

从互联网到网络空间

ARPANET P2

互联网发展漫话 P8

1. 分组交换 ARPANET 的重要前提：如何将数据信息传遍整个网络且只有接收者才能真正打开这个包，如何在不同的计算机系统间进行通信，让不同的计算机共享信息
2. P17-18 TCP / IP 协议（ARPANET建立的重要前提）
3. P19 互联网体系结构设计基本原则
 - 无连接的分组交换技术
 - 端到端原则
4. P23 Web 让全世界各地计算机进行超文本文档的共享，实现了计算机网络内容互联，开放、共享、可扩展也就意味着数据的安全和隐私更容易发生泄露

网络空间与安全

1. P39 定义
2. P41 四个要素：
 - 计算机(硬件和软件)
 - 数据资源
 - 网络基础设施和通信链路
 - 应用服务
3. P44 元宇宙定义，P46 元宇宙的特性，P47 生态系统与核心技术，P50 安全风险
4. P57 网络空间安全的目标：
 1. 保密性
 2. 可用性
 3. 完整性

网络科学

1. P81 网络的实质：事物 + 联系
2. P88 度分布，P89 平均路径长度，P90 聚合系数，P91 介数
3. P96 小世界特性
4. P107 马太效应与名人效应
5. P116 发现受保护区域边界，切断社区间的关键连接，可以有效隔绝已感染社区，防止病毒扩散

网络空间安全基本原理

第1节 沙箱 P10

沙箱通常用于执行未经测试或者不受信任的程序。恶意程序要对系统进行入侵或者破坏，需要获得文件读、写等必要的操作权限。如果能够对权限进行限制和隔离，就能有效限制恶意程序的破坏能力和范围，沙箱则是为此设计的一种防御机制。

沙箱的核心思想：隔离

第2节 入侵容忍 P19

入侵容忍的安全目标主要是在攻击可能存在的前提下使系统的机密性、完整性和可用性能够得到一定程序的保证。

第3节 可信计算 P29

可信计算的安全目标 P33

可信计算的核心思想：从信任根出发构建信任链 P35

可信计算的核心问题：信任问题。强调从可信根出发解决系统结构中的安全问题，即通过信任链确保每一个环节的身份可信，从而保证从起点的可信根到后续的可信应用的信任关系是可靠的，为计算机系统安全提供一体化的安全保证。

可信计算关键技术概念 P37

第4节 类免疫防御

思想出发点：借鉴生物学中的免疫机制，从而实现计算机系统的安全

类免疫防御的安全目标：P42

计算机系统中的免疫：“识别”+“清除” P43

类免疫防御的实现思路 P44

第5节 移动目标防御

如果系统内部是动态变化的，攻击者还能达成攻击目标吗？

移动目标防御的安全目标 P51

移动目标防御的基本思想：“动态”+“异构” P52

移动目标防御的五個层次 P53

MTD（移动性测试和诊断） P54-55

第6节 拟态防御

防拟防御的安全目标 P61

防拟防御的基本思想：“动态”+“异构”+“冗余” P62-63

防拟安全主动防御体系基础结构 P64

第7节 零信任网络

传统的从外网到内网的边界安全模型依赖于在网络边界进行安全检查，试图把攻击阻挡在边界之外。但内网是否绝对安全？

零信任网络的五个基本假设 P70

零信任网络的核心思想 P71

零信任网络结构 P73

零信任网络控制平面 P74

零信任网络的 7 条基本原则 P75-76

控制论

1. P40 负反馈调节

博弈论

1. P52 智猪博弈
2. P54 ~ 55 博弈要素
3. P58 博弈的鞍点
4. P59 纳什均衡：在包含兩個或以上參與者的非合作博弈中，假設每個參與者都知道其他參與者的均衡策略的情況下，沒有參與者可以透過改變自身策略使自身受益時的一個概念解
5. P66 合作博弈
6. P72 纳什讨价还价

最优化理论

1. P93 凸优化

概率论与随机过程

1. P107 撞库攻击

第3讲 数据加密

密码学简史 P9

密码学是一门历史悠久的学科，大致可以将其发展分为以下三个阶段：

- **古典密码**：主要是通过替换和移位等简单手段来加密信息。
- **近代密码**：包括复杂的机械或电子密码机，如二战期间的德国恩尼格玛密码机。
- **现代密码**：主要指的是计算机时代的密码技术，比如公钥密码和哈希函数等。

隐写术与密码学 P10

隐写术和密码学是信息隐藏领域的两个重要分支。他们都关注如何将信息安全地传递到接收者，但是方法和目的有所不同。

- **隐写术**：源自希腊语，意为“隐秘书写”。它是一门关于信息隐藏的技巧和科学，主要是将信息隐藏在其他无害的信息中，以达到传递秘密信息的目的。
- **密码学**：源自希腊语，意为“隐藏的书写”。它是一门研究如何隐密地传递信息的学科。密码学的目的不仅仅是隐藏信息，而是确保信息的安全性，包括保密性、完整性和可用性等。

密码学者 *Ron Rivest* 的名言：“密码学是关于如何在敌人存在的环境中通讯”。

密码学的基本概念 P12

- **密码编码**：通过信息编码使信息保密。
- **密码分析**：用分析方法解密信息。
- **基本术语**：
 - 明文(plain text)：原始要加密的信息。
 - 密文(cipher text)：经过加密的信息。
 - 加密(encrypt, encryption)：将明文转化为密文的过程。
 - 解密(decrypt, decryption)：将密文转化为明文的过程。
 - 密码算法(Algorithm)、密码(Cipher)：用来加密和解密的数学函数。
 - 密钥(Key)：密码算法中的一个变量。

密码编码主要有以下两种操作：

- **代替**：每个明文元素映射为另一个元素。
- **换位**：明文中的元素重新排列。

针对密钥的数量，有以下两种加密方式：

- **对称加密**：同一个密钥既用于加密又用于解密。
- **非对称加密**：使用一对密钥，一个用于加密，另一个用于解密。

对明文加密的方式主要有：

- **块加密**：一次处理一个数据块进行加密。
- **流加密**：一次处理一个数据单元进行加密。

密码技术的主要用途 P14

- **数据保密**：通过数据加密和解密保护数据的私密性。
- **认证技术**：实体身份认证和数据源发认证。
- **信息完整性保护**：保证信息在传输过程中没有被插入、篡改、重发。
- **数字签名和抗抵赖**：源发抗抵赖和交付抗抵赖。

古典密码 P16

古典密码主要包括：

- **代替密码 (Substitution Cipher)**
- **换位密码 (Transposition Cipher)**
- **代替密码与换位密码的组合**

这些密码体制的安全性主要依赖于保持算法**本身**的保密性。然而，它们都存在以下缺陷：

- 受限算法的缺陷
- 不适合大规模使用
- 不适合人员变动较大的组织
- 用户无法了解算法的安全性

例子：凯撒密码（PPT第17页）

凯撒密码是一种代替密码，它将英文字母向前移动k位。

例子：猪圈密码（PPT第18页）

例子：纵行换位密码（PPT第19页）

纵行换位密码是一种换位密码，明文的字母保持相同，但顺序被打乱了。例如，明文 "Can you understand" 在纵行换位密码中将被转化为 "Codtaueanurnynsd"。这种密码在美国南北战争中得到了广泛的应用。

1949年~1975年：戚继光反切码（PPT第20-21页）

戚继光反切码源自反切拼音注音方法，是用两个字为另一个字注音，取上字的声母和下字的韵母，"切"出另外一个字的读音。例如，明文 "5-25-2" 在戚继光反切码中将被转化为 "敌"。

近代密码 (P.22)

近代密码 的核心是将**算法和密钥分开**。在这种方式中，**密码算法可以公开，密钥保密**，从而将密码系统的安全性放在**保持密钥的保密性**上。

ENIGMA机 (P.23-P.26)

ENIGMA机 是一个由 *Arthur Scherbius* 在1919年发明并在1926年被德军装备的转轮密码机。这个机器使用了一系列的转轮和连接板来创建复杂的加密过程。

- **工作原理** (P.24-P.26): 它的工作方式基于三个核心要素：转子的初始设置，转子之间的相互位置，以及连接板连线的状况，这三者组合构成了所有可能的密钥。此外，由于每个转子的旋转，使得可能的加密变化在每次输入时都会变化。
- **工作过程** (P.27):
 - 根据密码本取得当日密钥
 - 首先发送一个新的密钥
 - 随机地选择三个字母，比如说PGH
 - 把PGH在键盘上连打两遍，加密为比如说KIVBJE（注意两次PGH被加密为不同的形式）
 - 把KIVBJE放在在电文的最前面
 - 重新调整三个转子的初始方向到PGH
 - 正式对明文加密

对称密钥密码系统的缺陷 (P.31)

对称密钥密码系统 在实践中表现出三个主要的缺陷：

1. 密钥必须经过安全的信道分配。
2. 无法用于数字签名。
3. 密钥管理复杂，随着用户数量的增加，密钥的数量将呈指数级增长，这是因为每个用户都需要与每个其他用户有一个唯一的密钥。

现代密码 (P.31)

现代密码的发展在很大程度上归功于**Diffie和Hellman**在1976年的文章《New Direction in Cryptography》中提出的**公钥密码**的概念。这个概念引入了两个密钥的使用，对密钥分配、数字签名、认证等方面产生了深远的影响。此外，他们的方法基于数学函数，而不是代替和换位，被视为密码学历史上唯一的一次真正的革命，实现了密码学发展史上的第二次飞跃。

量子密码 (P.32-P.33)

量子密码是一个新兴的密码体系，它采用量子态作为信息载体，经由量子通道在合法的用户之间传送密钥。它的安全性由量子力学原理所保证。

- **工作原理**：量子密码利用光的偏振状态来编码信息。例如，“水平垂直方向”偏振和“对角方向”偏振可以被用来代表二进制的0和1。
- **量子密钥生成方法**：在量子密码中，发送方生成并发送偏振，接收方则进行测量。由于量子力学的原理，任何未经授权的测量都会改变量子态，从而被检测到。

对称密码

对称密码是使用相同的密钥 ($K_E = K_D$) 进行加密和解密的密码系统。这种系统的密钥需要通过秘密的信道进行分配。常用的对称密钥密码算法包括**DES (Data Encryption Standard)**及其各种变体和**AES (Advanced Encryption Standard)**。

对称密码算法的两条基本设计原则 (P.35)

1. **扩散 (Diffusion)**：重新排列消息中的每一个比特，使明文中的冗余度能够扩散到整个密文，将每一个比特明文的影响尽可能作用到较多的输出密文位中。
2. **扰乱 (Confusion)**：密文和密钥之间的统计特性关系尽可能复杂化。如果密钥的一位发生变化，密文的绝大多数位也发生变化。这两种设计原则是由信息论的创始人克劳德·艾尔伍德·香农提出的。

分组密码 (P.38)

分组密码是每次只能处理特定长度的一块数据的一类密码算法。对不同分组主要有5种迭代模式

分组密码算法只能加密固定长度的分组，但是我们需要加密的明文长度可能会超过分组密码的分组长度，这时就需要对分组密码算法进行迭代，以便将一段很长的明文全部加密。而迭代的方法就称为分组密码的**模式(mode)**。

ECB模式（电子密码模式）(P.39)

在**ECB模式**中，明文分组加密后的结果直接作为密文分组。每个分组的处理相互独立，可以并行操作。一个密文块传输错误不会影响后续密文解密。

但由于分组相互独立，信息块可被替换、重排、删除、重放，对明文的主动攻击是可能的。

CBC模式（密码分组链接模式）(P.40)

CBC模式能够隐藏明文的数据模式，相同的明文可对应不同的密文，能在一定程度上抵抗主动攻击。但加密不支持并行计算。

CFB模式（密文反馈模式）（P.41）

CFB模式的优点包括能够隐藏明文的数据模式，抵抗主动攻击，解密支持并行计算。但加密不支持并行计算，一个密文块传输错误会影响后续密文解密。

OFB模式（输出反馈模式）（P.42）

OFB模式在每次加密时，都会将密码算法的输出反馈至输入中，即上一轮加密算法的输出会作为下一轮加密算法的输入。

CTR模式（计数器模式）（P.43）

DES(年年讲年年烂，建议看课件)

考又考不了，讲又非要讲

DES 的一般设计准则（P. 57）

- 随机性：输出输入无规律
- 雪崩效应：改变输入中的1位，平均导致大约一半的输出位被改变
- 非线性：每个输出位都是所有输入位的复杂函数
- 完全性：加密函数对于任何密钥值都是非线性的
- 相关关系免疫性：输出统计上独立于任何输入位的子集

流密码（P. 58）

分组密码处理完一个分组就结束了，因此不需要通过内部状态来记录加密的进度；相对地，流密码是对一串数据流进行连续处理，因此需要保持内部状态。

一次性密码本（P. 61）

公钥密码

公钥密码可以解决密钥分发问题。主要的方法有：

- **事先共享密钥**：在加密通信之前，以安全的方式将密钥交给接收方。
- **通过密钥分配中心（Key Distribution Center, KDC）**：当参与加密通信的人过多时，可以使用KDC。当需要进行加密通信时，KDC会生成一个通信密钥，每个人只需和KDC事先共享密钥即可。
- **通过Diffie-Hellman密钥交换**：进行加密通信的双方仅需要交换一些信息，即使这些信息被窃听，也无法对算法进行解密，而通信双方则可以通过这些信息各自生成相同的密钥。
- **通过公钥密码**：由于加解密使用不同的密钥，所以无需进行密钥分发。

公钥密码系统的加密原理 (P.64-65)

在公钥密码系统中，每个通信实体都有一对密钥（公钥和私钥）。公钥公开，用于加密和验证签名；私钥保密，用于解密和签名。如果A想要向B发送消息，会使用B的公钥进行加密。B收到密文后，使用自己的私钥进行解密。任何人向B发送信息都可以使用同一个密钥（B的公钥）进行加密，但只有B可以使用自己的私钥进行解密。

公钥密码系统的签名原理 (P.66)

A向B发送消息，使用A的私钥进行加密（也叫做签名）。B收到密文后，使用A的公钥进行解密（也叫做验证）。

公钥密码算法的表示 (P.67)

- 对称密钥密码
 - 密钥：会话密钥(K_s)
 - 加密函数： $E_{K_s}[P]$
 - 对密文C，解密函数： $P=D_{K_s}[C]$
- 公开密钥
 - (K_{Ua} , K_{Ra})
 - 加密/签名： $C = E_{K_{Ub}}[P]$, $E_{K_{Ra}}[P]$
 - 解密/验证： $P = D_{K_{Rb}}[C]$, $D_{K_{Ua}}[C]$

对公开密钥密码算法的要求 (P.68-69)

- 参与方B能容易地生成密钥对(K_{Ub} , K_{Rb})。
- 已知 K_{Ub} ，A的加密操作应该是容易的： $C=E_{K_{Ub}}[P]$ 。
- 已知 K_{Rb} ，B的解密操作应该是容易的： $P=D_{K_{Rb}}(C) = D_{K_{Rb}}(E_{K_{Ub}}(P))$ 。
- 已知 K_{Ub} ，求 K_{Rb} 应该是在计算上不可行的。
- 已知 K_{Ub} 和C，欲恢复P应该是在计算上不可行的。

公钥密码算法的误解 (P.603)

以下列出了三个对公钥密码算法的常见误解：

1. 公开密钥算法比对称密钥密码算法更安全：事实上，任何一种算法的安全性都依赖于密钥长度和破译密码的工作量。从抗分析的角度来看，没有哪一种算法是绝对优越的。
2. 公开密钥算法使对称密钥成为过时的技术：事实上，公开密钥算法运行速度较慢，只能用在密钥管理和数字签名上。因此，对称密钥密码算法仍将长期存在。
3. 使用公开密钥加密，密钥分配非常简单：事实上，密钥分配既不简单也不有效。

RSA算法简介 (P.71)

RSA算法是由Ron Rivest、Adi Shamir和Leonard Adleman开发的一种公钥加密算法。RSA的安全性基于大数分解的难度。RSA已经成为事实上的工业标准，可用于公钥密码和数字签名。

RSA算法的诞生 (P.71)

1977年，Rivest、Shamir和Adleman在MIT合作发表了一篇完整描述RSA算法的论文。这三个人风格迥异，但组成了一个技能互补的完美团队。

RSA算法的性能表现 (P.73, 74)

RSA的安全性主要取决于对大数分解的难度。在实际运行中，RSA的速度比DES慢，无论是在软件还是硬件上的实现。

RSA算法的操作过程 (P.604-608)

RSA算法的操作过程包括密钥产生和加密/解密两个部分。

密钥产生：取两个大素数 p 和 q ，计算 $n=pq$ ，公开 n ；计算欧拉函数 $\phi(n)=(p-1)(q-1)$ ，选取一个与 $\phi(n)$ 互素的整数 e ，使得 $\gcd(e, \phi(n))=1$ ， $1<e<\phi(n)$ ；找到一个整数 d ，使得 $de\equiv 1 \pmod{\phi(n)}$ ，将 d 保密。

加密/解密：公开密钥为 $KU=\{e, n\}$ ，秘密密钥为 $KR=\{d, n\}$ 。加密过程为 $C=M^e \bmod n$ ，解密过程为 $M=C^d \bmod n$ 。

公钥密码系统的应用 (P.77)

- 加密/解密
- 数字签名：发送方用自己的私钥签署报文，接收方用对方的公钥验证对方的签名
- 密钥交换：双方协商会话密钥

RSA Diffie-Hellman DSA 适用场景 (P. 78)

公钥密码系统主要用于加密/解密、数字签名和密钥交换。例如，RSA可以用于这三种应用，而Diffie-Hellman只能用于密钥交换，DSA则只能用于数字签名。

公钥基础设施 PKI P79-105

前面章节我们也看到了，IP地址伪造是非常容易的，由此也带来了各种安全风险。所以我们需要一种能够在互联网上确认身份的机制，特别用户需要准确确认服务器是否真实，尤其是用户需要访问网上银行进行涉及财产安全的操作时，更需要确认正在访问的网站是否是真实的网上银行。

和身份证机制类似，如果存在第三方权威机构，它负责给网络实体颁发身份证明书，在通信时验证身份证明书便可以确认对方的身份了，并且这个权威机构还必须是大家都可以信赖的，能将用户所持有的公开密钥与其身份信息结合在一起，当然用户也要好好保管自己的身份证明书。这个权威机构就是我们常说的CA（Certificate Authority），而整个体系就是公钥基础设施PKI（Public Key Infrastructure）。

PKI: Public Key Infrastructure，公钥基础设施：由 CA，数字证书等组成的公钥与身份验证体系

公钥基础设施PKI的核心组件：数字证书

PKI的应用——HTTPS P 82

PKI的应用——SSL VPN P 83

RPKI (Resource Public Key Infrastructure) P 85

BPG 源验证/路径验证 P 86-87

PKI 安全机制：证书撤销列表查询效率低，实时性差，通过在线证书状态协议建立一个可实时响应的机制，CA服务器实时响应验证证书。 P88-89

交叉认证 P 96 将以前无关的CA连接在一起的机制，认证主体和颁发者都是CA，根据CA是否属于同一信任域，分为**域内交叉认证**和**域间交叉认证**

桥CA模型克服层次信任模型和分布式信任模型的缺点，连接不同的PKI系统

风险和解决方案 P 99-105

中心监管带来安全脆弱性和监督失效，证书误发、恶意颁发或通过钓鱼网站颁发，证书维护存在风险

解决方法：证书透明机制+经济激励机制

摘要与签名

散列函数 (P.108)

散列函数，又称哈希（Hash）函数，它可以把任意长度的消息压缩为固定长度的二进制串：散列值（hash value）： $h = H(m)$ 。散列函数是进行消息认证的基本方法，主要用于消息完整性检测和数字签名。

数字签名是使用私钥加密证明自己是公钥所有者，而摘要是指哈希散列值。

散列函数的性质 (P.110)

1. 可以根据任意长度的消息计算出固定长度的散列值。
2. 能够快速计算出散列值。
3. 单向性：根据消息计算出摘要，但不可反向。
4. 消息不同散列值也不同

散列函数的安全性：HASH 碰撞 (P.84)

哈希碰撞是指不同的输入产生相同的哈希输出。这种情况在理论上是可能的，因为哈希函数接受任意长度的输入，但输出的长度是固定的。所以，存在两个不同的输入产生相同输出的可能性。如果这种碰撞能在实践中找到，那么哈希函数就被认为是不安全的。

哈希算法实例 (P.115)

- **MD5**: 生成128位的哈希值，广泛用于确保信息传输的完整无误。
- **SHA-1**: 安全哈希算法，主要用于各种安全认证中，生成160位的哈希值。
- **SHA-256**: 属于SHA-2标准下的哈希算法，生成256位的哈希值，被广泛用于比特币中。

散列函数的应用: 比特币 (P.116)

比特币使用了SHA256、RIPEMD160这两种散列函数。SHA256用于区块的头部信息、交易数据、工作量证明、比特币地址等，输出256位的哈希值。而RIPEMD160则用于比特币地址生成，可以让地址更短，输出160位的哈希值。

SHA256算法 (P.117)

SHA256是SHA-2标准下细分出的一种散列函数。算法的实现主要分为三个步骤：常量初始化，信息预处理，生成摘要。

消息认证码 (MAC) (P.124)

消息认证码（Message Authentication Code）的输入包括任意长度的消息和一个发送者与接受者之间共享的密钥，它可以输出固定长度的数据，这个数据称为MAC值。

HMAC (P.125)

HMAC是一种使用散列函数来构造消息认证码的方法。该方法所使用的散列函数不仅限于一种，任何高强度的散列函数都可以被用在HMAC。使用SHA-1，SHA-256，所构造的HMAC分别称为HMAC-SHA1，HMAC-SHA256。

