安全接入专题

2011Q1

网络之路

Route To Network

2011年第1季度，总第15期

H3C

IToIP Solutions Expert

目录 CONTENTS

概述

001 接入与身份认证技术概述 徐庆伟

认证协议

008 RADIUS协议简介 袁亚屏

033 TACACS+ 技术 袁亚屏

046 LDAP概述及应用 李 丹

057 数字证书与PKI 邓宏波

077 Kerberos认证浅析 曾 劼

087 EAP技术简介 袁 江

接入认证技术

091 802.1X技术基础 盛冬旭

103 Portal概述 李朝辉

116 MAC认证技术基础 盛冬旭

120 Triple认证技术 王 瑞

EAD

124 EAD方案简介 王君菠

测试相关

136 EAD多客户端模拟工具 饶 云

■文/徐庆伟

网络接入身份认证技术概述

→概　述←

随着信息化的快速发展，对国家、组织、公司或个人来说至关重要的信息越来越多的通过网络来进行存储、传输和处理，为获取这些关键信息的各种网络犯罪也相应急剧上升。当前，网络安全在某种意义上已经成为一个事关国家安全，社会经济稳定的重大问题，得到越来越多的重视。

在网络安全中，身份认证技术作为第一道，甚至是最重要的一道防线，有着重要地位，可靠的身份认证技术可以确保信息只被正确的“人”所访问。身份认证技术提供了关于某个人或某个事物身份的保证，这意味着当某人（或某事）声称具有一个特别的身份时，认证技术将提供某种方法来证实这一声明是正确的。

本文主要从局域网接入和接入用户身份认证的角度，讨论常用的认证机制、认证协议，以及采用这些认证技术的不同接入协议。

→认证技术概述←

身份认证可分为用户与系统间的认证和系统与系统之间的认证。身份认证必须做到准确无误地将对方辨认出来，同时还应该提供双向的认证。目前使用比较多的是用户与系统间的身份认证，它只需单向进行，只由系统对用户进行身份验证。随着计算机网络化的发展，大量的组织机构涌入国际互联网，以及电子商务与电子政务的大量兴起，系统与系统之间的身份认证也变得越来越重要。

身份认证的基本方式可以基于下述一个或几个因素的组合：

所知（Knowledge）：即用户所知道的或所掌握的知识，如口令；

所有（Possesses）：用户所拥有的某个秘密信息，如智能卡中存储的用户个人化参数，访问系统资源时必须要有智能卡；

特征（Characteristics）：用户所具有的生物及动作特征，如指纹、声音、视网膜扫描等。

根据在认证中采用的因素的多少，可以分为单因素认证、双因素认证、多因素认证等方法。身份认证系统所采用的方法考虑因素越多，认证的可靠性就越高。

→认证机制与协议←

一般而言，用于用户身份认证的技术分为两类：简单认证机制和强认证机制。简单的认证中认证方只对被认证方的名字和口令进行一致性的验证。由于明文的密码在网上传输极容易被窃听，一般解决办法是使用一次性口令（OTP，One-Time Password）机制。这种机制的最大优势是无需在网上传输用户的真实口令，并且由于具有一次性的特点，可以有效防止重放攻击。RADIUS协议就是属于这种类型的认证协议。

强认证机制一般将运用多种加密手段来保护认证过程中相互交换的信息，其中，Kerberos协议是此类认证协议中比较完善、较具优势的协议，得到了广泛的应用。

下面简单介绍一些常用的认证机制与协议。

常用的认证机制

基于口令的身份认证机制

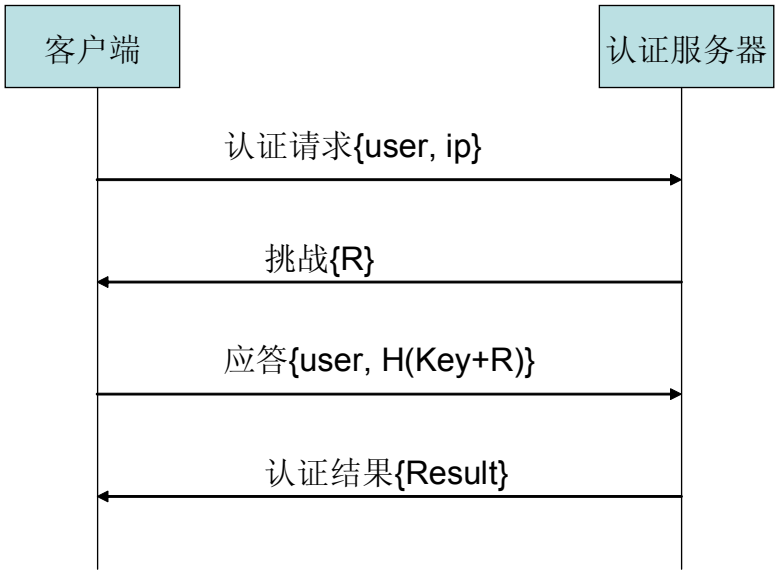
基于口令的身份认证技术因其简单易用，得到了广泛的使用。但随着网络应用的深入和网络攻击

手段的多样化，口令认证技术也不断发生变化，产生了各种各样的新技术。最常采用的身份认证方式是基于静态口令的认证方式，它是最简单、目前应用最普遍的一种身份认证方式。但它是一种单因素的认证，安全性仅依赖于口令，口令一旦泄露，用户即可被冒充；同时易被攻击，采用窥探、字典攻击、穷举尝试、网络数据流窃听、重放攻击等很容易攻破该认证系统。相对静态口令，动态口令也叫一次性口令，它的基本原理是在用户登录过程中，基于用户口令加入不确定因子，对用户口令和不确定因子进行单向散列函数变换，所得的结果作为认证数据提交给认证服务器。认证服务器接收到用户的认证数据后，把用户的认证数据和自己用同样的散列算法计算出的数值进行比对，从而实现对用户身份的认证。在认证过程中，用户口令不在网络上传输，不直接用于验证用户的身份。动态口令机制每次都采用不同的不确定因子来生成认证数据，从而每次提交的认证数据都不相同，提高了认证过程的安全性。

挑战/响应认证机制

挑战/响应方式的身份认证机制就是每次认证时认证服务器端都给客户端发送一个不同的“挑战”码，客户端程序收到这个“挑战”码，根据客户端和服务器之间共享的密钥信息，以及服务器端发送的“挑战”码做出相应的“应答”。服务器根据应答的结果确定是否接受客户端的身份声明。从本质上讲，这种机制实际上也是一次性口令的一种。

一个典型的认证过程如下图所示：



认证过程为：

1) 客户向认证服务器发出请求，要求进行身份认证；

2) 认证服务器从用户数据库中查询用户是否是合法的用户，若不是，则不做进一步处理；

3) 认证服务器内部产生一个随机数，作为“挑战”码，发送给客户；

4) 客户将用户名字和随机数合并，使用单向Hash函数（例如MD5算法）生成一个字节串作为应答；

5) 认证服务器将应答串与自己的计算结果比较，若二者相同，则通过一次认证；否则，认证失败；

6) 认证服务器通知客户认证成功或失败。

EAP认证机制

EAP（Extensible Authentication Protocol）扩展认证协议在RFC2248中定义，是一个普遍使用的认证机制，它常被用于无线网络或点到点的连接中。EAP不仅可以用于无线局域网，而且可以用于有线局

域网，但它在无线局域网中使用的更频繁。EAP实际是一个认证框架，不是一个特殊的认证机制。EAP提供一些公共的功能，并且允许协商所希望的认证机制。这些机制被称为EAP方法。由于EAP方法除了IETF定义了一部分外，厂商也可以自定义方法，因此EAP具有很强的扩展性。IETF的RFC中定义的方法包括EAP-MD5、EAP-OTP、EAP-GTC、EAP-TLS、EAP-SIM和EAP-AKA等。无线网络中常用的方法包括EAP-TLS、 EAP-SIM、 EAP-AKA、 PEAP、LEAP和EAP-TTLS。目前EAP在802.1X的网络中使用广泛，可扩展的EAP方法可以为接入网络提供一个安全认证机制。

公钥认证机制

随着网络应用的普及，对系统外用户进行身份认证的需求不断增加，即某个用户没有在一个系统中注册，但也要求能够对其身份进行认证，尤其是在分布式系统中，这种要求格外突出。这种情况下，公钥认证机制就显示出它独特的优越性。

公钥认证机制中每个用户被分配给一对密钥，称之为公钥和私钥，其中私钥由用户保管，而公钥则向所有人公开。用户如果能够向验证方证实自己持有私钥，就证明了自己的身份。当它用作身份认证时，验证方需要用户方对某种信息进行数字签名，即用户方以用户私钥作为加密密钥，对某种信息进行加密，传给验证方，而验证方根据用户方预先提供的公钥作为解密密钥，就可以将用户方的数字签名进行解密，以确认该信息是否是该用户所发，进而认证该用户的身份。

公钥认证机制中要验证用户的身份，必须拥有用户的公钥，而用户公钥是否正确，是否是所声称拥有人的真实公钥，在认证体系中是一个关键问题。常用的办法是找一个值得信赖而且独立的第三方认证机构充当认证中心（Certificate Authority，CA），来确认声称拥有公开密钥的人的真正身份。要建立安全的公钥认证系统，必须先建立一个稳固、健全的CA体系，尤其是公认的权威机构，即“Root CA”，这也是当前公钥基础设施（PKI）建设的一个重点。

使用认证机制的认证协议

许多协议在向用户或设备授权访问和访问权限之前需要认证校验，通常要用到认证相关的机制，前面讨论了常用的认证机制，本节介绍使用这些认证机制的协议，这些协议包括RADIUS、TACACS、Kerberos、LDAP等。RADIUS和TACACS通常用在拨号环境中，Kerberos是在校园网中用的比较多的协议。LDAP提供一种轻量级的目录服务，严格来说不能算作一种认证协议，而对用户进行认证收取只是LDAP的一种应用。

RADIUS认证协议

RADIUS（Remote Authentication Dial In User Service）协议最初是由Livingston公司提出的，目的是为拨号用户进行认证和计费。后来经过多次改进，形成了一个通用的AAA协议。

RADIUS协议认证机制灵活，能够支持各种认证方法对用户进行认证。可以采用上述任何一种认证机制。RADIUS是一种可扩展的协议，它进行的全部工作都是基于属性进行的，由于属性可扩展性，因此很容易支持不同的认证方式。

RADIUS协议通过UDP协议进行通信，RADIUS服务器的1812端口负责认证，1813端口负责计费工作。采用UDP的基本考虑是因为NAS和RADIUS服务器大多在同一个局域网中，使用UDP更加快捷方便。

TACACS认证协议

TACACS（Terminal Access Controller Access Control System）最先是由BBN为MILNET开发的一种基于UDP的访问控制协议，一些厂商对协议进行了扩展，最终形成了一种新的AAA协议，其中CISCO公司对TACACS协议多次进行增强扩展，目前成为TACACS+协议，H3C在TACACS（RFC1492）基础上进行了功能增强，形成了H3C扩展的TACACS协议。无论TACACS、TACACS+还是H3C扩展TACACS协议，其认证、授权和计费是分离的，并且与原始TACACS协议相比，TACACS+和HWTACACS可以使用TCP作为传输层协议，端口号为49。

TACACS+允许任意长度和内容的认证交换，与RADIUS一样，具有很强的扩展性，并且客户端可以使用任何认证机制。由于TACACS+的认证与其他服务是分开的，所以认证不是强制的，这点与RADIUS是不同的。

Kerberos认证协议

在一个分布式环境中，采用上述两种认证协议时，如果发生账号改动的情况，每台机器上的都要进行相应的账号修改，工作量非常大。Kerberos是MIT为解决分布式网络认证而设计的可信第三方认证协议。Kerberos基于对称密码技术，网络上的每个实体持有不同的密钥，是否知道该密钥便是身份的证明。网络上的Kerberos服务起着可信仲裁者的作用，可提供安全的网络认证。Kerberos常见的有两个版本：第4版和第5版，目前使用的标准版本是版本5。

Kerberos是一种受托的第三方认证服务，它是建立在Needham和Schroeder认证协议基础上，它要求信任第三方，即Kerberos认证服务器（AS）。AS为客户和服务器提供证明自己身份的票据以及双方安全通信的会话密钥。Kerberos中还有一个票据授予服务器（TGS），TGS向AS的可靠用户发出票据。除客户第一次获得的初始票据是由Kerberos认证服务器签发外，其他票据都是由TGS签发的，一个票可以使用多次直至期限。客户方请求服务方提供一个服务时，不仅要向服务方发送从票据授予服务器领来的票据，同时还要自己生成一个证(Authenticator，Ac)一同发送，该证是一次性的。

LDAP协议

LDAP（Lightweight Directory Access Protocol）是基于X.500标准的，但是比X.500简单，并且可以根据需要定制。与X.500不同，LDAP支持TCP/IP，这对访问Internet是必须的。LDAP是一个目录服务协议，目前存在众多版本的LDAP，而最常见的则是V2和V3两个版本，它们分别于1995年和1997年首次发布。一般在分布式、跨平台认证的场景下，LDAP比前面介绍的认证协议具有一定优势。

LDAP协议严格来说并不属于单纯认证协议，对用户进行授权认证是LDAP协议的一个典型应用。例如Microsoft的Windows操作系统就使用了Active Directory Server来保存操作系统的用户、用户组等信息，用于用户登录Windows时的认证和授权。目录服务其实也是一种数据库系统，只是这种数据库是一种树形结构，而不是通常使用的关系数据库。目录服务与关系数据库之间的主要区别在于：二者都允许对存储数据进行访问，只是目录主要用于读取，其查询的效率很高，而关系数据库则是为读写而设计的。所以LDAP协议非常适合数据库相对稳定，而查询速度要求比较高的认证场合。

→接入认证技术←

以上介绍了常见的认证技术，这些认证技术与不同的局域网接入技术结合，可以实现对接入用户的认证授权，从而实现对局域网接入用户的访问控制。在局域网中对接入用户的认证控制，已成为网络安全的重要组成部分。目前在局域网中常见的接入认证技术包括802.1X、Portal、MAC地址认证等。本节简单介绍这些常用的局域网接入认证技术，具体协议细节以及实现原理在后续文章中会有相信介绍。

IEEE 802.1X接入认证技术

802.1X协议是为解决无线局域网网络安全问题而提出来的。但是随着该技术的广泛应用，目前802.1X协议作为局域网接入控制机制在以太网中被广泛应用，主要解决以太网内认证和安全方面的问题。

802.1X协议是一种基于端口的网络接入控制协议。“基于端口的网络接入控制”是指，在局域网接入设备的端口这一级，对所接入的用户设备通过认证来控制对网络资源的访问。802.1X可以与不同的认证协议结合实现对接入用户的认证，目前在使用802.1X接入时常与RADIUS配合实现对接入用户的认证与授权。

802.1X技术可以扩展到基于MAC地址对用户接入进行控制，即对共用同一个物理端口的多个用户分别进行认证控制。另外H3C对802.1X接入协议进行了扩展，与EAD解决方案结合，加强了对用户的集中管理，提升了网络的整体防御能力。

Portal接入认证技术

Portal技术通常也称为Web认证技术，一般将Portal认证网站称为门户网站。

Portal认证是一种三层网络接入认证，在认证之前用户首先要获取地址，这点与802.1X不同。未认证用户上网时，设备强制用户登录到特定站点，用户可以免费访问其中的服务。当用户需要使用互联网中的其它信息时，必须在门户网站进行认证，只有认证通过后才可以使用互联网资源。

不同的组网方式下，可采用的Portal认证方式不同。按照网络中实施Portal认证的网络层次来分，Portal的认证方式分为两种：二层认证方式和三层认证方式。前者在二层接口上启用Portal功能，而后者是在三层接口上启用Portal功能。

目前H3C Portal除了支持Web认证方式之外，还支持客户端方式的Portal认证，这样Portal协议就可以与EAD解决方案结合，提供端点接入控制，从而提高网络的安全性。

MAC接入认证技术

MAC地址认证是一种基于端口和MAC地址对用户的网络访问权限进行控制的认证方法，是一种二层网络接入认证技术，与802.1X以及Portal不同，用户不需要安装任何客户端软件就可进行认证。设备在启动了MAC地址认证的端口上首次检测到用户的MAC地址以后，即启动对该用户的认证操作。认证过程中，不需要用户手动输入用户名或者密码。若该用户认证成功，则允许其通过端口访问网络资源，否则该用户的MAC地址就被添加为静默MAC。在静默时间内（可通过静默定时器配置），来自此MAC地址的用户报文到达时，设备直接做丢弃处理，以防止非法MAC短时间内的重复认证。

Triple接入认证技术

在客户端形式多样的网络环境中，不同客户端支持的接入认证方式有所不同，为灵活地适应这种网络环境中的多种认证需求，需要在接入用户的端口上对多种认证方式进行统一部署，使得用户可以选择任何一种适合的认证机制来进行认证，且只需要成功通过一种方式的认证即可实现接入，无需通过多种认证。

Triple认证方案可满足以上需求，允许在设备的二层端口上同时开启Portal认证、MAC地址认证和802.1X认证功能，使得选用其中任意一种方式进行认证的客户端均可通过该端口接入网络。

→结束语←

用户身份识别与认证是网络安全的重要组成部分，对接入用户进行身份认证，是保障网络安全的重要环节。本文简单介绍了常见认证机制、认证协议以及接入认证技术，相关内容的详细介绍，请参考后续文章。■

摘 要：

九十年代中期以来，Internet 业务量的增长已构成数据业务的主要增长因素，IP成为电信网是不争的事实。但是目前的IP网络还不是一个电信级的网络，它的可运营、可管理特性同PSTN相比还存在较大差距。而且提高IP网络的可管理性是当前宽带网络发展急需解决的问题。从可管理性方面来说，对用户实施业务管理的前提就是解决用户的认证、授权和计费问题，即众所周知的AAA。目前，电信运营商和服务提供商所采用的认证方式主要有本地认证、RADIUS认证和不认证；而计费策略更是丰富多彩，常见的有不计费（包月）、按时长计费、按流量计费、按端口计费等等。目前在所有这些认证计费方式中，以采用RADIUS Server进行集中认证计费应用的最为普及和广泛。

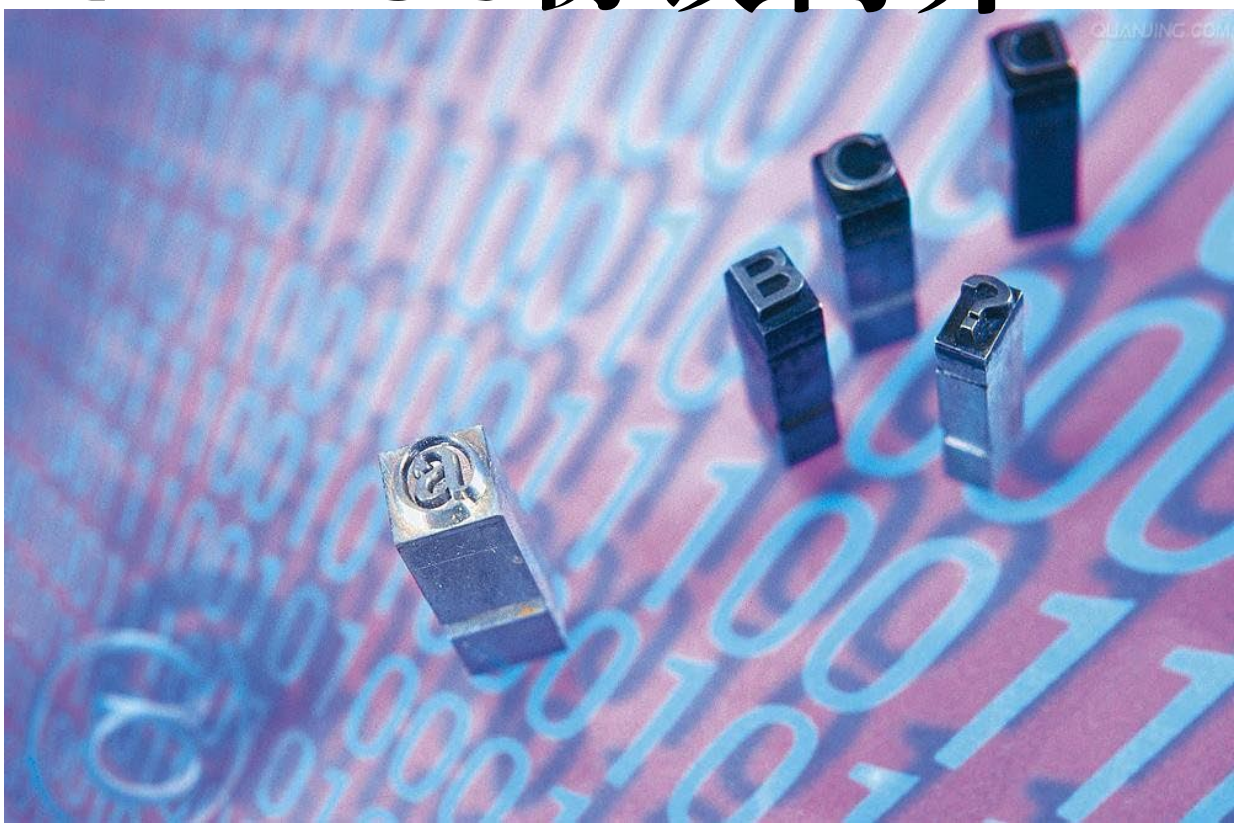
AAA是认证（Authentication）、授权（Authorization）和计费（Accounting）的简称，它是一种对网络安全进行管理的基本框架。而RADIUS（Remote Authentication Dial-In User Service）即远程认证拨号用户服务，是一种分布式的、客户机/服务器结构的信息交互协议，它能保护网络不受未授权访问的干扰。二者结合可以实现对远程接入用户身份、权限以及流量的严格控制。RADIUS为服务供应商和公司提供了一种灵活通用的协议,用来完成集中式的用户认证、口令加密、服务选择、过滤和帐目核对等工作。当进行PAP/CHAP连接或连接第三方认证服务器时,一个单独的RADIUS数据库服务器可以在多个复杂网络上同时管理多个安全系统,并可用于维护成千上万用户的信息安全。

RADIUS不仅指运行于服务器上的软件,还包括网络访问服务器与RADIUS服务器之间的交互操作协议。

以下即针对RADIUS协议的原理和具体的应用做详细介绍。

■文/袁亚屏

RADIUS协议简介



→AAA介绍←

AAA概述

AAA是认证（Authentication）、授权（Authorization）和计费（Accounting）的缩写，它是运行于NAS上的客户端程序。提供了一个用来对认证、授权和计费这三种安全功能进行配置的一致性框架，实际上是对网络安全的一种管理。这里的网络安全主要指访问控制。包括哪些用户可以访问网络服务器；具有访问权的用户可以得到哪些服务；以及如何对正在使用网络资源的用户进行计费。

下面简单介绍一下AAA所提供的三种服务：

1. 认证功能

AAA支持以下认证方式：

◆ 不认证：对用户非常信任，不对其检查，一般情况下不采用这种方式。

◆ 本地认证：将用户信息（包括本地用户的用户名、密码和各种属性）配置在接入服务器上。本地认证的优点是速度快，降低运营成本；但存储信息量受设备硬件条件限制。

◆ 远端认证：支持通过RADIUS协议或HWTACACS协议进行远端认证，由接入服务器作为Client端，与RADIUS服务器或TACACS服务器通信。

2. 授权功能

AAA支持以下授权方式：

◆ 直接授权：对用户非常信任，直接授权通过。

◆ 本地授权：根据宽带接入服务器上为本地用户账号配置的相关属性进行授权。

◆ HWTACACS授权：由TACACS服务器对用户进行授权。

◆ f-authenticated授权：如果用户通过了认证，并且使用的认证方法不是none，则对用户授权通过。

◆ RADIUS认证成功后授权：RADIUS协议的认证和授权是绑定在一起的，不能单独使用RADIUS进行授权。

3. 计费功能

AAA支持以下计费方式：

◆ 不计费

◆ 远端计费：支持通过RADIUS服务器或TACACS服务器进行远端计费。

AAA的实现

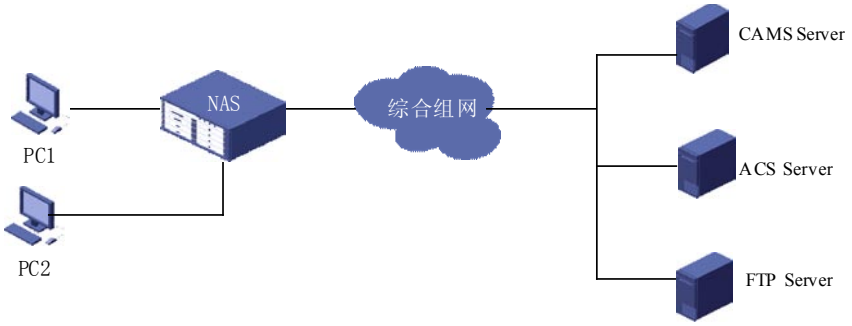


图1 AAA框架图

本地认证——即在NAS端进行认证、授权和计费，

远程认证——通过协议进行远程的认证、授权和计费。

AAA的实现可采用RADIUS协议。RADIUS用来管理使用串口和调制解调器的大量分散用户。

网络接入服务器简称NAS。当用户想要通过某个网络(如电话网)与NAS（网络接入服务器）建立连接从而获得访问其他网络的权利时，NAS可以选择在NAS上进行本地认证计费，或把用户信息传递给RADIUS服务器，由RADIUS进行认证计费；RADIUS 协议规定了NAS与RADIUS 服务器之间如何传递用户信息和计费信息，即两者之间的通信规则；RADIUS服务器负责接收用户的连接请求，完成认证，并把用户所需的配置信息返回给NAS。用户获得授权后，在其正常上线、在线和下线过程中，RADIUS服务器还完成对用户账号计费的功能。

→RADIUS协议概述←

引　论

RADIUS远程认证拨入用户服务，是用于网络访问服务器（NAS）和AAA服务器间通信的一种协议。RADIUS对AAA的三个组件都提供支持：认证、授权和计费。

一个网络允许外部用户通过公用网对其进行访问，于是用户在地理上可以极为分散。用户可以把自己的信息传递给这个网络，也可以从这个网络得到自己想要的信息。由于存在内外的双向数据流动，网络安全就成为很重要的问题了。管理的内容包括：哪些用户可以获得访问权，获得访问权的用户可以允许使用哪些服务，如何对使用网络资源的用户进行记费。AAA很好的完成了这三项任务。

RADIUS通过建立一个唯一的用户数据库，存储用户名，用户的密码来进行认证; 存储传递给用户的服务类型以及相应的配置信息来完成授权。

RADIUS关键特性

RADIUS协议有如下的关键特性：

客户端/服务器模式(Client/Server)

网络接入服务器（Network Access Server）是作为RADIUS的客户端运作的。这个客户端负责将用户信息传递给指定的RADIUS服务器，并负责执行返回的响应。RADIUS服务器负责接收用户的连接请求，鉴别用户，并为客户端返回所有为用户提供服务所必须的配置信息。一个RADIUS服务器可以为其他的RADIUS Server或其他种类认证服务器担当代理。

网络安全 (Network Security)

客户端和RADIUS服务器之间的事务是通过使用一种从来不会在网上传输的共享密钥机制进行鉴别的。另外，在客户端和RADIUS服务器之间的任何用户密码都是被加密后传输的，这是为了避免某些人在不安全的网络上监听获取用户密码的可能性。

灵活的认证机制 (Flexible Authentication Mechanisms)

RADIUS服务器能支持多种认证用户的方法。当用户提供了用户名和原始密码后，RADIUS服务器可以支持点对点的PAP认证（PPP PAP）、点对点的CHAP认证（PPP CHAP）、UNIX的登录操作（UNIX Login）、和其他认证机制。

扩展协议(Extensible Protocol)

RADIUS协议具有很好的扩展性。RADIUS包是由包头和一定数目的属性（Attribute）构成的。新的属性可以在不中断已存在协议执行的前提下进行增加。在报文属性中有详细描述。

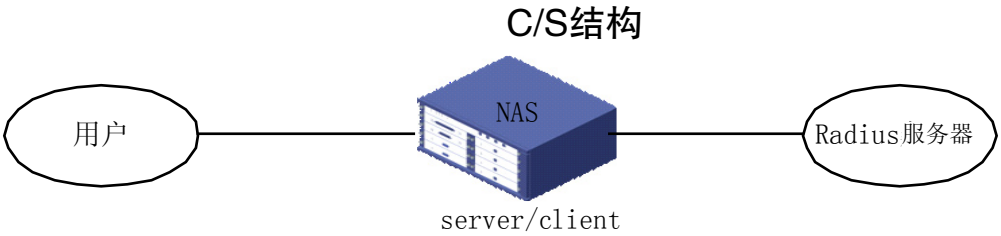


图2 用户， NAS，RADIUS 服务器的关系

RADIUS采用典型的客户/服务器（Client/Server）结构，它的客户端最初就是NAS（Network Access Server）服务器，现在任何运行RADIUS客户端软件的计算机都可以成为RADIUS的客户端。NAS上运行的AAA程序对用户来讲为服务器端，对RADIUS服务器来讲是作为客户端。负责传输用户信息到指定的RADIUS服务器，然后根据从服务器返回的信息进行相应处理（如接入/挂断用户）。RADIUS服务器负责接收用户连接请求，认证用户，然后给NAS返回所有需要的信息。

1、RADIUS的客户端通常运行于接入服务器（NAS）上，RADIUS服务器通常运行于一台工作站上，一个RADIUS服务器可以同时支持多个RADIUS客户（NAS）。

2、RADIUS的服务器上存放着大量的信息，接入服务器（NAS）无须保存这些信息，而是通过RADIUS协议对这些信息进行访问。这些信息的集中统一的保存，使得管理更加方便，而且更加安全。

3、RADIUS服务器可以作为一个代理，以客户的身份同其他的RADIUS服务器或者其他类型的认证服务器进行通信。用户的漫游通常就是通过RADIUS代理实现的。简单地说，代理就是一台服务器，可以作为其他RADIUS服务器的代理，负责转发RADIUS认证和计费数据包。所谓漫游功能，就是代理的一个具体实现，这样可以让用户通过本来和其无关的RADIUS服务器进行认证。

RADIUS在协议栈中的位置

RADIUS是一种流行的AAA协议，采用UDP协议传输，在协议栈中位置如下：

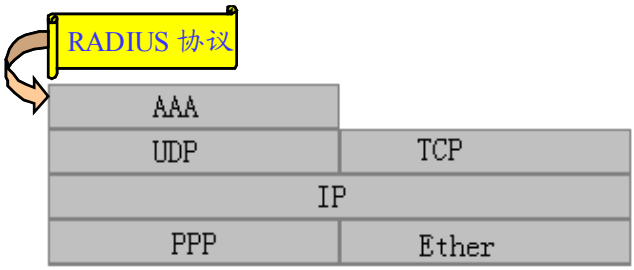


图3 在协议栈中的位置

RADIUS为何采用UDP，而不是TCP的原因如下：

1、 NAS和RADIUS服务器之间传递的一般是几十至上百个字节长度的数据，用户可以容忍几秒到十几秒的验证等待时间。当处理大量用户时服务器端采用多线程，UDP简化了服务器端的实现过程。

2、 TCP是必须成功建立连接后才能进行数据传输的，这种方式在有大量用户使用的情况下实时性不好。

3、当向主用服务器发送请求失败后，还要必须向备用的服务器发送请求。于是RADIUS要有重传机制和备用服务器机制，它所采用的定时，TCP不能很好的满足。

RADIUS协议采用的是UDP协议，数据包可能会在网络上丢失，如果客户没有收到响应，那么可以重新发送该请求包。多次发送之后如果仍然收不到响应，RADIUS客户可以向备用的RADIUS服务器发送请求包。

RADIUS网络安全

RADIUS协议的加密是使用MD5加密算法进行的，在RADIUS的客户端（NAS）和服务器端（RADIUS Server）保存了一个密钥（key），RADIUS协议利用这个密钥使用MD5算法对RADIUS中的数据进行加密处理。密钥不会在网络上传送。

RADIUS的加密主要体现在以下两方面：

包加密

在RADIUS包中，有16字节的验证字（authenticator）用于对包进行签名，收到RADIUS包的一方要查看该签名的正确性。如果包的签名不正确，那么该包将被丢弃，对包进行签名时使用的也是MD5算法（利用密钥），没有密钥的人是不能构造出该签名的。

口令加密

在认证用户时，用户的口令在NAS和RADIUS Server之间不会以明文方式传送，而是使用了MD5算法对口令进行加密。没有密钥的人是无法正确加密口令的，也无法正确地对加密过的口令进行解密。

口令加密与口令验证过程

当用户上网时，NAS将决定对用户采用何种认证方法。下面对使用RADIUS认证的情况下PPP用户与NAS之间的PAP和CHAP认证过程进行介绍。

RADIUS服务器可使用H3C的iMC服务器，也可以使用CISCO的ACS服务器或其它第三方服务器。

PAP 验证：

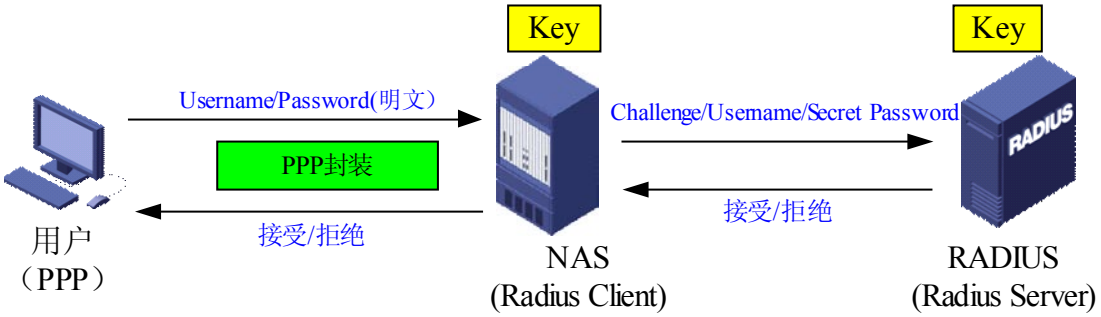


图4 RADIUS Server PAP认证

用户以明文的形式把用户名和他的密码传递给NAS，NAS把用户名和加密过的密码放到验证请求包的相应属性中传递给RADIUS服务器。RADIUS服务器根据NAS上传的帐号进行验证来决定是否允许用户

上网并返回结果。NAS可以在其中包含服务类型属性AttributeService-Type=Framed-User，和Framed-Protocol=PPP作为提示来告诉RADIUS服务器PPP是所希望的服务。

Secretpassword =Password XOR MD5（Challenge ＋Key）

（Challenge就是RADIUS报文中的Authenticator）

CHAP 验证：

对于CHAP（挑战握手认证协议），它提供对用户口令进行加密的机制。

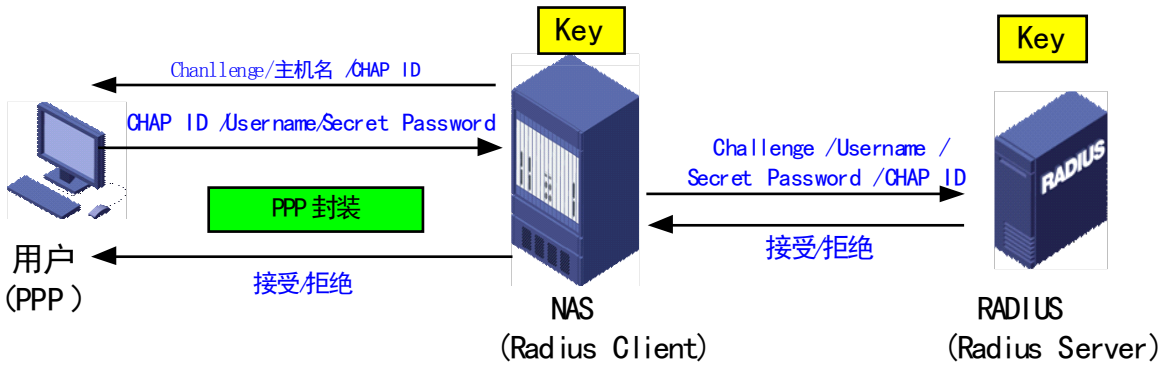


图10 RADIUS Server CHAP认证

当用户请求上网时，NAS产生一个16字节的随机码给用户（同时还有一个ID号，本地路由器的Host name）。用户端得到这个包后使用自己独有的设备或软件客户端将CHAP ID、用户密码（口令字）用MD5算法对该随机码进行加密生成一个Secret Password，随同用户名user name一并传给NAS。

NAS把传回的user name和Secret Password分别作为用户名和密码，并把原来的16字节随机码以及CHIP ID传给RADIUS服务器。RADIUS根据用户名在服务器端查找数据库，得到和用户端进行加密所用的一样的密钥，用MD5算法对CHAPID，密钥和传来的16字节的随机码进行加密，将其结果与传来的Password作比较，如果相匹配，服务器送回一个接入允许数据包，否则送回一个接入拒绝数据包。

→RADIUS体系结构←

当用户PC通过某个网络(如电话网)与 NAS建立连接从而获得访问其他网络的权利时，NAS可以选择在NAS上进行本地认证计费，或把用户信息传递给RADIUS服务器，由RADIUS进行认证计费；RADIUS协议规定了NAS与RADIUS 服务器之间如何传递用户信息和计费信息；RADIUS服务器负责接收用户的连接请求，完成认证，并把传递服务给用户所需的配置信息返回给NAS。

通常对RADIUS协议的认证服务端口号是1645（早期实现）或1812，计费服务端口号是1646（早期实现）或1813。当我们进行NAS设备和RADIUS服务器对接的时候，由于生产厂家可能不同，所对应的服务器端口号不同，那么就需要调整服务器端口号，例如与其他厂家RADIUS协议默认端口号是1645和1646，需要与他们协商，调整到相同的端口号，这样才能对接成功。

RADIUS报文的交互流程

RADIUS服务器对用户的认证过程通常需要利用NAS等设备的代理认证功能，RADIUS客户端和

RADIUS服务器之间通过共享密钥认证相互间交互的消息，用户密码采用密文方式在网络上传输，增强了安全性。RADIUS协议合并了认证和授权过程，即响应报文中携带了授权信息。操作流程图和步骤如下所示：

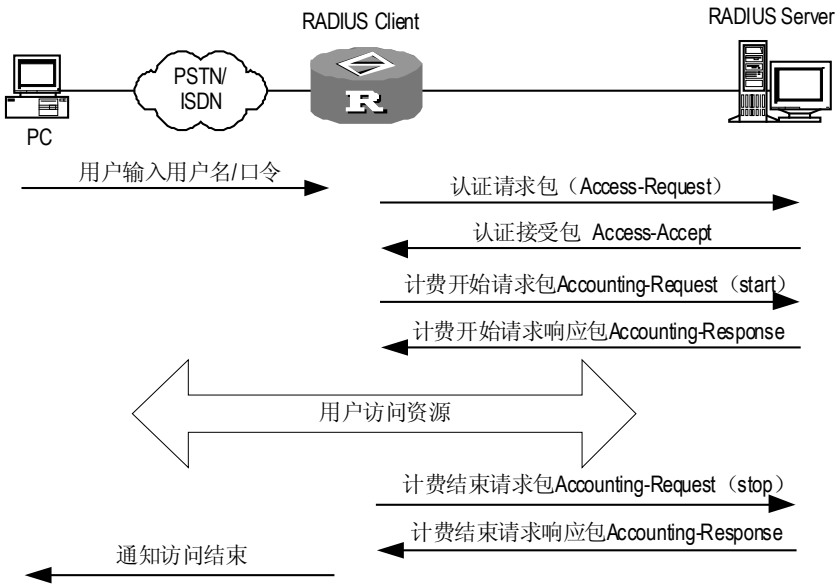


图6 我司RADIUS的典型实现流程

在一个客户端被设置使用RADIUS协议后，任何使用这个终端的用户都需要向这个客户端提供认证信息。该信息或者以一种定制化的提示信息出现，用户需要输入他们的用户名和密码；或者也可以选择比如像点对点的传输协议（Point-to-Point Protocol）,通过认证包来传递这种认证信息。

基本交互过程如下：

1) 用户输入用户名和口令；

2) RADIUS客户端根据获取的用户名和口令，向RADIUS服务器发送认证请求包（Access-Request）。

3) RADIUS服务器将该用户信息与Users数据库信息进行对比分析，如果认证成功，则将用户的权限信息以认证响应包（Access-Accept）发送给RADIUS客户端；如果认证失败，则返回Access-Reject响应包。

4) RADIUS客户端根据接收到的认证结果接入/拒绝用户。如果可以接入用户，则RADIUS客户端向RADIUS服务器发送计费开始请求包（Accounting-Request），Status-Type取值为start；

5) RADIUS服务器返回计费开始响应包（Accounting-Response）；

6) RADIUS客户端向RADIUS服务器发送计费停止请求包（Accounting-Request），Status-Type取值为stop；

7) RADIUS服务器返回计费结束响应包（Accounting-Response）。

RADIUS协议的报文结构

RADIUS采用UDP传输消息，通过定时器管理机制、重传机制、备用服务器机制，确保RADIUS服务

器和客户端之间交互消息正确收发。标准RADIUS协议包结构如下：

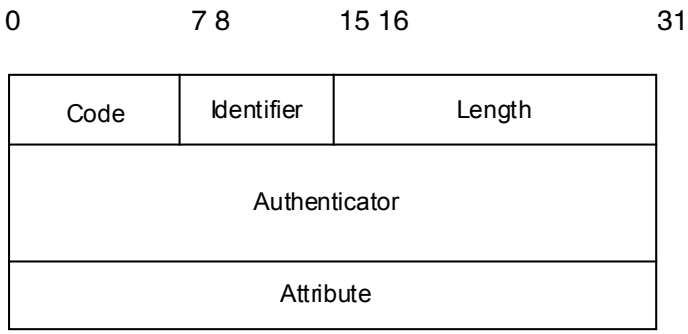


图7 RADIUS包格式

Code：包类型；1字节，指示RADIUS包的类型。在接收到一个无效编码域的数据包后，该数据包只是会被简单的丢弃。

主要包括如下类型：

表1 Code域主要取值的说明

Code	报文类型	报文说明
1	Access-Request认证请求包	方向Client->Server，Client将用户信息传输到Server以判断是否接入该用户。该报文中必须包含User-Name属性，可选包含NAS-IP-Address、User-Password、NAS-Port等属性。
2	Access-Accept认证接受包	方向Server->Client，如果Access-Request报文中所有Attribute值都是可以接受（即认证通过），则传输该类型报文。
3	Access-Reject认证拒绝包	方向Server->Client，如果Access-Request报文中存在任何Attribute值无法被接受（即认证失败），则传输该类型报文。
4	Accounting-Request计费请求包	方向Client->Server，Client将用户信息传输到Server，请求Server开始计费，由该报文中的Acct-Status-Type属性区分计费开始请求和计费结束请求。该报文包含属性和Access-Request报文大致相同。
5	Accounting-Response认证响应包	方向Server->Client，Server通知Client侧已经收到Accounting-Request报文并且已经正确记录计费信息。该报文包含端口上输入/输出字节数、输入/输出包数、会话时长等信息。

Identifier： 包标识；1字节，取值范围为0 ~255；用于匹配请求包和响应包，同一组请求包和响应包的Identifier应相同。如果在一个很短的时间片段里，一个请求有相同的客户源IP地址、源UDP端口号和标识符，RADIUS服务器会认为这是一个重复的请求而不响应处理。1字节意味着客户端在收到服务器的响应之前最多发送255（28-1）个请求。Identifier段里的值是序列增长的。

Length： 包长度；2字节；标识了分组的长度，整个包中所有域的长度。长度域范围之外的字节被认为是附加的，并在接受的时候超长部分将被忽略。如果包长比长度域给出的短，也必须丢弃，最小长度为20，最大长度为4096。

Authenticator：验证字域；16 字节明文随机数；最重要的字节组最先被传输。该值用来认证来自RADIUS服务器的回复，也用于口令隐藏算法（加密）。

该验证字分为两种：

请求验证字——Request Authenticator

用在请求报文中，采用唯一的16字节随机码。

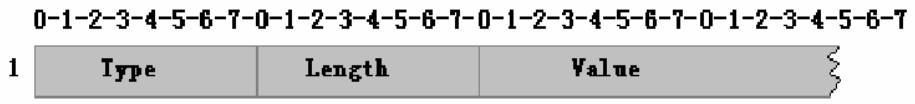
响应验证字——Response Authenticator

用在响应报文中，用于鉴别响应报文的合法性。

响应验证字＝MD5(Code+ID+Length+请求验证字+Attributes+Key)

Attributes：属性域；用来在请求和响应报文中携带详细的认证、授权、信息和配置细节，来实现认证、授权、计费等功能。采用（Type、Length、Value）三元组的形式提供，

属性在数据报中的格式：



Type：Attribute Number（属性号），用来说明数据包中表示的属性类型。范围是1~255。服务器和客户端都可以忽略不可识别类型的属性。值为26，标识厂商私有属性。

Length（属性长度)：说明整个属性域的长度。长度域最小值为3（表示属性域至少占3个字节）。如果AAA服务器收到的接入请求中属性长度无效，则发送接入拒绝包。如果NAS收到的接入允许、接入拒绝和接入盘问中属性的长度也无效，则必须以接入拒绝对待，或者被简单的直接丢弃。

Value（值）：表示属于自己的特性和特征。即使该域为null，这个域也必须出现在属性域中。有6种属性值——整数(INT)；枚举(ENUM)；IP地址(IPADDR)；文本(STRING)；日期(DATE)；二进制字符串（BINARY)；

RFC中定义的标准Attribute域大致包括：

表2 Attribute域主要取值的说明

Type	属性类型	Type	属性类型
1	User–Name	23	Framed–IPX–Network
2	User–Password	24	State
3	CHAP–Password	25	Class
4	NAS–IP–Address	26	Vendor–Specific
5	NAS–Port	27	Session–Timeout
6	Service–Type	28	Idle–Timeout
7	Framed–Protocol	29	Termination–Action
8	Framed–IP–Address	30	Called–Station–Id
9	Framed–IP–Netmask	31	Calling–Station–Id
10	Framed–Routing	32	NAS–Identifier
11	Filter–ID	33	Proxy–State
12	Framed–MTU	34	Login–LAT–Service
13	Framed–Compression	35	Login–LAT–Node
14	Login–IP–Host	36	Login–LAT–Group
15	Login–Service	37	Framed–AppleTalk–Link
16	Login–TCP–Port	38	Framed–AppleTalk–Network
17	(unassigned)	39	Framed–AppleTalk–Zone
18	Reply_Message	40–59	(reserved for accounting)
19	Callback–Number	60	CHAP–Challenge
20	Callback–ID	61	NAS–Port–Type
21	(unassigned)	62	Port–Limit
22	Framed–Route	63	Login–LAT–Port

RADIUS协议具有良好的可扩展性，协议中定义的26号属性（Vender-Specific）被用来扩展以支持供应商自己定义的扩展属性，主要指不适于常规使用的属性扩展。但不允许对RADIUS协议中的操作有影响。当服务器不具备去解释由客户端发送过来的供应商特性信息时，则服务器必须忽略它（过程可以被记录下来）。当没有收到预期的供应商特性信息的情况下，客户端也应该尝试在没有它的情况下运作（即使是在降级模式中）。

以下是具体厂商属性格式的总结。域的传输是从左向右。报文结构如下图所示：

Type	Length	Vendor-ID	
Vendor-ID		type (specified)	length (specified)
specified attribute value			

图8 自定义属性格式

RADIUS细节描述

下图是针对RADIUS远程认证时，认证、授权和计费的交互的详细过程。

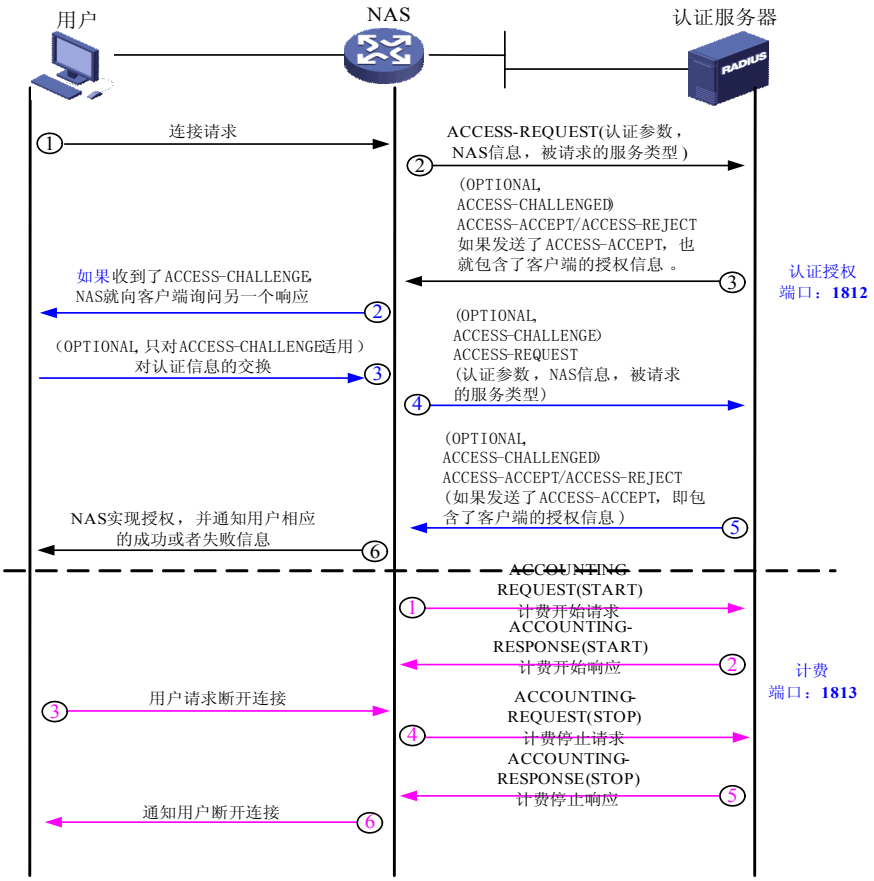


图9 RADIUS认证、授权和计费

综述：RADIUS认证、授权和计费

RADIUS的认证和授权的相关内容已经在前面的章节里中介绍过。下面将其从分散的描述中作细节描述。

认证

RADIUS的认证是以NAS和RADIUS服务器之间交换两个消息的形式进行的。

（1）用户发起连接请求

（2）NAS在接受到用户进行认证的请求后，发送一个访问-请求（Access-Request）消息到RADIUS服务器上。

访问-请求消息包含一下信息：

◆ 用户名

◆ 加密个时的用户口令

◆ NAS IP 地址和端口号

◆ 用户希望的服务类型

（3）RADIUS服务器使用以下消息之一进行相应：

◆ 访问-接受　Access-Accept

◆ 访问-拒绝　Access-Reject

◆ 访问-询问　Access-Challenge

如果AAA服务器对用户进行了成功的认证，就会发送一个访问-接受Access-Accept消息。该消息也包含了被应用到用户的授权属性。

当AAA服务器不接受NAS提供给AAA服务器的任何一个值，就会发送一个访问-拒绝Access-Reject消息。该消息也可能包含一到多个回答消息属性，以及一个NAS可能显示给用户的文本消息。在这一点上，NAS决定用户是否在使用另外一个AAA服务器进行认证。

