# 综合面试应对策略（补充）

## 1.1中文问题

### 1.1.1 专业课问题

**1.计算机网络**

**1.整体计网**

（1）介绍OSI七层参考模型

（2）TCP与UDP的区别？

**2.网络层高频问题**

（1）简述一下IP协议

（2）简述一下分类IP地址

（3）简述一下RIP

（4）简述一下OSPF

（5）简述一下BGP

（6）简述一下IGP、EGP

（7）简述一下ICMP

（8）简述一下IGMP

（9）简述一下IPv4与IPv6的区别

（10）简述一下ARP

（11）简述一下IP多播

（12）简述一下MPLS

（13）简述一下FEC

（14）简述一下SR

（15）简述一下SDN

（16）简述一下NAT

（17）简述一下VPN

（18）简述一下openflow

**2.网络安全（湖科大）**

（1）什么是古典密码？

感谢老师的提问！

古典密码就是早期人们用来加密信息的方式，主要通过 **字符替换或重新排列** 来隐藏内容，相比现代密码学，它的数学原理比较简单，但安全性较低。

比如：

凯撒密码，采用的是**移位的方式**，属于**替换密码**，将字母表**整体循环右移三位**与原来的字母表建立一一对应关系。后来推广为**维吉尼亚密码**，即为每个字母的偏移量都不一样，提高了安全性，但仍可以通过**频率分析的方法**来进行破译。

还有**栅栏密码**：属于**置换加密**，把明文按照特定规则写成网格，然后按行读取，打乱原来的顺序。

最后是密码机如二战时期德国用的埃尼格玛密码机，它的加密方式每天都会变化，非常难破解，但后来被图灵团队攻破了。

（2）对称密码（**DES、AES、SM4**）非对称密码（**RSA、SM2、ECC、ElGamal**）

**1.解释一下对称密钥密码体制。**

感谢老师的提问！

对称密钥密码体制，就是加密和解密使用同一个密钥的加密方式。也就是说，加密方和解密方必须共享一个密钥，只有双方都拥有这个密钥，才能正确加密和解密数据。

**1. 具体工作方式**

**首先，发送方使用密钥加密明文**，得到密文。

**然后，接收方用相同的密钥解密密文**，恢复出原始数据。

这个过程中，如果密钥被第三方窃取，那数据也就不安全了。

**2. 主要特点**

**第一、加密速度快**：因为它的计算过程比非对称加密简单，所以适合大规模数据加密，比如文件加密、磁盘加密、VPN 等场景。

**第二、密钥管理是个难题**：由于通信双方必须共享同一个密钥，所以 **如何安全地分发密钥** 是对称加密最大的挑战。如果密钥被截获，攻击者就能解密所有数据。

**3. 常见的对称加密算法有**

**DES（数据加密标准）和AES（高级加密标准）**

总结来说，对称加密在速度和效率上很有优势，但密钥管理是个难点，所以实际应用中 **经常和非对称加密结合**，比如用 RSA 先加密对称密钥，再用 AES 进行数据加密，这样既安全又高效。”

***4. 现实应用***

***文件加密****：比如 ZIP 压缩加密、数据库加密等。*

***网络通信****：HTTPS 里用对称加密保护数据传输，像 AES。*

***磁盘加密****：BitLocker、Android 文件加密等，都是对称加密。*

**2.解释一下非对称密钥密码体制**

感谢老师的提问！

“公钥密码体制，也叫 **非对称加密**，它和对称加密的最大区别是——它使用 **一对密钥：公钥和私钥**，而不是一个密钥。

**1. 工作原理**

**公钥（Public Key）是公开的，任何人都能获取**。

**私钥（Private Key）必须严格保密，只能由拥有者掌握**。

加密和解密使用不同的密钥，**用公钥加密的数据，只有对应的私钥才能解密**，反之亦然。

**2. 主要特点**

**1.安全性高**：即使攻击者拿到了公钥，没有私钥也无法解密。

**2.计算复杂，速度较慢**：公钥加密依赖于大整数因子分解等数学难题，计算量比对称加密大很多，适合小数据加密，而不是直接加密整个文件。

**3.密钥管理更方便**：不像对称加密需要双方共享密钥，公钥可以公开，私钥由个人保管，避免了密钥分发的问题。

**3. 常见的公钥加密算法有**

**1.RSA（最经典的非对称加密算法）**：基于大素数因子分解难题，公钥和私钥通过数学关系生成，广泛用于 HTTPS、数字签名等场景。

**2.ECC（椭圆曲线加密）**：比 RSA 更高效，密钥更短但同样安全，适用于移动设备和区块链。

**4. 总结来说**

公钥加密的 **安全性比对称加密高，但速度慢**，所以实际应用中，通常 **结合对称加密** 使用，比如 HTTPS 里用 RSA 加密 AES 密钥，然后用 AES 处理数据，这样既高效又安全。”