公钥密码与数字签名 (P.126)

在公钥密码系统中，用公钥进行加密，用私钥进行解密。数字签名是用私钥加密的过程，用公钥进行验证。

数字签名方法 (P.127)

直接对消息签名，或者对消息的摘要签名。

RSA数字签名方案 (P. 128)

数字签名的分类 (P.129)

包括

- 盲签名（签名者不知道代签名文件内容时使用的数字签名）
- 门限签名（如果一个群体中有 n 个人，那么至少需要 p 个人签名才视为有效签名($n > p$)）
- 群签名（一个群体由多个成员组成，某个成员可以代表整个群体来进行数字签名，而且该成员作为签名者可以被验证）
- 代理签名（密钥的所有者可以将签名权利授予第三方，获得权力的第三方可以进行数字签名）
- 双重签名（签名者希望有个中间人在他与验证者之间进行验证授权操作）。

数字签名的应用 (P.130)

包括

- 网站认证（通过对网站域名信息、主体身份信息、域名权属信息等进行严格鉴证审核，并利用PKI数字签名技术形成不可篡改的认证标识，在互联网终端以安全可靠的方式进行展示，使网民更直观确认网站的真实身份）、
- 比特币（比特币是一种完全匿名的数字货币，它的身份认证是基于ECDSA，比特币的账户地址就是对公钥计算摘要得到的，并用私钥确认账户拥有者）、
- 代码签名（如果Windows上的可执行程序程序来源于正规公司

密码分析技术 (P.131)

密码分析技术是在未知密钥的前提下，从密文恢复出明文、或者推导出密钥，对密码进行分析的尝试。

唯密文攻击(Ciphertext only) (P.133)

分析者有一些消息的密文，都是用同一算法加密的。分析者的目标是恢复尽量多的明文或者推算出密钥。

已知明文攻击(Known Plaintext) (P.133)

密码分析者不仅可以得到一些消息的密文，而且也知道这些消息的明文。分析者的任务是得到加密的密钥或者得到一个算法，该算法可以解密用同样的密钥加密的消息。

选择明文攻击(Chosen Plaintext) (P.133)

分析者不仅知道一些消息的明文和密文，而且可以选择被加密的明文，这比已知明文攻击更加有效。分析者的任务是得到加密的密钥或者得到一个算法，该算法可以解密用同样的密钥加密的消息。攻击者拥有加密机的访问权限，可构造所选一定数量的明文所对应的密文。

选择密文攻击(Chosen Ciphertext) (P.133)

密码分析者能选择不同的被加密的密文，并可以得到对应的解密的明文，密码分析者的任务是推出密钥。主要针对公钥算法。攻击者拥有解密机的访问权限，可构造所选一定数量的密文所对应的明文。

相关密钥攻击(Related Key) (P.133)

攻击者可以得到被两个不同的钥匙所加密（或解密）得到的密文（或明文）。攻击者不知道这两个钥匙的数值，但知道这两个钥匙之间的关系，比如两个钥匙之间相差一个比特。

密码分析技术 (P.135)

密码算法的相对安全性的衡量标准包括破解算法的代价是否大于加密数据本身的价值，以及破解算法的时间是否超过了信息的生命期。对于不同密钥长度的加密，进行穷举密钥搜索所需的平均时间也会大不相同，随着密钥长度的增加，所需的时间将急剧增加。

第4讲 隐私保护

第1节 隐私保护技术初探 P16-

隐私保护技术概述

在大量的网络空间数据中获取有用信息时，如何在保护数据相关者的隐私的同时获取有用信息变得尤为重要。数据隐私保护技术的研究主要分为两个方面：**面向数据发布**和**面向数据挖掘**。

面向数据发布的隐私保护 P17-

面向数据发布的隐私保护包含以下几个部分：

1. **基于限制发布的隐私保护**：这种技术在将数据公布给数据挖掘者之前，对数据进行扰动、加密、匿名等处理，将数据中的隐私藏起来。研究主要集中于数据匿名化，例如**有选择的发布原始数据、不发布或者发布精度较低的敏感数据**。
2. **基于数据失真的隐私保护**：该技术通过对原始数据进行扰动，目的是隐藏真实数据，只呈现出数据的统计学特征。失真后的数据满足两个条件：保持原本的某些特性不变，且攻击者不能根据失真数据重构出真实的原始数据。此技术主要包括随机化、阻塞、变形、交换等。

3. **基于数据加密的隐私保护**：这种技术对原始数据进行加密，通过密码机制实现其他参与方对原始数据的不可见性以及数据的无损失性。由于加密技术可解决安全通信的问题，因此多应用于分布式应用。两种数据存储模式：垂直划分（每个人参与者存储部分属性）和水平划分（每个参与者存储部分数据）。

基于限制发布	基于数据失真	基于数据加密
有选择的发布原始数据、不发布或者发布精度较低的敏感数据	对原始数据进行扰动 目的是隐藏真实数据 只呈现出数据的统计学特征	对原始数据进行加密 通过密码机制实现其他参与方对原始数据的不可见性以及数据的无损失性

面向数据发布的隐私保护技术	优点	缺点
基于限制发布的隐私保护技术	发布的数据真实可靠	数据丢失部分信息
基于数据失真的隐私保护技术	算法效率较高	由于干扰使数据丢失部分信息
基于数据加密的隐私保护技术	数据的安全性和准确性均较高	计算开销很大

面向数据挖掘的隐私保护 P21-26

面向数据挖掘的隐私保护包括以下几个部分：

1. **关联规则的数据挖掘**：关联规则挖掘是数据挖掘领域研究的重点之一，是从大量数据中挖掘数据项之间隐藏的关系，发现数据集中项集之间的关联和规则的过程。例如购物篮分析，寻找商品之间隐藏的关联规则。
2. **隐私保护的关联规则挖掘**：主要有下面变换和隐藏的两类方法，两类方法都会影响对于非敏感规则的挖掘

变换 (distortion)	隐藏 (blocking)
修改支持敏感规则的数据，使得规则的置信度和支持度小于一定的阈值而实现规则的隐藏	不修改数据，而是隐藏生成敏感规则的频繁项集，尽可能降低敏感规则的置信度或者支持度，以此使得需要保护或隐藏的规则不被挖掘出来

3. 隐私保护的分类和聚类挖掘 P25，聚类和分类都会暴露数据集中的隐私敏感信息

第2节 匿名化

在数据发布过程中，为了保护用户的敏感数据和个人身份之间的对应关系，需要采取匿名化隐私保护模型。传统的匿名化方法往往无法抵抗链接攻击，而 k-anonymity 是一种有效的隐私保护模型，能够解决链接攻击问题。

传统的匿名化方法 P29

传统的匿名化方法包括删除容易关联到个人的属性（如姓名和家庭住址），或者将姓名替换为假名。然而，这些方法存在一定的局限性，无法提供足够的隐私保护，容易受到链接攻击。

链接攻击 P31-33

链接攻击是指攻击者通过对发布的数据和其他渠道获取的数据进行链接操作，推断出隐藏在匿名化数据中的隐私信息。例如，攻击者可以将医疗数据集与其它公共数据集的准标识符进行联系，从而推断出匿名化数据中的敏感信息。

k-anonymity (PPT 第 34 页)

k-anonymity 是一种数据匿名化技术，当攻击者尝试链接攻击时，由于任意一条记录的攻击，都会同时关联到等价类中的其他 $k-1$ 条记录，因此攻击者无法确定特定用户。 k 值越大，隐私保护效果越好，但是相应的数据丢失也越严重。

具体做法：将 k 个记录放入一个等价类中，要求任意一条记录与其他至少 $k-1$ 条记录相似而不可区分，这样数据中的每一条记录都能找到与之相似的记录，降低了数据的识别度，如果一条记录由于样本太少而无法找到 $k-1$ 条相似的记录，那么这条数据不应当被纳入数据集。

同质性攻击和背景知识攻击 (PPT 第 36-37 页)

同质性攻击：在数据匿名化过程中可能存在的攻击模式包括**同质性攻击**和**背景知识攻击**。同质性攻击是指没有对敏感属性进行约束，最终结果可能造成隐私泄露。例如，如果一名选民的年龄和邮政编码符合第一个等价类，而第一个等价类里**全有心脏病**，那么攻击者可推断该选民可能患有心脏病。

背景知识攻击：攻击者可以通过掌握的足够的相关背景知识以很高的概率确定敏感数据与个体的对应关系，得到隐私信息。例如，如果一名选民符合第二个等价类，第二个等价类里面不是糖尿病就是哮喘，且攻击者发现他不像是患有哮喘，那么攻击者可推断该选民可能患有糖尿病。

l-diversity (PPT 第 38-40 页)

l -diversity 在 k -anonymity 的基础上，要求保证每一个等价类的敏感属性至少有 l 个不同的值，即每个用户的敏感属性值在等价类中可以找到与此值不同的至少 $l - 1$ 个属性值，使攻击者最多只能以 $\frac{1}{l}$ 的概率确认某个用户的敏感信息，但无法保证隐私不被泄露。

1. 敏感值的分布显著不同
2. 没有考虑语义信息

t-closeness (P41)

在 k -anonymity 和 l -diversity 的基础上， t -closeness 考虑了敏感属性的分布问题，**要求所有等价类中的敏感属性的分布尽量接近该敏感属性的全局分布**，差异不能超过阈值 t 。

以信息损失为代价，隐私保护效果逐个提高，但是它们不一定能保证隐私不被泄露。

隐私泄露风险

造成较大的信息损失，信息损失可能会使数据使用者们做出误判；对所有敏感属性提供了相同程度的保护并且没有考虑语义关系，造成了不必要的信息损失；不同的用户对于隐私信息有着程度不同的隐私保护要求；属性与属性之间的重要程度并不相同；没有考虑数据动态更新后重发布的隐私保护问题。

数据匿名化方法 (页码：43)

数据匿名化的方法包括**泛化**、**抑制**、**聚类**、**微聚集**、**分解**和**置换**等。目前提出的匿名化主要通过**泛化**和**抑制**实现，它们能保持发布前后数据的真实性和一致性，但是信息肯定有损失。

匿名化方法	思想
泛化	用更抽象、概括的值或区间代替精确值
抑制	将数据表中的数据直接删除或隐藏
聚类	按照给定的规则将数据集分成各类簇，尽量保证簇内对象相似，不同簇的对象相异
微聚集	相似的数据划分在同一个类中，每个类至少有 k 条记录，用类质心代替类中所有记录的准标
分解	根据敏感属性值对数据表分组，尽量使得同一组的敏感属性值不同，将分组后的数据表拆分为分别包含准标识符属性信息和包含敏感属性信息的两张表
置换	对数据表分组，把每组内的敏感属性值随机交换，打乱顺序，再拆分数据表，对外发布

泛化（页码：44）

泛化是一种常用的数据匿名化方法，其思想是**将准标识符的属性用更一般的值或者区间代替**。它不会引入错误数据，方法简单，泛化后的数据适用性强，对数据的使用不需要很强的专业知识。但是其预定义泛化树没有统一标准，信息损失大，对不同类型数据的信息损失度量标准不同。

泛化树（页码：45）

泛化树可以看作是一种将底层取值泛化为高层取值的结构，每层的取值构成一个泛化域。底层的取值最具体，顶层的取值最模糊。数值型数据泛化树和分类型数据泛化树是两种常见的泛化树形式。

- **数值型数据泛化树**：值被一个覆盖精确数值的区间代替
- **分类型数据泛化树**：用一个更一般的值代替原值

域泛化（全局泛化）（页码：46）

在数据匿名化过程中，可以采用**域泛化**（全局泛化）和**值泛化**（局部泛化）两种策略。

- **域泛化**：将一个给定的属性域泛化为一般域，将准标识符属性值从底层开始同时向上泛化，一层层泛化直到满足隐私保护要求，然后同时停止泛化

全域泛化	子树泛化	兄弟节点泛化
某个属性的全部值必须在同一层上进行泛化	泛化树中的同一个父亲节点下的所有孩子节点全部泛化或者全部不泛化	在同一个父亲节点下，如果对部分孩子节点进行泛化，其他兄弟节点不要求泛化，父亲节点只能代替泛化了的孩子节点

- 全域泛化：如果“巴黎”和“里昂”泛化到“法国”，则必须同时把“伦敦”和“爱丁堡”泛化到“英国”，保证泛化树中所有路径的泛化粒度相同
- 子树泛化：如果把34泛化到[31,40]，那么37和39也要泛化到[31,40]，而42、48和50可以不泛化到[41,50]
- 兄弟节点泛化：如果“巴黎”泛化到“法国”，那么“里昂”可以不泛化到“法国”

值泛化（局部泛化）(页码：47)

由于早期的泛化操作都采用整体泛化方案，造成的信息损失大，因此提出了局部泛化方案，将原始属性域中的每个值直接泛化成一般域中的唯一值，将准标识符属性值从底层向上泛化，但是可以泛化到不同的层次。**局部泛化是相同的属性值不全部被泛化，其中一部分值不变。**

单元泛化：如果年龄为34的数据有两条，可以把其中一条泛化到[31,40]，另一个值不变。**区别于子树泛化，子树泛化对于相同的数据必须进行同样的泛化！**

多维泛化：单位泛化是依次将单个属性的属性值转化为相应泛化树中的值，而多维泛化则是在转化过程中同时将多个属性的属性值以笛卡尔乘积的形式转化为相应属性的泛化树中的笛卡尔积形式

单元泛化	多维泛化
对某个属性的一部分值进行泛化，另一部分值保持不变	对多个属性的值同时泛化，只需要对不符合限制要求的等价类进行泛化，要求一个等价类中所有记录都泛化成相同的值

抑制（P48）

抑制（Suppression），也称为隐藏或隐匿，是一种数据匿名化方法，用于将准标识符属性值从数据集中直接删除或用代表不确定值的符号（如" *"）替代。抑制可以与泛化方法结合使用。

在抑制中，可以采用以下三种方式：

记录抑制	值抑制	单元抑制
对数据表中的某条记录进行抑制处理	对数据表中的某个属性的值全部进行抑制处理	对数据表中某个属性的部分值进行抑制处理

抑制方法可以用于敏感数据的保护，但需要权衡数据可用性和隐私保护之间的平衡。

第3节 差分隐私(P49)

差分隐私是一种严格的、可证明的隐私保护模型，相对于传统的匿名化方法，它提供了更严格的隐私保护。差分隐私在数据匿名化方面与传统的匿名化方法有所不同。传统的匿名化方法不能提供足够的数学保障，没有严格定义攻击模型，无法抵抗背景知识攻击，并且难以提供严格和科学的方法来证明其隐私保护水平。相比之下，差分隐私提供了更强的隐私保护，并且能够提供严格的定义和量化评估方法。

- **严格定义的隐私保护**：差分隐私对隐私保护进行了严格的定义，并提供了量化评估方法，使不同参数处理下的数据集所提供的隐私保护水平具有可比较性。
- **抵抗背景知识攻击**：差分隐私假设攻击者掌握最大的知识背景，即能够获得除目标记录外所有其他记录的信息。因此，差分隐私的设计考虑了更强的攻击模型，能够更好地保护隐私。
- **严格的隐私保护证明**：差分隐私提供了严格和科学的方法来证明其隐私保护水平。当模型参数改变时，可以对其隐私保护水平进行定量分析，以便在隐私与数据可用性之间进行权衡。

差分攻击 P51

前三行有两人患病，前四行有三人患病，判断出数据集中的第四行代表的用户患有癌症，如果此时攻击者知道第四行代表的用户是钱六，那么攻击者就可以通过这种方式在没有具体查询特定某人个人信息的前提下获得其隐私数据。为抵抗差分攻击，**差分隐私要求保证任意一个个体在数据集中或者不在数据集中时，对最终发布的查询结果几乎没有影响。**

差分隐私思想 P52

差分隐私的核心思想是通过引入随机化扰动的方式，在查询结果中添加噪声，从而保护个体隐私。随机算法 M 会对信息进行扰动，使得同一查询在两个数据集上产生相同结果的概率的比值接近于 1。这样可以保证任意一个个体在数据集中或者不在数据集中时，对最终发布的查询结果几乎没有影响。

差分概念 (P53)

隐私保护机制：对数据集 D 的各种映射函数被定义为查询 (Query)，用 $F = \{f_1, f_2, \dots\}$ 来表示一组查询，算法 M 对查询 F 的结果进行处理，使之满足隐私保护的条件的，此过程称为隐私保护机制（类比查询病患的例子）

邻近数据集：设数据集 D 和 D' 具有相同的属性结构，两者的对称差记作 $D \Delta D'$ ， $|D \Delta D'|$ 表示 $D \Delta D'$ 中记录的数量，若 $|D \Delta D'| = 1$ ，则称 D 和 D' 为邻近数据集 (Adjacent Dataset) 例如，设 $D = \{1, 2, 3, 4, 5\}$ ， $D' = \{1, 2, 4\}$ ，则 $D \Delta D' = \{3, 5\}$ ， $|D \Delta D'| = 2$

差分隐私定义 P54

算法 M 提供 ϵ - 差分隐私保护，其中参数 ϵ 称为隐私保护预算。

- 当 $\epsilon = 0$ 时，攻击者无法区分相邻数据集，保护程度最高，但数据可用性最差。
- 当 ϵ 增大时，保护程度逐渐降低， ϵ 过大会造成隐私泄露。

差分隐私通过严格定义和数学方法的引入，提供了一种可靠的隐私保护模型，能够在隐私和数据可用性之间取得平衡。

差分隐私的实现(P56)

差分隐私可以通过在查询函数的返回值中加入噪声来实现隐私保护。这种噪声的加入可以使用不同的机制来实现，例如拉普拉斯机制和高斯机制。

全局敏感度和局部敏感度 P57~59

全局敏感度和局部敏感度是衡量查询函数敏感度的重要指标，可以用来确定加入的噪声的大小，从而在隐私保护和数据可用性之间取得平衡。给定的数据集 = 全局敏感度中 1 阶范式距离达最大的数据集时，局部敏感度就等于全局敏感度。

全局敏感度	局部敏感度
对任意的邻近数据集 D 和 D' ；只由查询函数决定。	对给定的数据集 D 和它的任意邻近数据集 D' ；由查询函数和给定的数据集中的数据共同决定。

数值型差分隐私：拉普拉斯和高斯机制

数值型差分隐私的实现机制有拉普拉斯机制和高斯机制，通过在查询结果中加入随机噪声实现隐私保护 拉普拉斯机制提供的是严格的 $(\epsilon, 0)$ 一差分隐私保护，而高斯机制提供的是松弛的 (ϵ, δ) 一差分隐私保护。

拉普拉斯机制使用拉普拉斯分布来生成随机噪声，高斯机制使用高斯分布来生成随机噪声，并分别加到查询结果中。在选择拉普拉斯或高斯机制时，需要平衡隐私保护和数据可用性，并根据具体的应用需求选择合适的参数。较小的 ϵ 值表示更强的隐私保护，但可能导致较大的噪声和较低的数据可用性。

非数值型差分隐私：指数机制

对于非数值型（离散型）的数据，可以使用指数机制实现差分隐私。指数机制使用指数分布来生成随机噪声，并根据查询的可用性函数对输出结果进行加权。可用性函数衡量了输出结果的优劣程度，指数机制提供 $(\epsilon, 0)$ -差分隐私保护。

总结起来，差分隐私的实现可以使用拉普拉斯机制和高斯机制处理数值型数据，而非数值型数据可以使用指数机制来保护隐私。选择适当的机制和参数可以在隐私保护和数据可用性之间取得平衡，确保数据的隐私得到有效保护。

第4节 同态加密(P69)

同态加密的基本思想是对密文进行操作，而计算结果的解密值与对应明文的计算结果相同。这种特性使得同态加密在安全数据外包的场景中具有重要应用。

通过同态加密技术，可以将数据安全地外包给数据计算方，而不必担心隐私泄露。数据所有者可以对数据进行同态加密，然后将加密后的数据传输给计算方。计算方可以在不知道原始数据的情况下对密文进行计算和处理，并返回处理后的结果。最后，数据所有者使用其私钥对结果进行解密，获得最终的处理结果。

通过同态加密实现安全的数据外包，既保护了数据隐私，又充分利用了数据计算平台的计算能力，实现了数据隐私和计算能力之间的平衡。

组成P76

- **KeyGen 算法**：通过计算安全参数生成一对公私钥。
- **Encrypt 算法**：使用公钥将明文加密为密文。
- **Evaluate 算法**：在密文上进行运算，例如加法或乘法。
- **Decrypt 算法**：使用私钥将密文解密为明文。

同态加密的发展

1. **仅支持加法同态的加密体制**：最早的同态加密体制只支持加法同态或乘法同态，但不能同时满足两者。
2. **半同态加密**（Partially Homomorphic Encryption, PHE）：半同态加密体制同时满足加法同态和乘法同态的性质，但只能进行有限次的加和乘运算。
3. **浅同态加密**（Somewhat Homomorphic Encryption, SWHE）：浅同态加密体制也同时满足加法同态和乘法同态的性质，但可以进行任意多次加和乘运算。
4. **全同态加密**（Fully Homomorphic Encryption, FHE）：全同态加密体制是最理想的同态加密形式，它可以在不解密的条件下对加密数据进行任何可以在明文上进行的运算，实现了深度和无限的数据分析，对加密信息进行深入分析而不影响其保密性。

目前，全同态加密仍面临着复杂度问题，离实际应用仍有一定距离。

同态加密的应用

- **医疗机构的数据分析**：在医疗机构中，数据处理能力较弱，可以借助云服务商提供的计算服务。使用同态加密，医疗数据可以在加密的状态下存储和计算，而不泄露隐私信息。云服务商可以进行数据搜索、分析和处理等功能，同时保护数据隐私。
- **电子投票**：同态加密可以用于设计安全的电子选举系统。统计方可以在不知道投票者投票内容的情况下对投票结果进行统计，既保证了投票者的隐私安全，又能够保证投票结果的公正性。