如果AAA服务器需要询问用户以获取一个新的口令，它就会发送包含对用户询问的Access-Challenge消息到NAS。NAS将此消息发送给用户，然后将用户名和询问响应以Access-Request消息转发给AAA服务器。接着AAA服务器用相应的响应作为回答。

授权

RADIUS授权和认证过程是协同进行的。当RADIUS服务器返回一个Access-Accept消息，该消息也包含了一系列的描述用户具有的会话属性（在RADIUS属性域内）。该信息即对NAS进行授权的信息。NAS实现授权，并通知用户相应的成功或者失败信息。

在接受消息中包含的授权数据一般如下：

◆ 用户可以访问的服务，包括Telnet、rlogin、或者本地传输（LAT）连接，以及PPP、穿行Internet协议（SLIP）或者EXEC服务。

◆ 连接参数，包括主机或者客户端IP地址、访问列表和用户超时值。

上图中服务器和客户端的交互实现了对用户进行服务授权。

计费

RADIUS计费通过在会话的开始和结束的时候发送包含有关会话的消息执行。RADIUS计费功能允许

在服务开始和结束时发送数据，标识会话期间众多信息中资源的数量。消息被发送到UDP端口1813。

当一个客户端被配置为使用RADIUS计费时，在服务传送开始时，NAS将产生一个计费开始请求包，用来描述被传送的服务类型以及服务将传送到的用户，并向RADIUS服务器发送该计费开始请求包（Accounting-Request）（1），RADIUS服务器收到并成功记录该请求包后要回应一个计费响应（Accounting-Response）（2）。

当用户断开连接时（连接也可以由接入服务器断开）（3），NAS向RADIUS服务器发送一个记费停止请求包（4），其中包含用户上网所使用网络资源的统计信息（上网时长、进/出的字节/包数等），RADIUS服务器会返回对该计费停止请求包的确认（5）。

由于计费是用户敏感的问题，我们提供的计费信息要求准确外更要求完备，不能丢失话单，而RADIUS计费服务器的计费很容易受通讯不畅或计费服务器工作超负荷等意外因素的影响而丢失话单，为此我们提供了本地计费的保护能力，即话单从NAS发送出去以后如果没有收到计费服务器的确认信息我们就认为计费失败，并将该话单（不论是实时计费中间话单还是离线话单）送到本地计费数据库中。

Shared Secrets共享密钥

共享密钥是带外传输（客户端和服务器都知道的一个字符串，不需要单独进行网络传输）的一个随机值。它用来提供报文数据的加密和解密。

推荐共享密钥长度16个字节以保证安全性，这个密钥长度不可能用暴力破解来解密。

现在大规模的RADIUS服务的应用，使得动态密钥协商机制来保护传输的安全性是应该被谨慎使用的。动态密钥协商机制存在的缺陷有：

1、不能保证在恰当的时刻同步到密钥

2、即使能发生同步，但当client忙于处理重要请求时，可能把同步新密钥的请求丢了，同时老密钥失效。

3、相当于当银行给你一个新账号时，你没有收到。同时老账号注销，则可能到老账号的钱不能成功打入。

4、RADIUS客户端、服务器端与共享密钥一一对应的。也就是说共享密钥对特定的RADIUS服务器和客户端对是唯一的。

那么当在网络传输中有多个共享密钥时，共享密钥通过报文的源IP地址进行区分，如何与RADIUS服务器、客户端对进行一一对应。

RADIUS代理

对RADIUS代理服务器来说，一个RADIUS服务器在收到一个来自RADIUS客户端（例如NAS服务器）的认证请求（或者计费请求）后，向一个远程的RADIUS服务器提交该请求，收到来自远程服务器的回复后，将这个回复传输给客户，这个回复可能带有反映本地管理策略的变化。使用RADIUS代理服务器通常是为了漫游。漫游功能使两个或更多的管理实体允许每一个用户为某项服务而拨入到任一个实体网络中。

一个RADIUS服务器可以同时作为转送服务器和远程服务器运行。在某些域中作为一个转发服务

器，在其他域中作为一个远程服务器。一个转发服务器可以作为任何数量远程服务器的转发者。一个远程服务器可以有任意数量的转发服务器向它转发，也能向任意数量域提供认证。一个转发服务器可以向另一个转发服务器转发，从而生成一个代理链，应当注意避免循环引用。

下面的过程解释了一个代理服务器在一个NAS服务器、转发服务器和远程服务器之间的通信。

1. NAS向一个转发服务器发出接入请求。
2. 转发服务器把这个请求转发给一个远程服务器。
3. 远程服务器给转发服务器送回接入允许、接入拒绝或接入盘问。此时，服务器送回的是接入允许。
4. 转发服务器将接入允许传输给NAS。

转发服务器必须把已经存在于数据包中的任何代理状态属性当作不可见的数据。它的操作禁止依靠被前面服务器添加到代理状态属性中的内容

如果收到来自客户端的请求中有任何代理状态属性，在给客户端的回复中，转发服务器必须在给客户端的回复中包括这些代理状态属性。当转发服务器转发这个请求时，它可以把代理状态属性包含在其中，也可以在已转发的请求中忽略代理状态属性。如果转发服务器在转发的接入请求中忽略了代理状态属性，它必须在响应返回给用户之前把这些代理状态属性添加到该响应中。

RADIUS配置及实现

RADIUS远程认证实例

配置需求

用户主机PC或者路由器MSR30-20上运行PPPoE Client，通过路由器MSR50-40接入到综合组网上。路由器作为PPPoE Server，配置本地认证，并通过地址池为用户分配IP地址。PPPOE用户通过MSR50-40作为接入设备接入到CAMS Server上进行远程RADIUS的认证授权和计费。

配置步骤

（1）在NAS上配置RADIUS方案

（2）在RADIUS服务器上设置与路由器交互报文时的共享密钥，添加PPP用户名及密码。

在NAS上配置如下：

```
[up-isp-aaa]dis th
#
domain aaa
 scheme radius-scheme luo
 authentication radius-scheme luo
 ip pool 50 50.1.1.1 50.1.1.10
domain system
 ip pool 0 70.1.1.1 70.1.1.10
#
```

```
return
[up] radius scheme  luo
[up-radius-luo]dis  this
#
 radius scheme system
 accounting primary ip 127.0.0.1 port 1813
  radius scheme luo
 authentication primary ip 101.3.209.1 key simple name
 accounting primary ip 101.3.209.1 key simple name
 timer realtime-accounting 3
 user-name-format without-domain
#
return
```

调试信息

（一）报文说明

（1）RADIUS报文中，code＝1是认证请求报文，code＝2和code＝3分别是认证通过和认证失败报文，这三种报文只在用户上线时产生。在用户上网的漫长过程中，是依靠code＝4报文来维系计费和用户在线信息。

关于RADIUS里面code＝4和code＝5的报文：

code＝4：计费请求，由客户端发出（一般是配置了RADIUS的路由器）

code＝5：计费回应。由RADIUS服务器发出（CAMS发/ACS发）

开始计费和结束计费是由code＝4里面的属性Acct-Status-Type(40)来决定的。

1、Start （计费开始）

2、Stop （计费结束）

3、Interim-Update（计费更新）

4、Reset-Charge

7、Accounting-On

8、Accounting-Off

9-14、Reserved for Tunnel Accounting

15、Reserved for Failed

其中Reset-Charge (Value=4)报文类型是针对华为扩展RADIUS定义的。

我们主要看1、2、3这三个属性，后面的属性暂时看不到。

（2）PPP用户和LOGIN用户在认证时，在认证请求包的属性域中，Service-Type是不相同的。

* PPP拨号用户的Service-Type为User-Password；
* LOGIN用户的Service-Type为User-Name。

（二）PPP用户

以下是PPP拨号用户在采用cams认证时，NAS上一个完整的上线、计费、更新、下线信息的DEBUG信息分析：

PPP用户上线——认证请求（code=1）

```
%Oct 27 19:15:05:227 2006 up IFNET/5/UPDOWN:Line protocol on the interface Virtual-Template1:0 is UP
*0.634967270 up RDS/8/DEBUG:Recv MSG,[MsgType=Normal auth request Index = 31, ulParam3=645247172]
*0.634967270 up RDS/8/DEBUG:Send attribute list:
*0.634967271 up RDS/8/DEBUG:
[1  User-name                ] [5 ] [luo]
[3  Challenge-Password       ] [19] [0116BF15A92FCCDC97E708507967157C5A]
[60 CHAP_Challenge           ] [18] [3BE7264A3CFB67318E8CFEAFDAA6481C]
[4  NAS-IP-Address           ] [6 ] [192.168.100.22]
[32 NAS-Identifier           ] [4 ] [up]
[5  NAS-Port                 ] [6 ] [0]
*0.634967271 up RDS/8/DEBUG:
[61 NAS-Port-Type            ] [6 ] [5]
[6  Service-Type             ] [6 ] [2]    属性类型：User-Password
[7  Framed-Protocol          ] [6 ] [1]
*0.634967271 up RDS/8/DEBUG:Send: IP=[101.3.201.1], UserIndex=[31], ID=[10], RetryTimes=[0], Code=[1], Length=[96]
*0.634967271 up RDS/8/DEBUG:Send Raw Packet is:
*0.634967271 up RDS/8/DEBUG:
 01 0a 00 60 3b e7 26 4a 3c fb 67 31 8e 8c fe af
 da a6 48 1c 01 05 6c 75 6f 03 13 01 16 bf 15 a9
 2f cc dc 97 e7 08 50 79 67 15 7c 5a 3c 12 3b e7
 26 4a 3c fb 67 31 8e 8c fe af da a6 48 1c 04 06
 c0 a8 64 16 20 04 75 70 05 06 00 00 00 00 3d 06
 00 00 00 05 06 06 00 00 00 02 07 06 00 00 00 01
```

PPP用户上线——认证响应（code=2）

```
*0.634967280 up RDS/8/DEBUG:Recv MSG,[MsgType=PKT response Index = 90, ulParam3=645956420]
*0.634967280 up RDS/8/DEBUG:Receive Raw Packet is:
*0.634967280 up RDS/8/DEBUG:
 02 0a 00 5a 22 69 9f b9 90 b2 7e 90 35 e5 1d fc
 94 f7 79 c5 06 06 00 00 00 01 07 06 00 00 00 01
```

```
09 06 00 00 00 00 0e 06 00 00 00 00 0f 06 00 00
00 00 58 04 35 30 08 06 ff ff ff fe 19 1e 43 49
53 43 4f 41 43 53 3a 30 30 30 34 35 61 33 33 2f
63 30 61 38 36 34 31 36 2f 30
*0.634967280 up RDS/8/DEBUG:Receive:IP=[101.3.201.1],Code=[2],Length=[90]
*0.634967280 up RDS/8/DEBUG:
[6  Service-Type          ] [6 ] [1]      //与LOGIN用户相同
[7  Framed-Protocol       ] [6 ] [1]
[9  Framed-Netmask        ] [6 ] [0.0.0.0]
[14 Login-Host            ] [6 ] [0.0.0.0]
[15 Login-Service         ] [6 ] [0]
[88 Framed-Pool           ] [4 ] [50]
*0.634967280 up RDS/8/DEBUG:
[8  Framed-Address        ] [6 ] [255.255.255.254]
[25 Class                 ] [30] [434953434F4143533A30303034356133332F63306138363431362
F30]
```

PPP用户上线——计费开始请求（code=4）

```
*0.634967300 up RDS/8/DEBUG:Recv MSG,[MsgType=Account request Index = 31, ulParam3=0]
*0.634967300 up RDS/8/DEBUG:Send attribute list:
*0.634967300 up RDS/8/DEBUG:
[1  User-name             ] [5 ] [luo]
[32 NAS-Identifier        ] [4 ] [up]
[5  NAS-Port              ] [6 ] [0]
[61 NAS-Port-Type         ] [6 ] [5]
[31 Caller-ID             ] [16] [303030302D303030302D30303030]
[40 Acct-Status-Type      ] [6 ] [1]      ——计费开始
*0.634967300 up RDS/8/DEBUG:
[45 Acct-Authentic        ] [6 ] [1]
[44 Acct-Session-Id       ] [14] [106092719159]
[4  NAS-IP-Address        ] [6 ] [192.168.100.22]
[55 Event-Timestamp       ] [6 ] [1161976505]
[25 Class                 ] [30] [434953434F4143533A30303034356133332F63306138363431362
F30]
*0.634967300 up RDS/8/DEBUG:Send: IP=[101.3.201.1], UserIndex=[31], ID=[19], RetryTimes=[0],
Code=[4], Length=[125]
*0.634967300 up RDS/8/DEBUG:Send Raw Packet is:
```

```
*0.634967300 up RDS/8/DEBUG:
 04 13 00 7d df 96 41 82 21 5a a2 c3 8c 78 07 16
 db c9 6a 13 01 05 6c 75 6f 20 04 75 70 05 06 00
 00 00 00 3d 06 00 00 00 05 1f 10 30 30 30 30 2d
 30 30 30 30 2d 30 30 30 30 28 06 00 00 00 01 2d
 06 00 00 00 01 2c 0e 31 30 36 30 39 32 37 31 39
 31 35 39 04 06 c0 a8 64 16 37 06 45 42 5a b9 19
 1e 43 49 53 43 4f 41 43 53 3a 30 30 30 34 35 61
 33 33 2f 63 30 61 38 36 34 31 36 2f 30
%Oct 27 19:15:05:320 2006 up IFNET/5/UPDOWN:PPP IPCP protocol on the interface Virtual-Template1:0 is UP
%Oct 27 19:15:05:321 2006 up IFNET/5/UPDOWN:Line protocol on the interface Virtual-Template1 is UP
```

PPP用户上线——计费响应（code=5）

```
*0.634967370 up RDS/8/DEBUG:Recv MSG,[MsgType=PKT response Index = 20, ulParam3=645954308]
*0.634967370 up RDS/8/DEBUG:Receive Raw Packet is:
*0.634967370 up RDS/8/DEBUG:
 05 13 00 14 30 51 86 44 f9 8a a8 02 7b 98 c2 32
 05 62 1b 77
*0.634967370 up RDS/8/DEBUG:Receive:IP=[101.3.201.1],Code=[5],Length=[20]
*0.634967370 up RDS/8/DEBUG:NULL
```

PPP用户上线——实时计费请求（code=4）

```
*0.635067330 up RDS/8/DEBUG:Send attribute list:
*0.635067330 up RDS/8/DEBUG:
[1  User-name              ] [5 ] [luo]
[32 NAS-Identifier         ] [4 ] [up]
[5  NAS-Port               ] [6 ] [0]
[61 NAS-Port-Type          ] [6 ] [5]
[31 Caller-ID              ] [16] [303030302D303030302D30303030]
[40 Acct-Status-Type       ] [6 ] [3]    ——计费更新
*0.635067330 up RDS/8/DEBUG:
[45 Acct-Authentic         ] [6 ] [1]
[44 Acct-Session-Id        ] [14] [106092719159]
[8  Framed-Address         ] [6 ] [50.1.1.8]
[4  NAS-IP-Address         ] [6 ] [192.168.100.22]
[55 Event-Timestamp        ] [6 ] [1161976605]
```

```
[25 Class                    ] [30] [434953434F4143533A30303034356133332F63306138363431362F30]
*0.635067330 up RDS/8/DEBUG:
[46 Acct-Session-Time        ] [6 ] [100]
[41 Acct-Delay-Time          ] [6 ] [0]
[42 Acct-Input-Octets        ] [6 ] [0]
[47 Acct-Input-Packets       ] [6 ] [0]
[43 Acct-Output-Octets       ] [6 ] [736]
[48 Acct-Output-Packets      ] [6 ] [4]
*0.635067331 up RDS/8/DEBUG:
[52 Acct_Input_Gigawords     ] [6 ] [0]
[53 Acct_Output_Gigawords    ] [6 ] [0]
*0.635067331 up RDS/8/DEBUG:Send: IP=[101.3.201.1], UserIndex=[31], ID=[20], RetryTimes=[0], Code=[4], Length=[179]
*0.635067331 up RDS/8/DEBUG:Send Raw Packet is:
*0.635067331 up RDS/8/DEBUG:
 04 14 00 b3 4e 57 f3 21 05 5a 0b c1 77 c7 f4 76
 70 13 dd b7 01 05 6c 75 6f 20 04 75 70 05 06 00
 00 00 00 3d 06 00 00 00 05 1f 10 30 30 30 30 2d
 30 30 30 30 2d 30 30 30 30 28 06 00 00 00 03 2d
 06 00 00 00 01 2c 0e 31 30 36 30 39 32 37 31 39
 31 35 39 08 06 32 01 01 08 04 06 c0 a8 64 16 37
 06 45 42 5b 1d 19 1e 43 49 53 43 4f 41 43 53 3a
 30 30 30 34 35 61 33 33 2f 63 30 61 38 36 34 31
 36 2f 30 2e 06 00 00 00 64 29 06 00 00 00 00 2a
 06 00 00 00 00 2f 06 00 00 00 00 2b 06 00 00 02
 e0 30 06 00 00 00 04 34 06 00 00 00 00 35 06 00
 00 00 00
```

PPP用户在线——实时计费响应（code=5）

```
*0.635067340 up RDS/8/DEBUG:Recv MSG,[MsgType=PKT response Index = 20, ulParam3=645976580]
*0.635067340 up RDS/8/DEBUG:Receive Raw Packet is:
*0.635067340 up RDS/8/DEBUG:
 05 14 00 14 e7 ce bc c2 eb 7b 74 7d 1d 2c 17 77
 bc 71 0d 9f
*0.635067340 up RDS/8/DEBUG:Receive:IP=[101.3.201.1],Code=[5],Length=[20]
*0.635067340 up RDS/8/DEBUG:NULL
```

PPP用户下线——计费结束请求（code=4）

```
%Oct 27 19:17:04:345 2006 up IFNET/5/UPDOWN:Line protocol on the interface Virtu
al-Template1:0 is DOWN
%Oct 27 19:17:04:345 2006 up IFNET/5/UPDOWN:PPP IPCP protocol on the interface Virtual-
Template1:0 is DOWN
%Oct 27 19:17:04:345 2006 up IFNET/5/UPDOWN:Line protocol on the interface Virtu
al-Template1 is DOWN
*0.635086370 up RDS/8/DEBUG:Recv MSG,[MsgType=Account off request Index = 31,
ulParam3=0]
*0.635086370 up RDS/8/DEBUG:Send attribute list:
*0.635086370 up RDS/8/DEBUG:
[1  User-name                ] [5 ] [luo]
[32 NAS-Identifier           ] [4 ] [up]
[5  NAS-Port                 ] [6 ] [0]
[61 NAS-Port-Type            ] [6 ] [5]
[31 Caller-ID                ] [16] [303030302D303030302D30303030]
[40 Acct-Status-Type         ] [6 ] [2]   ——计费结束
*0.635086370 up RDS/8/DEBUG:
[45 Acct-Authentic           ] [6 ] [1]
[44 Acct-Session-Id          ] [14] [106092719159]
[8  Framed-Address           ] [6 ] [50.1.1.8]
[4  NAS-IP-Address           ] [6 ] [192.168.100.22]
[55 Event-Timestamp          ] [6 ] [1161976624]
[25 Class                    ] [30] [434953434F4143533A30303034356133332F63306138363431362
F30]
*0.635086371 up RDS/8/DEBUG:
[46 Acct-Session-Time        ] [6 ] [119]
[41 Acct-Delay-Time          ] [6 ] [0]
[42 Acct-Input-Octets        ] [6 ] [0]
[47 Acct-Input-Packets       ] [6 ] [0]
[43 Acct-Output-Octets       ] [6 ] [736]
[48 Acct-Output-Packets      ] [6 ] [4]
*0.635086371 up RDS/8/DEBUG:
[52 Acct_Input_Gigawords     ] [6 ] [0]
[53 Acct_Output_Gigawords    ] [6 ] [0]
[49 Terminate-Cause          ] [6 ] [1]
```

```
*0.635086371 up RDS/8/DEBUG:Send: IP=[101.3.201.1], UserIndex=[31], ID=[21], RetryTimes=[0], Code=[4], Length=[185]
*0.635086371 up RDS/8/DEBUG:Send Raw Packet is:
*0.635086371 up RDS/8/DEBUG:
 04 15 00 b9 00 8f df b8 20 ff e9 e9 e5 31 6c 41
 c5 e4 8d 1b 01 05 6c 75 6f 20 04 75 70 05 06 00
 00 00 00 3d 06 00 00 00 05 1f 10 30 30 30 30 2d
 30 30 30 30 2d 30 30 30 30 28 06 00 00 00 02 2d
 06 00 00 00 01 2c 0e 31 30 36 30 39 32 37 31 39
 31 35 39 08 06 32 01 01 08 04 06 c0 a8 64 16 37
 06 45 42 5b 30 19 1e 43 49 53 43 4f 41 43 53 3a
 30 30 30 34 35 61 33 33 2f 63 30 61 38 36 34 31
 36 2f 30 2e 06 00 00 00 77 29 06 00 00 00 00 2a
 06 00 00 00 00 2f 06 00 00 00 00 2b 06 00 00 02
 e0 30 06 00 00 00 04 34 06 00 00 00 00 35 06 00
 00 00 00 31 06 00 00 00 01
```

PPP用户下线——计费结束响应（code=5）

```
*0.635086440 up RDS/8/DEBUG:Recv MSG,[MsgType=PKT response Index = 20, ulParam3=643952004]
*0.635086440 up RDS/8/DEBUG:Receive Raw Packet is:
*0.635086440 up RDS/8/DEBUG:
 05 15 00 14 42 04 6d 16 e7 1a 92 d4 dc ed c2 98
 55 55 1b 66
*0.635086440 up RDS/8/DEBUG:Receive:IP=[101.3.201.1],Code=[5],Length=[20]
*0.635086440 up RDS/8/DEBUG:NULL
```

在路由器上显示的在线用户信息：

```
 [up]dis connection
Index=31  ,Username=luo@aaa
IP=50.1.1.8,
UserState=ONLINE
Total 1 connections matched, 1 listed.
```

（三）LOGIN用户

如果是LOGIN用户上线，NAS上的调试信息如下：

LOGIN用户上线——认证请求（code=1）

```
*0.858402110 up RDS/8/DEBUG:Recv MSG,[MsgType=Normal auth request Index = 120, u
lParam3=645386756]
*0.858402110 up RDS/8/DEBUG:Send attribute list:
*0.858402110 up RDS/8/DEBUG:
[1  User-name                ] [5 ] [luo]
[2  Password                 ] [18] [A88EE282CAD7708A060A286AA331213D]
[4  NAS-IP-Address           ] [6 ] [192.168.100.22]
[32 NAS-Identifier           ] [4 ] [up]
[5  NAS-Port                 ] [6 ] [0]
[61 NAS-Port-Type            ] [6 ] [5]
*0.858402110 up RDS/8/DEBUG:
[6  Service-Type             ] [6 ] [1]     //属性类型：User-Name
[14 Login-Host               ] [6 ] [0.0.0.0]
[31 Caller-ID                ] [16] [303030302D303030302D30303030]
[8  Framed-Address           ] [6 ] [1.1.1.10]
*0.858402110 up RDS/8/DEBUG:Send: IP=[101.3.201.1], UserIndex=[120], ID=[28], Re
tryTimes=[0], Code=[1], Length=[99]
*0.858402110 up RDS/8/DEBUG:Send Raw Packet is:
*0.858402111 up RDS/8/DEBUG:
 01 1c 00 63 00 00 0d 6f 00 00 0c a7 00 00 56 f2
 00 00 56 50 01 05 6c 75 6f 02 12 a8 8e e2 82 ca
 d7 70 8a 06 0a 28 6a a3 31 21 3d 04 06 c0 a8 64
 16 20 04 75 70 05 06 00 00 00 00 3d 06 00 00 00
 05 06 06 00 00 00 01 0e 06 00 00 00 00 1f 10 30
 30 30 30 2d 30 30 30 30 2d 30 30 30 30 08 06 01
 01 01 0a
```

LOGIN用户上线——认证响应（code=2）

```
*0.858402140 up RDS/8/DEBUG:Recv MSG,[MsgType=PKT response Index = 90, ulParam3=
643913796]
*0.858402140 up RDS/8/DEBUG:Receive Raw Packet is:
*0.858402140 up RDS/8/DEBUG:
 02 1c 00 5a 8f 72 27 8f 05 be b2 0f 36 95 ad 51
 a0 b3 6d bc 06 06 00 00 00 01 07 06 00 00 00 01
```

```
09 06 00 00 00 00 0e 06 00 00 00 00 0f 06 00 00
00 00 58 04 35 30 08 06 ff ff ff fe 19 1e 43 49
53 43 4f 41 43 53 3a 30 30 30 34 35 61 33 37 2f
63 30 61 38 36 34 31 36 2f 30

*0.858402140 up RDS/8/DEBUG:Receive:IP=[101.3.201.1],Code=[2],Length=[90]
*0.858402140 up RDS/8/DEBUG:
[6  Service-Type          ] [6 ] [1]
[7  Framed-Protocol       ] [6 ] [1]
[9  Framed-Netmask        ] [6 ] [0.0.0.0]
[14 Login-Host            ] [6 ] [0.0.0.0]
[15 Login-Service         ] [6 ] [0]
[88 Framed-Pool           ] [4 ] [50]
*0.858402140 up RDS/8/DEBUG:
[8  Framed-Address        ] [6 ] [255.255.255.254]
[25 Class                 ] [30] [434953434F4143533A30303034356133372F63306138363431362
F30]
```

LOGIN用户上线——计费开始请求（code=4）

```
*0.858402160 up RDS/8/DEBUG:Recv MSG,[MsgType=Account request Index = 120, ulPar
am3=0]
*0.858402160 up RDS/8/DEBUG:Send attribute list:
*0.858402160 up RDS/8/DEBUG:
[1  User-name             ] [5 ] [luo]
[32 NAS-Identifier        ] [4 ] [up]
[5  NAS-Port              ] [6 ] [0]
[61 NAS-Port-Type         ] [6 ] [5]
[31 Caller-ID             ] [16] [303030302D303030302D30303030]
[40 Acct-Status-Type      ] [6 ] [1]
*0.858402160 up RDS/8/DEBUG:
[45 Acct-Authentic        ] [6 ] [1]
[44 Acct-Session-Id       ] [14] [10609300918d]
[8  Framed-Address        ] [6 ] [1.1.1.10]
[4  NAS-IP-Address        ] [6 ] [192.168.100.22]
[55 Event-Timestamp       ] [6 ] [1162199937]
[25 Class                 ] [30] [434953434F4143533A30303034356133372F63306138363431362
F30]
```

```
*0.858402160 up RDS/8/DEBUG:NULL
*0.858402160 up RDS/8/DEBUG:Send: IP=[101.3.201.1], UserIndex=[120], ID=[19], RetryTimes=[0], Code=[4], Length=[131]
*0.858402160 up RDS/8/DEBUG:Send Raw Packet is:
*0.858402160 up RDS/8/DEBUG:
 04 13 00 83 7e ce 2d bc 21 77 ee b1 31 48 ee fd
 7f d2 79 bb 01 05 6c 75 6f 20 04 75 70 05 06 00
 00 00 00 3d 06 00 00 00 05 1f 10 30 30 30 30 2d
 30 30 30 30 2d 30 30 30 30 28 06 00 00 00 01 2d
 06 00 00 00 01 2c 0e 31 30 36 30 39 33 30 30 39
 31 38 64 08 06 01 01 01 0a 04 06 c0 a8 64 16 37
 06 45 45 c3 81 19 1e 43 49 53 43 4f 41 43 53 3a
 30 30 30 34 35 61 33 37 2f 63 30 61 38 36 34 31
 36 2f 30
```

LOGIN用户上线——计费响应（code=5）

```
*0.858402250 up RDS/8/DEBUG:Recv MSG,[MsgType=PKT response Index = 20, ulParam3=643916484]
*0.858402250 up RDS/8/DEBUG:Receive Raw Packet is:
*0.858402250 up RDS/8/DEBUG:
 05 13 00 14 20 85 7b ae 53 78 c0 3d ff ef de e2 57 65 42 08
*0.858402250 up RDS/8/DEBUG:Receive:IP=[101.3.201.1],Code=[5],Length=[20]
*0.858402250 up RDS/8/DEBUG:NULL
%Oct 30 09:18:57:250 2006 up SHELL/5/LOGIN: luo@aaa login from 1.1.1.10
```

LOGIN用户下线——计费结束请求（code=4）

```
%Oct 30 09:28:57:450 2006 up SHELL/5/LOGOUT: luo@aaa logout from 1.1.1.10
*0.859002470 up RDS/8/DEBUG:Recv MSG,[MsgType=Account off request Index = 120, ulParam3=0]
*0.859002470 up RDS/8/DEBUG:Send attribute list:
*0.859002470 up RDS/8/DEBUG:
[1  User-name                ] [5 ] [luo]
[32 NAS-Identifier           ] [4 ] [up]
[5  NAS-Port                 ] [6 ] [0]
[61 NAS-Port-Type            ] [6 ] [5]
[31 Caller-ID                ] [16] [303030302D303030302D30303030]
[40 Acct-Status-Type         ] [6 ] [2]
*0.859002471 up RDS/8/DEBUG:
```

```
[45 Acct-Authentic          ] [6 ] [1]
[44 Acct-Session-Id         ] [14] [10609300918d]
[8  Framed-Address          ] [6 ] [1.1.1.10]
[4  NAS-IP-Address          ] [6 ] [192.168.100.22]
[55 Event-Timestamp         ] [6 ] [1162200537]
[25 Class                   ] [30] [434953434F4143533A30303034356133372F63306138363431362
F30]
*0.859002471 up RDS/8/DEBUG:
[46 Acct-Session-Time       ] [6 ] [600]
[41 Acct-Delay-Time         ] [6 ] [13]
[42 Acct-Input-Octets       ] [6 ] [0]
[47 Acct-Input-Packets      ] [6 ] [0]
[43 Acct-Output-Octets      ] [6 ] [0]
[48 Acct-Output-Packets     ] [6 ] [0]
*0.859002471 up RDS/8/DEBUG:
[52 Acct_Input_Gigawords    ] [6 ] [0]
[53 Acct_Output_Gigawords   ] [6 ] [0]
[49 Terminate-Cause         ] [6 ] [1]
*0.859002471 up RDS/8/DEBUG:Send: IP=[101.3.201.1], UserIndex=[120], ID=[20],
RetryTimes=[0], Code=[4], Length=[185]
*0.859002471 up RDS/8/DEBUG:Send Raw Packet is:
*0.859002471 up RDS/8/DEBUG:
 04 14 00 b9 ad 5f 8f f3 42 c8 74 d4 cf 99 4d d8
 14 07 47 ac 01 05 6c 75 6f 20 04 75 70 05 06 00
 00 00 00 3d 06 00 00 00 05 1f 10 30 30 30 30 2d
 30 30 30 30 2d 30 30 30 30 28 06 00 00 00 02 2d
 06 00 00 00 01 2c 0e 31 30 36 30 39 33 30 30 39
 31 38 64 08 06 01 01 01 0a 04 06 c0 a8 64 16 37
 06 45 45 c5 d9 19 1e 43 49 53 43 4f 41 43 53 3a
 30 30 30 34 35 61 33 37 2f 63 30 61 38 36 34 31
 36 2f 30 2e 06 00 00 02 58 29 06 00 00 00 0d 2a
 06 00 00 00 00 2f 06 00 00 00 00 2b 06 00 00 00
 00 30 06 00 00 00 00 34 06 00 00 00 00 35 06 00
 00 00 00 31 06 00 00 00 01
```

LOGIN用户下线——计费结束响应（code=5）

```
*0.859002510 up RDS/8/DEBUG:Recv MSG,[MsgType=PKT response Index = 20, ulParam3=
643916164]
*0.859002510 up RDS/8/DEBUG:Receive Raw Packet is:
*0.859002510 up RDS/8/DEBUG:
 05 14 00 14 9a 22 bd 42 3f 3d a4 48 b8 bc f4 85 37 5a 92 91
*0.859002510 up RDS/8/DEBUG:Receive:IP=[101.3.201.1],Code=[5],Length=[20]
*0.859002510 up RDS/8/DEBUG:NULL
```

TACACS+技术

■文/袁亚屏

→TACACS+概述←

什么是TACACS+

TACACS+（Terminal Access Controller Access Control System，终端访问控制器控制系统协议）是在TACACS协议的基础上进行了功能增强的安全协议。该协议与RADIUS协议的功能类似，采用客户端/服务器模式实现NAS与TACACS+服务器之间的通信。

TACACS+的用途

TACACS+协议主要用于PPP和VPDN（Virtual Private Dial-up Network，虚拟私有拨号网络）接入用户及终端用户的AAA。

AAA是Authentication、Authorization、Accounting（认证、授权、计费）的简称，是网络安全的一种管理机制，提供了认证、授权、计费三种安全功能。

认证：确认访问网络的远程用户的身份，判断访问者是否为合法的网络用户。

授权：对不同用户赋予不同的权限，限制用户可以使用的服务。例如用户成功登录服务器后，管理员可以授权用户对服务器中的文件进行访问和打印操作。

计费：记录用户使用网络服务中的所有操作，包括使用的服务类型、起始时间、数据流量等，它不仅是一种计费手段，也对网络安全起到了监视作用。

AAA一般采用客户机/服务器结构，客户端运行于NAS（Network Access Server，网络接入服务器）上，服务器上则集中管理用户信息。NAS对于用户来讲是服务器端，对于服务器来说是客户端。AAA的基本组网结构如下图：

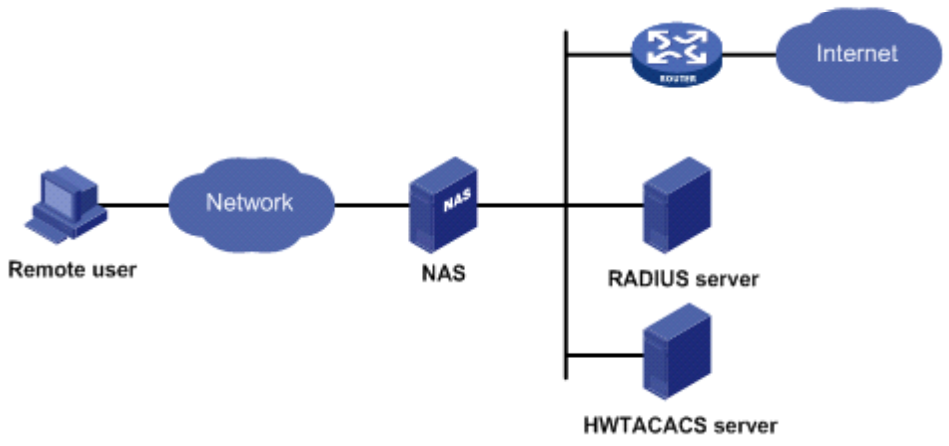


图1 AAA的基本组网结构图

当用户想要通过某网络与NAS建立连接，从而获得访问其它网络的权利或取得某些网络资源的权利时，NAS起到了验证用户的作用。NAS负责把用户的认证、授权、计费信息透传给服务器（RADIUS服务器或HWTACACS服务器），RADIUS协议或HWTACACS协议规定了NAS与服务器之间如何传递用户信息。

图1的AAA基本组网结构中有两台服务器，用户可以根据实际组网需求来决定认证、授权、计费功能分别由使用哪种协议类型的服务器来承担。例如，可以选择HWTACACS服务器实现认证和授权，RADIUS服务器实现计费。

当然，用户也可以只使用AAA提供的一种或两种安全服务。例如，公司仅仅想让员工在访问某些特定资源的时候进行身份认证，那么网络管理员只要配置认证服务器就可以了。但是若希望对员工使用网络的情况进行记录，那么还需要配置计费服务器。

TACACS+的典型应用是对需要登录到设备上进行操作的终端用户进行认证、授权、计费。设备作为TACACS+的客户端，将用户名和密码发给TACACS+服务器进行验证。用户验证通过并得到授权之后可以登录到设备上进行操作。

→TACACS+和RADIUS的比较←

从上面的描述来看，TACACS+和目前被广泛使用的RADIUS很相似，那么他们有什么区别和联系呢？下面这个表格可以说明这个问题：

TACACS+协议	RADIUS协议
使用TCP，网络传输更可靠	使用UDP，网络传输效率更高
除了TACACS+报文头，对报文主体全部进行加密	只对验证报文中的密码字段进行加密
协议报文较为复杂，认证和授权分离，使得认证、授权服务可以分离在不同的安全服务器上实现。例如，可以用一个TACACS+服务器进行认证，另外一个TACACS+服务器进行授权	协议报文比较简单，认证和授权结合，难以分离
支持对设备的配置命令进行授权使用。用户可使用的命令行受到用户级别和AAA授权的双重限制，某一级别的用户输入的每一条命令都需要通过TACACS+服务器授权，如果授权通过，命令就可以被执行	不支持对设备的配置命令进行授权使用 用户登录设备后可以使用的命令行由用户级别决定，用户只能使用缺省级别等于/低于用户级别的命令行

→TACACS+的基本原理←

TACACS+的基本消息交互流程

下图是TACACS+协议的基本信息交互流程：

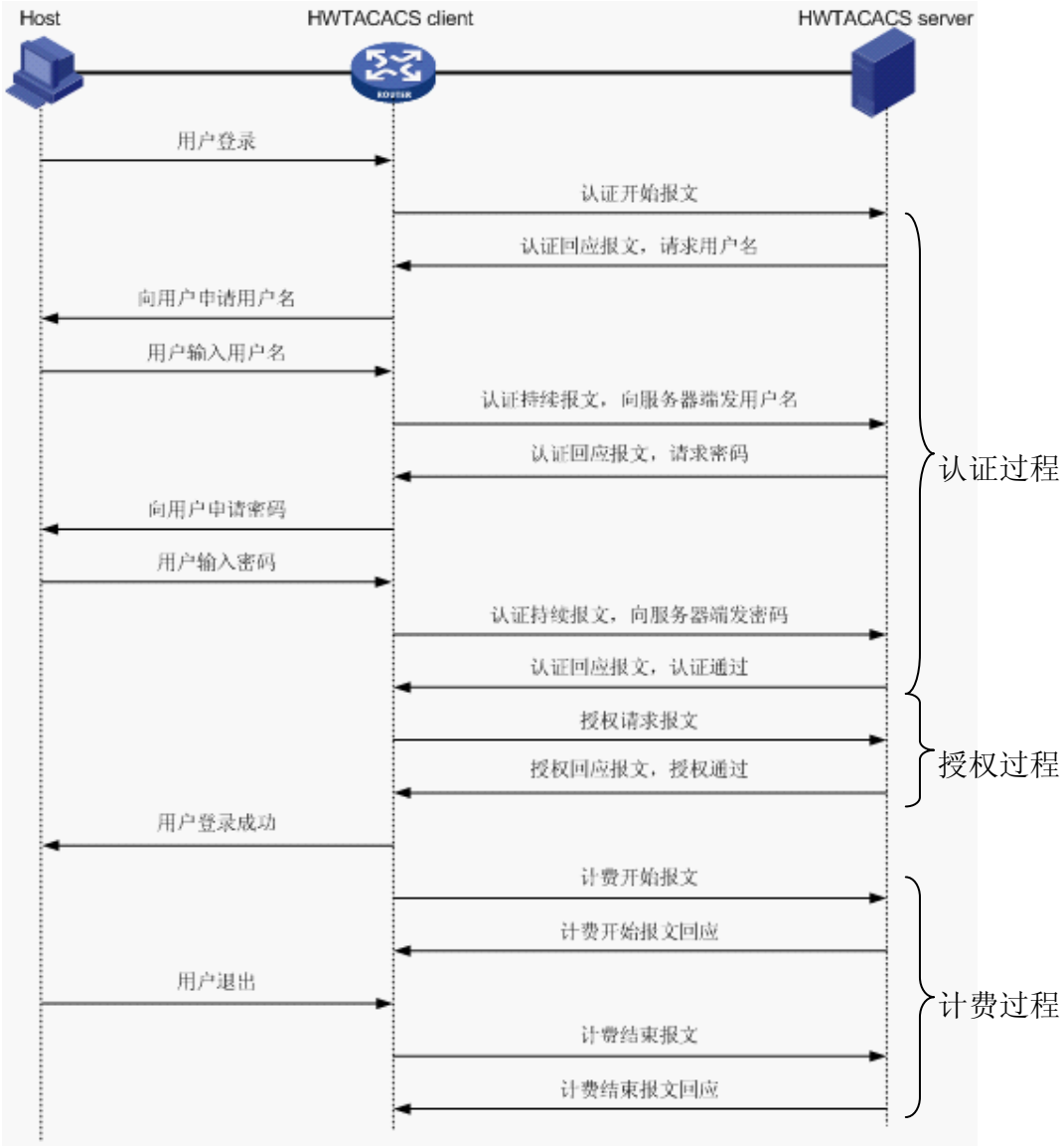


图2 TACACS+协议基本信息交互流程

以Telnet用户认证过程为例，基本消息交互流程如下：

(1) Telnet用户请求登录设备。

(2) TACACS+客户端收到请求之后，向TACACS+服务器发送认证开始报文。

(3) TACACS+服务器发送认证回应报文，请求用户名。

(4) TACACS+客户端收到回应报文后，向用户询问用户名。

(5) 用户输入用户名。

(6) TACACS+客户端收到用户名后，向TACACS+服务器发送认证持续报文，其中包括了用户名。

(7) TACACS+服务器发送认证回应报文，请求登录密码。

(8)TACACS+客户端收到回应报文，向用户询问登录密码。

(9) 用户输入密码。

(10) TACACS+客户端收到登录密码后，向TACACS+服务器发送认证持续报文，其中包括了登录密码。

(11) TACACS+服务器发送认证回应报文，指示用户通过认证。

(12)TACACS+客户端向TACACS+服务器发送授权请求报文。

(13) TACACS+服务器发送授权回应报文，指示用户通过授权。

(14) TACACS+客户端收到授权回应成功报文，向用户输出设备的配置界面。

(15) TACACS+客户端向TACACS+服务器发送计费开始报文。

(16)TACACS+服务器发送计费回应报文，指示计费开始报文已经收到。

(17) 用户请求断开连接。

(18) TACACS+客户端向TACACS+服务器发送计费结束报文。

(19) TACACS+服务器发送计费结束报文，指示计费结束报文已经收到。

→TACACS+的报文结构和具体工作过程←

TACACS+的报文类型

TACACS+共有7种类型的消息：

1、Authentication_START

2、Authentication_CONTIUNE

3、Authentication_REPLY

4、Authorization_REQUEST

5、Authorization_RESPONSE

6、Accounting_REQUEST

7、Accounting_REPLY

TACACS+包头

所有的TACACS+数据包都使用12字节长的包头，结构如下：

```
 1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8

+----------------+----------------+----------------+----------------+
|major   | minor  |                |                |                |
|version| version|      type      |     seq_no     |   flags        |
+----------------+----------------+----------------+----------------+
|                                                                    |
|                             session_id                             |
+----------------+----------------+----------------+----------------+
|                                                                    |
|                               length                               |
+----------------+----------------+----------------+----------------+
```

图3 TACACS+包头

```
⊟ TACACS+
    Major version: TACACS+
    Minor version: 0
    Type: Accounting (3)
    Sequence number: 2
  ⊟ Flags: 0x00 (Encrypted payload, Multiple Connections)
      .... ...0 = Unencrypted: Not set
      .... .0.. = Single Connection: Not set
    Session ID: 74
    Packet length: 5
    Encrypted Reply
```

图4 TACACS+包头（实例）

下面对各个字段分别进行说明：

Major version的取值为0x0C

Minor version一般为0，一些特殊场景取值为1，本文后面会有说明

Type为1表示认证，2表示授权，3表示计费

Seq_no，序列号，从1开始，随报文交互递增

Flags，用来表示一些特殊条件，比如不加密（0x01），支持单连接多会话（0x04）等

Session_id为TACACS+会话的ID，是个随机数。

Length为TACACS+报文除头部之外的长度

TACACS+数据包的加密

TACACS+支持除包头之外所有信息的加密，加密方法如下：

1、将session_id、secret key, 版本号和sequence number一起进行MD5运算（其中secret key 为TACACS客户端和服务器之间的共享秘密），计算结果为MD5_1

2、后续的MD5运算将上次MD5运算的结果也纳入运算范围，如下：

MD5_1 = MD5{session_id, key, version, seq_no}

MD5_2 = MD5{session_id, key, version, seq_no, MD5_1}

....

