1.WhyLearnNetworkSecurity

信源 ---(信道)---> 信宿

三网合一：计算机网络、电信网络、有线电视网

计算机系统进行各种数据处理，组成资源子网（外圈的）
通信链路和网络节点提供通信功能，组成通信子网 （内圈的）

Enigma

转轮机

R是个反射，关键在反射，它使得加密和解密的过程是一样的

看ppt，键盘按下一个键之后，滚轮N先向前移动一格(N0->N1)，然后再读取转换后的字母

解密的时候，好像滚轮初始状态要和加密时一样

2.1 Cryptography

密码算法/密码（cipher，c）

c = E(m), m = D(c)

密钥（key, k）

明文有时以m（message）表示，有时以p (plain text)表示

密码学：

- 密码编码学
- 密码分析学

置换：重新排列（位置）

代换：元素映射为另外一个元素

传统加密：对称加密，发送方和接收方密钥相同

现代加密：非对称加密，发送方和接收方密钥不同

分组密码/块密码：每次处理一个输入分组，相应地输出一个输出分组

流密码/序列密码 ： 连续地处理输入元素，每次输出一个元素

古典密码

数据安全性依赖于算法保密性。

代换技术

单表代换密码

Caesar 密码 （凯撒）： c=(m+3) Mod 26 。 (固定用往后的第三个字母代换)

多表代换密码

Playfair密码：

字母矩阵，把密钥扔到矩阵最前面去。见2.1ppt P35。一次加密两个字母（**分组密码**）。若字母对中两字母相同，要有填充符(x)！同行右，同列下，（右下），否则每个字母固定行，使用另外一个字母的列。

Hill密码：

n个线性方程（最后%26），矩乘

$$\begin{pmatrix} c_1 \\ c_2 \\ c_3 \end{pmatrix} = \begin{pmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \mod 26 \quad (1)$$

优点是完全隐蔽了单字母的频率特性

Vigenere：

与Caesar 密码不同的是，它是有密钥的，由密钥决定每个明文字母往右移多少位(A=0, Z=25)（其实就是字母转数字相加再模26），密钥自动循环延长至和明文长度一样。

一次一密是牢不可破的，Vernam密码： 直接明文和密钥转 0~25 相加模26 得到密文

置换技术

比如按行写入，按列读出

破译

穷举法、频率分析法（分析字母出现频率以及其他规律/统计特性）

对称密码算法

密钥使用秘密信道分配，安全性不在于算法保密性而在于密钥保密性。

S-DES (简化DES)

一堆置换、代换、异或的操作，花里胡哨

见ppt P71，注意每个函数是什么。书桌右上柜子里有个练习。（练习中有个swap的问题，注意）（注意题目逻辑，填入表格的最后一步才是进行对应的函数操作）

记住S0和S1函数怎么计算，第1、 4位作为二进制数决定S盒的行 ，第2、 3位作为二进制数决定S盒的列，输出一个两位二进制数.

LS-1表示前5位和后5位分别循环左移1位。
LS-2表示前5位和后5位分别循环左移2位

SW将输入的左4位和右4位交换，交换过后直接输出两个四位二进制数（不用再管后续的R4, L4字眼了）

Feistel密码结构

流密码

Vigenere密码和Verman密码都是流密码

分组密码

绝大多数基于网络的对称密码应用使用的都是分组密码（将一个明文组作为整体加密，分组大小诸如64 bit, 128 bit）

现在使用的**对称**分组密码算法都基于Feistel分组密码结构的

Feistel建议使用乘积密码(依次使用两个或以上的基本密码)的概念来逼近简单代换密码。建议交替使用代换和置换 。

扩散：是指使**明文的统计特征**消散在**密文**中，让每个明文数字尽可能地影响多个密文数字

混淆：是尽可能地使密文和加密**密钥**间的统计关系更复杂，以挫败推导出密钥的企图

拆分、迭代。这个好像讲得比较简略

feistel密码结构的重要参数：

- 分组长度和密钥长度
- 迭代轮数
- 子密钥产生算法
- 轮函数

DES

采用分组加密。也是多轮迭代，内部有部分和Feistel结构相同

常用的对称密码

3DES，Blowfish，RC5，AES...