同态加密的优势与挑战 P82

优势：

- **降低计算代价**：同态加密可以对多个密文进行计算后再解密，降低了计算代价。
- **降低通信代价**：同态加密可实现无密钥方对密文的计算，无需经过密钥方，降低了通信代价。
- **保证数据安全性**：同态加密可以实现让解密方只能获知最终的结果，而无法获得每个密文的消息，从而保证了信息的安全性。

挑战：

- **计算效率**：当前的同态加密方案的计算复杂度较高，如何设计高效的全同态加密方案仍然是一个问题。
- **安全性**：同态加密方案大多基于未论证的困难问题，寻找可论证的困难问题仍然是一个挑战。
- **噪音消除**：同态加密需要额外的消除噪音算法，如何设计具有自然同态性的全同态加密方案仍然是一个问题。

尽管同态加密正逐步向实用性靠近，但其安全性和实用性方面的研究还有很长的路要走。

半同态加密 P83

半同态加密体制同时满足加法同态和乘法同态的性质，但只能进行有限次的加和乘运算。

在一个加密方案中，如果加密算法和解密算法满足以下条件：

$$D(Enc(a) \otimes Enc(b)) = a \oplus b$$

其中， \otimes 表示在密文域上的运算， \oplus 表示在明文域上的运算，那么该加密方案被称为半同态加密。

半同态加密分为乘法同态加密和加法同态加密：

- 乘法同态加密：RSA 公钥加密算法，ElGamal 公钥加密算法 P84
- 加法同态加密：Paillier 公钥加密算法 P85，是最常用且最具实用性的加法同态加密算法。

全同态加密 P91

全同态加密是指同时满足加法同态性和乘法同态性的加密方案，可以进行任意多次加和乘运算的加密函数。加密算法功能更强大，但由于计算复杂度较高，加密算法设计更加复杂，整体性能远不及半同态加密算法。目前，全同态加密仍然是一个热门的研究领域，尚在不断发展和完善中。

第5节 安全多方计算(P93)

安全多方计算（Secure Multi-Party Computation, MPC）是一种解决互不信任的多方参与者在保护各自数据的前提下进行合作计算的方法。在安全多方计算中，多个参与者希望共同计算一个函数，同时保护各自的隐私或秘密数据，而不愿意让其他参与者知晓自己提供的信息。它解决了一组互不信任的参与方之间保护隐私的协同计算问题。安全多方计算协议使得参与者能够进行合作计算，而不需要彼此泄露输入信息，从而保护隐私。

安全多方计算的形式化描述 P98

安全多方计算的目标是在无可信第三方的情况下安全地计算一个约定函数，同时要求每个参与方除了计算结果外不能得到其他参与方的任何输入信息。安全多方计算需要满足以下特征：

- **输入独立性**：各方能独立输入数据，计算时不泄露本地数据。

- **计算正确性**：计算结束后各方能够得到正确的计算结果。
- **去中心化性**：各参与方地位平等，提供了去中心化的计算模式。

安全多方计算的威胁模型 P99

- **诚实模型**：参与者按照协议要求行动，不提供虚假数据，不泄露、窃听数据，不终止协议，完全按照协议执行。
- **半诚实模型**：在诚实模型基础上保留所有收集到的信息，推断其他参与者的秘密信息。
- **恶意模型**：无视协议要求，可能提供虚假数据、泄露数据、窃听甚至终止协议。

安全多方计算的计算模型 P100

- **基于"可信第三方"的计算模型**：参与方得到计算结果，可信第三方得到参与方的输入信息和计算结果，信息的保密性由可信第三方来保证。然而，在实际情况下很难找到完全可信的第三方，所以这种模型很少使用。
- **交互计算模型**：参与方按照协议步骤执行计算，按协议的要求将中间结果发送给其他参与方，同时接收其他参与方计算的中间结果，信息的保密性由协议的安全性来保证。这是安全多方计算中**最常用的模型**，提供了一种去中心化的计算方式。
- **外包计算模型**：随着云计算的发展而发展起来的计算模型，各个参与方希望使用云计算提供的计算资源，但不想直接将信息委托给云计算服务提供商，也不想让其得知计算结果。参与方将信息处理后存储在外包服务器上，由外包处理器对所有参与方的秘密信息进行计算，并将结果发送给各参与方，信息的保密性由协议的安全性来保证。

基本密码协议 P102

在安全多方计算中，基本密码协议是实现安全多方计算的关键工具。它包括茫然传输协议（Oblivious Transfer, OT）、混淆电路协议（Garbled Circuit, GC）和秘密共享协议（Secret Sharing）等。这些协议使用了多种密码学技术，如同态加密技术，来实现安全的计算过程。

安全多方计算的应用 P103

安全多方计算的优势在于能够在保护隐私的同时进行计算，并具有较高的安全性和准确性。它被广泛应用于以下领域：

- **门限签名**：将私钥拆分为多个秘密分片，只有在达到门限值的参与者共同协作时才能生成有效的签名。
- **电子拍卖**：在不直接公开竞拍者的出价情况下，能够计算出所有参与者输入的最大值或最小值，使得在线拍卖成为现实。
- **联合数据查询**：多个数据库可以共同进行数据查询，使用安全多方计算保护各数据库的私有信息或知识产权。
- 分布式机器学习：联邦学习

安全多方计算牵涉到密码学的各个分支，有着广阔的应用领域，其优势为比较安全和准确，但涉及的加密技术开销、通信开销也很大。目前的研究主要集中于降低计算开销、优化分布式计算协议。

百万富翁问题 P105-109

百万富翁Alice和Bob想相互比较一下谁更富有，但是他们都不想让对方知道自己拥有多少财富，如何不借助第三方比较两个人的财富多少？

第6节 联邦学习 P111

联邦学习的定义 P115

联邦学习的分类 P121

- 横向联邦学习
- 纵向联邦学习
- 联邦迁移学习

第5讲 系统硬件安全

硬件攻击技术

硬件攻击技术是网络安全中的重要领域，它涉及利用硬件设计和实现中的漏洞、硬件木马、硬件故障以及侧信道攻击等手段来攻击系统。以下是硬件攻击技术的相关内容：

漏洞攻击 P16

漏洞攻击利用硬件设计中存在的缺陷来攻击系统。硬件漏洞主要指硬件设计和生产中存在的缺陷，这些缺陷通常由于设计说明不完善、设计人员编码不规范、硬件性能优化、测试与验证覆盖率不够高等因素引起。

- **指令集漏洞**：不同系统架构对相同指令集的实现方式有所不同，指令集漏洞是由于处理器开发人员在指令集的硬件实现过程中，对一些特殊情况考虑不周导致的。各大处理器厂商会定期发布产品的勘误表，列出指令集漏洞并提供修复措施。
- **浮点除（FDIV）的实现漏洞**：在某些处理器中，浮点除法的实现存在错误，导致计算结果不准确。例如，英特尔奔腾处理器中的FDIV错误就是一个经典的实现漏洞。
- **性能优化缺陷**：Meltdown和Spectre源于乱序执行和分支预测（提高CPU性能）

硬件木马(P21)

硬件木马是指被第三方故意植入或设计者有意留下的特殊模块和电路，它们潜伏在正常的硬件中，并在特定条件下被触发，实施恶意功能，如窃取敏感信息或拒绝服务攻击。

不可信的产业链（页码：22）

硬件木马通常由不可信的合作厂商引入，特别是对于我国的高端服务器、操作系统等核心软硬件，大部分都是进口的。这导致了木马和后门问题的严重性。

木马分类（页码：23）

硬件木马可以根据以下特征进行分类：

- **插入阶段**：指木马的植入阶段，可以是在制造过程中的偶然插入（由制造过程的缺陷引起），或者是蓄意插入（由攻击者植入）。

- **抽象级别**：指木马在电路中的位置，可以是在较高级别的模块中（如处理器、芯片组），也可以是在较低级别的模块中（如外围接口芯片）。
- **激活机制**：指木马被触发的机制，可以是在已知的功能状态下触发，也可以是在特定状态的组合或序列下触发。
- **影响**：指木马对系统的影响程度，可以是功能参数错误，也可以是泄露敏感信息等。
- **位置**：指木马在电路中的具体位置，可以是在信号路径上，也可以是在控制路径上。

木马与故障的区别（页码：24）

硬件木马与电路故障是不同的概念：

- **电路故障**是指在电路的生产加工过程中由于工艺或流程不完美导致的电路缺陷。
- **硬件木马**是指人为蓄意加入的恶意电路模块，其功能超出了指定功能之外。与电路故障不同，硬件木马通常较难检测到。

	电路故障	硬件木马
激活	通常在已知的功能状态下	任意电路中间节点特定状态的组合或者序列（可以为数字或者模拟信号）
植入	偶然（制造过程的某些缺陷）	蓄意（由产业链的攻击者植入）
作用	电路功能或参数错误	电路功能参数错误、泄漏信息等

软件木马与硬件木马对比（页码：25）

软件木马是一种带有恶意代码的软件，与硬件木马攻击目标相似，软件木马通常可以在应用领域内解决，而硬件木马一旦植入就很难移除。

	软件木马	硬件木马
激活	存在恶意代码的软件，在代码运行过程中激活	存在集成硬件中，在硬件运行过程中激活
感染	用户交互过程中传播	由集成硬件设计流程中不可信的参与方植入
补救措施	通过软件更新移除	流片一旦加工成型就不能移除

硬件故障（页码：26）

硬件故障是常见的硬件问题，攻击者可以利用合适的硬件故障来窃取敏感信息或实施硬件攻击。这利用了硬件的异常行为来达到攻击的目的。

故障注入攻击（P27）

故障注入技术最初被用来验证系统的可靠性和可用性，并在电子器件的整个制造过程中得到广泛使用，主要做法是试图通过受控实验改变正在运行的系统的工作环境，从而引入故障，并根据系统的工作状态评价系统的可靠性和可用性。最初，故障注入技术被用于验证系统的可靠性和可用性，并在电子器件制造过程中广泛使用。然而，这种技术也可以被恶意使用来攻击系统的安全性。

故障注入分类 (P28)

故障注入可以根据实施方法的不同进行分类。其中，基于硬件的故障注入攻击是一种较为常见的方法。它的实现相对简单，成功率较高，并且故障注入过程也比较可控，因此被广泛研究和应用。

光故障注入

光故障注入利用光电效应来激发半导体硅片产生大量的自由电子和空穴，从而造成晶体管的导通，引入故障。例如，研究人员发现通过将激光调至精确频率并对准智能语音设备，可以像用户声音一样激活语音助理并进行交互，从而解锁汽车、打开车库门等。这种方法的作用距离甚至可以达到一百多米。

电磁故障注入

随着CMOS工艺特征尺寸的不断缩小，电子器件变得更加敏感，对电磁攻击更为脆弱。电磁故障注入利用产生的电磁场对目标设备的内部产生干扰，引入故障。例如，电磁故障注入可以通过产生瞬态感应电压毛刺来改变芯片内部晶体管的状态，从而导致芯片的不可预期行为。

时序电路与故障注入

时序电路是一种按照时钟信号进行同步操作的电路。故障注入可以破坏时序电路中的约束条件，导致电路的输出产生错误。例如，提高时钟频率或降低电压都可能破坏时序电路的约束条件，导致电路功能不稳定或输出保留上次的值。

影响因素

故障注入攻击的效果受多个因素的影响，包括攻击时刻、攻击强度、作用时间和空间位置等。不同的因素会对故障注入攻击的有效性产生影响。

有效的故障注入攻击可以通过精确控制这些因素来引入故障，并对系统的安全性产生重大影响。因此，对于系统的设计和开发人员来说，理解和防范故障注入攻击至关重要。

侧信道攻击与旁路侧信道 (页码：39)

故障注入攻击对系统进行人为干预，侧信道攻击则利用系统产生的各种旁路信息展开攻击

侧信道攻击是利用硬件系统的旁路信息实施攻击的一种方式。它通过从密码系统的物理实现中获取信息，如时间信息、功耗消耗、缓存使用等。侧信道攻击的设备成本低，但攻击效果显著，严重威胁了密码设备的安全性。

硬件隔离并不能完全保证安全，因为旁路侧信道攻击可以利用硬件系统的旁路信息来获取敏感数据，绕过硬件隔离的保护机制。（页码：40）

物理侧信道攻击 (页码：42)

物理侧信道攻击是一种侧信道攻击方法，基于从密码系统的物理实现中获取的信息，如时间侧信道、功耗侧信道、电磁侧信道和声波侧信道。

- **时间侧信道**：通过系统在不同行为流程下所表现出的执行时间差异，推断有用的信息，以达成或辅助攻击。
- **功耗侧信道**：通过分析设备随时间的功耗曲线，推测CPU执行过程中的某些信息。

时间侧信道攻击（页码：43-44）

时间侧信道利用加密系统计算过程中时间花费和数据之间的关联性。加密系统对于不同的输入所执行的运算过程不同进而导致所花费的运行时间不同，导致该现象的原因有代码执行过程中的分支条件语句、相关计算数据是否在缓存中命中，以及一些CPU的优化措施等。因此加密系统的运算时间与密钥以及输入数据之间存在一定的关联性，攻击者通过测量加密系统的运行时间，有针对性地进行数据分析，就可以通过时间侧信道提取加密系统的密钥或者其他敏感数据

时间侧信道攻击利用系统在不同行为流程下所表现出的执行时间差异来推断有用的信息。例如，可以通过观察登录验证的时间差异来判断用户名是否存在，进而进行针对性的攻击。

攻击者可以提交多个请求，统计从发出请求到收到服务器回执所耗费的时间，并通过测量多次取平均值或中值等方法消除网络延迟引入的时间抖动。通过统计时间较长的请求，攻击者可以推断出用户名是否存在，从而发动针对性的攻击。

功耗侧信道攻击（页码：45-47）

功耗侧信道是现代侧信道攻击中最常用的侧信道信息。加密系统的功耗可以分为静态功耗和动态功耗，静态功耗是指集成电路本身的能耗或者晶体管的漏电流，静态功耗一般是个稳定值，在功耗侧信道中使用更多的是动态功耗

集成电路的动态功耗和输入数据之间关系密切，比如当十六进制字节从A1 变化到B2 时，会比从01 变化到00 产生更多的电容充放电，因此对于正在执行数据处理的加密系统来说，动态功耗的测量数据实际上包含了大量处理数据的信息，攻击者通过分析目标设备的功耗信息就可以破解密钥从而获得敏感数据。

- 1) 简单功耗: 观察功耗的变化趋势和电压的时间变化趋势。
- 2) 差分功耗分析: 相比于简单功耗分析为更高级的功耗分析手段，允许攻击者通过对多次密码学操作所收集到数据进行统计分析计算出在密码学运算中的中间值
- 3) 相关功耗分析: 引入多数据源及不同的时间偏置，再进行数据统计和分析

功耗侧信道攻击利用设备在不同指令执行时的功耗特征来推测CPU执行过程中的某些信息。不同指令触发的半导体数量、访问内存和缓存等因素会导致不同指令执行时产生不同的功耗特征。

功耗侧信道攻击可以分为以下三种：

- **简单功耗分析 (Simple Power Analysis, SPA)**：通过直接分析设备随时间的功耗曲线，推断CPU执行的顺序或指令。
- **差分功耗分析 (Differential Power Analysis, DPA)**：通过比较不同输入对应的功耗曲线之间的差异，推断出密钥或敏感数据。
- **相关功耗分析 (Correlation Power Analysis, CPA)**：通过建立功耗曲线与已知输入的相关性模型，推断出密钥或敏感数据。

这些方法可以从设备的功耗变化中获取信息，从而对密码系统进行攻击（密码爆破），窃取敏感数据。

电磁侧信道攻击（页码：50）

电磁侧信道攻击利用与数据相关的电流在处理器中的变化导致的磁场变化，通过分析磁场来获取数据。电磁分析类似于功耗分析，都是一种非接触式的攻击方法。

密码芯片在工作过程中会产生不可避免的电磁辐射，辐射的信息和芯片内部的数据存在一定的相关性。攻击者可以分析这些电磁辐射，从中获取敏感数据。

微架构侧信道攻击（页码：51）

微架构侧信道攻击是基于现代处理器的优化技术（如乱序执行和推测机制）的副作用而产生的漏洞。随着处理器性能的提升，与之相关的漏洞也越来越多。Meltdown和Spectre等侧信道攻击就是这种类型的攻击。

这些攻击利用了异常延迟处理和推测错误导致的微架构状态的变化，在架构层级上未显示，但在处理器的微架构状态中留下痕迹。通过隐蔽信道，攻击者可以传输微架构状态的变化到架构层级，从而恢复出秘密数据。

Cache侧信道攻击（页码：52-59）

Cache侧信道攻击利用多核之间的缓存数据共享以及缓存命中和失效所对应的响应时间差异来推测缓存中的信息，从而获取隐私数据。

Cache是存储器层次结构中的一级缓存，包含L1、L2和L3等级，用于存放从主存调入的指令和数据块。Cache具有地址转换部件和替换部件，通过建立目录表来实现主存地址到缓存地址的转换，并在缓存已满时按一定策略进行数据块替换。

Cache侧信道攻击主要包括以下四种方法：

- **Prime-Probe**：通过填充特定的缓存组，并测量读取时间来推测缓存中的信息。
- **Flush-Reload**：通过驱逐缓存中的数据，并测量重新加载的时间来推测缓存中的信息。
- **Evict-Reload**：与Flush-Reload类似，通过驱逐共享内存中的数据块并重新加载来推测缓存中的信息。
- **Flush-Flush**：通过执行flush指令清空缓存中的原始数据，并测量刷新时间来判断原始数据是否被缓存。Flush Flush侧信道攻击技术在整个攻击过程中是不需要对内存进行存取的，因此该攻击技术更加隐蔽

这些攻击方法利用了缓存的工作原理和多核之间的共享特性，通过测量访问时间的差异来推测缓存中的信息，从而获取隐私数据。

硬件防护技术

木马检测技术

木马检测是一种防护措施，用于检测和防御恶意软件中的木马程序。木马检测方法大致可以分为破坏性和非破坏性两类。（P62）

检测挑战

- 选择合适的木马模型。
- 生成测试向量来激活木马或增加侧信道测量中的木马敏感度。
- 消除/校准测试中的环境噪声和测量噪声。

集成电路生命周期（页码：63）

在集成电路的生命周期内的各个阶段都可能存在不可信的组件/人员。一般来说，芯片设计阶段是可信的，因此可以获得标准的设计和测试向量来进行木马检测。制造阶段通常不可信，其他阶段则可能同时存在可信和不可信的情况。

方法分类（页码：64）

破坏性检测

- 化学去封
- 电路扫描
- 逆向分析
- 芯片重构
- 模板匹配

破坏性检测方法费时耗力，不能逐个检测，实用性较低。

非破坏性检测

- 实时监测（在线监测）
- 芯片测试
- 测试向量
- 特征检测
- 结果分析

非破坏性检测方法可以逐个测试，但测试向量不一定全面，检测分析可能不完善。

逻辑测试和侧信道分析（页码：65）

木马检测可以使用逻辑测试法和侧信道分析法。

逻辑测试法侧重于生成并使用**测试向量**来尝试激活可能的木马电路，并观察对于主输出端有效荷载的影响。

侧信道分析法基于在芯片中植入任何恶意电路都会影响某些**旁路信道**的参数值，如漏电流、静态电源电流、动态功耗轨迹、路径延迟特性和电磁辐射。

	逻辑测试	侧信道分析
优点	对小型木马有效	对大型木马有效
缺点	测试生成复杂，大型木马检测具有挑战性	对过程噪声脆弱，小型木马检测具有挑战性

隔离技术（P66）

隔离技术是一种访问控制手段，用于限制用户访问非授权的计算资源。通过硬件隔离技术，可以实现计算资源的共享、安全计算环境和不安全计算环境之间的隔离。以下是几种常见的隔离技术：

存储器隔离（页码：67）

存储器隔离是现代处理器通常具备的一种特性。它通过存储器保护技术、EPT硬件虚拟化技术和存储加密技术来实现。系统使用IOMMU（IO Memory Management Unit）来限制设备对存储器的访问。IOMMU通过重定位寄存器和界地址寄存器来实现，其中重定位寄存器包含物理地址，界寄存器包含逻辑地址。

EPT硬件虚拟化技术（页码：68-69）

EPT（Extended Page Tables）是Intel针对软件虚拟化性能问题推出的硬件辅助虚拟化技术。它提供了一种地址转换机制，将客户机虚拟地址（GVA）转换为客户机物理地址（GPA），然后通过EPT将客户机物理地址转换为宿主机物理地址（HPA）。这两次转换由CPU硬件自动完成，转换效率非常高。

ARM TrustZone（页码：60-71）

ARM TrustZone将CPU内核隔离成安全和普通两个区域。单个处理器内核包含了安全处理器核和普通处理器核，可以以时间片的方式从安全区域和普通区域执行代码。安全核只能访问安全世界的系统资源，而非安全核只能访问普通世界的系统资源。切换安全核和非安全核之间通过SMC（Secure Monitor Call）指令或硬件异常机制的一个子集来实现。

Intel SGX（页码：71-72）

Intel SGX（Software Guard Extensions）是一组CPU指令，可让应用程序创建安全区域（Enclave），这些区域是应用程序地址空间中的受保护区域。即使存在特权恶意软件，SGX可以提供机密性和完整性。SGX可以减少应用程序的攻击面，提供了机密性和完整性，具有低学习曲线和远程认证的特点。

