第16课 安全优化 SSL的引入

信息安全的 CIA 三要素

CIA 三要素是信息安全的核心原则，分别指 **保密性（Confidentiality）**、**完整性（Integrity）** 和 **可用性（Availability）**。这三个要素共同定义了信息在被存储、传输和处理时的安全属性。

保密性（Confidentiality）

定义

确保信息仅对授权用户或系统可见，防止未经授权的访问或泄露。

实现方法

- **加密**：使用对称加密或非对称加密保护数据（如 AES、RSA）。
- **访问控制**：通过身份验证和权限管理限制数据访问（如用户认证、ACL）。
- **隐私保护**：如数据脱敏、匿名化处理。
- **物理安全**：防止物理介质的非授权访问。

攻击威胁

- **窃听**：网络中截获敏感信息。
- **社会工程学攻击**：如通过钓鱼攻击获取机密信息。
- **数据泄露**：员工失误、系统漏洞或恶意行为导致的敏感数据泄露。

例子

- HTTPS 通过加密保护传输的数据，确保保密性。
- 银行账户密码只有用户和银行知道。

完整性（Integrity）

定义

确保数据在传输、存储和处理过程中没有被未经授权地修改或破坏，保持数据的准确性和一致性。

实现方法

- **哈希校验**：通过哈希算法（如 SHA-256）验证数据是否被篡改。
- **数字签名**：确保数据和发送方身份的真实性。

- **访问控制**：限制对数据修改的权限。
- **日志记录**：记录和追踪数据的变更情况。

攻击威胁

- **数据篡改**：攻击者在传输过程中修改数据。
- **中间人攻击（MITM）**：拦截并更改通信数据。
- **恶意软件**：如勒索软件对数据进行加密或破坏。

例子

- 文件传输后使用哈希值校验文件完整性。
- 银行转账金额在发送和接收时一致，未被篡改。

可用性（Availability）

定义

确保信息和系统在需要时可被授权用户访问和使用，避免因中断或资源耗尽导致服务不可用。

实现方法

- **冗余和备份**：如 RAID 磁盘阵列、数据库备份。
- **容错机制**：如服务器集群、负载均衡。
- **网络防护**：使用防火墙和防 DDoS 攻击设备保护网络资源。
- **定期维护**：如系统补丁更新、漏洞修复。

攻击威胁

- **DDoS 攻击**：通过海量请求耗尽系统资源，导致服务不可用。
- **硬件故障**：服务器或网络设备损坏。
- **自然灾害**：如地震或火灾对数据中心的破坏。

例子

- 在线支付系统在高峰期能够持续响应用户请求。
- 云存储服务通过数据副本和容灾备份确保可用性。

HTTP与HTTPS

HTTP（Hypertext Transfer Protocol）和HTTPS（Hypertext Transfer Protocol Secure）是用于在互联网上传输数据的主要协议，它们之间的主要区别如下：

1. 安全性：
   - HTTP：**HTTP是一个明文传输协议，意味着在客户端与服务器之间传输的数据是未加密的。任何能够访问网络传输路径的人都可以查看或篡改这些数据**。
   - HTTPS：HTTPS是在HTTP的基础上增加了SSL/TLS协议进行加密，提供了端到端的安全通信通道。所**有传输的数据都会被加密**，包括请求头、响应头以及正文内容，从而保护了用户隐私和敏感信息。
2. 身份验证：
   - HTTP：不提供任何形式的身份验证机制，任何人都可以假装服务器发送响应给客户端。
   - **HTTPS：通过SSL证书来验证服务器的身份，确保客户端连接的是真实的、经过认证的服务提供商，而非中间人攻击者。**
3. 信任度：
   - HTTP：使用HTTP的网站通常会在浏览器地址栏显示为“不安全”，这可能影响用户的信任感和网站的信誉。
   - HTTPS：使**用HTTPS并具有有效SSL证书的网站会被浏览器标识为“安全”**，增强用户对网站的信任，并且对于涉及个人隐私和金融交易的网站是必不可少的。
4. 数据完整性：
   - HTTP：由于HTTP协议不提供数据完整性保护，传输过程中数据可能被第三方篡改或注入恶意内容。
   - HTTPS：**SSL/TLS协议通过消息认证码（MAC）和数字签名确保了数据的完整性和一致性**。在数据传输过程中，任何对数据的修改都将导致接收端验证失败。
5. 会话劫持与中间人攻击防护：
   - HTTP：易遭受会话劫持、中间人攻击等安全威胁，攻击者可以截取并操控用户与服务器之间的通信内容。
   - HTTPS：通过加密和证书验证机制有效防止了中间人攻击，使得攻击者无法解读或篡改网络通信内容，极大地提高了网络安全性。
6. 预加载与预连接：
   - HTTP：浏览器通常不会对HTTP站点进行预加载或预连接操作，因为这样做不能保证数据的安全性。
   - HTTPS：现代浏览器支持HTTPS网站的预加载和预连接功能，可以提前建立连接和获取资源，提高网页加载速度。
7. HSTS（HTTP Strict Transport Security）政策：
   - HTTP：不支持HSTS策略，该策略用于强制浏览器始终使用HTTPS与特定域名交互，以防止降级攻击。

