计算机和网络安全课程笔记 Week 1

Nite

2025-03-19T12:18:00+11:00

信息安全基础概念 (Introduction to Security Basic Concepts)

安全现状与影响 (Security Headlines & Impacts)

- 网络攻击是当前世界面临的重要问题。
- 联合国专家正在调查涉及价值 **30 亿美元**的 58 起网络攻击。
- 2017 年至 2023 年间，针对加密货币相关公司的网络攻击价值约为 **30 亿美元**。
- 政府部门也面临数据泄露风险，例如美国政府数据泄露被认为与企业软件制造商 Atlassian 的协作套件 Confluence 中的一个错误（bug）有关。
- 软件漏洞（vulnerabilities）是攻击者利用的常见方式。微软曾在一个补丁中修复了 80 个漏洞。例如，某个漏洞可能允许攻击者绕过 Office 的“受保护视图”（Protected View），直接以编辑模式打开文件，而不是受保护模式。
- 我们生活在由全球互联网连接和高速无线技术驱动的数字信息时代。
- 数字化转型已经影响了各个领域的关键基础设施资产（critical infrastructure assets），包括能源、银行、医疗保健、食品、水和交通等。
- 物联网（Internet of Things - IoT）正在快速发展。

信息安全的主要目标 (Main Aim of Information Security)

- 信息安全本质上是一场资源博弈（resources game）。
- 你的投入是 X，目的是让对手为了达到他们不希望发生的事情（即攻击成功）而必须投入 Y。
- 目标是让对手的投入 Y 远大于你的投入 X。对手往往不止一个。
- 这涉及到时间（Time）、金钱（Money）和计算能力（Computational Power）的资源较量。计算能力可以理解为时间乘以金钱。
- 这意味着：
  - 只要有足够的资源，总会有人能够成功入侵系统。
  - 只要有足够多的攻击者，总会有人能够成功入侵系统。
  - 只要有足够的时间，总会有人能够成功入侵系统。
  - 因此，**所有系统都可能失败，并且最终都会失败**。
- 信息安全的目标是将安全级别提升到对你试图保护的资产来说**足够（adequacy）**的程度。这里的“安全”（security）也隐含着“不安全”（insecurity）的概念。

安全基本原则 (Basic Principles of Security)

信息安全通常关注以下核心原则：

- **认证性（Authenticity）**：证明消息的来源。这包括消息的完整性（integrity）和新鲜性（freshness），即消息不是重放（replay）。
- **机密性（Confidentiality）**：在一段时间 t 内保持消息秘密的能力。
- **完整性（Integrity）**：消息在传输过程中不能被修改。攻击者不能替换伪造的消息。
- **不可否认性（Non-repudiation）**：发送者不能否认已经发送了消息。这与认证性相关。
- **可用性（Availability）**：保证服务质量（quality of service），通常通过容错（fault tolerance）来实现。
- **隐蔽性（Covertness）**：消息存在性的秘密，这与匿名性（anonymity）相关。

其他定义 (Other definitions)

- **保密性（Secrecy）**：一个技术术语，指限制信息访问的行动效果。
- **隐私（Privacy）**：保护你或你家人的个人秘密的能力和/或权利，包括你的个人空间。
- **匿名性（Anonymity）**：保持消息来源/目的地保密的能力/愿望。

信任 (Trust)

- 一个被信任的系统（trusted system）是指其失败可能破坏安全策略的系统。
- 问题是：你能信任谁？

系统失败的原因 (Reasons for System Failure)

系统失败常常是因为设计者：

- 保护了错误的东西。
- 用错误的方式保护了正确的东西。
- 对他们的系统做了错误的假设。
- 未能正确理解威胁模型（threat model）。
- 未能考虑到范式转移（paradigm shifts），例如互联网的出现。
- 未能理解其系统的范围（scope）。

此外，还有其他原因导致系统不安全：

- 软件供应商可能不会修复软件中的所有错误（bugs）。
- 互联网协议（Internet protocols）在设计时并非都是安全的。
- 管理者可能没有投入足够的资源用于安全。
- 员工和用户可能不关心安全，选择弱密码，使用错误的配置，容易受到社会工程威胁（social engineering threats），例如网络钓鱼（phishing）。人类往往是**最薄弱的环节（weakest link）**。
- 加密密钥交换算法（Cryptographic key exchange algorithms）可能容易受到能够分解大数的量子计算机（quantum computers）的攻击。