***4. 现实应用***

***1.HTTPS****：浏览器访问安全网站时，服务器的公钥用于加密对称密钥，然后双方用 AES 进行数据加密。*

***2.数字签名****：发件人用私钥签名，接收方用公钥验证，确保数据没有被篡改，广泛用于软件签名、电子合同等。*

***3.加密邮件和消息****：像 PGP（Pretty Good Privacy）邮件加密，用户的公钥用来加密邮件，只有拥有私钥的人能解密。*

**3.解释一下对称密钥密码体制与公钥密钥密码体制的区别**

感谢老师的提问！

“对称加密和公钥加密是两种常见的加密方式，它们的核心区别在于 **密钥的使用方式、速度与性能**和**应用场景**。

**第一、密钥使用方式不同**

**对称加密（Symmetric Encryption）**：加密和解密用 **同一个密钥**，就像一把锁和一把钥匙，既能锁门，也能开门。

**公钥加密（Asymmetric Encryption）**：用 **一对密钥**，**公钥加密，私钥解密**，就像一个保险箱，任何人都可以往里放东西（用公钥加密），但只有持有私钥的人能打开（解密）。

**第二、速度与性能对比**

**对称加密速度快**，因为它的数学运算相对简单，适合大数据量的加密，比如 AES 就广泛用于磁盘加密、VPN、HTTPS 传输数据等。

**公钥加密速度慢**，因为它依赖于复杂的数学运算，比如大素数因子分解（RSA）或椭圆曲线（ECC），通常**不会直接用于大数据加密**，而是用于 **密钥交换、身份认证** 等场景。

**第三、安全性对比**

**对称加密的安全性依赖于密钥的保密性**，如果密钥被泄露，攻击者就可以直接解密数据，因此 **密钥分发和管理是个难题**。

**公钥加密更安全**，因为公钥可以公开，只有私钥需要保护，不用担心密钥分发问题。但它仍然有一定的风险，比如私钥被盗取，就会造成安全问题。

**最后、总结来说**

**对称加密**：快，但密钥管理难，适合大数据加密。

**公钥加密**：慢，但密钥管理方便，适合身份认证和密钥交换。

***5. 现实应用***

***对称加密常用于数据传输和存储****，比如* ***AES****，主要用于文件加密、磁盘加密、Wi-Fi 加密等。*

***公钥加密主要用于密钥交换和身份验证****，比如* ***RSA、ECC****，广泛用于 HTTPS、数字签名、加密邮件等。*

***6. 实际应用中的结合***

*由于公钥加密慢、对称加密快，在现实中，****两者通常结合使用****。比如在 HTTPS 里，****公钥加密（RSA/ECC）先加密对称密钥（AES），然后用对称加密传输数据****，这样既安全又高效。*

***现实中两者配合使用****，既保证安全，又保证效率。”*

**4.什么是DES?/AES?/SM4?**

**<1> DES**

感谢老师的提问！

DES（数据加密标准，Data Encryption Standard）是早期的一种 对称加密算法，曾经被广泛用于数据加密，但现在已经不够安全，被 AES 取代了。

1.DES 的加密方式

**简单版本：**

DES 的核心思想是“分组加密”，首先将明文进行分组，每个分组为64位，然后把数据按 64 位（8 字节） 为一组进行加密，每组数据通过多个 置换（Permutation）和替换（Substitution） 进行复杂变换，最终得到64位的密文。最后将64位的密文拼接起来，即可得到最终的密文，其中它的密钥长度是 56 位，外加 8 位用于校验。

**专业版本：**

DES 是一种对称加密算法，采用 **分组加密** 方式，每次处理 **64 位（8 字节）** 的数据块。它的密钥长度是 **56 位**，额外的 **8 位是奇偶校验位（Parity Bits）**，用于错误检测，但不参与加密计算。加密时，数据会经历 **初始置换（IP）**，然后进入 **16 轮 Feistel 变换**。每一轮：

把数据分成 **左右两部分**（L 和 R）。

右半部分 **通过 F 函数与子密钥结合** 进行非线性替换和置换。

计算结果 **与左半部分异或（XOR）**，然后左右交换。

经过 16 轮后，数据再经过 **逆置换（IP⁻¹）**，生成 **64 位密文**。 如果明文长度不是 **64 位的整数倍**，则需要进行 **填充（Padding）**。最终，所有的密文块拼接在一起，形成完整的加密结果。

2. DES 的优点和缺点

优点：

结构清晰，计算速度快，基于 Feistel 结构。

缺点：1.密钥长度太短，只有 56 位，现代计算机很容易通过 暴力破解（穷举法）在短时间内破解 DES。容易受到差分分析、线性分析攻击，安全性不足。

3. 现实应用

虽然 DES 现在已经不再推荐使用，但在一些 老旧的银行系统、支付系统 里，仍然能看到 3DES 的身影。不过大多数现代系统都已经全面转向 AES 或 ChaCha20。

4. 总结

DES 是一种对称加密算法，采用 56 位密钥，对 64 位数据进行 16 轮加密。

由于密钥太短，已经不安全，被 AES 取代，但对后续加密算法影响深远。

现实中 3DES 仍有一些应用，但大多数系统已经升级到 AES。”

**<2> AES**

“AES（高级加密标准，Advanced Encryption Standard）属于对称密码密钥体制，安全性比 DES 强得多~~，现在被广泛用于各种加密场景，比如 HTTPS、VPN、无线网络（WPA2/WPA3）、文件加密等。”~~

1. AES 的核心特点

分组加密（Block Cipher）：AES 以 128 位（16 字节）为一个数据块 进行加密，而不是像流加密那样逐字节加密。

支持不同密钥长度：AES 可以使用 128 位、192 位或 256 位密钥，密钥越长，安全性越高。

速度快，安全性高：相比于 DES，AES 的加密速度更快，而且 不容易被暴力破解。

2.AES的加密过程（大致业余）

“AES 的加密过程其实可以简单理解为一个多轮变换的过程。”