通过隔离技术，可以限制对计算资源的访问，提高系统的安全性和隐私保护能力。这些技术在硬件和软件层面上实现了资源隔离和保护，有助于防止恶意行为和攻击对系统的影响。

密码技术（页码：75）

现代密码技术可以分为对称密钥和非对称密钥两种体系。对称密钥常用于批量数据的加解密，而非对称密钥更多用于密钥交换。此外，还有一种新型的密钥生成电路称为**物理不可克隆函数（Physical Unclonable Function, PUF）**电路，它可以保证密钥的唯一性和不可克隆性。

物理不可克隆函数（PUF）利用在半导体生产过程中自然发生的深亚微米变化，赋予每个晶体管些许随机的电特性。PUF具有唯一性、隐匿性、稳定性和随机性等特点。根据实现方式的不同，PUF可以分为非电子PUF、模拟电路PUF和数字电路PUF。PUF的应用领域包括身份认证、密钥生成和创建信任根。（页码：75-78）

旁路信息隐藏与掩码技术（页码：82-84）

攻击者利用旁路信道的输出来获取足够的信息，以确定和芯片操作相关的敏感数据。为了降低攻击难度，**可以通过增大噪声或减小信号来降低信噪比**。旁路信息隐藏的一种措施是**掩码技术**，它从功能模块的中间节点入手，移除输入数据与旁路信道的相关性。掩码技术可以基于门或模块实现。

模块划分和物理防御 P81

在设计防护措施时，可以将芯片操作中的明文操作区和密文操作区分开，并对芯片不同区域进行物理隔离。除了物理隔离，还需要对片内共享的基础设施进行分离，包括供电基础设施、时钟基础设施和测试基础设施。模块划分以及物理安全是降低攻击者利用旁路信道进行攻击的关键。（页码：80-81）

另外，一些已知的旁路信道攻击包括**Meltdown**、**Spectre**和**VoltJockey**，它们利用了处理器中的旁路信息泄露敏感数据的漏洞。

典型漏洞分析

MOLES (页码: 83)

MOLES是一种利用木马通过侧信道泄漏芯片内部信息（如密钥）的攻击方式，然后利用差分能量分析技术提取密钥。

Meltdown漏洞 (页码: 87)

Meltdown是一种利用乱序执行漏洞的攻击，它可以允许攻击者在任意地址读取数据，包括内核内存。

乱序执行 (页码: 85)

乱序执行是一种现代处理器的执行机制，它允许处理器在指令执行的顺序上进行重排序，以提高执行效率。乱序执行过程包括获取指令、解码后存放执行缓冲区（保留站）、乱序执行指令并将结果保存在一个结果序列中，然后进行重排和安全检查，并将结果提交到寄存器。

攻击过程 (页码: 89)

Meltdown攻击利用乱序执行漏洞来泄露内核内存。攻击者在用户级特权下执行以下指令：

1. 将目标内核地址加载到寄存器rcx和rbx。
2. 对寄存器rax进行清零。
3. 从目标内核地址中读取一个字节并加载到寄存器al中（步骤1）。
4. 将寄存器rax左移12位（相当于乘以4096）。
5. 从rbx和rax的和地址中加载一个qword（8字节）。

攻击过程中的乱序执行使敏感数据泄露到缓存中，然后通过Cache侧信道进行提取。

1. 指令获取解码
2. 乱序执行3条指令，line 4和line 5要等line 3中的读取内存地址的内容完成后才开始执行，line 5会将要访问的rbx数组元素所在的页加载到CPU Cache中
3. 对2的结果进行重新排列，对line 3、line4、line5这3条指令进行安全检测，发现访问违例，会丢弃当前执行的所有结果，恢复CPU状态到乱序执行之前的状态，但是并不会恢复CPU Cache的状态
4. 通过缓存侧信道攻击，可以知道哪一个数组元素被访问过，也即对应的内存页存放在中，从而推测出内核地址的内容

攻击场景

Meltdown攻击的目标是读取内核数据。攻击者以用户级特权运行，并利用乱序执行漏洞泄露敏感数据到缓存中。通过观察缓存集的痕迹，攻击者可以提取出敏感数据。

这种攻击可以影响几乎所有的Intel处理器和ARM Cortex-A75处理器。

Spectre漏洞 (页码: 90-92)

Spectre是一种利用分支预测错误的攻击方式。分支预测是一种处理器的执行技术，它通过在分支指令执行结束之前猜测哪一条分支将会被运行，以提高指令流水线的性能。

攻击过程中，攻击者训练CPU的分支预测单元，使其在运行代码时执行特定的预测。然后，攻击者进行分支预测越权访问敏感数据，并将其映射到缓存中。最后，通过缓存侧信道，攻击者可以通过观察缓存使用情况来窃取敏感数据。

Spectre漏洞的关键步骤类似于Meltdown漏洞。当CPU发现分支预测错误时，会丢弃分支执行的结果，但不会恢复CPU Cache的状态。利用这一点，攻击者可以突破进程间的访问限制，从而获取其他进程的数据。攻击者可以通过在代码中设置故意的分支预测错误来利用Spectre漏洞。

VoltJockey漏洞（页码：93-99）

VoltJockey是一种利用动态电源管理技术（Dynamic Voltage and Frequency Scaling, DVFS）的漏洞。DVFS是一种在满足用户对性能需求的前提下，根据处理器的负载状态动态改变电压和频率，以实现节能的技术。它被广泛应用于现代处理器中的低功耗技术。

攻击者利用VoltJockey漏洞，通过控制DVFS的参数，将处理器的电压设置得较低，导致特定操作中的AES加密函数发生电压故障。然后，利用缓存侧信道监视被攻击函数的缓存使用情况，并在故障注入点引入特定时间的电压故障。最后，通过差分故障分析技术提取AES的加密密钥。

在VoltJockey攻击中，攻击者需要进行一些准备工作，如分析公开的加密函数以找到合适的注入点，分析故障注入的合适电压与时间，并使用NOP指令精确控制时间。攻击者还需要将进程绑定到特定的处理器内核，并使用缓存侧信道监控被攻击函数的缓存使用情况。通过调整故障电压分析、确定临界电压、预备延迟等参数，攻击者可以成功地利用VoltJockey漏洞来窃取AES的加密密钥。

COVID-19家庭检测套件的漏洞（页码：96）

COVID-19家庭检测套件包括棉签拭子、试剂和一个分析仪。用户使用棉签从鼻子或喉咙中收集粘液样本，与测试溶液混合并滴在检测卡上。用户通过手机应用程序发送检测请求，并在大约20分钟内收到检测结果。

然而，该应用存在漏洞，使得一些信息可能被匿名发送到云端，从而可能导致个人隐私泄露的风险。此外，定制PCB板上的LED灯珠和透镜也可能用于照亮测试条和读取结果线，分析仪通知移动应用程序的过程中也存在安全风险。

操作系统安全

为什么存在安全问题 P5-8

1. 现代操作系统是规模庞大的软件系统，现代操作系统是**系统之系统**，各个模块之间的依赖关系复杂
2. 其二，现代操作系统的设计以性能为最主要目标，而非安全性

面向系统攻击的前提 P9

本章中假设攻击者位于操作系统外部；仅能通过正常I/O方式与操作系统下的受害进程进行交互。

- 攻击者可以构造任意输入并接受受害进程输入校验；
- 攻击者无法**直接**读写系统下进程的内存，无法**直接**干预处理器上指令的执行

面向系统攻击的目标 P10

攻击者通过构造恶意输入，使操作系统环境下运行的**进程产生异常行为**：使操作系统下受害进程产生攻击者期望的异常行为的攻击效果被称为进程的控制流劫持。

操作系统基础攻击方案

操作系统内存基础 P14

- Linux 内核为每一个进程维护一个**独立的**线性逻辑地址空间，以便于实现进程间内存的相互隔离

- 这一线性逻辑地址空间被分为**用户空间**和**内核空间**；用户态下仅可访问用户空间，系统调用提供接口以访问内核空间；内核态下亦无法访问用户空间

用户区内存空间包含了6个重要区域：

1. 文本段：进程的可执行二进制源代码
2. 数据段：初始化了的静态变量和全局变量
3. BSS 段：未初始化的静态变量和全局变量
4. 堆区：由程序申请释放
5. 内存映射段：映射共享内存和动态链接库
6. 栈区：包含了函数调用信息和局部变量

区域名称	存储内容	权限	增长方向	分配时间
文本段	二进制可执行机器码	只读	固定	进程初始化
数据段	初始化了的静态、全局变量	读写	固定	进程初始化
BSS段	未初始化的静态、全局变量	读写	固定	进程初始化
堆区	由进程执行的逻辑决定	读写	向高地 址	堆管理器申请内核分 配
内存映射 段	动态链接库、共享内存的映射 信息	内容相 关	向低地 址	运行时内核分配
栈区	函数调用信息与局部变量	读写	向低地 址	函数调用时分配

需要注意的是，上述分区均存在于虚拟地址空间当中，进程可见的地址均为虚拟地址，内存物理地址对进程不可见；虚拟地址需要经过页式内存管理模块才可转换为物理地址，本节提到地址均为逻辑地址（虚拟地址）。

栈区内存的作用 P19

进程的执行过程可以看作一系列函数调用的过程，栈区内存的根本作用：保存主调函数（Caller）的状态信息以在调用结束后恢复主调函数状态并创建被调函数(Callee)的状态信息。保存主调函数状态的连续内存区域被称作栈帧（Stack Frame）；当调用时栈帧进栈，当返回时栈帧出栈；栈帧是调用栈的最小逻辑单元。

密切相关的寄存器 P20

我们关注与函数调用相关的四个寄存器：

3个通用寄存器: ESP（Stack Pointer）记录栈顶的内存地址；EBP（Base Pointer）记录当前函数栈帧基地址；EAX（Accumulator X）用于返回值的暂存。

1个控制寄存器: EIP（Instruction Pointer）记录下一条指令的内存地址。

正常的函数调用流程 P22~29

栈区溢出攻击 P30

若攻击者希望劫持进程控制流，产生其预期的恶意行为，则必须让**EIP寄存器**指向恶意指令。注意到：在函数调用结束时，会将栈帧中的返回地址赋值给EIP寄存器；攻击者可以修改栈帧当中的返回地址，使EIP指向准备好的恶意代码段实现进程控制流劫持。

栈区溢出攻击是一种攻击者越界访问并修改栈帧当中的返回地址，以控制进程的攻击方案的总称。栈溢出攻击有多个分类和变体，但其本质均是对于栈帧中返回地址的修改，导致EIP寄存器指向恶意代码。

1. 返回至溢出数据 P32
2. 返回至库函数 P35

总结 P37

以上简单的栈溢出攻击在现实操作系统环境下几乎无法成功；为防御栈区溢出，已有诸多内存级别的保护机制，例如NX、ASLR、Stack Canary、DEP等将在第二节当中介绍。栈区溢出攻击是被最广泛使用的控制流劫持手段。

基础堆区攻击 P38

正常工作的堆管理器 P39 ~ 43

堆管理器处于用户程序与内核中间地位，主要做以下工作：

响应用户的申请内存请求。向操作系统申请内存，然后将其返回给用户程序；堆管理器会预先向内核申请一大块连续内存，然后过堆管理算法管理这块内存；当出现了堆空间不足的情况，堆管理器会再次与内核行交互

管理用户所释放的内存。一般情况下，用户释放的内存并不是直接返还给操作系统的，而是由堆管理器进行管理；这些释放的内存可以用来响应用户新申请的内存的请求。

堆管理器的缓冲作用显著降低了动态内存管理的性能开销。

堆管理器通常不属于操作系统内核的一部分，而是属于标准C函数库的一部分，根据标准C函数库的实现而采用不同堆管理器。堆管理器的根本区别在于堆管理算法和管理元数据。

堆溢出攻击 P44

面向堆区攻击是一类攻击者越界访问并篡改堆管理数据结构，实现恶意内存读写的攻击。堆区溢出攻击是堆区最常见的攻击方式，这种攻击方式可以实现恶意数据的覆盖写入，进而实现进程控制流劫持。堆区溢出攻击是堆区最常见的攻击方式，这种攻击方式可以实现恶意数据的覆盖写入，进而实现进程控制流劫持。

1. P45 直接覆盖malloc_chunk首部为无意义内容，在堆管理器处理管理元数据时将造成崩溃
2. P46~49 构造堆块重叠：堆块重叠是一种病态堆区内存分配状态，同一堆区逻辑地址被堆管理器多次分配。如右图所示，造成Heap Overlap之后攻击者可以通过写入一个堆块，实现对另一堆块内容的写入；同理，读出被覆盖堆块当中的数据。
3. P50 更加复杂的堆区溢出攻击：利用堆管理其他机制。例如，基于unlink机制的堆区溢出攻击
4. Use-After-Free 51是进程由于实现上的错误，使用已被释放的堆区内存。被free函数释放的堆块内存仍然可以被继续使用，当再次调用malloc分配内存时，会同时有两个指针指向同一堆块造成 堆块重叠。

5. Double-Free 52 进程多次释放统一堆块，被多次释放的堆块将被堆管理器分配多次，最终产生堆块重叠。
6. Heap Over-Read 53 直接越界读出堆区数据，造成信息泄露
7. Heap Spray 堆喷：堆喷申请大量的堆区空间，并将其中填入大量的滑板指令（NOP）和攻击恶意代码；堆喷使用户空间存在大量恶意代码，若EIP指向堆区时将命中滑板指令区，受害进程最终将“滑到”恶意代码。堆喷对抗地址的随机浮动类型的防御方案，并实现了恶意代码的注入。

操作系统基础防御方案 P55

P56 W^X机制：是**写与执行不可兼得**，即每一个内存页拥有写权限或者执行权限，不可兼具两者。当W^X生效时，返回至溢出数据的栈溢出攻击失效，因为无法执行位于栈区的注入的恶意代码。

P57 ASLR：**对虚拟空间当中的基地址进行随机初始化的**保护方案；以防止恶意代码定位进程虚拟空间当中的重要地址；目前在各主流操作系统下均有实现。返回至溢出数据的栈溢出攻击失效。因为恶意代码位于栈区，地址被ASLR随机化，攻击者无法确定并写入恶意代码的绝对内存地址。

P60 Stack Canary 金丝雀：**在保存的栈帧基地址（EBP）之后插入一段信息，当函数返回时验证这段信息是否被修改过。**

P61：内存隔离技术：

SMAP：禁止内核访问用户空间的数据

SMEP：禁止内核执行用户空间代码

SMAP/SMEP 和 W^X 均需要处理器硬件的支持。

防御技术的缺陷

1. 对于Stack Canary，作为Canary的内容可能被泄露给攻击者，或被暴力枚举破解
2. 对于ASLR已有去随机化方案，泄露内存分布信息
3. ROP等进程控制流劫持方案亦可以绕过ASLR、NX的保护机制

高级控制流劫持方案 P64

进程执行的更多细节 P65

Linux下进程可处于内核态或用户态，内核态下拥有更高的指令执行权限（在Intel x86_32下对应ring0）用户态下只拥有低权限（对应ring3）。Linux下内核态与用户态的切换主要由三种方式触发：（1）系统调用（2）I/O设备中断（3）异常执行；其中系统调用是进程主动转入内核态的方法。因而也称系统调用是内核空间与用户空间的桥梁。

共享库机制 P67

在进程的执行过程中，操作系统按需求将共享库以虚拟内存映射的方式映射到用户的虚拟内存空间，位于内存映射段（Memory Map Segment）

与编译器的动态链接机制对应的是编译器静态链接机制，静态链接库在编译时将目标代码直接插入程序。静态链接库无法实现代码共享，因为静态链接不属于共享库机制的一部分。

P71 面向返回地址编程 ROP Gadget

基于栈区溢出攻击，将返回地址设置为代码段中的合法指令，组合现存指令修改寄存器，劫持进程控制流。面向返回地址编程构造恶意程序所需的指令片段被称为 Gadget，Gadget均以RET指令结尾，当一个Gadget执行后，RET指令将跳转执行下一个Gadget。

面向返回地址编程（ROP）可以绕过NX 防御机制，因为虚假的返回地址被设置在代码段（Text Segment），代码段是存放进程指令的内存区域，必有执行权限。

面向返回地址编程（ROP）可以绕过ASLR 防御机制，因为目前ASLR的随机性不强，且依赖模块自身的支持。

ROP 总结

ROP的本质是利用程序代码段的合法指令，重组一个恶意程序，每一个可利用的指令片段被称作 Gadget，可以说ROP是：a chain-of-gadgets。

ROP可以绕过NX和ASLR防御机制，但对于Stack Canary则需要额外的信息泄露方案才可绕过这一防御机制。

ROP使用的Gadget以RET指令结尾；若Gadget的结尾指令为JMP，则为面向跳转地址编程（Jump-Oriented Programming, JOP），原理与ROP类似。

P82 全局偏置表劫持GOT Hijacking

为了使进程可以找到内存中的动态链接库，需要维护位于数据段的全局偏移表（Global Offset Table, GOT）和位于代码段的程序连接表（Procedure Linkage Table, PLT）

程序使用CALL指令调用共享库函数；其调用地址为PLT表地址，而后由PLT表跳转索引GOT表，GOT表项指向内存映射段，也就是位于动态链接库的库函数。

PLT表在运行前确定，且在程序运行过程中不可修改（Text Segment 不可写）。GOT表根据一套“惰性的”共享库函数加载机制，GOT表项在库函数的首次调用时确定，指向正确的内存映射段位置。动态链接器将完成共享库在的映射，并为GOT确定表项。

PLT表不直接映射共享库代码位置的原因有二：

1. ASLR将随机浮动共享库的基地址，导致共享库的位置无法被硬编码
2. 并非动态链接库当中的所有库函数都需要被映射（降低内存开销）

GOT Hijacking (全局偏置表劫持)攻击的本质是恶意篡改GOT表，使进程调用攻击者指定的库函数，实现控制流劫持。

P88 ~ 89具体步骤与 GOT Hijacking 的总结

GOT Hijacking本质上是一种修改GOT表项来实现的控制流劫持；这种攻击方案要依赖于栈区溢出等基础攻击方式才可以实现

这种攻击方案可以绕过NX与ASLR防御机制

目前已有RELRO（read only relocation）机制，可将GOT表项映射到只读区域上，一定程度上预防了对GOT表的攻击

P90 虚假vtable劫持Fake vtable Hijacking

虚假Vtable劫持攻击的核心是：篡改文件管理数据结构中的vtable字段，通过把vtable指向攻击者控制的内存，并在其中布置函数指针。

P93 两种具体实现

1. 直接改写vtable指向的函数指针，可通过构造堆块覆盖完成
2. 覆盖vtable字段，使其指向攻击者控制的内存，然后在其中布置函数指针

高级操作系统基础防御方案 P94

P95~99 控制流完整性保护

CFI防御机制的核心思想是限制程序运行中的控制流转移，使其始终处于原有的控制流图所限定的范围
主要分为两个阶段：

1. 通过二进制或者源代码程序分析得到控制流图 (CFG)，获取转移指令目标地址的列表
2. 运行时检验转移指令的目标地址是否与列表中的地址相对应；控制流劫持会违背原有的控制流图，CFI 则可以检验并阻止这种行为

对于CFI的一系列改进：原始的CFI机制是对所有的间接转移指令进行检查，确保其只能跳转到预定的目标地址，但这样的保护方案开销过大。

目前CFI方案的平均情况下，额外开销为常规执行的2-5倍，距离真实部署仍然存在比较大的距离。目前已经提出了大量的CFI的方案，但其中部分方案存在安全问题而失效，或者无法约束开销而彻底不可用。

指针完整性保护 P101

与CFI提出动机相同：CPI的提出动机仍然是因为ASLR和DEP等内存保护无法防御ROP等进程控制流劫持方案。CPI希望解决CFI的高开销问题，其核心在于控制和约束指针的指向位置。

信息流控制 P103

一种操作系统访问权限控制方案（Access Control），即便程控制流被劫持，IFC可以保证受害进程无法具备正常执行之外的能力；例如，访问文件系统中的密钥对，调用操作系统的网络服务。信息流控制解决的根本问题是访问控制。信息流控制缺陷是：IFC是否生效严重依赖于配置的正确性；IFC的三要素：权限、属性、约束都需要具体问题具体分析得到。

I/O子系统保护 P107

针对I/O子系统的攻击的本质是：发掘通讯协议中的漏洞。外设I/O因为其功能复杂多样，一直是操作系统安全问题的“重灾区”。

需要时刻牢记，内核协议栈是操作系统的组成部分之一，因而面向网络当中端节点的攻击方案的本质均是面向操作系统网络I/O子系统的攻击。例如，TCP侧信道攻击发掘内核协议栈当中的设计或者实现上的缺陷，是对操作系统网络I/O的攻击方案，需要借助网络入侵检测系统进行保护。

TCP/IP 协议栈安全

TCP/IP 在网络中所处的位置：

TCP/IP协议工作原理 P11

纵向横向传递 P11-19

局域网中的client远程访问Server时，在数据传递处理的每一步，都有可能产生安全问题：DNS劫持，ARP污染，嗅探监听，地址伪造、路由劫持，TCP连接劫持、DoS攻击。

链路层安全 P19

链路层功能 P20

功能主要有：将数据组合成帧；控制帧在物理信道上的传输，包括处理传输差错，调节发送速率；提供数据链路通路的建立、维持和释放的管理。

ARP欺骗与污染 P21~29

ARP欺骗（ARP spoofing），又称ARP污染（ARP poisoning），是针对以太网地址解析协议（ARP）的一种攻击技术；可让攻击者成为中间人（Man-in-the-Middle, MITM），获取局域网上的数据包，甚至可篡改数据包，同时迫使受害主机无法正常接收报文。