Kerckhoffs 原理 (Kerckhoffs' s Principle)

- **通过模糊性实现安全（Security through obscurity）**是无效的。
- 安全算法和系统的机制（除了秘密密钥材料）应完全公开。
- **Kerckhoffs 原理：对于一个真正安全的系统，所有秘密必须仅存在于密钥（key）中**。
- 如果算法是公开的，但无法被破解，那么这个系统是好的系统。
- 如果一个算法是秘密的，并且没有人研究过它，那么无法断言其安全性。

不同系统的安全需求 (Different Systems have Different Security Requirements)

不同的系统有不同的安全需求侧重点：

- **银行系统**：
  - 记账系统（Bookkeeping system）：最高级别的**完整性（integrity）**。
  - ATM：**物理安全（physical security）**，客户**认证（authentication）**。
  - 网上银行（Internet Banking）：**认证（authentication）**，**可用性（availability）**。
- **军事系统**：
  - 军事通信（Military communications）：**隐蔽性（covertness）**，**可用性（availability）**，**机密性（confidentiality）**。
  - 分层隔离（Compartmentalisation）：**机密性（confidentiality）**，**可用性（availability）**，抵抗流量分析（traffic analysis）。
- **医院系统**：
  - 在线数据库（Online database）：**完整性（integrity）**，**隐私（privacy）**。
  - 远程医疗（Telehealth）：**认证（authentication）**，**机密性（confidentiality）**，**可用性（availability）**。

系统安全 (Securing Systems)

一个“系统”可以非常广泛，包括：

- 一个产品或组件，例如软件程序、加密协议、智能卡。
- 基础设施，例如 PC、操作系统、通信。
- 应用，例如网络服务器、工资系统。
- IT 员工。
- 用户和管理层。
- 客户和外部用户。
- 合作伙伴、供应商。
- 法律、媒体、竞争对手、政治家、监管机构等外部因素。

攻击类型 (Passive and Active Attacks)

网络攻击可以分为被动攻击和主动攻击。

- **被动攻击（Passive attacks）**：不涉及数据的修改或伪造。攻击者只是监听和收集信息。
  - 例子包括：窃听通信（eavesdropping），消息拦截和信息泄露（message interception and release），流量分析（traffic analysis）。
- **主动攻击（Active attacks）**：攻击者试图修改、破坏或伪造数据/资源。
  - **伪造（Fabrication）**：未经授权的一方将伪造的对象插入系统。例子包括冒充（masquerading）合法实体以获取系统访问权。
  - **中断（Interruption）**：系统资产被破坏或变得不可用/无法使用。例子包括对网络的拒绝服务攻击（denial-of-service attacks）。
  - **修改（Modification）**：未经授权的一方不仅获取访问权，还篡改资产。例子包括改变数据文件中的值或病毒攻击。

风险评估 (Assessing Risks)

- 系统失败的原因常常与风险评估不当有关（见上文系统失败原因）。
- 风险评估通常会使用工具或框架，例如风险评估矩阵（Risk Assessment Matrix）。

总结 (Summary)

- 信息安全是一场资源博弈。
- 所有系统都有漏洞（buggy），系统越大，漏洞越多。
- 没有事物是孤立运作的。
- 了解你的系统，了解你的威胁模型。
- 没有什么是绝对安全的，所有系统都可能失败，并且最终都会失败。
- 人类往往是**最薄弱的环节**。
- 打破一个系统比使其安全容易得多。证明一个失败案例足以声称系统是不安全的。而抵抗大量的攻击并不能证明系统是安全的。

密码学 (Cryptography)

引言 (Introduction)

- 密码学（Cryptography）是研究与密码（ciphers）设计相关的数学技术的学科。
- 密码学的基础构件（building blocks）也称为**密码学原语（cryptographic primitives）**。
  - 例子包括：哈希函数（hash functions），分组密码（block ciphers），流密码（stream ciphers），数字签名（digital signatures），随机数生成器（random number generators）。

基本概念 (Basic Concepts)