AES不是Feistel结构！

2.2 Cryptography

非对称密码算法

对称密钥密码系统的缺陷：
• 密钥必须经过安全的信道分配
• 无法用于数字签名（验证）
• 密钥管理复杂，密钥的数量： O(n2)

公钥密码是密码学历史上唯一的一次真正的革命。公钥密码是基于**数学函数**而不是代换和置换

KUa：用户a的公钥 （public）

KRa：用户a的私钥 （private）

$E_{KUa}[P]$： 用KUa对明文P进行加密

公钥：加密，验证签名

私钥：解密，签名

签名：发送信息，发送端对私钥加密来签名，接收端用对方的公钥解密来验证确实是发送端发过来的

会话密钥：Ks，就是对称密钥里面的，用同一个密钥Ks来加密解密

同时使用数字签名和加密：见ppt P12

公钥密码的数学原理：陷门单向函数。

1. 单向函数是求逆困难的函数
2. 给定y, 计算x使y=f(x)是困难的
3. 存在δ，已知δ时对给定的任何y，若相应的x存在，则计算x使y=f(x)是容易的

单向函数：求逆困难的函数，满足1,2点

条件3是陷门性，δ是陷门信息（相当于提供了逆函数）

公钥密码系统除了加密解密、数字签名之外，还有一个用途：

- 密钥交换：双方协商会话密钥，用于对称密钥数据加密

公钥密码算法	加密解密	数字签名	密钥交换
RSA	Y	Y	Y
Diffie-Hellman	N	N	Y
DSA	N	Y	N

公钥密码比传统密码更安全 。 **这句话是错误的！**(安全性依赖于密钥长度和破译密文所需要的计算量。不能简单地说传统密码和公钥密码哪个更安全 ）

公钥密码是一种通用方法，传统密码已经过时 。 **这句话是错误的！**（公钥密码需要大量计算，仅限于**密钥管理和签名**这类应用中，所以基本不太可能取代传统密码 ）

传统密码中与密钥分配中心的握手是一件异常麻烦的事情，而公钥密码实现密钥分配则是非常简单的 。 **这句话是错误的！**（公钥密码实现密钥分配也需要某种形式的协议 ，也很麻烦)

P18 有个对对称密码和公钥密码的总结

RSA算法

RSA体制是一种分组密码，其明文和密文都是0~n-1之间的整数！

回顾：欧拉函数φ(n)： n是正整数,φ(n) 是比n小且与n 互素的正整数个数（φ(6) = 2)

欧拉定理： m,n互素，则$m^{\phi(n)} \equiv 1(mod\ n)$

RSA算法见P27 28。

$$\begin{gathered} 大素数p,q. \quad 令n=pq公开n \\ 取一个与\phi(n)=(p-1)(q-1)互质的小整数e \\ 寻找d(d<\phi(n))使得de \equiv 1 \mod \phi(n) \end{gathered} \tag{2}$$

得到公钥KU={e, n}，私钥KR={d,n}，然后加密解密如下：

$$\begin{aligned} C &= M^e \mod n \\ M &= C^d \mod n \end{aligned} \tag{3}$$

($C = M^e \mod n = C^{de} \mod n = C^{k\phi(n)+1} \mod n = C$)

有一种计时攻击：通过记录计算机解密消息所用的时间来确定私钥

RSA抗穷举攻击的方法是使用大密钥空间，它比DES慢了100~1000倍。

DH密钥交换算法 (Diffie-Hellman)

$ind_{a,p}(b)$ ： b 的以a为底，模p的离散对数或指数，即$ind = \log_a b \mod p, \quad b = a^{ind} \mod p$

求离散对数是比较困难的（BSGS $O(\sqrt{p})$求解）

DH算法过程见P42，使用原根，两个人贡献一部分信息凑出会话密钥：

$$
\begin{gathered}
\text{公开：}q\text{素数，}a\text{是}q\text{的原根} \\
A:\text{私有}X_A(<q)\text{，公开}Y_A = a^{X_A} \mod q \\
B:\text{私有}X_B(<q)\text{，公开}Y_B = a^{X_B} \mod q \\
A\text{计算会话密钥}K = Y_B^{X_A} \mod q \\
B\text{计算会话密钥}K = Y_A^{X_B} \mod q \\
\text{会话密钥}K\text{都等于}a^{X_A X_B} \mod q
\end{gathered} \tag{4}
$$

