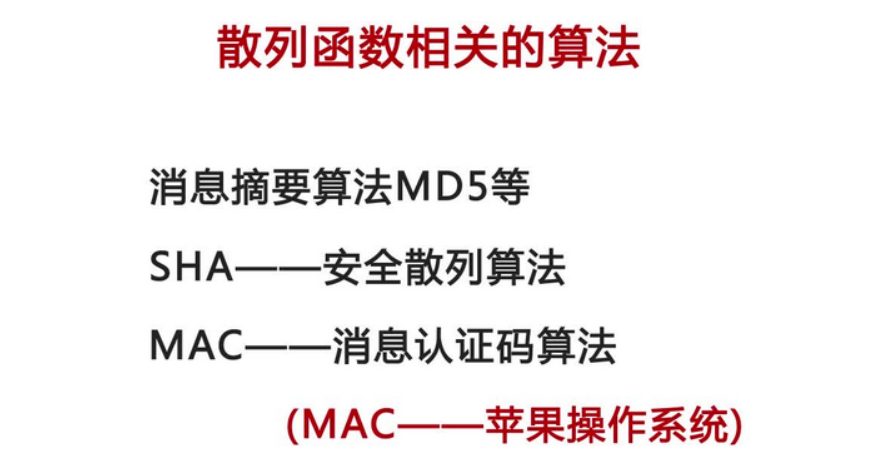
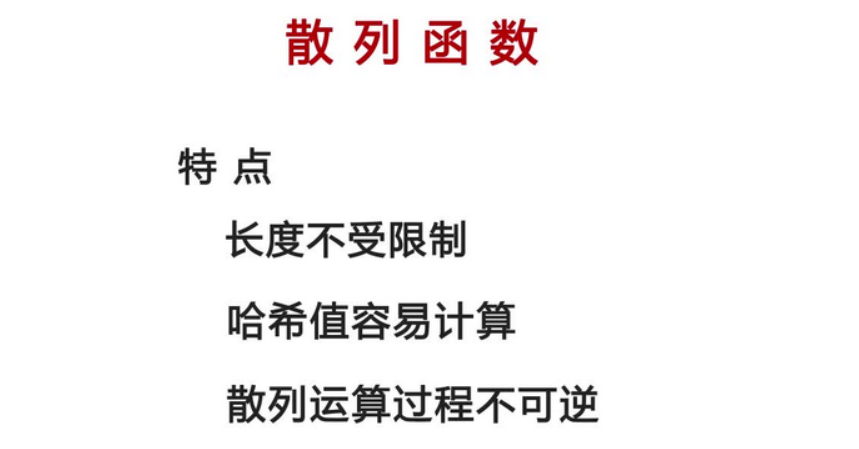
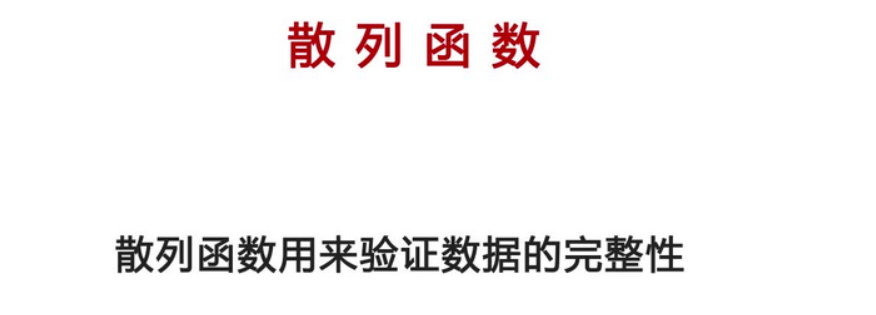
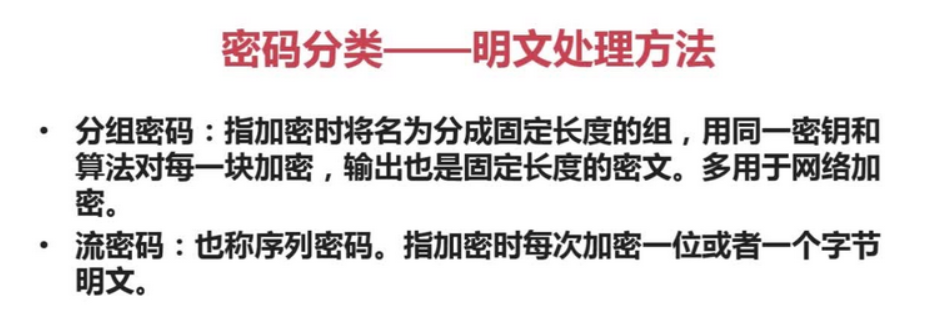
**-特点：1、长度不受限制；2、哈希值容易计算；3、散列运算过程不可逆（被称为单向函数的原因）。**