- 函数 $f : X \to Y$ 定义如下：
  - 定义域（Domain）：集合 $X = \{x_1, x_2, \dots, x_n\}$。
  - 值域（Codomain）：集合 $Y = \{y_1, y_2, \dots, y_m\}$。
- 对于函数 $f : X \to Y$：
  - $x \in X$ 的像（image）称为 $f(x)$，是 Y 的一个元素。
  - 函数 f 的范围（range）是所有像的集合，是 Y 的一个子集。
  - 如果 $f(x) = y$，则 $x = f^{-1}(y)$ 称为 y 的一个原像（preimage）。
- 例如：设 $f : \{-1, 0, 1\} \to \{0, 1, 2\}$ 定义为 $f(x) = x^2$。
  - $f(-1) = 1, f(0) = 0, f(1) = 1$。
  - 1 的原像是 $f^{-1}(\{1\}) = \{-1, 1\}$。
  - 2 的原像是 $f^{-1}(\{2\}) = \{\}$。
  - f 的范围是 $\{0, 1\}$。

单向函数 (One Way Functions - OWF)

- 考虑所有长度为 n 的二进制字符串的集合，即 $\{0, 1\}^n$。例如：
  - $\{0, 1\}^1 = \{0, 1\}$。
  - $\{0, 1\}^2 = \{00, 01, 10, 11\}$。
- 一个函数 $f : \{0, 1\}^n \to \{0, 1\}^m$ 是一个**单向函数（One-Way Function - OWF）**，如果：
  - 对于所有 $x \in X$，计算 $f(x)$ 是“容易的”（easy）。

  - 找到一个原像（preimage）是“计算不可行的”（computationally infeasible）。
- 直观地说：
  - 给定 x，计算 $f(x)$ 很容易。
  - 给定 $f(x)$，计算 x 很难。
- **碎盘子例子** (Smashed Plate Example)：
  - 在盘子侧面写一条消息：x。
  - 把盘子砸碎：$f(x)$。
  - 找到逆函数 $f^{-1}(x)$ 是困难的（但不是不可能的）。
- 数据加密标准（Data Encryption Standard - DES）也可以看作一个例子：
  - $f(x) = \text{DES}(m, k) = c$。
  - 给定密文 c，找到密钥 k 和明文 m 是困难的。

哈希函数 (Hash Functions)

- 哈希函数（Hash function），记作 h，是一种将任意长字符串有效地映射到固定短长度字符串的计算。
- 哈希函数的最小属性：
  - **压缩性（Compression）**：将任意数量的比特映射到固定短长度的比特串。通常输出长度小于等于 512 比特。常见的哈希函数包括 MD5, SHA256, SHA512。
  - **易计算性（Ease of computation）**：给定 h 和 x，计算 $h(x)$ 通常是容易的。
- **密钥哈希函数（Keyed Hash Functions）**：有些哈希函数既需要密钥（k）也需要消息（m）作为输入。
  - 记作 $\text{MAC}_k(m) = h(m, k)$。
  - 它们也被称为**消息认证码（Message Authentication Codes - MAC）**或基于哈希的消息认证码（hash-based message authentication codes - HMAC）。MACs 的密钥是秘密的，用于验证哈希值 $\text{MAC}_k(m)$。它可被视为一个加密校验和（cryptographic checksum）。

安全哈希函数的性质 (Properties of Secure Hash Functions)

设 $h : X \to Y$ 是一个哈希函数。为了安全，它必须满足以下性质：

- **#1 原像抵抗性（Preimage Resistance）**：
  - 给定 y，找到一个原像 x 使得 $h(x) = y$ 是“困难的”（hard）。
- **#2 第二原像抵抗性（Second Preimage Resistance）**：
  - 给定一个特定的 x（因此也知道 $y = h(x)$），找到另一个不同的 $x' \neq x$ 使得 $h(x') = h(x) = y$ 是“困难的”。
- **#3 抗碰撞性（Collision Resistance）**：
  - 找到任意一对不同的输入 $x \neq x'$ 使得 $h(x) = h(x')$ 是“困难的”。
  - 注意：#3 抗碰撞性蕴含（implies）#2 第二原像抵抗性。因为如果 #2 不成立（即容易找到 x'），那么 #3 也不成立。
- 一个单向哈希函数（one-way hash function）满足 #1 和 #2。
- 一个抗碰撞哈希函数（collision resistant hash function）满足 #3（因此也满足 #2）。

哈希函数的应用 (Uses of Secure Hash Functions)

哈希函数在无需透露所知秘密的情况下进行知识确认（confirmation of knowledge）方面非常有用。

- 例如，无需通过互联网发送秘密信息给 Alice，只需发送该秘密信息的哈希值。如果 Alice 知道该秘密，她可以计算其哈希值并验证你是否也知道这个秘密。只要哈希函数是强的，这比发送秘密信息本身要安全高效得多，因为后者可能被截获。