密钥分配

密钥分配中心KDC

情况一：传统的对称密码分配

每个用户与密钥分配中心KDC共享唯一的一个主密钥。（A有一个除了自己只有KDC知道的主密钥Ka）。A可以向KDC发送请求，获得A与B建立连接所需要的会话密钥Ks，以及用Kb加密后的会话密钥Ks和A标识符，A将Kb加密后的信息发给B。这样会话密钥就安全地发给了A和B。

在网络规模很大的时候，密钥的分配功能不限定在单个的KDC上面，而是使用**层次式**的KDC 。

情况二：非对称密码中的公钥分配

公开发布、公开可访问目录、公钥授权、公钥证书

情况三：公钥密码用于传统密码体制的密钥分配

由于公钥密码速度较慢，几乎没有用户愿意在通信中完全使用公钥密码，因此公钥密码更适合作为传统密码中实现密钥分配的一种手段。

3.Authentication

认证=比较

消息认证

消息认证就是验证所收到的消息**确实是来自真正的发送方**且**未被修改**的消息。（数字签名是一种认证技术，他还可以用来抗击发送方否认）

认证函数： 产生认证符的函数。分为三类：

- 消息加密：整个消息的密文作为认证符。
- 消息认证码MAC： MAC是消息和密钥的公开函数，它产生定长的值，该值作为认证符。
- Hash函数：它是将任意长的消息映射为定长的hash值的公开函数，以该hash值作为认证符。

消息加密

直接对消息进行加密来认证。

帧校验序列FCS或者校验和来确认消息是真实的。附加在消息后面

必须先计算FCS，再加密！（防止伪造——构造信息以混淆）

对称加密可以提供保密性、认证（这条消息来自A），但是不能提供数字签名（接收方伪造信息、发送方否认信息。数字签名表示我确实曾经写过这个信息！）

而公钥加密都可以提供：保密性、认证、数字签名

可知：共享密钥就不能提供数字签名（因为另一方可以进行篡改），所以MAC也不能提供数字签名

消息认证码MAC

算一个MAC来认证。

MAC（消息认证码）是消息和密钥的函数： $MAC = C_k(M)$

C是MAC函数，K是双方的共享密钥

MAC函数与加密类似，区别就是MAC算法**不要求可逆性** 。

- A和B共享密码K
- A向B发送消息时， A计算MAC，将其附加在消息后面，一起发送给接收方 。接收方也算一个MAC，比较两个MAC

密文有关的认证：先加密，再对密文算MAC。

在实际中，一般将认证和保密性相结合使用。

画图：认证+保密

Hash函数

算一个hash来认证。单向的hash函数是消息认证码的一种变形 。但是**hash不使用密钥**，hash具有错误检测能力。hash码也称为消息摘要。

图中的|| 表示拼接，M || Hash(M)

Hash: $H(M)$

传送时，只对hash值加密就相当于是MAC， $MAC = E_K[H(M)]$

还有用公私钥对hash码进行加密的...

$E_K[M||E_{KRa}[H(M)]]$ 这个使用了hash、对称加密、公钥加密，能够同时提供认证、保密性、数字签名（敌营前面的方法）

hash函数的要求：

单向性（求逆困难），抗弱碰撞性（给m，很难找到m'使得$H(m') = H(m)$），抗强碰撞性（很难找到x,y使得$H(x) = H(y)$）

也就是说对于hash函数来说，抗弱碰撞性更强

hash里面重复使用了压缩函数f（比如取模），设计安全hash函数可以归纳为设计具有抗碰撞能力的压缩函数问题，并且该压缩函数的输入是定长的

HASH: CV0=IV是初始值，CVi = f(CVi-1, Yi-1)， 最终的H(M)就为CVL。 f就是压缩函数。

压缩函数抗碰撞，那么hash也抗碰撞

MD5 (Message Digest)

MD5被王小云攻破了（碰撞相关）

MD5的**输入**是任意长度的消息，对输入按照512位的**分组**为单位进行处理，算法的**输出**是128位的消息摘要

- Step1：增加填充位 (第一位填1，后面填0)
- Step2：填充长度。（填充前的长度，小端，64位二进制数，若长度超过2^{64}，模一下）（填充了填充位与长度之后，消息为512的倍数）
- Step3：初始化MD 缓存 （就是计算过程中的那个128位，要赋初始值）
- Step4：以512位的分组(16个字)位单位处理消息 （这里面是有多轮运算的）
- Step5：输出