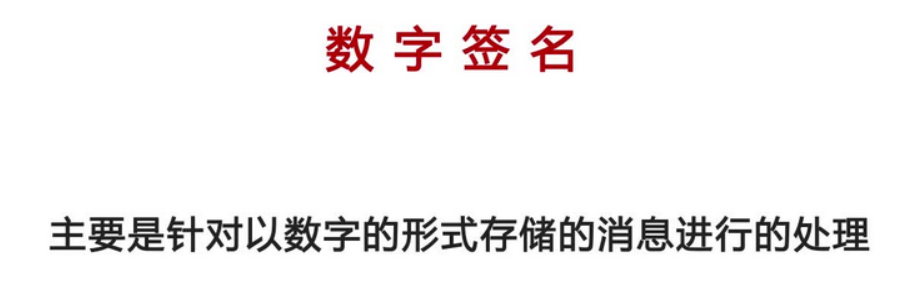
**明文处理方法：**

**-分组密码：指加密时将名为分成固定长度的组，用同一密钥和算法对每一块加密，输出也是固定长**

**散列函数是为了验证数据的完整性，并不是用来加解密的 长度不受限制，运算过程不可逆，单向**

****

****

****

**3. OSI模型是(Open System Interconnection):**

**开放式系统互联,该模型定义了不同计算机互联的标准,是设计和描述计算机网络通信的基本框架.**

**OSI模型把网络通信的工作分为7层,分别是物理层、数据链路层、网络层、传输层、会话层、表示层和应用层.**

**安全服务:安全服务指的是加强网络信息系统安全性,对抗安全攻击而采取的一系列措施.**

**安全机制:安全服务采取一系列措施使用的机制.**

**安全机制会对应网络通信的某一层,或某几层.针对每一层的处理方式未必一样.