首先，把明文数据分成 128 位（16 字节） 的小块，然后与密钥进行一次初始混合（XOR 运算）。

然后，进入多轮加密，每轮包含三个主要步骤：

1.字节替换（SubBytes）：用一个特定的“替换盒”把每个字节替换成新的值，让数据更加混淆。

2.行移位（ShiftRows）：对数据的排列进行一定规则的移动，使得相邻的数据块互相影响。

3.列混淆（MixColumns，最后一轮省略）：对数据做更深层次的混合，增加复杂性。

再与密钥进行一次 异或运算（AddRoundKey），确保加密的不可预测性。

**经过 10/12/14 轮加密（取决于密钥长度），最终生成加密后的密文。**

解密的过程和加密相似，只是顺序反过来。AES 由于其安全性高、计算速度快，广泛用于各种数据加密，比如 HTTPS、Wi-Fi 安全、文件加密等。

**2. AES 的加密过程（详细专业）**

AES 并不像 DES 那样使用 Feistel 结构，而是采用了一种叫做 “SP网络（Substitution-Permutation Network）” 的设计，也就是 “替换 + 置换” 方式，使得数据更加难以破解。

AES 主要有 四个核心操作，每个数据块都会经过多轮加密处理（10 轮 / 12 轮 / 14 轮，取决于密钥长度）：

首先，用密钥对明文做一个初始 轮密钥加（AddRoundKey），相当于给数据“加盐”，让加密的随机性更高。

然后，进入多轮加密，每轮都包含四步：

字节代替（SubBytes）：用 S 盒（S-Box）对数据进行非线性替换，增加混淆性，防止攻击者直接推测明文。

行移位（ShiftRows）：打乱数据的排列，使得相邻的数据块彼此影响。

列混淆（MixColumns）（最后一轮省略）：通过矩阵运算让数据更加扩散，增加复杂度。

轮密钥加（AddRoundKey）：再和子密钥进行 XOR 运算，提高安全性。

最后，完成所有轮次后，生成最终的 128 位密文。

解密过程和加密类似，但操作顺序相反。

3. 为什么 AES 更安全？

密钥长度更长（128/192/256 位），比 DES 的 56 位密钥强得多，不容易被暴力破解。

抗攻击能力强，AES 不会受到差分分析、线性分析等常见攻击方式的影响。

没有 Feistel 结构的局限，AES 的 SP网络结构让数据的混淆和扩散更均匀，提高了安全性。

4. AES 的应用场景

AES 现在已经成为 全球通用的加密标准，被广泛用于：

HTTPS 加密，保证网络通信的安全。

WPA2/WPA3 无线网络加密，用于 Wi-Fi 安全性。

文件加密，比如 ZIP 加密、BitLocker 磁盘加密。

区块链、电子支付等场景，保护交易数据的安全。

5. 总结

AES 是一种对称加密算法，以 128 位为分组，加密速度快，安全性高。

它使用 SPN 结构，主要通过 S 盒替换、行移位、列混淆和轮密钥加等操作进行加密。

**<3> SM4**

“SM4 属于我国国家的标准加密算法，属于对称加密。它的设计理念和 AES 类似，用来保护数据的机密性，广泛应用于国内的金融、政府、通信等领域。”

**1. SM4 的基本特点**

对称加密：SM4 和 AES 一样，是一种对称加密算法，意味着加密和解密使用同一个密钥。

分组加密：它也是一种 分组加密算法，每次加密 128 位（16 字节）数据。

密钥长度：SM4 使用 128 位的密钥，与 AES 相同。

安全性：SM4 的安全性是经过中国国家密码管理局认证的，设计时考虑了防止各种常见的攻击，比如差分分析、线性分析等。

**2. SM4 的加密过程**

**业余版本**

“SM4 的加密过程其实就是分几个简单的步骤。”

首先，把明文数据分成 128 位（16 字节）的小块。

然后，通过密钥生成一系列 加密密钥，每一轮都使用不同的密钥。

接着，把数据分成左右两部分，进行 32 轮的加密，每一轮主要做：

1.替换数据，让数据变得复杂。

2.移动数据的位置，增加混淆。

3.再混合数据，让数据更加难以猜测。

4.用密钥进行混合，确保加密数据是不可预测的。

最后，经过 32 轮处理，得到最终的密文。

**专业版本**

SM4 使用 “Feistel 网络” 的结构进行加密，类似于 DES，但它有 32 轮的加密过程，相比 DES 更多轮的加密能提供更高的安全性。

数据分组：每次加密 128 位的数据，分成左右两部分。

轮密钥生成：SM4 通过密钥扩展算法生成 32 个轮密钥，每一轮都使用不同的密钥。

加密过程：数据会通过 非线性替换、行移位、列混合等步骤进行变换，最终得到密文。

解密过程：解密时和加密过程类似，只是密钥顺序相反，保证了算法的可逆性。

**3. 总结**

SM4 是中国自主研发的对称加密算法，采用 128 位密钥，适用于加密 128 位的数据。

它通过 Feistel 网络和 32 轮加密，保证了数据的安全性。

广泛应用于国内的金融、政府、通信等领域，符合国家标准。

**5.什么是RSA? /ECC? /SM2?/EIGamal?**

**<1>RSA**

RSA属于公钥密钥密码体制，基于大数质因数分解难题设计的，之所以可以基于大数质因数分解安全，是因为从一个大数的乘积中找到其质因数非常复杂，尤其当数的位数非常大时。RSA使用公钥加密，私钥解密。虽然安全性高，但计算复杂度较大，因此在实际应用中常用于密钥交换，而不会直接用于大数据加密。