也是一种特殊的hash，重复使用某一个压缩函数f，只不过这里f是MD5自己的压缩函数。见P47页流程

MD5相比MD4多一轮运算，多使用一个基本逻辑函数

SHA (Secure Hash Algorithm)

SHA算法建立在MD4之上，基本框架与MD4类似

SHA-1:

输入: 长度小于2^{64}位

输出: 160位的消息摘要（5 * 32)

分组: 以512位为单位进行分组处理

算法步骤与MD-5一样都是5步，压缩函数变为了SHA-1压缩函数。

RIPEMD-160

输入: 长度小于2^{64}位

输出: 160位的消息摘要（5 * 32)

分组: 以512位为单位进行分组处理

也是那五个步骤

160表示步数，这个步数比MD5和SHA都要长。

三个算法都不容易受到弱碰撞性的攻击。

MD5由于消息摘要短，易于受到强碰撞性的攻击。

上面三个算法，密码分析（就是破译）一个比一个难。

但是MD5执行速度是最快的。

只有SHA-1采用高位在前的结构（大端序）

数字签名算法DSS

认证和数字签名的区别：

消息认证可以保证通信双方不受**第三方**的攻击，但是它不能处理通信**双方自身**发生的攻击 （之前说的两种情况，接收方伪造信息、发送方否认信息）。数字签名相当于手写签名，验证签名者、签名日期等。

两类：

直接数字签名：只涉及通信双方 。弱点在于方法的有效性依赖于发送方私钥的安全性

仲裁数字签名：从发送方X到接收方Y的每条已签名的消息都先发给仲裁者A， A对消息及其签名进行检查以验证消息源及其内容，然后给消息加上日期，并发给Y，同时指明该消息已通过仲裁者的检验 。

DSS只能提供数字签名功能，（给出了一种新的数字签名方法，即数字签名算法DSA ）

大概是：根据接收到的信息算出一个r（签名由s,r两部分组成），拿算出的r与信息中的r比较，相等则签名有效

身份认证

确认行为参与者（主体或客体，通常称为用户）身份的解决方法 。 比如居民身份证、银行卡及密码 等

互联网世界中一切信息（包括各类对象的身份信息）都是用一组特定的数据来表示的 ，因此身份认证的过程也是针对这组特定的数据进行的 。需要大量使用消息认证技术

网站身份认证技术

超文本传输协议HTTP （ Hyper Text Tansfer Protocol ） 是基于TCP/IP的应用层协议

- 可以通过HTTP协议传输任意类型的数据对象
- HTTP是面向一次连接的无状态网络协议 （只要一次建立连接，之后就是请求-响应即可）

最简单的网站用户认证： HTTP的Basic认证。改进解决Basic认证的问题： 基于表单的身份认证 。增强认证：手机短信口令等等。

Basic认证：用户身份凭证：账号+静态口令 。**每次**发出HTTP请求时，把用户身份凭证的明文发送到服务器端，服务器与存储在服务器端的用户凭证进行比较，认证用户身份。

每次都要传递账号和口令，所以安全性低

基于表单的身份认证 ：改进解决Basic认证的问题

session机制：一个Session包括特定的客户端、特定的服务器端以及特定的操作时间段

当某个Session首次启用时，服务器会产生一个唯一的标识符发送到客户端，客户端收到后会存储下来，后续请求都包括这个session，用于身份认证

引入Challenge/Response机制，避免口令明文通过网路传输，并且能避免重放攻击

使用传输层SSL协议传输HTTP请求 （https）（在传输层对网络连接进行加密 ）

不安全的：

把账号口令以Cookie机制存放在浏览器端

把账号口令进行加密后Ek[U||P)以Cookie机制存放在浏览器客户端 （带来的问题：口令泄露，重放攻击 ）

数字证书：一个经证书授权中心数字签名的文件，最简单的证书包含拥有者的公钥等

4.CIA-IPsec-IKE

一个树形图：从上往下，每层依次是：安全目标、安全服务、安全机制、 (算法、实现)

安全目标：CIA

Confidentiality + Integrity + Availability

保密性 + 完整性 + 可用性 (比如防DOS)

安全服务

•Authentication： 认证服务
•Confidentiality： 保密服务
•Integrity： 数据完整性保护
•Access Control： 访问控制服务
•Non-repudiation： 抗抵赖服务
•Availability： 可用性服务

安全机制

分为普通安全机制和特定安全机制

普通安全机制：不属于任何协议层或者安全服务的安全机制

特定安全机制：在特定的协议层实现的安全机制

八种特定的安全机制（了解一下吧）

1. 加密机制
2. 数字签名机制
3. 访问控制机制
4. 数据完整性机制
5. 认证机制
6. 业务流填充机制
7. 路由控制机制
8. 公证机制

安全性攻击：分为主动攻击和被动攻击。被动攻击包括窃听和监测。

网络层安全协议: IPsec !

