第十章：背景知识补充

10.1 计算机基础

栈（Stack）

栈（Stack）是计算机科学中的一种数据结构，它遵循后进先出（Last In, First Out，LIFO）的原则。栈可以看作是一种特殊的线性表，只能在表的一端进行插入和删除操作，该端被称为栈顶。栈的另一端称为栈底。栈的基本操作包括压栈（Push）和弹栈（Pop）。

压栈操作将一个元素添加到栈的顶部，使其成为新的栈顶。弹栈操作将栈顶的元素移除，并返回被移除的元素。栈的特点是后进入栈的元素先被弹出，而先进入栈的元素则会被推迟弹出。

栈的应用非常广泛。在计算机程序中，栈被用于函数调用和返回的过程中，用于保存函数的局部变量和返回地址等信息。栈还可以用于解析和计算表达式，进行括号匹配，以及在深度优先搜索等算法中的状态管理。

栈的实现可以使用数组或链表。在数组实现中，可以使用一个指针来表示栈顶的位置，通过不断修改指针的值来实现压栈和弹栈操作。在链表实现中，可以使用节点来表示栈的元素，并通过修改节点之间的链接关系来实现栈的操作。

总结一下，栈是一种遵循后进先出原则的数据结构，具有压栈和弹栈两个基本操作。栈常用于函数调用、表达式求值和算法中的状态管理等场景。它可以用数组或链表来实现。

时间戳（Timestamp）

时间戳（Timestamp）是指某一特定事件发生的日期和时间的表示。它通常以一种可读的格式呈现，例如年-月-日 时:分:秒。时间戳可以用于记录事件的发生时间、排序事件或进行时间相关的计算。

根据维基百科的定义，时间戳是"计算机科学中用于标记和记录时间的一种方式，通常是一个从特定起点开始的经过了一定时间的累积值"。时间戳可以基于不同的起点和时间单位，如UNIX时间戳（以1970年1月1日UTC时间为起点，以秒为单位）或自纪元以来的毫秒数。

时间戳在计算机科学和信息技术领域有广泛的应用。它在操作系统中用于记录文件的创建时间、修改时间和访问时间。在数据库中，时间戳可以用于跟踪记录的变化和版本控制。在网络通信中，时间戳可以用于同步不同设备之间的事件顺序。此外，时间戳还在日志记录、数据分析和时间序列分析等领域发挥着重要作用。

通用唯一辨识码（Universally unique identifier）

通用唯一辨识码（Universally Unique Identifier，简称UUID）是一种标识符，用于在计算系统中唯一地标识实体。它是一个128位的值，通常以32个十六进制数字的形式表示，中间用连字符分隔。UUID的生成算法保证了其在全球范围内的唯一性。

UUID的设计目的是在分布式系统中标识实体，以避免冲突和重复。每个UUID都可以看作是一个独特的标识符，不同实体可以使用不同的UUID来进行唯一标识。UUID的生成算法通常基于时间戳、计算机的唯一标识符和随机数等因素，以保证生成的UUID具有足够的唯一性。

UUID在许多领域都有广泛的应用。在数据库中，UUID可以用作主键，确保每个记录具有唯一的标识符。在分布式系统中，UUID可以用于标识不同节点或实体，实现数据同步和一致性。在软件开发中，UUID可以用于生成临时文件名、会话标识符等。

需要注意的是，UUID并不保证全局唯一性，但在实践中，由于其生成算法的设计，碰撞的概率非常低。如果需要更高的唯一性保证，可以考虑使用更长的标识符或结合其他因素来生成唯一标识。

总结一下，UUID是一种128位的标识符，用于在计算系统中唯一地标识实体。它具有广泛的应用，可以在数据库、分布式系统和软件开发中使用。UUID的生成算法保证了其在实践中的唯一性，但并不保证全局唯一性。

正则表达式（Regular expression）

正则表达式（Regular expression，常简写为regex、regexp或RE），又称规律表达式、正规表示式、正规表示法、规则运算式、常规表示法，是计算机科学概念，用简单字串来描述、匹配文中全部符合指定格式的字串，现在很多文本编辑器都支援用正则表达式搜寻、取代符合指定格式的字串。

正则表达式可以用于各种编程语言和文本处理工具中，如Python、Java、JavaScript、Perl等。它提供了一种灵活而强大的方式来处理字符串，包括匹配、替换、提取等操作。