详细过程

 **首先**，生成两个大质数 **p 和 q**，然后计算它们的乘积 **n = p \* q**，这个值会作为公钥的一部分。

 **然后**，计算 **n-1** 的欧拉函数（Euler’s Totient Function），也就是 **φ(n) = (p-1)(q-1)**。

 **接着**，选择一个数字 **e**（通常是 65537），它必须与 φ(n) 互质，也就是说它和 φ(n) 没有共同的因子。

 **再然后**，计算 **d**，使得 **e \* d ≡ 1 (mod φ(n))**。这个 **d** 就是私钥的一部分。

 这样，就得到了 **公钥 (e, n)** 和 **私钥 (d, n)**。

**<2>ECC**

**“ECC（椭圆曲线密码学）属于** **公钥密钥体制**，即使用公钥加密、私钥解密。它的核心原理是基于‘椭圆曲线离散对数问题’， 利用椭圆曲线上的点运算，使得从一个基点跳跃多个步数生成公钥，而想要逆推回私钥几乎不可能，从而保证安全性。相比 RSA，ECC 在提供相同安全性的情况下，密钥长度更短（如 256 位 ECC ≈ 3072 位 RSA），计算速度更快，资源占用更少，特别适合计算能力受限的场景，如物联网、移动设备、区块链、TLS 证书等。”

原理：

ECC 依赖于 “椭圆曲线离散对数问题”，它的安全性比 RSA 依赖的质因数分解难题更强。

首先，想象一个二维坐标平面，上面有一条特殊的曲线，叫做椭圆曲线。

然后，在这条曲线上，我们选一个基点（G），按照一定的数学运算，反复“跳跃”多个点，得到另一个点 P。

公钥 就是这个 P，而私钥是“跳了多少步”（也就是一个很大的随机数）。

解密的难点在于：如果别人知道 G 和 P，想要反推出跳了多少步（也就是私钥），这个计算非常难，几乎不可能完成。

这就像一个单向的数学迷宫，进入很容易，出来却极难。

**<3>SM2**

“SM2 是我国自主设计的椭圆曲线公钥加密算法，属于（公钥密码密钥体制）非对称加密体制，和 ECC（椭圆曲线密码学）同属一类。它基于‘椭圆曲线离散对数问题’，

相对于 RSA：SM2 密钥更短，但提供相同的安全性（如 256 位的 SM2 ≈ 2048 位的 RSA），计算速度更快，加密和签名效率更高。

相对于 ECC（如 ECDSA）：SM2 采用了优化的算法，特别适用于国密标准，增强了防御攻击能力，比如抗侧信道攻击的特性更强。

安全性高，密钥长度更短，计算效率更高，主要用于数字签名、密钥交换和加密通信。”

1. SM2 的核心原理

SM2 的原理和 ECC 类似，都是基于椭圆曲线上的数学运算。

首先，在一条特定的椭圆曲线上，选择一个基点 G，然后利用大数运算生成私钥 d，并计算出对应的公钥 P（P = dG）。

然后，在加密时，发送方会用接收方的公钥进行加密，保证只有持有私钥的人才能解密。

最后，由于椭圆曲线离散对数问题非常难以求解，即使攻击者知道公钥 P，也很难反推出私钥 d，从而保证安全性。

2. SM2 的应用场景

SM2 是我国国家商用密码标准，广泛应用于政府、金融、通信等需要高安全性的领域，包括：

HTTPS/TLS：用于网站安全证书，保障数据传输安全。

数字签名：如电子合同、电子政务，确保数据不可篡改、身份可验证。

密钥交换：用于安全通信，确保只有通信双方能解密消息。

移动支付、金融安全：很多银行、支付平台都在使用 SM2 进行交易验证和身份认证。

**<4>EIGamal**

ElGamal（埃尔加马尔）是一种基于‘离散对数问题’的非对称加密算法，主要用于加密和数字签名。它的核心思想是：用一个大素数下的指数运算来生成公钥和私钥，依靠数学上的单向难题保证安全性。

相比 RSA：ElGamal 依赖于离散对数问题，而 RSA 依赖于大数因子分解问题，ElGamal 的加密过程计算量更大，但密钥扩展性好。

相比 ECC：ECC 也依赖离散对数问题，但 ECC 在椭圆曲线的数学运算上更高效，所需密钥更短，而 ElGamal 的密钥较长。

一个缺点是：ElGamal 生成的密文比明文长，这在某些存储或带宽受限的场景下不够高效。

ElGamal 的应用场景

数字签名：ElGamal 变种（如 DSA，数字签名算法）被用于电子签名。

加密通信：用于一些端到端加密协议，保护数据传输的安全。

区块链：某些加密货币使用基于 ElGamal 的变种方案来保障交易安全

**（4）描述常见的网络攻击类型及其防御措施。**

1. DDoS（分布式拒绝服务攻击）

“首先，DDoS 是一种让服务器‘忙到崩溃’的攻击。攻击者利用网络设备漏洞控制一堆‘僵尸主机’（被黑客控制的设备），不断向目标服务器发送海量请求，导致服务器资源被耗尽，正常用户就无法访问了。

怎么防呢？可以用 CDN 来分流流量，部署 Web 应用防火墙（WAF）来识别异常请求，还可以对单个 IP 限流，避免被刷爆。”

2. SQL 注入

“然后是 SQL 注入，它的本质就是攻击者在输入框里偷偷塞进一段 SQL 语句，让服务器执行本不该执行的数据库操作，比如直接把整个用户表都查出来。

防御的方法主要是用预编译 SQL 语句，不让用户输入的内容直接拼接进 SQL 语句里，同时对输入内容进行严格校验，防止恶意代码被执行。”