IPsec: IP + 安全

概述

应用层有：S/MIME, PGP, SET

传输层有：SSL, TSL

网络层有：IPSsec

IPsec保障了IP级的安全性，包括：

- 认证：确保来源，未被篡改
- 保密：加密传输，防止窃听
- 密钥管理

IP协议是无状态无连接的

IP packet: 由20 bytes的报头和来自传输层的数据组成

IPSec的原理在于可以在IP层加密和/或认证所有流量 （在网络层实现端到端的安全性）

IPSec的应用：虚拟专用网 VPN， 远程安全访问公司网络

IPSec的实施：主机（与操作系统集成）、防火墙（在防火墙上实施时，IPSec在网络层和数据链路层之间。为内部所有的应用提供安全服务 ）、路由器（虚拟专用网VPN）

IPSec体系结构

IPSec文档 -> IPSec服务 -> 安全关联SA -> 传输模式 隧道模式

SA：Security Association 安全关联

IPSec服务：AH协议、ESP（只加密）、ESP（加密并认证）

安全关联SA：IPsec通信双方之间对某些要素的一种协商，一组安全信息参数集合 （协议、操作模式、密码算法、认证算法、密钥、密钥生存期等 ）

关联是发送方和接收方之间的**单向关系**，该关联为双方的通信提供了安全服务。 （所以双方安全交换需要两个SA）

安全服务可由AH或者ESP提供，但不能两者都提供。

一个安全关联SA由三个参数唯一确定：

- 安全参数索引SPI
- IP目的地址IPDA
- 安全协议标识

在任何IPSec实现中，都有一个安全关联数据库SADB，它定义了与每个SA相关联的参数。

AH和ESP均支持两种模式：传输模式和隧道模式

传输模式主要**为上层协议提供保护**，同时增加了IP包载荷的保护

隧道模式**对整个IP包提供保护**

IPSec认证头：AH (Authentication Header)

与认证有关

认证头AH支持数据完整性和IP包的认证

一个包长这样：IP头 | AH | TCP/UDP头 | 数据

网原编程作业里面遇到的Next Header、 Payload Length这些都是AH的组成

重放攻击：攻击者在得到一个经过认证的包后，又将其传送到目的站点的行为

重复接收经过认证的IP包可能会以某种方式中断服务或者产生不可预料的后果，序列号域就可以防止重放攻击 （使用计数器，计序列号）

传输模式：

- 通常以端到端方式实现（在主机上实现）
- 不修改IP头，只添加AH头

隧道模式：

- 通常在防火墙或路由器上实现
- 把整个IP包作为数据，**增加一个新的IP头**、AH头

IPSec：封装安全载荷ESP

与保密有关，可保密也可认证

封装安全载荷ESP提供保密性服务包括报文内容保密和流量限制保密， ESP还可以提供和AH同样的认证服务

传输模式：加密和认证(可选)直接由两个主机提供

隧道模式：ESP用于加密整个IP包

可以见P79左右的两张图

IPSec：安全关联组合！！每年都考

单个SA可以实现AH或者ESP， 但是**不能两者都实现**

安全关联组合（安全关联束）是指提供特定的IPSec服务集所需的一个SA序列， SA可通过两种方式组合成束：

- 传输邻接：在不使用隧道的情况下，对一个IP包使用多个安全协议；组合AH和ESP的方法仅允许一级组合
- 隧道迭代：指通过IP隧道应用多层安全协议；由于每个隧道可以在路径上的不同IPSec节点起始和结束，因此该方法允许多层嵌套。