**

**TCP/IP传输控制协议,是一个协议族,它分为4层:网络接口层、网络层、传输层、应用层,每层又有各自的协议.**

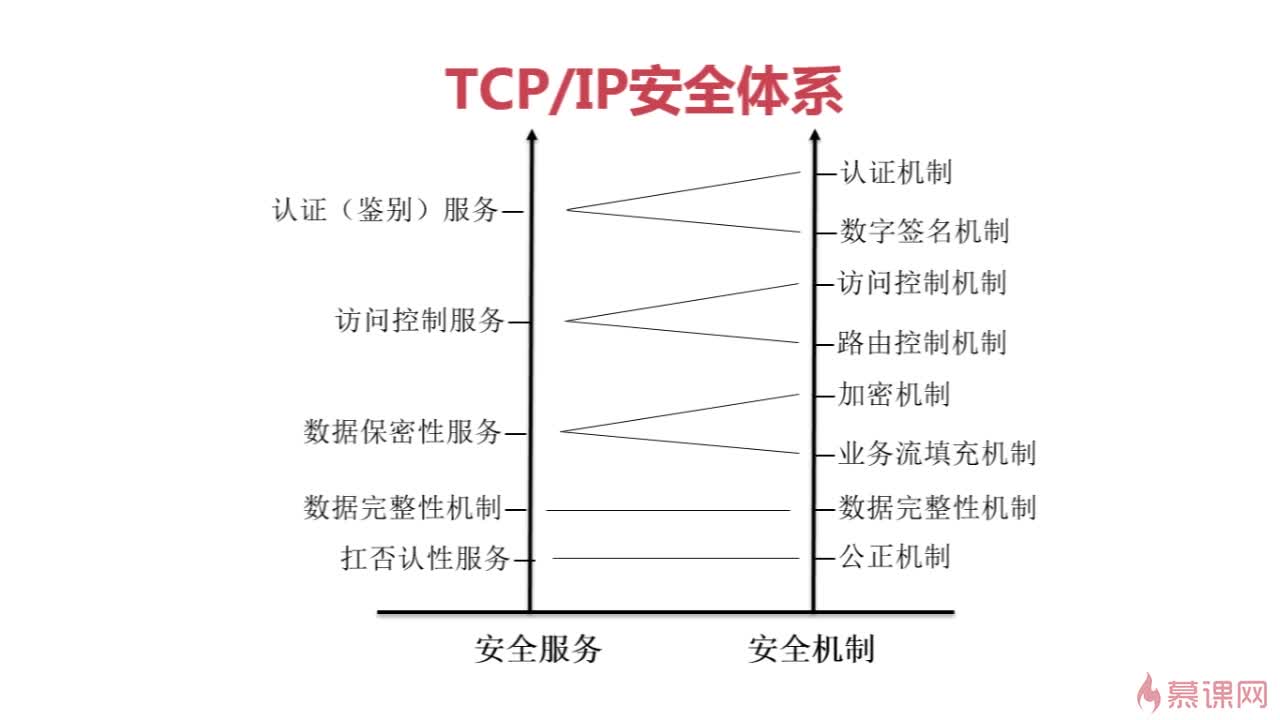
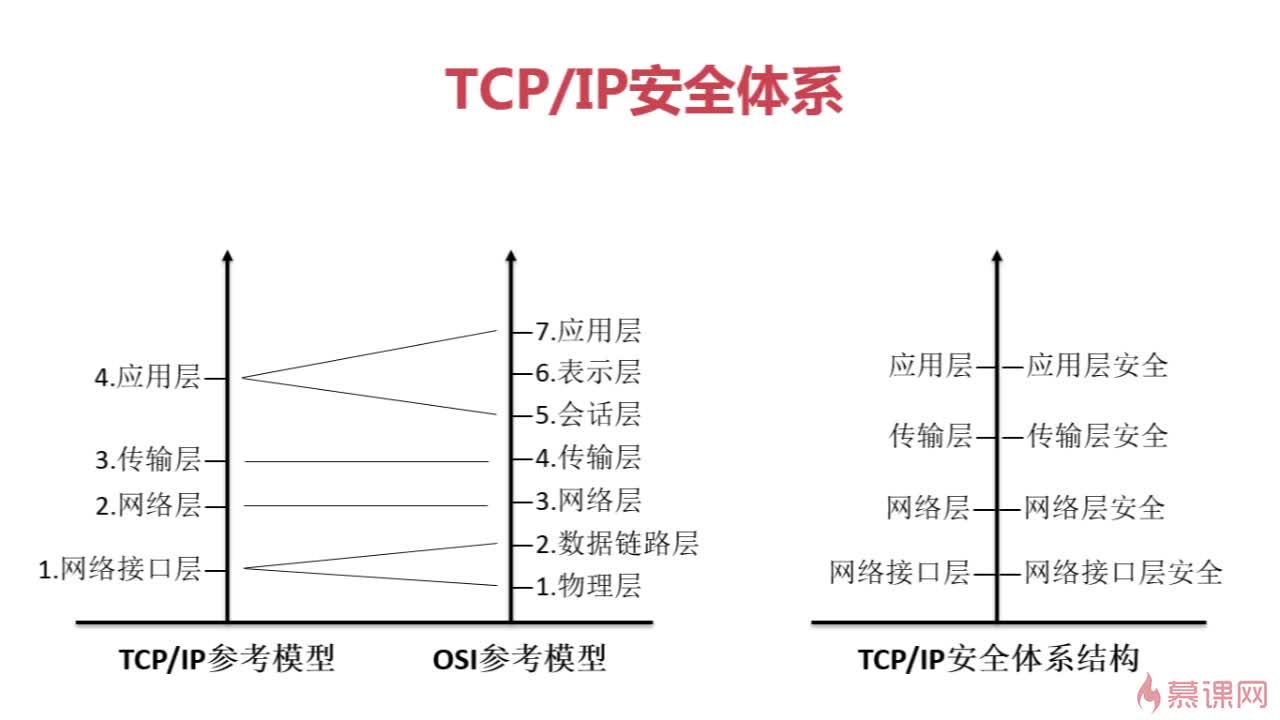
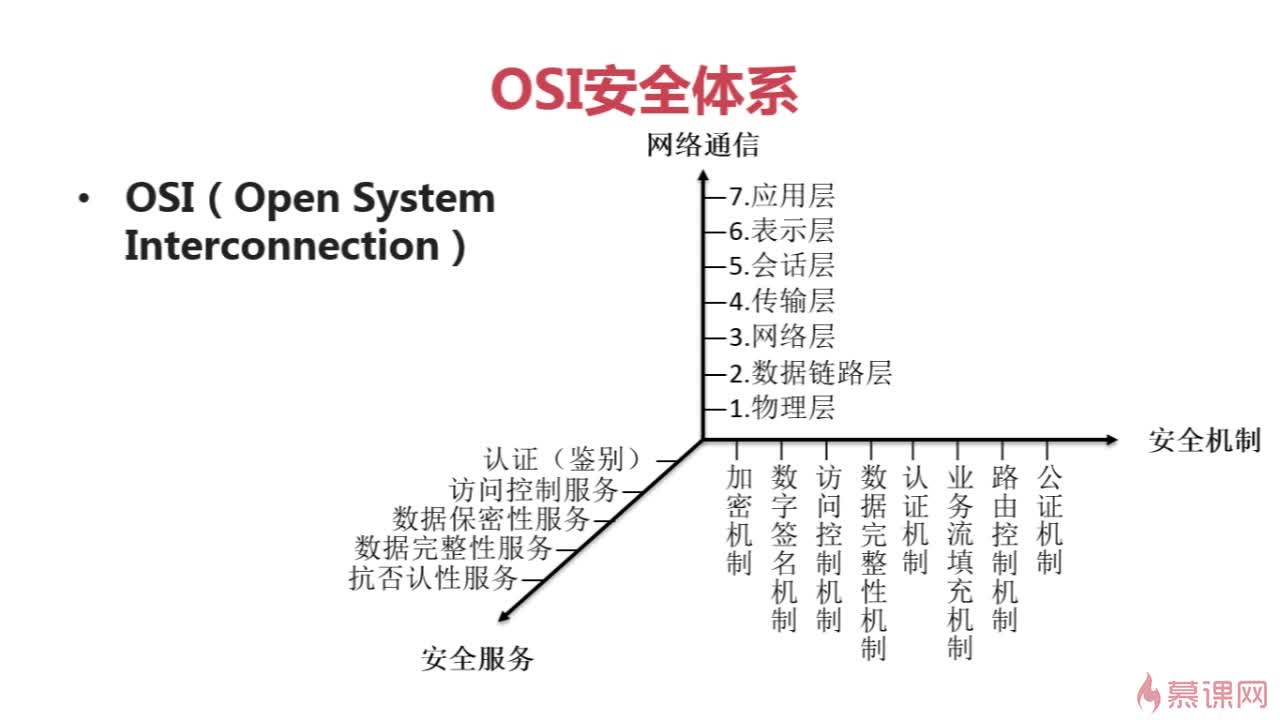
**网络接口层(以太网、令牌环网、FDDI、IEEE802.3等)**