正则表达式中的字符和特殊符号具有特定的含义和功能。例如，常见的特殊符号包括：

- `.`：匹配任意单个字符（除了换行符）。
- `*`：匹配前面的元素零次或多次。
- `+`：匹配前面的元素一次或多次。
- `?`：匹配前面的元素零次或一次。
- `[]`：定义字符集，匹配其中的任意一个字符。
- `()`：定义捕获组，用于提取匹配的部分。

通过组合和使用这些字符和特殊符号，可以构建复杂的模式来匹配特定的字符串。例如，可以使用正则表达式来验证电子邮件地址的格式、提取URL链接、过滤文本中的敏感词等。

正则表达式的语法和功能非常丰富，如果需要详细了解正则表达式，可以查阅相关的教程和文档，以深入学习其用法和应用。

总结一下，正则表达式是一种用于匹配、搜索和操作文本的工具，通过字符和特殊符号构建模式来描述字符串的特定模式或规则。它在各种编程语言和文本处理工具中广泛应用，提供了强大的字符串处理功能。

JSON（JavaScript Object Notation）

JSON（JavaScript Object Notation）是一种轻量级的数据交换格式。它以易于阅读和编写的文本形式表示结构化数据，并且可以被多种编程语言解析和生成。JSON最初是由Douglas Crockford在2001年提出的，并且在Web开发中得到了广泛应用。

JSON的数据结构是基于键值对的集合。它由两种主要的数据类型组成：对象（Object）和数组（Array）。对象是一个无序的键值对集合，每个键值对由一个键（key）和一个值（value）组成。键是一个字符串，值可以是字符串、数字、布尔值、对象、数组或者null。数组是一个有序的值的列表，每个值可以是字符串、数字、布尔值、对象、数组或者null。

JSON的语法非常简洁明了。对象使用花括号（{}）表示，键值对之间用冒号（:）分隔，每个键值对之间用逗号（,）分隔。数组使用方括号（[]）表示，数组中的值之间也用逗号（,）分隔。字符串需要用双引号（""）括起来。

JSON的设计目标是易于理解和使用。它在Web开发中被广泛应用于数据交换和配置文件等场景。许多编程语言提供了内置的JSON解析和生成函数或库，使得开发人员可以方便地处理JSON数据。

总结一下，JSON是一种轻量级的数据交换格式，以易于阅读和编写的文本形式表示结构化数据。它由对象和数组两种数据类型组成，对象是无序的键值对集合，数组是有序的值的列表。JSON的语法简洁明了，被广泛应用于Web开发和数据交换中。

10.2 计算机网络

计算机网络是指将多台计算机通过通信设备和通信介质连接起来，实现信息交换和资源共享的系统。本书中涉及的部分计算机网络知识，将在本节中自底向上地介绍。

互联网协议套件（Internet Protocol Suite）

互联网协议套件（Internet Protocol Suite，简称IPS）是网络通信模型，以及整个网络传输协议家族，为网际网络的基础通信架构。它常通称为TCP/IP协议族（TCP/IP Protocol Suite，或TCP/IP Protocols），简称TCP/IP。因为该协议家族的两个核心协议：TCP（传输控制协议）和IP（网际协议），为该家族中最早通过的标准[3]。由于在网络通信协议普遍采用分层的结构，当多个层次的协议共同工作时，类似计算机科学中的堆栈，因此又称为TCP/IP协议栈（TCP/IP Protocol Stack）

TCP/IP提供了点对点链接的机制，将资料应该如何封装、寻址、传输、路由以及在目的地如何接收，都加以标准化。它将软件通信过程抽象化为四个抽象层，采取协议堆栈的方式，分别实现出不同通信协议。协议族下的各种协议，依其功能不同，分别归属到这四个层次结构之中。