图见P84 AH和ESP的典型组合，比如

传输模式(Transport)： `[IP1][AH/ESP][upper]`，`[IP1][AH][ESP][upper]`

隧道模式(Tunnel)：`[IP2][AH/ESP][IP1][upper]`

网络层安全协议： IKE

为IPSec管理密钥， Internet Key Exchange（互联网密钥交换协议）

SADB & SPDB

IPSec的安全通信之前必须建立安全关联SA， SA存储在本地的安全关联数据库（SADB)中， 决定了进行何种安全操作。

安全策略数据库SPDB是把IP信息流与SA联系起来的手段，决定了对流入和流出的哪些数据包进行安全操作

IKE

IKE(Internet Key Exchange)协议为IPSec提供了自动协商交换密钥、建立安全关联SA的服务，简化了IPSec的使用和管理。IKE协议解决了在不安全的网络环境中安全地建立或更新共享密钥的问题

目前IKE协议只在IPSec协议中得到了应用

精髓：永远不在不安全的网络上交换密钥。即使黑客截获了双方用于计算密钥的所有交换数据，也不足以计算出真正的密钥 。因为使用了**DH密钥交换算法**，具备完善的前向安全性（一个密钥被破解，不影响其他密钥的安全性）

IKE不但可自动地为参与通信的实体协商安全关联SA，还可以维护安全关联数据库SADB

IKE报文格式

继承自ISAKMP，可以在任何传输层协议（TCP、UDP）或IP层上实现

13种载荷

IKE体系结构

IKE的两个阶段：（建立通道+传输？）

1. 协商创建一个通信信道（**IKE SA**），并对信道进行验证
2. 使用已建立的IKE SA建立**IPSec SA**

简而言之，**第一阶段：协商IKE SA。 第二阶段：协商IPsec SA。**

第一阶段又分为主模式和快速模式 （主模式提供了对通信双方的身份保证，身份保证不必要的时候可以使用积极模式提高协商效率）

第二阶段只有快速模式

一个IKE SA协商（第一阶段）可为多个IPsec SA协商（第二阶段）提供服务

IKE的工作模式

端节点到端节点：采用传输模式

安全网关到安全网关：采用隧道模式

常见的上面两种模式的嵌套组合

IKE的工作过程

IKE是以守护进程的方式在后台运行（两个守护进程通过UDP协议来传递消息）

IKE协议使用两个数据库：安全关联数据库SADB和安全策略数据库SPDB，这两个数据库都保存在操作系统内核

如果协商成功，把新协商的SA增加到SADB中。

5.SSL-HTTPS

传输层安全协议 SSL

为应用层服务。

概述

TLS可以看作SSLv3.1

- SSL工作在TCP上，不支持UDP。
- SSL与应用层协议无关，可以支持HTTP、Telnet、FTP等面向连接的应用层协议。
- SSL为端到端的应用提供 保密性、完整性、身份认证
- SSL协议可确保信息在传输过程中不被修改

支持SSL的网站都以https开头。HTTP层先将用户需求翻译为http请求，然后SSL层协商出一个密钥加密http请求，TCP层传递SSL层处理之后的数据。

SSL体系结构

会话session：是客户端和服务器之间的一个关联，一个虚拟的连接。一个会话协商的信息可以由多个连接共享。（通过握手协议建立 ）

连接connection： 特定的通信信道，通常映射成一个TCP连接。连接是短暂的，一次访问可能需要多个连接

SSL协议由：记录协议、握手协议、告警协议、修改密码归约协议组成

层：

握手协议、告警协议、修改密码归约协议这三个和HTTP（应用层）是一层的，再往下：

SSL记录协议

TCP

IP

SSL记录协议

定义传输格式，为高层协议(比如握手协议、http啥的)提供数据封装、压缩、加密等基本功能

提供保密性、报文完整性。最后形成SSL 记录协议数据单元

SSL握手协议

协商密钥，在数据传输开始前，进行身份认证、协商加密算法、交换密钥等

SSL的部分复杂性来自于握手协议

四个阶段：建立安全能力；服务器认证和密钥交换；客户认证和密钥交换；结束

允许客户端和服务器端相互认证、协商加密和MAC算法 ，保护数据使用密钥通过SSL记录传送

SSL安全性分析

建立在RSA算法的安全性上

由于美国政府的出口限制，使得进入我国的实现了SSL的产品（浏览器和服务器）均只能提供512bits的RSA公钥、 40bits对称密钥的加密，系统安全性差

SSL协议可以很好地防范 “重放攻击”