**网络层(IP/ICMP/IGMP)**

**传输层(TCP传输控制协议和UDP用户数据报协议)**

**应用层(Telnet/FTP/HTTP/DNS/SNMP/SMTP等)**

**各层协议完成不同功能**

****

**4. Java实现Base64加密——JAVA安全组成**

**JCA（Java Cryptography Architectrue）JAVA加密体系。**

**JCE（Java Cryptography Extension）JAVA加密拓展：DES 、AES 、RSA算法通过JCE提供。**

**JSSE（Java Secure Socket Extesion）JAVA套接字安全拓展：提供基于SSL的加密功能，主要用于网络传输。**

**JAAS（Java Authentication and Authentication Service）JAVA鉴别与安全服务。**

**Ps：使用第三方的加密算法：**

**-第一种在JDK的根目录下的jre7--lib--security--java.security中的：**

**security.provider.1=sun.security.provider.Sun**

**security.provider.2=sun.security.rsa.SunRsaSign**

**security.provider.3=sun.security.ec.SunEC**

**...**

**是引进其他公司的加密包的，属于JAVA加密拓展。可以自己拓展算法实现，例如：security.provider.11=com.imooc.Provider //属于自己添加的拓展。**

**-第二种方法使用第三方的加密算法可以通过Security中的addProvider或者insertProviderAt的方法初始化在上列加密算法实现类，像是反射机制动态生成。**

**Java的安全组成**

**JCA(java加密体系结构)**

**JCE(java加密扩展包)，DES、AES、RSA算法通过JCE提供**

**JSSE(java安全套接字扩展包)：提供基于SSL的加密功能，主要用于网络传输**

**JAAS(java鉴别与安全服务)：提供在Java这个平台上的身份认证功能**

**相关java包、类**

**java.security(信息摘要)**

**javax.crypto(安全信息摘要，消息认证（鉴别）码)**

**java.net.ssl(安全套接字)主要用到的类：HttpsURLConnection、SSLContext**

**(第三方扩展)**

**Bouncy Castle是一种用于Java平台的开放源码的轻量级密码术包.它支持大量的密码术算法,并提供JCE的实现.**

**Commons Codec(apache提供的包)**

**支持Base64、二进制、十六进制、字符集编码、Url编码/解密.**

**JDK内部也提供了Base64算法.但通常还是使用Commons Codec下提供的Base64算法.**

**使用第三方提供的加密算法可以在JDK目录下的资源文件中添加,或者在程序中调用Security的addProvider()或者insertProviderAt()方法来添加,这样可以为JDK添加新的算法.如果没有此操作,可以直接导入第三方jar包来使用即可.**

**JAVA安全相关的包.**

**(JDK提供)**

**java.security(消息摘要):为安全框架提供类和接口.(简单加解密)**

**javax.crypto:安全消息摘要,消息认证码.(严瑾复杂加解密)**

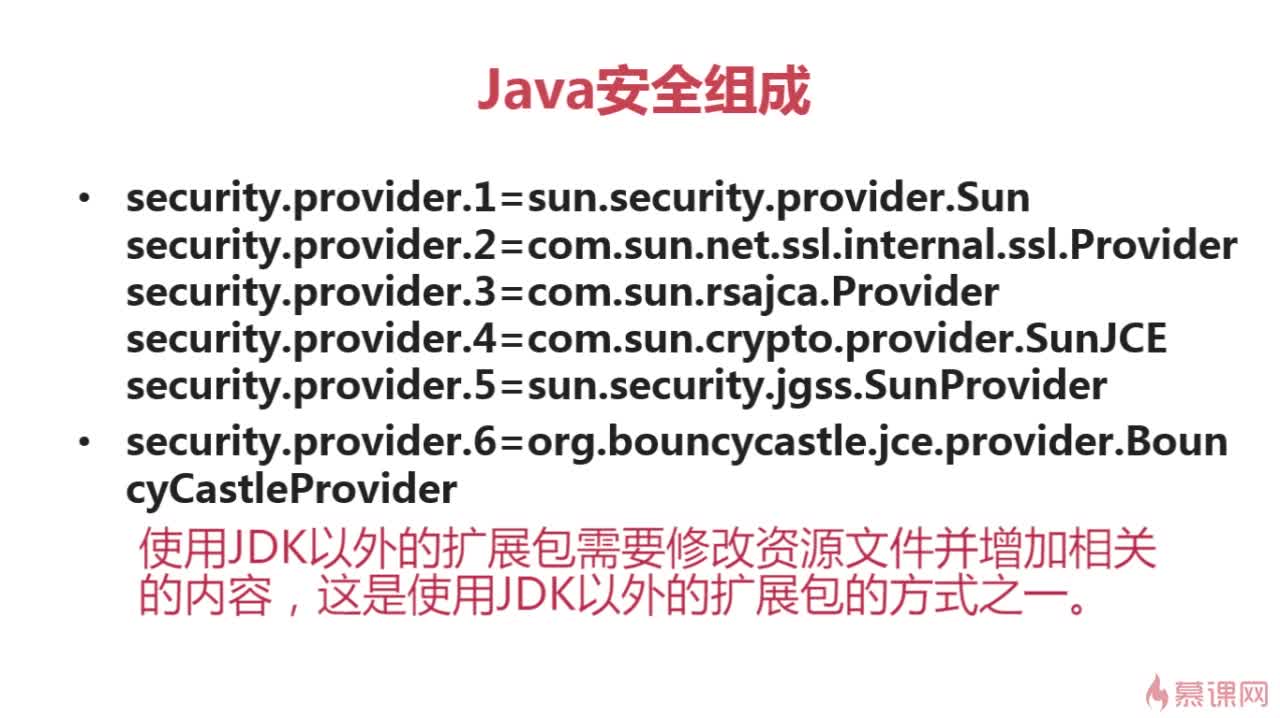
**java.net.ssl:安全套接字包,主要作用是与网络传输数据相关的一些加解密操作.**