3. 中间人攻击（MITM）

“这个攻击方式是，攻击者‘卡’在你和服务器之间，偷偷拦截和篡改你的数据，比如你以为在访问某个安全网站，实际上流量已经被劫持了。

防御方式很明确：一定要使用 HTTPS，确保数据传输是加密的，不要随便连接不安全的公共 Wi-Fi，或者使用 VPN 加密流量，避免被监听。”

**（5）解释防火墙的作用及其类型。**

“防火墙，专门负责检查进出网络的数据流，决定谁能进、谁不能进。它的主要作用是防止非法访问、阻挡恶意流量，保护内部网络不受攻击。”

1. 包过滤防火墙（Packet Filtering Firewall）

📌 作用：“这种防火墙最基础，它通过检查每个数据包的源 IP、目标 IP、端口号、协议等信息，来决定是否放行。”  
🔹 举个例子：“就像小区门口的保安，看一眼你的门禁卡或者身份证，如果符合规定，就让你进；如果不符合，直接拦住。”  
🔹 优势：“简单高效，性能开销小。”  
🔹 缺点：“但它只检查表面信息，不看数据内容，黑客可以伪造 IP 地址绕过它。”

2. 状态检测防火墙（Stateful Inspection Firewall）

📌 作用：“比包过滤更智能，不只是检查单个数据包的基本信息，还会跟踪整个通信过程，确保数据流是正常的。”  
🔹 举个例子：“还是小区门禁的例子，除了检查你的身份证，它还会查你的历史记录，看看你之前是不是正常进入过小区，而不是冒充别人混进来的。”  
🔹 优势：“比单纯的包过滤更安全，因为它能防止伪造的连接，比如 TCP 三次握手不完整的请求会被拦截。”  
🔹 缺点：“但它对系统资源消耗更大，需要记录大量会话信息。”

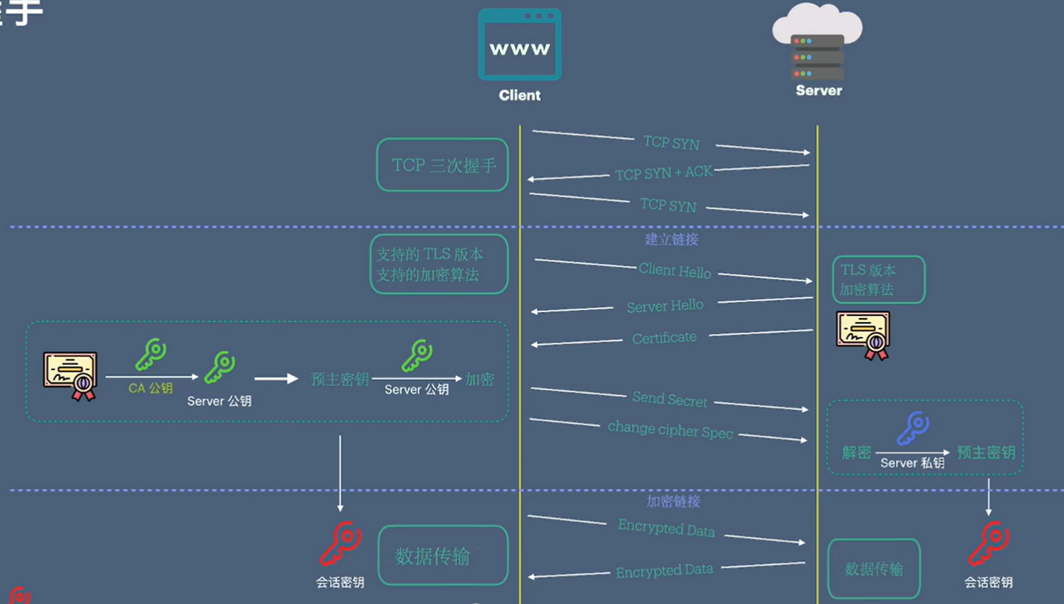
3. 代理防火墙（Application Gateway / Proxy Firewall）

📌 作用：“它不直接让你的数据包通过，而是作为中间人，代表你去访问目标网站，然后把数据返回给你，相当于加了一道‘中转’。”  
🔹 举个例子：“你去银行取钱，不能直接进金库，而是把取款请求交给柜员，柜员审核后，帮你把钱取出来再交给你。这样，即使有坏人想抢银行，他也接触不到真正的金库。”  
🔹 优势：“它能检测到更高级的攻击，比如 SQL 注入、XSS（跨站脚本攻击）等。”  
🔹 缺点：“但这种方式会影响访问速度，因为数据都要先经过代理。”

4. Web 应用防火墙（WAF，Web Application Firewall）

📌 作用：“这个是专门保护网站的，主要防止 SQL 注入、XSS、CSRF（跨站请求伪造）等 Web 攻击。”  
🔹 举个例子：“如果你的网站是个餐厅，普通防火墙就像大门的保安，而 WAF 则是厨房里的食品安全员，专门检查送来的食材，防止有毒食品进入。”  
🔹 优势：“特别适合保护 Web 应用，比如电商网站、银行系统等。”  
🔹 缺点：“但它只针对 Web 攻击，对其他类型的网络攻击，比如 DDoS，就没那么有效。”