应用层安全协议： HTTPS

Web安全

Web安全目标：
① 确保**Web服务器**存储数据的安全
② 确保**Web客户端**的计算机是安全的
③ 确保服务器和浏览器**之间**的信息传输的安全

提供web安全的三个方法：

1. IP级安全：IPSec
2. TCP级安全：使用SSL、TLS作为下层协议
3. 应用级安全：比如安全电子交易SET

HTTP的缺陷

- 明文传输，没有数据完整性校验
- 无状态链接，无法验证双方身份的真实性

HTTPS

解决了数据加密、完整性校验、对服务器的身份认证等问题

地址解析协议ARP：工作在数据链路层， 实现MAC地址与IP地址的映射 （建立和维护IP-MAC映射表，也就是ARP缓存表）

ARP欺骗是一种中间人攻击的常用手段。攻击者可以通过制造伪造的ARP frame，修改网内任何计算机的映射表， 从而切断目标主机的互联网通讯， 窃取目标主机关键信息（把信息骗到攻击者的计算机上）。

HTTP端口号为80，HTTPS端口号为443

HTTP是简单的无状态连接，HTTPS协议是由SSL+HTTTP协议构建的可进行加密传输、身份认证的连接

使用HTTPS也无法避免ARP欺骗和嗅探攻击 ， 但是即使被嗅探也无法得到明文信息。

使用HTTPS后，攻击者无法进行报文篡改。

HTTPS的加密证书是在CA（证书管理机构）申请后发放，所以拥有该证书的单位和个人是唯一的、不可被仿冒

HTTP无法验证服务器身份 。

当用户通过钓鱼WIFI或者恶意代理访问HTTPS网站时，服务器的证书可能就被替换掉了。此时信息可能被盗取。

攻击者使用自己的证书替换掉服务器的证书 ，通信过程就是用攻击者的证书进行加密，则对攻击者来说就是明文通信 。（证书替换）

会话劫持：如果一个网站只是在登录的时候使用HTTPS，其他时候使用HTTP，那么攻击者可以偷取cookie从而不用输入用户名和密码登录。

6.SET-SMIME

这些也都是应用层安全协议

安全电子交易协议SET

SET协议为在Internet上进行安全的电子商务提供了一个开放的标准，规定了交易各方进行安全交易的具体流程。为持卡人、商家、银行提供了一个多方参与的安全通信信道。

保证交易信息的私密性、保密性、完整性、抗抵赖

支付处理过程 :

1. 购买请求
2. 支付授权（才能给用户发货 ）
3. 支付获取（才能完成银行的转帐业务 ）

cardholder --- OI, PI ---> merchant --- PI ---> bank

购买请求中，Payment Info 和 Order Info 要分开加密和签名，保证用户的隐私不被泄露，给商家PI和OI，只给银行PI。PI 和OI必须有联系（这个联系就是双签名，把PI和OI绑在一起签名，防止商家单独修改其中一个），以防止商家篡改信息，产生纠纷 。

双签名

双签名，在第一步购买请求中

双签名的目的是为了连接两个发送给不同接收者的报文（具体场景见上）。

P25 有个图

PI --H--> PIMD（PIMD：PI报文摘要）

OI --H--> OIMD

POMD = H(PIMD || OIMD)（拼接之后还要来一个Hash函数，防止公钥解开双签名之后进行分割修改）

双签名=$E_{KRc}[POMD] = E_{KRc}[H(\ H(PI)||\ H(OI)\ )]$

KRc是用户的私钥（来签名）

安全电子邮件协议 S/MIME

电子邮件安全需求

匿名转发、电子邮件欺骗、垃圾邮件和邮件炸弹 、邮件病毒

安全需求包括：机密性、完整性、可认证、抗否认

电子邮件的地址表示为： 用户名@主机名； 地址的大小写不与区分

P41 电子邮件系统的组成。有一个邮件中继主机，连接不同的邮件服务器

一些关键词：

- MTA: Transfer
- MDA: Delivery
- MUA: User

主要协议：SMTP（Simple Mail Transfer Protocol，用于发送）， POP3(Post Office Protocol 3，用于收取)，IMAP4（用于收取），RFC822（电子邮件格式），MIME （是RFC822的扩展）