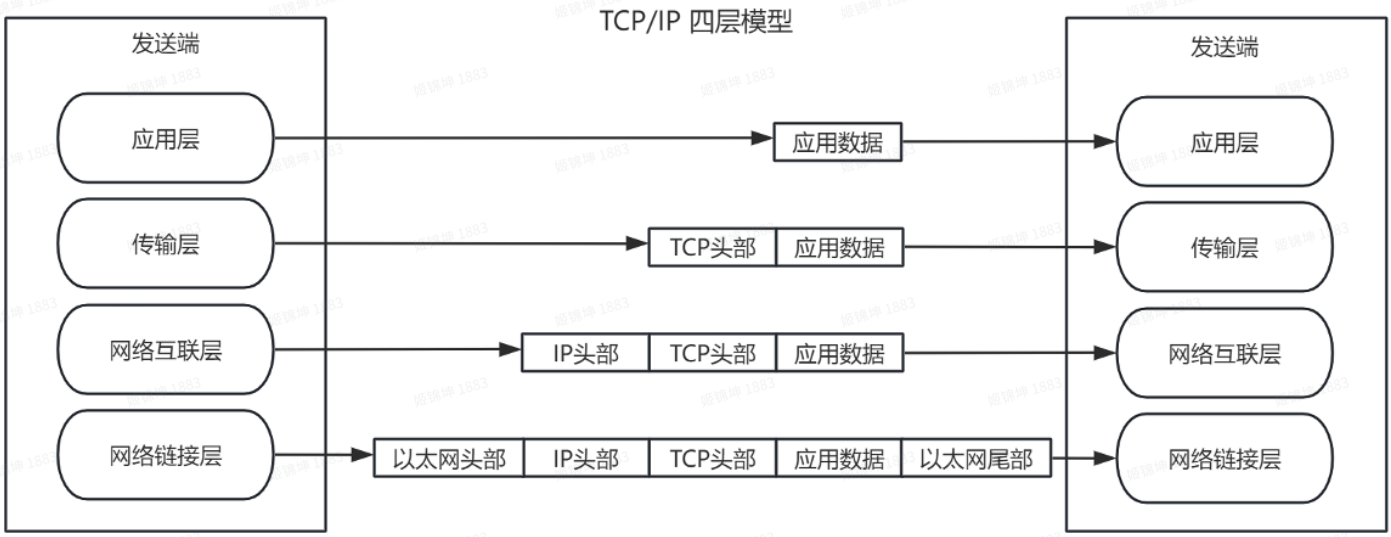


图 11.2.1 TCP/IP 四层参考模型

在此参考模型中，网络连接层是不常关注的，因为此层实际上并不是TCP/IP协议族中的一部分，但是它是数据包从一个设备的网络层传输到另外一个设备的网络层的方法。也是TCP/IP协议族必不可少的支持，但是不是研究TCP/IP协议组的关注对象。

下面将从网络互联层开始自底向上介绍此协议族中重要协议与一些常用知识。

网络互联层（internet layer）

IP（Internet Protocol）

IP是在TCP/IP协议族中网络互联层的主要协议，任务是仅根据数据包标头中的IP地址将数据包从源主机传递到目标主机。为此，IP协议定义了封装要传递的数据的数据包结构。它还定义了用于用源和目的地信息标记数据报的寻址方法

IP的主要作用是在网络中唯一标识和定位设备。每个连接到互联网的设备都被分配一个唯一的IP地址，类似于门牌号码，用于在网络中准确定位和寻找设备。常见的IP地址分为IPv4与IPv6两大类，IP地址由一串数字组成。IPv4为32位长，通常书写时以四组十进制数字组成，并以点分隔，如：172.16.254.1；IPv6为128位长，通常书写时以八组十六进制数字组成，以冒号分割，如：2001:db8:0:1234:0:567:8。目前广泛使用的IP版本是IPv4，它使用32位的地址空间，约有42亿个可用地址。由于互联网的快速发展，IPv4的地址空间已经不够用了，因此逐渐推广使用IPv6。IPv6使用128位的地址空间，提供了更多的地址，以满足日益增长的设备连接需求。

IP使用数据包来传输数据。数据包是网络中传输的基本单位，它包含了数据的源地址、目标地址和实际数据。当发送方要发送数据时，它将数据分割成适当大小的数据包，并在每个数据包中添加源地址和目标地址。这些数据包在网络中通过路由器进行转发，直到到达目标主机。

IP使用路由选择算法来确定数据包的最佳路径。路由器是网络中的设备，它负责将数据包从一个网络发送到另一个网络。路由器根据每个数据包中的目标地址，查找路由表来确定下一跳的路由器。路由表中包含了网络之间的连接信息，路由选择算法根据这些信息选择最佳路径。

