# 1、包含的内容

（1）安全和密码

（2）常用安全体系介绍

（3）密码分类及Java的安全组成

（4）JDK相关包及第三方扩展

（5）Base64算法介绍

# 2、如何学习

（1）掌握用法

（2）深入理解

（3）不断实践

（4）反复总结

（5）再次深入理解与实践

# 3、密码常用术语

（1）明文：待加密的信息

（2）密文：经过加密后的明文

（3）加密：明文转为密文的过程。

（4）密钥：密钥是一种参数，它是在明文转换为密文或将密文转换为明文的算法中输入的参数。密钥分为对称密钥与非对称密钥。

（4）加密算法：明文转为密文的转换算法

（5）加密密钥：通过加密算法进行加密操作用的密钥。

（6）解密：将密文转为明文的过程。

（7）解密算法：密文转为明文的算法

（8）解密密钥：通过解密算法进行解密操作用的秘钥。

（9）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者秘钥的过程。

（10）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。（对密文具有破坏作用）

（11）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。

（12）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。

（13）密码协议：也称安全协议，指以密码学以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。

（14）密码系统：用于加密、解密的系统。

（15）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

# 4、密码分类

## （1）按时间进行分类

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。

## （2）保密内容算法

受限制算法：算法的保密性基于保持算法的秘密

基于秘钥算法：算法的保密性基于对密钥的保密

## （3）密码体制（重点）

对称密码：指加密密钥和解密密钥相同。

非对称密码：指加密密钥与解密密钥不同，密钥分为公钥、私钥。

对称密码算法：指应用于对称密码的加密、解密算法。

非对称密码算法：指对应于非对称密码的加密、解密算法。

## （4）明文处理方法

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：也称为序列密码，指加密时每次加密一位或者一个字节明文。