**（6）描述 SSL/TLS 协议的工作流程，数据库如何实现SSL加密连接？MQTT如何采用TLS加密传输MQTTS？**

当客户端开始访问服务器时，首先由客户端发起请求，并通过TCP/IP的三次握手，与服务器端建立连接，然后客户端向服务器端发送Hello请求，请求中包含TLS所支持的TLS版本以及支持的加密算法等信息，服务器端在接收到来自客户端的Hello请求后，会向客户端返回一个Hello请求，请求中将包含服务器端将采用的TLS版本以及加密算法等信息，另外服务器端还会将自己的证书发送给客户端，客户端收到证书后，首先通过验证指纹和签名验证书的真实性。一旦验证成功，即采用系统中的CA公钥提取出服务器端的公钥，然后客户端生成预主密钥，并使用从证书中提取出的公钥对其进行加密。在完成该过程后，客户端会将加密后的预主密钥及使用的加密算法等一并传递给服务器端，客户端与服务器端使用相同的加密函数与算法，以此来生成完全一致的会话密钥。接下来的通信采用会话密钥进行。

**详细的版本**

“SSL/TLS 主要是用来保证客户端和服务器之间的安全通信。它的工作流程可以分成几个关键步骤：

**第一步**，客户端和服务器先通过 **TCP 三次握手** 建立连接。

**第二步**，客户端发起 **Client Hello**，告诉服务器它支持的 **TLS 版本、加密算法**，并发送一个 **随机数**。

**第三步**，服务器回复 **Server Hello**，确认使用的 TLS 版本、加密算法，并再发送一个随机数。同时，服务器会把自己的 **SSL 证书** 也发过来，这个证书由 **CA（证书颁发机构）** 签发，里面包含服务器的 **公钥**。

**第四步**，客户端检查这个证书是否有效，客户端用 CA 公钥验证服务器证书的签名，确保它真实可信，没有被篡改！比如 **是不是被可信 CA 签发的，有没有过期，域名是否匹配**。如果证书没问题，就继续下一步。

**第五步**，双方开始 **密钥交换**。如果是老的 **RSA 密钥交换**（TLS 1.2），客户端会生成一个 **预主密钥（Pre-Master Secret）**，用服务器公钥加密后发给服务器，服务器再用自己的私钥解密。更现代的 **ECDHE 密钥交换**（TLS 1.3 默认），客户端和服务器会交换 **公钥**，然后各自用 **椭圆曲线 Diffie-Hellman 算法** 计算出相同的共享密钥。

**第六步**，客户端和服务器用各自的密钥计算 **会话密钥**，这个密钥之后会用来加密通信。

**第七步**，双方确认密钥已经正确生成，并通知对方 **“接下来咱们就正式加密传输了”**。

**最后一步**，正式进入加密通信，后续数据（比如登录信息、交易数据）都会用 **对称加密**（AES、ChaCha20 等）来保护，确保传输的安全性。

如果是 **TLS 1.3**，整个过程会更快，因为它优化了握手流程，减少了延迟，也更加安全，比如去掉了老旧的 **RSA 密钥交换**，默认使用 **前向安全（PFS）** 的 **ECDHE**。

所以，SSL/TLS 其实就是一套 **身份验证 + 密钥交换 + 数据加密** 的完整机制，确保我们在使用 HTTPS 网站时，信息不会被窃听或篡改。”

**（7）简述一下DDoS攻击，以及防御措施？**

该服务器因为繁忙，拒绝向该主机提供服务，攻击者也可以利用系统漏洞控制因特网上成百上千台主机，使他们成为僵尸主机，同时利用这些僵尸主机向某个服务器发送猛烈的攻击。

导致该服务器瘫痪，当其他主机向该服务器发送正常请求时，该服务器因为瘫痪而拒绝向主机提供服务。

**（9）HTTP与HTTPS的区别？**

HTTP 和 HTTPS 的主要区别在于安全性。HTTP（超文本传输协议）是明文传输的，数据没有加密，容易被窃听、篡改或冒充。而 HTTPS（超文本传输安全协议）是在 HTTP 的基础上加入了 SSL/TLS 加密，确保数据在传输过程中是加密的，并且通过数字证书验证服务器身份，防止中间人攻击。

**3.操作系统**

**4.编程语言**

（1）**解释面向对象编程（OOP）的四大特性。**

### 1.1.2 自我介绍+简历问题

**1.这个项目中有什么与网络安全相关的？**

**（1）简述数字签名？**

数字签名的作用就是保证数据的完整性、身份认证和防抵赖，它的工作过程可以分为三个步骤：首先是签名，然后是传输，最后是验证。

首先，发送方会先对原始数据做 **哈希运算**，得到一个固定长度的摘要（比如用 SHA-256）。然后，发送方用自己的 **私钥** 对这个哈希值加密，生成 **数字签名**，并把签名和原始数据一起发送给接收方。

然后，在传输过程中，接收方收到数据后，需要验证签名的真实性。

最后，接收方会用相同的哈希算法计算数据的哈希值，同时用 **发送方的公钥** 解密数字签名，得到原始的哈希值。如果两个哈希值一致，说明数据完整、未被篡改，并且确实来自发送方；如果不一致，就说明数据可能被篡改或者来源不可信。

数字签名广泛应用在 **SSL/TLS 证书、软件签名、电子合同、区块链** 等场景，是确保数据安全的重要技术。

**（2）SSL/TLS在HTTPS中的工作过程**

在 HTTPS 中，SSL/TLS 的工作流程主要分为两个阶段：首先是握手阶段，然后是数据传输阶段。

首先，客户端（比如浏览器）向服务器发送一个 Client Hello，告诉服务器自己支持哪些加密算法，并附带一个随机数。然后，服务器回复 Server Hello，选择加密算法，并提供自己的数字证书，里面包含服务器的公钥，同时也带上一个随机数。

接下来，客户端会验证服务器的证书是否可信。如果证书没问题，客户端会生成一个 **预主密钥（Pre-Master Secret）**，然后用服务器的公钥加密后发送出去。服务器收到后，会用自己的私钥解密，并和前面的随机数一起计算出最终的会话密钥。