MD5_n = MD5{session_id, key, version, seq_no, MD5_n−1}

3、将所有的运算结果连接起来，直到总长度大于需要加密的数据的长度，然后截断到实际数据的长度，得到pseudo_pad：

pseudo_pad = {MD5_1 [,MD5_2 [... ,MD5_n]]} truncated to len(data)

4、随后将需要加密的数据和上面的pseudo_pad进行XOR运算，得到密文：

ENCRYPTED {data} == data ^ pseudo_pad

由于TACACS+对整个数据包进行加密，私密性要好于RADIUS，窃听者无法根据报文的内容来猜测网络的配置和用户的身份。

Authentication 消息

Authentication 消息包括三种类型：START、CONTINUE和REPLY。

Authentication Start

认证开始时，客户端发送START消息，START消息中包括认证类型，同时可能包括用户名和一些认证数据。

START消息一般来说是认证过程的第一个数据包，其seq_no总是1。

如果收到RESTART消息，客户端需要发送START消息重新开始认证（使用一个新的会话）。

Server发送REPLY消息来回应START消息，表示认证结束或者仍需要继续。

```
 1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8

+----------------+----------------+----------------+----------------+
|    action      |    priv_lvl    |  authen_type   |     service    |
+----------------+----------------+----------------+----------------+
|    user len    |    port len    |  rem_addr len  |    data len    |
+----------------+----------------+----------------+----------------+
|    user ...
+----------------+----------------+----------------+----------------+
|    port ...
+----------------+----------------+----------------+----------------+
|    rem_addr ...
+----------------+----------------+----------------+----------------+
|    data...
+----------------+----------------+----------------+----------------+
```

图5 Authentication Start消息

Action字段表示具体的认证操作，如认证请求、上传密码等等。

priv_lvl字段表示用户的权限级别。

authen_type字段表示认证的类型，如PAP、CHAP等。

Service字段表示服务类型。

user字段表示用户名，该字段不一定在START消息中存在。

port字段表示客户端上发生认证行为的端口，具体取值客户端可以自行定义，没有明确的要求。

rem_addr字段表示用户的地址信息，属于可选字段，一般使用客户端IP地址、ISDN Caller ID等填充。

data字段用于针对action和authen_type字段的值来传递一些信息。

Authentication Reply

REPLY消息是TACACS+服务器向客户端发送的唯一一种Authentication 消息，用于向客户端反馈当前认证的状态。

```
 1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8

+---------------+---------------+---------------+---------------+
|     status    |     flags     |     server_msg len            |
+---------------+---------------+---------------+---------------+
|          data len             |     server_msg ...
+---------------+---------------+---------------+---------------+
|          data ...
+---------------+---------------+
```

图6 Authentication Reply消息

Status字段表示认证的状态。

flags字段用于控制客户端是否将用户输入的密码回显，如果该标志位置1，用户输入的密码不会回显。

server_msg字段为可选字段，用于服务器将一些附加信息带给用户。

data字段用于向客户端（NAS）提供一些信息。

Authentication Continue

客户端收到REPLY消息后，如果确认认证过程没有结束，使用CONTINUE消息应答。

```
 1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8

+---------------+---------------+---------------+---------------+
|          user_msg len         |            data len           |
+---------------+---------------+---------------+---------------+
|     flags     |  user_msg ...
+---------------+---------------+---------------+---------------+
|    data ...
+---------------+
```

图7 Authentication Continue消息

user_msg字段用于回应REPLY消息中的server_msg字段，向服务器提供客户端或用户的一些信息

flags字段用于中断认证过程

TACACS+认证过程

TACACS+认证的工作过程取决于START消息中的action和authen_type字段的取值。

不同的action和authen_type字段的组合需要配合不同的priv_lvl、service、port和rem_addr字段，实现不同的业务。

TACACS+协议中目前描述了13个action和authen_type字段的组合，这里列出几个比较常见的组合：

Enable Requests

Enable Requests通常用于提升当前用户的级别的场合，比如Linux系统的su命令，Comware的super

命令等，认证过程和后面马上要讲到的Inbound ASCII Login很相似

Action = TAC_PLUS_AUTHEN_LOGIN

priv_lvl = implementation dependent

authen_type = not used

service = TAC_PLUS_AUTHEN_SVC_ENABLE

Inbound ASCII Login

管理用户login的场景，非常常见，该场景下START消息可以携带用户名，整个交互过程包括一个START，中间可能存在一对或多对CONTINUE/REPLY，最后以一个REPLY结束

action = TAC_PLUS_AUTHEN_LOGIN

authen_type = TAC_PLUS_AUTHEN_TYPE_ASCII

Inbound CHAP login

CHAP认证，最常见的是PPP认证的场景，整个交互过程包括一个START和一个REPLY消息，START消息中必须包含用户名。

Action = TAC_PLUS_AUTHEN_LOGIN

authen_type = TAC_PLUS_AUTHEN_TYPE_CHAP

minor_version = 0x1

具体认证过程

如下是Comwarev5平台Telnet用户使用HWTACACS认证过程：

AUTHEN_REQUEST ------------->

<------------- AUTHEN_REPLY（status:AUTHEN_STATUS_GETPASS）

AUTHEN_CONTINUE（with password） ------------->

<------------- AUTHEN_REPLY（status:AUTHEN_STATUS_PASS）

认证过程的中止

客户端可以通过在CONTINUE消息中携带一个TAC_PLUS_CONTINUE_FLAG_ABORT标志位来中止正在进行的认证过程，同时可以在data字段携带中止认证的原因。该CONTINUE消息不需要REPLY消息的回应。

Authorization消息

Authorization 消息包括两种类型：REQUEST和RESPONSE。

Authorization Request

Authorization Request消息的机构比较复杂，如下：

```
 1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8

+----------------+----------------+----------------+----------------+
|  authen_method |    priv_lvl    |  authen_type   | authen_service |
+----------------+----------------+----------------+----------------+
|    user len    |    port len    |  rem_addr len  |    arg_cnt     |
+----------------+----------------+----------------+----------------+
|   arg 1 len    |   arg 2 len    |      ...       |   arg N len    |
+----------------+----------------+----------------+----------------+
|   user ...
+----------------+----------------+----------------+----------------+
|   port ...
+----------------+----------------+----------------+----------------+
|   rem_addr ...
+----------------+----------------+----------------+----------------+
|   arg 1 ...
+----------------+----------------+----------------+----------------+
|   arg 2 ...
+----------------+----------------+----------------+----------------+
|   ...
+----------------+----------------+----------------+----------------+
|   arg N ...
+----------------+----------------+----------------+----------------+
```

图8 Authorization Request消息

Authorization Request消息中包括了授权所需的一切信息，这些信息分为两类，一类是必选的，另一类是可选的。

其中，authen_method字段表示授权的方式，TACACS+的认证和授权是分离的，用户可以使用TACACS+认证而使用其它协议进行授权。

priv_lvl字段、authen_type字段、authen_service字段、port字段、rem_addr字段的含义和authentication消息中的相应字段一样。

user字段表示用户名。

arg_cnt字段表示REQUEST消息中携带的argument的数量， argument是一个AVP的结构，其中attribute和value之间使用等号（=）或星号（*）连接，当使用等号连接时表示该AVP是必选的，使用星号连接时表示该AVP是可选的。

AVP的类型很多，这里就不一一列举了。

Authorization Response

Authorization Response消息的结构如下：

```
 1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8

+----------------+----------------+----------------+----------------+
|    status      |     arg_cnt    |         server_msg len          |
+----------------+----------------+----------------+----------------+
+            data len             |    arg 1 len   |    arg 2 len   |
+----------------+----------------+----------------+----------------+
|      ...       |   arg N len    |         server_msg ...
+----------------+----------------+----------------+----------------+
|   data ...
+----------------+----------------+----------------+----------------+
|   arg 1 ...
+----------------+----------------+----------------+----------------+
|   arg 2 ...
+----------------+----------------+----------------+----------------+
|   ...
+----------------+----------------+----------------+----------------+
|   arg N ...
+----------------+----------------+----------------+----------------+
```

图9 Authorization Response消息

Response消息中包括了授权的结果和一些其它的参数

status字段表示授权的结果和权限的操作方式

server_msg字段是服务器给用户的提示信息。

data字段是服务器提供给管理员的信息。

arg_cnt字段表示RESPONSE消息中携带的argument的数量，argument的格式和REQUEST消息中相同。

具体授权过程

Comwarev5设备使用HWTACACS认证/授权

AUTHOR_REQUEST ------------->

<------------- AUTHOR_REPLY (status:AUTHOR_STATUS_PASS_ADD)

Accounting消息

Accounting消息包括两种类型：REQUEST和REPLY。

Accounting Request

Accounting Request消息的结构如下：

```
 1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8

+----------------+----------------+----------------+----------------+
|      flags     | authen_method  |    priv_lvl    |  authen_type   |
+----------------+----------------+----------------+----------------+
| authen_service |    user len    |    port len    |  rem_addr len  |
+----------------+----------------+----------------+----------------+
|    arg_cnt     |   arg 1 len    |   arg 2 len    |      ...       |
+----------------+----------------+----------------+----------------+
|   arg N len    |    user ...
+----------------+----------------+----------------+----------------+
|   port ...
+----------------+----------------+----------------+----------------+
|   rem_addr ...
+----------------+----------------+----------------+----------------+
|   arg 1 ...
+----------------+----------------+----------------+----------------+
|   arg 2 ...
+----------------+----------------+----------------+----------------+
|   ...
+----------------+----------------+----------------+----------------+
|   arg N ...
+----------------+----------------+----------------+----------------+
```

图10 Accounting Request消息

Accounting Request消息中包括了计费所需的信息。

flags字段表示计费报文的类型，包括计费开始。计费停止和实时计费。

其它字段的含义和与authorization和authentication中对应字段的含义一致，这里不再赘述。

Accounting REQUEST消息中也可以携带很多AVP形式的参数，参数的数量很多，这里就不一一列举了。

data字段是服务器提供给管理员的信息。

Accounting Reply

Accounting Reply消息的结构如下：

```
 1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8  1 2 3 4 5 6 7 8

+----------------+----------------+----------------+----------------+
|         server_msg len          |             data len            |
+----------------+----------------+----------------+----------------+
|     status     |         server_msg ...
+----------------+----------------+----------------+----------------+
|     data ...
+----------------+
```

图11 Accounting Reply消息

其中status字段表示计费的状态，标志计费是否成功。

server_msg字段表示服务器发给用户的信息，由客户端来决定是否显示给用户。

具体计费过程

Comwarev5设备使用HWTACACS进行PPP用户计费

下面是Comwarev5设备使用HWTACACS进行PPP用户计费的过程，包括计费开始，计费持续和计

费停止的全过程。

ACCOUNTING_REQUEST（flag:ACCT_FLAG_START ------------->

<------------- ACCOUNTING_REPLY（status:ACCT_STATUS_SUCCESS）

ACCOUNTING_REQUEST（flag:ACCT_FLAG_WATCHDOG） ------------->

<------------- ACCOUNTING_REPLY（status:ACCT_STATUS_SUCCESS）

ACCOUNTING_REQUEST（flag:ACCT_FLAG_STOP） ------------->

<------------- ACCOUNTING_REPLY（status:ACCT_STATUS_SUCCESS）

TACACS+的附加功能

除了正常的AAA功能外，为确保功能实现的冗余和实时计费准确性等等，很多TACACS+实现都包括下面的内容：

1、计费停止报文的缓存和重传机制；

2、认证、授权、计费服务器的主备切换功能。

TACACS in Comware（以V5为例）

Comware v5的HWTACACS功能基本上实现了TACACS+所规定的功能：

```
#
 hwtacacs scheme tacacs
 primary authentication 101.3.201.1
 primary authorization 101.3.201.1
 primary accounting 101.3.201.1
 key authentication h3c
 key authorization h3c
 key accounting h3c
 timer realtime-accounting 3
 user-name-format without-domain
#
```

LDAP概述及应用

■文/李 丹

→LDAP概述←

LDAP简介

LDAP的英文全称是Lightweight Directory Access Protocol，简称为LDAP。它是基于X.500标准的，与X.500不同的是，LDAP支持TCP/IP，这对访问Internet是必须的。LDAP的核心规范在RFC中都有定义， LDAPv3由以下RFC定义：

RFC2251——Lightweight Directory Access Protocol (v3)

RFC2252——Lightweight Directory Access Protocol (v3):Attribute Syntax Definitions

RFC2253——Lightweight Directory Access Protocol (v3):UTF-8 String Representation of Distinguished Names

RFC2254——The String Representation of LDAP Search Filters

RFC2255——The LDAP URL Format

RFC2256——A Summary of the X.500(96) User Schema for use with LDAPv3

RFC2829——Authentication Methods for LDAP

RFC2830——Lightweight Directory Access Protocol (v3): Extension for Transport Layer Security

RFC3377——Lightweight Directory Access Protocol (v3):Technical Specification

LDAP是目录服务（DAP）在TCP/IP上的实现。它是对X500的目录协议的移植，但是简化了实现方法，所以称为轻量级的目录服务。

LDAP最大的优势是：可以在任何计算机平台上，用很容易获得的而且数目不断增加的LDAP的客户端程序访问LDAP目录。而且也很容易定制应用程序为它加上LDAP的支持。

LDAP是一个存储静态相关信息的服务，适合“一次记录多次读取”。LDAP对查询进行了优化，与写性能相比LDAP的读性能要优秀很多。

在LDAP中，目录是按照树型结构组织的，目录由条目（Entry）组成，条目由属性集合组成，每个属性说明对象的一个特征。每个属性有一个类型和一个或多个值。属性类型说明包含在此属性中的信息的类型，而值包含实际的数据。条目相当于关系数据库中表的记录；条目是具有区别名DN（Distinguished Name）的属性（Attribute）集合，DN相当于关系数据库表中的关键字（Primary Key）；属性由类型（Type）和多个值（Values）组成，相当于关系数据库中的域（Field）由域名和数据类型组成，只是为了方便检索的需要，LDAP中的Type可以有多个Value，而不是关系数据库中为降低数据的冗余性要求实现的各个域必须是不相关的。LDAP中条目的组织一般按照地理位置和组织关系进行组织，非常的直观。LDAP把数据存放在文件中，为提高效率使用基于索引的文件数据库，而不是关系数据库。

组织数据

LDAP采用目录的方式来组织数据。目录的典型应用是UNIX，在UNIX的文件系统中，最顶层是根目录（root），在根目录的下面有很多的文件和目录。

但是LDAP目录和UNIX文件系统之间有三点不同：

第一是LDAP模型没有真正的根条目。文件系统有一个根目录，它是所有文件和目录的祖先。在LDAP目录中，根条目是概念性的，不作为一个可以存放数据的条目存在。

第二是目录的每个节点都可以包含数据，任何节点都可能作为一个容器，即LDAP条目允许在他的下面有子节点。文件系统的每个节点或者是文件或者是目录，但不能同时是二者。在文件系统中，只有目录可以有子节点，而且只有文件可以包含数据。LDAP中的条目既可以是数据同时也可以兼任目录的角色。

第三是节点命名方式的不同。LDAP名称与文件系统名称的方向是相反的。文件系统的表示方法是从根目录到深层目录一层一层表示，而LDAP则相反，将叶子节点放在前面逐层目录向上推导，最后写出根目录，下一小节详细介绍。

表示数据

LDAP中记录的表示方法，首先确认以下几个概念：

dn ：Distinguished Name；唯一可区别的名称，记录了一条记录的位置

dc ：一条记录所属区域

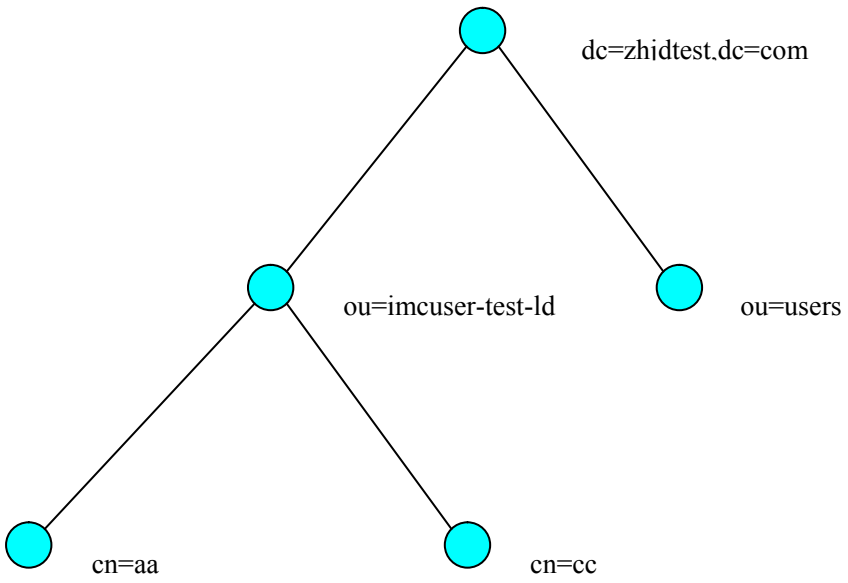
ou ：Organization Unit；一条记录所属组织

cn/uid：Common Name；可用来区分身份的属性，等同于一条记录的名字/ID

一个完整的dn如下所示，以此查询得到一条完整的记录，然后再根据需要取出其中的指定的属性(Attribute)的值。

dn：cn=cc,ou=imcuser–test–ld,dc=zhjdtest,dc=com

如下所示的结构中，dn是从树的叶子节点逐层向根节点推导出的一条唯一路径，表明了这个数据结构中的唯一一条记录。



按照以上dn在LDAP Server上查询到的数据如下：

attribute type	value
cn	**cc**
instanceType	4
nTSecurityDescriptor	
objectCategory	CN=Person,CN=Schema,CN=Configuration,DC=zhjdtest,DC=com
objectClass	top
objectClass	person
objectClass	organizationalPerson
objectClass	user
accountExpires	9223372036854775807
badPasswordTime	0
badPwdCount	0
codePage	0
countryCode	0
displayName	cc
distinguishedName	CN=cc,OU=imcuser-test-ld,DC=zhjdtest,DC=com
lastLogoff	0
lastLogon	0
logonCount	0
name	cc
objectGUID	(non string data)
objectSid	(non string data)
primaryGroupID	513
pwdLastSet	129297896275781250
sAMAccountName	cc
sAMAccountType	805306368

操作数据

查询类操作

查询类操作允许用户搜索目录并取回目录数据，有两个查询操作：查询和比较。

LDAP查询操作用来在目录中搜索条目，并取出单个目录条目。LDAP没有读操作，当需要读取某条目时，必须使用一种特殊格式的查询操作，其中限定了你要取回的条目内容。根据取回的条目内容，查询操作分为三种：一种是单个节点查询，是指查询指定了路径的条目的具体节点，这通常用来在目录中查找一个具体的条目；二是子节点查询，是指查询给出条件的节点的下级的条目；三是子树查询，是指查询从给出的条目为根的整个子树的所有条目。

另一类查询操作是比较操作，它用于检查某条目是否包含某个属性值。如果条目有此值，则比较结果为真；如果没有，则比较结果为假。

更新类操作

LDAP更新类操作包括四种操作：添加、删除、修改和重命名，这四种操作定义了在目录中操作数据的方式。

添加操作创建新的目录条目，它必须携带的两个参数为：要创建的条目的分辨名DN和新条目中包含的属性/属性值对的集合。为了使添加操作成功，必须满足以下四个前提条件：

◆ 新条目的父条目必须已经存在

◆ 不能存在同名（分辨名）的条目

◆ 新条目必须与有效的模式(schema)相一致

◆ 访问控制必须允许执行此操作

删除操作只需指明要删除的条目DN。只能删除目录的叶节点，即不支持删除子树。删除一经执行，无法恢复。为了使删除操作成功，必须满足以下两个前提条件：

◆ 要删除的条目的父条目必须存在

◆ 访问控制必须允许执行此操作

修改操作除指明操作的条目DN外，还包括一个数组。其中每个数组元素是一个修改动作，记录修改的操作类型和操作数据。修改操作功能强大，能完成对条目的属性类型和属性值的修改操作。

重命名，用于为条目重命名。它必须指定要重命名的条目、条目新的RDN和删除原RDN标志(delete-old-RDN)。为了使修改RDN操作成功，必须满足如下前提条件：

◆ 被重命名的条目必须已经存在

◆ 条目的新名称不能已经被其他条目使用

◆ 访问控制必须允许执行此操作

认证和控制类操作

认证和控制类操作包括两个LDAP认证操作——绑定和解绑定，一个控制操作——放弃。

绑定操作：一般常用的是使用简单密码的绑定。使用SSL/TLS绑定更加复杂，一般需要预先配置和多步操作，目前RFC没有对此作出规定。

解绑定操作：用于中断持续进行的LDAP操作，关闭连接。

放弃操作用于中断正在进行的操作。

Schema

目录的模式(schema)是一组规则,用来确定目录能存储什么和服务器和客户端在进行目录操作时如何对待数据。schema被LDAP各方(服务器、客户端、应用程序)用来作为数据交换的“标准”，以保证对数据理解的正确性,从而保证了LDAP协议的开放性。

schema也被用作目录中存储的数据的长度、范围和格式的强制约束。

最后,schema也有利于目录中数据的规整和对访问者权限的控制。

构成schema的元素有属性类型(attribute types)、对象类(object classes)、语法(syntaxes)、匹配规则(matching rules)、目录信息树内容规则(DIT content rules)、目录信息树结构规则(DIT strctural rules)和命名形式(name forms)。

属性类型(attribute types)

属性类型控制属性格式,包括属性的语法、匹配规则、是否可以多值、修改权限和用法等。属性类型可直接由0或多个属性类型继承而来,形成属性类型的层次关系树。

对象类(object classes)

对象类是“共享某些特性的对象的识别家族”,即对象的模板。对象类通过定义条目中所含的属性来

定义目录中的条目类型。

在LDAP目录数据库中，所有的条目都必须定义objectClass这个属性。每个条目（LDAP Entry）都要定义自己的Object Classes。Object Class可以看作是LDAP Entry的模板，它定义了条目的属性集，包括必有属性（requited attribute）和可选属性（option attribute）。这里要着重指出的是，在LDAP的Entry中是不能像关系数据库的表那样随意添加属性字段的，一个Entry的属性是由它所继承的所有Object Classes的属性集合决定的，此外可以包括LDAP中规定的“操作属性”（操作属性是一种独立于Object Class而存在的属性，它可以赋给目录中的任意条目）。如果你想添加的属性不在Object Classes定义属性的范畴，也不是LDAP规定的操作属性，那么是不能直接绑定（在LDAP中，给Entry赋予属性的过程称为绑定）到条目上的，你必须自定义一个含有你需要的属性的Object Class，而后将此类型赋给条目。

语法(syntaxes)

语法(syntaxes) 规定了如何用具体的语言和格式来操作目录中的条目数据。

匹配规则(matching rules)

为服务器在搜索操作过程中如何比较字符串提供准则。在国际搜索中，匹配规则告知服务器所用的对照顺序及运算符。

用途：

◆ server执行查询或比较操作时比较属性值

◆ server修改条目时确定要添加或删除的属性值

◆ server在比较DN和条目名称时使用

其他几个元素在LDAPv3中没有提到，在这里就不做介绍了。

安全特性

LDAP的认证分为：无认证、基本认证、SSL/TLS三种，简单介绍一下。

无认证(匿名) 是最简单的一种方法，这种方法只在没有数据安全问题并且不涉及访问控制权限的时候才能使用。例如，当你的目录是任何人都可以浏览的地址簿时，就是这种情况。当你调用API的绑定操作时分辨名(DN)和密码保留为空，目录会假定是无认证，LDAP服务器会自动假定一个匿名用户会话并且服务相应的访问控制权限。

基本认证(简单密码) 当LDAP客户进程和服务进程之间建立连接时，会协商安全机制，这是在LDAP应用程序接口(API)中指定的方法。除了根本不使用认证之外，最简单的LDAP安全机制是基本认证。

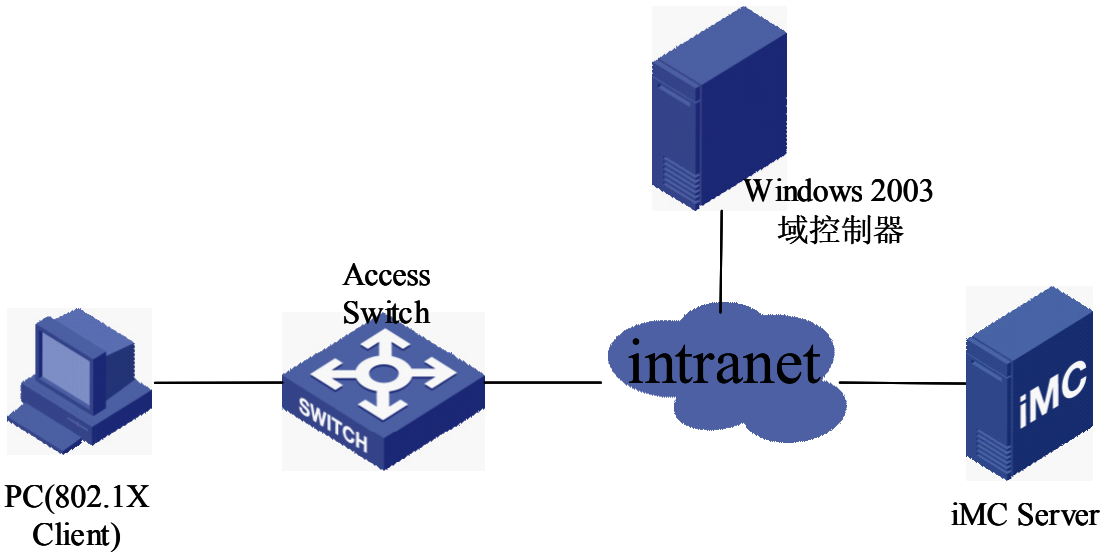
当使用LDAP的基本安全认证时，客户进程通过网络向服务进程发送一个分辨名(DN)和口令来标识自己。服务进程检查客户进程发送的分辨名(DN)和密码是否与目录中存储的分辨名(DN)和密码相匹配，如果匹配则认为通过了认证。

SSL/TLS 安全套接层协议的意图是提供认证和数据的安全性，它封装了TCP/IP，以便于每个TCP/IP应用都能够使用它保证通讯的安全。

→LDAP应用←

下面介绍一个H3C对LDAP的典型应用，其中涉及客户端、接入设备、业务软件服务器以及Windows 2003域控制器。

组网图



配置介绍

PC（802.1x Client）

PC作为接入客户端，只要有802.1x客户端即可，一般在网络连接上都自带802.1x客户端；在我司的EAD解决方案中，使用iNode客户端来进行802.1x的接入，推荐使用iNode客户端；也可以使用自己安装的其他802.1x客户端。

接入交换机（Access Switch）

该交换机只需要支持802.1x接入功能即可，交换机管理口地址需要和iMC Server能够ping通；在PC接入的接口下开启802.1x认证功能，然后设置domain和RADIUS scheme，具体配置如下：

RADIUS scheme配置，验证及计费服务器的IP地址根据实际使用地址填写：

```
RADIUS scheme ld_20090811
 server-type huawei
 primary authentication 162.105.255.30 1812
 primary accounting 162.105.255.30 1813
 key authentication test
 key accounting test
 user-name-format with-domain
```

domain配置，RADIUS-scheme和之前的RADIUS scheme名称要一致：

```
domain zhjdtest
 RADIUS-scheme ld_20090811
 access-limit enable 1024
 state active
 vlan-assignment-mode integer
 idle-cut disable
 self-service-url disable
 messenger time disable
```

接口配置，vlan可根据实际需要划分：

```
interface Ethernet0/24
 port access vlan 24
 dot1x
```

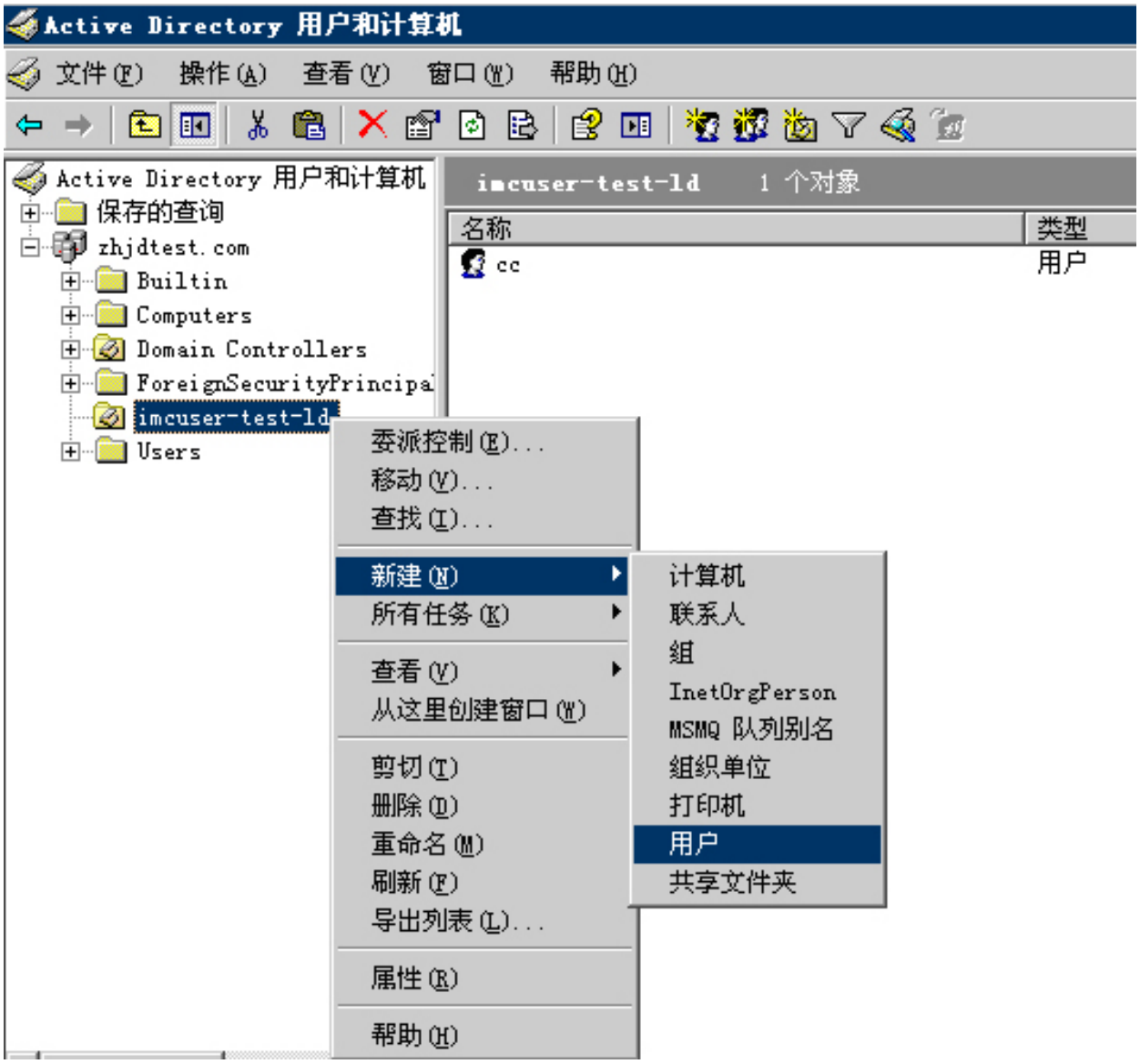
Windows 2003域控制器

Windows 2003域控制器主要使用的是微软的活动目录，将其作为LDAP服务器，让iMC从中提取账户信息。

关于Windows 2003安装成域控制器的方法在这里就不介绍了，可以通过查看相关文档进行安装，简单说需要先安装DNS服务，再安装Active Directory；这里介绍一下账户的添加方法：

首先，在域控制器下新建一个组织单位：如下，名为：imcuser-test-ld

其次，在刚才创建的组织单位下创建用户，该操作可重复多次，创建多个用户：如下用户名为：cc



iMC Server

首先，iMC Server上要将接入交换机作为接入设备加入；

其次，需要添加一个和接入交换机上的domain相对应的服务，即服务后缀等于域名；

以上两条配置不涉及到LDAP部分，不详细介绍，下面介绍LDAP部分的配置：

1、LDAP服务器配置：

LDAP服务器信息

基本信息

* 服务器名称 zhjdtest.com　　* 服务器版本 3

* 服务器IP地址 162.105.255.30　　* 端口 389

* 服务器类型 微软活动目录　　服务同步策略 手工指定

实时认证 是　　* 连接服务器静默时长 1分钟

* 连接超时时间 30 秒　　* 同步超时时间 0 秒

☑ 连通服务器

服务器信息

* Base DN ou=imcuser-test-ld,dc=zhjdtest,dc=com

* 管理员DN cn=administrator,cn=users,dc=zhjdtest,dc=com

管理员密码 ••••••

* 用户名属性名称 sAMAccountName

用户密码属性名称

备份服务器信息

备份服务器IP　　使用中的服务器 ◉ 主服务器 ○ 备份服务器

备到主自动切换 ☐　　切换时间间隔 24 小时

确定　检测　取消

其中需要注意的是：Base DN的写法，ou后面是组织单位的名称，dc后面是域名，每一级域名分别依次使用dc表示；管理员DN的写法，cn后面是windows 2003的管理员用户名，后续的写法固定，其中dc部分和Base DN相同；用户名属性名称，如果使用Windows活动目录的话，用户名属性基本是固定的，密码不从活动目录读取。如果使用其他LDAP服务器的话，需要注意用户名属性和密码属性需要填写具体的Attribute名称，其他基本一致。

同步策略配置：

* 同步策略名称 tongbu

服务器名称 zhjdtest.com

* 过滤条件 (&(objectclass=user)(sAMAccountName=*))

配置状态 有效

同步选项 ☑ 自动同步

☑ 新增用户及其接入帐号

☑ 为已存在用户新增接入帐号

下一步　取消

上图中过滤条件基本不需要修改即可使用，如果接入帐号已经存在并且不是从LDAP同步过去的，那么将会出现同步失败；

注意这里一定要勾选需要的服务，如果使用微软活动目录的话，其他属性基本不用修改，密码需要自己输入；如果使用其他LDAP Server的话，属性要自己重新选择。

典型应用

使用PC上的802.1x客户端登录时，接入交换机到iMC Server上做认证，iMC Server从微软活动目录上同步用户信息，对客户端进行认证；认证通过后客户端可正常使用网络，否则客户端无法接入。

数字证书与PKI

■文/邓宏波

什么是PKI？下面的具有高度概括性的解释相当的准确和全面。

“PKI（Public Key Infrastructure，公钥基础设施）是一种运用公钥的概念与技术来实施并提供安全服务的具有普遍适用性的网络安全基础设施。 它由公私钥对、数字证书、CA 、RA、 CRL库、证书库、LDAP目录、参与验证的设备和支持PKI的应用程序组成”

为了能对PKI和数字证书有一个清晰的理解，我们将从网络安全的需求、密码学、数字证书再到PKI及其组件一步步地深入下去。

网络安全的需求

全球互联网用户快速增长,1996年不足0.4亿，到2000年6月已经达到2.6亿以上，2005年底互联网用户达到10.8亿，并且仍在不断增长，国内用户达到1.8亿 。

电子商务有着巨大的市场与无限的商业机遇，孕含着现实的和潜在的丰厚商业利润。2007年中国电子商务的交易额超过600亿人民币；2010年全球的电子商务交易额可达1万亿美元，未来10年1/3的全球国际贸易将以网络贸易的形式来完成。

而且基于网络、互联网的应用逐步在电子商务、电子政务以及各个领域应用成熟，新的业务模式层出不穷。

电子商务为我们带来便利、商机的同时，信息安全隐患也日益加剧，比如，用户帐号、密码被大量窃取/误用；私有或机密资料被泄露或被篡改；网站遭到非法攻击导致重大损失甚至被迫停产；个人被假冒身份而造成损害；由于证据有限而增加的纠纷及解决成本；由于感受到安全方面的弱点，造成用户裹足不前；钓鱼网站层出不穷，用户安全无法保证；企业内部网络安全管理缺乏有效手段。这真是噩梦，也同时制约了电子商务的发展。

构建一个安全的电子商务交易环境，实现网络安全就成为了必须。

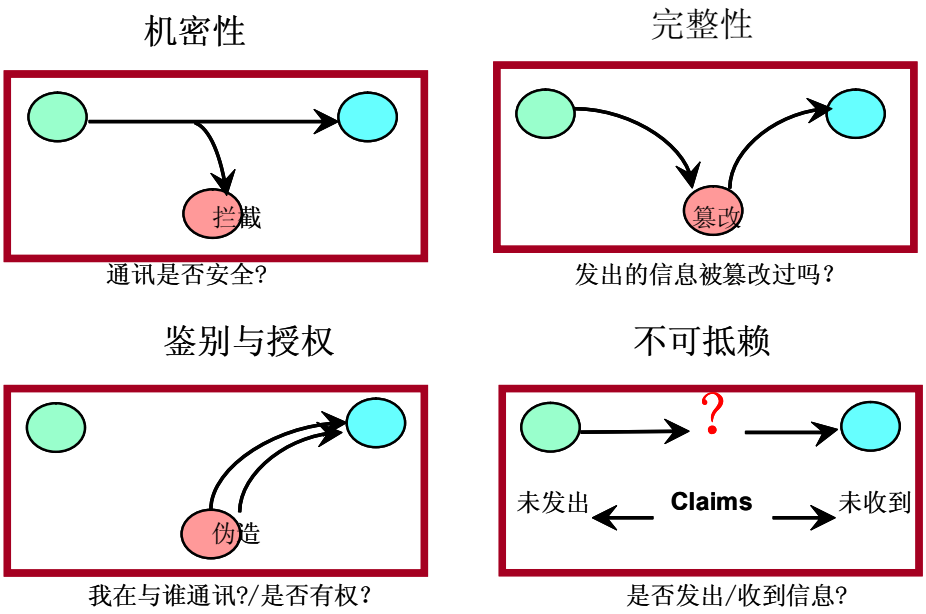
但如何定义网络安全问题的问题域？问题现象上面已经描述了，那么我们的问题本质是什么呢？即：

- 我们必须确保数据能够保持私有或保存为一个秘密的格式。我们称之为“机密性”。
- 我们需要一种授权方法，使需要它的人可以访问那些私有的或秘密的数据。这叫“访问控制”。
- “完整性”保证是另一个非常重要的要求，特别是当你在处理电子交易的时候。
- “真实性”。
- “不可抵赖性”。

把机密性、访问控制、完整性、真实性和不可抵赖性结合起来，能够组成一个很高程度的网络安全。例如，假设你在网上银行发出一个要求说要将￥10,000在两个账户之间转移。银行必须能够确信他们所收到的正是你所发出的。这一点非常重要。

现在，我们问题域的核心出来了，就是：

- Privacy（保密性）：确认信息的保密，不被窃取
- Authentication & Authorization（鉴别与授权）：确认对方的身份并确保其不越权
- Integrity（完整性）:确保你收到信息没有被篡改
- Non–Repudiation（抗抵赖）：有证据保证交易或者其他操作不被否认



在现实世界里，应对上述安全问题的措施，我们耳熟能详，比如，证明身份的身份证、你我的工卡、签名、我们的安全封条、安全岗等等，在数字世界里，也有达到相同目的的措施。如下表所示：

信任类型	现实世界	数字世界
身份认证(鉴别与授权)	身份证、护照、信用卡、驾照	数字证书、数字签名
完 整 性	签名、支票、第三方证明	数字签名
保 密 性	保险箱、信封、警卫、密藏	加密
不可否认性	签名、挂号信、公证、邮戳	数字签名

就如上表中所言，数字世界的安全措施包含但不限于上述的方案，实际上，数字世界里传统的安全解决方案有：账号/口令、虚拟键盘、动态口令、验证码、键盘保护等。

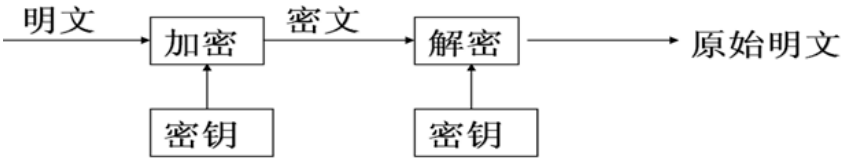
但一个良好的安全解决方案必须同时兼顾：身份认证、信息传输加密、信息完整校验、操作抗抵赖。这就是为何我们需要了解：数字证书、基于PKI技术实现的安全解决方案

在进入“数字证书”内容的学习之前，我们非常有必要了解以下密码学的一些知识，这是后续内容的基础。

密码学的必要知识

我们先说明以下关于密码技术的基本概念：

- 明文： 需要被隐蔽的消息
- 密文： 明文经变换形成的隐蔽形式
- 加密：把明文信息转化为密文的过程
- 解密：把密文信息还原成明文的过程



上图就是加密技术的基本原理。

尽管加密技术只是在二战期间开始流行，并且近期才用于网络安全，但实际上它的内容已经存在了好几千年了。现代的加密技术只是对这个过程添加了一个高得多的数学逻辑而已。

加密领域源于Julius Caesar 和罗马帝国。Caesar用一简单的加密方法（如下图），来跟他的将军们联系。Caesar的方法只是简单地将正文的每个字母移动一个固定的偏移量（在这个例子中是12）。例如，“A”变成“M”，“B”变成“N”，如此等等。

A=M	N= Z
B=N	O=A
C=O	P=B
D=P	Q=C
E=Q	R=D
F=R	S=E
G=S	T=F
H=T	U=G
I=U	V=H
J=V	W=I
K=W	X=J
L=X	Y=K
M=Y	Z=L

输入 = JITCA

算法 = 字母右移

密钥 = 12

输出 = VUFOM

考察Caesar的加密方法，我们会发现两个内容：加密算法和密钥。加密算法是用来改变原始数据的方法。在这个例子中“字母移位”就是加密算法。密钥是使加密过程得出一个唯一结果的变量。这个例子中密钥是12。用一个不同的密钥，将得到一个不同的结果。

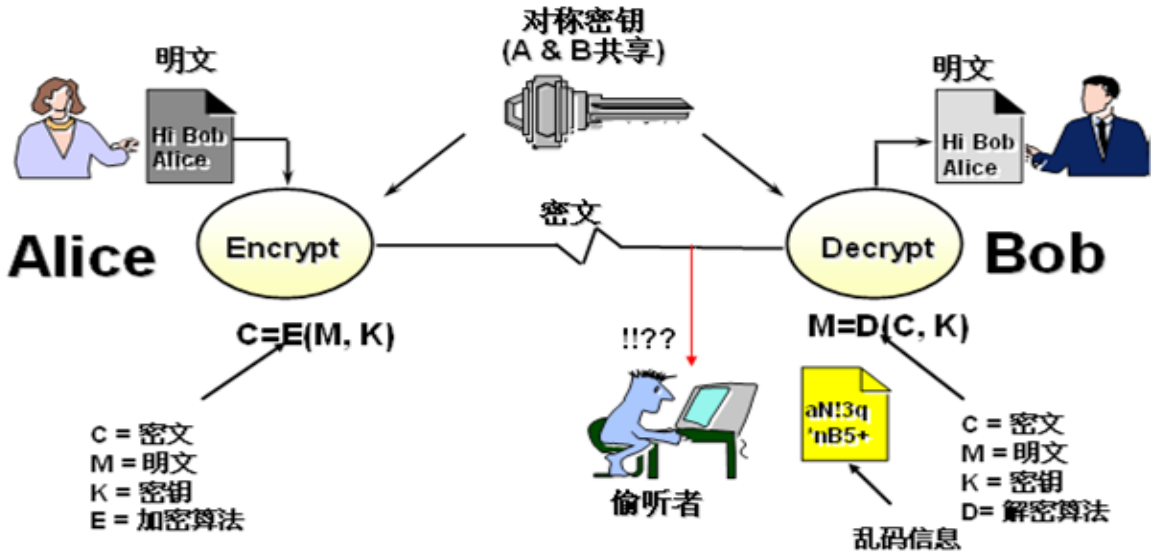
在我们的简单的加密例子中，你可能可以看到，当Caesar为他的将军加密了一个消息后，这个将军将用同样的密钥来解密这则消息。

现代的的密码技术当然要复杂得多了。接下来，我们将对密码技术以图示的方法逐一做简单介绍。

分为：对称算法（共享密钥）、非对称算法（公/私密钥对）、组合密码技术、摘要算法、数字签名技术及其这些算法的结合使用。

对称加密算法

在两个通讯者之间需要一把共享的密钥

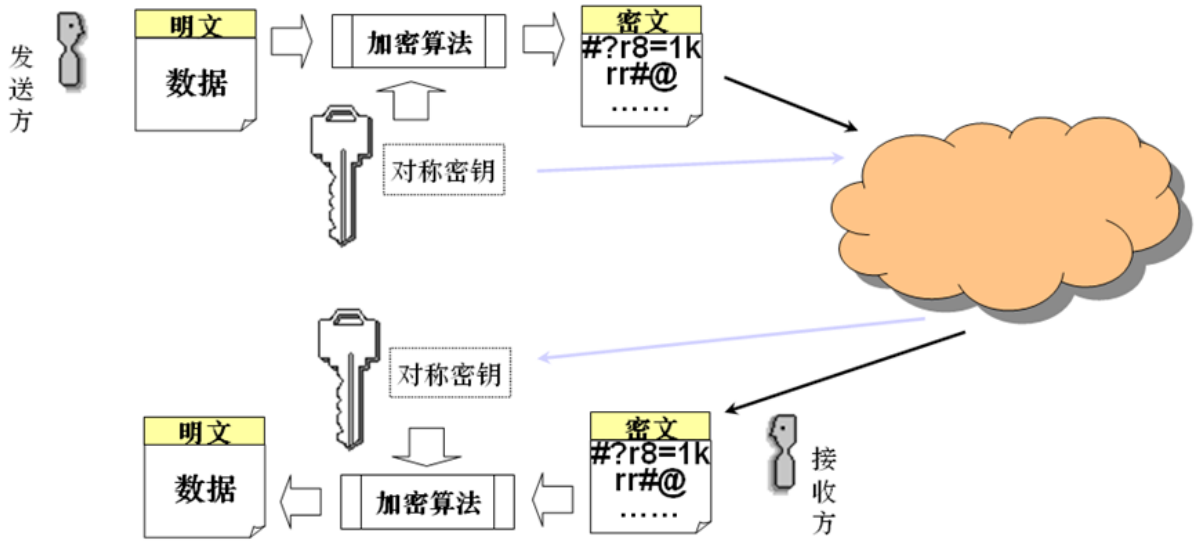


对称加密算法的优点是速度快、性能好。

但是对称加密算法的缺点主要是：每对通信者都需要一对不同的密钥，N个人通信就需要N！的密钥，这导致密钥难以管理，不适用于大用户量的应用；并且，密钥的分发也是大问题，不可能和与你不曾谋面的人通信。

基于对称加密算法的特性，其适用于需要快速的加密/解密的应用场景，比如大数据量的加密传输（数据传输模型）。如下例所示：

1. 选择一种加密算法，例如3-DES
2. 得到一个随机数（密钥）
3. 对需要发送的数据加密
4. 将加密后的结果发送给接受者
5. 密钥交付接受者
6. 接受者通过密钥解密



常见的对称加密算法有：DES、3-DES、 SSF33、 IDEA、AES、RC2、RC4

算法	密钥	开发者	备注
DES	56位	IBM为美国政府(NBS/NIST)开发	很多政府强制使用
3DES	3*56位	IBM为美国政府(NBS/NIST)开发	应用3次DES
CAST	40~256位	北方通信	比DES稍快
RC2	可变	Ron Rivest(RSA数据安全)	专利细节未公开
RC5		Ron Rivest(RSA数据安全)	
AES			

非对称密钥密码

1976年，Whitfield Diffie和Martin Hellman永远改变了密码学的范例（NSA宣称早在1966年就有这种概念的知识，但没有提供证据）。他们提出了公开密钥密码学。他们使用两个不同的密钥：一个是公开的，另一个是秘密的。持有公钥的任何人都可加密信息，但却不能解密。只有持有私钥的人才能解密。就好像有人把密码保险柜变成一个信箱，把邮件投进邮箱相当于用公开密钥加密，任何人都可以做，只要打开窗口，把它投进去。取出邮件相当于用私钥解密。一般情况下，打开它是很难的，你需要焊接机和火把。然而，如果你拥有私钥（开信箱的钥匙），就很容易从邮箱中取出邮件。

公钥加密体系是密码学上的一个重要里程碑。公钥加密的主要特点是加密和解密不需要用同一个钥匙。在公钥加密体系中，密钥论“对”，一个称为“公钥”，一个称为“私钥”。公钥和私钥互为“逆运算”，即用公钥加密的信息只有用它对应的私钥才能解密，而用私钥加密的信息也只有用它对应的公钥才能正确解密。公

钥要广为传播，越广泛越好；私钥为个人所有，越秘密越好。而且，令人惊奇的是，公钥的广泛传播并不会影响私钥的秘密性，即公钥和私钥之间几乎没有什么相关性，由公钥推出私钥的可能性几乎为零。

1976年Whitfield Diffie和Martin Hellman提出了公开密钥密码学，这给密码学界带来了极大的振奋。密钥成对产生，且从一个密钥推算出另一个是不可行的。这使得人们不再需要同其他人共享秘密密钥或者说至少他们可以以一种安全的方式共享秘密密钥。

公钥密码学的出现使大规模的安全通信得以实现，即 解决了密钥分发问题，并且公钥密码学还可用于另外一些应用，比如数字签名、防抵赖等。

注：公钥密码体制的基本原理就是陷门单向函数(troopdoor one-way function)，在此不作深入

公开密钥密码学的安全性基于解决“困难问题”的难度。举例来说，很多公开密钥算法把它们的安全性建筑在分解两个大素数乘积的难度上。

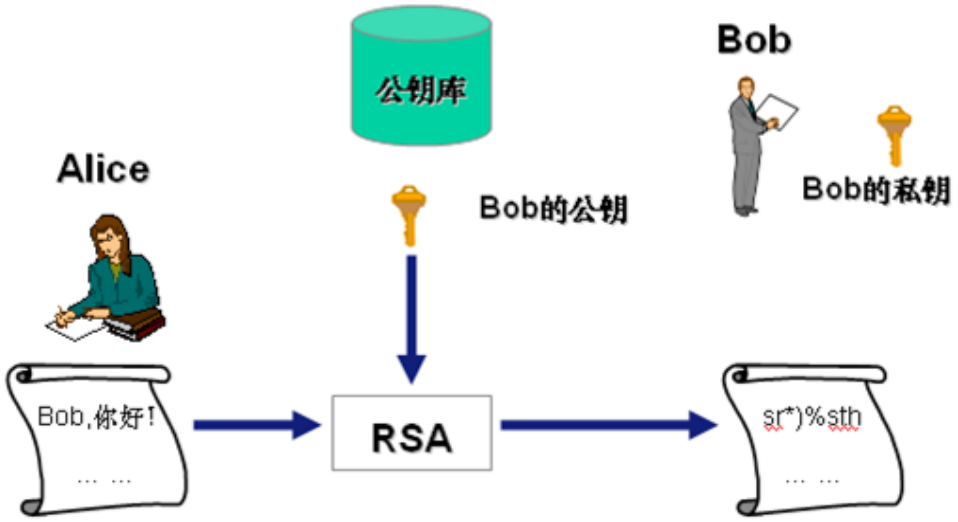
不是所有的公开密钥算法都具有相同的功能。正如你将要在下面看到的，一些算法用于加密，一些算法用于密钥交换，另一些算法用于数字签名。

公钥加密体系加密的基本过程【注意，用公钥体系不再需要在发送方和接收方之间传递秘密密钥】

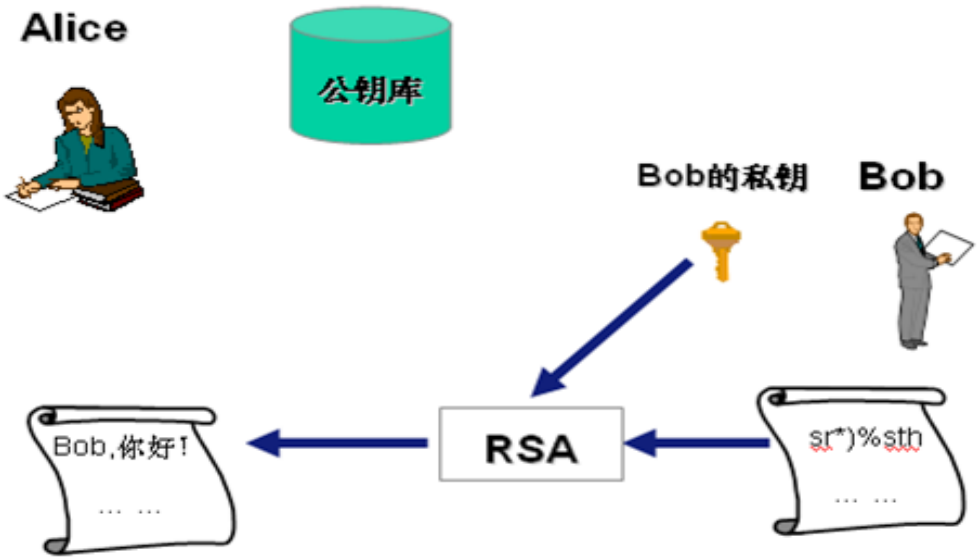
如图所描述的，Alice要用公钥体系给Bob发一封加密信，她应该：

1. 从一个公共路径得到Bob的公钥

2. 用这个公钥将信用一个公钥加密算法加密（如RSA）并发给Bob



3. 当Bob接到这个信后，他可用自己的私钥解开这个信函



简要的说，非对称加密算法工作过程就是：

用户甲拥有两个对应的密钥，用其中一个加密，只有另一个能够解密，两者一一对应；用户甲将其中一个私下保存（私钥），另一个公开发布（公钥）。

如果乙想送秘密信息给甲，那么乙获得甲的公钥，并使用该公钥加密信息发送给甲。

甲使用自己的私钥解密信息。

非对称加密算法的关键特点，即：

◆ 公钥加密_私钥解密；

◆ 私钥加密_公钥解密

◆ 不可互相推算

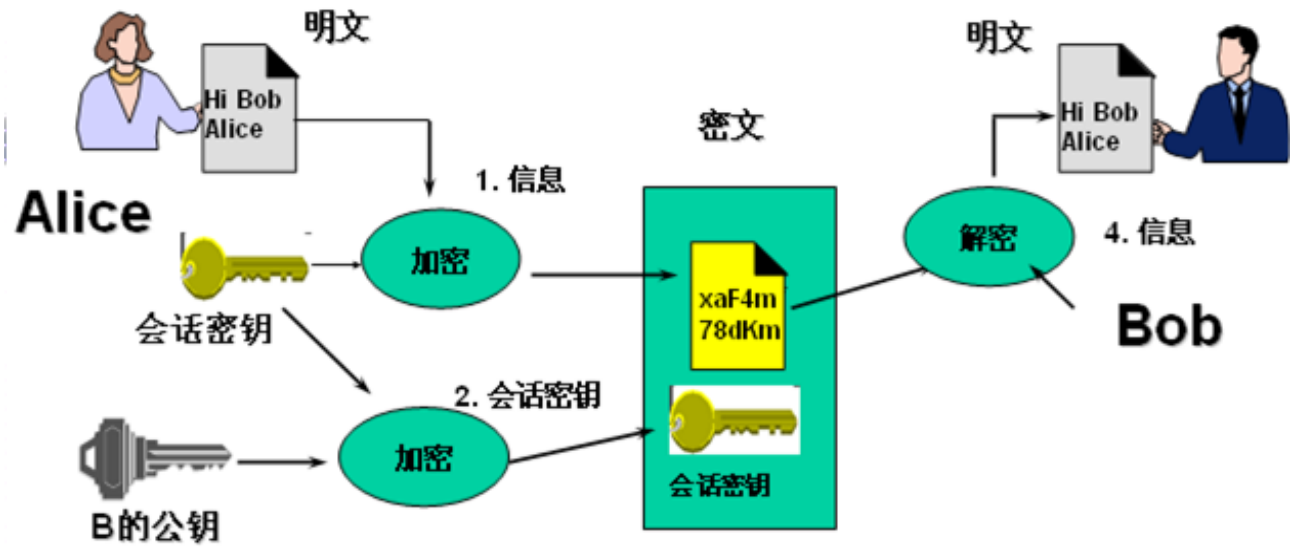
非对称加密算法的优点在于密钥管理方面，公钥可以公开、分布式存放；其缺点也是明显的，效率较慢，这导致了不适用于大量的数据加密应用场景。