IP是一种网络层协议，用于在计算机网络中唯一标识和定位设备。它使用数据包进行数据传输，通过路由选择算法确定数据包的最佳路径。IPv4和IPv6是两个主要的IP版本，用于分配和管理设备的地

址。理解IP的基本概念对于初学者来说是非常重要的，因为它是构建互联网和实现网络通信的基础。

无类别域间路由（Classless Inter-Domain Routing）

无类别域间路由（Classless Inter-Domain Routing，简称CIDR）是一个用于给用户分配IP地址以及在互联网上有效地路由IP数据包的对IP地址进行归类的方法。

在早期的互联网发展中，IP地址分为A类、B类、C类等固定的分类，每个分类有固定的网络位和主机位。这种分类方式存在一个问题，即每个分类的地址空间过大或过小，导致地址资源的浪费或不足。

为了更灵活地分配IP地址，CIDR引入了可变长度子网掩码（Variable Length Subnet Mask，VLSM）的概念。子网掩码用于将IP地址划分为网络位和主机位，指示哪些位是网络标识，哪些位是主机标识。而可变长度子网掩码允许将网络位和主机位的长度按需分配，从而实现更精细的地址分配和路由控制。

CIDR表示法使用斜线后跟着一个数字来表示子网掩码的长度。例如，192.168.0.0/24表示子网掩码为24位，即前24位是网络位，后8位是主机位。这意味着该网络可以容纳256个主机（2^8），可以表示192.168.0.0到192.168.0.254之间的IP地址。

总之，CIDR通过引入可变长度子网掩码，实现了更有效地管理和分配IP地址的目的。CIDR在现代互联网中被广泛应用，为网络的可扩展性和灵活性提供了重要支持。

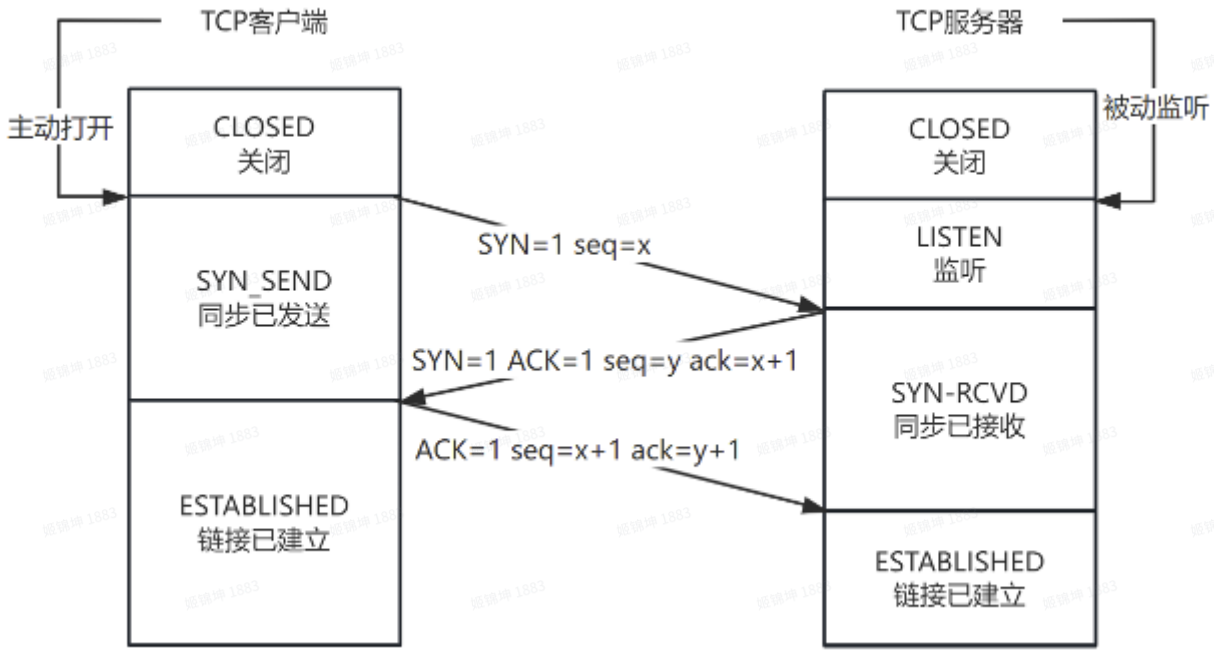
传输层（internet layer）

TCP（Transmission Control Protocol）

TCP（Transmission Control Protocol）是位于传输层的一种网络传输协议，用于在计算机网络上可靠地传输数据。它是互联网协议套件中的一部分，负责将数据分割成小的数据包并在网络上进行传输。