- **应用 1：密码文件（Password Files）**
  - 在服务器上存储用户密码的哈希值。当用户尝试登录时，计算输入的密码的哈希值，并与存储的哈希值进行比较。如果密码文件被盗，攻击者需要对哈希值进行逆向（即“破解密码”），然后才能使用这些密码。
- **应用 2：病毒防护和主机入侵检测（Virus Protection & Host Intrusion Detection - HIDS）**
  - 为每个文件 x 计算 $h(x)$ 并存储在一个远离系统的位置（off system）。定期重新计算所有文件的哈希值并检查是否匹配存储的哈希值。
  - 这里的**第二原像抵抗性（#2）**是至关重要的。攻击者应难以找到一个文件 x' 使得 $h(x) = h(x')$（即病毒或恶意软件 x' 具有与原始文件 x 相同的哈希值），否则病毒可以隐藏起来不被发现。

实际哈希函数 (Real World Hash Functions)

- MD5, SHA-1, SHA256, SHA512 是常见的哈希函数。它们产生固定长度的输出。
- 例如，字符串“ELEC5616”经过不同哈希函数会产生不同的固定长度输出。
  - MD5 (128 比特)
  - SHA-1 (160 比特)
  - SHA256 (256 比特)
  - SHA512 (512 比特)

哈希函数攻击 (Hash Function Attacks)

- **原像抵抗性攻击（Pre-Image Resistance Attack）**：
  - 暴力攻击（brute force）在密码分析中是指搜索所有可能的替代方案。字典攻击（dictionary attack）是其中的一个子集，攻击者使用预先计算好的密码/哈希对列表进行尝试。
  - 可以使用暴力攻击来攻击原像抵抗性：假设哈希函数产生一个 n 比特的输出 $y = h(x)$。为了找到原像 x，平均需要尝试 2^{n-1} 次哈希计算才能找到一个 a 使得 $\Pr\{h(a) = y\} \geq 0.5$。直观上，如果秘密密钥在 $2^{10} = 1024$ 个“盒子”中的一个，平均需要打开一半 ($2^9 = 512$) 才能找到它。
- **抗碰撞性攻击（Collision Resistance Attacks）**：
  - 生日攻击（birthday attack）是一种针对抗碰撞性的攻击。
  - 生日悖论（Birthday problem）示例：在一个房间里需要多少人才能使其中两人拥有相同的生日？概率超过 0.5 只需要 23 人。
  - 对于一个 n 比特的哈希函数，平均需要尝试 $2^{n/2}$ 次随机消息的哈希计算才能使碰撞攻击成功。这是因为生日悖论的原理，找到两个随机输入具有相同输出的概率比找到特定输出对应的特定输入的概率要高得多。
  - **SHAttered** 是一个著名的 SHA-1 碰撞攻击实例。Google 在 2017 年宣布了首次 SHA-1 碰撞。

哈希函数构造 (Hash Function Construction)

- **Merkle–Damgård 构造**用于构建 MD5, SHA-1 和 SHA-2。
  - 它使用一个初始化向量（IV）和一个单向压缩函数（one-way compression function，f）。
  - 消息 M 被分割成 n 个 r 比特的块。
  - 消息长度必须是固定数值（例如 512 比特）的倍数，因此消息必须首先通过**填充函数（padding function）**进行填充。
- **Sponge 构造**用于构建 SHA-3。

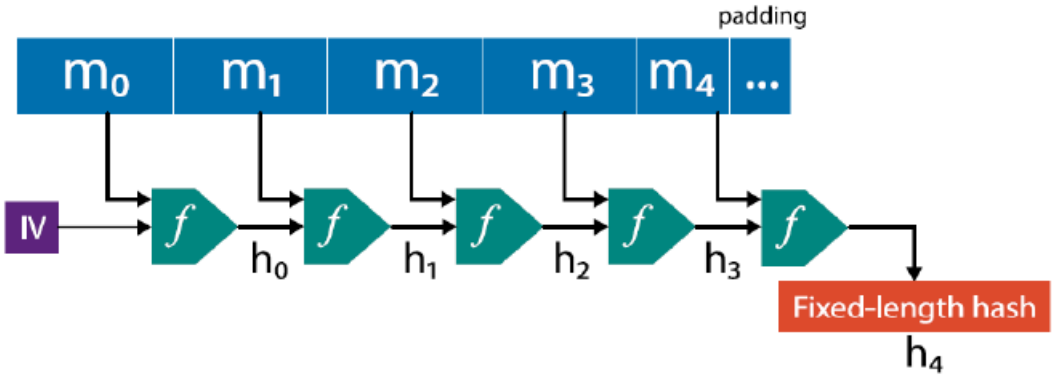

图 1: Merkle–Damgård 构造

消息认证码 (Message Authentication Codes - MACs)