**第三方Java扩展：**

**Bouncy Castle---两种支持方案：1.配置 2.调用**

**security.provider.1=sun.security.provider.Sun**

****

****

**5. bouncyCastle实现Base64编码方式**

**使用Base64类的encode()静态方法,将指定内容通过Base64编码.参数是字节数组,返回字节数组.**

**使用Base64类的decode()静态方法,将经过Base64编码的内容进行解码,参数是字节数组或字符串,返回字节数组.**

**编码(加密):**

**Byte[] b1=Base64.encode("imooc".getBytes());**

**解码(解密):**

**Byte[] b2=Base64.decode(b1);**

**commonsCodec实现Base64编码方式**

**使用Base64类的encodeBase64()静态方法,将指定内容通过base64算法进行编码,参数是字节数组,返回字节数组.**

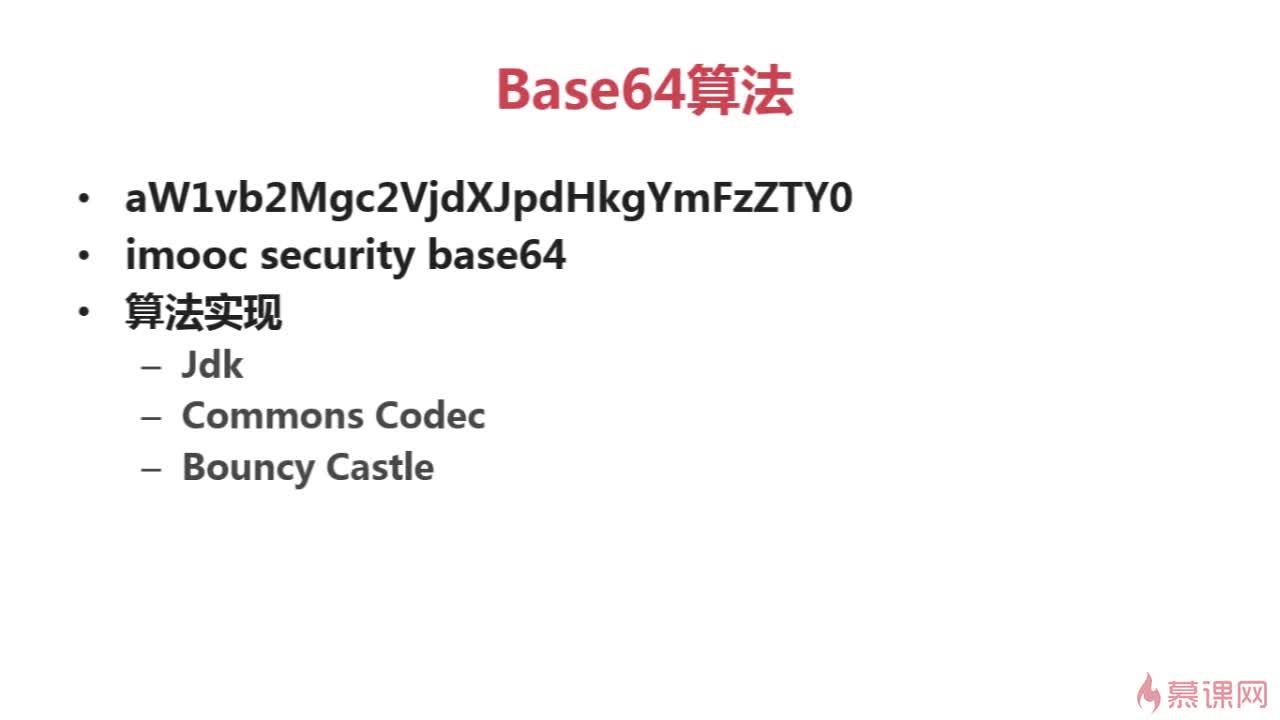
**使用Base64类的decodeBase64()静态方法,将经过base64算法加密的内容进行解码,参数是字节数组,返回字节数组.**