常见的非对称加密算法有RSA、ECC、Diffie–Hellman、DSA。

算法	设计者	用途	安全性
RSA	RSA数据安全	加密 数字签名 密钥交换	大数分解
DSA	NSA	数字签名	离散对数
DH	Diffie&Hellman	密钥交换	完全向前保密

对称加密算法和非对称加密算法各自有其优缺点，但我们心目中理想的解决方案，希望是，必须是安全的、加密速度必须快、密文必须紧凑、能够适应参与者数目很多的情况、能够抵抗密钥窃听攻击、不用要求事先在参与方之间建立联系、支持数字签名和不可否认性。因此，组合密码技术就出现了：使用对称加密算法进行大批量的数据加密（每次产生一个新的随机共享密钥）、使用非对称加密算法传递随机产生的共享密钥。

组合密码技术



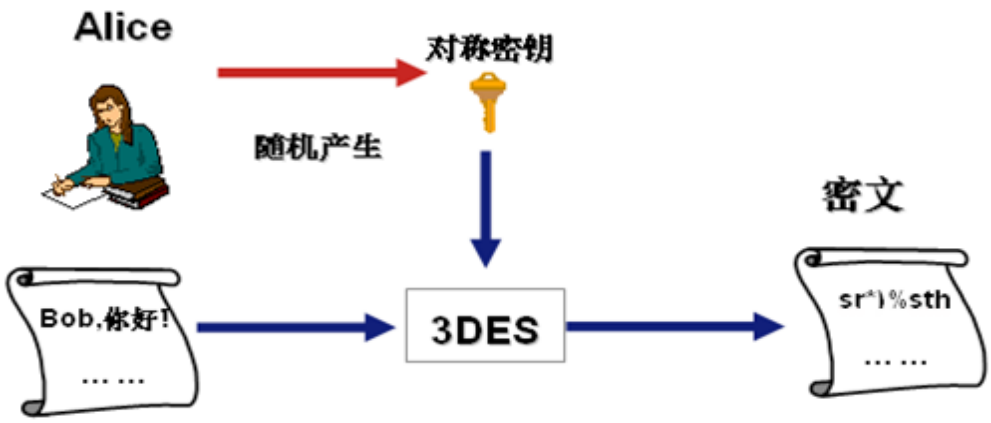
◆ 产生一个一次性，对称密钥——会话密钥

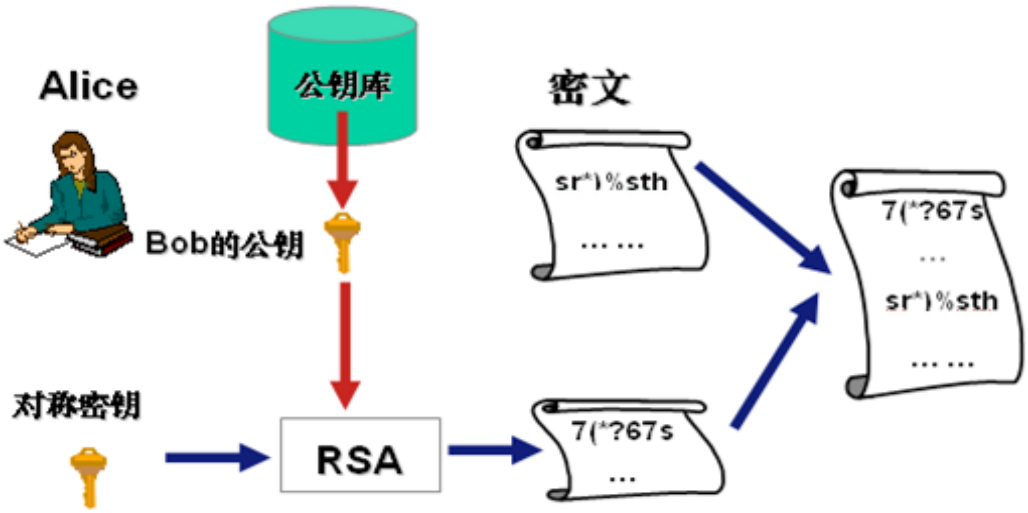
◆ 用会话密钥加密信息

◆ 最后用接收者的公钥加密会话密钥——因为它很短

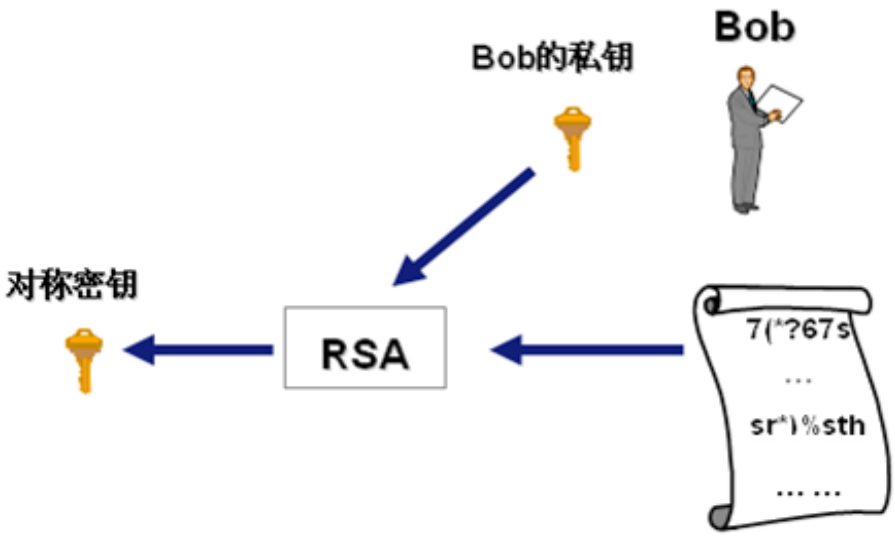
对称加密和公钥加密的结合应用图示如下

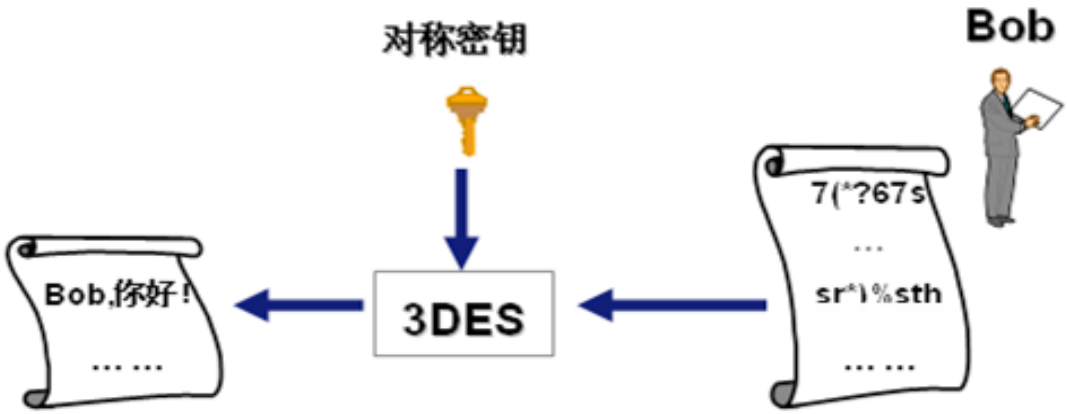
Alice的加密过程：





Bob在接收端的解密过程：





数字签名

公/私钥系统还有一个好处就是其非对称的天性。这使得每个密钥对的持有者都可以用他们的私钥进行数学运算，而这种运算是其他人都无法做到的。这就是数字签名和不可否认性的基础。

典型的数字签名技术使用非对称密码体制。在公钥密码体制中，数字签名技术可以用来保证原始信息的真实性，而且能够保证原始信息的完整性。

数字签名是用签名方的私钥对原始信息采取不可逆的单向散列算法提取出的一串唯一特征码（称为：信息摘要）进行加密的一个过程，这个过程所得到的密文即称为签名信息。

摘要算法（Hash）

Hash算法具有不可逆的特性，对任何长度的信息进行哈希后，结果都是一个固定长度的数据摘要，

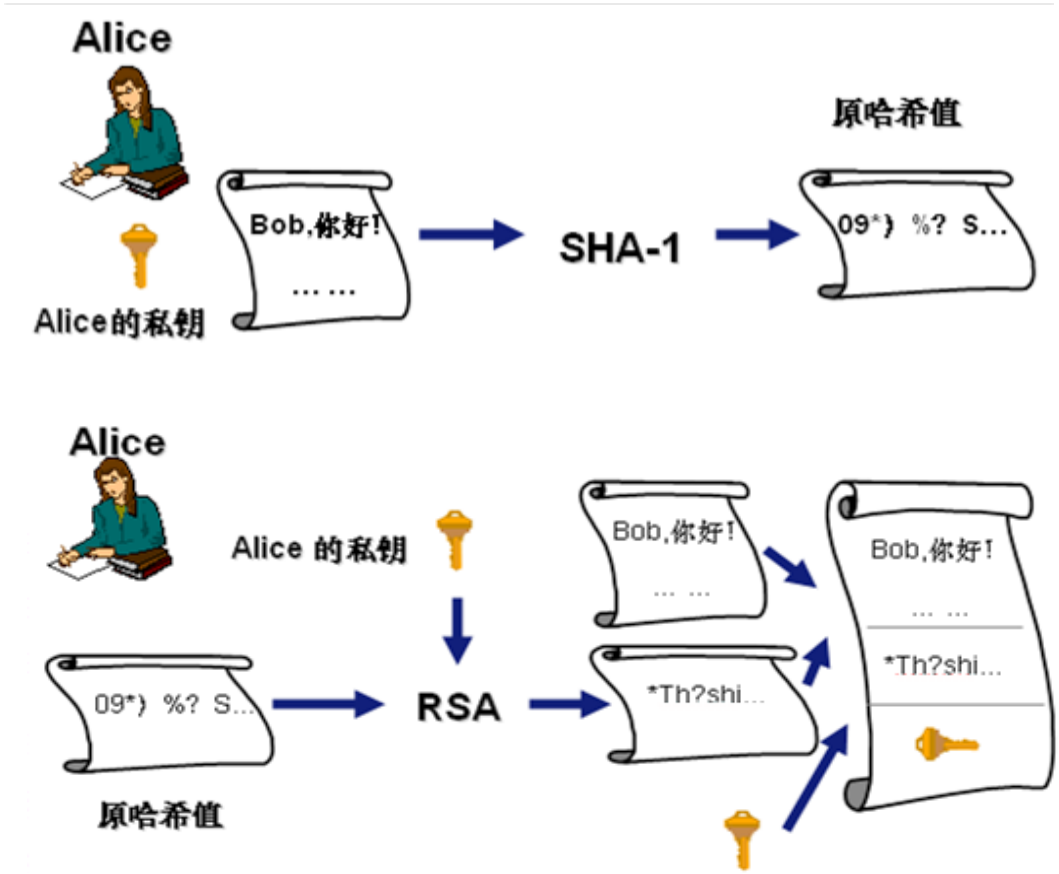
摘要的长度通常为128 bits或160 bits，原始信息中一个字节的改变会导致摘要后的结果发生变化。

常见的Hash算法有MD5和SHA-1。

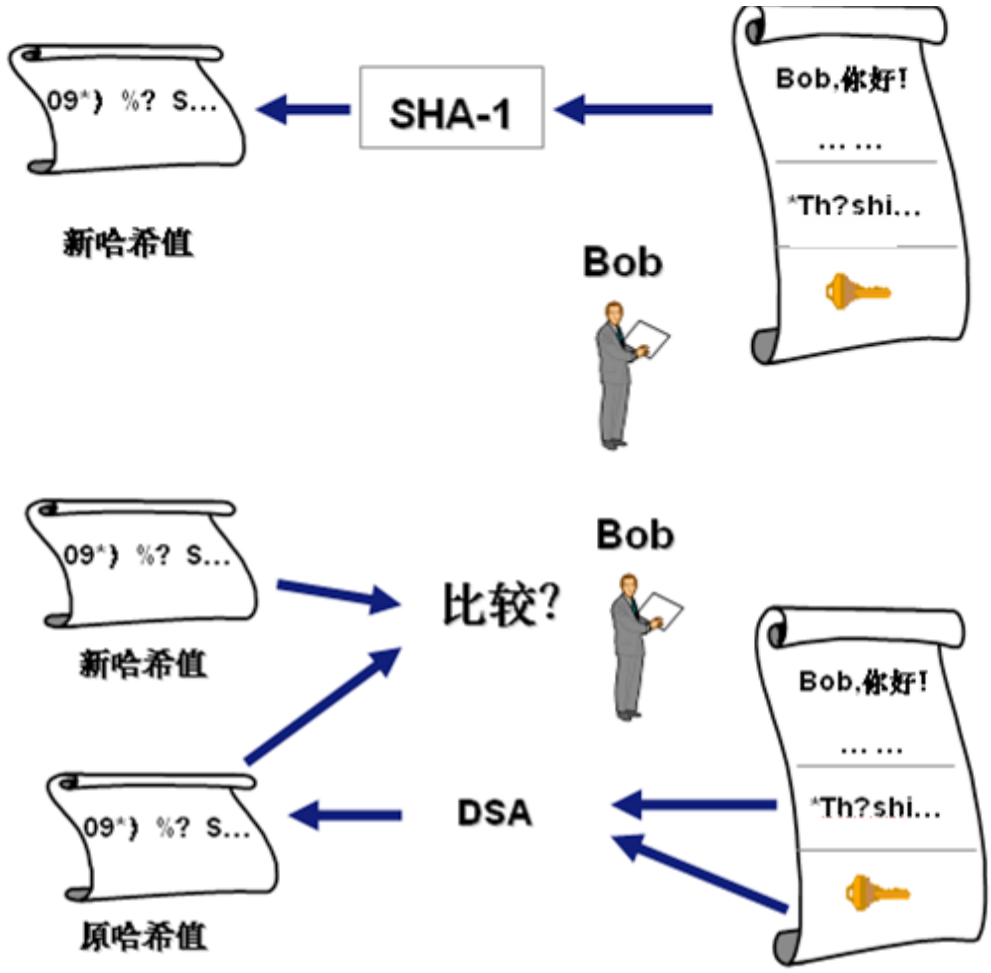
输入	哈希结果（使用 MD5）
Could you please transfer $100 from my checking account to the account of Mr.Smith?	D55f1123532d43a16a08557236615502
Could you please transfer $1000 from my checking account to the account of Mr.Smith?	b7b7ba62cae668d8a47bbdf5128a1055

哈希函数和公钥加密的结合(数字签名) 应用

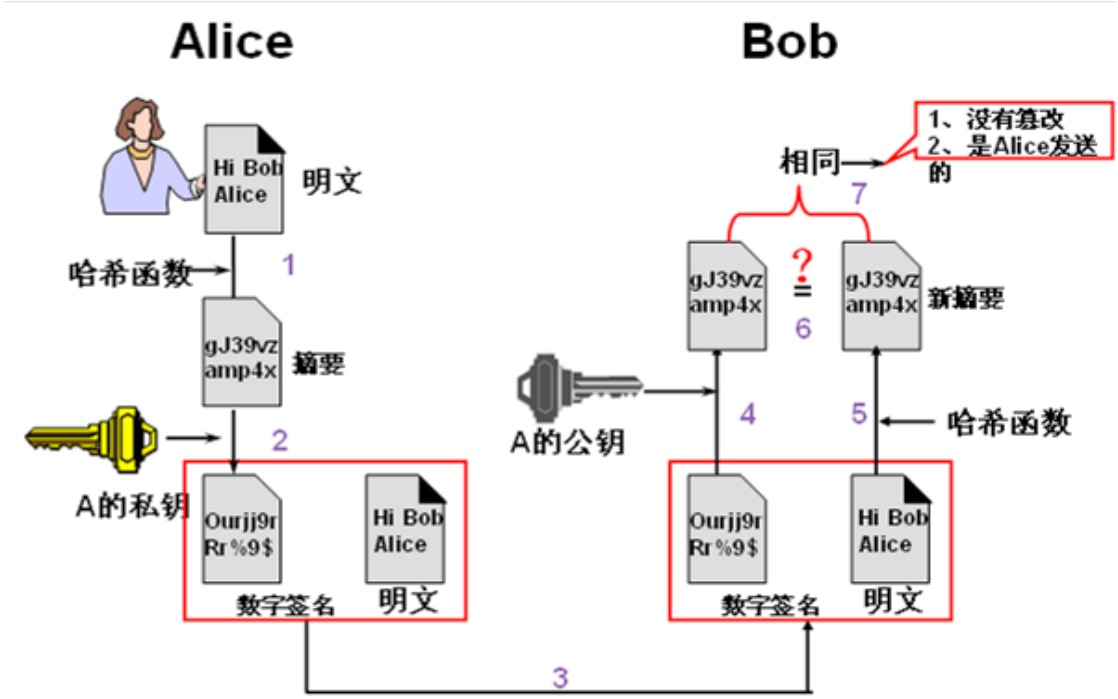
Alice对文件数字签名的过程：



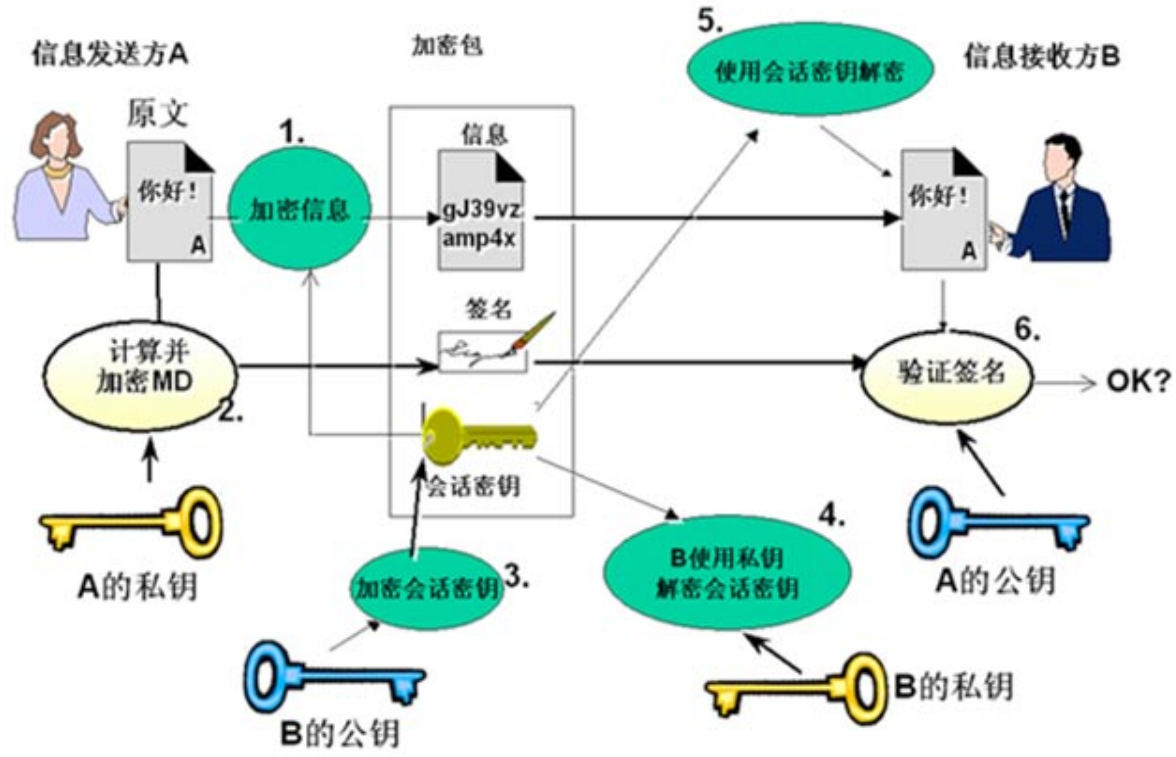
Bob验证数字签名的过程：



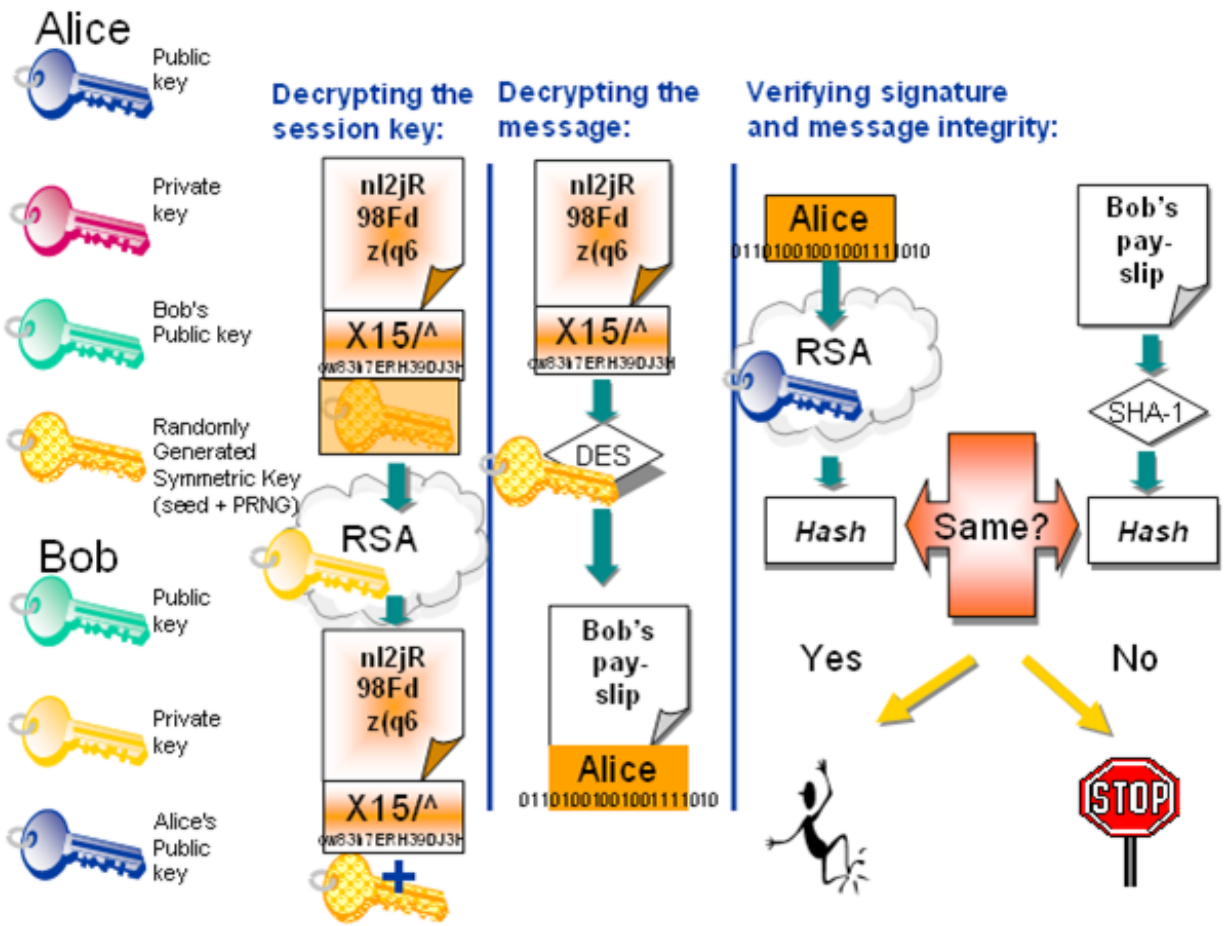
哈希算法和数字签名结合



加密和数字签名结合



接收机密数据的过程



为解决数据通信中的机密性、完整性、身份认证、不可抵赖性问题，我们引入了四种加密机制，对称加密、非对称加密、数字签名和Hash算法。但仅有这些是不够的，我们还面临着如何将此技术应用到具体的商业及应用中的问题？

即，如何验证数字签名？对方身份是否可信？因此，我们需要找到某种方式确保一个特定的公钥确实属于一个特定的人。

这个方式就是引入认证机构、数字证书和PKI。

［Tips：

PKI/CA发展历程

◆ 1976年，提出RSA算法

◆ 20世纪80年代，美国学者提出了PKI的概念

◆ 为了推进PKI在联邦政府范围内的应用，1996年就成立了联邦PKI指导委员会

◆ 1996年，以Visa、MastCard、IBM、Netscape、MS、数家银行推出SET协议，推出CA和证书概念

◆ 1999年，PKI论坛成立

◆ 2000年4月，美国国防部宣布要采用PKI安全倡议方案。

◆ 2001年6月13日，在亚洲和大洋洲推动PKI进程的国际组织宣告成立，该国际组织的名称为“亚洲PKI论坛”，其宗旨是在亚洲地区推动PKI标准化，为实现全球范围的电子商务奠定基础 】

数字证书

数字证书是各类实体(持卡人/个人、商户/企业、网关/银行等)在网上进行信息交流及商务活动的身份证明，在电子交易的各个环节，交易的各方都需验证对方证书的有效性，从而解决相互间的信任问题。证书是一个经证书认证中心数字签名的包含公开密钥拥有者信息以及公开密钥的文件。

从证书的用途来看，数字证书可分为签名证书和加密证书。签名证书主要用于对用户信息进行签名，以保证信息的不可否认性；加密证书主要用于对用户传送信息进行加密，以保证信息的真实性和完整性。

简单的说，数字证书是一段包含用户身份信息、用户公钥信息以及身份验证机构数字签名的数据。身份验证机构的数字签名可以确保证书信息的真实性。证书格式及证书内容遵循X.509标准。

证书格式

- 证书格式版本(目前是X509V3、V4)
- 证书序列号
- 签名算法标识符
- 认证机构的X.500名字
- 有效期
- 主题X.500名字
- 主题公钥信息
- 发证者唯一标识符
- 主题唯一标识符
- 扩展域

理解数字证书是理解PKI系统的中心问题。一个网络用户的证书等于他的电子版的身份证。它由CA签发，获得证书的个人或组织可以用它来证明自己的身份或存取信息的权利。当CA签发证书时，它验证证书申请者的身份，这样如果CA给商家发了一张证书，就意味着CA保证了商家使用它的名字和Web地址的权利。

数字证书应用在公开密钥系统中。在公开密钥系统中，当你的密钥对产生的时候，你将一个作为私有。这个密钥被称为“私钥”，除了你，没有人能够存取它。然而，对应的“公钥”作为数字证书的一部分被自由分配。你可以和任何人分享你的证书，或将你的证书公布到LDAP服务上去。如果有人想和你私下通信，在数据发送给你之前，他使用你的证书中的公钥去加密数据。只有你能够解密数据，因为只有你拥有私钥。

同样地，你能够使用你的密钥对签名一段信息。为了对一段信息签名，你使用你的私钥加密信息。加密的信息可以使用包含在你的证书中的公钥解密。很多人能够使用你的证书，而只有你能够对信息签

名，因为只有你有权使用你的私钥。

一张数字证书是一个二进制文件。它包含了证书持有者的名字、能够证明证书持有者身份的可靠信息还有证书持有者的公钥，这给出了证书持有者和他的公钥之间的对应关系。在证书中的公钥既可被他人用于加密也可用来验证持有者的数字签名。数字证书中也包含序列号、有效期、与证书相联系的权利和使用信息等。最后，数字证书中包含发证机关CA的信息。所有证书都用CA的私钥进行数字签名。

通常，CA的证书（也称为根证书）广泛地配置在软件包中，这使得用户可以方便自然地验证CA签发的证书的合法性。如果CA能够安全地保护它的私钥，对任何人来说伪造证书都是不可能的。

证书不仅仅发给个人，组织、还有诸如服务器和路由器这样的实体都能获得证书。

讲了这么多，回来总结一下什么是数字证书？

◆ PKI系统(CA系统)的产物

◆ 数字证书是PKI技术的现实载体，通过数字证书我们可以实现身份认证、信息加密、签名等一系列的安全操作

◆ 网络世界的电子身份证

与现实世界的身份证类似，能够证明个人、团体或设备的身份

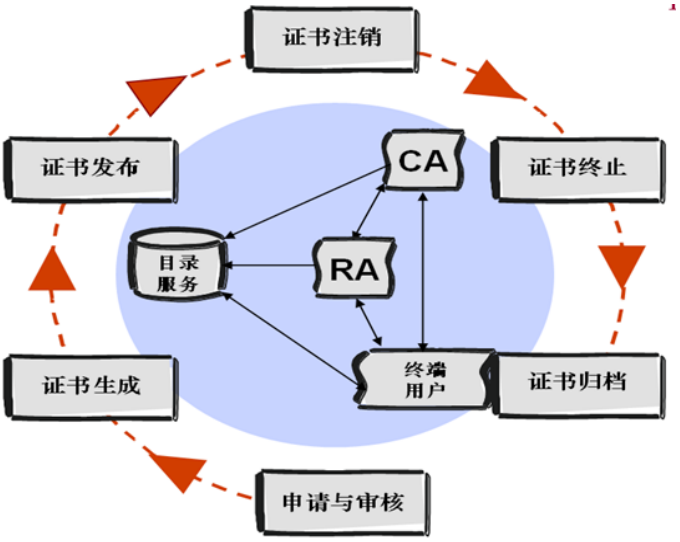
◆ 包含相关信息：姓名、地址、公司、电话号码、Email地址等（与身份证上的姓名、地址等类似）、包含所有者的公钥

◆ 拥有者拥有证书公钥对应的私钥

◆ 由可信的颁发机构颁发，比如身份证由公安局颁发一样

◆ 颁发机构对证书进行签名，与身份证上公安局的盖章类似，可以由颁发机构证明证书是否有效，可防止擅改证书上的任何资料

PKI及其构件

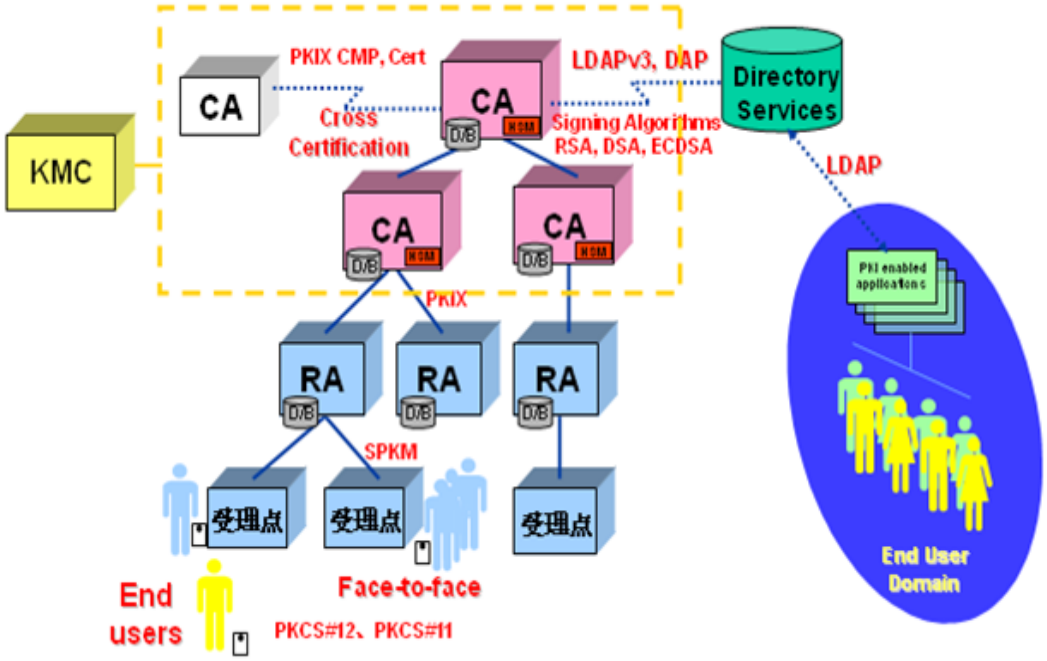


现在我们再回过头来看PKI的定义：“PKI（Public Key Infrastructure，公钥基础设施）是一种运用公钥的概念与技术来实施并提供安全服务的具有普遍适用性的网络安全基础设施。 它由公私钥对、数字证书、CA 、RA、 CRL库、证书库、LDAP目录、参与验证的设备和支持PKI的应用程序组成”。

现在，这个定义不再那么云里雾里了吧？PKI系统的前两个部分我们已经做了阐述，现在我们再来看剩下的。

从总体来说，PKI的作用就是要完成创建、管理、保存、发放和废止证书的功能，通过管理数字证书的生命周期来管理对应的身份信息的生命周期。

要实现上述功能，PKI系统采用的标准结构如图示：

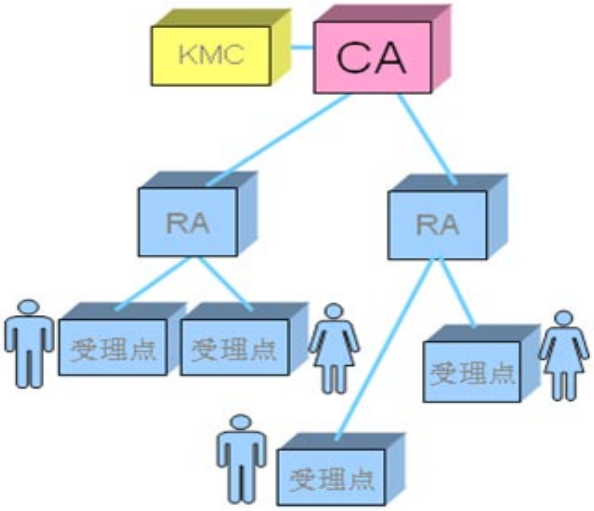


下面我们逐一解释各个组成部分。

证书机构(CA)：

PKI体系中,公钥系统采用需要一个可信任的第三方充当认证中心,来确认公钥拥有者的真正身份,签发并管理用户的数字证书. CA的主要功能接受并验证最终用户数字证书的申请,证书的审批,证书的签发,证书的更新,证书的查询和撤消,产生和发布证书废除列表等。

- ◆ 发布本地CA策略
- ◆ 对下级机构进行认证和鉴别
- ◆ 产生和管理下属机构的证书
- ◆ 接收和认证RA证书请求
- ◆ 签发和管理证书、CRL
- ◆ 发布证书CRL
- ◆ 密钥安全生成及管理
- ◆ 交叉认证



注册机构（RA）

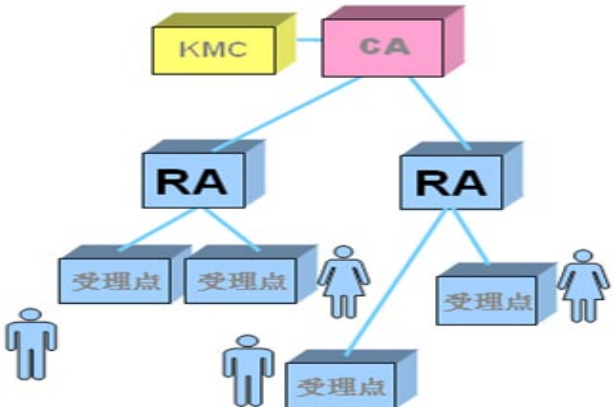
注册机构RA是PKI信任体系的重要组成部分,是用户和认证中心CA之间的一个接口,是认证机构CA信任范围的一种延伸.注册机构RA接受用户的注册申请,获取并认证用户的身份,主要完成收集用户信息和确认用户身份的功能。

- ◆ 进行用户身份信息的审核，确保其真实性；
- ◆ 本区域用户身份信息管理和维护；

◆ 数字证书的下载；

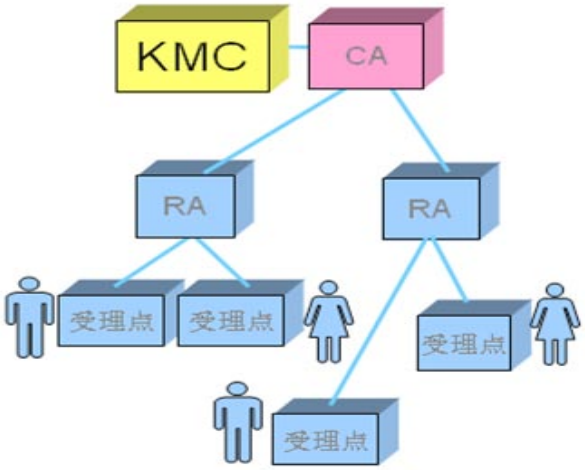
◆ 数字证书的发放和管理

◆ 登记黑名单



密钥管理中心（KMC）

负责：密钥的生成、密钥的分发、密钥的备份、密钥的恢复、密钥的更新、密钥的归档、密钥查询、密钥销毁。



证书/密钥的存储

一个PKI体系由终端实体、证书机构、注册机构和PKI存储库四类实体共同组成如下图所示。其中PKI存储库包括LDAP（Lightweight Directory Access Protocol，轻量级目录访问协议）服务器和普通数据库，用于对用户申请、证书、密钥、CRL和日志等信息进行存储和管理，并提供一定的查询功能。

存储的信息包括：用户的证书、用户的私钥、CA根证书。

PKI工作过程

1. 实体向RA提出证书申请

2. RA审核实体身份，将实体身份信息和公开密钥以数字签名的方式发送给CA

3. CA验证数字签名，同意实体的申请，颁发证书

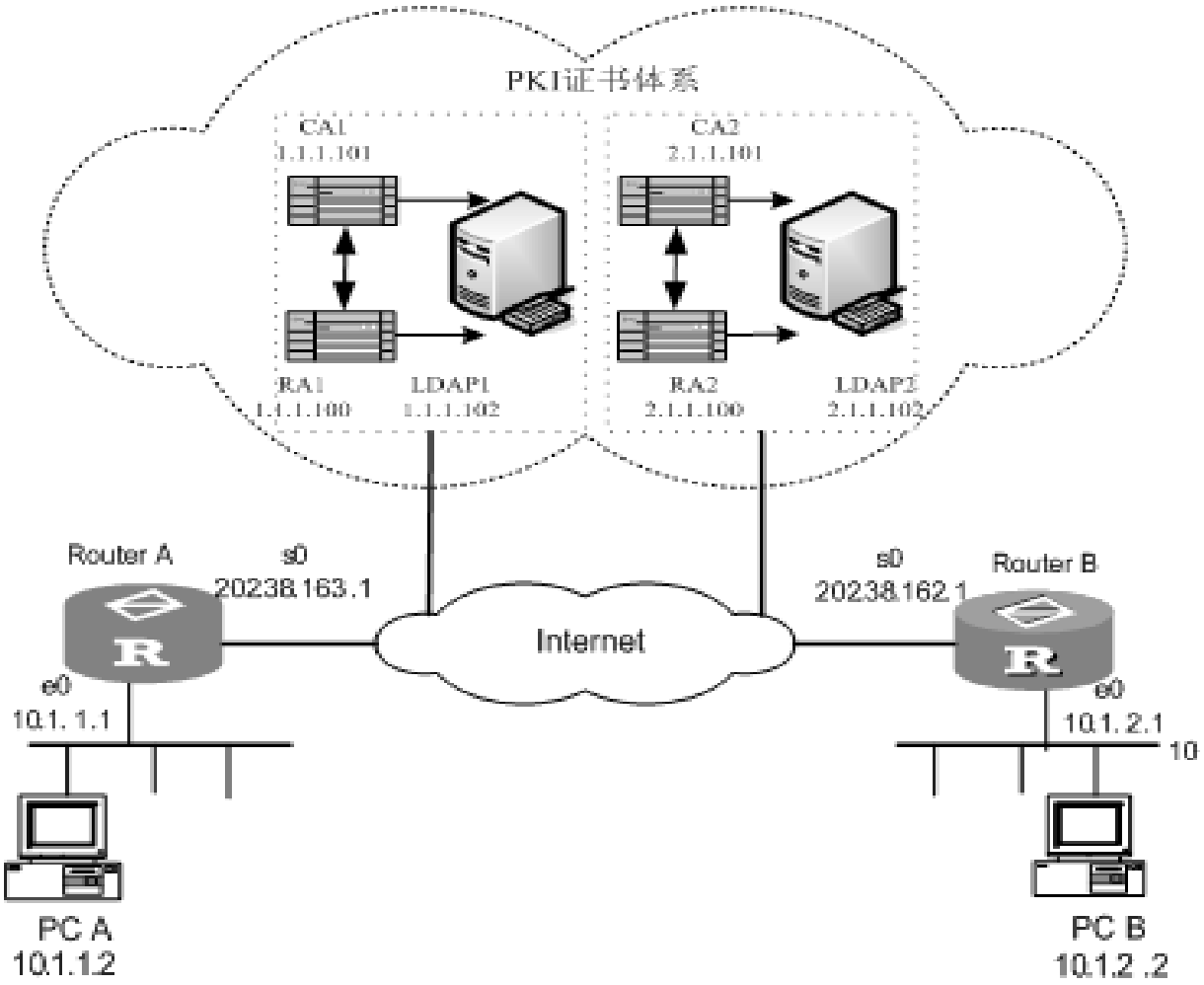
4. RA接收CA返回的证书，发送到LDAP服务器以提供LDAP浏览服务，并通知实体(即终端实体，可以是个人，组织或者设备)证书发行成功

5. 实体获取证书，利用该证书可以与其它实体使用加密、数字签名进行安全通信

6. 实体希望撤消自己的证书时，向CA提交申请。CA批准实体撤消证书，并更新CRL，发布到LDAP服务器。

用下图可以清楚地看到PKI的工作过程：

PCA与PCB通过路由器连到因特网上，从而可以与网络中的某个认证机构建立连接。PCA要申请证书时，先登陆此PKI证书体系，向它发出自己的申请，并按照当地CA的策略填写相关信息。此申请信息会被注册机构RA转给CA服务器，CA负责对其验证,通过后会给PCA颁发相应级别的证书。同时PCA的用户信息和刚颁发的数字证书也会被置于LDAP（轻量级目录访问协议）服务器中，便于PCA登陆取回自己的证书，或者提供给其他用户进行访问。



LDAP在CA中的应用

PKI系统中，CA需要解决的两个问题：

1. 证书生命周期管理问题：如何创建、维护和销毁证书

2. 证书定位问题：如何快速找到对方的证书

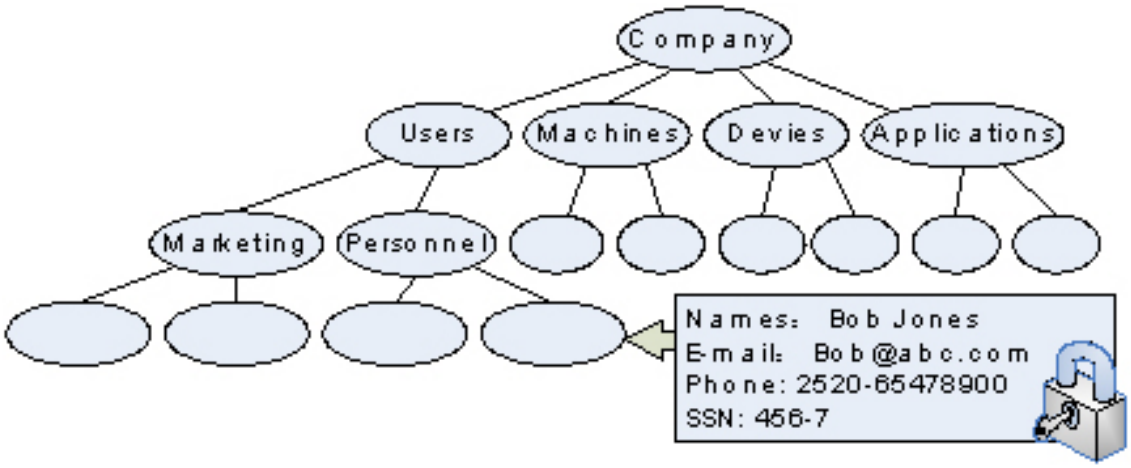
这两个问题是通过LDAP目录解决的。

目录是整个证书生命周期的管理中心；在目录中存储、管理证书(Certificate)和撤销列表(CRL)；通过目录服务提供有效的证书发布和查询功能；通过目录服务安全可靠地发布CRL,提供CRL查询服务

LDAP——Light-weight directionary access protocol 轻量级目录访问协议。LDAP服务器提供目录浏览服务，负责将注册机构服务器传输过来的用户信息以及数字证书加入到LDAP服务器上。用户通过访问LDAP服务器就能够得到自己和其他用户的数字证书。

LDAP服务器提供针对树状信息的查询语言；数据访问控制；快速的精确查询；有详细的访问协议；数据分布存储与访问。适用于以快速查询为主的应用、采用树状结构、存储基本数据、对数据的访问控制可到条目属性级别。

其具有以下特点：



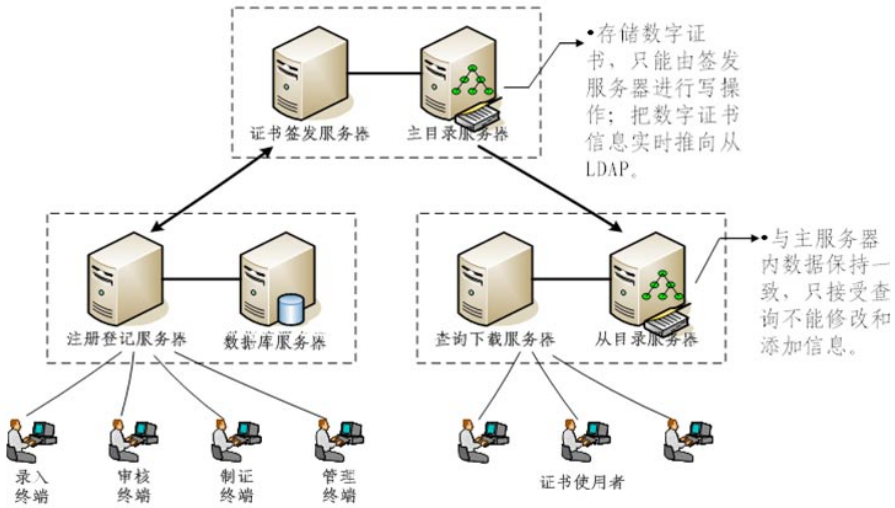
◆ 独立性： LDAP既是一个X.500的访问协议，又是一个灵活的可以独立实现的LDAP系统。