SMTP本身不提供任何验证 和安全服务。

RFC 821(SMTP)不能传输二进制数据，如音频、视频等 ，也不能传输非ASCII码的字符。

安全电子邮件协议：S/MIME

产业界广泛认可的协议。

提供的安全服务：认证、数据机密性、消息完整性和非否认。

组合三种公钥算法

- DSS是其推荐的数字签名算法
- Diffie-Hellman是其推荐的密钥交换算法
- 3DES是其推荐的消息加密算法

另外， RSA既可以用做签名(对消息hash之后的MD摘要进行加密即签名)，也可以加密会话密钥

其他安全电子邮件协议

安全电子邮件协议：PEM

基于RSA和DES算法，报文加密用DES，报文认证用MD5

安全电子邮件协议：MOSS

对PEM的改进

安全电子邮件协议：PGP

既是一个特定的安全电子邮件应用，也是一个安全电子邮件标准

7.Intrusion

入侵

用户入侵：入侵者入侵。未授权登录、授权用户非法获得更高级别的权限和操作。

软件入侵：恶意软件对计算机系统进行攻击。病毒、蠕虫、特洛伊木马等

有的入侵并不是基于网络的攻击，比如终端用户入侵本地系统可以不通过网络 ， 病毒或特洛伊木马可以通过磁盘传播到系统 ； 蠕虫才是网络特有的入侵方式 。

用户入侵

口令文件的保护：

- 单向加密：系统只保存用户口令的密文，验证时与密文对比
- 访问控制：对口令文件的访问仅限于几个账号

入侵检测的基本工具是审计记录。 用户活动记录应该作为入侵检测系统的输入。

审计记录有两种方法：

- 原始审计记录
- 检测专用的审计记录：需要额外收集

入侵检测方法：

- 基于统计：阈值检测、基于轮廓的检测（统计**每个用户过去**的行为特征，发现重大偏差）
- 基于规则（专家系统定义的规则集，判定入侵行为）

蜜罐技术

诱导潜在攻击者**远离重要系统**的一个圈套

软件入侵

P29 病毒、蠕虫、木马的区别

P30 各种恶意软件的树状图

恶意程序分为 需要宿主程序 和 独立于宿主程序 两类，

独立于宿主程序：Zombie，蠕虫。 都可复制

需要宿主程序：病毒（可以复制）；陷门、逻辑炸弹、特洛伊木马（不进行复制）

只有Zombie和蠕虫是独立于宿主程序的！

可以复制的只有Zombie，蠕虫、病毒。

陷门

程序的**秘密入口**，通过它，程序员可以不按照通常的访问步骤获得访问权。

一个例子：一个数据库被发现有一个万能用户名和密码

解决方法是程序的开发和软件的更新。

逻辑炸弹

预先规定了病毒和蠕虫的发作时间，是嵌在合法程序中的、只有当待定的事件出现时才会进行破坏的一组程序代码

（比如课上的例子，特定发作时间）

特洛伊木马

表面上看起来有某种有用功能的程序，它内部含有隐蔽代码，当被调用时会产生一些意想不到的后果（**伪装**成其他文件）

木马一般由木马配置程序、控制端程序和木马程序(服务器程序) 组成

控制端程序负责控制远程服务器。木马程序(服务器程序)在受害者的系统中发送数据给控制端。

Zombie

秘密地接管其它依附在Internet上的计算机，并使该计算机发动攻击，而且这种攻击很难通过追踪Zombie的创建者查出来 。被用在对拒绝服务的攻击上，尤其是对Web站点的攻击。

像僵尸一样，计算机被zombie控制了，从而进行攻击。

病毒

一种可以通过修改自身来感染其它程序的程序 （包括对病毒程序的复制 ）

通过寄居在宿主程序上，计算机病毒可以暂时控制该计算机的操作系统盘。

带毒计算机上面运行无毒软件可以使得无毒软件具有传染能力。

病毒也有潜伏阶段、传染阶段、触发阶段、发作阶段。

攻击者经常利用系统（操作系统、硬件平台等）的细节和弱点来设计病毒程序。

引导型病毒：在系统刚启动时就开始运作，阶段比较靠前

蠕虫

Morris Worm

能够自我复制、自主传播的计算机程序。利用网络上计算机系统的漏洞自主的将自己复制到其它计算机上 。

是一个独立的程序，不需要宿主。

往往占用系统或者网络资源、破坏其他程序 ，不伪装成其他程序。

另外还有一个网络蠕虫病毒。