完整的TCP连接分为四个步骤：

- **建立连接（三次握手）**：在数据传输之前，发送方和接收方需要建立一个连接。这是通过三次握手来实现的。首先，发送方发送一个带有SYN（同步）标志的数据包给接收方。接收方收到后，回复一个带有SYN/ACK（同步/确认）标志的数据包给发送方。最后，发送方再回复一个带有ACK（确认）标志的数据包给接收方。这样，连接就建立起来了。



- **数据传输：**一旦连接建立，发送方可以开始将数据分割成小的数据包并发送给接收方。每个数据包都包含序列号，用于在接收方重新组装数据时进行排序。接收方会确认已接收的数据并发送确认消息给发送方。如果发送方没有收到确认消息，它会重新发送相应的数据包。
- **拥塞控制**：TCP具有拥塞控制机制，用于防止网络拥塞并保证数据传输的可靠性。当网络出现拥塞时，发送方会减少发送的数据量，以避免造成更严重的拥塞。这种机制可以确保网络资源的公平分配，并提供较好的性能。
- **关闭连接（四次挥手）**：当数据传输完成后，发送方和接收方需要关闭连接。这是通过四次挥手来实现的。首先，发送方发送一个带有FIN（结束）标志的数据包给接收方，表示发送方已经完成数据传输。接收方收到后，回复一个带有ACK标志的数据包给发送方，确认收到了结束请求。然后，接收方发送一个带有FIN标志的数据包给发送方，表示接收方也完成了数据传输。最后，发送方回复一个带有ACK标志的数据包给接收方，确认接收到了结束请求。这样，连接就成功关闭了。

总结起来，TCP是一种可靠的传输协议，通过建立连接、数据传输、拥塞控制和关闭连接等步骤，确保数据在计算机网络上的可靠传输。它在互联网中被广泛使用，例如在Web浏览器、电子邮件和文件传输等应用中。

UDP（User Datagram Protocol）

UDP（User Datagram Protocol）是一个简单的面向数据包的通信协议，位于OSI模型的传输层。该协议由David P. Reed在1980年设计且在RFC 768中被规范。典型网络上的众多使用UDP协议的关键应用在一定程度上是相似的。

在TCP/IP模型中，UDP为网络层以上和应用层以下提供了一个简单的接口。UDP只提供数据的不可靠传递，它一旦把应用程序发给网络层的数据发送出去，就不保留数据备份（所以UDP有时候也被认为是不可靠的数据包协议）。UDP在IP数据包的头部仅仅加入了复用和数据校验字段。

UDP适用于不需要或在程序中执行错误检查和纠正的应用，它避免了协议栈中此类处理的开销。对时间有较高要求的应用程序通常使用UDP，因为丢弃数据包比等待或重传导致延迟更可取

由于UDP缺乏可靠性且属于无连接协议，所以应用程序通常必须容许一些丢失、错误或重复的数据包。某些应用程序（如TFTP）可能会根据需要在应用程序层中添加基本的可靠性机制

一些应用程序不太需要可靠性机制，甚至可能因为引入可靠性机制而降低性能，所以它们使用UDP这种缺乏可靠性的协议。流媒体，实时多人游戏和IP语音（VoIP）是经常使用UDP的应用程序。 在这些特定应用中，丢包通常不是重大问题。如果应用程序需要高度可靠性，则可以使用诸如TCP之类的协议。

10.3 常见网络安全概念

中间人（Man-in-the-Middle）

中间人（Man-in-the-Middle，简称MITM）是一种网络安全攻击技术，攻击者在通信的两端之间插入自己，并伪装成合法的通信参与者，以窃取、篡改或劫持通信数据。攻击者可以截获双方的通信内容，并在不被察觉的情况下对数据进行修改或篡改，从而破坏通信的机密性和完整性。中间人攻击攻击可能发生在各种通信协议中，如网络通信、无线通信和加密通信，对信息安全构成严重威胁。