- HTTPS：支持HSTS策略，有助于进一步强化安全性，一旦设置后，浏览器将自动拒绝所有非HTTPS链接请求，即使用户手动输入HTTP链接也会被重定向到HTTPS。

8. 性能：
   - HTTP：由于无需执行加密解密过程，HTTP的连接建立速度相对较快，资源加载时间理论上更短。
   - HTTPS：虽然引入了额外的加密握手环节，导致初次建立连接时会有一定延迟，但现代计算机硬件和优化的SSL/TLS实现使得这一开销越来越小。并且，HTTP/2及后续版本在HTTPS上实现了多路复用等性能优化，实际应用中HTTPS在某些情况下反而能提升整体性能。

SSL

什么是 SSL?

SSL（Secure Sockets Layer） 是一种安全协议，用于在网络通信中建立加密连接，确保数据在客户端（如浏览器）和服务器之间传输的**保密性**、**完整性**和**身份验证**。SSL 的目标是保护数据免受窃听和篡改。

目前，**SSL** 已被其升级版本 **TLS（Transport Layer Security）** 所取代，但人们常用“SSL”泛指 SSL 和 TLS 协议。

SSL 的核心功能

1. **数据加密**：
   - 通过对称加密算法，保护数据在传输过程中不被窃听或拦截。
   - 常用加密算法包括 AES、DES 等。
2. **数据完整性**：
   - 使用消息摘要（如 SHA）确保数据在传输中未被篡改。
3. **身份验证**：
   - 通过数字证书验证服务器的真实性，防止用户被欺骗访问伪装的服务器。
   - 在双向认证中，还可以验证客户端的身份。
4. **防止中间人攻击**：
   - SSL 建立安全通信隧道，确保双方直接通信，无第三方干扰。

SSL 的工作原理

SSL 通过以下步骤建立安全连接：

5. **SSL/TLS 握手**

- 客户端和服务器通过握手过程协商加密协议和密钥，建立安全通信。
- 握手主要过程：
  a. **客户端 Hello**：
     - 客户端发送支持的加密算法和协议版本。
  b. **服务器 Hello**：
     - 服务器选择加密算法和协议版本，返回给客户端。
  c. **数字证书验证**：
     - 服务器发送数字证书（包含公钥）给客户端，客户端验证证书的真实性。
  d. **密钥交换**：
     - 客户端生成对称密钥，并通过服务器的公钥加密后发送给服务器。
  e. **会话密钥生成**：
     - 双方使用协商的密钥加密后续通信。

6. **数据传输**

- SSL 使用对称加密保护通信数据。
- 每次传输的数据都会附加消息摘要，用于完整性校验。

SSL 的核心技术

1. **对称加密**：
   - 使用同一个密钥加密和解密数据。
   - 优点：速度快，适合大规模数据传输。
   - 缺点：密钥需要安全分发。
2. **非对称加密**：
   - 使用一对密钥（公钥和私钥）进行加密和解密。
   - 优点：安全性高，适合密钥分发。
   - 缺点：加密速度慢。
3. **数字证书**：
   - 由 CA（证书颁发机构）颁发，包含服务器的身份信息和公钥。
   - 客户端通过证书验证服务器身份。

4. **消息摘要：**
    - 使用哈希算法（如 SHA-256）生成数据的唯一摘要，防止数据被篡改。

数字证书

数字证书是信息安全领域中的一个重要概念，它是一种包含经过加密签名的数据结构，用于证明实体（如个人、组织或服务器）的身份。数字证书通常包含了证书持有者的标识信息（如域名、组织名称等）、公钥、证书的有效期以及其他相关元数据，并且最重要的是，它由一个可信任的第三方权威机构——即证书颁发机构（Certificate Authority, CA）签发和认证。