◆ 基于Internet的协议：LDAP基于Internet协议，X.500基于OSI协议。

◆ 简单性：LDAP继承了X.500最好的特性，同时去掉了它的复杂性。

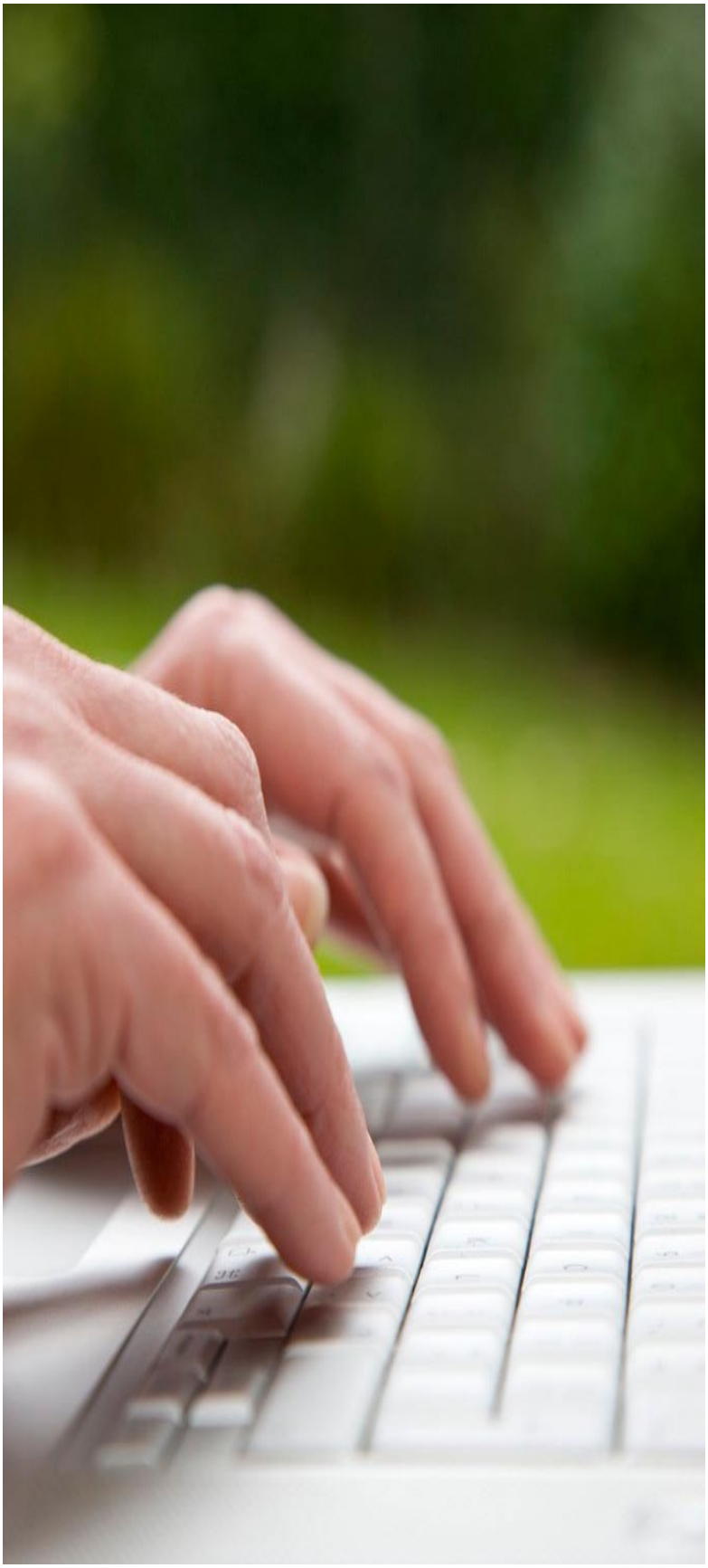
◆ 分布式：LDAP通过引用（Referral）机制实现分布式：X.500 DSA通过服务器之间的链操作实现分布式的访问

LDAP服务器在CA中的位置



■文/曾劼

Kerberos认证浅析



在希腊神话中Kerberos是守护地狱之门的一条凶猛的三头神犬，而我们在本文中所要介绍的Kerberos认证协议是由美国麻省理工学院(MIT)首先提出并实现的，是该校雅典娜计划的一部分。这个定名是贴切的，因为Kerberos认证是一个三路处理过程，依赖称为密钥分发中心（KDC）的第三方服务来验证计算机相互的身份，并建立密钥以保证计算机间安全连接。本质上每台计算机分享KDC一个秘密，而KDC有两个部件：一个Kerberos 认证服务器和一个票据授权服务器。如果KDC不知道被请求目标服务器，则会求助于另一个KDC来完成认证。它允许在网络上通讯的实体互相证明彼此的身份，并且能够阻止窃听和重放等攻击手段。不仅如此，它还能够提供对通讯数据保密性和完整性的保护。

Kerberos从提出到今天，共经历了五个版本的发展。其中第一版到第三版主要由该校内部使用。当发展到第四版的时候，已经取得了在MIT校外的广泛认同和应用。由于第四版的传播，人们逐渐发现了它的一些局限性和缺点，例如适用网络环境有限、加密过程存在冗余等等。MIT充分吸取了这些意见，对第四版进行了修改和扩充，形成了今天非常完善的第五版。第五版由John Kohl和Clifford Neuman设计，在1993年作为RFC 1510颁布，在2005年由RFC 4120取代，目的在于克服第四版的局限性和安全问题，用更细化和明确的解释说明了协议的一些细节和使用方法。

▲ Kerberos认证应用环境假定▲

Kerberos协议的基本应用在一个分布式的Client/Server 体系结构中，采用一个或多个Kerberos 服务器提供鉴别服务。客户端想请求应用服务器上的资源，首先客户端向密钥分发中心请求一张身份证明，然后将身份证明交给应用服务器进行验证，在服务器验证通过后，即为客户端分配请求的资源。

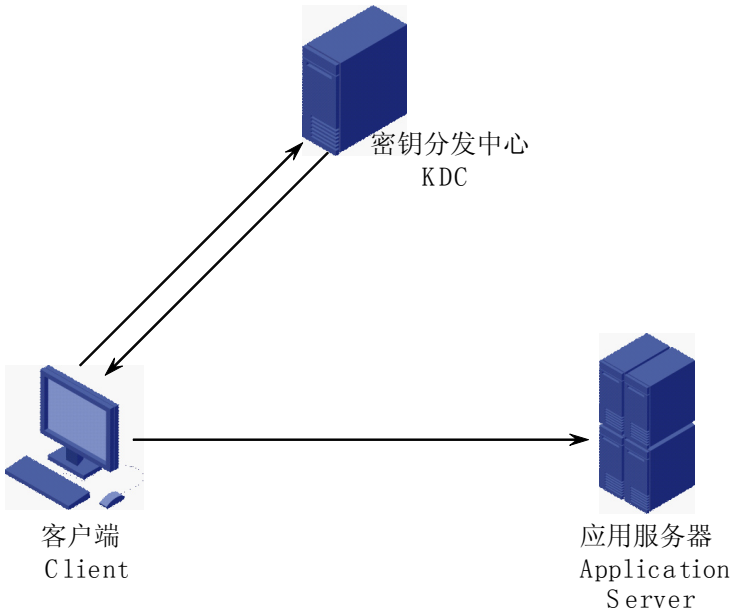


图1 Kerberos的基本应用

Kerberos协议本身并不能完全解决网络安全性问题，它是建立在一些假定之上的，只有在满足这些假定的环境中它才能正常运行。

（1）不存在拒绝服务(Denial of Service)攻击。Kerberos不能解决拒绝服务攻击，在该协议的很多环节中，攻击者都可以阻断正常的认证步骤。这类攻击只能由管理员和用户来检测和解决。

（2）主体必须保证他们的私钥的安全。如果一个入侵者通过某种方法窃取了主体的私钥，他就能冒充身份。

（3）Kerberos无法应付口令猜测攻击。如果一个用户选择了弱口令，那么攻击者就有可能成功地用口令字典破解掉，继而获得那些由源自于用户口令加密的所有消息。

（4）网络上每个主机的时钟必须是松散同步的。这种同步可以减少应用服务器进行重放攻击检测时所记录的数据。松散程度可以以一个服务器为准进行配置。时钟同步协议必须保证自身的安全，才能保证时钟在网上同步。

（5）主体的标识不能频繁地循环使用。由于访问控制的典型模式是使用访问控制列表(ACLs)来对主体进行授权。如果一个旧的ACL还保存着已被删除主体的入口，那么攻击者可以重新使用这些被删除的用户标识，就会获得旧ACL中所说明的访问权限。

▲Kerberos认证的工作过程▲

Kerberos认证的基本原理

在网络中，认证主要用来解决各个通讯实体之间相互证明彼此身份的问题。对于如何进行认证，我们通常会采用这样的方法：如果一个秘密仅仅有认证方和被认证方知道，认证方可以通过让被认证方提供这个秘密来证明对方的身份。这个过程实际上涉及到认证的三个重要方面：秘密如何表示、被认证方如何向认证方提供秘密、认证方如何识别秘密。

基于这三个方面，Kerberos认证可以进行最大限度的简化成Client和Server两个通讯实体，他们之间共同的秘密用KServer-Client来表示。Client在认证过程中向Server提供以明文形式表示的Client标识和使用KServer-Client加密的Client标识以便于让Server进行有效的认证：

由于这个秘密仅仅被Client和Server知晓，所以被 Client加密过的Client标识只能被Client和Server解密。Server接收到Client传送的这两组信息，先通过KServer-Client对后者进行解密，随后将解密的数据同前者进行比较，如果完全一样，则可以证明Client能够提供正确的KServer-Client，而这个世界上，只有真正的Client和自己知道KServer-Client，这样就可以证明对方的真实性。

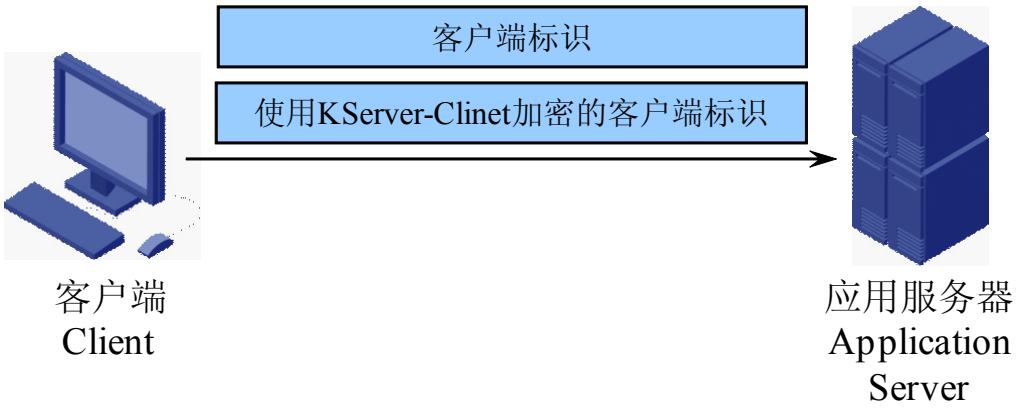


图2 客户端认证的简单过程

整个过程看起来非常简单，但是实际上Kerberos认证远比这个过程复杂的多，在了解Kerberos真实的认证过程之前我们先给出两个重要的概念：

长期密钥：在安全领域中，有的密钥可能长期内保持不变，比如密码，可能几年都不曾改变。这样的密钥以及由此派生的其他密钥被称为长期密钥。长期密钥有这样的原则：被长期密钥加密的数据不应该在网络上传输。因为任何加密算法都不可能做到绝对保密，一旦这些被长期密钥加密的数据包被黑客截获，在理论上，只要有充足的时间，都是可以通过计算获得用户用于加密的密钥的。

对于一个账户来说，密码仅限于该账户的所有者知晓，甚至对于管理员都应该是保密的。但是密码却又是证明身份的凭据，所以必须通过基于密码的派生信息来证明用户的真实身份，在这种情况下，一般将账户密码进行Hash运算得到一个Hash值，也可以称之为主密钥。由于Hash算法是不可逆的，同时可以保证密码和主密钥派生的确定性，这样既保证了密码的保密性，同时又保证主密钥和密码本身在证明用户身份时具有相同的效力。

短期密钥：由于被长期密钥加密的数据包不能在网络上传送，所以需要使用另一种密钥来加密需要进行网络传输的数据。这种密钥只在一段时间内有效，即使加密过的数据包被黑客截获，等他把密钥计算出来的时候，这个密钥早就已经过期了。我们把这种密钥称为短期密钥。

密钥分发中心——SServer-Client的分发

通过让被认证方提供一个仅限于他和认证方知晓的密钥来鉴定对方的真实身份。而被这个密钥加密的数据包需要在Client和Server之间传送，所以这个密钥只可能是一个短期密钥，它仅仅在Client和Server的一个Session中有效，所以我们称这个密钥为Client和Server之间的会话密钥，暂时表示为SServer-Client。

在介绍Client和Server是如何得到会话密钥之前要引入一个重要的角色：Key Distribution Center（KDC），也就是密钥分发中心。KDC在整个Kerberos认证系统中作为Client和Server共同信任的第三方起着至关重要的作用。

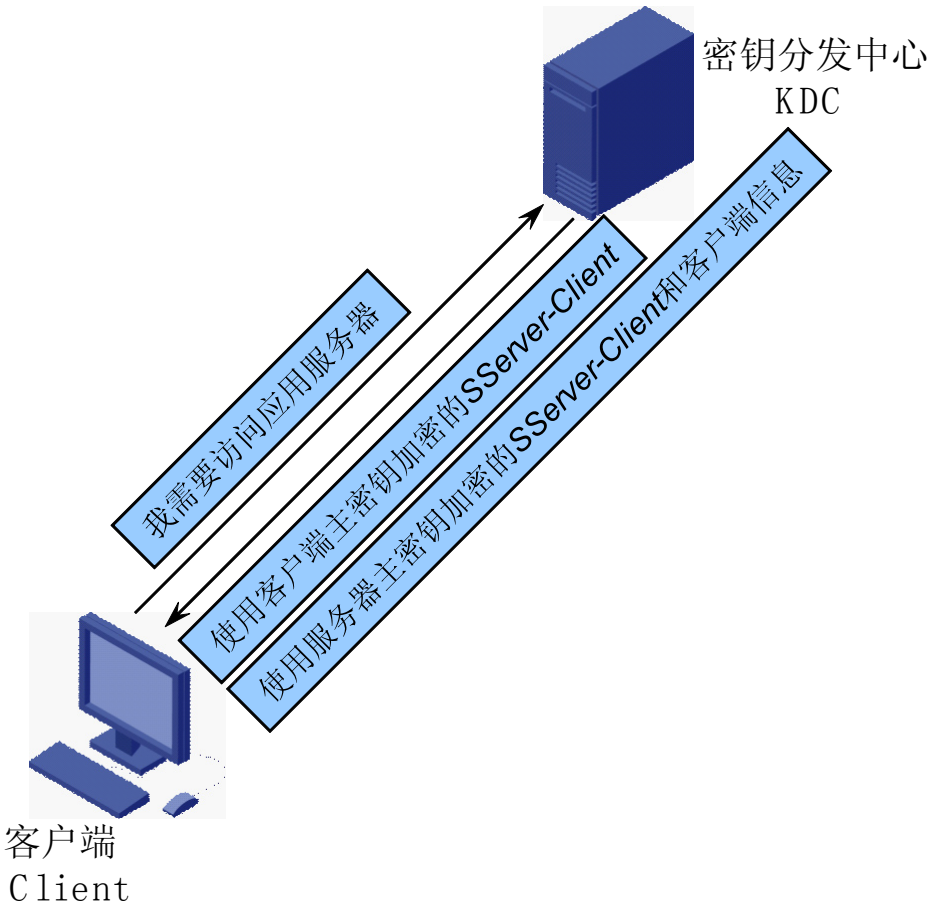


图3 KDC分发会话密钥的简单过程

KDC分发SServer-Client的简单过程如上图所示：首先 Client向KDC发送一个对SServer-Client的申请，其中包括自己的信息以及希望访问的Server的信息。KDC在接收到这个请求的时候，生成一个会话密钥，为了保证这个密钥仅限于发送请求的Client和他希望访问的Server知晓，KDC会为这个密钥生成两个拷贝，分别被Client和Server使用。然后从数据库中提取Client和Server的主密钥分别对这两个拷贝进行对称加密。对于后者，和会话密钥一起被加密的还有一些Client的信息。

KDC有了两个分别被Client和Server 的主密钥加密过的会话密钥，KDC会将这两个被加密的拷贝一并发给Client，属于Server的那一份将会由Client发送给Server。这样做可以轻松地解决两个问题：首先

Server不用维护一张庞大的会话密钥列表来应付不同的Client的访问，降低了Server的负荷；其次避免出现因为网络延时，Client的认证请求比Server的会话密钥早到达Server端，进而导致认证失败的情况。

鉴别码是有效证明自己的证据

在上述的过程之后Client实际上获得了两组信息：一个是通过自己的主密钥加密的会话密钥，另一个则是被Server的主密钥加密的数据包，其中包含会话密钥和关于自己的一些确认信息。虽然通过一个双方知晓的密钥就可以对对方进行有效的认证，但是在一个不安全的网络环境中，这种简单的做法是存在安全漏洞的，假设Client向Server发送的数据包被黑客截获，该黑客随后将数据包作为自己的凭证冒充该Client对Server进行访问，在这种情况下，依然可以很顺利地获得Server的成功认证。为此，Client需要提供更多的证明信息，称为鉴别码。在Kerberos中这个鉴别码实际上就是关于Client的一些信息和当前时间的一个时间戳。

Client使用自己的主密钥对KDC加密的会话密钥进行解密从而获得SServer-Client，随后创建鉴别码并用SServer-Client对其加密。最后连同从KDC获得的、被Server的主密钥加密过的数据包一并发送到Server端。我们把通过Server的主密钥加密过的数据包称为会话票据。

当Server接收到这两组数据后，先使用他自己的主密钥对会话票据进行解密，从而获得SServer-Client。随后使用SServer-Client解密鉴别码获取时间戳，同当前的时间进行比较，如果偏差超出一个可以接受范围，Server会直接拒绝该Client的请求。Server会维护一个列表，记录在可接受的时间范围内所有进行认证的Client和认证时间。对于时间偏差在可接受的范围中的Client，Server会从列表中获得该Client最近一次认证时间，只有当Client提供的时间戳晚于这个时间，Server才会比较鉴别码中的客户端信息和会话票据中的客户端信息从而实现对Client的认证。这也就是为什么我们从一开始就需要假定网络中每台主机的时钟必须是松散同步的了。

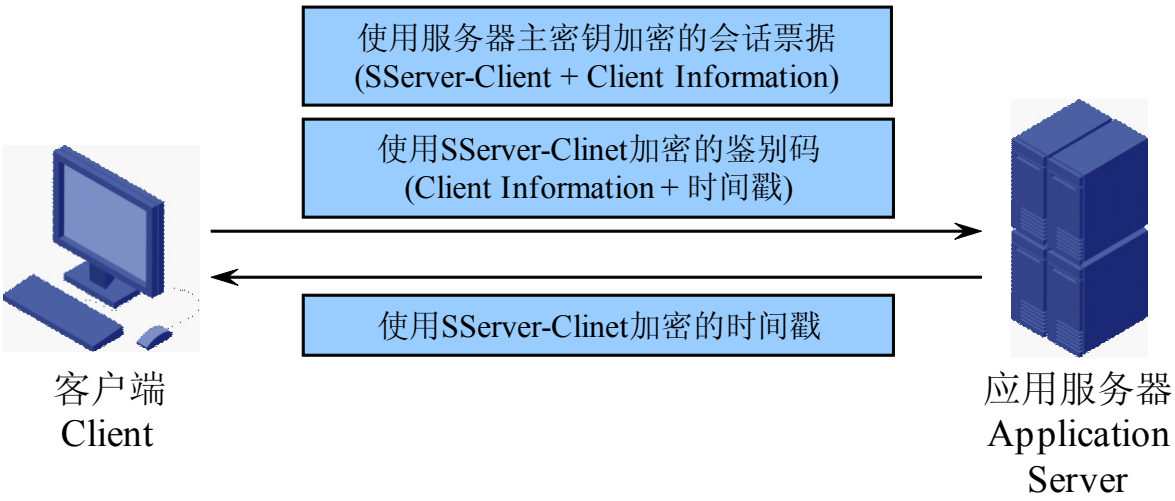


图4 Kerberos双向验证过程

Kerberos一个重要的优势在于它能够提供双向认证：不但Server可以对Client进行认证，Client也能对Server进行认证。如果Client需要对他访问的Server进行认证，会在他向Server发送的认证请求中设置一个是否需要双向认证的标志位。Server在对Client认证成功之后，会把鉴别码中的时间戳提取出来，通

过SServer-Client进行加密，当Client使用SServer-Client对Server回应的报文进行解密之后，如果确认时间戳和原来的完全一致，那么他可以认定这台Server正是他希望访问的Server。

票据授权服务——TGS

Kerberos实际上是一个基于票据的认证方式。 Client想要获取Server端的资源，先得通过Server的认证；而认证的先决条件是Client向Server提供从KDC获得的一个由 Server的主密钥加密过的会话票据。所以对Client来说，获得会话票据是整个认证过程中最为关键的部分。

在 Kerberos中票据的分发过程远没有上面所讲的那么简单，Client在从KDC处获得会话票据之前，需要先获得一个授权，被称为Ticket Granting Ticket（票据授权票据，简称TGT），TGT的分发仍然是KDC来完成的。

首先Client向KDC发起对TGT的申请，KDC在收到该申请后，生成一个用于该Client与KDC进行安全通信的会话密钥，暂时用SKDC-Client来表示。为了保证该会话密钥仅供该Client和自己使用，KDC使用Client的主密钥和自己的主密钥对生成的会话密钥进行加密，从而获得两个加密的SKDC-Client的拷贝。对于后者，随会话密钥一起被加密的还包含以后用于鉴定Client身份的关于Client的一些信息，这就是前面所说的TGT。最后KDC将这两份拷贝一并发送给Client。

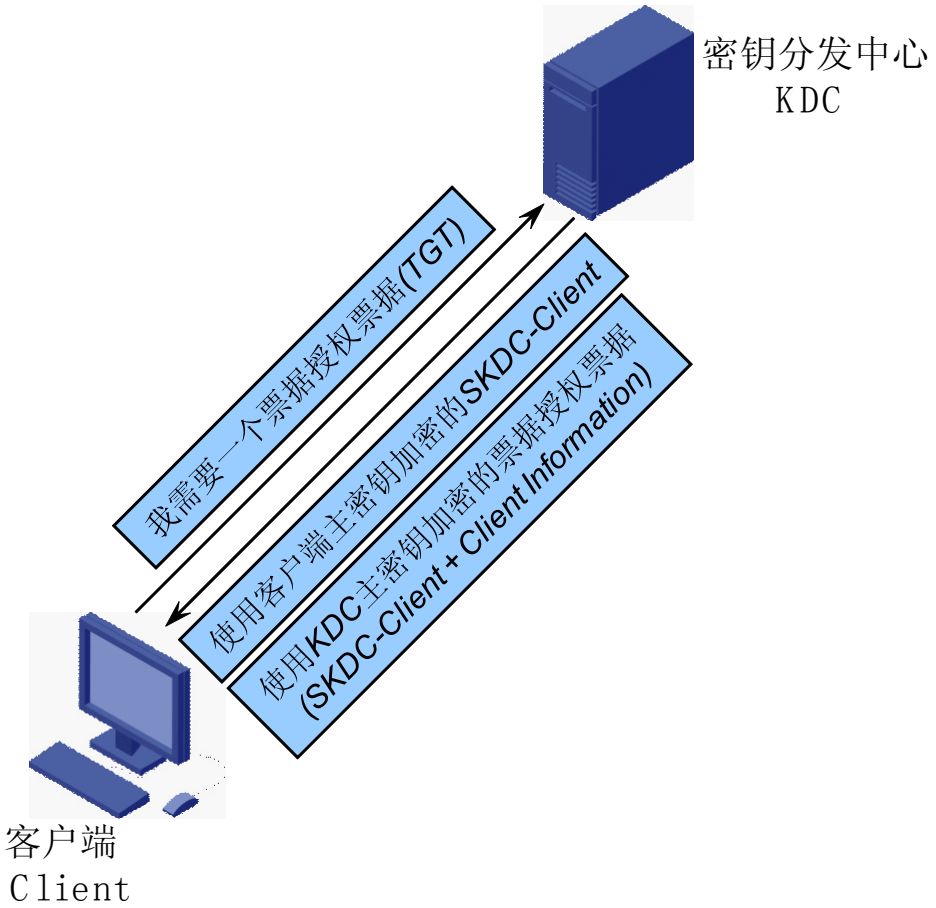


图5 TGT申请过程

当Client收到KDC的两个加密数据包之后，先使用自己的主密钥对第一个拷贝进行解密，从而获得SKDC-Client，并把它和TGT进行缓存。有了SKDC-Client和TGT，Client自己的主密钥将不再需要，因为此后Client可以使用SKDC-Client向KDC申请用以访问各个Server的票据，SKDC-Client是一个短期密钥，安全得到更好的保障，也正是因为如此，SKDC-Client具有自己的生命周期，同时TGT和SKDC-

Client相互关联，当SKDC-Client过期，TGT也就宣告失效，此后Client不得不重新向KDC申请新的TGT，KDC将会生成一个不同会话密钥和与之关联的TGT，另外Client的注销操作也会导致SKDC-Client失效。

Client在获得SKDC-Client之后，生成自己的鉴别码以及所要访问的Server名称，并使用SKDC-Client进行加密。 随后连同TGT一并发送给KDC。KDC使用自己的主密钥对TGT进行解密，提取客户端信息和SKDC-Client，然后使用SKDC-Client解密鉴别码，获得鉴别码中的客户端信息，对两个信息进行比较进而验证对方的真实身份。验证成功，生成一份基于Client所要访问的Server的票据给Client，之后的过程就像上面所讲述的一样进行了。

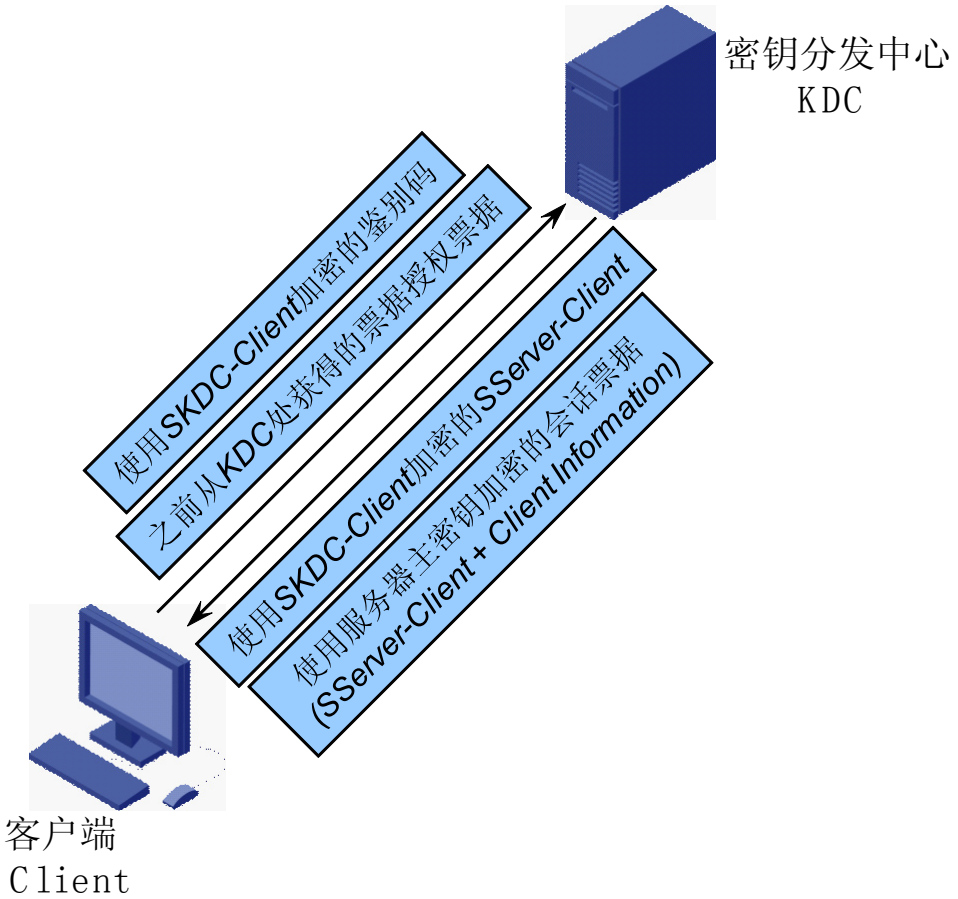


图6 会话票据分发过程

通过以上的介绍，Kerberos认证的过程已经基本介绍清楚了，首先Client向KDC申请TGT，之后Client通过获得TGT向KDC申请用于访问Server的票据，最后Client向Server提交票据用于认证。

▲Kerberos的子协议▲

上一节所讲述的Kerberos认证过程的三步是通过三个子协议来完成，它们分别是：Authentication Service Exchange、Ticket Granting Service Exchange和Client/Server Exchange。这三个子协议分别完成了Client向KDC申请TGT、Client通过获得的TGT向KDC申请用于访问Server的票据和Client向Server提交票据进行认证的三个过程。

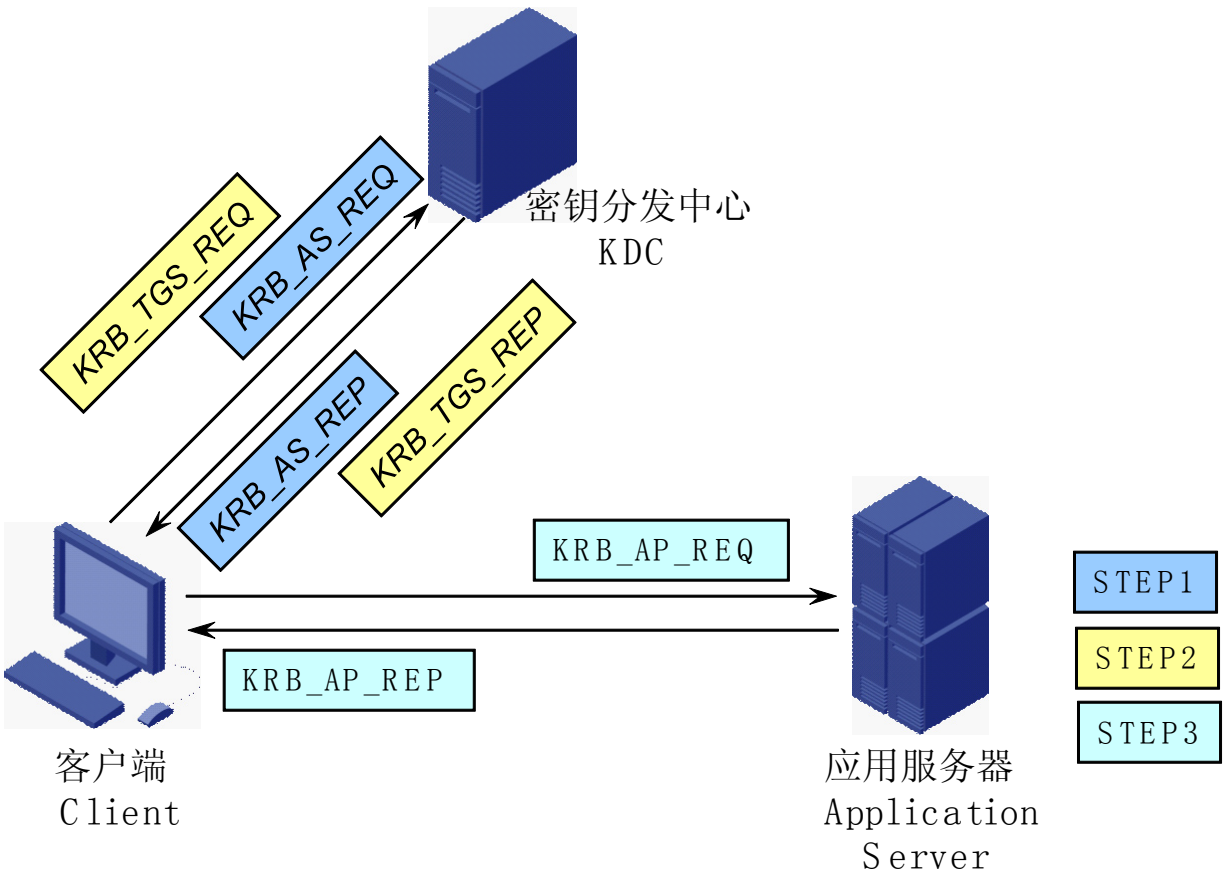


图7 Kerberos三步认证过程

Authentication Service Exchange

首先Client向KDC的认证服务发送Authentication Service Request（KRB_AS_REQ），大体内容包含：客户端名称、票据授权服务器名称以及一个被Client的主密钥加密的时间戳，我们可以把它看作是票据的有效期。

认证服务从数据库中提取KRB_AS_REQ中携带的客户端名称所对应的主密钥对被加密的时间戳进行解密，如果这个时间戳合法，则可以证明发送方提供的是正确的密码， 认证服务将一份包含使用Client的主密钥加密的会话密钥和被自己的主密钥加密的TGT的回应（KRB_AS_REP）发送给Client。这个TGT中包含了SKDC-Client、客户端名称和TGT到期的时间。

Client对第一部分解密获得SKDC-Client之后，携带着TGT便可以进入下一步——TGS（Ticket Granting Service）Exchange。

Ticket Granting Service Exchange

Client向KDC的票据授权服务发送Ticket Granting Service Request（KRB_TGS_REQ），内容包括Client通过AS Exchange获得的TGT、使用SKDC-Client加密的客户端信息以及所需要访问的服务器名称。

票据授权服务收到请求后，先得确认Client提供的那个TGT是否是AS颁发给它的。方法很简单，KDC先使用自己的主密钥对Client提供的TGT进行解密，获得SKDC-Client，再使用SKDC-Client解密鉴别码进行验证。验证通过向对方发送Ticket Granting Service Response（KRB_TGS_REP）。回应报文

也由两部分组成：使用SKDC-Client加密的Client和Server的会话密钥（SServer-Client）和使用Server的主密钥加密的票据，包含SServer-Client，客户端名称和票据的到期时间。

Client收到回应报文后，使用SKDC-Client解密第一部分，获得SServer-Client。有了SServer-Client和票据，Client就可以直接和Server进行交互，而无需再通过KDC了。

Client/Server Exchange

最后Client使用票据和Server进行交互就由CS（Client/Server）Exchange来完成。Client通过TGS Exchange获得SServer-Client，随后创建用于证明自己就是票据真正所有者的鉴别码，并使用SServer-Client进行加密。最后将这个被加密过的鉴别码和票据作为应用服务请求（KRB_AP_REQ）发送给Server，另外在请求中还包含一个标志位用于表示Client是否需要进行双向验证。

Server接收到请求之后，通过自己的主密钥解密票据，从而获得SServer-Client。接着使用SServer-Client解密鉴别码，进而验证Client的身份。如果验证通过，则让Client访问需要访问的资源，否则直接拒绝对方的请求。

如果Client需要进行双向验证，Server从鉴别码中提取时间戳，使用SServer-Client进行加密，并将其发送给Client用于Client验证Server的身份。

通过对上述三个子协议的介绍，我们已经了解了整个Kerberos的认证过程。但是细心的读者也许会发现，基于这三个子协议的Kerberos系统是存在一定的安全隐患的。在CS Exchange阶段，Client携带的票据是被Server的主密钥加密的，这显现不符合前面提出长期密钥的使用原则，降低了Server的安全系数。

最直接的解决方式就是采用一个会话密钥，而不是Server的主密钥对票据进行加密。这就是Kerberos的第四个子协议：User-to-User Authentication Exchanges。票据是由KDC生成之后通过Client发送给Server的，所以用于加密票据的自然是KDC和Server之间的会话密钥（SKDC-Server）。但是

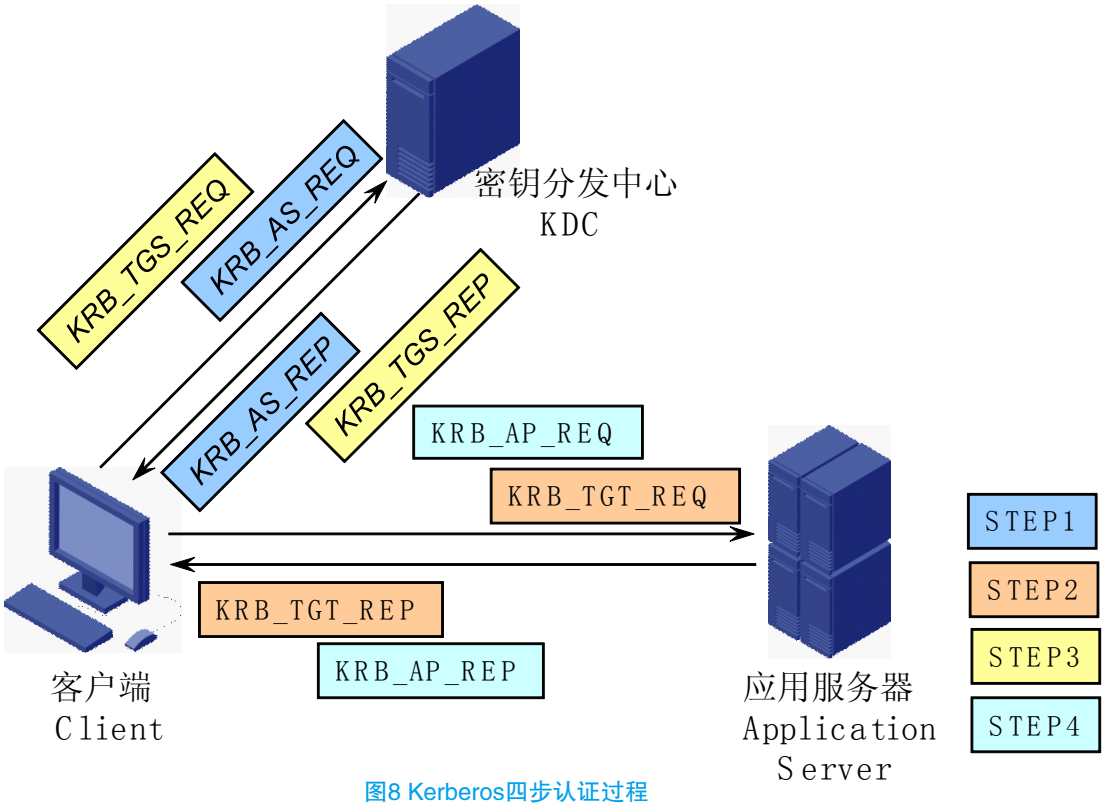


图8 Kerberos四步认证过程

KDC是不会维护任何会话密钥的，所以SKDC-Server只能靠申请票据的Client提供。

上图描述了包含User-to-User Authentication Exchanges的认证过程，这个过程由四步组成。较之前基于传统的认证过程多了第二步。

首先通过AS Exchange，Client获得了属于自己的TGT，便可凭此向KDC申请用于访问某个Server的票据。

第二步的主要任务是获得封装了SKDC-Server的属于Server的TGT。如果该TGT存在于Server的缓存中，则Server会直接将其返回给Client。否则Server将通过AS Exchange从KDC获取。

接着Client通过向KDC提供自己的TGT，Server的TGT以及鉴别码来申请用于访问Server的票据。KDC先用自己的主密钥解密Client的TGT获得SKDC-Client，通过SKDC-Client解密鉴别码验证发送者的身份，验证通过后再用自己的主密钥解密Server的TGT获得SKDC-Server，并用SKDC-Server加密票据返回给Client。

最后，Client将使用SKDC-Server加密的票据和SServer-Client加密的鉴别码访问Server，Server通过SKDC-Server解密票据获得SServer-Client，通过SServer-Client解密鉴别码实现对Client的验证。

▲小 结▲

在分析了整个Kerberos认证过程之后，Kerberos的优点也体现出来了。首先它具有较高的性能，一旦Client获得用于访问某个Server的票据，则该Server就能根据票据实现对Client的验证，不再需要KDC的参与；其次Kerberos可以进行双向验证，Client在访问Server的资源之前可以要求对Server的身份进行验证；第三就是互操作性，Kerberos最初由MIT提出并实现的，现在已经成为计算机领域一个被广泛接受的标准，所以使用Kerberos可以轻松实现不同平台之间的互操作。

但是Kerberos的缺点同样存在，比如Kerberos身份认证采用的是对称加密机制，加密和解密使用相同的密钥，安全性有所降低；Kerberos中身份认证服务和票据授权服务时集中式管理的，容易形成瓶颈，系统的性能和安全性也过分依赖于这两个服务的性能和安全。■

EAP技术简介

■ 文/袁江

EAP（Extensible Authentication Protocol），可扩展认证协议，是一种普遍使用的支持多种认证方法的认证框架协议，主要用于网络接入认证过程。

该协议一般运行在数据链路层上，即可以直接运行于PPP或者IEEE 802之上，不必依赖于IP。EAP可应用于无线、有线链路。

EAP的架构是灵活的，在Authenticator（认证者）和Peer（客户端）交互足够多的信息之后，才会决定选择一种具体的认证方法，即允许协商所希望的认证方法。Authenticator不用支持所有的认证方法，因为EAP架构允许使用一个后端的认证服务器，此时认证者将扮演一个透传的角色在客户端和服务器之间传递消息。

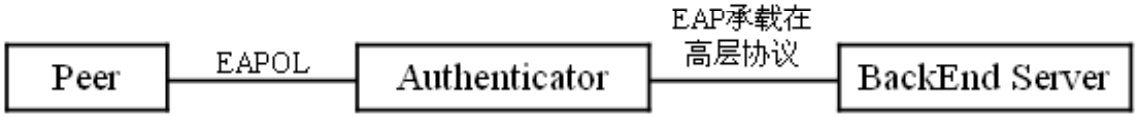


图1 EAP典型使用组网（Ethernet）

该协议支持重传机制，但这需要依赖于底层对报文顺序的保证，对于乱序报文的接收EAP是不支持的。协议本身不支持分片与重组，一些EAP认证方法生成大于MTU的数据时，该认证方法需要自己提供支持分片与重组。

EAP认证方法目前大约有40种，IETF的RFC中定义的方法包括：EAP-MD5, EAP-OTP, EAP-GTC, EAP-TLS, EAP-SIM和EAP-AKA, 还包括一些厂商提供的方法和新的建议。

▲EAP过程▲

EAP过程由Authenticator发起，而认证协议一般都是由客户端发起，为了在特定的认证协议上支持EAP过程，认证协议需要增加一个或多个的附加报文来满足着条件。如以太网上的认证通常情况下是以客户端发送EAPOL报文开始的。

EAP的典型过程如下：

1、Authenticator发送请求报文EAP-Request给Peer（客户端），表明EAP认证开始。该请求由一个字段标明需要请求的数据是什么。这个字段包含Identity（询问对端的身份），MD5-challenge（MD5挑

战字）等。上述发送EAP-Request的步骤在某些情况下也可省去，比如在有线网络中，Authenticator即NAS已经根据客户端连接的端口识别用户身份，或者通过MAC地址，IMSI卡等识别用户身份的情况下。

2、Peer针对Authenticator发过来的请求回应一个响应消息EAP-Response，该消息包含了请求消息中请求的字段信息，如身份识别信息。

3、Authenticator继续发送EAP-Request消息，消息中包含具体EAP Type及其协议数据，Peer对此做出响应。根据EAP Method的不同此步骤可能进行交互多次。

EAP交互过程是Peer与Authenticator一来一往的过程，每次只能处理一个EAP报文，在收到响应之前不能发送新的请求，即锁步机制（Lockstep）。因此Authenticator要负责请求消息的重传。如果一定数量的重传都没有收到应答消息，这次的EAP过程就要被终结。重传后收到应答消息或者一直没有收到应答消息，Authenticator都不发送成功或者失败消息。

4、交互多次之后，会出现两种情形：Authenticator不能确认Peer的接入权限，此时必须发送EAP-Failure，终结此次的EAP过程；或者确认接入权限，此时必须发送EAP-Success消息。

Authenticator也可以作为Pass Through设备，透传EAP报文。这种情形下Authenticator检查EAP的Code，Id及Length并将报文转发出去。Authenticator需要将EAP Code为2（Response）的报文转给后端的认证服务器，将EAP Code=1（Request），3（Success），4（Failure）的报文转给Peer。在这个处理过程中除非Pass Through Authenticator实现了某种EAP Methods，一般是只将消息转发，并不去检查EAP Methods Layer的的消息头（即EAP Type）。

▲EAP报文的封装格式▲

PPP链路

当PPP帧的协议字段是0xC227的时候，表示PPP帧的信息字段里封装了一个完整的EAP报文。

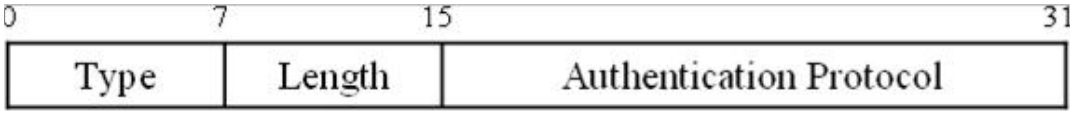


图2 EAP报文在PPP链路的封装

Type：一个字节，取值为3

Length：一个字节，取值为4

Authentication Protocol：两个字节，C227 (Hex) for Extensible Authentication Protocol (EAP)

IEEE 802/Ethernet：

EAP在LAN中的封装定义在IEEE-802.1X。

PAE Ethernet Type	
Protocol Version	Type
Length	
Packet Body	

图3 EAP在LAN中的封装

PAE（Port Access Entity）Ethernet Type：2字节，表示协议类型，为0x888E。

Protocol Version：1字节，表示EAPOL帧的发送方所支持的协议版本号。

Type：1字节，表示EAPOL数据帧类型，具体类型如表1：

表1 EAPOL类型表

类型	说明
EAP-Packet（值为0x00）：认证信息帧，用于承载认证信息	该类型的帧在设备端重新封装并承载于RADIUS等协议上，便于穿越复杂的网络到达认证服务器
EAPOL-Start（值为0x01）：认证发起帧	这两种类型的帧仅在客户端和设备端之间存在
EAPOL-Logoff（值为0x02）：退出请求帧	

Length：2字节，表示数据长度，也就是“Packet Body”字段的长度，单位为字节。如果为0，则表示没有后面的数据域。

EAPOL-Start和EAPOL-Logoff报文的length值都为0。

Packet Body：表示数据内容，根据不同的Type有不同的格式。

▲EAP报文格式▲

当EAPOL数据帧格式Type域为EAP-Packet时，Packet Body为EAP数据报文，结构如图4：

Code	Identifier
Length	
Data	

图4 EAP报文格式

Code：一个字节，指明EAP包的类型，共有4种，域值定义如下：

1--------Request

2--------Response

3--------Success

4--------Failure。

由于该字段值只定义了1到4，如果EAP报文的该字段为其他值，则应被Authenticator和Peer丢弃。

Identifier：占一个字节，用于应答报文和请求报文之间进行匹配。

Length：占两个字节，EAP包的长度，包含Code、Identifier、Length和Data域，单位为字节。

Data：占零个或多个字节，EAP包的内容，由Code类型决定

RFC 3748中定义的Success和Failure类型的包没有Data域，相应的Length域的值为4。但是某些软件的实现中，为了说明认证失败的原因，在Length域后面增加了字段用于说明下线的原因，故Length域的值可能为其他值。

Request和Response类型报文的Data域格式如下：

Type	Type Data

图5 Data域的格式

Type：占一个字节，标识EAP的认证类型

Type data：该字段的内容由Type字段的值决定。

Type字段目前定义的值及其简要说明如下：

Type = 1 ----Identifier（用来询问对端的身份）

Type = 2 ----Notification（非必须的一个消息，传送一些警告消息，比如提示密码将要超期、OTP的顺序号码接近零以及认证失败的警告等）。

Type = 3 ----Nak (Response Only)（Request报文中的认证类型不可接受时回复该类型的报文）

Type = 4 ----MD5-Challenge（类似于CHAP中的MD5-Challenge，使用MD5算法）

Type = 5 ----One-Time Password (OTP)（一种密码交互的方式）

Type = 6 ----Generic Token Card（通用令牌卡类型，适用于各种需要用户输入信息的令牌卡的实现）

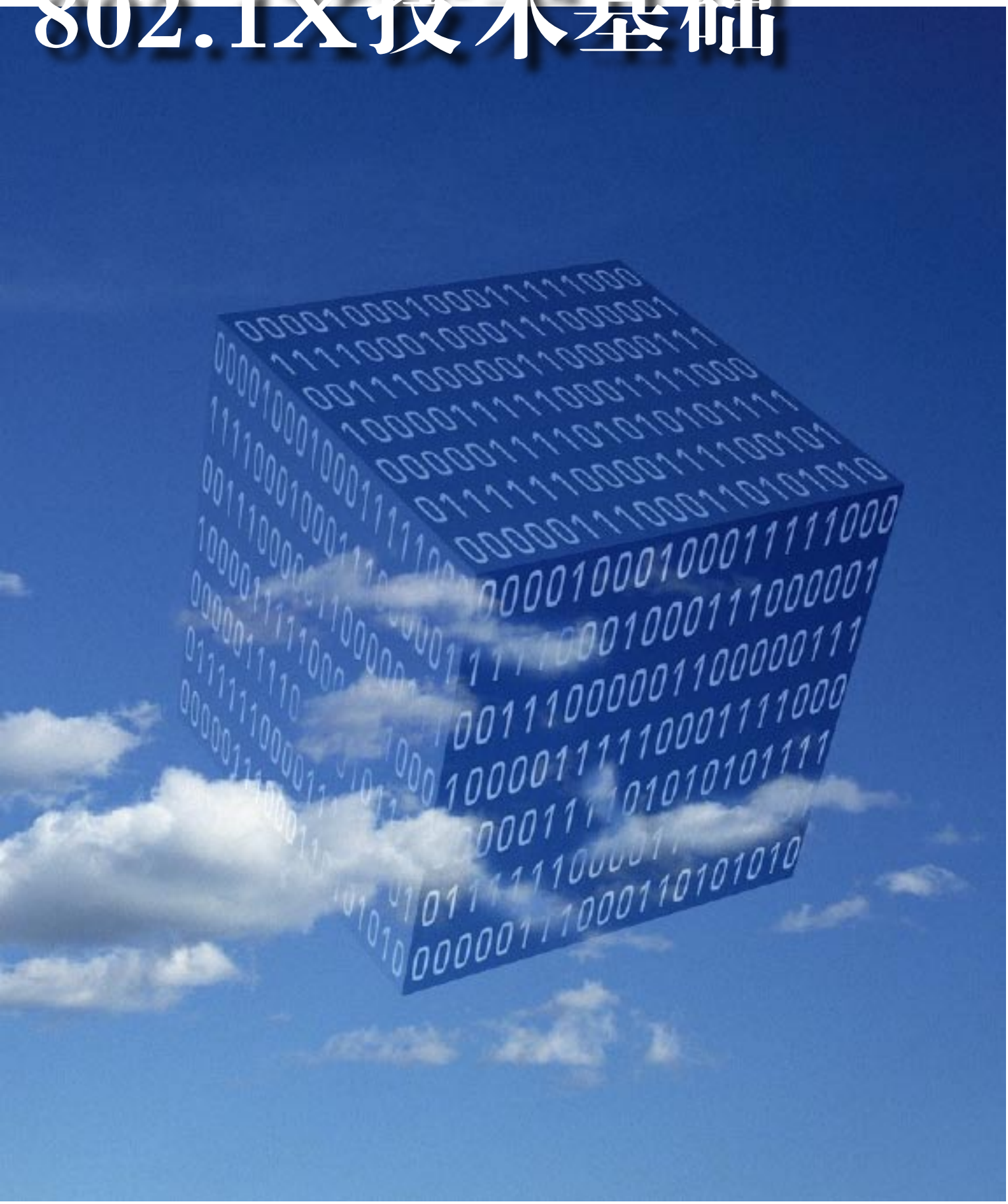
Type = 254 ----Expanded Types（支持厂商自己扩展的类型）