**编码(加密):**

**Byte[] b1=Base64.encodeBase64("imooc".getBytes());**

**解码(解密):**

**Byte[] b2=Base64.decodeBase64(b);**

****

**6. Java实现Base64加密——Base64算法应用场景和由来**

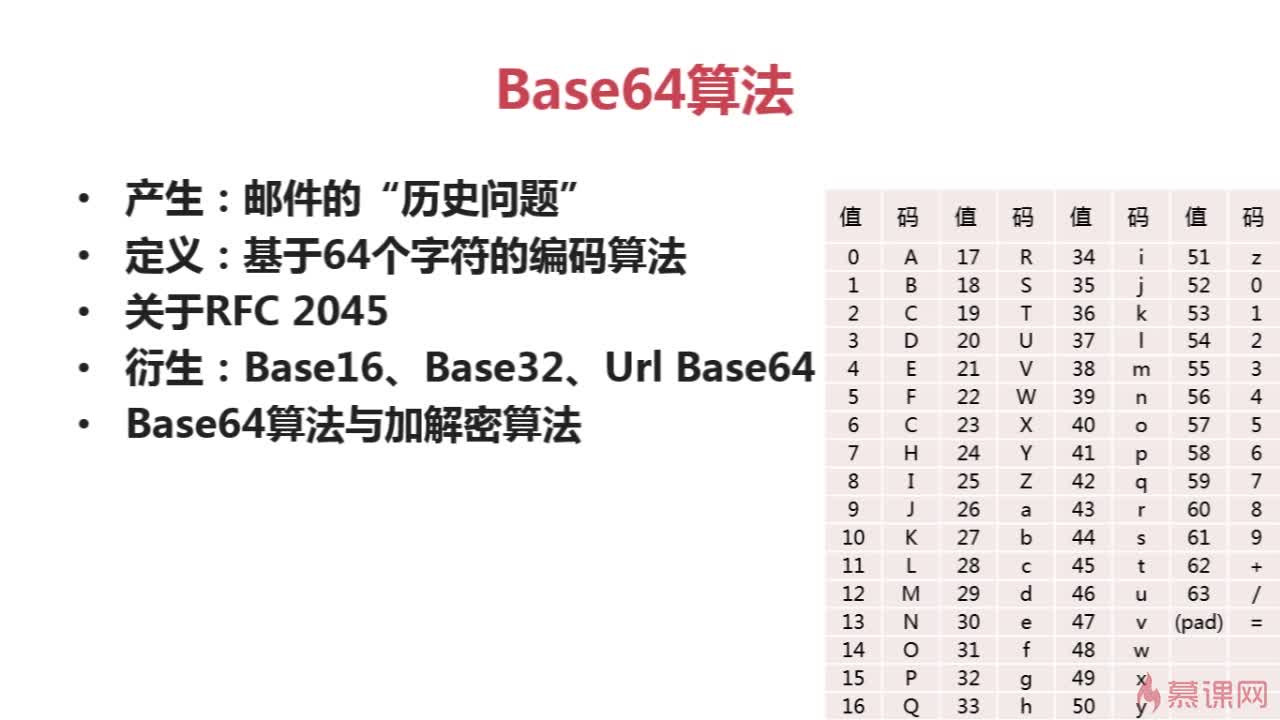
**-Base64算法的应用场景：E-Mail；密钥；证书文件。**

**-产生的历史原因：邮件的“历史问题”；原因是邮件在传输一些非ASCLL码的值，在一些不同的网关或者网络会出现一些问题。**

**-定义：基于64个字符的编码算法。**

**-RFC 2045关于Base算法的基本规范。**

**-衍生：Base16、Base32、Url Base64。**

****