数字证书的工作原理：

当客户端（例如Web浏览器）与服务器建立安全连接时，服务器会向客户端发送其拥有的数字证书。这个证书中包含了服务器的公钥以及CA对该公钥所有者身份的确认信息。客户端收到证书后，将验证以下几点：

证书完整性：通过检查证书的数字签名来确保证书在传输过程中没有被篡改。

证书链有效性：查看证书是否是由一个受信任的根证书颁发机构直接或者间接签发的。如果服务器提供的不是一个自签名的证书，而是通过一个或多个中间证书颁发机构层层传递的信任关系，那么客户端需要验证从服务器证书到根证书的整个链条，形成一条“信任链”。

证书状态：某些情况下，客户端还会通过在线查询证书撤销列表（CRL）或使用OCSP协议来检查证书是否已被吊销。

信任链的概念：

信任链的核心在于构建起一种信任体系，从最顶层的根证书颁发机构开始，该机构的公钥是预先安装在客户端（如浏览器或操作系统）中的，被认为是绝对可信的。然后，根证书颁发机构可以签发中间证书颁发机构的证书，这些中间证书颁发机构再签发下一级的证书，直至签发最终给服务器或其他实体的终端实体证书。

在验证过程中，客户端沿着证书链向上逐级验证每个证书的有效性和签名，直到找到链顶端的根证书为止。如果整个证书链上的签名和信任关系均有效，则客户端就认为服务器的身份得到了验证，从而建立起对服务器的信任，允许进行安全通信。

简而言之，数字证书为网络通信提供了必要的安全保障，而信任链则是实现这一安全性的关键机制，确保了客户端能够信任与其通信的远程服务器的真实身份。

SSL作用：

- 数据加密：SSL/TLS保证了在网络中传输的数据不被窃听和篡改，保护敏感信息的安全性。

- 身份验证：通过证书验证服务端的身份，避免冒名顶替或中间人攻击。
- 完整性校验：通过散列算法和消息验证码（MAC）确保数据在传输过程中未被修改。

对称加密

定义

对称加密（Symmetric Encryption）使用**相同的密钥**进行数据的加密和解密。发送方使用密钥加密数据，接收方使用相同的密钥解密数据。

特点

1. **单密钥**：加密和解密使用的是同一个密钥。
2. **加解密速度快**：算法计算复杂度低，适合大数据量的加密。
3. **密钥分发问题**：需要通过安全通道分发密钥，否则密钥泄露会导致安全性下降。

非对称加密

定义

非对称加密（Asymmetric Encryption）使用**一对密钥**（公钥和私钥）：

- **公钥**用于加密，任何人都可以使用。
- **私钥**用于解密，只有持有私钥的人才能解密密文。

特点

1. **双密钥**：密钥分为公钥和私钥，互为匹配。
2. **加密速度较慢**：计算复杂度高，不适合大规模数据加密。
3. **高安全性**：无需共享私钥，安全性更高。

常见非对称加密算法

- **RSA**：
  - 最广泛使用的非对称加密算法，基于大整数分解的数学难题。
- **ECC（椭圆曲线加密）**：
  - 提供与 RSA 相当的安全性，但密钥长度更短，性能更高。
- **DSA（数字签名算法）**：
  - 用于数字签名，验证数据来源和完整性。

会话重用与会话恢复：

在SSL/TLS协议中，为了优化性能并减少网络延迟，引入了会话恢复机制。当客户端和服务端首次完成握手并成功建立安全连接后，可以将这次连接期间生成的一些参数（如会话密钥、加密套件等）保存下来，以便后续连接时复用。

1. 会话标识符(Session ID)：
    - 服务器在第一次握手过程中为每个成功的连接分配一个唯一的会话ID，并将其发送给客户端。
    - 客户端在发起新的连接请求时，可以在ClientHello消息中携带这个会话ID。
    - 如果服务器能够识别并验证该会话ID的有效性，则可以跳过完整的握手过程，直接使用之前协商过的参数继续通信，从而减少握手所需的时间和计算资源。
2. 会话Ticket(Ticket-Based Session Resumption)：
    - 在TLS 1.3及以前版本的TLS协议中，会话恢复还可以通过Session Ticket方式进行。
    - 服务器在结束会话时，会加密一组会话参数并将它作为Session Ticket发给客户端存储。
    - 当客户端再次连接时，提交这个Ticket而不是Session ID。
    - 服务器解密Ticket以恢复会话信息，然后如同Session ID方式一样跳过部分握手步骤。
    - TLS 1.3 特别优化了这一过程，允许在某些情况下实现0-RTT(Zero Round-Trip Time)连接，即客户端在初始数据传输的同时完成会话恢复。