握手完成后，双方开始进入数据传输阶段。这个时候，HTTP 数据就会用 **对称加密**（比如 AES）进行加密传输，确保数据不会被窃听。同时，还会加上 **消息认证码（HMAC）** 来保证数据完整性，防止被篡改。

最后，当通信结束时，双方会协商关闭连接，销毁会话密钥，确保数据安全。这就是 SSL/TLS 让 HTTP 变成 HTTPS 的大致过程

**3.你对网络安全有什么了解？（你了解哪些网络安全的事件？）**

感谢老师的提问！

我在复试备考期间，了解到的最让我印象深刻的网络安全案例是16年的Mirai僵尸网络攻击，攻击者首先通过随机的IP地址扫描，寻找哪些默认开启Telnet和SSH端口的IoT设备，然后使用默认的用户名与密码尝试登录这些设备，因为有相当一部分的IoT设备并没有修改默认的用户名与密码，所以这类IoT设备就很容易通过这种弱口令暴力破解的方式破解。然后，Mirai登录成功后，首先终止掉运行在该设备上的其他恶意程序，然后关闭如Telnet、SSH这样的端口，防止用户修改或其他恶意软件入侵。最后，Mirai从提前部署好的指挥控制服务器下载恶意代码，代码会驻留在设备的内存中，不写入存储器，这样设备重启后感染就会消失，避免被长期跟踪，留下证据。这样变成僵尸的设备就会等待指挥控制服务器下发指令，随时对某个服务器进行大规模的DDoS攻击。如在16 年，Mirai 利用数十万台 IoT 设备发动 **1.2 Tbps 的 DDoS 攻击**，瘫痪了美国的 DNS 服务商 Dyn，导致 Twitter、Netflix、GitHub 等网站无法访问。

**追问：1.DDoS的攻击种类有哪些？**

Mirai 的主要目标是发动 **大规模 DDoS（分布式拒绝服务）攻击**，导致目标服务器瘫痪。它支持多种攻击模式：

**这四种攻击都是 DDoS（分布式拒绝服务）攻击 的常见手段，目的是让目标服务器资源耗尽，无法为正常用户提供服务。下面详细介绍它们的攻击原理。**

**1. TCP SYN Flood（SYN 泛洪攻击）**

**攻击原理**

**SYN Flood 利用 TCP 三次握手的机制，通过伪造大量 SYN 请求，让服务器不断分配资源，最终导致资源耗尽。**

**正常的 TCP 三次握手过程：**

1. **客户端 → 服务器：发送 SYN 请求，表示希望建立连接。**
2. **服务器 → 客户端：回复 SYN-ACK，表示同意建立连接，并分配连接资源。**
3. **客户端 → 服务器：发送 ACK，确认连接成功。**

**SYN Flood 的攻击方式：**

* **攻击者发送大量伪造源 IP 地址的 SYN 数据包。**
* **服务器收到 SYN 后，会发送 SYN-ACK 并分配连接资源。**
* **但因为攻击者伪造了 IP，服务器永远收不到最终的 ACK，连接会一直处于 "半连接"（Half-Open） 状态。**
* **服务器资源被大量“未完成的连接”占满，导致拒绝正常用户的连接请求。**

**危害**

* **服务器因连接数被占满，无法接受新的请求。**
* **资源被攻击者耗尽，导致服务器崩溃。**

**防御方法**

**✅ 使用 SYN Cookie：不直接分配资源，而是用 Cookie 机制验证请求的合法性。  
✅ 缩短 TCP 超时时间：降低“半连接”的存活时间。  
✅ 启用防火墙或 DDoS 防护系统：检测和丢弃异常流量。**

**2. UDP Flood（UDP 泛洪攻击）**

**攻击原理**

**UDP 是一种无连接协议，不像 TCP 需要三次握手，服务器在接收到 UDP 数据包时，不会检查来源是否真实。攻击者可以利用这一特点发送大量 UDP 数据包，消耗目标服务器的带宽和处理资源。**

**UDP Flood 主要有两种攻击方式：**

1. **直接发送大量 UDP 数据包**
   * **攻击者向目标服务器的多个 UDP 端口发送大量无意义的 UDP 数据，服务器需要不断处理这些数据，占用 CPU 资源。**
   * **服务器可能会返回 ICMP “端口不可达” 响应，进一步增加系统负担。**
2. **UDP 反射攻击（Amplification Attack）**
   * **攻击者伪造受害者 IP 地址，向公共 UDP 服务器（如 DNS、NTP、SNMP 服务器） 发送请求。**
   * **这些服务器会向受害者回传大流量数据，造成带宽耗尽。**

**危害**

* **占满服务器带宽，导致网络延迟和丢包。**
* **消耗服务器 CPU 资源，影响正常服务。**

**防御方法**

**✅ 限制 UDP 流量速率，使用流量控制策略（Rate Limit）。  
✅ 启用防火墙，屏蔽不必要的 UDP 服务（如未使用的 DNS、NTP 端口）。  
✅ 使用流量清洗（Scrubbing），过滤掉恶意 UDP 流量。**

**3. ACK Flood（ACK 泛洪攻击）**

**攻击原理**

**ACK Flood 主要针对防火墙和服务器的连接跟踪机制（TCP Connection Tracking）。**

**正常的 TCP 连接状态：**

1. **客户端发送 SYN，服务器回复 SYN-ACK。**
2. **客户端再回复 ACK，连接建立。**
3. **数据传输过程中，TCP 需要不断发送 ACK 确认收到数据。**