# 5、散列函数

概念：散列函数用来验证数据的完整性。

特点：长度不受限制、哈希值容易计算、散列运算过程不可逆。

## 散列函数相关的算法

（1）消息摘要算法MD5等

（2）SHA：安全散列算法

（3）MAC：消息认证码算法

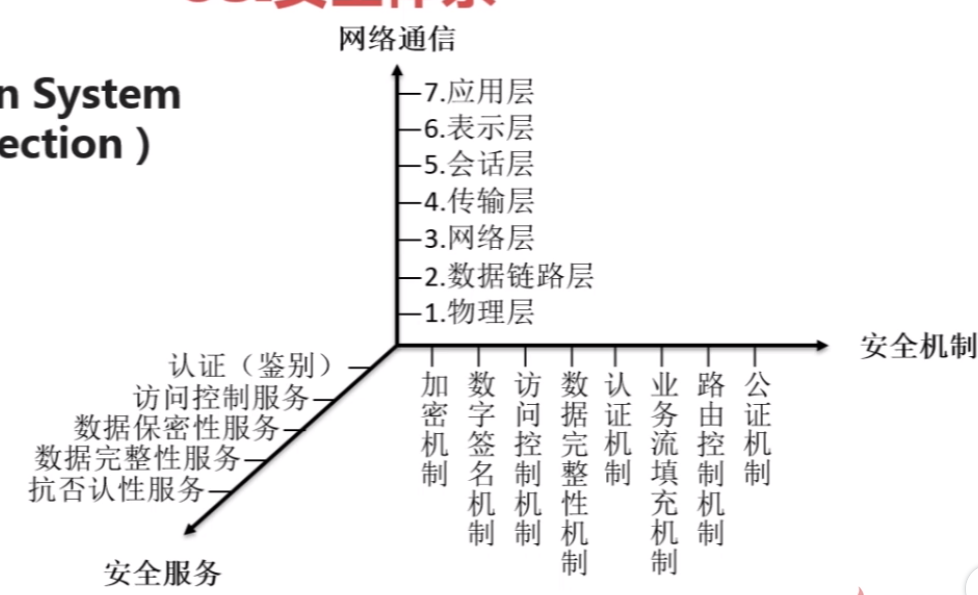
# 6、数字签名

主要是针对以数字的形式存储的消息进行的处理。数字签名产生以操作者身份信息的编码。

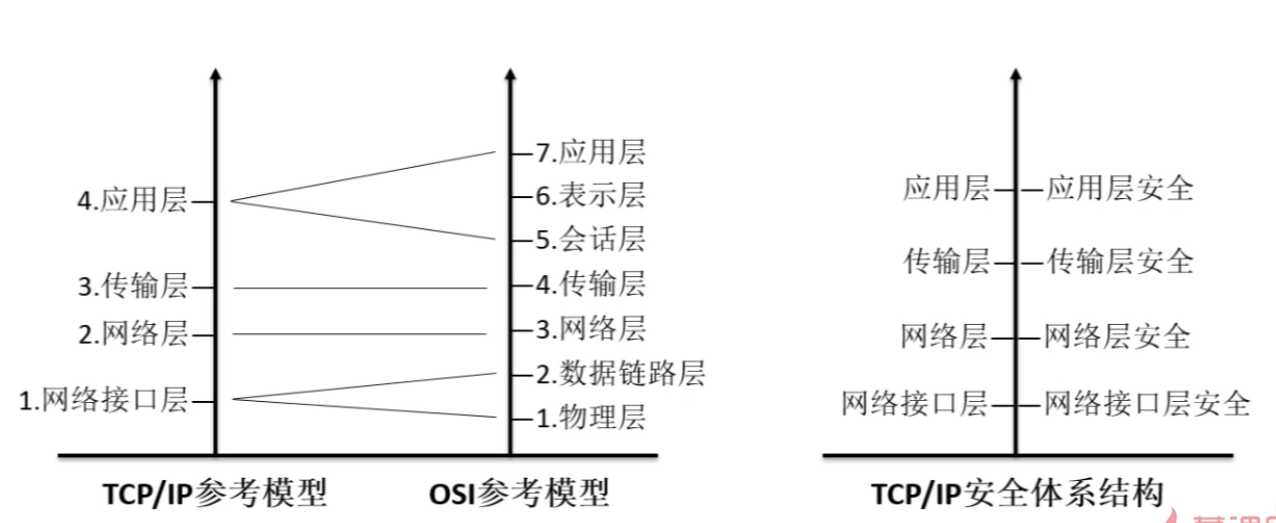
# 7、OSI安全体系

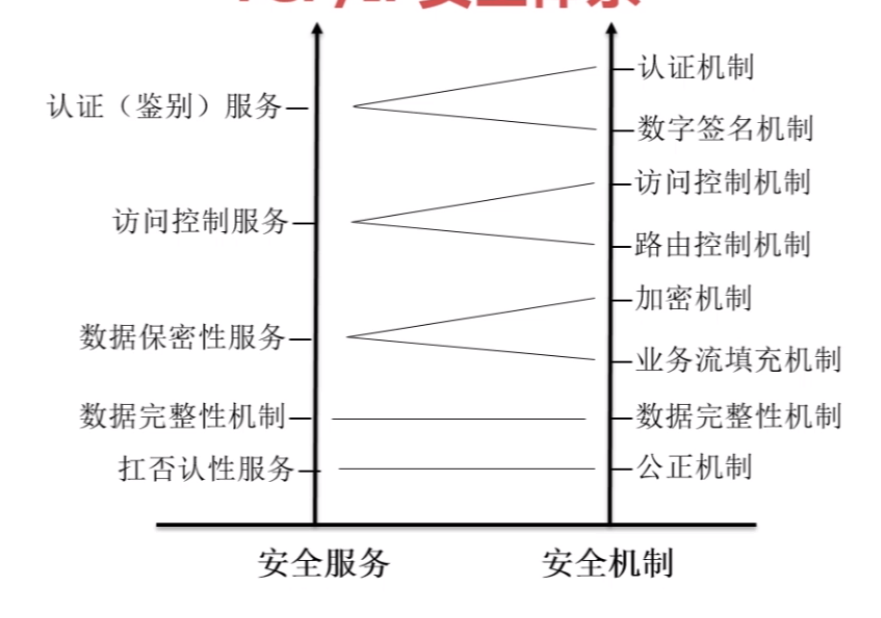
OSI（open System Interconnection）：开放系统交互

网络通信、安全机制、安全服务的关系：



# 8、TCP/IP安全体系





# 9、Java安全组成

JCA（java cryptography Architecture）

JCE（Java cryptography Extension）

JSSE（java Secure Socket Extension）

JAAS（Java Authentication and Authentication Service）

DES AES RSA算法通过JCE提供，JCE主要是在jdk的一些包中。JSSE提供基于SSL的加密功能，主要用于网络传输。可以通过调用Security类中的addProvider或insertProviderAt方法来动态注册提供程序。

## 相关java包、类

（1）java.security：这个包用于消息摘要

（2）javax.crypto：安全消息摘要，消息认证（鉴别）码，用于加解密。

（3）java.net.ssl：安全套接字包

HttpsURLConnection SSLContext

## 第三方扩展包

### 1、Bouncy Castle

两种支持方案：配置、调用

maven dependency:

<dependency>

<groupId>org.bouncycastle</groupId>

<artifactId>bcprov-jdk15on</artifactId>

<version>1.59</version>

</dependency>

### 2、Commons Codec

Apache

Base64 二进制 十六进制、字符集编码

URL编码、解码

说明：这个用的多，不要用jdk中的。

maven dependency:

<dependency>

<groupId>commons-codec</groupId>

<artifactId>commons-codec</artifactId>

<version>1.11</version>

</dependency>

## 应用

（1）Base64算法

（2）消息摘要算法

（3）对称加密算法

（4）非对称加密算法

（5）数字签名算法

（6）数字证书

（7）安全协议

### Base64算法

定义：基于64个字符的编码算法

衍生：base16 base32 Url Base64

算法实现：

（1）JDK

（2）Commons Codec

（3）Bouncy Castle

### 代码示例

（1）JDK原生的Base64加解密

public static void JDKCrypto(){

String encodeStr = Base64.getEncoder().encodeToString(str.getBytes());

System.out.println(encodeStr);

byte[] decodeStr = Base64.getDecoder().decode(encodeStr);

String str = new String(decodeStr);

System.out.println(str);

}

（2）使用commonsCodes进行加解密

public static void commonsCodesBase64(){

byte[] encodeByte = org.apache.commons.codec.binary.Base64.encodeBase64(str.getBytes());

System.out.println(encodeByte);

byte[] decodeByte = org.apache.commons.codec.binary.Base64.decodeBase64(encodeByte);

System.out.println(new String(decodeByte));

}

（3）使用BoundcyCastle进行加解密

public static void BoundcyCastleBase64(){

byte[] encodeByte = org.bouncycastle.util.encoders.Base64.encode(str.getBytes());

System.out.println(encodeByte);

byte[] decodeByte = org.bouncycastle.util.encoders.Base64.decode(encodeByte);

System.out.println(new String(decodeByte));

}

### 应用场景

email、密钥、证书文件