Type = 255 ----Experimental use（实验新的类型时做测试用的类型）

每种Type对应的Type Data字段这里不作详细描述，有兴趣的读者请查阅RFC 3748相关章节。■

■文/盛冬旭

802.1X技术基础



前言

802.1X作为一种基于端口的用户访问控制安全机制，由于其低成本、良好的业务连续性和扩充性以及较高的安全性和灵活性，自从2001年6月正式成为IEEE 802系列标准开始便迅速受到了设备制造商、各大网络运营商和最终用户的广泛支持和肯定。

那么802.1X从何而来？有什么作用？体系结构是怎么样的？用到什么协议？并且又是如何实现的？这些都将是我们本文讨论的范围。

802.1X起源

在802.1X出现之前，企业网上有线LAN应用都没有直接控制到端口的方法，因此只要有物理连接，用户就可以访问局域网中的任何资源。但是随着无线LAN的应用以及LAN接入在电信网上大规模开展，有必要对端口加以控制，以实现用户级的接入控制。

802.1X就是IEEE为了解决基于端口的接入控制而定义的一个标准。802.1X，全称是Port-Based Networks Access Control，即基于端口的网络访问控制，它起源于802.11协议，802.11协议是标准的无线局域网协议，802.1X协议最初的目的是为了解决无线局域网用户的接入认证问题，但由于它的原理对于所有符合IEEE 802标准的局域网具有普适性，因此后来它在有线局域网中也得到了广泛的应用。

现在802.1X协议作为局域网端口的一个普通接入控制机制在以太网中被广泛应用，主要解决以太网内认证和安全方面的问题。

体系结构

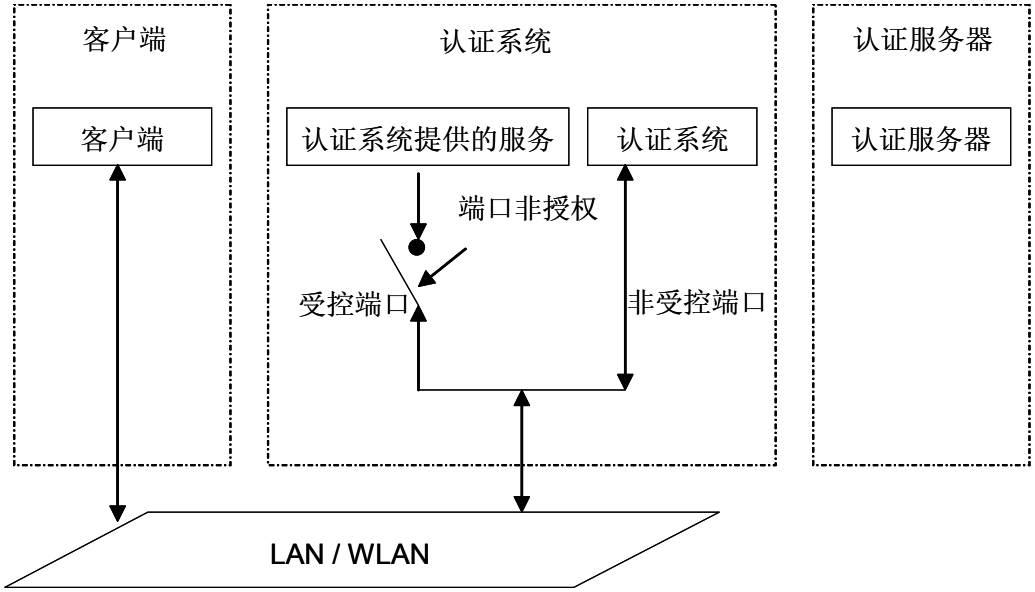


图1 802.1X协议的体系结构

IEEE 802.1X协议采用典型的客户端/服务器体系结构，包括三个主要的部分：客户端（Supplicant System）、认证系统（Authenticator System）以及认证服务器（Authentication Server System）。图1描述了三者之间的关系以及互相之间的通信。

客户端系统一般为一个用户终端系统，该终端系统通常要安装一个客户端软件，用户通过启动这个客户端软件发起802.1X协议的认证过程。为支持基于端口的接入控制，客户端系统必须支持EAPOL（Extensible Authentication Protocol Over LAN，局域网上的可扩展认证协议）。

认证系统也称NAS（Network Access System）,通常为支持802.1X协议的网络设备。在启动802.1X认证下，该设备每个物理端口对应两个逻辑端口：受控端口（Controlled Port）和非受控端口（Uncontrolled Port）。非受控端口始终处于双向连通状态，主要用来传递 EAPOL协议帧，可保证客户端始终可以发出或接收认证信息。而受控端口只有在认证通过的状态下才打开，用于传递业务信息帧。为适应不同的应用环境，受控端口可配置为单向受控和双向受控两种方式。实行双向受控时，禁止帧的发送和接收；实行单向受控时，禁止从客户端接收帧，但允许向客户端发送帧。目前，H3C公司设备只支持单向受控。如果用户未通过认证，则受控端口处于未认证状态，用户无法访问NAS提供的服务。图1中认证系统的受控端口处于未认证状态，因此无法访问认证系统提供的服务。

认证服务器通常为RADIUS服务器，该服务器可以存储有关用户的认证、计费、业务信息，比如用户所属的VLAN、CAR参数、优先级、用户的访问控制列表等等。这里认证服务器所提供的服务是指通过检验客户端发送来的身份标识，来判断该请求者是否有权使用认证系统所提供的网络服务。当用户通过认证后，认证服务器会把用户的相关信息传递给NAS，由NAS构建动态的访问控制列表，用户的后续流量就将接受上述参数的监管。本文中提到的认证服务器均以RADIUS服务器为例进行说明。

以上三个部分，除客户端系统外，认证系统和认证服务器间既可以分布在两个不同的物理实体上，也可以是集中在一个物理实体上，即由NAS独立完成用户的认证授权（本地认证）。

协议实现及报文格式

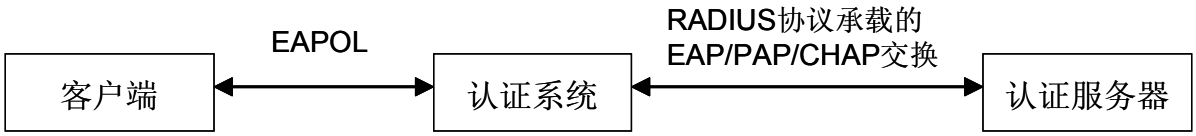


图2 802.1X协议实现

如图2所示，802.1X协议在实现整个安全认证的过程中，其三个关键部分客户端、认证系统（NAS）、认证服务器之间是通过通信协议进行交互的，因此有必要对其相关的通信协议做个介绍。

EAPOL协议

EAPOL（EAP over LANs）是802.1X协议定义的一种报文封装格式，主要在客户端和NAS之间传送EAP协议报文，可以允许EAP协议报文直接承载于LAN环境中。

EAPOL消息封装格式如图3所示：

	Octet Number
PAE Ethernet Type	1-2
Protocol Version	3
Packet Type	4
Packet Body Length	5-6
Packet Body	7-N

图3 EAPOL的帧结构

PAE Ethernet Type：2字节，该值表示协议类型，802.1X分配的协议类型为0x888E。

Protocol Version：1字节，表示EAPOL帧的发送方所支持的协议版本号。

Packet Type：1字节，表示EAPOL数据帧类型，有如下三种帧类型，如图4所示。

Packet Body Length：2字节，表示Packet Body数据段的长度，单位为字节。如果为0，则表示没有后面的数据域。且EAPOL-Start和EAPOL-Logoff报文的length值都为0。

Packet Body：表示数据内容，根据不同的Type有不同的格式。

类型	说明
EAP-Packet（值为0x00）：认证信息帧，用于承载认证信息	该帧在设备端重新封装并承载于RADIUS协议上，便于穿越复杂的网络到达认证服务器
EAPOL-Start（值为0x01）：认证发起帧	这两种类型的帧仅在客户端和设备端之间存在
EAPOL-Logoff（值为0x02）：退出请求帧	

图4 Packet Body类型及说明

EAP协议

802.1X协议采用EAP（Extensible Authentication Protocol，可扩展认证协议），来实现客户端、NAS和认证服务器之间认证信息的交互。

通过支持EAP协议，认证系统只需控制其受控端口的状态，但是并不干涉通过非受控端口在客户端和认证服务器之间传递的认证信息。这样，就实现了认证流和业务流的完全分离。可以使用认证服务器来实现各种认证机制，NAS仅仅需要传送认证信息，并根据认证返回的结果控制受控端口的状态。

当EAPOL数据包格式Type域为EAP-Packet时，Packet Body为EAP数据包。EAP数据包结构如图5所示：

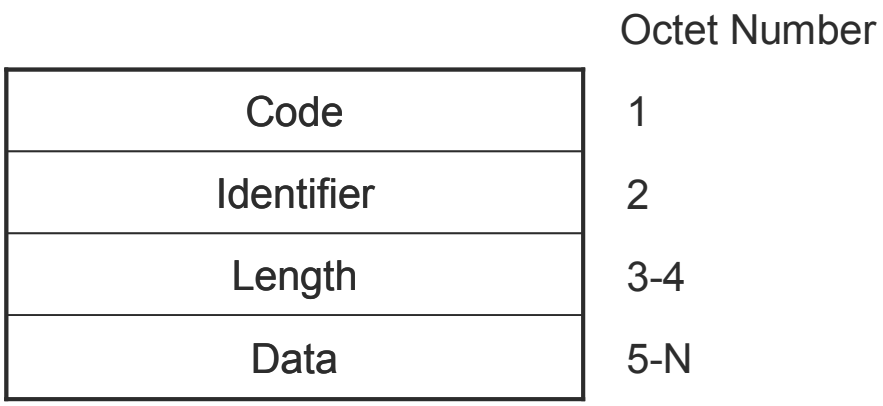


图5 EAP的帧结构

Code：1字节，该值表示EAP帧类型，共有4种：Request、Response、Success、Failure。

Identifier：1字节，该值用于匹配Request消息和Response消息。Identifier字段和系统端口一起唯一标识一个认证过程。

Length：2字节，该值表示EAP帧的总长度，包含Code、Identifier、Length和Data域，单位为字节。

Data：0或更多字节，表示EAP包的内容，由Code类型决定。

对于Code类型不同的报文，Data域的格式也略有不同：

若Code类型为Request或Response的数据包的Data域格式如图6所示。Type为EAP的认证类型，Type data的内容由类型决定。

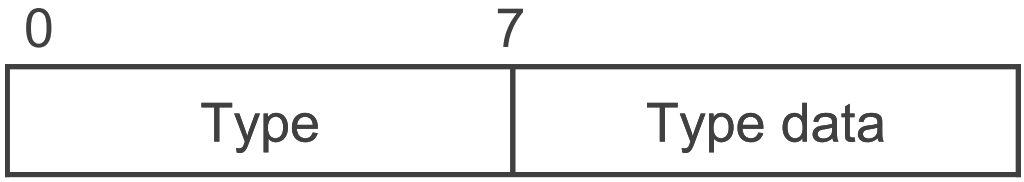


图6 Type报文类型

此时，Type报文类型包括以下常见几种：

Type = 1 – Identifier（要求客户端程序发送用户输入的用户名信息）

Type = 2 – Notification（非必须的一个消息，传送一些警告消息）

Type = 3 – Response Only（Request报文中的认证类型不可接受时回应该类型的报文）

Type = 4 – MD5–Challenge（类似于CHAP中的MD5–Challenge）

Type = 5 – One–Time Password (OTP)（一种密码交互的方式）

Type = 6 – Generic Token Card（支持Challenge和Response时使用令牌）

Type = 254 – Expanded Types（支持厂商自己扩展的类型）

Type = 255 – Experimental use（实验新的类型时做测试用的类型）

若Code类型为标准的Success或Failure的数据包，则没有Data域，相应的Length域的值为4。在H3C公司的实现中，在Length域后面增加了字段用于说明下线的原因，故Length域的值可能为其他值。

此外，RADIUS为支持EAP认证增加了两个属性：EAP-Message（EAP消息）和Message-Authenticator（消息认证码）。

如图7所示，EAP-Message属性用来封装EAP数据包，类型代码为79，Value域最长253字节，如果EAP数据包长度大于253字节，可以对其进行分片，依次封装在多个EAP-Message属性中。该属性必须配合Message-Authenticator属性使用。

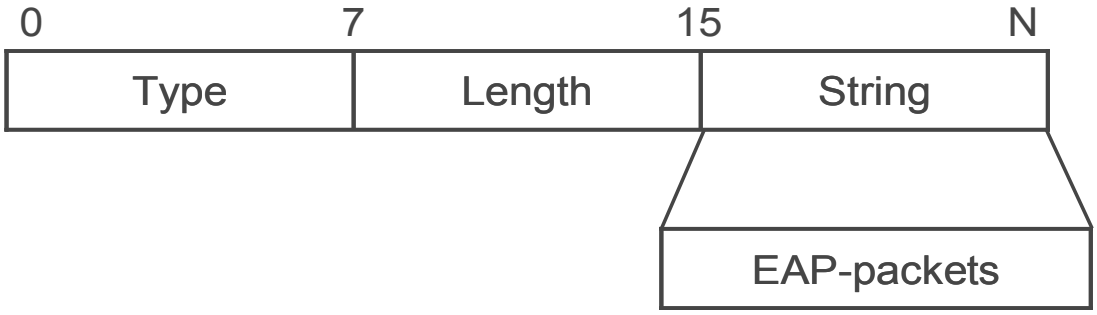


图7 EAP-Message属性

而上面提到的Message-Authenticator属性，如图8所示，类型代码为80，主要用于在EAP认证过程中验证携带了EAP-Message属性的RADIUS报文的完整性，避免接入请求包被窜改。在含有EAP-Message属性的数据包中，必须同时也包含Message-Authenticator，否则该数据包会被认为无效而被丢弃。此外，若接收端对接收到的RADIUS报文计算出的完整性校验值与报文中携带的Message-Authenticator属性的Value值不一致，该数据包也会被认为无效而丢弃。

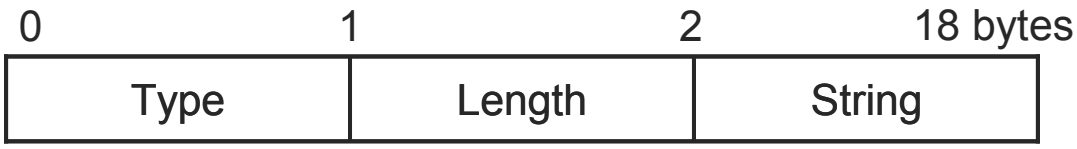


图8 Message-Authenticator属性

认证的基本过程

认证发起

802.1X的认证发起可以由NAS发起，也可以由客户端主动发起。当NAS探测到未经过认证的用户使用网络，就会主动发起认证；客户端则可以通过客户端软件向认证系统发送EAPOL-Start报文发起认证。

- NAS主动触发认证方式

当NAS检测到有未经认证的用户使用网络时，会每隔N秒（系统默认30秒）主动向客户端以组播报文来触发认证。在认证开始之前，端口的状态被强制为未认证状态。

如果客户端的身份标识不可知，则NAS会发送EAP-Request/Identity报文，请求客户端发送身份标

识。这样，就开始了典型的认证过程。且客户端在收到来自NAS的EAP-Request报文后，将发送EAP-Response报文响应NAS的请求。

这种触发方式用于支持不能主动发送EAPOL-Start报文的客户端，例如Windows XP自带的802.1X客户端。

- 客户端主动触发认证方式

如果用户要上网，则可以通过客户端软件主动发起认证。客户端软件会向NAS发送EAPOL-Start报文主动发起认证。该报文目的地址为IEEE 802.1X协议分配的一个组播MAC地址：01-80-C2-00-00-03。

NAS在收到客户端发送的EAPOL-Start报文后，会发送EAP-Request/Identity报文响应用户请求，要求用户发送身份标识，这样就启动了一个认证过程。

此外，由于网络中有些设备不支持上述的组播报文，使得NAS无法收到客户端的认证请求，因此NAS还支持广播触发方式，即可以接收客户端发送的目的地址为广播MAC地址的EAPOL-Start报文。这种触发方式需要与H3C公司的iNode客户端配合使用。

退出认证

通过以下几种方式NAS设备可以将端口状态从已认证状态改变成未认证状态：

- 客户端未通过认证服务器的认证；
- 由于管理性的控制端口始终处于未认证状态，而不管是否通过认证；
- 与端口对应的MAC地址出现故障（管理性禁止或硬件故障）；
- 客户端与NAS之间的连接失败，造成认证超时；
- 客户端未响应NAS发起的认证请求；
- 客户端发送EAPOL-Logoff报文，主动下线。

退出已认证状态的直接结果就是导致用户下线，如果用户要继续上网则需要再发起一个认证过程。

为什么要专门提供一个EAPOL-Logoff机制，主要是出于如下安全的考虑：

当一个用户从一台终端登录后，很可能其他用户不通过发起一个新的认证请求，就可以利用该设备访问网络。提供专门的退出机制，以确保用户与认证系统专有的会话进程被中止，可以防止用户的访问权限被他人盗用。通过发送EAPOL-Logoff报文，可以使认证系统将对应的端口状态改变为未认证状态。

认证过程

当用户通过认证后，认证服务器会把用户的相关信息传递给NAS，NAS会根据RADIUS服务器的指示，如Accept或Reject，来决定受控端口的授权/非授权状态。

当前802.1X系统支持两种方式与远端RADIUS服务器交互完成认证，即EAP中继方式和EAP终结方式。

EAP中继方式

EAP中继方式是IEEE 802.1X标准规定的，将EAP承载在其它高层协议中，如EAPOR(EAP over RADIUS)，以便EAP报文穿越复杂的网络到达认证服务器。一般来说，EAP中继方式需要RADIUS服务器

支持EAP属性：EAP-Message（值为79）和Message-Authenticator（值为80），以上两个属性分别用来封装EAP报文及对携带EAP-Message的RADIUS报文进行保护。

在具有802.1X认证功能的以太网络系统中，当一个用户需要对网络资源进行访问之前必须先要完成以下的认证过程。下面就以EAP-MD5方式为例，介绍802.1X基本认证流程，如图9所示。其中EAP-MD5是一种单向认证机制，可以完成网络对用户的认证，但认证过程不支持加密密钥的生成。

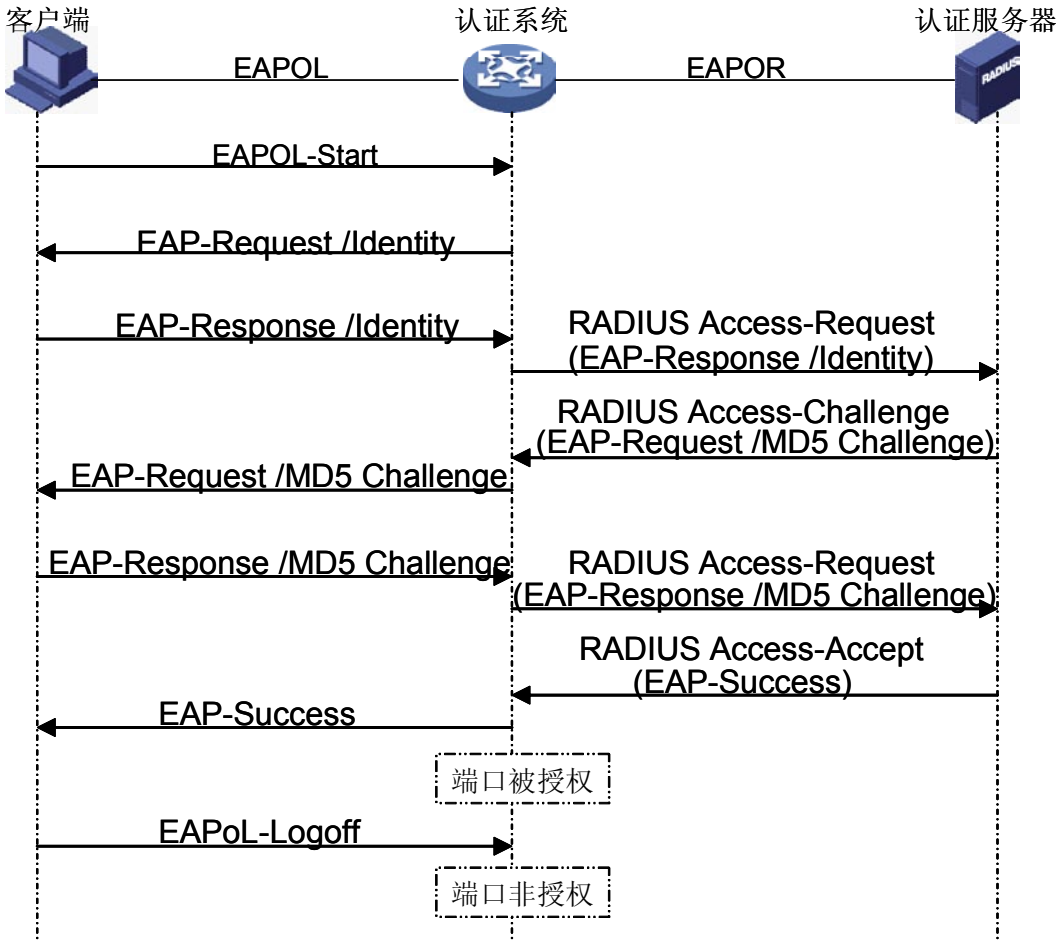


图9 IEEE 802.1X认证系统的EAP中继方式交互流程

1. 当用户有网络连接需求时打开802.1X客户端程序，输入已经申请、登记过的用户名和口令，发起连接请求。此时，客户端程序将发出请求认证的报文给NAS，开始启动一次认证过程。

2. NAS收到请求认证的数据帧后，将发出一个请求帧要求用户的客户端程序将输入的用户名送上来。

3. 客户端程序响应NAS发出的请求，将用户名信息通过数据帧送给NAS。NAS将客户端送上来的数据帧经过封包处理后送给认证服务器进行处理。

4. 认证服务器收到NAS转发上来的用户名信息后，将该信息与数据库中的用户名表相比对，找到该用户名对应的口令信息，用随机生成的一个加密字对它进行加密处理，同时也将此加密字传送给NAS，由NAS传给客户端程序。

5. 客户端程序收到由NAS传来的加密字后，用该加密字对口令部分进行加密处理（此种加密算法通

常是不可逆的），并通过NAS传给认证服务器。

6. 认证服务器将送上来的加密后的口令信息和其自己经过加密运算后的口令信息进行对比，如果相同，则认为该用户为合法用户，反馈认证通过的消息，并向NAS发出打开端口的指令，允许用户的业务流通过端口访问网络。否则，反馈认证失败的消息，并保持NAS端口的关闭状态，只允许认证信息数据通过而不允许业务数据通过。

7. 用户访问网络资源完毕后，可以通过客户端发送报文给NAS，主动终止已认证状态，NAS将端口状态从授权状态改变成未授权状态。

这里要提出的一个值得注意的地方是：在客户端与认证服务器交换口令信息的时候，没有将口令以明文直接送到网络上进行传输，而是对口令信息进行了不可逆的加密算法处理，使在网络上传输的敏感信息有了更高的安全保障，杜绝了由于下级接入设备所具有的广播特性而导致敏感信息泄漏的问题。

EAP终结方式

EAP终结方式将EAP报文在设备端终结并映射到RADIUS报文中，通过标准的RADIUS协议完成认证、授权和计费。设备端与RADIUS服务器之间可以采用PAP（Password Authentication Protocol，密码验证协议）或者CHAP（Challenge Handshake Authentication Protocol，质询握手验证协议）认证方法。二者主要区别是CHAP密码是通过密文方式在客户端和NAS之间传输，而PAP密码通过明文的方式传输的。以下以CHAP认证方式为例介绍认证流程，如图10所示。

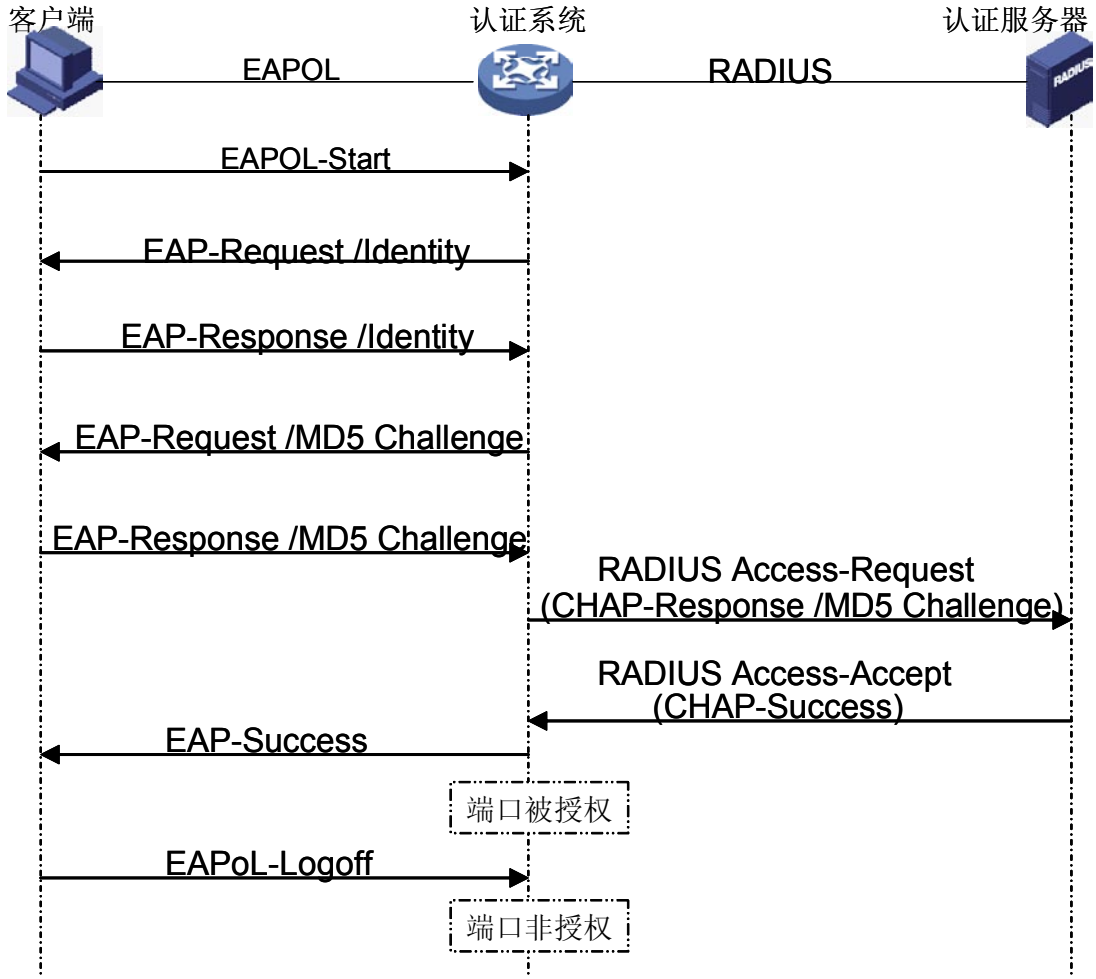


图10 IEEE 802.1X认证系统的EAP终结方式交互流程

EAP终结方式与EAP中继方式的认证流程相比，基本流程类似，不同之处主要在于加密字的生成处理及传递方式。在EAP终结方式中，用来对用户密码信息进行加密处理的随机加密字由设备端生成，设备端会把用户名、随机加密字和客户端加密后的密码信息一起送给RADIUS服务器，进行相关的认证处理；而在EAP中继方式中，用来对用户口令信息进行加密处理的随机加密字由RADIUS服务器生成，交换机只是负责将EAP报文透传RADIUS服务器，整个认证处理都由RADIUS服务器来完成。

两种方式对比

除了上面提到的认证流程中对于加密字的生成处理及传递方式不同外，EAP终结与EAP中继两种认证方式其优缺点也很明显：

EAP中继方式优点是认证系统设备处理更简单，支持更多的认证方式，缺点则是认证服务器必须支持EAP；而EAP终结方式的优点是认证服务器无需升级，现有的RADIUS服务器可以继续使用；缺点是认证系统设备处理更复杂。

两种方式各有优缺点，企业可根据实际情况自行选择。若为了兼容RADIUS服务器，节省用户投资，建议认证系统选择EAP终结方式。目前H3C公司的交换机产品均支持这两种认证方式，在默认情况下，选择EAP终结方式。

H3C公司对802.1X的扩展属性

目前802.1X技术在宽带城域网和园区网中获得了大量的应用。在现在竞争激烈的宽带网络建设中，基于业务类似的情况下，802.1X及其扩展业务特性往往成为关键点。H3C公司的ComWare平台的802.1X业务除兼容标准外，根据需要进行了多项扩充。

目前H3C公司的交换机产品均支持IEEE802.1X认证技术，并在严格遵守协议标准的同时加入独特的实现方式，提高了灵活性。

支持基于端口和基于MAC的不同认证方式

IEEE 802.1X的标准是一个基于端口的访问控制协议，其基本要求是对用户接入端口进行控制。这对无线接入访问点而言是足够的，因为一个用户占用一个信道。但在电信接入和学校等环境中，以端口为对象的控制粒度难以满足要求，基于MAC地址逻辑端口控制势在必行。H3C公司的802.1X特性不仅提供基于端口（Portbased）的用户访问控制，而且提供基于用户MAC地址（MACbased）的访问控制。

基于端口的认证方式下，一旦有一个用户通过认证，端口就处于打开状态，任何其他用户都可以通过该端口访问网络。

基于MAC的认证方式下，H3C公司的实现是通过MAC表项中添加/删除用户主机MAC的方式来实现的，通过认证的用户MAC将被添加到MAC表项中，这样用户就具备权限访问网络了。反之，没有通过认证的用户主机MAC是不会出现在MAC表项中的，因此也没有访问网络的权限。

灵活的用户业务下发

目前H3C公司设备支持的下发业务包括VLAN业务、ACL业务和CAR业务等。

VLAN业务下发是指，在用户认证过程中，通过与RADIUS服务器配合，支持给用户重新指定VLAN并下发到NAS。在LAN Switch中，VLAN与路由虚接口有映射关系，因此可以根据这一特性来规划某类用户的管理策略。例如，校园网中，给学生群体分配VLAN 10，教师群体分配VLAN20。通过分别给VLAN 10和20虚接口所在路由段指定不同的访问控制策略可增强可管理性。同时，由于VLAN资源是动态下发的，因此用户可以在园区网中漫游而不影响访问。

而ACL业务下发与VLAN业务下发不同，它为用户访问网络提供了良好的过滤条件设置功能。802.1X认证支持认证服务器授权下发ACL功能，即当用户通过认证后，如果认证服务器上配置了授权ACL，则设备会根据服务器下发的授权ACL对用户所在端口的数据流进行控制。为使下发的授权ACL生效，需要提前在设备上配置相应的ACL规则。而且在用户访问网络的过程中，可以通过改变服务器的授权ACL设置来改变用户的访问权限。

CAR业务即用户流量控制业务。目的是根据资费情况动态设定用户的上下行流量参数。这一业务除了需要RADIUS服务器配合外，还需要NAS本身支持流量控制功能。

定期握手机制取代了重新认证机制

重认证机制是IEEE 802.1X协议中建议的一种及时发现用户意外下线的机制，H3C公司在IEEE 802.1X实现中对这部分机制进行了优化创新，将重认证机制用定期握手机制取代。定期握手机制是NAS定期发送握手请求报文给在线用户，客户端程序收到后，回复响应报文，NAS以此认为用户在线。若在规定期限内，如3个定期握手计时器超时，没有收到客户端的回应，NAS就认为用户已下线，进而中断连接，完成计费等操作。采用定期握手机制能够很好的发现用户意外下线的情况，杜绝了由于用户的意外下线而导致的计费误差问题等。提高计费信息的可靠性，防止了用户“永远在线”。

独特的代理用户检测

在宽带园区网环境中，特别是校园网中，用户通过Proxy上网来实现“一个账号多人使用”的方法较普遍，引起了运营商的强烈关注。通过Proxy可以隐藏真实身份并能访问网络资源。

交换机的802.1X代理用户检测特性主要检测三类特殊用户：其一检测使用代理服务器登录的用户；其二检测使用IE代理服务器登录的用户；其三则检测用户是否使用多网卡，即用户登录时，其PC上处于激活状态的网卡超过一个。

根据客户端检测的结果，应答的结果有两种：正常上网或者通过Proxy上网。如果检测结果是正常方式上网，则设备继续与客户端握手，并保持客户端在线状态；如果结果为Proxy上网，且NAS发现了以上提到的任意一种情况时，则可以采取两种不同措施：一种是只发送下线消息Logoff切断用户连接，拒绝用户上网，不发送Trap报文。另一种则是与之相反，即不切断用户连接，保持用户在线，但会根据设备端的配置给网管报送Trap信息。

该功能的实现需要NAS与H3C公司的802.1X客户端程序iNode配合使用，实现对Proxy的检测，提高了网络的可运营性。

综 述

802.1X认证系统提供了一种用户接入认证的手段，它仅关注端口的打开与关闭。对于合法用户接入

时，该端口打开，而对于非法用户接入或没有用户接入时，则使端口处于关闭状态。认证的结果在于端口状态的改变，而不涉及其它认证技术所考虑的IP地址协商和分配问题，是各种身份认证技术中最为简化的实现方案。

在采用802.1X认证技术时，必须考虑到它的操作颗粒度为端口，合法用户接入端口之后，端口始终处于打开状态，此时其它用户(合法或非法)通过该端口接入时，不需认证即可访问网络资源。对于无线局域网接入而言，认证之后建立起来的信道(端口)被独占，不存在其它用户非法使用的问题。但如果802.IX认证技术应用于宽带IP城域网，就存在端口打开之后，其它用户(合法或非法)可自由接入且难以控制的问题。因此，在提出可运营、可管理要求的宽带IP城域网中如何使用该认证技术，还需要谨慎分析所适用的场合，并考虑与其它信息绑定组合认证的可能性。■

Portal 概述

■ 文/李朝辉

PORTAL概述

Portal在英语中是入口的意思。Portal认证通常也称为Web认证，一般将Portal认证网站称为门户网站。它提供了一种较为简单的用户认证方法，对用户而言，相对其它认证方式易于使用，其中有两大特色：

- 免客户端；

免客户端软件对于像宾馆、酒店等公共网络节点，免客户端软件是一个基本要求。

- 新业务载体；

利用PORTAL认证的门户功能，运营商可以将小区广播、广告、信息查询、网上购物等业务放到PORTAL上。用户上网时会强制地看到上述信息。

Portal认证的基本方式是通过在Portal页面的显著位置设置认证窗口，用户开机获取IP地址后通过登陆Portal认证页面进行认证，认证通过后即可访问Internet。

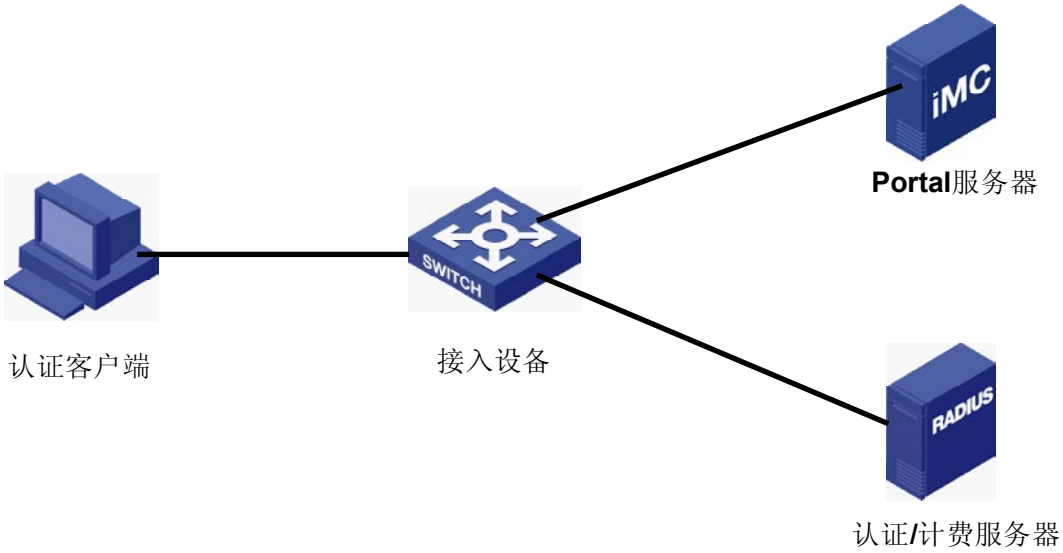
对于用户来说有两种方式访问认证页面：

- 主动Portal：用户必须知道PORTAL服务器的IP地址，主动登陆PORTAL服务器进行认证，之后才能访问网络。
- 强制Portal：未认证用户访问网址，都会先强制定向到PORTAL服务器进行认证，用户不需要记忆Portal服务器的IP地址。

Portal业务可以为运营商提供方便的管理功能，基于其门户网站可以开展广告、社区服务、个性化的业务等，使宽带运营商、设备提供商和内容服务提供商形成一个产业生态系统。

PORTAL系统组成

Portal的四大主要系统



● 认证客户端

安装于用户终端的客户端系统，为运行HTTP/HTTPS协议的浏览器或运行Portal客户端软件的主机。对接入终端的安全性检测是通过Portal客户端和安全策略服务器之间的信息交流完成的。

● 接入设备

交换机、路由器等宽带接入设备的统称，主要有三方面的作用：

在认证之前，将用户的所有HTTP请求都重定向到Portal服务器。

在认证过程中，与Portal服务器、安全策略服务器、认证/计费服务器交互，完成身份认证/安全认证/计费的功能。

在认证通过后，允许用户访问被管理员授权的互联网资源。

● Portal服务器

接收Portal客户端认证请求的服务器端系统，提供免费门户服务和基于Web认证的界面，与接入设备交互认证客户端的认证信息。

● 认证/计费服务器

与接入设备进行交互，完成对用户的认证和计费。

以上四个基本要素的交互过程为：

1. 未认证用户访问网络时，在Web浏览器地址栏中输入一个互联网的地址，那么此HTTP请求在经过接入设备时会被重定向到Portal服务器的Web认证主页上；

2. 用户在认证主页/认证对话框中输入认证信息后提交，Portal服务器会将用户的认证信息传递给接入设备；

3. 然后接入设备再与认证/计费服务器通信进行认证和计费；

4. 认证通过后，如果未对用户采用EAD安全策略，则接入设备会打开用户与互联网的通路，允许用户访问互联网；

Portal认证的实现机制

发起认证的方式

Portal虽然是可实现免客户端认证，但也有专用的客户端用来实现更安全灵活的功能，这两种情况的认证方法大致如下：

对于通过WEB进行认证的用户，采用对HTTP报文重定向的方式，BAS对用户连接进行TCP仿冒和Portal-Client建立TCP连接，然后将页面重定向到Portal-Server，而从实现向客户推出认证页面。用户通过该页面将用户信息传递给了Portal-Server，随后Portal-Server向BAS通过PAP或CHAP的方式传递用户信息。BAS获取到用户信息后，将该信息通过AAA模块完成认证。

对于使用客户端进行认证的用户，直接使用portal协议报文与portal-server进行交互，实现对客户端的相关控制和用户状态的实时上报。随后PortalServer与BAS交互，BAS再通过AAA模块完成认证。

用户保活机制

用户使用专用的客户端时，使用Portal握手报文来确认用户是否在线。对于客户端来，4个心跳没有收到答复，就认为自己已经下线，重新发起认证；对于Portal-Server，在指定的时间内没有收到心跳报文，就认为用户下线，并通知BAS。

用户通过WEB实现认证时，认证后在线窗口处于开打状态，并采用上层的http协议的get动作实现心跳机制，用户下线时需要在该页面上主动点击下线按钮触发下线动作，如果该页面或用户PC不正常关闭，会导致用户无法手动下线。

产品实现原理

产品对Portal的实现是基于ACL的，通过QACL模块来支持对用户报文的重定向以及限制用户可以使用的相关资源;通常我们把Portal使用的ACL分为4类(对于底层没有什么区别，主要是查找匹配的顺序)

Type1：FreeIP规则，动作是permit(到Portal-Server的规则为第1个freeip)

Type2：用户认证通过后添加的规则，动作是permit

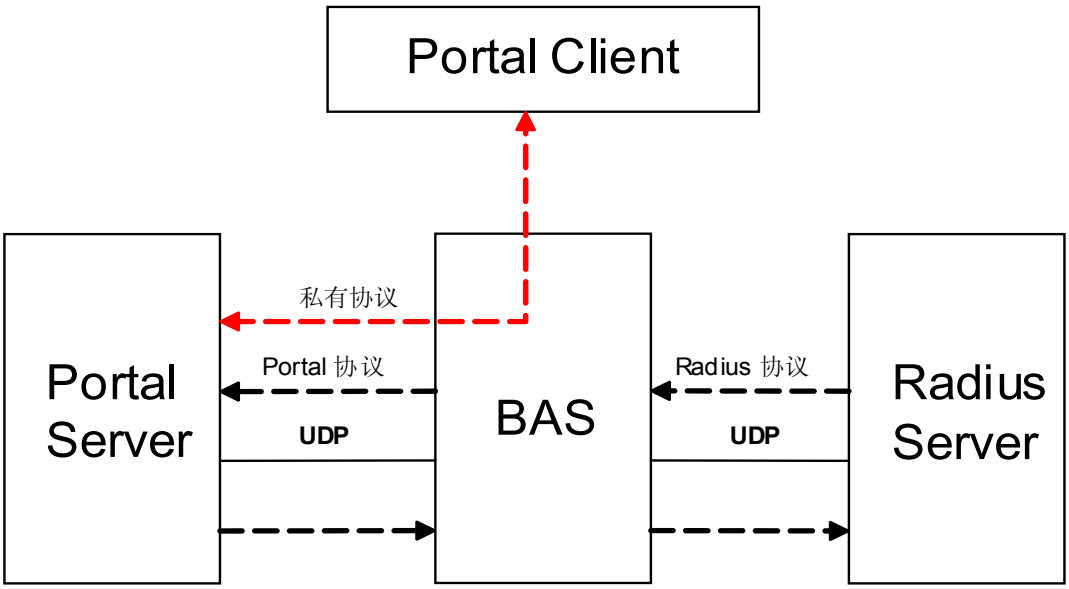
Type3：用户网段重定向规则，对HTTP报文重定向到CPU(实现认证页面的推出

Type4：用户网段禁止规则，动作是deny

Portal规则在端口上下发，排列需要有严格的顺序，匹配时按照Type1-4的顺序从前往后排列。如果在一个端口上既有普通ACL规则（通过命令行配置的ACL）下发，又启用了portal，则portal所添加的规则会排列在普通ACL之后。

PORTAL协议框架

Portal协议主要涉及PortalServer和BAS，采用C/S结构，基于UDP



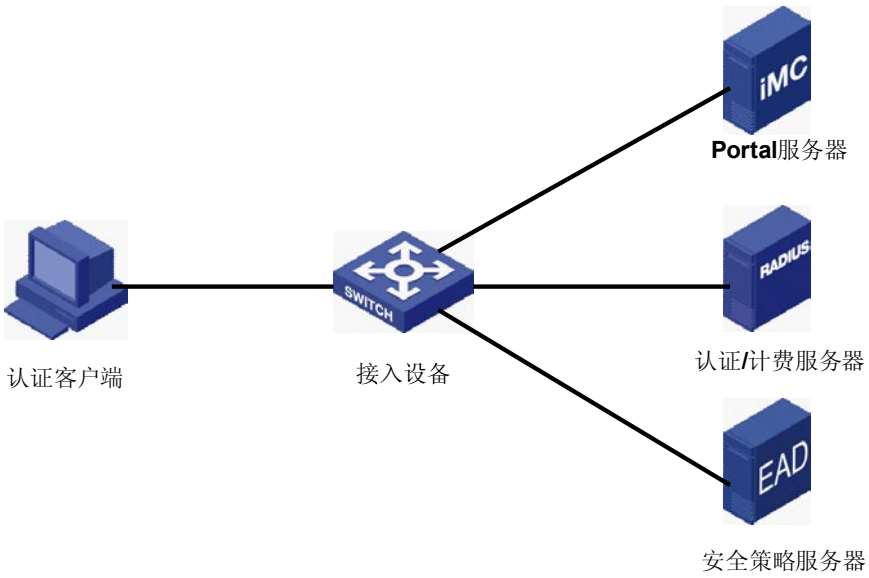
端口定义：

PortalServer通过默认端口（50100）侦听BAS发来的报文；

BAS通过端口2000侦听来自PortalServer的所有报文。

对EAD系统的支持

通过EAD的系统中的安全策略服务器，Portal可以实现其扩展认证功能，实现基于客户端与安全策略



服务器之间的交互来进行后续的安全检测功能。

Portal对EAD的支持需要用户在终端上安装专用的Portal客户端软件，用户在通过Portal认证后，安全策略服务器通过与Portal客户端、接入设备进行交互，完成对用户的安全认证，若对用户采用了安全策略，则用户的安全检测通过之后，安全策略服务器根据用户的安全性授权用户访问非受限资源。

Portal在EAD系统中通过联动的机制主动的实施安全策略，联动的基本方式如下：

● 客户端与安全策略服务器联动

1、客户端上线时Portal-Server会在Login-Response报文中携带EAD服务器的IP地址及端口号；

2、用户上线后客户端和安全策略服务器进行交互，服务器下发检查策略，客户端按策略检查所在PC的安全情况，并上报服务器；

3、用户在线过程中客户端仍会定期上报安全情况，以适应动态检测(即EAD心跳)，安全检测使用的报文为UDP，通常端口号是9019(Server)/10102(Client);

● BAS与安全策略服务器的联动

H3C对RADIUS协议进行了扩展，定义了Type20(Session-Control)，当用户上线后，通过该类型的RADIUS报文下发隔离ACL，等通过安全检查后，再下发安全ACL。

PORTAL认证方式

认证方式分类

不同的组网方式下，可采用的Portal认证方式不同。按照网络中实施Portal认证的网络层次来分，Portal的认证方式分为两种：二层认证方式和三层认证方式。

● 二层认证方式

这种方式支持在接入设备连接用户的二层端口上开启Portal认证功能，只允许源MAC地址通过认证的用户才能访问外部网络资源。目前，该认证方式仅支持本地Portal认证，即接入设备作为本地Portal服务器向用户提供Web认证服务。

● 三层认证方式

这种方式支持在接入设备连接用户的三层接口上开启Portal认证功能。三层接口Portal认证又可分为三种不同的认证方式：直接认证方式、二次地址分配认证方式和可跨三层认证方式。直接认证方式和二次地址分配认证方式下，认证客户端和接入设备之间没有三层转发；可跨三层认证方式下，认证客户端和接入设备之间可以跨接三层转发设备。

● 直接认证方式

用户在认证前通过手工配置或DHCP直接获取一个IP地址，只能访问Portal服务器，以及设定的免费访问地址；认证通过后即可访问网络资源。认证流程相对二次地址较为简单。

● 二次地址分配认证方式

用户在认证前通过DHCP获取一个私网IP地址，只能访问Portal服务器，以及设定的免费访问地址；认证通过后，用户会申请到一个公网IP地址，即可访问网络资源。该认证方式解决了IP地址规划和分配问题，对未认证通过的用户不分配公网IP地址。例如运营商对于小区宽带用户只在访问小区外部资源时才

分配公网IP。

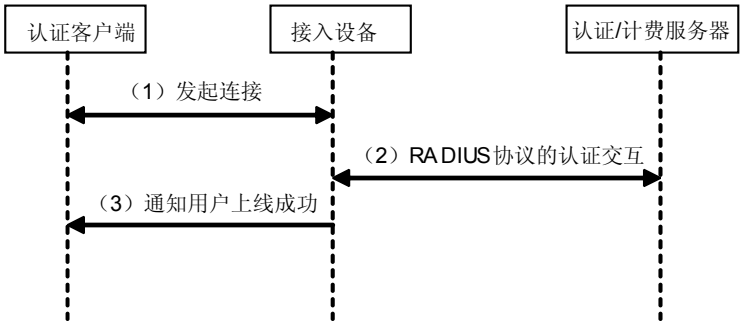
● 可跨三层认证方式

和直接认证方式基本相同，但是这种认证方式允许认证用户和接入设备之间跨越三层转发设备。

对于以上三种认证方式，IP地址都是用户的唯一标识。接入设备基于用户的IP地址下发ACL对接口上通过认证的用户报文转发进行控制。由于直接认证和二次地址分配认证下的接入设备与用户之间未跨越三层转发设备，因此接口可以学习到用户的MAC地址，接入设备可以利用学习到MAC地址增强对用户报文转发的控制粒度。

二层Portal认证过程

(1)Portal用户通过HTTP或HTTPS协议发起认证请求。HTTP报文经过配置了本地Portal服务器的接



入设备的端口时会被重定向到本地Portal服务器的监听IP地址，本地Portal服务器提供Web页面供用户输入用户名和密码来进行认证。该本地Portal服务器的监听IP地址为接入设备上一个与用户之间路由可达的三层接口IP地址（通常为Loopback接口IP）。