**ACK Flood 的攻击方式：**

* **攻击者伪造大量 TCP ACK 包，直接发送给服务器。**
* **服务器接收到 ACK 包后，防火墙会尝试在连接表中找到对应的 SYN-ACK 记录。**
* **由于这些 ACK 包是伪造的，不对应任何真正的连接，防火墙和服务器的连接跟踪表会被填满，导致正常的连接无法建立。**

**危害**

* **耗尽防火墙和服务器的连接跟踪表，导致新连接无法建立。**
* **服务器需要不断检查连接状态，消耗 CPU 资源，影响正常业务。**

**防御方法**

**✅ 提高防火墙连接跟踪表的容量，避免快速被填满。  
✅ 设置连接跟踪超时策略，减少无效连接的影响。  
✅ 使用速率限制（Rate Limit），控制 ACK 流量的速率。**

**4. HTTP Flood（HTTP 泛洪攻击）**

**攻击原理**

**HTTP Flood 是一种应用层 DDoS 攻击，攻击者伪装成正常用户，向 Web 服务器发送大量 HTTP 请求，导致服务器过载。**

**常见的 HTTP Flood 攻击方式：**

1. **普通 HTTP 请求泛洪**
   * **攻击者控制大量僵尸设备，向服务器发送高频 HTTP 请求（如 GET / POST）。**
   * **服务器需要处理每个请求，消耗大量 CPU 和数据库资源。**
2. **慢速 HTTP 攻击（Slowloris Attack）**
   * **攻击者发送不完整的 HTTP 请求，但不关闭连接，让服务器一直等待数据。**
   * **服务器的连接池被占满，导致正常用户无法访问。**
3. **高消耗请求**
   * **攻击者发送需要大量计算或数据库查询的请求（如搜索、动态页面生成）。**
   * **服务器资源被攻击流量耗尽，影响正常业务。**

**危害**

* **服务器的 CPU、内存、数据库 资源被耗尽，导致 Web 服务崩溃。**
* **无需伪造 IP 地址，因此难以通过传统防火墙拦截。**

**防御方法**

**✅ 启用 WAF（Web 应用防火墙），检测异常 HTTP 请求并拦截。  
✅ 设置速率限制（Rate Limit），限制单个 IP 的访问频率。  
✅ 启用验证码（CAPTCHA），防止僵尸网络模拟正常用户访问。  
✅ 使用 CDN（内容分发网络），减少源服务器的负载。**

**追问2.如何预防Mirai类型的DDoS攻击？**

我认为可以从3个方面来防范Mirai僵尸网络攻击，第一，在购买IoT设备后，要修改默认密码，设置更加复杂的个人密码，防止攻击者通过弱口令暴力破解IoT设备，第二，**关闭不必要的服务**，禁用 Telnet、SSH 等远程访问服务，减少攻击面。第三，**定期更新固件**：IoT 设备厂商应提供安全补丁，修复已知漏洞。

**追问3：如何预防网络中出现的常见DDoS攻击？**

**第一，在网络层方面，可以设置让默认的网络管理员的个别IP地址有权限访问服务器的IP地址，即采用IP白名单机制。**

**第二，限制单个 IP 访问频率，防止僵尸主机（Botnet）高频请求，如限制每秒 50 个 HTTP 连接，避免 HTTP Flood 攻击。**

**第三，关闭不必要的服务端口，默认情况下，服务器可能会开放许多端口（如 Telnet、SSH、RDP），但攻击者可利用它们进行入侵或 DDoS 放大攻击。所以可以仅允许必要端口开启，关闭易被攻击的端口，降低被攻击的概率。**

**第四，DDoS 清洗（Scrubbing Center）现在大部分的云服务器如华为云、阿里云等提供 DDoS 流量清洗服务，能实时检测异常流量并丢弃恶意请求。**

**可以在华为云设置规则组，仅允许特定用户访问。**

**4.你对最新的网络安全技术有什么了解？**

**我了解到现在将人工智能（AI）与机器学习（ML）在网络安全中的应用**

**（1）入侵检测与威胁情报分析**

**如：传统的入侵检测系统（IDS）和防火墙主要依靠规则（Rule-based）和签名（Signature-based）方法来检测攻击，但面对未知威胁或变种攻击时，这些方法往往无能为力。AI 和 ML 的引入使得系统能够：**

* **通过 异常行为检测（Anomaly Detection） 发现从未见过的新型攻击（如 APT 攻击）。**
* **结合 威胁情报（Threat Intelligence） 识别恶意 IP、域名、文件哈希值等，提前阻断威胁。**

**🔹 应用示例**

* **深度包检测（DPI）：使用 AI 解析网络流量，判断是否包含恶意代码。**
* **DNS 异常检测：使用 ML 模型检测恶意域名，防止 DGA（域名生成算法）攻击。**

**（2）DDoS 攻击防御**

**分布式拒绝服务（DDoS）攻击通常利用大量僵尸网络对目标服务器发起高流量攻击，ML 可以：**

* **通过流量分析识别异常访问模式。**
* **采用自适应模型动态调整防御策略。**

**🔹 应用示例**

* **Cloudflare 采用 AI 识别 DDoS 攻击流量，自动调整防护策略。**

**🔹 解决方案：**

* **采用联邦学习（Federated Learning），在本地训练 AI，避免数据泄露。**

**（3）误报率与准确性**

**AI 可能会误判正常流量为恶意攻击，导致误报率高。**

**🔹 解决方案：**

* **采用混合模型（AI+规则引擎），提高准确率。**
* **通过持续AI 训练和优化，减少误报。**