密钥交换算法及其对会话恢复的影响：

密钥交换算法用于在不安全的网络环境中安全地交换加密通信所需的密钥。它确保通信双方能够生成共享的会话密钥，而无需在传输过程中暴露该密钥。

常见的密钥交换算法

1. **RSA（Rivest-Shamir-Adleman）**

- **特点**：基于大数分解问题，广泛使用于SSL/TLS握手。
- **优点**：成熟、安全性高。
- **缺点**：计算量大，性能较低。

2. **Diffie-Hellman（DH）**

- **特点**：基于离散对数问题，允许双方在不共享秘密的情况下生成共享密钥。
- **优点**：简单、有效，支持前向保密（Perfect Forward Secrecy）。
- **缺点**：需要高质量的随机数生成。

3. **Elliptic Curve Diffie-Hellman（ECDH）**

- **特点**：基于椭圆曲线数学，提供与DH相同的功能，但密钥长度更短。
- **优点**：高安全性，计算效率高，支持前向保密。
- **缺点**：实现复杂度较高。

4. **Finite Field Diffie-Hellman（FFDHE）**

- **特点**：DH算法的一种变种，使用有限域数学。
- **优点**：适用于不同安全级别和性能需求。
- **缺点**：仍受限于传统DH算法的缺点，如需要高质量随机数。

5. **ChaCha20-Poly1305**

- **特点**：结合了ChaCha20流密码和Poly1305消息认证码的加密算法。
- **优点**：高性能，适用于移动设备，抗侧信道攻击。
- **缺点**：相对较新，尚未完全普及。

RSA算法的基本原理：

RSA算法是一种非对称加密算法，由Ron Rivest、Adi Shamir和Len Adleman在1977年发明，因此得名RSA（三人姓氏首字母组合）。该算法是目前最常用的公钥密码体制之一，在信息安全领域有着广泛的应用，包括数据加密、数字签名以及密钥交换等。

1. **密钥生成**：
   - 首先选择两个大素数p和q，并计算它们的乘积n=p*q。n作为公开模数。
   - 计算欧拉函数φ(n) = (p-1)(q-1)，它是小于n且与n互质的整数的数量。
   - 选择一个整数e，满足1 < e < φ(n)，且e与φ(n)互质。
   - 找到e对于φ(n)的模反元素d，即满足e * d ≡ 1 (mod φ(n))，这意味着ed - k * φ(n) = 1，其中k是某个整数。
   - 公钥由{n, e}组成，私钥由{n, d}组成。

2. **加密过程**：

- 发送方使用接收方的公钥（n, e）来加密消息m。加密公式为：C ≡ m^e (mod n)。这里的C是密文。

3. **解密过程**：
   - 接收方收到密文后，使用自己的私钥（n, d）进行解密。解密公式为：m ≡ C^d (mod n)。通过这种方法可以恢复原始消息m。

RSA的安全性基于大数分解难题，即给定两个大素数p和q构成的大数n（已知），找到这两个素数是极其困难的。只有知道p和q才能计算出φ(n)进而确定d，所以即使公钥被截获，只要足够大的素数因子难以分解，密文也就无法破解。

注意事项：

- 实际应用中，选取的p和q应当足够大以保证安全性，并且通常需要随机生成以防止攻击者通过特定方法猜解。
- 在计算过程中要确保足够的强度，例如使用合适的素数长度（如至少2048位或更长）和足够大的公钥指数e（常选用65537）。
- RSA不适合加密大量数据，因为它相对慢于对称加密算法。实际应用中往往结合对称加密算法，用RSA加密对称密钥，然后用对称密钥加密数据内容。

数字签名

数字签名是一种用于验证电子文档完整性和来源的技术，它是基于密码学原理实现的，确保数据在传输过程中不被篡改，并能确认发送者的身份。

工作原理：

1. 生成摘要（哈希）： 在数字签名中，首先对原始信息使用一个单向散列函数（如SHA-256）生成固定长度的摘要或指纹。这个摘要代表了原始信息的唯一特征。
2. 私钥加密摘要： 发送者用自己的私钥对生成的信息摘要进行加密。私钥是只有发送者自己知道的秘密密钥，它与对应的公钥成对出现。
3. 附加签名： 将加密后的摘要附加到原始信息上，形成带有数字签名的消息。接收方收到消息后，可以从中分离出数字签名部分。
4. 验证签名： 接收者使用发送者的公钥来解密数字签名，得到原始信息的摘要。然后，接收者也独立计算接收到的原始信息的摘要。如果两个摘要一致，则说明信息未被篡改且确实来自拥有对应私钥的发送者。