安全问题：在ARP回复时，主机A并不会验证ARP回复包的真实性。由此引出一个局域网攻击方式

ARP欺骗：恶意主机C，企图冒充B，欺骗A

网络监听与嗅探 P33

网络嗅探（Sniffers）是一种网络流量数据分析的手段，常见于网络安全攻防技术使用，也有用于业务分析领域。嗅探所使用的工具称为“嗅探工具”，也称为“数据包分析器”，还有别称为“嗅探器”、“抓包工具”等。常见的开源嗅探工具包括TCPDUMP，WIRESHARK等。

网络嗅探具有**两面性**：

- 网络管理员使用嗅探器，通过捕获分析网络流量，进行高效的网络管理
- 恶意攻击者使用嗅探器，攫取网络中的大量敏感信息，进行网络攻击

恶意的网络嗅探，取决于攻击者的位置和攻击能力：

- 共享网络传输链路场景下，攻击者的恶意嗅探监听
- 攻击者控守了网络设备，对流经的流量进行拦截嗅探

防御方法：

洋葱路由 P36-39：洋葱路由网络的主要目的是保护用户的隐私和匿名性，使得在网络上进行监听和嗅探变得更加困难。通过在多个洋葱路由器之间进行多重加密的数据转发，洋葱路由网络可以有效地防止攻击者通过监听和嗅探获取用户的信息和通信内容。即使攻击者能够捕获到数据包，由于数据的多重加密，攻击者难以解密数据内容或者追踪数据包的发送者和接收者。

网络层安全 P35

网络层功能：分组与分组交换，路由，网络互联，网络连接复用，差错检测与恢复，服务选择，网络管理，分片与重组

源地址假冒攻击 P38

互联网在设计之初，对分组的源IP地址并不进行合法性验证，因此恶意的攻击者很容易篡改或伪造分组的源IP地址。网络层的源IP地址假冒攻击（IP spoofing），是最常见的一种针对IP协议的攻击。通常，IP spoofing也是进行复杂网络攻击，如会话劫持、DNS污染、TCP劫持等攻击，的能力基础和先决条件。攻击者在发送数据包时伪造源IP地址，从而隐藏其真实身份或者导致网络流量被错误地发送到其他目标。随着真实地址、真实身份等技术标准的出现，这一问题得到了缓解，但要彻底解决这一攻击威胁，仍需持续的技术推动和投入。

IP 分片攻击 P46

当互联网上的两台远程主机进行数据传输时，如果传输路径上的各跳间，存在不同的链路**最大传输单元**（Maximum Transmission Unit, **MTU**），那么可能会发生IP分组分片（IP fragmentation），以满足链路MTU的要求。

P46 ~ 48 分片攻击原理

IP分组的这种先分片、再重组的机制，为攻击者暴露出了IP层的一个攻击面。分组的碎片化发生在**源主机**或**中间路由器**上，而重组往往都在目的接收端发生。根据不同的攻击效果，IP分片攻击可以分为3种：

1. 拒绝服务攻击 P49~51：攻击者发送大量的IP分片，使得目标主机在重组分片时耗尽资源，从而导致拒绝服务
2. 污染攻击 P52~61：一种更高级的攻击方式，原始报文被分片、传输过程中，攻击者伪造一些分片，注入到正常分片流中，篡改原始报文内容
3. 安全策略逃逸 P62~64：通常网络防火墙会对IP分片进行重组，然后审查其中的恶意载荷。但防火墙和接收端主机往往存在重组策略不一致现象，即当分片出现重叠时，二者可能会采用不同的覆盖策略。因此，攻击者可以通过设置合理的分片偏移，逃逸防火墙的审查，但形成对接收端的破坏。

为了避免IP分片，一些相关技术标准也陆续被提出。例如，路径MTU发现（Path MTU Discovery, PMTUD）机制，该机制用于确定两台IP主机间的路径MTU。TCP协议有MSS选项，可以通过调整MSS、适合路径MTU，从而避免分片。

针对 UDP 的 IP 分片 P72

UDP是面向事务的，与IP层是松耦合的，UDP报文的大小直接由应用给定，因此UDP不能依据路径MTU大小、对传输层报文进行适应性调整，即针对UDP报文的IP分片很难避免。新的RFC标准提出，应用层和UDP协议合作、发现路径MTU，从而避免IP分片，但这些新标准和机制的实际部署，仍需很长时间。

安全防御一：IPSec协议 P80

IPSec (Internet Protocol Security, RFC4301)，即Internet协议安全性，是一种开放标准的框架结构，通过对IP协议的分组进行加密和认证，来保护基于IP协议的网络传输。在 IPv4 中，IPSec 的使用是一个可选项，在 IPv6（RFC 6434）中，初始设为必选项，未来随着 IPv6 的进一步流行，IPSec 可以得到更为广泛的使用。

IPSec协议工作在TCP/IP协议的第三层，使其在单独使用时适于保护基于IP协议的所有上层协议。

认证机制 AH P76

提供数据源认证、数据完整性校验和防报文重放功能，它能保护通信免受篡改，但不能防止窃听，适用于传输非机密数据。AH 的工作原理是在每一个数据包上添加一个身份验证报文头，此报文头插在标准IP包头后面，对数据提供完整性保护。可选择的认证算法有MD5（Message Digest）、SHA-1（Secure Hash Algorithm）等。

加密机制 ESP P77

提供加密、数据源认证、数据完整性校验和防报文重放功能。ESP的工作原理是在每一个数据包的标准IP包头后面添加一个ESP报文头，并在数据包后面追加一个ESP尾。与AH协议不同的是，ESP将需要保护的用户数据进行加密后再封装到IP包中，以保证数据的机密性。常见的加密算法有DES、3DES、AES等。同时，作为可选项，用户可以选择MD5、SHA-1算法保证报文的完整性和真实性。

在实际进行IP通信时，可以根据实际安全需求同时使用这两种协议或选择使用其中的一种。同时使用AH和ESP时，设备支持的AH和ESP联合使用的方式为：先对报文进行ESP封装，再对报文进行AH封装。封装之后的报文从内到外依次是原始IP报文、ESP头、AH头和外部IP头。

IPSec 的两种工作模式 P79~80

- 传输模式：只是传输层数据被用来计算AH或ESP头，AH或ESP头以及ESP加密的用户数据被放置在原IP包头后面。主要用于主机和主机之间，端到端通信的数据保护。封装方式：不改变原有的IP包头，在原数据包头后面插入IPSec包头，只封装数据部分。
- 隧道模式：用户的整个IP数据包被用来计算AH或ESP头，AH或ESP头以及ESP加密的用户数据被封装在一个新的IP数据包中。主要用于私网与私网之间，通过公网进行通信，建立安全VPN通道。封装方式：增加新的IP（外网IP）头，其后是IPSec包头，之后再将原来的整个数据包封装。

网络入侵检测系统 IDS P81

入侵检测系统（Intrusion Detection System, IDS）是一种网络安全设备或应用软件，可以监控网络数据传输（NIDS），或者主机系统行为（HIDS），检查是否有可疑活动或者违反预定义的安全策略。从主要技术途径上区分，主要分为两类：

基于特征匹配的 IDS P83

基于专家知识或经验，预定义攻击行为特征或规则，然后对网络流量或主机行为进行匹配，如果匹配成功，则判定为攻击事件。优点：误报率低，检测速度快。缺点：无法识别未知攻击，存在漏报。

基于异常检测的 IDS P84

首先通过学习建模的方法，构建网络或主机的正常行为基线。然后结合当前网络或主机态势进行判断，如果网络或主机的行为偏离该基线，则判断发生了入侵或攻击，进行告警。优点：可以识别未知攻击。缺点：存在误报，检测速度慢。

路由协议安全 P86（这一部分在第八讲）

路由（routing）是通过互联的网络，把信息从源地址传输到目的地址的活动。路由发生在TCP/IP协议的第三层即网络层。路由引导分组转送，经过一些中间的节点后，到它们最后的目的地。Internet被划分为多个**自治系统**（AS），每个自治系统可以制定自己的路由策略。自治系统内部的路由器通过**域内路由协议**彼此交换路由信息；自治系统边界路由器通过**域间路由协议**交换路由信息。

域间路由安全 P88

AS（恶意攻击者）宣告一个实际上并不控制的IP地址前缀，进行虚假路由宣告。如果虚假路由宣告未被有效过滤，就有可能发生路由劫持攻击。

BGP劫持成功，恶意攻击者的宣告必须满足以下条件：宣告比以前其他AS更具体的IP地址范围；提供一条通往目标IP地址块的较短路径。需要注意的是，要发生BGP劫持，通常需要AS的运营商权限。目前，针对BGP协议暴露出的安全问题，已有相关的安全标准被提出，但这些标准技术的实际部署，还需要很长一段时间。

域内路由安全 P92

OSPF协议是目前使用最广泛的域内路由协议，因此针对域内路由的威胁攻击，也主要是针对OSPF协议展开。

OSPF攻击基本原理：攻击者通常假装成合法的域内路由器，伪造多个OSPF协议的LSA报文广播出去，迷惑其他路由器路由表的计算。

- P93: Fight-back安全机制：现有关于OSPF协议的威胁破坏，其核心就是如何在伪造LSA的时候，避免触发源合法路由器的fight-back机制。
- P94~96: PPeriodic injection 攻击
- P97~98: Disguised LSA 攻击

目前，随着OSPF协议的不断改进，之前的安全漏洞大多已被修补，其安全性也已大为改善。

传输层安全 P100

传输层主要负责提供不同主机应用程序进程之间的端到端的服务。传输层的基本功能如下：分割与重组数据，按端口号寻址，连接管理，差错控制和流量控制、纠错的功能。

拒绝服务攻击 P101

指任何对服务的干涉，使得其可用性降低或者失去可用性。例如一个计算机系统崩溃，或其带宽耗尽，或其硬盘被填满，导致其不能提供正常的服务，就构成拒绝服务。

最常见的DoS攻击有**计算机网络带宽攻击**和**连通性攻击**：带宽攻击指以极大的通信量冲击网络，使得所有可用网络资源都被消耗殆尽；连通性攻击指用大量的连接请求冲击计算机，使得所有可用的操作系统资源都被消耗殆尽。

SYN Flooding P103~105

针对TCP协议主要的DoS攻击方式之一，通过发送大量伪造的TCP连接请求，使被攻击方资源耗尽，CPU满负荷或内存不足，一般发生在TCP协议中的三次握手（Three-way Handshake）阶段。

TCP 劫持攻击 P106

相比较针对TCP的DoS攻击，针对TCP的劫持攻击是一种更高级、更隐蔽、危害也更大的攻击方式。TCP劫持攻击是指，一个TCP连接外的非法攻击者（即Client和Server之外的一个Attacker），将自己伪造的TCP报文，注入到TCP连接双方的合法数据流中，进而对连接进行攻击破坏。

包括：1. 伪造控制报文、如RST报文等，恶意阻断该连接；2. 伪造数据报文，污染数据流。

TCP 劫持原理 P107~114

1. 三种攻击攻击模式 P109~110
2. 基于challenge ACK侧信道漏洞的TCP连接劫持 P111~112
3. 基于IPID侧信道漏洞的TCP连接劫持 P113

TLS协议 P115~119

传输层安全性协议（Transport Layer Security, TLS）及其前身安全套接层协议（Secure Sockets Layer, SSL）是一种广泛采用的安全性协议，旨在促进 Internet 上通信的隐私和数据安全性。

TLS 应用 P120

TLS 协议通常在TCP等传输层协议之上运行，提供以下三个安全功能：

加密：阻止第三方对传输数据的窃听

身份验证：确保交换信息的各方是他们声称的身份

完整性：验证数据是否伪造而来或未遭篡改过

广泛应用：TLS是目前采用最广泛的一种安全性协议，应用于多个网络场景，如：对Web 应用程序和服务
器之间的通信进行加密保护，还可用于电子邮件、消息传递和 IP 语音（VoIP）加密等。

网络安全共性分析 P122

通过梳理分析上述针对网络协议栈的攻击，可以发现这些攻击基本都具备2个共性特征：1. 攻击者可以进行**身份欺骗**，伪装成网络通信的一端；2. 攻击者可以进行**推理猜测**，成功构造出可被通信对端接受的数据报文。

两个共性特征映射到实际的网络系统中，本质上是利用了当前协议栈中的两个基础安全缺陷：

1. P124：一是网络地址缺乏足够的真实性验证，可以被恶意伪造；
2. P125：二是网络系统在实现和部署过程中，随机化程度不高，致使网络的状态信息可被恶意攻击者预测推理。

协议栈安全的基本防御原理 P126

1. **基于真实源地址的网络安全防护**：网络地址验证，约束主机只能使用预分配给自己的IP地址发送数据，进行网络通信
2. **增强协议栈随机化属性**：另一方面也可以通过增强网络协议栈的随机化属性，提升攻击者猜测、推理出目标网络状态信息的难度。代表性的包括**移动目标防御**（Moving Target Defense, MTD），**拟态防御**等。

第8讲 互联网路由安全 总结

 image-20240514201256284

第一节 层次化路由体系结构

路由的定义：通过互联的网络，把分组从原地址传输到目的地址。发生在TCP/IP协议的第三层网络层。
路由引导分组转送，从源节点出发，经过一系列中间路由器后，到达最终的目的节点。

层次化路由

各个路由不是平等的，而是有层次的。

将路由器按区域组织，得到自治系统 autonomous system(AS)，然后再有 intra-AS（同一AS内的路由器运行同一种路由协议）和 inter-AS 路由协议。

网关路由器：

- 与边界网关路由器间执行 inter-AS 路由协议
- 与同一AS内的其它路由器执行 intra-AS 路由协议


域内路由

域内路由协议 OSPF: Open Shortest Path First Protocol (分布式的)

 image-20240514202838603

域间路由

域间路由协议 BGP

 image-20240514203627066

两种类型的BGP邻居关系：

- eBGP：不同AS之间建立的BGP关系
- iBGP：同一AS内部建立的BGP关系

【此处看得粗糙，之后回看】

第二节 路由安全问题

域内路由安全

针对域内路由的威胁攻击，也主要是针对OSPF协议展开。

OSPF攻击基本原理：攻击者通常假装成合法的域内路由器，伪造多个OSPF协议的**LSA**报文广播出去，迷惑其它路由器路由表的计算。其核心就是如何在伪造LSA的时候，避免出发合法路由器的**fight-back安全机制**（会发送纠正报文）。

Periodic injection 攻击

Periodic injection 攻击：

 image-20240514205037766

Disguised LSA 攻击


 image-20240514205541245

 image-20240514205643973

域间路由安全

路由劫持

 image-20240514210411555

 image-20240514210636025

因为存在路由泄漏问题

第三节 路由源验证

路由防劫持

BGP缺乏内建安全机制，没有检查路由是否正确；无法有效抵御恶意攻击（路由劫持/路由泄漏），且无法识别配置错误。

所以如何避免路由劫持/泄漏与配置错误呢？

- 可靠的IP前缀和自治系统的绑定关系和策略
- 全球互联网路由选择的稳定性、一致性和安全性
- 路由过滤（如BOGON）、防止意外或恶意的后缀声明

BOGON 主要指不应出现在公共互联网的码号资源（私有地址、未分配地址、保留地址）

IRR: Internet Routing Registry

全球路由策略分布式数据库

【粗略地看了一下，没有仔细看】

RPKI

RPKI 是一种数字证书系统，对码号资源提供密码学可验证担保。

ROA：某个IP地址所有者将IP地址授权给某个AS

image-20240514212350446

ROA对象：经过加密签名的对象，在RPKI框架中用于声明AS被授权发起的IP前缀。

Max Length：用于指定本次授权可以宣告的最大前缀长度

【粗糙地看了下，之后详细看】

MANRS Observatory

MANRS Observatory 是一个公共平台，提供路由安全相关信息。

第四节 路径验证

注意，路径验证是和路由源验证相并列的，是两种验证范式。一个是源头，一个是路径。

image-20240514213221617

为什么需要路径验证：

场景：当一个用户正在网上冲浪，正常访问各种互联网资源。此时，一名攻击者进行了攻击行动.....此时，攻击者伪装自己为正确的源ASN，**绕过路由源验证**。由于其具有更短的AS PATH长度，因此最佳路由将指向恶意攻击者。

BGPsec

RPKI提供资源分配认证，用于源验证；BGPsec则用于路径验证。

旨在解决域间路由的AS路径篡改问题。



未部署 BGPsec 与部署了 BGPsec 之间的对比：



FC-BGP

针对 BGPsec 的局限性进行改进：



FC 指的是 Forward Commitment，具体见下：



见 P133 -142

DNS安全

域名系统（DNS）位于协议栈**应用层**，为互联网提供核心服务，包括web页面访问，收发邮件等互联网应用通过DNS查询IP地址后获取资源，已成为互联网关键基础设施

作为一个分布式数据库，域名系统使得任意联网计算机能够通过域名访问互联网；灵活的扩展性和优异的解析性能，能够持续高效地支持数亿规模的域名解析。互联网渐进式演进发展模式决定了域名系统中大量安全威胁将长期存在，不可能通过重新设计的途径解决。

DDoS 攻击（P4出现）：DDoS（分布式拒绝服务，Distributed Denial of Service）攻击是一种试图使网络资源不可用的恶意行为。通过大量分散的计算机同时向目标系统发送大量请求，超出其处理能力，从而导致系统崩溃或无法正常提供服务。

DNS 概述：P9

DNS 演进 P10

网络空间两套命名体系：用于路由寻址的IP地址和便于人类记忆的域名（Domain Name）

域名系统（DNS）功能：实现域名与IP间转换

DNS域名结构：P12

域名系统采用层次化树形结构：树形最顶层为根（Root），进一步划分顶级域（TLD），顶级域管理机构授权给二级域名（SLD），层次化授权行为最多可迭代127次

DNS区域组织形式： P13

一个域名没有被分割成子域或包含了子域全部数据，则域名和区域相同（像下图中的 `cn` 与 `beijing`）

一个域名被分割成子域，每个子域有自己的区域时，域名和区域具有不同的意义（像下图中的 `cn` 与 `shanghai`）

权威域名服务器 P14

每个DNS区域的权威域名服务器，发布关于该区域的信息，并响应DNS查询请求

权威域名服务器可以配置主从服务器，主服务器存储所有区域记录的记录，而从服务器使用自动更新机制维护主记录的副本

DNS使用及解析过程 P17

1. P19~21 DNS使用 & 请求过程：并不是每一次域名解析都完成整个查询流程：按照域名空间层次化结构自上而下应答，本地DNS - Root - 顶级域 - 二级域名（第18页）
2. DNS迭代查询过程交互信息格式： P22~26
3. DNS 反向查询 P27

实际请求过程中，并不是每一次域名解析都要完成整个查询流程。

以访问thucsnet.com为例，客户端去本地DNS服务器的查询过程为递归查询，本地DNS服务器获取最终域名对应IP的过程为迭代查询。

一个实际的DNS域名解析过程包括：

- （1）客户端查询本机缓存，如缓存没有，则查询hosts文件，如果找到该域名IP地址，直接读取该地址。
- （2）如hosts记录里不存在，则查询指定本地DNS服务器。
- （3）本地DNS服务器查找本地DNS服务器缓存，有结果就返回。
- （4）如没有结果，则查找根服务器，如根服务器查找无结果，则会告知本地服务器.com顶级域位置。
- （5）本地服务器去.com顶级域查找，.com顶级域查找无结果，则告知本地DNS服务器本地无记录，但xy.com应该会知道。
- （6）本地DNS服务器从xy.com获取结果，并向客户端返回查询结果，此时可在本地DNS服务器缓存一份结果，方便再次查询时直接获取。

DNS攻击 P29

攻击原因：1. 客观上协议设计的不完美；2. 主观上基于利益驱动，攻击者不断挖掘漏洞

共同特征：1. 针对明文传输和无身份认证的实体进行欺骗性攻击；2. 寻找并突破域名间复杂依赖关系，实现对域名服务器攻击；3. 针对防护措施不足的服务器发起拒绝服务攻击。

DNS 各个层面的攻击 P30

用户主机；本地 DNS 服务器；互联网 DNS 服务器。

操控用户机 P31

1. 修改/etc/resolv.conf: 将主机DNS服务器修改成攻击者控制的服务器, 攻击者回复恶意内容
2. 修改/etc/hosts: 直接修改地址与域名映射关系, 修改为攻击者控制的IP地址
3. 嗅探回复: 监听主机发出的DNS查询请求, 注入恶意回复

本地缓存中毒攻击 P32~35

伪造DNS回复, 攻击者需要知道请求中的一些参数, 如UDP源端口号、请求交易ID、请求问题等; 由于UDP包没有加密, 攻击者可以在局域网直接捕获并解析请求。

1. 只针对回复部分伪造, 影响面: 一个主机名; 在伪造回复中, 把主机名映射到攻击地址, 告诉本地DNS服务器域名对应地址是攻击者的计算机。
2. 针对授权部分攻击, 影响面: 整个域; 将ns.attack.com放在授权部分, 查询目标域内任何一个主机名时, 本地DNS服务器把请求发给ns.attack.com。(NOTE: 其实没太懂这个)

远程缓存中毒攻击 P36

远程缓存中毒攻击—

主动访问不存在的域名, 这种域名在dns cache server没有缓存, 从而不存在TTL约束

- 1.攻击者向Apollo询问一个随机产生的如abfjsdf.example.com的域名
 2. Apollo向权威域名服务器请求
 - 3.攻击者发送大量回复, 猜测源端口号和交易ID
- 猜对了, 则本地DNS服务器会接受其中的IP和NS记录, 污染Apollo的缓存
- 4.猜错了, 重新开始用另一个随机产生的域名查询实施新一轮攻击

攻击难点: 由于不能嗅探DNS请求, 很难获取两个数据: UDP16bit的头部端口号, DNS头部的16bit交易ID。远程攻击者猜测准确的概率为 $1/2^{32}$, 成功概率极低。