中间人在网络安全领域有广泛的应用。从业人员可以使用中间人攻击技术来评估系统和网络的安全性，模拟攻击者的行为，发现潜在的漏洞和弱点。此外，中间人攻击还可以用于网络流量监控和分析，帮助检测恶意活动和数据泄露。中间人攻击技术是网络安全领域重要的基石技术。

网络爬虫（Crawler）

网络爬虫（crawler），也叫网路蜘蛛（spider），是一种自动获取网页信息的程序，它按照一定的规则，自动地浏览互联网并抓取所需信息。爬虫首先访问一份网页列表，读取页面内容，再从这些内容中提取出其他页面的链接，进一步访问和抓取。

爬虫访问网站的过程会消耗目标系统资源。不少网络系统并不默许爬虫工作。因此在访问大量页面时，爬虫需要考虑到规划、负载，还需要讲“礼貌”。 不愿意被爬虫访问、被爬虫主人知晓的公开站点可以使用robots.txt文件之类的方法避免访问。这个文件可以要求机器人只对网站的一部分进行索引，或完全不作处理。

攻击载荷（Payload）

攻击载荷（Payload）是网络攻击中的关键组成部分，通常指的是攻击者用来对目标系统造成伤害或实现特定目的的数据。例如，它可以用来传播恶意软件，窃取信息，或者使系统崩溃。攻击载荷可以通过各种方式传输，如电子邮件附件，网页链接，或者插入到网络数据包中。

概念验证（Proof of concept）

概念验证（Proof of concept，简称POC）是对某些想法的一个较短而不完整的实现，以证明其可行性，示范其原理，其目的是为了验证一些概念或理论。POC通常被认为是一个有里程碑意义的实现的原型。在网络安全领域，POC也扮演着非常重要的角色。在这个场景下，PoC通常用于验证一个系统、网络或应用中存在的安全漏洞是否可以被利用，以及这种利用可能导致的后果。

例如，一个安全研究人员可能会发现一个理论上的漏洞，他们可以通过构建一个PoC来证明这个漏洞在实际环境中是否可以被利用。这可能涉及到编写特定的代码、构建特定的网络环境，或者模拟特定的攻击行为。

通用平台枚举（Common Platform Enumeration）

通用平台枚举（Common Platform Enumeration，简称CPE）是一种针对信息技术系统、软件和包的结构化命名方案。基于统一资源标识符（URI）的通用语法，CPE包括一种正式的名称格式、一种检查系统中名称的方法，以及一种将文本和测试绑定到名称的描述格式。简单来说，CPE是一种给各种IT产品命名的标准方式。

为了统一标准，有一个被称为"CPE产品字典"的东西，它提供了一个公认的官方CPE名称列表。这个字典是以XML格式提供的，任何人都可以查看和使用。这个字典由美国国家标准与技术研究院（NIST）托管和维护，非政府组织可以自愿使用。

在实际应用中，CPE标识符常常被用于搜索影响所识别产品的公共漏洞和暴露（CVE）。这是一种全球公认的安全漏洞和暴露数据库，其中包含了各种IT产品可能存在的安全问题和漏洞。通过CPE标识符，我们可以快速找到与特定产品相关的所有已知漏洞，从而更好地理解和应对可能的安全风险。

指纹识别（fingerprint detect）

指纹识别（fingerprint detect）是一种技术，用于识别和辨别计算机网络中的设备、应用程序或服务的唯一特征或标识。它通过收集和分析网络设备、网络协议、应用程序或服务的特定属性、行为或配置来生成唯一的指纹，从而实现对网络中实体的识别和区分。

网络设备指纹识别主要关注于识别和区分网络上的设备，如路由器、交换机、防火墙等。它通过收集设备的网络协议、端口状态、操作系统类型、设备特定的行为等信息来生成设备指纹。这些信息可以通过网络扫描、协议分析、设备响应等方式获取。通过分析这些特征，可以生成唯一的设备指纹，用于标识和识别特定的网络设备。

应用程序或服务指纹识别则关注于识别和区分网络上的应用程序或服务。它通过收集应用程序或服务的网络协议、通信模式、特定的数据包结构、响应行为等信息来生成应用程序或服务指纹。这些信息可以通过网络流量分析、协议解析、应用程序响应等方式获取。通过分析这些特征，可以生成唯一的应用程序或服务指纹，用于区分和识别不同的应用程序或服务。