数字签名的作用：

- 认证身份：通过验证数字签名，接收者可以确定信息确实是发送者发出的。
- 保证完整性：任何对原始信息的更改都将导致新的摘要值，因此，如果摘要匹配，证明信息从发送到接收的过程中没有被改变。
- 防止抵赖：由于只有拥有私钥的实体才能创建有效的数字签名，一旦发送方对其签名的数据进行了数字签名，他们无法否认曾发送过该信息。

互联网安全优化面试常问问题及答案

问题1：什么是HTTPS？它如何工作？

回答：

HTTPS是在HTTP基础上加入SSL/TLS加密层的协议，用于安全地传输数据。通过加密、认证和数据完整性，确保通信的机密性、真实性和完整性。工作流程包括SSL/TLS握手、密钥交换和加密数据传输。

问题2：解释SSL/TLS握手过程的主要步骤。

回答：

1. **客户端Hello**：发送支持的协议版本和加密算法。
2. **服务器Hello**：选择协议版本和加密算法，发送数字证书。
3. **密钥交换**：双方协商会话密钥，如使用Diffie-Hellman。
4. **握手完成**：确认密钥交换成功，开始加密通信。

问题3：什么是中间人攻击（MITM），如何防范？

回答：

中间人攻击是攻击者在客户端和服务器之间拦截和篡改通信内容。防范方法包括：

- 使用HTTPS加密通信。
- 验证服务器的数字证书。
- 启用证书钉扎（Certificate Pinning）。
- 使用强大的加密算法和协议。

问题4：解释公钥加密和对称加密的区别。

回答：

- **公钥加密**：

- 使用一对密钥（公钥和私钥）。
- 公钥用于加密，私钥用于解密。
- 支持身份认证和数字签名。

- **对称加密**：

- 使用相同的密钥进行加密和解密。
- 高效，适用于大规模数据加密。
- 需要安全的密钥分发机制。

问题5：什么是数字证书？它的作用是什么？

回答：

数字证书是由受信任的证书颁发机构（CA）签发的文件，用于验证实体（如服务器、用户）的身份。它包含公钥、持有者信息和CA的数字签名，确保公钥的真实性和所有者的身份。

接收方可以通过 CA 的公钥来验证数字签名。因为 CA 的公钥是公开的，并且 CA 是被信任的第三方机构。如果验证通过，就说明证书内容没有被篡改，并且可以信任证书中包含的公钥确实属于证书所标识的用户

问题6：如何优化SSL/TLS性能？

回答：

- 启用会话复用，减少握手次数。
- 使用高效的加密算法，如ChaCha20-Poly1305。
- 启用HTTP/2，利用其性能优化特性。
- 使用硬件加速，如SSL加速卡。
- 启用OCSP Stapling，减少证书验证延迟。
- 使用最新的TLS版本（如TLS 1.3），简化握手过程。

问题7：什么是SSL剥离攻击（SSL Stripping）？如何防范？

回答：

SSL剥离攻击是攻击者将HTTPS连接降级为HTTP，拦截和篡改通信内容。防范方法包括：

- 使用HTTP Strict Transport Security（HSTS）强制客户端使用HTTPS。
- 禁用HTTP，确保所有请求通过HTTPS。
- 实施证书钉扎，防止证书被伪造。

问题8：解释什么是TLS 1.3，它有哪些改进？

回答：

TLS 1.3是最新的TLS协议版本，具有以下改进：

- **简化握手过程**：减少握手阶段的往返次数，降低延迟。

- **强制使用前向保密**：所有加密套件默认支持前向保密。

- **去除弱加密算法**：只支持安全、高效的加密算法。

- **提高安全性**：减少协议复杂性，降低潜在漏洞风险。

问题9：什么是数字签名，如何在SSL/TLS中使用？

回答：

数字签名是用发送方的私钥对消息摘要进行加密，接收方使用发送方的公钥解密并验证消息的完整性和发送方的身份。在SSL/TLS中，数字签名用于验证服务器证书的真实性，确保客户端与真正的服务器通信。

问题9：解释什么是TLS握手中的“0-RTT”数据传输。

回答：

“0-RTT”（Zero Round Trip Time）数据传输允许客户端在TLS握手的初始阶段发送数据，无需等待服务器的响应。这减少了延迟，提高了性能，但可能引入重放攻击风险。TLS 1.3通过限制0-RTT数据的使用范围来平衡性能和安全性。