实现攻击的前提: 本地DNS服务器发起域名查询请求。

 image-20240603105247238

解决方案: 源端口随机化, 域名大小写增加攻击者猜测难度等

- 远程缓存中毒攻击: 侧信道攻击
- 远程缓存中毒攻击: 基于IPID的攻击
- 远程缓存中毒攻击: 基于交易ID的攻击

如何伪造一个被受害者接收的应答包 P38

接受欺骗应答面临的约束:

1. 缓存时限约束: 域名不能在缓存中
2. 应答包匹配约束: 端口号、交易ID匹配
3. 时间约束: 伪造包比正常DNS应答快

P39 构建回复包头；P40 包头参数猜测；P41 构建回复包负载；P42 侧信道攻击；P43 基于IPID的攻击；P44 基于交易ID的攻击；幽灵域名 P46~47 恶意域名被删除后，利用DNS漏洞继续存活。

P47~49 来自恶意DNS服务器的污染

在附加部分伪造数据，在授权部分伪造数据，反向查找回复部分伪造。

拒绝服务攻击 DDoS：P50

如果攻击者可以成功攻破root区域的服务器，则整个互联网将会崩溃。但是由于root域名服务器基础设施采用分布式部署方式，很难被全部攻破。

DNS攻击预防策略

通过非对称加密验证身份 - DNSSEC：P55~60

DNSSEC引入公钥加密/认证体系，通过签名提供端到端数据真实性和完整性保护。部署DNSSEC的权威域名服务器对其区域文件中资源记录用私钥进行数字签名。接收方用公钥验证签名，判定域名解析结果是否在传输过程被篡改。

一台支持dnssec的递归服务器向支持dnssec的权威服务器发起paypal.com的A记录请求，它除了得到A记录以外还得到了同名的RRSIG记录，其中包含了paypal.com这个ZONE的权威数字签名，它使用paypal.com的私钥来签名。为了验证这一签名是否正确，递归服务器再次向paypal.com权威查询其公钥，即请求paypal.com的dnskey类型的记录。递归服务器就可以使用公钥来验证收到的A记录是否是真实且完整的。但是注意：这种状态下，这台权威服务器可能是假冒的，递归服务器请求这台假冒的权威服务器，那么对于解析结果的正确性和完整性的验证上认为是正确的，但其实这个解析结果是假冒的，怎么发现？DNSSEC需要一条信任链，即必须要有一个或者多个相信的公钥，这些公钥被称为信任锚。

DNSSEC 的不足 P60~62

1. 经济因素制约其部署，大量签名和验证带来严重的负载，影响效率，验证能力取决于递归服务器以及客户端是否对数字签名记录做校验。
2. 离线存储与频繁使用密钥，私钥的安全无法保证；没有提供机密性和一致性检验，无法抵御重放攻击；只提供单向认证，没有对客户的认证，不能解决缓存中毒；没有提供DoS防护。
3. 引入了新的实现和配置错误。
4. 用 DoT / DoH 改进 P61~62

DNS-over-TLS 协议直接使用传输层安全协议（Transport Layer Security，TLS）对数据执行加密操作，保证了域名协议交互中信息的完整性与机密性。此外，为使DNS 加密流量能够与传统DNS 流量相互区分，DNS-over-TLS 使用了853 专用端口传输数据。

DNSSEC用DNS区域层次结构提供信任链，TLS/SSL协议依赖公钥基础设施（PKI），这其中包括许多证书授权中心（CA）

DNS-over-HTTPS 协议则是采用一种完全颠覆式的域名系统架构，不与现有域名系统相兼容，采用HTTPS 信道传输域名协议数据，将控制平面信息混杂在数据平面之中。

DOT使用了新的端口，虽然加密了，但是可能被防火墙阻止。

DOH使用https协议，端口仍是443，不会被阻止，和DOT一样也是加密隧道

基于系统管理：P63~64

1. 规范并梳理DNS配置过程中出现的漏洞
2. 限制用户在短时间内发起大量DNS查询
3. 增加端口猜测难度，如用于查询的UDP端口不再是默认的53，而在UDP端口范围内随机选择(排除预留端口)
4. 分布式部署、恶意流量过滤等方案
5. 严格检查：附加区记录和问题区问题不在同一域管辖，则不会采信，防范恶意权威DNS虚假记录污染缓存

新型架构设计 Blockstack：P65~66

Blockstack 旨在建立一个去中心化的域名系统及公钥基础设施

典型案例分析

Kaminsky攻击：P68~72

恶意人员发送随机查询请求到DNS服务器，抢在权威应答前伪造应答包发送给服务器，修改授权资源记录。关键：**在权威区和附加区实施欺骗。**

根本原因

- 缺乏端节点身份和发布内容验证
- 数据未采用加密传输
- 协议自身检查采信机制不足

防御策略

- 针对协议内容和通信实体设计可靠的验证方式
- 通过加密等策略确保对用户身份和发布内容进行鉴别，实现有效防御

恶意服务器回复伪造攻击：P73~76

攻击者可依赖目标权威服务器或 DNS 软件漏洞，控制特定权威域名服务器，篡改区域文件中的授权数据，形成恶意服务器回复伪造攻击，成功攻破权威域名服务器的难度较大，实际中可行性并不高

利用域名系统冗余的架构设计使很多域名之间存在错综复杂的解析依赖，通过控制其中一环，逐步实现劫持特定域名权威服务器

根本原因

- 域名系统存在复杂的解析依赖
- DNS管理方面缺乏验证配置内容能力，存在错误输入的记录

防御策略

- 规范域名应用管理，减少使用环节引入的安全威胁
- 对配置内容建立审核机制，确保每一条域名记录的正确性，从根上防止接受恶意信息

拒绝服务攻击：P77

向被攻击的服务器发送大量域名解析请求，给服务器带来了很大负载，超过一定数量造成 DNS服务器反应缓慢甚至停止服务

攻击者可以通过拒绝服务攻击使得整个国家的因特网受到严重威胁

攻击者通过控制大量用户发起DDoS攻击，受害域名服务器可以通过分布式部署的方式，提升自己的处理能力

根本原因

- 大量恶意请求到达被攻击服务器并被攻击服务器接受

防御策略

- 安全的网络体系架构，如采用真实网络地址保障每一台接入网络计算机的安全性，建立快速DDoS溯源机制
- 专用DNS请求过滤系统，针对DNS数据包进行恶意流量进行过滤；分布式部署降低攻击对性能的影响

真实源地址验证

设计背景 P67

IP地址唯一标识互联网中每台设备，在通信中充当发件人和收件人地址。

伪造源IP地址是非常简单的，构造IP数据包头时可以随意填一个IP地址

网关在进行数据包转发时，**不会验证**数据包中源IP地址是否来自网关所在的局域网中

基于 IP 地址伪造的攻击 P71

不伪造特定地址

目的是隐藏自身信息，使得目的主机即使被攻击，也无法追溯到攻击源，避免网络审查，不伪造特定地址不是不伪造地址。

攻击者随机伪造许多IP地址，同时向目的主机发送服务请求；目的主机的资源因为大量请求而占满，无法再响应其他请求，甚至直接崩溃。

伪造特定地址

基于特定IP地址和目的主机的信任关系，伪造特定IP地址取得目标主机的信任以执行恶意指令或获取机密信息。

利用DDoS攻击使被攻击主机暂时停止响应目的主机

猜测出被攻击主机和目的主机之间连接标识信息，达到与目的主机连通

向目的主机发送恶意脚本执行恶意指令，实现破坏目的主机，获取机密信息，甚至控制目的主机

危害严重 P?

IP协议是互联网的核心底层协议，源IP地址真实性的缺失影响到互联网体系结构的各个层面。基于IP的上层协议使用IP地址这个并不安全的标识作为通讯对方的标识，因而只要伪造了源地址，相应地就欺骗了这些协议，这使得伪造源地址攻击的能力超出网络层范围，危害到其它层的协议。

利用源地址伪造的手段，网络攻击的发起者可以隐匿自己的身份和位置，逃避法律的制裁。

基于真实地址的网络计费、管理、监控和安全认证等都无法正常进行，对互联网基础设施和上层应用都造成了严重的危害。

现有防御方法 P73

防御种类	方法	简介	缺点
数据包签名	SPM、Pasport, StackPi、Base等	在IP包头中的ToS或其它较少使用的选项字段加入用户身份鉴别标签	假冒者可以学习标签的添加方法，从而逃避验证
数据包签名	HIP	修改用户终端主机协议栈，在IP和传输层之间添加“主机标识层”	端系统的修改、应用存在着实际困难
出流量源地址检测	Ingress Filtering	路由器根据出流量源IP是否在该路由器所属网络内进行过滤	配置过于复杂
出流量源地址检测	uRPF	根据数据包的源地址反查路由表，判断转发端口是否与数据包的入端口一致	无法防止同方向上的地址假冒，同时路由的非对称性也可能导致假阳性的误判
出流量源地址检测	SAVE	通过为路由器提供拓扑、时钟等附加信息设计整套数据包验证及其路由机制	协议过于复杂并且需要修改用户主机协议栈

现有方法的缺陷 P74

现有源地址验证技术相互独立，只能部分地解决源地址验证的问题，没有形成一个完整的覆盖整个网络的整体结构

- 算法复杂，协议开销大
- 缺乏部署激励
- 完备性不足
- 可扩展性不足

真实源地址验证体系结构 SAVA P75

SAVA体系结构的三层结构包括接入子网层、自治系统内部层和自治系统间层。

1. 接入子网层：该层位于 SAVA 体系结构的最底层，负责对直接接入网络的主机进行源地址验证。在这一层，SAVA 设备会绑定主机和其IP地址，并验证从主机发出的数据包是否使用了正确的源地址。这一层的作用是防止本地网络中的源地址伪造攻击，确保只有合法的主机才能发送具有正确源地址的数据包。
2. 自治系统内部层：该层位于 SAVA 体系结构的中间层，负责在自治系统内部进行源地址验证。在自治系统内部的路由器上进行源地址验证，以确认数据包的源地址确实来自预期的接入点。这一层的作用是防止自治系统内部的源地址伪造，确保数据包的源地址符合预期的接入点。
3. 自治系统间层：该层位于 SAVA 体系结构的最顶层，负责在不同自治系统之间进行源地址验证。在自治系统间的路由器上进行源地址验证，以确认数据包的源地址确实来自预期的自治系统。这一层的作用是防止不同自治系统之间的源地址伪造，确保数据包的源地址符合预期的自治系统。

组织特点

不同层次实现不同粒度的IPv6源地址真实性验证；每一层次，允许不同运营商采用不同方法；整体结构简单性和各部分组成的灵活性的平衡

部署收益

1. 安全责任明确：易于追踪攻击事件，定位攻击者；杜绝基于伪造源地址的攻击
2. 网络管理简化：简化身份认证；流量可审计不可抵赖；支持基于IP地址的网络计费

真实 IP 原地址的三重含义 P77

真实IPv6源地址验证体系结构，用以在网络层提供一种透明的服务，确保互联网中转发的每一个分组都使用“真实IP源地址”

1. 经授权的：IP源地址必须是经互联网IP地址管理机构分配授权的，不能伪造
2. 唯一的：IP源地址必须是全局唯一的
3. 可追溯的：网络中转发的IP分组，可根据其IP源地址找到其所有者和位置

真实源地址验证体系结构设计原则

1. 自治域 P79
2. 域内路由 IGP-OSPF：P80~82
3. 域间路由 BGP：P83~84

SAVA设计原则：P85

可扩展性 P85~90

采用层次化的思想来满足体系结构可扩展性需求是一种常用的手段；各层之间可以共同协作，根据约定俗成的协议进行简单交互而不用关注对方内部的实现细节，大大提升体系的效率。

- 层与层之间相独立，每层可以独立进行技术的演化
- 每层拥有自己的优化目标
- 不同层次利用局部信息对决策进行分布式计算，以实现个体最优
- 综合起来，这些局部算法实现了全局最优

需要划分灵活可变的源地址验证粒度，满足不同部署区域的需求和整体架构的可扩展性需求。需要建立层次化，寻找合理的层次间关系，使得各层之间可以共同协作，同时避免层次之间的过度依赖。

可演进性 P91

1. 与已有协议兼容：SAVA体系结构建立在当前互联网体系结构基础上，整体的技术依附于现有体系结构实现，因此必须要求技术对应协议与现有体系结构协议兼容
2. 自身部署可演进：SAVA的部署是一个持续性的过程，所以会出现部分区域已经部署SAVA，而部分区域尚未部署SAVA的情况，需要考虑在发展部署的过渡阶段SAVA自身的兼容性
3. 运营商之间可演进：考虑到网络中不同运营商的存在，SAVA体系结构还应允许运营商可以采用各自不同的实现，SAVA系统各部分相互独立，且功能彼此不依赖

安全性 P92

SAVA体系结构的构建是支撑真实可信互联网体系结构实现，通过将安全性赋予现有体系结构，弥补其信任缺失的问题，所以保障SAVA自身的安全性至关重要

1. 可信标识风险：标识应当具备唯一性、可追溯性
2. 数据转发风险：需保证携带标签的数据包转发中不被篡改，即使篡改也应被及时并准确地识别
3. 单点信任风险：标签的验证应当不依赖于中心化的网络基础设施，以免引入网络基础设施的信任风险

SAVA体系结构与关键技术 P94

部署与地址管理范围的失配问题 P95

源地址在分配上存在层次性。自治域从区域的地址分配机构RIR获取多个前缀，这些前缀在被拆分为更细粒度的前缀之后被分配到自治域内的各个子网。主机在使用地址时，需要从所接入的子网获取地址。在早期SAVA的研究中，很自然地将源地址验证划分为自治域、前缀、主机这三个粒度。随着真实源地址验证技术的发展和应用的增量部署，出现一个自治域中部分子网部署了真实源地址验证技术，而剩余子网尚未部署的情形，这就导致SAVA部署与地址管理范围的失配。

面向地址域的新型源地址验证SAVA体系结构 P96~97

地址域：同一机构所属的全部IP地址中的可部分管理范围

- 以一个校园网为例，地址域可以是某一个院系下的某个所某个组，也可以是某个所、某个院系，甚至可以是整个校园网
 - “地址域”显著提升体系结构的灵活性，实现了部署结构灵活的源地址真实性验证体系结构
 - 接入网、地址域内和地址域间三层结构，具有松耦合、多重防御、支持增量部署等优点
1. 接入层面提供主机粒度的源地址验证能力，以保证地址使用的可追溯性
 2. 在地址域内层面提供前缀级别的保护能力，以保护核心设备不被攻击
 3. 在地址域间层面提供地址域级别的联盟内可验证能力以及保护自身不被伪造的能力

SAVI实现源地址验证的“三步曲” P99

1. 获取合法地址：监听控制类报文（如ND、DHCPv6），即CPS分组，获取地址分配信息以识别主机合法IP源地址
2. 建立绑定关系：将合法的IP地址与主机网络附属的链路层属性绑定（“绑定锚”）（可验证的，且比主机的IP源地址本身更难欺骗）
3. 匹配验证：对数据包中的IP源地址与其绑定锚进行匹配（**部署位置越靠近主机，有效性越高**）

接入网异构多样性 P100

1. 终端多样性：终端包括手机、电脑、服务器、嵌入式设备等各种类型设备，即使同一种设备，其上运行的系统也可能不同，比如手机可能是安卓系统，也可能是IOS系统
2. IP分配方式多样性：包括DHCP协议分配、SLAAC协议分配、静态配置等
3. 接入方式多样性：包括有线、无线等，不同接入模式可用的绑定锚可以不同

SAVI 验证方式 P101

针对多种接入网技术、多种地址分配方式、多种终端类型，SAVI设计了各种对应的验证方式

- 所有相关网络设备在同一个网络管理机构管理控制下
- 解决方案与接入子网地址管理分配和控制策略密切相关
- 解决方案与端系统的接入方式密切相关

DHCP P102~104

DHCP: Dynamic Host Configuration Protocol，动态主机配置协议

目标：

1. 允许主机在加入网络时从网络服务器动态获取IP地址
2. 在使用中的地址可以更新租期
3. 允许地址重用(仅在连接on时保留地址)
4. 支持想要加入网络的移动用户

无线局域网 WLAN P? ~?

WLAN网络架构分有线侧和无线侧两部分；有线侧是指接入点AP上行到Internet的网络使用以太网协议；无线侧使用802.11协议，连接终端STA和AP，又称WiFi；无线侧包括基于控制器AC的AP架构（瘦AP，FitAP）和传统的独立AP架构（胖AP，Fat AP）。

无线 SAVI P?

基于无线局域网的源地址认证技术通过嗅探地址分配相关控制报文（如NDP和DHCP报文），将IP地址与MAC地址绑定并依赖MAC地址的安全性对伪造IP源地址的报文进行过滤。增加对主机移动场景的支持，提升了SAVI的灵活性。

802.1x 认证 P?

802.1x是一种对用户进行认证的链路层协议。802.1x是基于端口的认证策略（可以是物理端口也可以是VLAN一样的逻辑端口，相对于无线局域网“端口”就是一条信道）。802.1x的认证的最终目的就是确定一个端口是否可用。802.1x认证通过后，将成为用户IP地址的绑定锚。

动态自适应的地址分配分组监听 P105~106

SAVI-DHCP等方法通常被视为彼此独立的，每个方法处理自己的条目。如果在同一设备中使用了多种方法而没有进行协调，每一种方法都会拒绝通过其他方式绑定的数据包；SAVI设计了SAVA-MIX统一管理绑定表，基于各种方式形成了动态自适应的地址分配分组监听。

一旦某种SAVI方法生成了IP地址和对应绑定锚，它将请求SAVI-MIX在绑定表中设置相应的条目。SAVI-MIX将检查绑定表中是否有任何冲突。如果没有冲突，将生成一个新的绑定条目。如果有冲突，将按照一定策略进行冲突解决。

对现有网络设备和主机协议栈透明（不修改协议、不修改主机），能够适应复杂、动态、大规模的实际网络环境。实现了地址级的无漏判源地址验证，并且通过将设备标识与合法IP地址关联，达到了端网协同的目标，同时也实现了接入网兼容性。

SAVI 技术方案概要 P107

主要步骤

1. 监听地址分配协议的控制报文，确定合法的IP源地址
2. 将合法的IP地址绑定到主机的链路层属性 (绑定锚)
3. 对数据包中的IP源地址与它们所绑定的绑定针是否匹配进行检验

实施效果

1. 附着在同一接入设备下的主机不能伪造本接入设备下 其余主机的IP地址
2. 附着在某一接入设备下的主机不能伪造其他接入设备 下主机的IP地址

域内真实源地址验证技术 P108

1. 如果一个地址域与接入网相连，需保证从接入网流入地址域的流量，其源地址不会假冒该接入网之外的地址
2. 如果接入网部署了SAVI，进行二次验证；如果接入网没有部署SAVI，缩小源地址假冒的范围（接入网级别）
3. 保证从地址域内产生并流出地址域的流量，其源地址不会假冒地址域之外的地址

精准过滤：地址过滤的精确率（precision）100%；地址过滤的召回率（recall）100%

自动更新：自适应接入网地址分配和域内路由策略的动态更新，不完全依赖手动配置

源地址验证表 SAVA-P P109

根据目的地址选择分组的出接口（路由转发表）

根据源地址验证分组的入接口（源地址验证表）

源地址验证表生成方法 P110

1. 正确性：解决路由不对称问题
2. 低开销：通信开销和计算开销
3. 激励性：网络通过主动部署受益

基本思路和基本框架 P111~113

路由器通过发送探测报文，探测域内转发路径，沿途的路由器根据收到的探测报文生成<源前缀和入接口>的对应关系。

利用FIB中的destination prefix指导探测报文的接力发送方向

- 考虑到域内策略路由的复杂性，路由器无法获知到destination prefix的准确路径信息
- 一个探测报文携带去往同一个下一跳的多个destination prefix，减少传输开销

FIB表：Forwarding Information Base，前向信息库。它是由路由信息库（RIB，Routing Information Base）派生而来，存储着具体的转发路径信息，用于快速查找数据包的转发路径。

DPP 报文的生成与处理 P114~125

Data Plane Probe (DPP): 一种用于评估和测试网络设备数据平面性能的报文，帮助网络管理员了解和监控网络设备的实际数据处理能力。

P114 原始DPP报文的生成及其格式

P115~116 路由器收到DPP报文的处理方式

P81 路由器收到多个同源DPP的四种场景，P82 路由环路检测，P83 开销分析，P84 实验分析

方案总结 P119

设计目标	URPF [RFC 3704, 8704]	SAVE [INFOCOM 2002]	O-CPF [CFI 2013]	SAVA-P
正确性 (解决路由 不对称问 题)		V	V	建立与转发表方向相反的 源地址验证表，克服 路由不对称问题
低开销 (通信开销 和计算开 销)	V			每台路由器处理的协议 报文数量约为O(N)，N 为网络内的路由器数量

域间真实源地址验证 SAVA-X P120

通过在地址域之间建立信任联盟实现源地址验证，部署在联盟成员的边界路由器上；边界路由器为本域内发往其它联盟成员的报文进行地址域级别的源地址前缀检查，保证源自本地地址域的报文携带的源地址确实属于本地地址域；边界路由器为源自本地地址域、宿于其它成员地址域的报文添加用以标识本地地址域身份的“标签”，该标签可验证，确保地址域地址前缀不被冒用。

P121 控制层，P122 数据层，P123 状态机，P124 数据转发，P125 标签验证，P126 基于区块链的域间信任联盟。P128 标识格式和验证替代，P129 信任联盟与节点管理

P127 层次结构

SAVA-X分层：支持五层结构（不要求全部具备），自上而下为信任联盟、子信任联盟（CERNET/电信网）、一级（AS、地址域）/二级（院系）/三级（楼宇）地址域

基于 IPV6 的可信身份识别 P? ~?