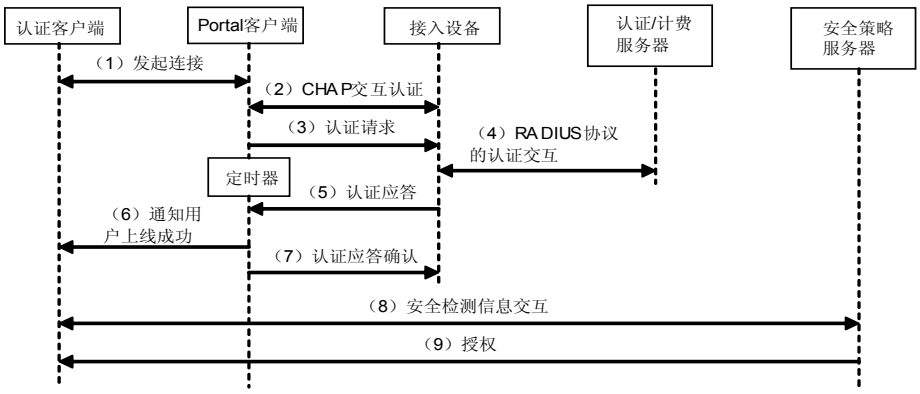
(2)接入设备与RADIUS服务器之间进行RADIUS协议报文的交互，对用户身份进行验证。

(3)如果RADIUS认证成功，则接入设备上的本地Portal服务器向客户端发送登录成功页面，通知客户端认证（上线）成功。

三层Portal认证过程

直接认证和可跨三层Portal认证的流程

直接认证/可跨三层Portal认证流程：



(1)Portal用户通过HTTP协议发起认证请求。HTTP报文经过接入设备时，对于访问Portal服务器或设定的免费访问地址的HTTP报文，接入设备允许其通过；对于访问其它地址的HTTP报文，接入设备将其重定向到Portal服务器。Portal服务器提供Web页面供用户输入用户名和密码来进行认证。

(2)Portal服务器与接入设备之间进行CHAP（ChallengeHandshakeAuthenticationProtocol，质询握手验证协议）认证交互。若采用PAP（PasswordAuthenticationProtocol，密码验证协议）认证则直接进入下一步骤。

(3)Portal服务器将用户输入的用户名和密码组装成认证请求报文发往接入设备，同时开启定时器等待认证应答报文。

(4)接入设备与RADIUS服务器之间进行RADIUS协议报文的交互。

(5)接入设备向Portal服务器发送认证应答报文。

(6)Portal服务器向客户端发送认证通过报文，通知客户端认证（上线）成功。

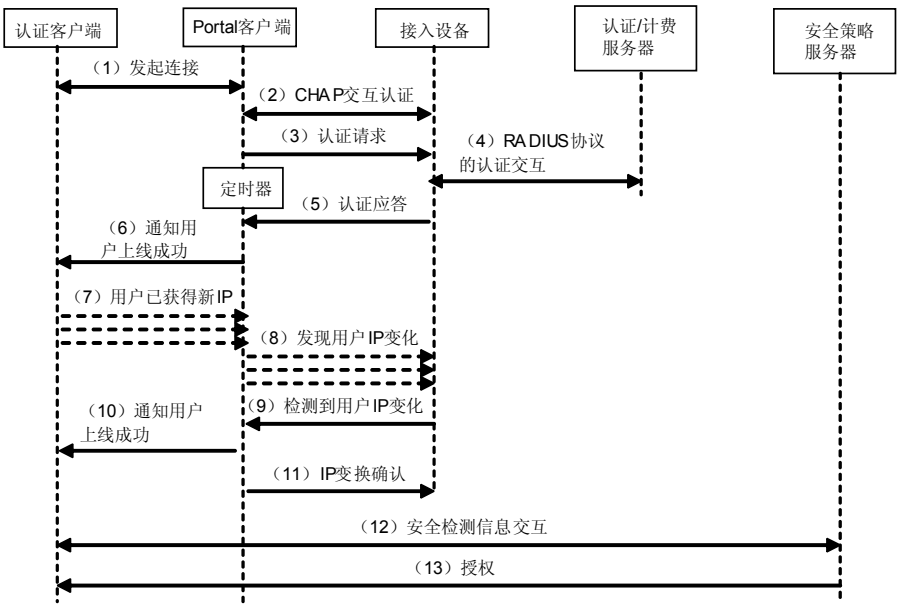
(7)Portal服务器向接入设备发送认证应答确认。

(8)客户端和安全策略服务器之间进行安全信息交互。安全策略服务器检测接入终端的安全性是否合格，包括是否安装防病毒软件、是否更新病毒库、是否安装了非法软件、是否更新操作系统补丁等。

(9)安全策略服务器根据用户的安全性授权用户访问非受限资源，授权信息保存到接入设备中，接入设备将使用该信息控制用户的访问。

步骤(8)、(9)为Portal认证扩展功能的交互过程

二次地址分配认证方式的流程：



二次地址分配认证流程：

(1) ~ (4)同直接/可跨三层Portal认证中步骤(1) ~ (4)。

(5)用户在接入设备上认证成功后，BAS向Portal-Server发送带有IP-Config属性的认证回应报文，指出用户需要更新IP地址;

(6)Portal-Server再向客户端发送带有IP-Config属性的认证通过报文（Login-Response），要求客户程序释放再申请IP地址;

(7)客户端成功更新IP地址后，向Portal-Server报告更新IP地址成功。

(8)Portal服务器通知接入设备客户端获得新公网IP地址。

(9)接入设备通过检测ARP协议报文检测到了用户IP变化，并通告Portal服务器已检测到用户IP变化。

(10)Portal服务器通知客户端上线成功。

(11)Portal服务器向接入设备发送IP变化确认报文。

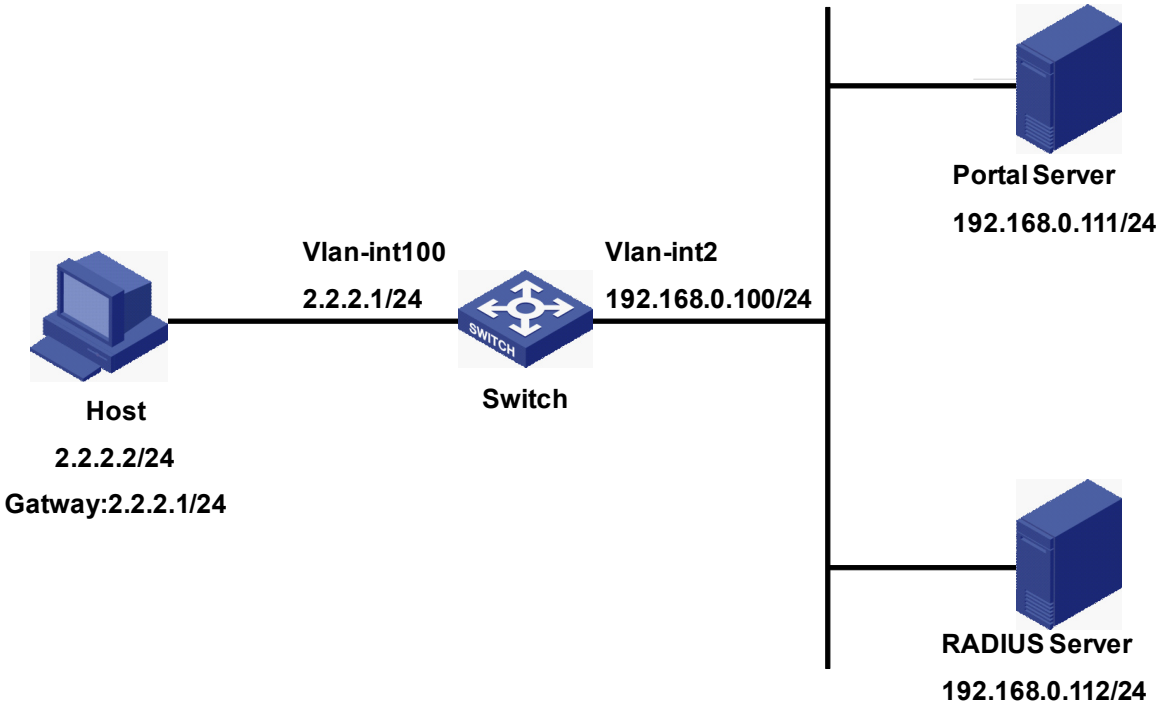
(12)客户端和安全策略服务器之间进行安全信息交互。安全策略服务器检测接入终端的安全性是否合格，包括是否安装防病毒软件、是否更新病毒库、是否安装了非法软件、是否更新操作系统补丁等。

(13)安全策略服务器根据用户的安全性授权用户访问非受限资源，授权信息保存到接入设备中，接入设备将使用该信息控制用户的访问。

步骤(12)、(13)为Portal认证扩展功能的交互过程

典型应用及配置举例

Portal直接认证配置举例



配置RADIUS方案

#创建名字为rs1的RADIUS方案并进入该方案视图。

<Switch>system-view

[Switch] radius scheme rs1

#配置RADIUS方案的服务器类型。使用CAMS/iMC服务器时，RADIUS服务器类型应选择extended。

[Switch-radius-rs1]server-typeextended

#配置RADIUS方案的主认证和主计费服务器及其通信密钥。

[Switch-radius-rs1]primary authentication 192.168.0.112

[Switch-radius-rs1]primary accounting 192.168.0.112

[Switch-radius-rs1]key authentication radius

[Switch-radius-rs1]key accounting radius

#配置发送给RADIUS服务器的用户名不携带ISP域名。

[Switch-radius-rs1]user-name-format without-domain

[Switch-radius-rs1]quit

配置认证域

#创建并进入名字为dm1的ISP域。

[Switch]domain dm1

#配置ISP域的AAA方法。

```
[Switch-isp-dm1]authentication portal radius-scheme rs1
[Switch-isp-dm1]authorization portal radius-scheme rs1
[Switch-isp-dm1]accounting portal radius-scheme rs1
[Switch-isp-dm1]quit
```

#配置系统缺省的ISP域dm1，所有接入用户共用此缺省域的认证和计费方法。若用户登录时输入的用户名未携带ISP域名，则使用缺省域下的认证方法。

```
[Switch]domain default enable dm1
```

配置Portal认证

#配置Portal服务器：名称为newpt，IP地址为192.168.0.111，密钥为portal，端口为50100，URL为http://192.168.0.111:8080/portal。

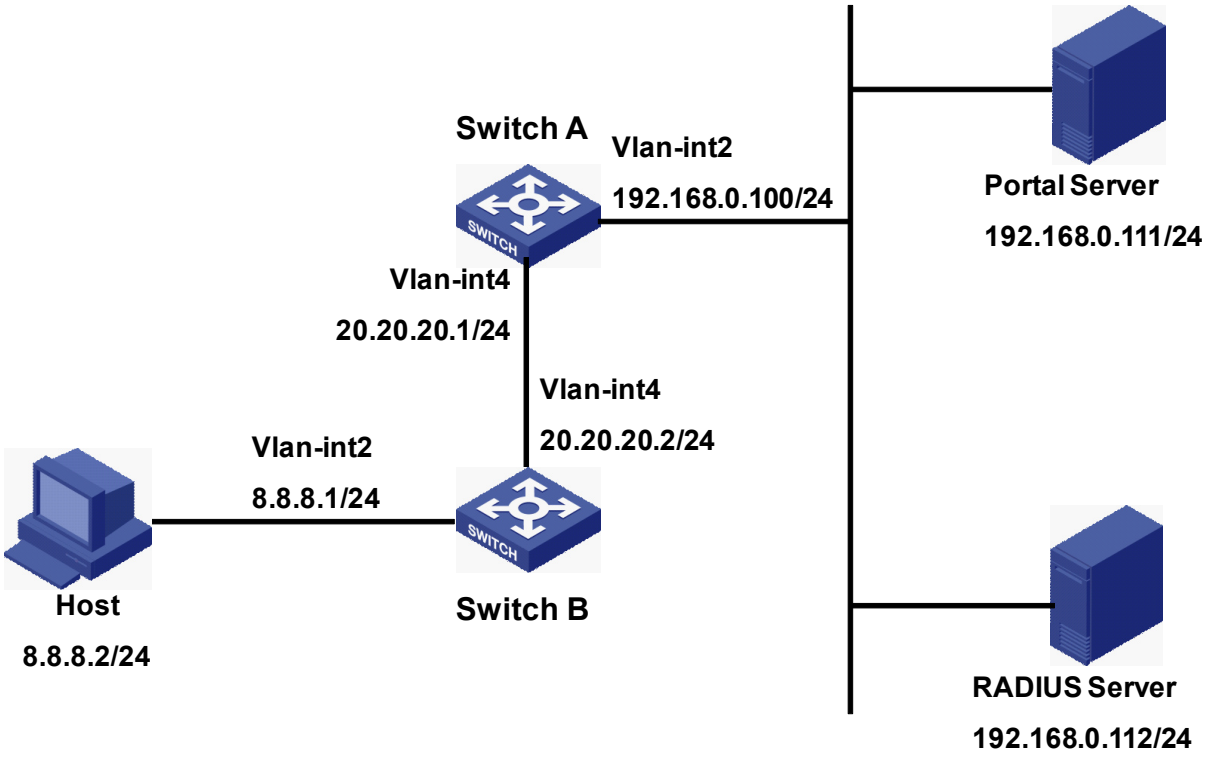
```
[Switch]portal server newpt ip 192.168.0.111 key portal port 50100 url http://192.168.0.111:
8080/portal
```

#在与用户Host相连的接口上使能Portal认证。

```
[Switch]interface vlan-interface 100
[Switch-Vlan-interface100]portal server newpt method direct
[Switch]quit
```

可跨三层Portal认证配置举例

(1)配置RADIUS方案



#创建名字为rs1的RADIUS方案并进入该方案视图。

<SwitchA>system-view

[SwitchA]radius scheme rs1

#配置RADIUS方案的服务器类型。使用CAMS/iMC服务器时，RADIUS服务器类型应选择extended。

[SwitchA-radius-rs1]server-typeextended

#配置RADIUS方案的主认证和主计费服务器及其通信密钥。

[SwitchA-radius-rs1]primary authentication 192.168.0.112

[SwitchA-radius-rs1]primary accounting 192.168.0.112

[SwitchA-radius-rs1]key authentication radius

[SwitchA-radius-rs1]key accounting radius

#配置发送给RADIUS服务器的用户名不携带ISP域名。

[SwitchA-radius-rs1]user-name-format without-domain

[SwitchA-radius-rs1]quit

(2)配置认证域

#创建并进入名字为dm1的ISP域。

[SwitchA]domain dm1

#配置ISP域的AAA方法。

[SwitchA-isp-dm1]authentication portal radius-scheme rs1

[SwitchA-isp-dm1]authorization portal radius-scheme rs1

[SwitchA-isp-dm1]accounting portal radius-scheme rs1

[SwitchA-isp-dm1]quit

#配置系统缺省的ISP域dm1，所有接入用户共用此缺省域的认证和计费方法。若用户登录时输入的用户名未携带ISP域名，则使用缺省域下的认证方法。

[SwitchA]domain default enable dm1

(3)配置Portal认证

#配置Portal服务器：名称为newpt，IP地址为192.168.0.111，密钥为portal，端口为50100，URL为http://192.168.0.111:8080/portal（请根据Portal服务器的实际情况配置，此处仅为示例）。

[SwitchA]portal server newpt ip 192.168.0.111 key portal port 50100 url http://192.168.0.111:8080/portal

#在与SwitchB相连的接口上使能Portal认证。

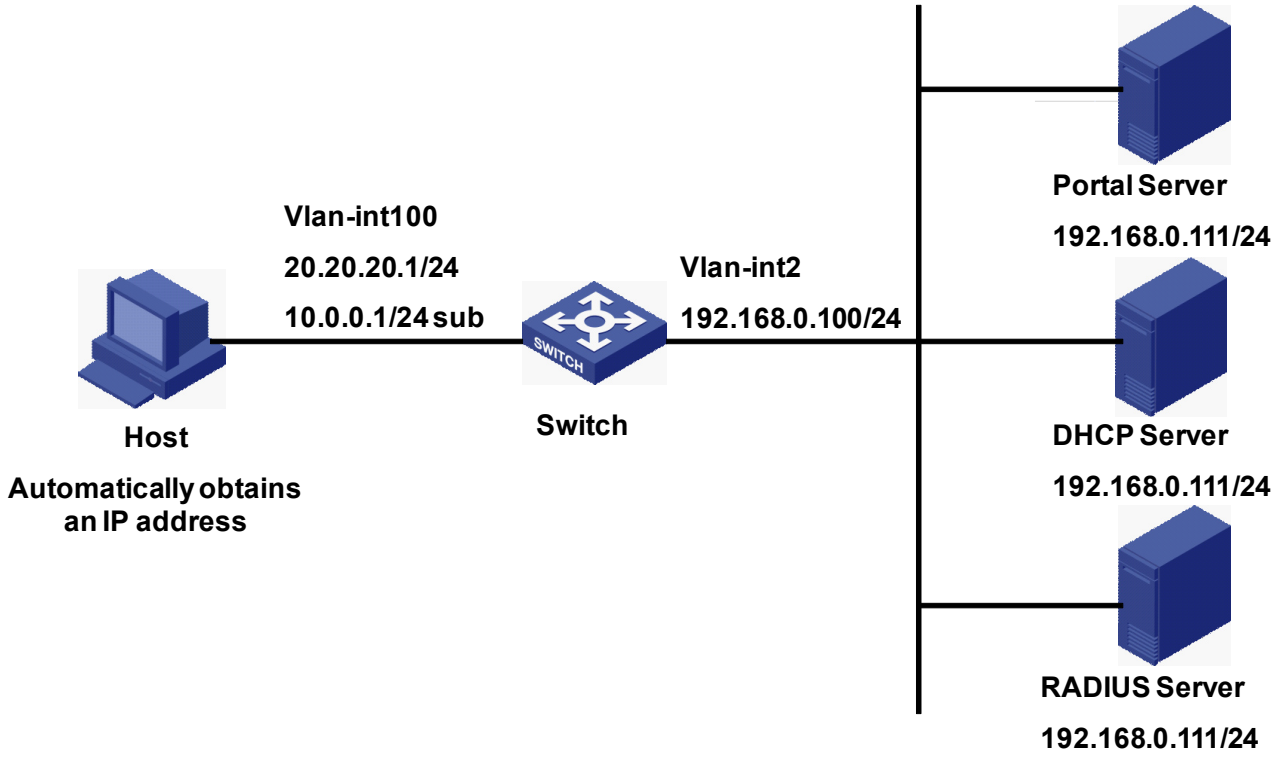
[SwitchA]interfacevlan-interface4

[SwitchA-Vlan-interface4]portalservernewptmethodlayer3

[SwitchA-Vlan-interface4]quit

SwitchB上需要配置到192.168.0.0/24网段的缺省路由，下一跳为20.20.20.1，具体配置略。

Portal二次地址分配认证配置举例



1)配置RADIUS方案

#创建名字为rs1的RADIUS方案并进入该方案视图。

<Switch>system-view

[Switch]radius scheme rs1

#配置RADIUS方案的服务器类型。使用CAMS/iMC服务器时，RADIUS服务器类型应选择extended。

[Switch-radius-rs1]server-type extended

#配置RADIUS方案的主认证和主计费服务器及其通信密钥。

[Switch-radius-rs1]primary authentication 192.168.0.113

[Switch-radius-rs1]primary accounting 192.168.0.113

[Switch-radius-rs1]key authentication radius

[Switch-radius-rs1]key accounting radius

#配置发送给RADIUS服务器的用户名不携带ISP域名。

[Switch-radius-rs1]user-name-format without-domain

[Switch-radius-rs1]quit

(2)配置认证域

#创建并进入名字为dm1的ISP域。

[Switch]domain dm1

#配置ISP域的AAA方法。

[Switch-isp-dm1]authentication portal radius-scheme rs1

[Switch-isp-dm1]authorization portal radius-scheme rs1

```
[Switch-isp-dm1]accounting portal radius-scheme rs1
[Switch-isp-dm1]quit
```

#配置系统缺省的ISP域dm1，所有接入用户共用此缺省域的认证和计费方法。若用户登录时输入的用户名未携带ISP域名，则使用缺省域下的认证方法。

```
[Switch]domain default enable dm1
```

(3)配置Portal认证

#配置Portal服务器：名称为newpt，IP地址为192.168.0.111，密钥为portal，端口为50100，URL为http://192.168.0.111:8080/portal（请根据Portal服务器的实际情况配置，此处仅为示例）。

```
[Switch]portal server newpt ip 192.168.0.111 keyportal port 50100 url http://192.168.0.111:
8080/portal
```

#配置DHCP中继，并启动DHCP中继的安全地址匹配检查功能。

```
[Switch]dhcp enable
[Switch]dhcp relay server-group 0 ip 192.168.0.112
[Switch]interface vlan-interface 100
[Switch-Vlan-interface100]ipaddress 20.20.20.1 255.255.255.0
[Switch-Vlan-interface100]ipaddress 10.0.0.1 255.255.255.0 sub
[Switch-Vlan-interface100]dhcp select relay
[Switch-Vlan-interface100]dhcp relay server-select 0
[Switch-Vlan-interface100]dhcp relay address-check enable
```

#在与用户Host相连的接口上使能Portal认证。

```
[Switch-Vlan-interface100]portal server newpt method redhcp
[Switch-Vlan-interface100]quit
```

MAC认证技术基础

■文/盛冬旭

简介

MAC地址认证是一种基于端口和MAC地址对用户的网络访问权限进行控制的认证方法，它不需要用户安装任何客户端软件。设备在启动了MAC地址认证的端口上首次检测到用户的MAC地址以后，即启动对该用户的认证操作。认证过程中，不需要用户手动输入用户名或者密码。

认证格式和认证方式

目前设备支持两种方式的MAC地址认证，通过RADIUS（Remote Authentication Dial-In User Service，远程认证拨号用户服务）服务器进行远程认证，以及在接入设备上进行本地认证。

在介绍认证方式之前，首先需要定义用于验证用户身份的用户名格式和内容。根据设备处理方式的不同，可以将MAC地址认证使用的用户名格式分为两种类型：

一种类型是以MAC地址为用户名的格式，即使用用户的MAC地址作为认证时的用户名和密码。

另外一种格式是使用固定用户名，即不论用户的MAC地址为何值，所有用户均使用设备上预先设置的一个固定用户名和密码作为身份信息进行认证。由于同一个端口下可以有多个用户进行认证，因此这种情况下端口上的所有MAC地址认证用户均使用同一个固定用户名进行认证，服务器端仅需要配置一个用户帐户即可满足所有认证用户的认证需求，适用于接入客户端比较可信的网络环境。

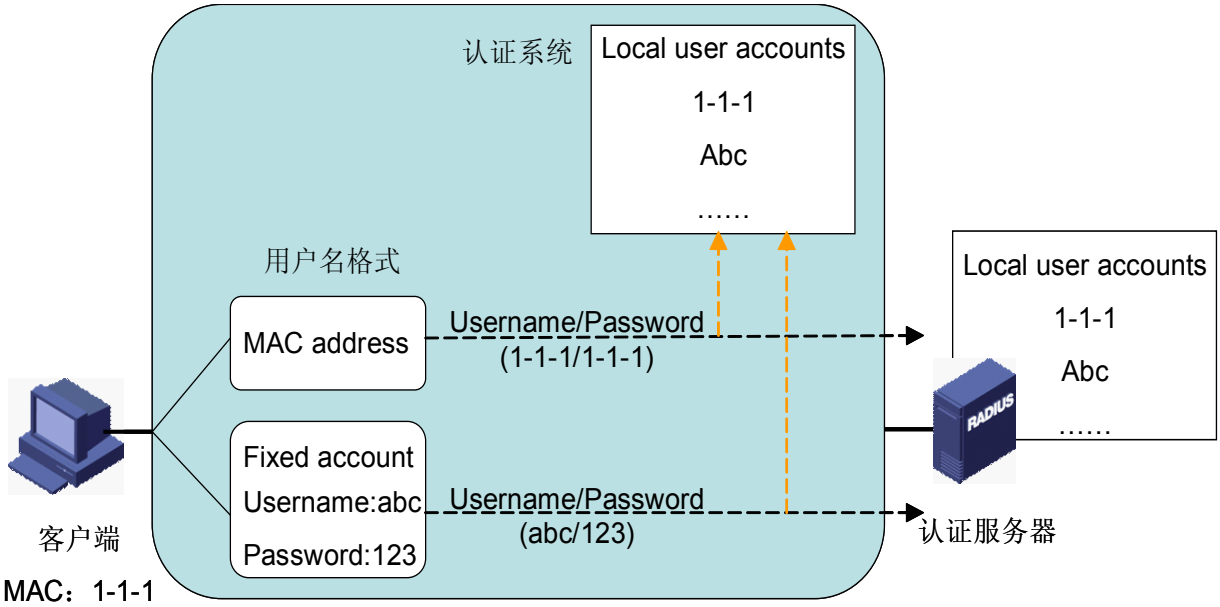


图1 不同用户名格式下的MAC地址认证示意图

两种验证用户身份的用户名格式确定后，回过头来看目前设备支持的两种MAC地址认证方式：

当选用RADIUS服务器认证方式进行MAC地址认证时，设备作为RADIUS客户端，与RADIUS服务器配合完成MAC地址认证操作。下面就两种验证用户身份的用户名格式简要说明一下该认证方式：

首先，若采用MAC地址用户名格式，则设备将检测到的用户MAC地址作为用户名和密码发送给RADIUS服务器进行验证；其次，若采用固定用户名格式，则设备将一个已经在本地指定的MAC地址认证用户使用的固定用户名和对应的密码作为待认证用户的用户名和密码，发送给RADIUS服务器进行验证。总之，无论设备采用何种用户名格式与RADIUS服务器完成对该用户的认证，所有认证通过的用户均可以访问网络。

当选用本地认证方式进行MAC地址认证时，直接在设备上完成对用户的认证。这时需要事先在设备上配置本地用户名和密码。

采用MAC地址用户名格式时，需要配置的本地用户名为各接入用户的MAC地址；而采用固定用户名时，需要配置的本地用户名为自定义的，所有用户对应的用户名和密码与自定义的一致。最后和RADIUS

服务器认证方式一样，用户名和密码匹配成功后，用户便可以访问网络资源。

认证触发与下线

由于这部分内容涉及到MAC地址认证定时器，所以首先简要介绍一下三个定时器的功能：

第一个就是下线检测定时器（offline-detect），它的主要功能是用来设置用户空闲超时的时间间隔。如果在两个时间间隔之内，某在线用户没有流量通过设备，设备将切断该用户的连接，同时通知RADIUS服务器，停止对该用户的计费。

第二个是静默定时器（quiet），用来设置用户认证失败以后，设备停止对其提供认证服务的时间间隔。在静默期间，设备不对来自该用户的报文进行认证处理，直接丢弃。静默期后，如果设备再次收到该用户的报文，则依然可以对其进行认证处理。

第三个是服务器超时定时器（server-timeout），用来设置设备同RADIUS服务器的连接超时时间。在用户的认证过程中，如果达到服务器超时定时器设置的超时时间，而设备一直没有收到RADIUS服务器的应答，则设备将在相应的端口上禁止此用户访问网络。

介绍完以上定时器的功能后，下面我们来看MAC地址认证的触发方式和下线方式：

MAC地址认证触发，目前只有一种触发方式：new-MAC事件触发，就是有了新的MAC到使能了MAC认证的端口，并被该端口学习到，即可触发MAC地址认证。

而MAC地址认证下线，则有多种方式实现，一种是用户主动要求下线，即用户主动终止访问流量后通过设备上配置的下线检测定时器，检测到在线用户没有流量通过，而后由设备切断该用户的连接；另外一种是用户被动下线，即通过设备切断在线用户的连接，使用户失去访问网络资源的权限；还有一种比较特殊，严格来说不属于用户下线，而是属于认证失败，即该用户未通过MAC地址认证，且该用户的MAC地址就被添加为静默MAC，即在静默时间内，来自此MAC地址的数据报文到达时，设备直接做丢弃处理。静默MAC的功能主要是防止非法MAC短时间内的重复认证。

扩展属性

下发VLAN

为了将受限的网络资源与未认证用户隔离，通常将受限的网络资源和用户划分到不同的VLAN。MAC地址认证支持认证服务器（远程或本地）授权下发VLAN功能，即当用户通过MAC地址认证后，认证服务器将指定的受限网络资源所在的VLAN作为授权VLAN下发到用户认证的端口。该端口被加入到授权VLAN中后，用户便可以访问这些受限的网络资源。

设备根据用户接入的端口链路类型，按不同情况将端口加入下发的授权VLAN中：

若用户从Access类型的端口接入，则端口离开当前VLAN并加入下发的授权VLAN中；若用户从Trunk类型的端口接入，则设备允许下发的授权VLAN通过该端口，并且修改该端口的缺省VLAN为下发的授权VLAN；若用户从Hybrid类型的端口接入，则设备允许授权下发的VLAN以不携带Tag的方式通过该端口，并且修改该端口的缺省VLAN为下发的授权VLAN。这里需要注意的是，若该端口上使能了MAC VLAN功能，则设备将根据认证服务器下发的授权VLAN动态地创建基于用户MAC的VLAN，而端口的缺

省VLAN并不改变。

下发ACL

从认证服务器（远程或本地）下发的ACL被称为授权ACL，它为用户访问网络的权限提供了良好的过滤功能。MAC地址认证支持认证服务器下发授权ACL功能，即当用户通过MAC地址认证后，如果认证服务器上配置了授权ACL，则设备会根据服务器下发的授权ACL对用户所在用户的数据流进行控制。为使下发的授权ACL生效，需要提前在设备上配置相应的ACL规则。而且在用户访问网络的过程中，可以通过改变服务器的授权ACL设置来改变用户的访问权限。

Guest VLAN

MAC地址认证的Guest VLAN功能允许用户在认证失败的情况下，访问某一特定VLAN中的资源，比如获取客户端软件、升级客户端或执行其他一些用户升级程序。这个VLAN称之为Guest VLAN。

如果接入用户的端口上配置了Guest VLAN，则该端口上认证失败的用户会被加入Guest VLAN，即该用户被授权访问Guest VLAN里的资源。此时若在设备上设置了静默定时器，则在静默期间内，即使用户发起认证请求设备也不会对该用户的报文进行认证处理，只有等到静默期过后，才可对此用户的报文进行认证处理。若Guest VLAN中的用户再次发起认证未成功，则该用户将仍然处于Guest VLAN内；若认证成功，则会根据认证服务器是否下发授权VLAN决定是否将用户加入到下发的授权VLAN中，在认证服务器未下发授权VLAN的情况下，用户回到加入Guest VLAN之前端口所在的VLAN。■

■文/王瑞

Triple认证技术

技术背景

MAC认证、802.1X认证、Portal认证是三种接入认证方式，三者分别由new MAC事件、EAP报文、HTTP报文来触发。针对不同的应用场景，使用相应的部署。802.1X认证需要客户端的配合，适用于安装了802.1X客户端的主机；而需要使用WEB界面来登录的用户可以选择Portal认证；MAC认证则适用于打印机终端等无法输入用户名密码需要通过MAC来认证的用户。

为了满足网络终端不同用户的需求，满足在同一端口上下挂不同的认证用户，推出了Triple认证。Triple认证可以支持在一个交换机端口上使能802.1X认证、MAC认证以及二层Portal认证，在Triple认证中采用基于MAC认证模式的802.1X。三种认证针对不同的应用场景，将三种认证在端口上同时支持，增加认证环境部署的灵活性。

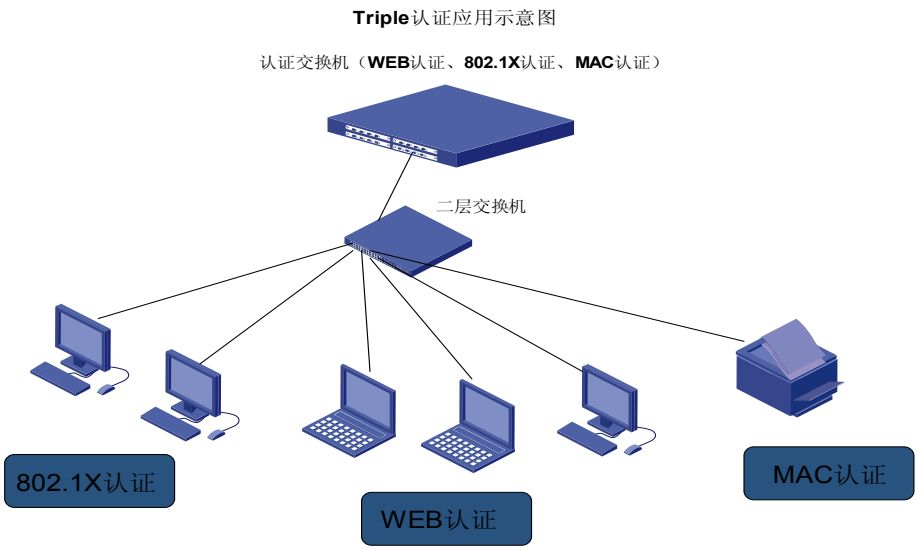


图1 Triple认证应用示意图

二层Portal认证

802.1X认证、MAC认证上两章已经介绍，在这里不再赘述。

二层Portal认证是在二层端口上开启Portal的一种认证方式，由接入设备作为本地Portal服务器向用户提供Web认证服务。本地Portal服务器提供Web页面供用户输入用户名和密码来进行认证，根据MAC来区分用户。

二层Portal认证可以由HTTP或者HTTPS报文触发，设备可以配置选择由HTTP或者HTTPS报文触发的方式。端口开启二层Portal后会拦截所有HTTP或者HTTPS报文送上CPU处理。

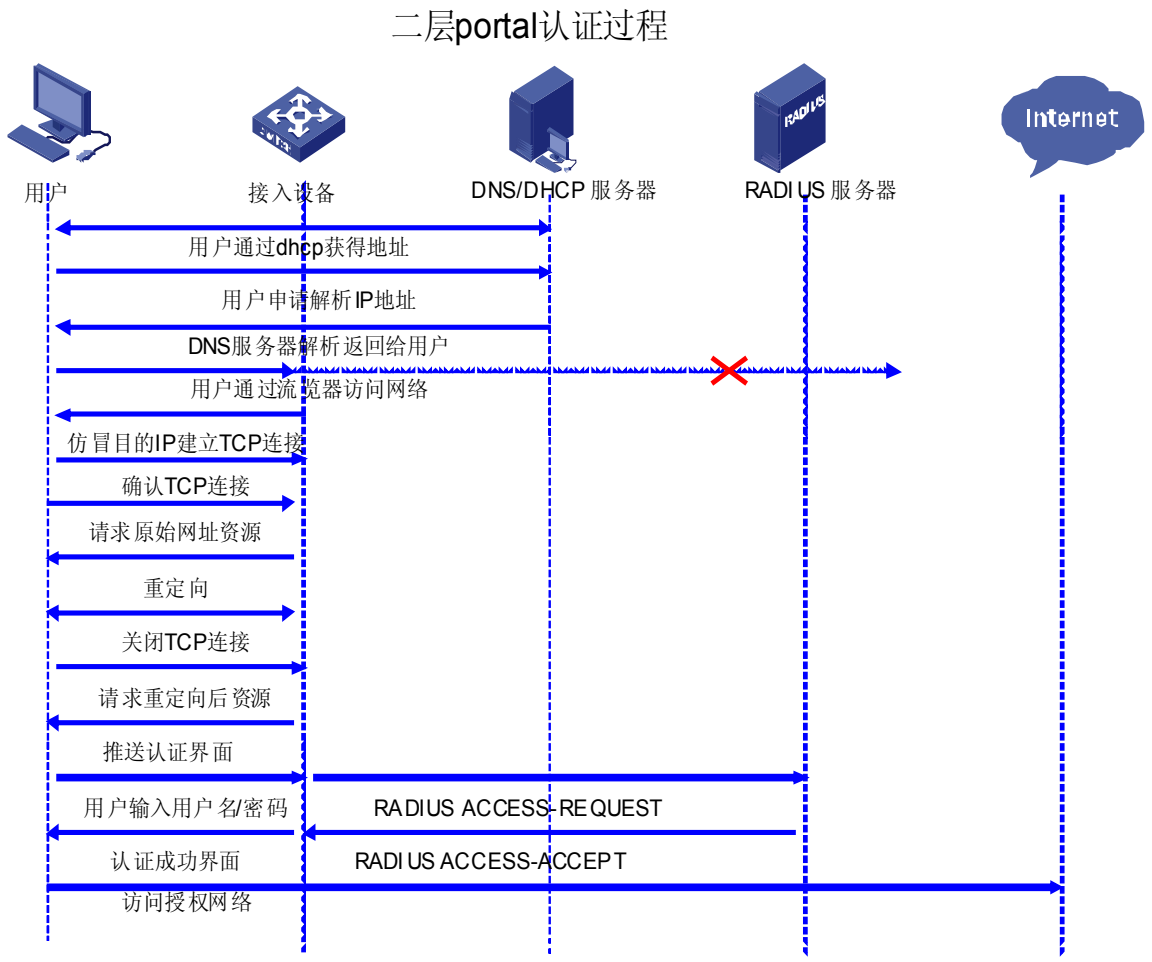


图2 二层Portal认证过程示意图

用户在未通过认证之前无法访问网络，此时只有DHCP、DNS报文可以通过。认证上线的过程如下：

（1）用户通过DHCP动态获得地址；

（2）DNS解析到目的URL的IP地址，通过浏览器向目的IP地址发起TCP连接；认证设备截获TCP连接请求并重定向到接入设备进行处理；

（3）接入设备收到TCP连接请求，将自己伪造成终端设备要访问的目的URL，以目的IP地址为源地址伪造报文回应接入设备，建立TCP链接；

（4）端收到伪造的IP报文，和接入控制设备建立连接并在此TCP连接上进行HTTP交互；

（5）终端向接入控制设备请求访问URL资源，接入控制设备发送HTTP重定向报文，通知终端PC所要访问的资源已经迁移，并指出迁移后的URL地址，该URL也是在接入设备上的WEB页面，用于推送认证页面；

（6）终端PC上的IE或其它浏览器终止TCP链接并转而访问迁移后的URL；

（7）认证设备和终端PC建立TCP和HTTP之后，管理设备向PC设备推送迁移后的页面即WEB认证页面；

（8）用户在页面上输入用户名称、密码发起认证；

（9）接入设备收到用户名称和密码，将这些信息封装在标准的RADIUS报文中，向RADIUS服务器发起RADIUS Access-Request请求；

（10）服务器验证用户名称和密码；认证成功后，用户可以成功访问网络。

同时二层Portal认证支持丰富的用户自定义页面功能，包括登录页面、登录成功页面、登录失败页面、在线页面、系统忙页面、下线成功页面均可以自定义。页面由用户自行编辑后上传到设备指定位置即可，页面元素可以包含HTML、JS、CSS和图片等内容。

几种认证之间的覆盖关系

对于接入用户，只要通过其中一种认证，就可以访问相关权限。用户通过了MAC认证以后，再使用802.1X客户端触发1X认证可以认证成功，覆盖之前MAC认证信息，而当802.1X认证失败后又可以重新通过new MAC事件来触发MAC认证。二层Portal认证以及802.1X认证成功后不能被其他认证覆盖。同等级别的认证先来先触发，后面的认证请求被忽略。

认证相关功能

授权下发VLAN

设备对认证成功的用户可以添加相应的授权VLAN，Triple认证应用中采用基于MAC的VLAN技术来实现对于不同的用户下发不同的权限。

基于MAC的VLAN按照报文的源MAC地址来定义VLAN成员，将指定报文加入该VLAN的Tag后在交换机内转发，当收到的报文为untagged报文时，会以报文的源MAC为根据去匹配MAC-VLAN表项。MAC-VLAN表项可以静态配置也可以动态的下发。在Triple认证应用中MAC-VLAN表项采用动态下发的方式，要求用户端口采用hybrid口，端口上启用MAC-VLAN。

在RADIUS服务器上配置下发VLAN信息，认证成功后接入设备将生成MAC-VLAN表项。同时会自动修改端口属性，允许下发的VLAN untagged通过。上线成功的用户根据的源MAC查询MAC-VLAN表项，按照表项中对应得VLAN进行转发。以下是设备上的MAC-VLAN表项信息，按照精确匹配的方式，不改变优先级，状态是Dynamic：

```
display MAC-vlan all
The following MAC VLAN addresses exist:
 S:Static   D:Dynamic
 MAC ADDR      MASK        VLAN ID  PRIO  STATE
 ------------------------------------------------------------
 000d-88f8-332b   ffff-ffff-ffff      40      0     D
 0015-e9a6-5c36   ffff-ffff-ffff      20      0     D
```

如表项所示通过认证的MAC 000d-88f8-332b可以访问VLAN 40里的资源，通过认证的MAC 0015-e9a6-5c36可以访问VLAN 20里的资源。

当用户下线时，设备删除相应MAC-VLAN表项，同时端口恢复之前的VLAN属性。

802.1X认证、MAC认证以及二层Portal认证都支持下发授权VLAN。

GUEST VLAN

GUEST VLAN设置了用户在认证成功前被授予的权限，在Triple认证中也是需要同MAC-VLAN结合实现的，Triple认证中支持基于MAC模式的802.1X GUEST VLAN。其触发机制区别于基于端口的802.1X GUEST VLAN。

之前802.1X的GUEST VLAN只能在基于端口认证模式下生效，在3个Handshake时间内没有认证成功的用户自动将端口加入设置的GUEST VLAN内，同时修改端口属性，后续从该端口接入的用户都可以访问GUEST VLAN里的资源。直到有用户认证成功，端口退出GUEST VLAN，当用户下线后，在检测时间内没有用户认证成功，端口又会加入GUEST VLAN。

而基于MAC模式的802.1X GUEST VLAN需要由未知源MAC来触发。即在有用户流量触发NEW MAC事件后，30s内会允许该用户访问GUEST VLAN里的资源，通过下发相应的MAC VLAN表项来实现，同时修改端口VLAN属性；当在规定的时长内没有流量通过，将会删除MAC VLAN表项，同时恢复端口VLAN属性。

Auth-fail VLAN

Auth-fail VLAN 划分了在用户认证失败后能够访问的资源。Triple认证支持MAC认证、802.1X认证和二层Portal认证的Auth-fail VLAN。

MAC认证的Auth-fail VLAN触发类似于GUEST VLAN，也是通过未知源MAC来触发，流量老化来删除。

而802.1X Auth-fail VLAN的触发需要客户端认证失败，例如用户名密码失败，此时设备会针对该用户下发MAC VLAN表项，为用户添加认证失败后访问的资源。退出Auth-fail VLAN，可以通过同一MAC认证通过，就会删除Auth-fail VLAN表项，如果用户不主动上下线，在用户空闲时启动定时器，在3次检查到用户没有流量的情况下强制删除Auth-fail VLAN。

二层Portal认证的Auth-fail VLAN加入删除同802.1X一致。■

EAD方案简介

■文/王君菠

>> 方案需求来源分析

伴随经济的高速发展，网络技术也突飞猛进，企业网络技术日新月异，需求也越来越多，随着网络规模越来越大，企业经营者对网络的理解也越来越深刻，越来越认识到网络对于企业运作的重要性，企业运作对网络的依赖性，因此企业经营者对网络的安全性和健壮性的建设都提高到了一个新的水平，因为他们知道，鉴于当前企业运作对网络的依赖性，网络一旦出现问题，就直接影响企业的运营效率，甚至所有业务都中断。

为了提高网络的防御能力，企业经营者纷纷在企业网部署了防火墙、IPS等安全设备，但是发现还是会病毒横行，尤其是新病毒、木马出现时，企业被感染的概率非常高。

分析其原因，一方面，员工的安全意识并没有网络管理者那么高，甚至没有安全意识，比如操作系统关键补丁更新后，虽然发布内部通告，但也很少有人去主动更新，而且管理员无法监控多少台PC更新了补丁。新的病毒出现后，也很少有员工去主动更新杀毒软件的病毒库，甚至在终端PC上都没有安装杀毒软件。

另外一方面，对于来访用户的控制也缺少必要的手段，只能在关键的服务器资源设置访问密码，但如果员工的安全意识差，可能就会导致密码的泄漏等，给非法用户访问提供途径。如果公司有无线网络，这种潜在的威胁就会更高，非法用户可以不进入公司就可以对内部网络进行非法接入和访问等。

除此之外，提高员工工作效率，适当约束员工行为也比较重要。在全民皆股的时代，员工上班时间炒股比较常见，另外，长时间聊QQ，打网络游戏造成企业的生产效率低下，甚至浏览非法网站等。

综上所述，企业经营者为了提高其经营、办公网络的安全性、健壮性和网络办公效率，需要一套立体、全面的解决方案。

>> 方案简介

EAD解决方案全称是Endpoint Admission Domination终端准入控制，其核心就是对终端用户的准入和控制。它以接入认证技术为载体，为网络提供一套立体的安全体系。

EAD主要组件分为终端用户、接入控制设备、第三方补丁和病毒服务器、安全策略服务器。逻辑上分为隔离区和授权安全区域。其接入控制流程如下：

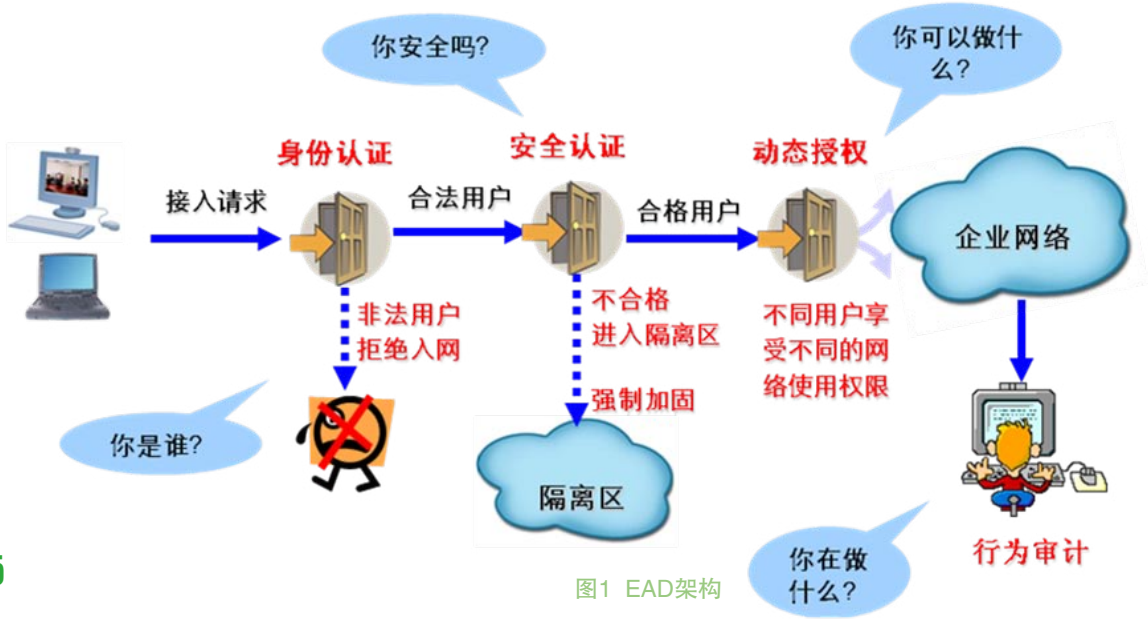


图1 EAD架构

终端用户首先通过接入认证技术进行身份认证，接入认证包括802.1X认证、PORTAL认证、L2TP认证。身份认证不通过，直接被拒绝，不能访问网络受保护资源。如果身份认证通过，则首先要把用户放到隔离区，然后进行安全检查，如果安全检查不合格，则用户将一直被隔离，只能访问隔离区域，直到自动修复或手工修复完成。如果安全检查合格，则用户会被动态授权，进入安全区域，可以访问受保护网络资源，但仅限于授权的网络资源，不能越权访问。此后用户在线过程中，其行为可以被约束，当打开或者安装禁止的进程或程序时，用户会被要求下线，禁止访问网络。这就是EAD的基本思想和操作。

>> 方案实现

部署方式

EAD方案以接入认证为载体实现用户的安全检查、行为监控和行为审计。接入认证技术可以是PORTAL、802.1X和L2TP，而且这三种接入技术分别适合不同的应用场景。

在企业网内部局域网部署EAD方案时，如果接入交换机都支持802.1X，就可以部署802.1X认证方式的EAD了。这种部署方式的好处是，控制点低，最大化的保护企业网络。