- 消息认证码（MACs）使用密钥（k）和一个哈希函数（可以是单向哈希函数）。
- MAC 是一个密钥哈希函数 $h_k : \{0,1\}^* \to \{0,1\}^n$。密钥是秘密的，用于验证哈希值 $h_k(m)$。
- MAC 的**目标（Goal）**是：
  - 在发送方和接收方共享秘密密钥时，提供**消息认证（message authentication）**。
  - 窃听者（eavesdropper）无法伪造具有有效 MAC 的消息。
  - MAC 用于**消息完整性（message integrity）**，但**不用于消息秘密性（message secrecy）**。

MACs 的属性 (Properties of MACs)

- 给定消息 m 和密钥 k，构造 $h_k(m)$ 是容易的。
- 给定消息-MAC 对 $(m_x, h_k(m_x))$，在不知道密钥 k 的情况下，构造一个新的有效对 $(m_y, h_k(m_y))$，其中 $m_x \neq m_y$，是困难的。
- **示例 1**：
  - Alice 和 Bob 共享秘密密钥 k。
  - 攻击者无法发送带有有效 MAC 的消息，因为攻击者不知道 k 来计算 MAC$(m) = h_k(m)$。
- **示例 2**：
  - 如果哈希函数用于病毒防护，并将每个文件的签名（即哈希值）存储在数据库中。病毒是否也能修改数据库呢?
  - 使用 MAC 时，病毒无法（自己）修改数据库并计算出有效的 MAC，因为它不知道密钥 k。
  - 但是，如果病毒拥有写权限，它仍然可以破坏数据库或用特洛伊木马/伪造程序替换验证程序。

HMAC (Hash Based MAC)

HMAC 是基于非密钥哈希函数 h 构建的 MAC。

- **尝试 1**：MAC$_k(m) = h(k|m)$。
  - 这是不安全的。由于 Merkle-Damgård 构造的特性，攻击者可以在消息末尾任意添加内容（padding）并计算出新的有效 MAC。
- **尝试 2**：MAC$_k(m) = h(m|k)$。
  - 这是不安全的。容易受到生日攻击。
- **尝试 3**：MAC$_{k,k'}(m) = h(k|m|k')$。
  - 这种包络方法（enveloped method）更安全。
- **最佳方法** (BEST)：MAC$_k(m) = h(k \oplus \text{opad}|h(k \oplus \text{ipad}|m))$。

- opad 是外部填充（outer padding），一个块的十六进制常量 0x5c5c5c5…。
- ipad 是内部填充（inner padding），一个块的十六进制常量 0x3636363…。
- 这一定义在 RFC 2104 中有详细说明。

CBC-MAC (Cipher-Based MAC)