技术难点

- 网络的分布式管理与终端标识的跨域有效性间的矛盾
- 要求真实性标识基于统一结构

研究思路

- IPv6地址空间承载终端标识保障跨域验证的有效性
- 1. 在地址真实性的基础上，设计嵌入可信设备标识符的IPv6 地址生成算法，兼顾多种IPv6地址分配机制和组网环境，将端设备信息携带于数据包中，实现端网协同

2. 地址标识采用动态更新机制，达到对终端设备或终端用户的身份动态标识和隐私保护的目的
3. 构建可信设备标识符的认证、管理、追溯和审计机制

基于真实地址验证关键技术，通过在IP地址中嵌入可信设备标识符，实现了端网协同的真实用户身份识别和溯源，赋予地址防重放、防逆推、防DDoS等特性。

数据包防篡改机制 P? ~?

技术难点：SAVA体系结构保障源地址的真实性，确保数据包源地址可追溯、可验证。然而数据包在域间传递的过程中仍然存在数据内容被篡改的风险

研究思路：数据包完整性校验

针对数据包传输路径上可能存在的恶意篡改问题，提出数据包防篡改机制；验证效率高、复杂度低、部署结构可扩展性强、防篡改能力突出。

利用SAVA-X标签证明源端身份，增加数据包摘要信息共同生存数据包签名；中间节点不具有源端SAVA-X标签，无法伪造数据包签名，防止其它恶意节点篡改数据包。

第10讲 互联网流量分析与检测

背景

流量分析技术诞生前夜 P2-9

- 流量规模激增
- 流量传播计算机病毒
- 恶意流量造成经济损失
- 流量传递隐私

常见考点：TLS协议

- 网络通讯匿名化

用户使用 **Tor (The Onion Router, 洋葱路由网络)** 作为匿名通讯设施

- 源路由：客户端通过洋葱路由算法选择 3 个中继节点 (Entry, Middle, Exit)
- 层次加密：根据选定的三个节点，将消息逐层加密。
- Entry 解密之后得到 Middle 的地址并转发给 Middle，Entry 虽然知道源地址但是不知道目的地址；Middle 解密得到 Exit 的地址，但不知道消息的来源和目的。
- 代理访问：Exit 解密得到原始数据包，虽然知道目的，但不知道数据的来源，最后 Exit 代理客户端向网站发起请求。

但是 Tor 网络的**层次化加密**应该不足以保证完全的匿名性。

核心问题 P10

- 识别和拦截攻击流量背后的技术是什么？
- 什么技术在加密和匿名化后仍然可以窃取隐私？

上述技术的核心在于对流量特征进行分析：根据流量的特征推测关联性信息。

流量分析定义 P11

定义：对二进制的网络流量进行变换，得到具备语义量化的特征，精确刻画流量的表示（例如流中的包数量、每一个包长度等等）

流量分析技术既可以用于攻击，也可以用于防御

流量分析目标 P12-15

- 推测是否为攻击流量进行入侵检测
网络入侵检测系统（NIDS, Network Intrusion Detection System）
- 进行流量分析攻击。E.G. 推测 Tor 网络访问对象的攻击被称为**网站指纹攻击, web fingerprinting**
流量分析攻击核心原理：虽然明文信息被加密，但信息的载体仍然通过其特征泄漏信息，即信息载体构成侧信道

流量分析三要素和分类法 P16

- 流量分析系统的三要素：流量特征、分析算法、分析目标

第1节 负载特征驱动流量检测

基于数据包负载匹配的传统检测方法 P19-24

即传统恶意流量识别方法：基于固定规则

以 Zeek 为例，现代固定规则检测系统支持的三大类匹配方法包括：

- 对负载的统计特征进行匹配
- 对包头字段进行匹配
- 对明文流量负载进行匹配

基于人工智能的流量检测方法总体框架 P25-27

基于人工智能的负载分析方法：网站防火墙 P28-36

网站应用防火墙式一种专门用于监控、过滤和阻止传入和传出 Web 应用程序的恶意流量的安全系统

实例：ZeroWall, ODDS, RETSINA

基于人工智能的负载分析方法：恶意软件检测 P37-47

实例：WebWitness, Nazca, DYNAMINER, BotSniffer, JACKSTRAWS, HinDom

第2节 统计特征驱动的流量识别方案

基于包粒度特征的检测方法 P48-54

为每一个数据包抽取一个特征向量

实例：Kitsune, nPrintML, CLAP

基于流粒度特征的检测方法 P55-67

为每一条流抽取一个特征向量

P57: 包粒度特征和流粒度特征的比较

实例: Disclosure, BotFinder, Whisper, Xatu, Lumen

可编程网络设备上的攻击流量检测 P69-78

- 可编程网络设备上的流量检测: 可编程交换机 P69-75

就是把机器学习算法实现在可编程交换机上, 从而达到更快的速度

实例: FlowLens, HorusEye, NetBeacon,

- 可编程网络设备上的流量检测: 智能网卡 P76-78

智能网卡 (Smart NIC) 在网卡芯片组当中集成通用处理器, 这样以来不需要进行IO操作消耗CPU资源即可在网卡上完成简单的包处理逻辑

实例: NIC

加密攻击流量检测 P79

现有的方法无法实现通用加密流量检测

简单介绍了基于流量交互图的加密恶意流量识别

第3节 检测后的防御方案

传统防御手段 P92

- 路由黑洞
- IP黑名单
- 流量清洗中心

检测后的防御手段: 可编程交换机方案 P95

代表方案: Poseidon, Jaqen, Ripple,

第4节 流量分析攻击

网站指纹攻击 P102-105

网站指纹攻击通过分析用户生成的 Tor 流量推测被访问的网站, 从而完全破坏 Tor 网络隐藏被访问的网站的功能。

注意: 网站指纹攻击采集用户加密流量的位置是用户到 Tor 第一跳的位置, 即距离用户最近的节点。即攻击者的位置是在第一跳之前窃听流量。

代表方案: k-fingerprinting, ARES, NetCLR,

网站指纹攻击的变种：流量关联攻击 P106-107

流量分析的目标：判断一个入端口和一个出端口是否为一个用户生成的（这样就可以将 Tor Enter 和 Tor Exit 的信息拼起来）

代表性方案：DeepCoFFA,

其他流量分析攻击 P108-111

- 针对 APP 应用的方案
- 针对即时通讯软件应用的方案
- 针对加密DNS流量的方案

流量分析攻击的防御手段 P112-114

- 基于特征扰动的方案
- 基于特征消除的方案

 image-20240605234440558

ch12 分布式系统

概述

分布式系统的组成 P7

如今所说的分布式系统通常是由分布在不同地理位置的系统组件所构成，这些系统组件通过网络传递消息完成动作协调，从而相互协作完成共同的任务

分布式系统的组成包括：

- 交互网络 — 实现协作的前提，系统组件基于交互网络实现消息交互
- 分布式算法 — 协调分布式组建之间的交互，确保系统稳定运行
- 信任机制 — 解决不同系统组件之间的信任问题，实现可信协作

分布式系统的三个属性 P8~9

- 一致性：分布式系统中各正确节点的所保存的同一变量状态信息都是一致的
- 可用性：分布式系统在收到客户端的请求后，必须给出回应，不能让客户端陷入无限等待过程中
- 分区容错性：分布式系统容忍其中节点 出现分区，当分区出现时，一个区域中节点发往另一个区域中节点的数据包全部丢失，即区域间无法进行通信

在一个分布式系统中，一致性、可用性、分区容错性，三者不可得兼

P（分区容错率）总需要满足，而C（一致性）和A（可用性）矛盾。

分布式系统安全问题的根源 P10

安全问题可能会造成系统分区，或是危害系统的一致性、有效性和隐私性

从分布式系统的组成来看，其安全问题的根源体现在交互网络、分布式算法和信任三个方面：

- 攻击者可以攻击交互网络，监听交互过程中的隐私信息或造成网络分区

- 协作过程中可能会出现冲突，故障等问题，从而影响一致性和可用性
- 系统内部可能存在恶意组件，系统和客户端交互时也存在信任问题

交互网络 P13

交互网络的组成 P13

在分布式系统中，节点之间可以通过交互信道实现消息传递；分布式节点选择邻居节点并建立交互信道，从而构成一个交互网络

交互信道：两个主机通过交互信道实现host-to-host的通信，常见的交互信道有UDP、TCP和TLS等

邻居选择机制：每个节点依靠邻居选择机制选取系统中的部分或全部节点建立交互信道，从而建立邻居关系，节点可通过邻居节点与其它非邻居节点交互

交互网络存在的安全隐患 P14~17

根据网络体系结构的不同层面进行分类：**网络层面，应用层面。**

根据攻击者的位置分类：**中间转发节点，其它节点**

具体的交互信道中的安全隐患包括：**数据包丢失和重复，信息泄漏，数据包篡改，数据包伪造，重放攻击**

TCP/TLS 交互信道 P18~19

TCP 协议采用滑动窗口协议确保数据包有序提交，并解决重复问题；采用超时重传机制解决链路拥塞导致的数据包丢失问题。

- 当未按序接收时，等待前面的数据包完成接收后再将数据包序列提交到上层应用。
- 当长时间未收到数据包时，触发超时空重传机制，请求源端超时重传数据包。

TLS 协议建立在 TCP 协议之上运行，提供以下三个安全功能：

1. 数据完整性：验证数据是否伪造而来或未遭篡改过
2. 身份认证：获取通信方的身份，为访问控制机制作支撑
3. 数据加密：协商会话密钥用于数据加密，防止第三方窃听传输数据

防止重放攻击 P20~23

1. **TLS 防止重放攻击** P20：TLS交互信道带来的优势——基于TCP协议确保传输层面数据包不会出现重复；通过数据加密确保恶意节点无法通过监听获取应用层消息。

但是 TLS 并不能完全解决重放攻击。以下消息依然可能遭受重放攻击：客户端发起的TLS请求建立信息ClientHello；TLS1.3为了加速握手过程而引入的0-RTT信息。

2. **基于时间戳防御重放攻击** P21：优点：容易实现；缺点：要求交互双方进行精确的时钟同步；存在一个可以被攻击的时间窗口。
3. **基于随机数防御重放攻击** P22：优点：无需时钟同步；不存在可被攻击的时间窗口；缺点：复用之前的数时，可能遭受重放攻击；需要记录大量已用过的随机数。
4. **时间戳和随机数相结合** P23：只需要在 δ 时间内随机数不重复即可 书page409

链路故障 P24~27

链路故障类型：链路崩溃，性能问题 P24

链路故障特征 P23

造成网络故障的主要原因：Internet的不稳定性，分布式系统所依赖的底层网络随时都可能出现物理链路故障、路由故障等因素。P25

解决思路：构建一个**弹性覆盖网络**，依靠其它主机实现消息的转发。P26

覆盖网络：注意这个覆盖网络是基于应用层的：

弹性覆盖网络RON P28

弹性覆盖网络是一种分布式覆盖网络体系结构，它可以使分布式应用检测到**路径的失效和周期性的性能降低现象**并能够迅速恢复；恢复时间少于20秒；对比：在目前的Internet中，BGP恢复时间为几分钟。

(P.S. BGP，全称为边界网关协议 (Border Gateway Protocol)，是互联网的主要路由协议，用于在不同自治系统 (AS, Autonomous Systems) 之间交换路由信息。)

RON节点监控它们之间的路径的质量并根据这些信息决定：直接利用Internet转发分组或者通过其他RON节点转发分组。

RON 的设计目标与应用场景 P28

RON 的设计 P30~31

RON 的稳定性分析 P32

P2P vs C/S p34~37

二者在结构和构成上有很大区别：管理能力、构态能力、功能（查找或发现）、组织、元素（DNS）和协议。但又无明显边界，都能运行在不同的（Internet / Intranet）平台上。都能服务传统或新的应用：eBusiness eServices ...

P2P (Peer-to-Peer)又称对等网络技术，也是建立在Internet之上的分布式Overlay网络，能够不依赖中心节点而只依靠对等协作实现资源发现与共享。（P.S. Overlay网络是一种构建在现有网络（通常称为底层网络或基础网络）之上的虚拟网络）

P2P 网络建立流程 P35~37 （主要包括网络节点发现和邻居节点维护两部分）

日蚀攻击 P38~41

相当于目标节点被“日蚀”，只能与恶意节点进行通信。

攻击目标：针对网络节点发现以及邻居节点维护两个过程进行攻击，使得受害节点的所有邻居节点都是由攻击者所控制的恶意节点。危害：攻击者可以选择性的转发对攻击者有利的消息给受害者，从而配合其它网络攻击；攻击者可以完全隔离受害者与网络中其它节点的信息交互。

挤占入向连接 P38，**挤占出向连接** P39 **配合路由劫持攻击** P40——主要思路 解决思路

分布式算法 P43

协作过程中的安全问题 P43

分布式算法的目的是协调分布式组件之间的交互和动作，从而确保它们能够稳定协作实现一个共同的任务。

协作过程中的安全问题包括：

- 缺乏全局时钟
- 并发
- 故障

时钟同步算法 P44~46

时钟不同步引发的问题 P43

时钟同步算法 P45~48。

基本思想：若节点A想与节点B完成时钟同步，则请求节点B发送本地时间T；然后预测A与B的网络时延RTT；之后将时间设置为 $T + RTT/2$ ；RTT预测的越精确，时钟同步的误差越小。

P48 网络时间协议（NTP协议）的主要目标和整体架构：NTP协议定义了时间服务的体系结构，以及如何在互联网上发布时间信息。

[TODO，这部分没有太懂]并发控制 P49~71

P49~50：并发问题破坏一致性或者导致指令之间相互影响。

P50 事务与分布式事务：事务是原子的一组请求，这组请求不受其它客户端干扰，并且要么全部完成，要么不对服务器造成任何影响。四个特性：1 原子性 2 一致性 3 隔离性 4 持久性

P51 **两阶段提交确保一致性** 在提交之前先询问所有存储该值的数据库是否可以更新，保证了一致性

P52 **基于锁机制确保隔离性** 在对 A 修改之前先上锁，在 Commit 或者 Decommit 之后解除 A 的锁

P55~57 对于两阶段提交协议，会出现三种故障：1 数据库节点故障 2 协调者故障 3 链路故障

无法容错的根源：在简单的二阶段提交协议中，必须所有相关节点都进行回应后才成功提交，因此协议无法容忍任何故障。

Quorum 机制 P58 书 P 419

- 满足一致性准则 和 有效性准则。写 W 个节点自后，认为写操作成功，在读取数据的时候，要读 R 个节点，由于 $W + R > N$ ，所以一定能读到正确的数据

Paxos 算法 书 P421 P60

- 无leader，每个节点提出自己的提案，模拟一群追求快速达成一致的立法者。缺陷在于，追求所有节点相对平等的关系，造成不必要的信息交互。甚至可能会形成**活锁**

Raft 算法 书 P423 P65

- 增加了领导者的角色，只有领导者能够提案。算法分为 **领导者选举** 和 **日志复制** 两个部分。只有领导者故障的时候进行领导者选举。当用户来了一个请求，给领导者，领导者试图和所有跟随者同步日志，如果同步成功，则提交事物，并给客户端返回，否则进行回滚。

信任问题 P73

零信任网络 P73~74：从来不相信，永远在校验。零信任对传统访问控制机制进行了范式上的颠覆其本质是以身份为基石的动态可信访问控制。

访问控制 P75

访问控制（Access Control）指系统按用户身份，以及该身份所在的策略组来限制用户对某些信息项的访问，或限制其对某些控制功能的使用。

基于角色的访问控制 P76

RBAC Role Based Access Control：不同角色有不同的操作集，RBAC设置不同的角色，并将各种身份信息与角色进行绑定，然后只对角色的访问权限进行控制，可极大地简化权限管理。

基于属性的访问控制 P77

ABAC Attribute：用户属性指的是可以对用户进行区分的特性。用户是访问控制的主体，以其属性作为访问网络数据的依据，可以实现更细粒度的访问控制。

包含了新的节点：1 PEP 策略实施节点 2 PDP 策略决策节点 3 PIP 策略信息节点

基于信任度的访问控制 P78

分为基本信任，直接信任和推荐信任。基本信任是由自身属性所决定，直接信任随着交互逐渐增加；推荐信任依赖信任值的传递。

访问控制的局限性 P79

访问控制只能解决被访问者对访问者的信任问题，但无法解决访问者对被访问者的信任问题，访问者可能访问到恶意、无效资源。

信任评价模型 P80~86

page81 包含三个角色，Trustee Trustor Recommender。系统举例 Eigentrust，节点根据历史交互评价计算信誉值。信任评价模型的五个要求 page83

拜占庭将军问题 P87

引入 P87-P95

一组拜占庭将军分别各率领一支军队共同围困一座城市。为了简化问题，将各支军队的行动策略限定为进攻或撤离两种。因为部分军队进攻部分军队撤离可能会造成灾难性后果，因此各位将军必须通过投票来达成一致策略，即所有军队一起进攻或所有军队一起撤离。因为各位将军分处城市不同方向，他们只能通过信使互相联系。在投票过程中每位将军都将自己投票给进攻还是撤退的信息通过信使分别通知其他所有将军，这样一来每位将军根据自己的投票和其他所有将军送来的信息就可以知道共同的投票结果而决定行动策略。

系统的问题在于，可能将军中出现叛徒，他们不仅可能向较为糟糕的策略投票，还可能选择性地发送投票信息。假设有9位将军投票，其中1名叛徒。8名忠诚的将军中出现了4人投进攻，4人投撤离的情况。这时候叛徒可能故意给4名投进攻的将领送信表示投票进攻，而给4名投撤离的将领送信表示投撤离。这样一来在4名投进攻的将领看来，投票结果是5人投进攻，从而发起进攻；而在4名投撤离的将军看来则是5人投撤离。这样各支军队的一致协同就遭到了破坏。

由于将军之间需要通过信使通讯，叛变将军可能通过伪造信件来以其他将军的身份发送假投票。而即使在保证所有将军忠诚的情况下，也不能排除信使被敌人截杀，甚至被敌人间谍替换等情况。因此很难通过保证人员可靠性及通讯可靠性来解决问题。

假使那些忠诚（或是没有出错）的将军仍然能通过多数决定来决定他们的战略，便称达到了拜占庭容错。在此，票都会有一个默认值，若消息（票）没有被收到，则使用此默认值来投票。

拜占庭问题的简化 P95



一致性和正确性条件 P95

一致性条件不够充分，忠诚将军的信息有可能被修改，所以必须引入另一个正确性的条件。

一致性（等价IC1）：每个将军必须得到相同的 (v_1, v_2, \dots, v_n) 指令向量或指令集合；忠诚的将军不一定使用 i 将军发来的信息作为 v_i ， i 将军有可能是叛徒。

正确性（等价IC2）：若 i 将军是忠诚的，其他忠诚的将军必须使用他发出的值作为 v_i 。

简化问题的解决方案 P96

主要问题：由于将军可能是叛徒，会对不同的副官发布不同的指令，因此副官不能完全信任将军的指令。

解决思路：副官在收到将军的指令后，互相交换从将军处收到的指令值，再基于交互结果选择最终执行指令。



口头消息 P97

口头消息下三将军问题无解：

P199 **口头消息协议 OM 协议：**当存在 m 个叛徒 而且 $n \geq 3m + 1$ 的时候，口头消息协议能够确保一致性，也即问题有解

签名消息协议 P111

签名消息协议 SM 协议：改进：在实际系统中，可以引入签名机制确保节点发送的消息无法被篡改，从而得到签名消息算法SM(Signed Messages)。

将军和副官发的消息都需要自己的签名，解决了口头消息协议交互复杂度过高的问题。结论，当 $n \geq 2m + 1$ 的时候，只需要**两轮交互**，就能确保共识达成

实用拜占庭容错 P115

区块链：构造一个对客户端来说可信的分布式账本；容忍少数拜占庭节点作恶解决分布式系统不同节点间协作时的信任问题。

同步异步 BFT 共识的 Quorum 设计 P117~119

同步BFT共识依靠同步网络假设确保共识安全性和活性，若网络延迟超过预设最大值，则会造成节点间出现状态不一致。

异步BFT共识只能采用无Leader的共识模式，因为异步网络中无法分辨Leader错误和网络延迟过高两种情况；因此，异步BFT共识不仅效率低下，还可能导致理论上的活锁（类似于前面分析的Paxos共识）。

P120 引入部分同步网络模型

P122 PBFT 部分同步算法

PBFT 基于 leader 完成共识，基于 Quorum 机制，这里总结 $n = 3m+1$; $q=2m+1$

PBFT 和 Paxos 和 Raft 区别是，增加了**共识之后的交易确认阶段**

Hotstuff 方案 page126-128

应用安全

Web 安全 P10

定义 P10

Web (World Wide Web) 即全球广域网，也称为万维网，它是一种基于超文本和 HTTP 的、全球性的、动态交互的、跨平台的分布式图形信息系统；为浏览者在 Internet 上查找和浏览信息提供了图形化的、易于访问的直观界面；Web 应用程序是运行在 Web 服务器上的应用软件，这些应用程序使用客户机/服务器 (Client/Server) 建模的结构进行编程。

XSS 攻击 P11~12

XSS (Cross-Site Scripting) 是攻击者通过往 Web 页面里插入恶意可执行网页脚本代码，当其他用户浏览被攻击的 Web 页面时，注入其中的恶意代码就会被执行，从而达到攻击者盗取、侵犯其他用户隐私的目的。

防御方法 P13

一般有字符转义和 CSP 白名单

跨站请求伪造 CSRF P14

利用用户已登录的身份，在用户毫不知情的情况下，以用户的名义完成非法操作

CSRF (Cross Site Request Forgery) 攻击的三个条件：

- 用户已经登录了站点 A，并在本地记录了 cookie；
- 在用户没有登出站点 A 的情况下 (cookie 生效的情况下)，访问了恶意攻击者提供的引诱危险站点 B (B 站点要求访问站点A)；
- 站点 A 没有做任何 CSRF 防御。

防范方法 P15：

- 对 Cookie 设置 SameSite 属性，避免第三方网站访问到用户 Cookie；
- 阻止第三方网站请求接口；
- 请求时附带验证信息，比如验证码或者 Token。

SQL 注入 P16~17

SQL注入是指web应用程序对用户输入数据的合法性没有判断或过滤不严，导致攻击者可以在web应用程序中事先定义好的查询语句的结尾上添加额外的SQL语句。

在管理员不知情的情况下实现非法操作，以此来实现欺骗数据库服务器执行非授权的任意查询，从而进一步得到相应的数据信息。

SQL 注入防御方法 P18

将数据与代码分离，具体四种办法。

点击劫持 P19

点击劫持是一种视觉欺骗的攻击手段，攻击者将需要攻击的网站通过 iframe 嵌套的方式嵌入自己的网页中，并将 iframe 设置为透明，在页面中透出一个按钮诱导用户点击。

用户在登陆 A 网站的系统后，被攻击者诱惑打开第三方网站，而第三方网站通过 iframe 引入了 A 网站的页面内容，用户在第三方网站中点击某个按钮（被装饰的按钮），实际上是点击了 A 网站的按钮。

URL 跳转漏洞 P21

借助未验证的URL跳转，将应用程序引导到不安全的第三方区域，从而导致的安全问题。黑客利用URL跳转漏洞来诱导安全意识低的用户点击，导致用户信息泄露或者资金的流失。原理：黑客构建恶意链接(链接需要进行伪装,尽可能迷惑)，发在QQ群或者是浏览量多的贴吧/论坛中，安全意识低的用户点击后，经过服务器或者浏览器解析后，跳到恶意的网站中。

OS 命令注入攻击 P22

OS命令注入攻击：通过Web应用，执行非法的操作系统命令达到攻击的目的。OS命令注入和SQL注入差不多，只不过SQL注入是针对数据库的，而OS命令注入是针对操作系统的。只要在能调用Shell函数的地方就有存在被攻击的风险。倘若调用Shell时存在疏漏，就可以执行插入的非法命令。命令注入攻击可以向Shell发送命令，让Windows或Linux操作系统的命令行启动程序，也就是说，通过命令注入攻击可执行操作系统上安装着的各种程序。

CDN 安全 P24

CDN 定义 P24

CDN (Content Delivery Network, 内容分发网络)：当前提高网站的性能、可靠性与安全性的最佳实践之一。CDN 是由分布在不同地理位置的服务器集群组成的分布式网络。目标：帮助其客户网站实现负载均衡、降低网络延迟、提升用户体验、过滤 SQL 注入等攻击，分散拒绝服务攻击的流量。

P25 工作流程

用户点击APP，APP会根据URL地址去DNS寻求IP地址解析。DNS服务器发现对应URL有CDN服务，将会返回CDN服务器对应的IP。用户向CDN服务器发起内容URL访问请求，如果CDN服务器有缓存内容，进行第4步，否则第5步。CDN服务器响应用户请求，将用户所需内容传送到用户终端。CDN缓存服务器上并没有用户想要的内容，CDN向网站的源服务器请求内容；源服务器返回内容给缓存服务器，缓存服务器发给用户。

P26 CDN 优势

- 加速了网站的访问
- CDN 提供一定安全性。

P27 ~ 33 RangeAmp 攻击

通过一些漏洞，可以通过CDN进行DoS攻击，从而破坏原有系统的可用性。

RangeAmp攻击：一台电脑便可让世界上最流行的网站瘫痪，一种利用CDN和HTTP协议设计缺陷对任意部署Web服务的站点实施DDoS的攻击。

CDN和HTTP范围请求（range requests）机制都致力于提升网络性能，但是CDN对HTTP范围请求机制的实现存在安全缺陷，攻击者能够滥用CDN的漏洞对源网站服务器或其他CDN节点实施DDoS攻击。

1. P28~29 HTTP Range 请求
2. P30 CDN 处理 Range 请求
3. P31小字节范围攻击
4. P32 重叠字节范围攻击
5. P33 解决方案：服务器侧，CDN侧，协议侧

社交网络安全 P34

P35 数据挖掘

数字档案收集，运维数据收集。

P36 垃圾信息传递

增加网络负载；信任缺失；身份假冒。

P37 网络钓鱼

攻击者可以伪装成为合法用户的好友，通过各种诱惑手段使得用户访问恶意URL。社交网络用户为了达到结交朋友的目的，并不排斥与陌生人沟通并接受交友邀请，因此，钓鱼攻击很容易发生。

P38 女巫攻击 Sybil attack

伪装成为多种身份参与到正常网络中，一方面利用虚假身份盗取合法用户的各种数据；另一方面影响数据转发路径，从而可伪造出多条不同的路由，破坏网络的可用性。

P40 女巫攻击账号检测

特征提取，联通图构建，Sybils检测。

云计算安全 P42

P42 定义

云计算的定义：通过网络按需提供可动态伸缩的廉价计算服务；是与信息技术、软件、互联网相关的一种服务。

云计算的五大特点：大规模；虚拟化；高可用性和扩展性；按需服务；网络安全。

P43 服务类型

基础设施即服务IaaS (Infrastructure as a Service)：向云计算提供商的个人或组织提供虚拟化计算资源；如虚拟机、存储、网络 and 操作系统等。

平台即服务PaaS (Platform as a Service)：为开发人员提供通过互联网构建应用程序和服务的平台；开发、测试和管理软件应用程序提供按需开发环境。

软件即服务SaaS (Software as a Service)：通过互联网提供按需软件付费应用程序；云计算提供商托管和管理软件应用程序；允许其用户连接到应用程序并通过互联网访问应用程序。

P44 虚拟化与虚拟机

虚拟化是为一些组件（例如虚拟应用、服务器、存储和网络）创建基于软件的（或虚拟）表现形式的过程；虚拟计算机系统称为“虚拟机”(VM)，它是一种严密隔离且内含操作系统和应用的软件容器；表面来看，这些虚拟机都是独立的服务器，但实际上，它们共享物理服务器的CPU、内存、硬件、网卡等资源。

P45 Hypervisor 虚拟机监视器

Hypervisor，又称虚拟机监视器（virtual machine monitor，缩写为VMM），是用来建立与执行虚拟机器的软件、固件或硬件。

P46 OpenStack 云管理平台

KVM这样的Hypervisor软件，实际上是提供了一种虚拟化能力，模拟CPU的运行，更为底层，但是它的用户交互并不良好，不方便使用。为了更好地管理虚拟机，就需要OpenStack这样的云管理平台。负责管理计算资源、存储资源、网络资源。本身不具备虚拟化能力，来自于各种虚拟化技术，VM、KVM、OpenStack等，都主要属于IaaS（基础设施即服务）

P47 Docker

容器（Container）：虚拟机是操作系统级别的资源隔离，而容器本质上是进程级的资源隔离

Docker是创建容器的工具，是应用容器引擎：启动时间很快，达到秒级，且对资源的利用率高（一台主机可以同时运行几千个Docker容器）；占用空间很小，虚拟机一般需几GB到几十GB，而容器仅需要MB级甚至KB级。

P48 K8S Kubernetes

K8S是Kubernetes的简称，中文意思是舵手或导航员。K8S是一个容器集群管理系统，主要职责是容器编排——启动容器，自动化部署、扩展和管理容器应用，还有回收容器。Docker和K8S，关注的不再是基础设施和物理资源，而是应用层，所以就属于PaaS。

P49 虚拟机逃逸

虚拟机逃逸指程序脱离正在运行并与主机操作系统交互的虚拟机的过程。虚拟化技术虽然可以在逻辑上提供软硬件的隔离，从而将各个用户分隔开，然而通过一些漏洞，虚拟机中的应用可以逃逸出逻辑的隔离，直接控制主操作系统，从而造成破坏。

P50 提权攻击

提权攻击是指用户通过系统漏洞，提升自己操作系统的使用权限的攻击。最简单的方法就是直接猜测管理员的弱口令等。比较可靠的提权方法就是攻击机器的内核，让机器以更高的权限执行代码，进而绕过设置的所有安全限制。

P51 侧信道攻击

侧信道攻击通过共享的信息通道，可以窃取到通道中的额外秘密。云计算中，虚拟机共享宿主硬件（CPU、内存、网络接口），因此可以通过CPU的计算时间，网络接口的占用时间，一定程度分析出其他用户的数据。已有攻击者利用侧信道攻击成功获取服务器中的私钥。

P52 镜像和快照攻击

镜像攻击：云计算平台往往通过特定的景镜像创建虚拟机或者服务实例。镜像的实例化是高度自动化的。攻击者入侵虚拟机管理系统并感染镜像。增大攻击效率和影响范围。

快照攻击：云平台可以随时挂起虚拟机并保存系统快照。攻击者非法恢复快照，造成一系列的安全隐患，且历史数据被清除，攻击行为被隐藏。

物联网安全 P53

定义 P53

物联网（Internet of Things）：通过各种信息传感器、射频识别技术、全球定位系统等，实时采集任何需要监控、连接、互动的物体或过程。通过各类可能的网络接入，实现物与物、物与人的泛在连接，实现对物品和过程的智能化感知、识别和管理。一个基于互联网、传统电信网等的信息承载体。让所有能够被独立寻址的普通物理对象形成互联互通的网络。

结构与挑战 P54 ~ 56

- 综合应用层：服务各种需求的物联网应用
- 管理服务层：将大规模数据存储、处理、分析
- 网络构建层：使得感知设备接入互联网中
- 感知识别层：物理世界与信息世界的纽带，获取现实世界的物理数据

移动应用安全 P57

P58 防御手段

- 源代码层面：应用代码调用的组合分析出潜在的恶意行为，从而进行识别
- 应用分发渠道层面：被篡改、盗版、二次打包、注入、反编译等破坏，需要对应用进行加固保护，构建正版指纹信息库
- 智能终端安全监测层面：通过样本特征、行为或者缺陷等分析技术，在终端处进行安全监控，检测异常行为，进行安全控制

应用安全攻击的共性特征及防御 P60

资源有限 DDoS P61

DDoS攻击（Distributed Denial of Service）：通过大量合法的请求占用大量网络资源，以达到使网络瘫痪的目的

分类：通过使网络过载来干扰甚至阻断正常的网络通讯；通过向服务器提交大量请求，使服务器超负荷；阻断某一用户访问服务器；阻断某服务与特定系统或个人的通讯。

占用网络资源类型：网络带宽、磁盘读写、CPU计算等。

Web、CDN、物联网、云计算都面临对应的安全风险

资源共享 非法访问 P62

攻击者可以合法、或者非法地获取应用数据，进而推理构造出目标数据

合法的越权访问：一些网络应用的数据是相互共享的，所有用户都可以合法的使用；这些数据往往由于隐私保护存在缺陷（差分隐私，互补隐私等）；这类攻击形式在Web应用和社交网络中最为常见。

非法的访问资源：逻辑上相互独立的用户数据存放在相同的一片物理区域；攻击者利用一些漏洞，非法访问其他用户的数据，造成一定程度的数据泄露；这种攻击形式主要存在于云计算中。

系统漏洞 P63

现有计算机体系结构复杂、应用丰富；不能确保多变、异构的应用硬件和软件的实现万无一失
完全没有任何漏洞是不可能的；漏洞的及时修复也存在问题。

防御原理 P64

1. P65 身份认证和访问控制：身份认证是保证信息安全的第一道门户；用户在被确认身份之后在信息系统中根据身份所有的权限享受相应的信息服务。
2. P66 隐私保护：隐私数据的泄露会引起严重的危险后果，通过身份认证和信任管理可以一定程度保护隐私，但是无法从数据本身保护；利用一些隐私保护算法或者技术来对隐私数据进行保护。
3. P67 监控防御，用第三方软件：一定网络应用系统的漏洞是无法避免的；在网络发生被攻击、破坏的情况下，可以通过监控检测，快速识别、恢复网络应用的服务，减少损失；在网络系统各个点上部署安全防御措施，避免出现安全的木桶效应。

典例

1. P69 微博被 XSS 攻击
2. P75 微信广告
3. P76 剑桥分析操纵大选

应用安全典型案例

- Web安全的机密性
- 社交网络安全的机密性

人工智能安全

人工智能安全应用 P34

- 网络信息安全应用：主要包括网络安全防护应用、信息内容安全审查应用、数据安全应用等
- 社会公共安全应用：主要包括智能安防应用、金融风控应用等

框架安全 P43

Pytorch 相比 TensorFlow 的优势

P47，设计简洁，易于理解，动态计算图

Keras P48 ~ 49

Keras：允许简单而快速的原型设计（由于用户友好，高度模块化，可扩展性）；同时支持卷积神经网络和循环神经网络，以及两者的组合；在 CPU 和 GPU 上无缝运行。

用户友好： Keras 是为人类而不是为机器设计的 API。它把用户体验放在首要和中心位置。将常见用例所需的用户操作数量降至最低，并且在用户错误时提供清晰和可操作的反馈

模块化： 模型被理解为由独立的、完全可配置的模块构成的序列或图。神经网络层、损失函数、优化器、初始化方法、激活函数、正则化方法等模块可以以尽可能少的限制组装在一起

易扩展性： 新的模块是很容易添加的（作为新的类和函数），现有的模块已经提供了充足的示例

基于Python实现：Keras没有特定格式的单独配置文件。模型定义在Python代码中，这些代码紧凑，易于调试，并且易于扩展

Caffe P50~51

具有表现力的结构：模型及优化是通过配置定义的，而不是使用硬编码的方式。可以在GPU和CPU之间无缝切换，可以用GPU训练，然后部署到集群或移动设备上

代码的可扩展性：Caffe第一年fork超过一千次，有许多有意义的贡献和反馈。由于众多的贡献者，Caffe框架跟踪并实现了当前最新的代码和模型

快速性：快速性使Caffe适合研究实验和工业开发。Caffe在单个的NVIDIA K40 GPU上每天能处理6千万张图片。识别时速度为1ms/张，训练时速度为4ms/张，BLVC相信Caffe具有最快的卷积神经网络实现

框架安全漏洞

1. TensorFlow P53 ~ 55：推理/训练中的拒绝服务攻击，分段错误，在数据流图中插入恶意操作后，不影响模型的正常功能，也就是说模型的使用者从黑盒角度是没有感知的
- Pytorch、Keras 和 Caffe P56 ~ 57：内存泄漏，权重丢失隐患，SQL注入漏洞

环境接触带来的漏洞

1. 第三方基础库漏洞 P60 ~ 61：Numpy 拒绝服务，OpenCV 堆溢出
2. 可移植软件容器漏洞 P63 ~ 64：Kubeflow 挖矿，部署恶意容器

算法安全 P65

鲁棒性安全 P73

- 人工智能算法的优化原理 P74 ~ 75
- 人工智能算法的可解释性 P76 ~ 81
 1. 建模前可解释性方法：数据可视化，寻找 ProtoTypes 和 Criticisms（典例与特例）——Prototype 是指能够表示大部分数据的数据实例，Criticism 是指不能由Prototype很好表示出的数据实例
 2. 建立本身具备可解释性的模型：一些具有良好可解释性的模型包括决策树模型、线性模型以及贝叶斯实例模型等
 3. 使用可解释性方法对模型进行解释：敏感性分析和隐层分析，基于可视化方法的模型解释（结构可视化、训练可视化）
- 人工智能算法的鲁棒性评估 P82

深度学习领域的鲁棒性可以理解为模型对数据变化的容忍度，鲁棒性越高的模型，其识别噪声和对抗样本的准确率越高。鲁棒性差的模型，在受到对抗攻击时，容易给出高可信度的错误结果。鲁棒性差的模型，在更换数据集时（如训练数据更换为测试数据或投入使用时的实际数据），性能往往表现出巨大的差异。

分类维度？

- 白盒攻击：攻击者能够获知机器学习所使用的算法，以及算法所使用的参数。攻击者在产生对抗性攻击数据的过程中能够与机器学习的系统有所交互。也就是说，攻击者可以将样本送入模型中以获取梯度等信息，然后依据这些信息对输入进行调整
- 黑盒攻击：攻击者对攻击的模型的内部结构，训练参数，防御方法（如果加入了防御手段的话）等等一无所知，只能通过输出输出与模型进行交互。攻击者会使用one-pixel-attack暴力攻击或使用迁移样本进行攻击。这里的迁移样本指的是，对抗样本往往是可迁移特性，即针对机器学习模型A构造的对抗性图像，也会有很大的比例能欺骗机器学习模型B。
- 目标攻击：生成的对抗样本被DNNs误分类为某个指定类别。目标攻击一般发生在多分类问题中。
- 无目标攻击：生成的对抗样本识别结果和原标注无关，即只要攻击成功就好，对抗样本最终属于哪一类不做限制。因为无目标攻击有更多的选择和更大的输出范围，所以比目标攻击更易实现。
- 基于梯度攻击：考虑模型对样本的梯度，根据梯度的方向和大小等对样本进行调整，使损失函数增大
- 基于优化攻击：将输入样本视为可变量，并通过优化算法来最小化或最大化某个特定的目标函数，以找到最优的输入样本，使得模型在该样本上产生误导性的输出结果

安全角度审视机器学习系统？

机密性（Confidentiality）——模型隐私，数据隐私；完整性（Integrity）——数据投毒攻击（训练阶段），对抗样本攻击（测试阶段）；可用性（Availability）——系统决策是否准确可靠

投毒攻击 P85 ~ 90

设计攻击样本，混入到训练数据，让人工智能算法失效。

模型学习训练样本的分布，同时假设训练样本和测试样本是同分布的。如果扰乱训练数据的分布，自然会使模型对测试数据给出错误的输出。

P91 ~ 92 投毒攻击的防御

为了保障机器学习或深度学习分类模型在受到投毒攻击后的性能，可以在开发模型阶段选取一个干净的数据集，模拟一些投毒策略并据此进行防御。但是，由于可能的攻击空间几乎无限，所以并不能保证防御措施是安全的，即，不能保证一个对已知攻击集有效的防御将不会对新的攻击失效。

对抗样本攻击 P104

对原始数据构造人类难以分辨的扰动，将会引起深度学习算法决策输出的改变，造成人类与深度学习模型认知的差异。近年来对抗样本被证明存在于现实物理世界中，并可能会对多种机器学习系统产生影响。图神经网络(GNN)广泛用于各类场景，然后研究发现图神经网络同样易被实施黑盒对抗样本攻击。各类商用语音识别系统也易遭受黑盒对抗样本攻击，可以导致定向和非定向攻击效果。

P105 投毒攻击与对抗攻击的区别

投毒攻击：强调的是通过混入特殊样本的形式，直接对模型进行修改，而不修改测试数据。基于反馈的投毒攻击是指将用户反馈系统武器化来攻击合法用户和内容。一旦攻击者意识到模型利用了用户反馈对模型进行惩罚(penalization)，他们就会基于此为自己谋利。

对抗攻击：攻击者在不改变目标机器学习系统的情况下，通过构造特定输入样本以完成欺骗目标系统的攻击。如基于梯度的攻击方法FGSM等，均属于逃逸攻击

模型机密性攻击 P113

模型萃取攻击

模型萃取攻击是指攻击者通过多次查询操作，获取目标模型全部或者部分信息（如网络结构或参数等）

成员推理攻击

目的是确定特定的数据是否被用于模型的训练过程，即特定的数据是否存在于训练集中。

模型逆向攻击

从模型的预测结果中恢复出关于输入数据的信息。

属性推理攻击

揭露数据中未直接公开的敏感属性，这些属性一般与训练任务无关。

算法局限性 P122

数据难以获取 P123

小数据：数据量太小而无法深度学习。假数据：需要手工生成的数据，有时没有有效性。孤岛数据：A、B数据维度单一，但可以互补，在现实中因为某种原因无法获得两种数据。

数据不完整或偏斜 P125

AI的数据通常包含不完整或偏斜的信息。因为在获取数据时，往往不能获取整个样本空间的数据集。取而代之的是获取一个样本空间的子集。而某些子集的属性并不能代表整个样本空间，因而用这个数据是具有“偏见”的。偏差可以有意的，也可以是无意的。数据，算法和选择它们的人员都可能存在偏见。偏见可能与种族，性别，年龄，位置或时间有关。

成本局限性 P127

随着训练数据量急剧增长，大模型的训练时间开始以“星期”甚至“月”为单位计量。越来越长的训练时间远远满足不了业务快速迭代的需求。

优秀的人工智能模型背后往往隐藏着巨量的经济开销，主要表现在以下几个方面：数据成本——这一点与数据的局限性相关联，想要获得好的数据就必须付出高昂的成本。开发成本——开发合适的人工智能模型需要一定数量的技术人员，与此对应的财力支出也不可忽略。算力成本——结果显示，算力成本与模型大小成正比，然后对模型调参以提高最终精度的过程中，成本呈爆炸式增长，然而性能收益微乎其微。

偏见局限性 P130

AI也同样可能产生偏见，特别是当它向我们人类学习时。**词嵌入**层面就具有偏见。

伦理局限性 P133

AI的不可解释性或将长期存在，通过解释某个结果如何得出而实现算法的透明化，在技术上几乎不具有实操性，因而AI的判断也会有错判的情况。如果算法只是帮助我们更好的娱乐、工作，这个问题似乎不那么紧迫。可当算法被用到刑侦、医疗中时，AI被用于给出错误的犯罪者线索、疾病诊断结果时，这是一个必须回答的问题。