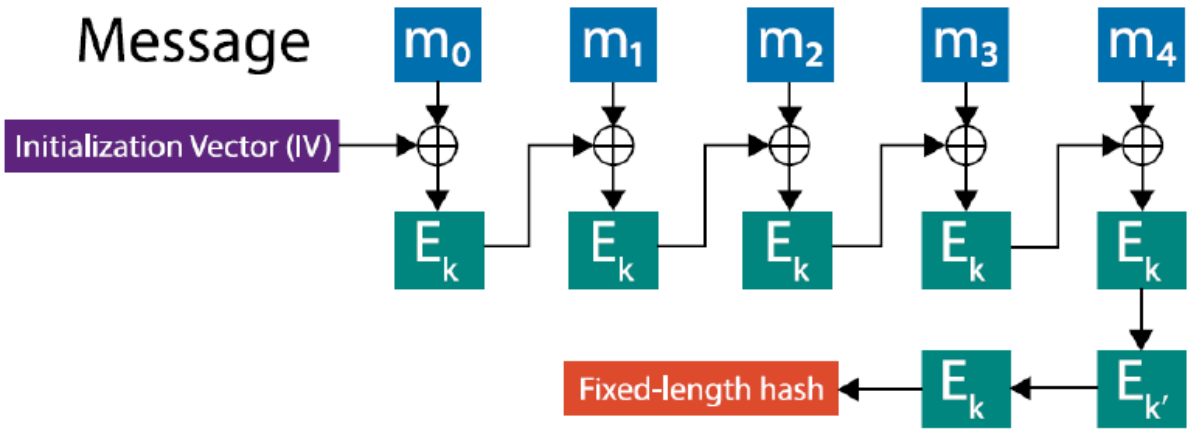

图 2: CBC-MAC

- CBC-MAC 使用一种称为**密文分组链接（Cipher Block Chaining - CBC）**的技术。
- 它将消息分割成块。
- 使用一个分组密码（block cipher）进行重复加密，并将结果进行异或（⊕）。
- 通常使用一个秘密密钥 k 和一个初始化向量（IV），IV 通常是一些随机比特。
- 如果 E 是一个 MAC，那么 CBC-E 也是一个 MAC。CBC-MAC 在银行业务中经常使用。

可能的考点与示例考题 (Possible Exam Points & Sample Questions)

- **信息安全的三要素（机密性、完整性、可用性）**是基础概念，需要理解和区分。
- **消息认证码（MAC）**和**哈希函数**的定义、属性和用途是重要考点。
- **安全哈希函数的三大性质（原像抵抗性、第二原像抵抗性、抗碰撞性）**及其相互关系是核心知识。
- **单向函数（OWF）**的概念和与哈希函数的关系。
- **主动攻击与被动攻击**的定义及示例。
- **Kerckhoffs 原理**及其含义。
- 了解不同应用场景（如银行、军事、医院）对安全的**不同需求侧重**。
- 知道**暴力攻击**和**生日攻击**是攻击哈希函数的常见方法。
- 了解**哈希函数的主要用途**（密码存储、病毒检测/完整性校验）。
- 知道**人类是最薄弱的环节**以及导致系统不安全的其他因素。
- 理解信息安全是**资源博弈，没有绝对安全**的系统。

示例考题（及参考答案要点）：

1. **什么是安全哈希函数？请列举其主要安全性质并解释其含义。**
   - 参考答案：安全哈希函数是一种将任意长输入映射为固定短输出，且满足特定安全性质的函数。主要性质包括：**原像抵抗性**（难以从输出逆推输入）、**第二原像抵抗性**（给定一个输入和输出，难以找到另一个不同的输入产生相同的输出）、**抗碰撞性**（难以找到任意两个不同的输入产生相同的输出）。

2. **安全哈希函数主要用于哪些方面？请举例说明。**
    - 参考答案：主要用于确认知识而无需透露秘密。例如，**密码存储**（存储密码的哈希值而非明文）和**文件完整性校验/病毒防护**（存储文件的哈希值，定期检查文件是否被篡改或替换）。
3. **什么是暴力攻击（Brute Force Attack）？如何通过增加哈希输出长度来抵抗针对哈希函数的暴力攻击？**
    - 参考答案：暴力攻击是指尝试所有可能的输入组合来找到目标。针对哈希函数的原像抵抗性进行暴力攻击时，如果哈希输出是 n 比特，平均需要尝试 2^{n-1} 次。通过增加哈希输出长度 n，攻击所需的计算量呈指数级增长（2^n），从而使其计算上不可行。
4. **解释消息认证码（MAC）的概念及其主要作用。MAC 和安全哈希函数有什么区别？**
    - 参考答案：MAC 是一种使用**秘密密钥**的哈希函数，用于提供**消息认证和完整性**。MAC 的作用是确保消息来自拥有共享密钥的合法发送者，并且在传输过程中没有被修改。MAC 与安全哈希函数的主要区别在于 MAC **需要一个秘密密钥**作为输入，而标准的非密钥哈希函数不需要密钥。MAC 提供了认证功能，而仅使用非密钥哈希函数不能提供认证。
5. **为什么说“人类是信息安全中最薄弱的环节”？请举例说明。**
    - 参考答案：系统失败的一个主要原因是人类因素。例如，员工和用户可能不关心安全，选择**弱密码**，使用错误的系统配置，或者容易受到**社会工程攻击**（如网络钓鱼）而泄露敏感信息或执行危险操作。这些行为往往绕过了技术安全控制。