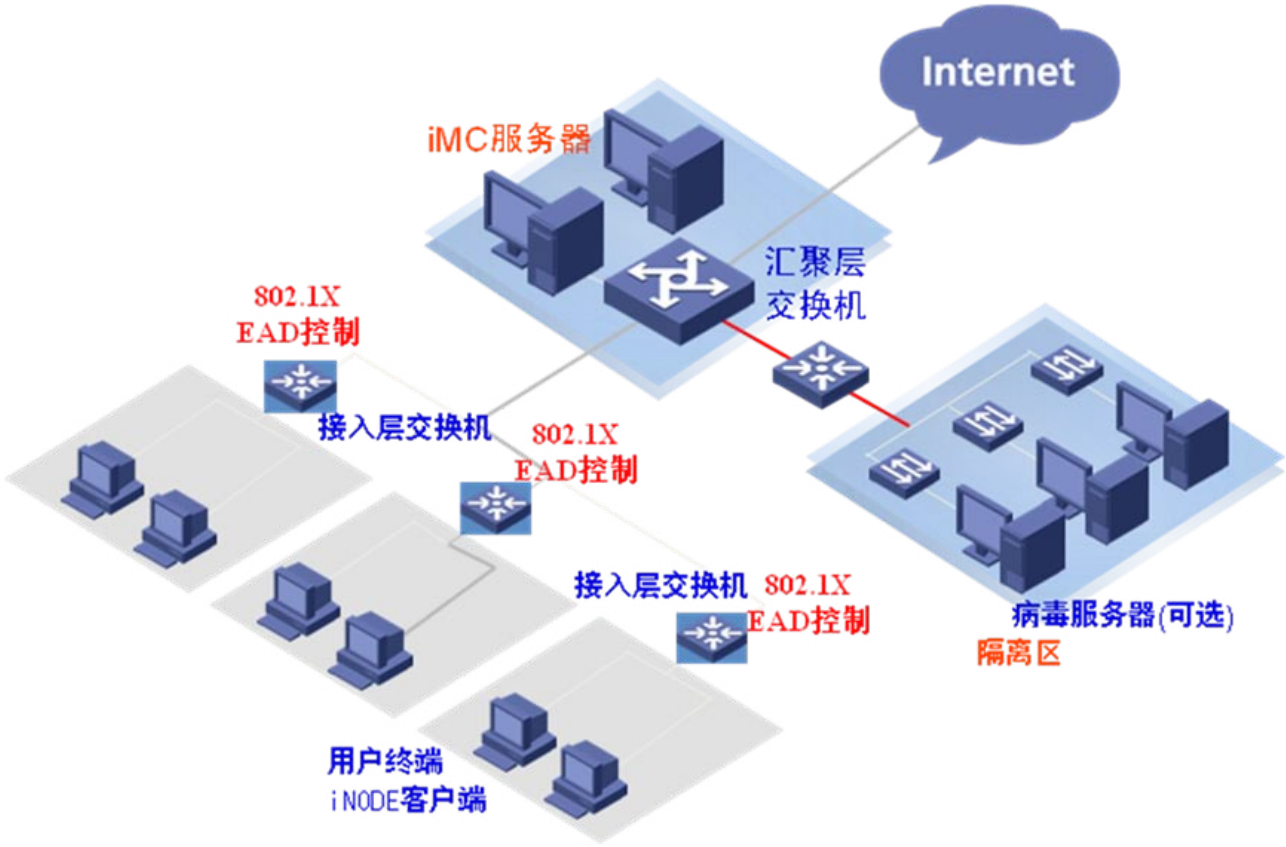


图2 EAD 802.1X认证组网

如果接入交换机不支持802.1X认证，可以在汇聚层部署控制点，这种情况下，推荐使用PORTAL认证方式。如果是老网络改造，可以把PORTAL网关旁挂在汇聚层交换机上。这种部署方式的好处是，对现网影响小，而且旁挂方式与直接替换汇聚层交换机比，节省单板和接口成本，对现网改动小。此

时需要在汇聚设备上使用策略路由，终端PC上行的流量全部被策略路由重定向到PORTAL网关上进行强制认证。下行流量可以直接在汇聚交换机下行，不经过PORTAL网关，这样做的目的是减少PORTAL网关的压力。而且旁挂的另外一个好处是，当一台PORTAL网关认证用户达到上限时，可以再旁挂一台PORTAL网关，扩容方便而且对现网改动小。如果不是旁挂，而是使用汇聚交换机直接做PORTAL，那么扩容时，就只能对网络“大动干戈”，新增一台PORTAL网关，而且把一部分接入交换机接入到该新增的网关上了。

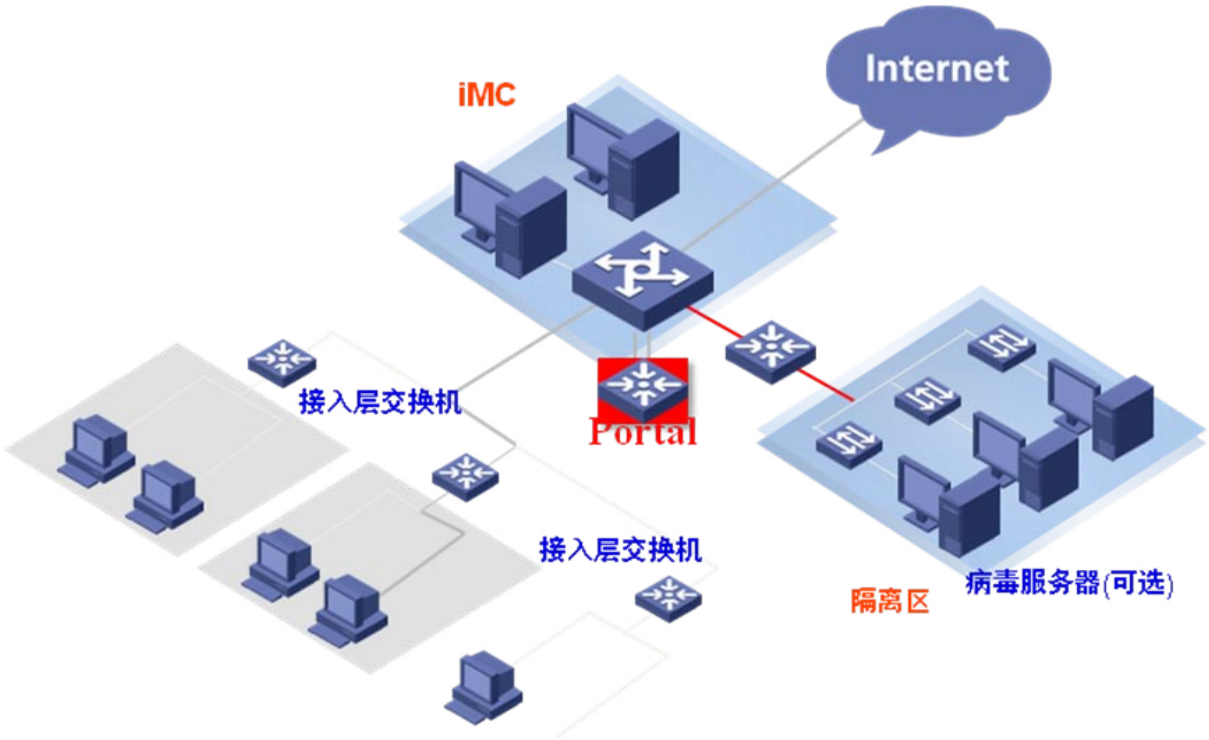


图3 EAD PORTAL旁挂组网

如果企业网有无线接入终端，可以部署无线EAD，无线EAD的控制点在无线控制器AC上，AC可以通过旁挂的方式部署到汇聚交换机或核心交换机上。AC和AP间的网络可以是二层网络，也可以是三层网络，不同支持仅在于AP和AC建立的隧道是二层隧道还是三层隧道，对于EAD本身来讲，区别不大如图:

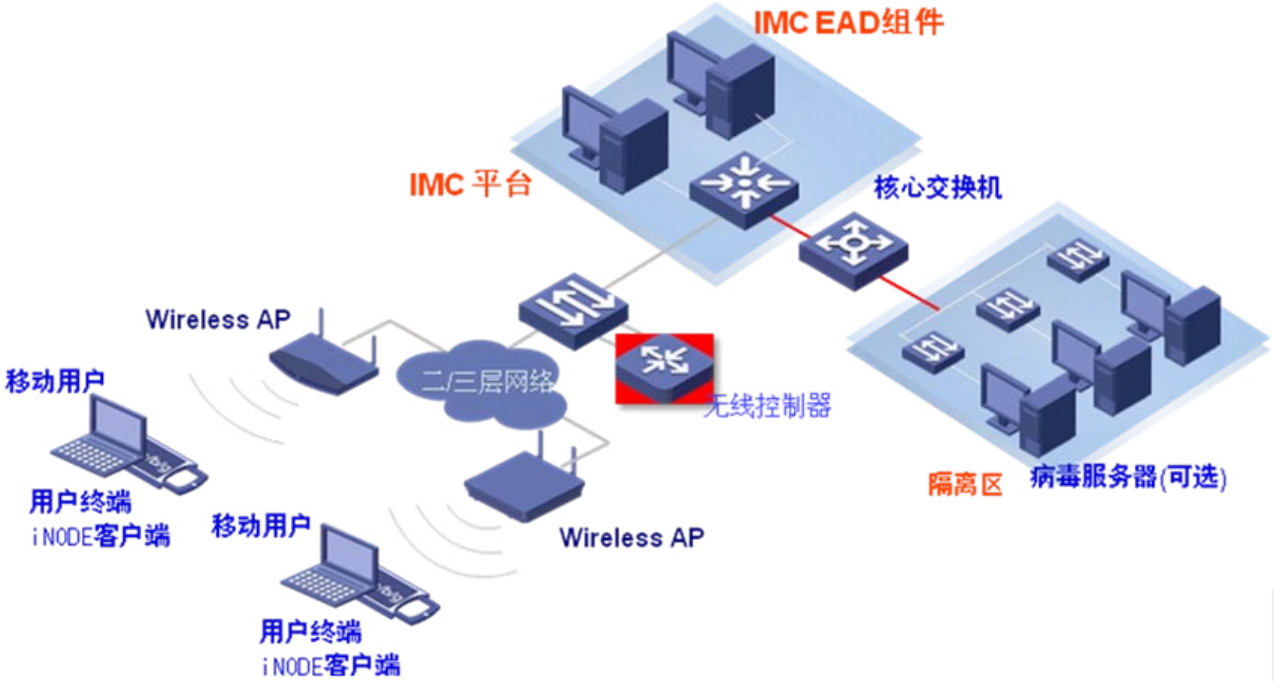


图4 EAD 无线AC旁挂组网

第二种方式是使用交换机的无线控制器AC插卡，直接插到核心或汇聚交换机上，逻辑上这种组网和无线控制器AC旁挂的组网无异，只不过是由无线控制器AC单独机箱变成了插卡。如下图：

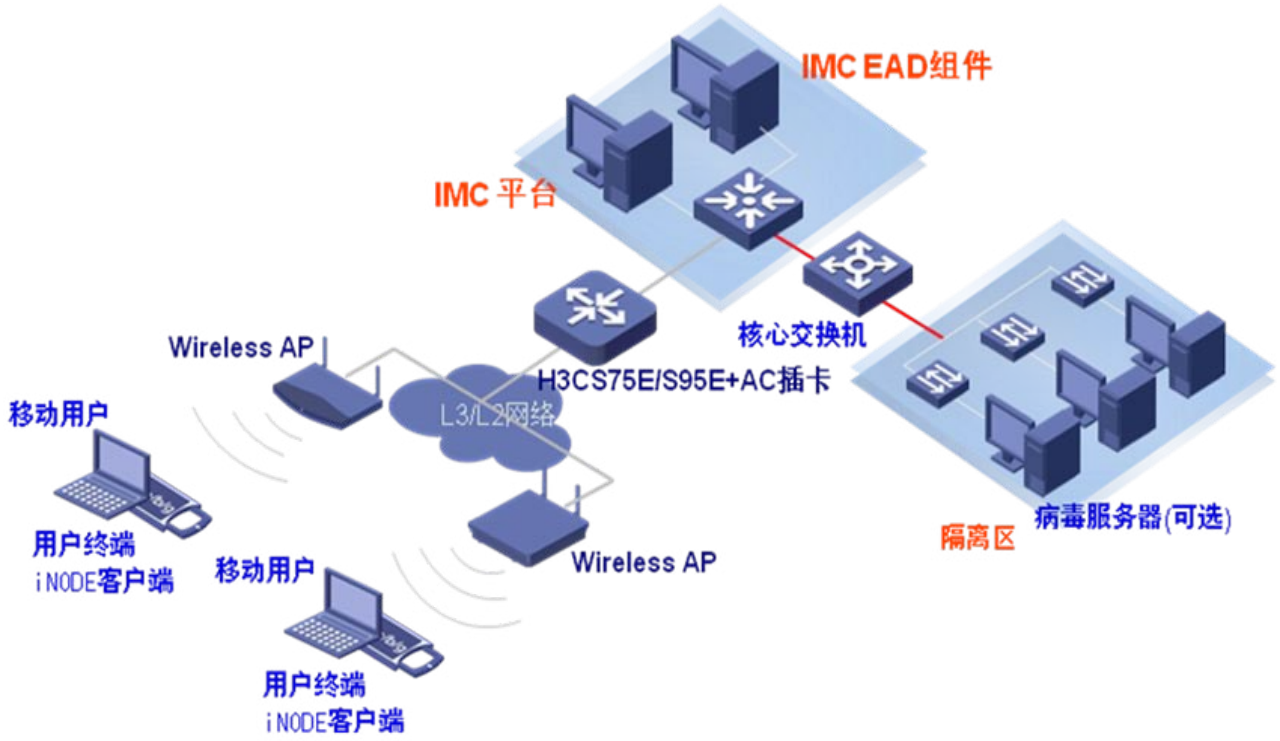


图5 EAD 无线插卡组网

这两种部署方式，虽然物理上无线控制器AC是旁挂，但就EAD接入来讲，其实是直连的，逻辑上EAD终端用户是直接挂在无线控制器AC下，通过认证后进入网络的。但是其扩展性仍然不受影响，和有线EAD的PORTAL网关旁挂扩展性类似。当用户量增加超过无线控制器AC的规格后，可以在网络相同的地方继续旁挂新的无线控制器，然后把新增AP注册到新的无线控制器上，达到扩容的目的。

一些公司随着业务的发展，除了总部，还会出现越来越多的分支机构。而分支机构在业务上也需要访问总部数据，为了保证总部网络的安全，对于分支机构接入企业总部时也需要对其身份合法性和终端PC的安全性进行检查。对于此类企业，可以选择在企业分支的出口路由器上部署PORTAL EAD，用户在访问总部前首先需要通过总部的策略服务器的身份合法性检查和终端PC的安全状况检查，以保障总部网络数据的安全。但如果分支出口路由器不支持PORTAL认证，那么可以在总部路由器广域网接口启动PORTAL认证，由总部的路由器统一对分支访问总部的用户进行认证。这两种部署方式各有优缺点，在各分支出口路由器部署PORTAL认证，PORTAL网关压力下，但其控制比较分散，不利于企业总部管理员的集中管理。如果部署到企业总部的接入路由器上，可便于企业管理员的集中控制，但如果分支节点过多，那么该接入路由器的性能压力较大，而且存在单点故障。实际部署时，可以根据分支用户量的大小及配备的分支路由器的情况灵活把握，在整个网络中，两种方式也可以混和使用。

如果分支机构和总部相连的网络并不是企业私有网络，而是通过Internet连接，或者一些移动用户直接通过终端PC拨号上网，对于这两类用户，适合使用VPN EAD的接入方式。此时在企业总部部署接入认证网关，开启L2TP EAD功能，如果数据敏感可同时在该网关上开启IPSec功能，然后在分支终端或者移动用户的终端PC上安装H3C的iNode客户端，客户端部署L2TP或者L2TPoverIPSec，发起认证前，终端PC先拨号上网，保证可以访问Internet，然后使用iNode，直接向总部发起L2TP和IPSec连接的隧道连接请求，发起身份认证和安全认证。

EAD共有5种模式，隔离模式、监控模式、VIP模式（提醒模式）、下线模式、访客模式。这几种模式指的是针对用户安全检查不通过的时候，几种处理模式。

其中比较常用的是隔离模式，顾名思义，当使用隔离模式时，安全检查不通过的用户将被隔离在特定的区域，该区域内仅有用于修复终端PC的病毒、补丁服务器等资源。终端PC被隔离后，可在隔离区内自动或手动修复，然后重新进行安全认证，认证通过后，就可以访问被授权的安全区域。这种模式适合普通员工使用，可按照部门分配策略，给不同的部门授权访问不同的网络，员工即使通过了安全认证，也只能在授权的网络中进行访问，不能越权访问网络资源。而且此后的网络使用也可以被监控或者审计，终端PC只能安装和运行被允许的软件和进程，否则PC就会被要求下线，禁止访问网络。

监控模式的管理较为宽松，当终端PC安全检查不通过时，终端PC得不到任何提示，仅网管上有记录，提示该终端PC存在的一些安全隐患。如果想保证网络的整体安全，此时需要人为干预。这种模式适合公司访问权限较大的访问者，但一旦出现处于“监控”状态的PC，管理员要进行人为干预。如果公司的管理策略较为宽松，也可以整个公司部署这种模式。监控模式中，也可以对用户进行授权，限制用户的网络访问权限，但这种模式在对用户行为进行监控或审计时，如果有不符合安全策略的行为，只能进行审计，不能要求用户下线或隔离。

VIP模式，在以前的iMC版本中，这种模式被称为提醒模式，即当用户安全检查不通过时，不会被隔离，可以正常访问网络，但策略服务器会向终端用户发起提醒，提示该用户终端PC存在一些安全隐患。这种模式的使用，和监控模式的使用比较类似，管理较为宽松。

下线模式，这种模式对网络的管理是最严厉的，当终端PC安全认证不通过时，会被要求下线，不能访问网络。此模式的使用需要提前对隔离区域进行定义，使用户为通过身份认证前，就可以访问隔离区域，否则用户一旦存在网络安全隐患，被要求下线且无法访问隔离区域的话，就永远不能上线了。这种模式特别适合与PORTAL认证一起使用，可以使用PORTAL的free-rule对隔离区域进行定义，用户无线进行身份认证就可以访问隔离区域资源，进行自我修复。802.1X方式虽然也可以通过FREE-IP定义一个隔离区，允许用户在身份认证前可以访问资源，但这种方式比较浪费设备的ACL资源，不推荐使用。可能有的客户认为隔离区域的资源也需要被保护，如果用户不认证就能访问这些资源，那么这些服务器还是存在一定的安全隐患。对于这种需求，虽然定义安全策略，在要求用户下线前等待十几分钟，给用户修复PC留一点时间，对隔离区的资源进行了保护，但不得不说，不安全的终端PC提前进入安全区域，对安全区域也是一种隐患。建议使用下线模式时，使用FREE-RULE或FREE-IP的方式定义隔离区域，用户不安全时被要求立即下线。如果用户的网络不用授权，所有用户访问的资源相同，那么使用PORTAL的下线模式，可以最大化的起到节省ACL资源的目的。

访客模式，这种模式适合给来公司的访客人员使用，通常情况下，访客的终端PC都不会符合公司的安全策略，因此必然需要进行修复，基于这种考虑，访客模式下，如果终端PC不符合安全策略，则先会提示，要求进行修复，几分钟（可以设置）后，再次进行安全检查，如果仍然不符合要求，则要求用户下线。

除系统预定义的这几种模式外，管理员可以根据自己的需求自定义最适合自己公司使用的模式。

认证细节

802.1X EAD认证细节

802.1X的认证细节前文已有详细描述，此处不再赘述，本小节重点讲述802.1X和EAD认证相关的细节。

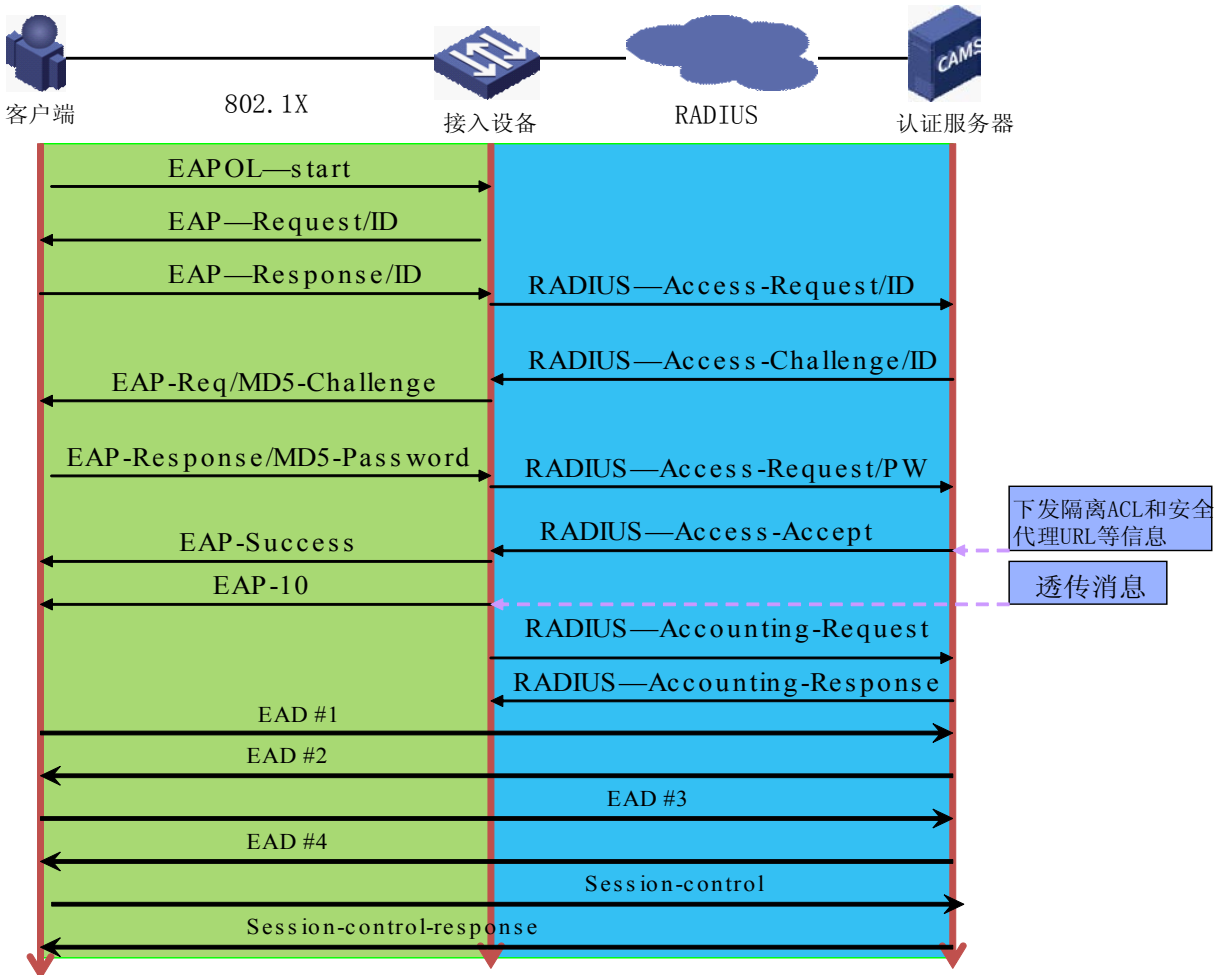


图6 EAD 802.1X认证流程

802.1X认证过程如上图所示，和普通802.1X认证不同之处在于后续的EAD认证过程。如果在策略服务器上配置的是EAD的隔离模式，那么在用户身份认证通过后，策略服务器通过RADIUS-access-accept报文告知接入认证NAS设备该用户认证通过，且携带一个filter-id属性，告知接入认证设备为该认证用户下发ACL，此ACL即为该用户的隔离ACL，此时用户被隔离，只能访问隔离区域，该报文中还会携带一个要求NAS设备透传的属性，该属性中包含了策略服务器地址等信息，接入认证NAS设备把该属性封装为eap报文后透传给客户端，客户端根据该报文中的策略服务器信息发起EAD认证。如下图：

```
No.  Time      Source        Destination   Protocol  Info
  1  10:50:3   90.13.0.2     192.168.0.86  RADIUS    Access-Request(1) (id=112, l=287)
  2  10:50:3   192.168.0.86  90.13.0.2     RADIUS    Access-challenge(11) (id=112, l=80)
  3  10:50:3   90.13.0.2     192.168.0.86  RADIUS    Access-Request(1) (id=113, l=290)
  4  10:50:3   192.168.0.86  90.13.0.2     RADIUS    Access-Accept(2) (id=113, l=280)
  5  10:50:3   90.13.0.2     192.168.0.86  RADIUS    Accounting-Request(4) (id=80, l=258)
  6  10:50:3   192.168.0.86  90.13.0.2     RADIUS    Accounting-Response(5) (id=80, l=62)
  7  10:50:3   90.13.0.2     192.168.0.86  RADIUS    Accounting-Request(4) (id=81, l=312)
  8  10:50:3   192.168.0.86  90.13.0.2     RADIUS    Accounting-Response(5) (id=81, l=38)
  9  10:50:3   90.13.0.101   192.168.0.86  EAD
 10  10:50:3   192.168.0.86  90.13.0.101   EAD

⊞ User Datagram Protocol, Src Port: radius (1812), Dst Port: 5001 (5001)
⊟ Radius Protocol
    Code: Access-Accept (2)
    Packet identifier: 0x71 (113)
    Length: 280
    Authenticator: D18CD4C1B668D3BED52B471DE7714558
  ⊟ Attribute Value Pairs
    ⊞ AVP: l=6  t=Service-Type(6): Framed-User(2)
    ⊞ AVP: l=10  t=State(24): 4F4B755037384D66
    ⊞ AVP: l=6  t=Termination-Action(29): Default(0)
    ⊞ AVP: l=6  t=Filter-Id(11): 2001                    下发隔离ACL
    ⊟ AVP: l=6  t=Session-Timeout(27): 30331
        Session-Timeout: 30331
    ⊞ AVP: l=6  t=Acct-Interim-Interval(85): 600
    ⊟ AVP: l=12  t=Vendor-Specific(26) v=HUAWEI Technology Co.,Ltd(2011)
      ⊟ VSA: l=6 t=Unknown-Attribute(26): 00000172
          Unknown-Attribute: 00000172                    透传策略服务器相关信息
    ⊟ AVP: l=184  t=Vendor-Specific(26) v=HUAWEI Technology Co.,Ltd(2011)
      ⊟ VSA: l=178 t=Unknown-Attribute(61): 360600000000370600000000380600000000003977543D303B...
          Unknown-Attribute: 360600000000370600000000380600000000003977543D303B...
    ⊞ AVP: l=6  t=EAP-Message(79) Last Segment[1]
    ⊞ AVP: l=18  t=Message-Authenticator(80): D4DB73664BBCC05B967F25308DB1ACFE
```

图7 EAD 802.1X认证透传信息

iNode客户端收到透传的EAP报文后，解析出策略服务器的地址、端口等信息，开始发起EAD认证，EAD认证报文共2对。1、2、3、4号报文。1号报文是认证请求，包含主要内容如下：认证用户名、IP地址（可选）、客户端监听端口号、客户端版本号等。2号报文是服务器回应的安全检查报文，它需要通知客户端哪些安全项需要进行检查并且上报；通告客户端补丁服务器信息、补丁检查模式等补丁信息；通告客户端策略服务器对病毒软件版本要求、联动模式等；通告客户端一些软件黑白名单等信息。此外还携带了心跳相关信息、服务器版本等。如果有一些系统通知、广告URL等，也是通过2号报文携带的。EAD3号报文是用于向策略服务器反馈终端PC的一些安全检查结果的。客户端根据策略服务器的要求，对本终端安全检查结果进行搜集，然后通告EAD3号报文反馈给策略服务器。该报文携带内容包含用户名属性、黑、白软件名单、杀毒软件版本引擎等信息、补丁信息等。

最后策略服务器根据客户端3号报文的结果，给终端PC的一个最终反馈结果，如果安全检查通过，则直接向客户端发送EAD 4号报文。如果安全检查不通过，则向终端PC发送EAD 6号报文。在EAD6号报文中，有一个属性标识不符合安全检查策略的用户，可以正常使用网络的时间，默认情况下该属性值为0。管理员可以在策略服务器上进行设置，允许安全检查不通过的用户使用非受限网络的时长。

一次正常的上线过程，当策略服务器发现用户安全检查结果为通过时，向终端PC发送EAD4号报文。如果是隔离模式，那么策略服务器会同时向接入认证设备发送报文用于通告接入设备向该用户授权，使该用户可以正常访问其被授权的网络资源。如下图：

No.	Time	Source	Destination	Protocol	Info
5	10:50:3	90.13.0.2	192.168.0.86	RADIUS	Accounting-Request(4) (id=80, l=258)
6	10:50:3	192.168.0.86	90.13.0.2	RADIUS	Accounting-Response(5) (id=80, l=62)
7	10:50:3	90.13.0.2	192.168.0.86	RADIUS	Accounting-Request(4) (id=81, l=312)
8	10:50:3	192.168.0.86	90.13.0.2	RADIUS	Accounting-Response(5) (id=81, l=38)
9	10:50:3	90.13.0.101	192.168.0.86	EAD	
10	10:50:3	192.168.0.86	90.13.0.101	EAD	
11	10:50:3	90.13.0.101	192.168.0.86	EAD	
12	10:50:3	192.168.0.86	90.13.0.101	EAD	
13	10:50:3	192.168.0.86	90.13.0.2	RADIUS	Unknown Packet(20) (id=1, l=78)
14	10:50:3	90.13.0.2	192.168.0.86	UDP	Source port: 5001 Destination port: 9033

```
⊞ Frame 13 (120 bytes on wire, 120 bytes captured)
⊞ Ethernet II, Src: Vmware_7a:c0:d2 (00:0c:29:7a:c0:d2), Dst: Hangzhou_2e:50:9e (00:0f:e2:2e:50:9e)
⊞ Internet Protocol, Src: 192.168.0.86 (192.168.0.86), Dst: 90.13.0.2 (90.13.0.2)
⊞ User Datagram Protocol, Src Port: 9033 (9033), Dst Port: radius (1812)
⊟ Radius Protocol
    Code: Unknown (20)
    Packet identifier: 0x1 (1)
    Length: 78
    Authenticator: 7192CEA72477828BC190BDEC20E248E1
  ⊟ Attribute Value Pairs
    ⊞ AVP: l=16  t=Acct-Session-Id(44): 110003140021aa
    ⊟ AVP: l=6  t=Filter-Id(11): 2000
        Filter-Id: 2000
    ⊞ AVP: l=12  t=Vendor-Specific(26) v=HUAWEI Technology Co.,Ltd(2011)
    ⊞ AVP: l=12  t=Vendor-Specific(26) v=HUAWEI Technology Co.,Ltd(2011)
    ⊞ AVP: l=12  t=Vendor-Specific(26) v=HUAWEI Technology Co.,Ltd(2011)
```

图8 EAD session-control报文

该报文是RADIUS的session-control报文，通过filter-id向接入认证通告该用户应该被授权下发的ACL，通过ACL的约束来达到对用户授权的目的。接入认证设备下发ACL成功后，需要向策略服务器发送一个session-control应答报文，确认下发成功，策略服务器收到该报文后，将该用户的状态置为“安全”状态。

此后，在用户下线前，EAD终端PC和策略服务器会通过EAD 11号报文和12号报文来维持心跳保活。该报文内容比较简单，携带username属性、地址信息和序列号。

当用户主动请求下线时，向策略服务器发送EAD13号报文，策略服务器会回应EAD14号报文，其携带属性和EAD11号报文大致相同。

一个用户从进行EAD认证到该用户EAD下线，其发送的EAD报文中，都携带一个属性eventseqID，用于表示本次认证过程，其值是独一无二的。

>> PORTAL认证细节

PORTAL EAD的认证所使用的EAD报文相同，不过认证过程有所不同。本小节讲描述PORTAL EAD的详细认证过程。

PORTAL 认证分为PORTAL客户端、NAS服务器、PORTAL服务器、RADIUS服务器，其详细的交互过程及报文中所保护含义，在前面章节已有详细描述，本小节不再充分，仅讲述与EAD认证相关的部分。

PORTAL EAD认证上线过程如下图：

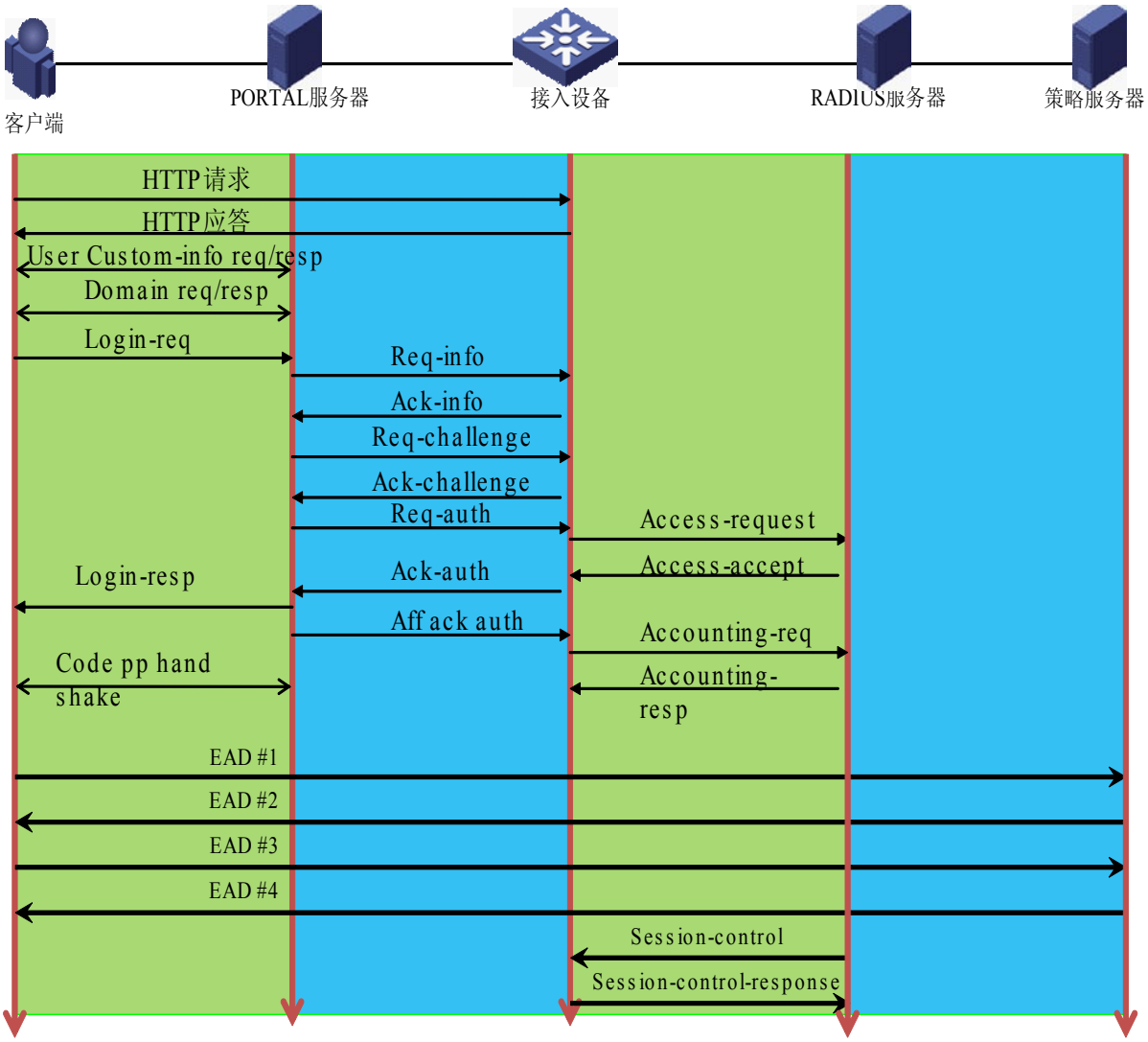


图9 EAD PORTAL认证流程

与802.1X相同，当用户身份认证通过后，如果用户安全策略的模式是隔离模式，那么RADIUS服务器会通过access-accept报文中的filter-id属性下发隔离ACL。PORTAL服务器通过CODE-PP-LOGIN-RESPONS 报文通告终端PC的iNode安全策略服务器的IP地址、端口号等信息，如下图：

```
 4 14:27:42.163512 60.1.1.2      1.1.1.5      PORTAL Type ACK_INFO
 5 14:27:42.182912 1.1.1.5       60.1.1.2     PORTAL Type REQ_CHALLENGE
 6 14:27:42.354136 60.1.1.2      1.1.1.5      PORTAL Type ACK_CHALLENGE
 7 14:27:42.382860 1.1.1.5       60.1.1.2     PORTAL Type REQ_AUTH
 8 14:27:42.554571 60.0.0.2      1.1.1.5      RADIUS Access-Request(1) (id=63, l=
 9 14:27:42.579331 1.1.1.5       60.0.0.2     RADIUS Access-Accept(2) (id=63, l=1
10 14:27:42.754109 60.1.1.2      1.1.1.5      PORTAL Type ACK_AUTH
11 14:27:42.754731 60.0.0.2      1.1.1.5      RADIUS Accounting-Request(4) (id=51
12 14:27:42.770556 1.1.1.5       60.1.1.2     PORTAL Type AFF_ACK_AUTH
13 14:27:42.770698 1.1.1.5       60.1.1.3     PORTAL Type CODE_PP_LOGIN_RESPONSE
14 14:27:42.770812 1.1.1.5       60.0.0.2     RADIUS Accounting-Response(5) (id=5
    Attribute Length: 0x0a
    EX_ATTR_USER_ID: 390982C01E82CA72
    <---Attribute Type--->: EX_ATTR_PORTAL_URL (0x5e)
    Attribute Length: 0x1c
    EX_ATTR_PORTAL_URL: http://1.1.1.5:8080/portal
    <---Attribute Type--->: EX_ATTR_USERSELFHELP_URL (0x5f)
    Attribute Length: 0x02
    <---Attribute Type--->: EX_ATTR_PORTAL_SERVER_ADDR (0x61)
    Attribute Length: 0x09
    EX_ATTR_PORTAL_SERVER_ADDR: 1.1.1.5
    <---Attribute Type--->: EX_ATTR_PORTAL_SERVER_PORT (0x62)
    Attribute Length: 0x06
    EX_ATTR_PORTAL_SERVER_PORT: 50100
    <---Attribute Type--->: EX_ATTR_EAD_AGENT_IP (0x3e)
    Attribute Length: 0x06
    EX_ATTR_EAD_AGENT_IP: 1.1.1.5 (1.1.1.5)
    <---Attribute Type--->: EX_ATTR_EAD_AGENT_PORT (0x3f)
```

图10 EAD PORTA认证策略服务器信息

然后客户端发起EAD安全认证过程，此后过程和802.1X认证相同，此处不再赘述。EAD认证通过后，PORTAL认证也是通过session-control报文进行下发安全ACL，如下图：

```
19 14:27:51.567573 60.1.1.3          1.1.1.5          EAD
20 14:27:51.623764 1.1.1.5           60.1.1.3         EAD
21 14:27:52.234440 60.1.1.3          1.1.1.5          EAD
22 14:27:52.270887 1.1.1.5           60.1.1.3         EAD
23 14:27:52.323083 1.1.1.5           60.0.0.2         RADIUS Unknown Packet(20) (id=128, l=77)
24 14:27:52.444541 60.0.0.2          1.1.1.5          UDP    Source port: 1024  Destination port: 1380

⊞ Frame 23 (119 bytes on wire, 119 bytes captured)
⊞ Ethernet II, Src: Intel_e4:b2:48 (00:0e:0c:e4:b2:48), Dst: HuaweiTe_17:e5:f6 (00:e0:fc:17:e5:f6)
⊞ Internet Protocol, Src: 1.1.1.5 (1.1.1.5), Dst: 60.0.0.2 (60.0.0.2)
⊞ User Datagram Protocol, Src Port: 1380 (1380), Dst Port: radius (1812)
⊟ Radius Protocol
    Code: Unknown (20)
    Packet identifier: 0x80 (128)
    Length: 77
    Authenticator: E9C6D7107F16C7EB7776B6097015CA3B
  ⊟ Attribute Value Pairs
    ⊞ AVP: l=15  t=Acct-Session-Id(44): 108071814283e
    ⊞ AVP: l=6  t=Filter-Id(11): 3000
    ⊞ AVP: l=12  t=Vendor-Specific(26) v=HUAWEI Technology Co.,Ltd(2011)
    ⊞ AVP: l=12  t=Vendor-Specific(26) v=HUAWEI Technology Co.,Ltd(2011)
    ⊞ AVP: l=12  t=Vendor-Specific(26) v=HUAWEI Technology Co.,Ltd(2011)
```

图11 EAD PORTAL认证session-control报文

至此用户上线认证完成。

>> EAD新特性介绍

客户端ACL介绍

EAD解决方案的亮点之一，是可以对受感染或存在安全隐患的终端PC进行隔离和对通过安全检查的终端进行授权。

但是在推广中，也经常遭遇一些尴尬。比如，接入设备是友商设备。我们下发ACL使用的都是私有属性，和友商配合做EAD时，就无法进行隔离和授权了，方案效果打了一个很大的折扣。即使是我司的交换机设备，也会受限于芯片ACL的资源，比如用户要求授权的ACL条目太多，“每用户ACL”乘以“用户数”大于芯片ACL资源，这时候也不满足部署条件，而能够供用户选择的选项就是要么买更多的设备或更贵的设备，要么减少授权ACL的数量。

为了改善这种场景下EAD的实施效果，我司为EAD开发了新的特性，客户端ACL。所谓客户端ACL，顾名思义就是在客户端上下发ACL，此时接入设备需要做的就是做一个基本的802.1X认证。

客户端ACL的流程和基本的EAD流程很类似，用户首先通过身份认证，然后iNode客户端发起安全认证。在安全认证过程中，新增了客户端ACL的相关报文，iMC会把ACL的规则通告给iNode客户端，然后客户端首先下发隔离ACL到终端网卡驱动，然后进行EAD的安全检查，最后根据安全检查结果来判断是否下发安全ACL到网卡驱动。

客户端ACL和普通的EAD认证流程不同之处在于：

1. iMC上需要配置具体的ACL规则，而不再是一个ACL号；

2. 新增了EAD 61 62号报文；

3. 认证回应中无需通过filter-id向设备下发隔离ACL；

4. 安全认证通过后，对用户授权时，RADIUS服务器无需向设备发送session-control报文。

客户端ACL所带来的好处显而易见：首先，ACL的下发不再依赖于接入设备，因此接入设备无论是友商的还是我司设备，都不再重要，而且ACL资源也不再受限于接入设备的芯片资源。其次，ACL配置是在iMC上，配置和维护比较方便。

客户端ACL特性，使EAD解决方案在推广中，适应性更强，为我司设备的推广也做好了铺垫。

PORTAL 逃生

在PORTAL EAD解决方案中，无论PORTAL网关是旁挂模式还是普通的直连模式，用户在上网前都被强制到PORTAL网关上做PORTAL认证，可一旦PORTAL服务器故障或者PORTAL网关到PORTAL服务器网络不可达时，用户都会认证失败而导致无法上网。而且已经通过认证的用户则会在PORTAL网关上吊死。对于一些控制比较严格的企业，这种情况符合他们的预期，但有些企业控制不是特别严格，他们更期望服务器故障时，不要影响用户的上网业务，因此这时候我们就需要一个解决方案来解决PORTAL服务器故障时用户无法通过PORTAL认证而导致用户无法上网的问题。

PORTAL逃生方案就是用来解决PORTAL服务器不可达所带来的问题。PORTAL逃生方案中，PORTAL网关和PORTAL服务器之间要维持两个心跳信息，一个心跳是用于PORTAL网关和PORTAL服务器之间的保活（keepalive），另外一个心跳（或者叫同步信息更确切）是用于PORTAL网关和PORTAL服务器间的用户同步。PORTAL网关如果在3个心跳周期内都没有收到PORTAL服务器的心跳信息，那么它就进入PORTAL逃生状态（免认证），此时所有用户无需通过PORTAL认证，就可以访问网络。在逃生状态下，一旦收到PORTAL服务器的心跳信息，PORTAL网关马上切换到普通认证模式。而用户同步信息是用来解决用户挂死的情况，当PORTAL服务器发过来的在线用户信息和PORTAL网关不一致时，PORTAL网关会把交集外的用户全部下线，防止用户挂死。

>> EAD方案现状

EAD解决方案分为802.1X认证方案，无线认证方案，PORTAL网关认证方案，VPN认证方案四种形式，而802.1X方案控制效果好，对网络的保护最大话，这种方案对交换机依赖较大。PORTAL方案对接入网络没有要求，而且我司推出的PORTAL网关旁挂模式，对用户的网络改动非常小，而且可扩展性强。对于分公司分散全国各地，又要求访问总部进行认证的企业，我司的广域网EAD可以很好的满足。PORTAL网关部署到总部，利于集中管理，而且对分支部门的接入设备降低了要求，因此降低了用户的网络成本。

目前EAD解决方案在我司的销售中，常常做为我司销售的攻坚利器，在友商设备云集的环境中占用一席之地，进而为我司设备后续的进入打下良好基础。■

EAD多客户端模拟工具

■文/饶 云

测试工具的基本概念

测试工具的定义

在软件测试中，我们常常发现很多手工难于实施或根本无法实施的测试，如参数遍历测试，工作量大、重复性强、又容易出错；再如某些特定的测试项在时间、精度、计算量等方面超过了手工可达的范围，这时候就需要测试工具来解决。测试工具是能帮助我们提高软件测试深度和广度的特殊程序，本文的测试工具专指运行在PC上的、纯软件的、数据通信领域的测试工具，简称数通测试工具。

测试工具的适用场景

A. 需要大量数据或大量并发用户的测试，如RADIUS服务器的多用户接入认证测试。

B. 与时序、死锁、资源冲突、多线程等有关的测试，如网络设备的HTTP连接的新建能力测试。

C. 长时间、高强度的测试，如可能需要持续数年的可靠性测试。

测试工具的分类

测试工具分类图

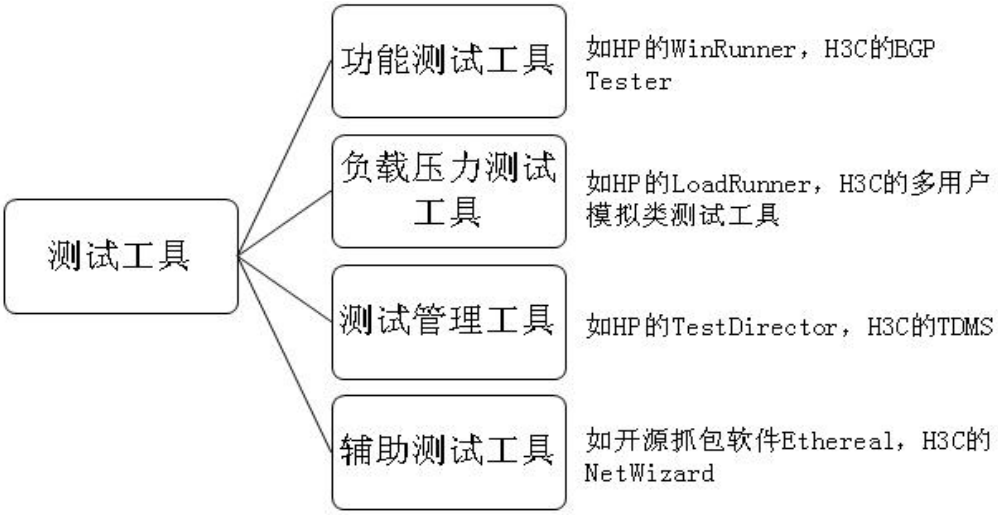


图1 测试工具分类图

功能测试工具

功能测试工具的主要目的是检测应用程序是否能够达到预期的功能并正常运行，它的工作原理是先和SUT (System Under Test ，待测系统) 进行一定的交互或在SUT上进行一定的应用操作，然后将输出同预先给定的标准结果做比较。

功能测试工具常用于功能测试、异常测试、协议一致性测试等测试活动中。它的代表有HP的WinRunner，H3C的BGP Tester等也属于功能测试工具。

负载压力测试工具

负载压力测试工具的主要目的是度量应用系统的可扩展性和性能，是一种预测系统行为和性能的测

试工具，它以“多用户、并发、长时间、大流量、频繁操作”等方式给SUT负载压力。在实施负载压力过程中，通过实时性能监测和测试记录来确认和查找问题，并针对所发现的问题对系统性能进行优化或找到系统的最佳部署方案（如修改参数的默认值，使系统的默认性能更优）。

负载压力测试工具常用于性能测试、规格测试、稳定性测试等测试活动中。它的代表有HP的LoadRunner，H3C的多用户模拟类测试工具也属于负载压力测试工具。

测试管理工具

测试管理工具用于对测试计划、测试用例、测试实施等进行管理，还可包括缺陷跟踪管理等功能。它的代表有HP的TestDirector，H3C的TDMS系统也属于测试管理工具。

测试辅助工具

辅助测试工具本身不参与测试，但和测试过程相关，能提升测试的效率。如开源的Ethereal抓包软件、H3C的NetWizard和VTP。

H3C自主测试工具开发概况

H3C一直很重视自主测试工具的开发，目前已开发了数十个测试工具，广泛应用于各产品测试，并获得普遍好评。

工具种类

按工具功能划分

A. 功能测试工具

B. 负载压力测试工具

C. 测试管理工具

D. 测试辅助工具

按应用领域划分

A. 交换测试工具

B. 路由测试工具

C. 业务软件测试工具

D. 安全测试工具

E. 存储测试工具等

按网络环境划分

A. 有线网络环境测试工具

B. 无线网络环境测试工具

工具特点

极大地支持了私有特性的测试

自主开发的测试工具是私有特性测试的唯一选择，并能及时和私有特性更新保持同步。

可定制性高、需求响应快

广泛收集工具需求，整个工具生命周期里均可对工具进行修改，可满足不断变化的特性要求和功能变更。

跨平台

跨平台指的是同一套程序，可以不经修改的运行于Windows和Linux操作系统，它能极大提升工具开发效率、方便测试工具的部署。

和自动化测试的融合

和自动化测试的融合包括如下3个方面：

A. 和自动化平台ATF（Auto Test Framework，自动测试框架）的完美融合。

B. 测试工具内嵌TCL解释器。

C. 测试工具以动态链接库的形式提供对外接口，允许TCL脚本调用测试工具的功能。

专有技术

H3C的测试工具开发，通过一系列的专有技术，如虚拟协议栈技术、多TCP网络节点实现新方法等，突破了工具开发的一些瓶颈，扩展了工具开发的空间。

虚拟协议栈技术：测试工具不使用操作系统提供的TCP/IP协议栈，而是工具自己实现一个虚拟协议栈，成功实现了多个网络节点的模拟，打开了多用户模拟类的负载压力测试工具开发的广阔天空。

多TCP网络节点实现新方法：传统上，测试工具开发有一个困难，就是如何实现多个MAC/IP地址不同的TCP网络节点。H3C自创的多TCP网络节点实现新方法解决了这一难题，能够实现上万个MAC/IP地址不同的TCP网络节点，TCP协议栈也稳定、高效、互通能力强。

EAD多客户端模拟工具

在H3C的自主测试工具开发中，负载压力测试工具是开发最多、最被产品所需、开发模式最成熟的工具类型之一。本章节讲解一个H3C测试工具的优秀实践：EAD多客户端模拟工具，总体上它属于负载压力测试工具，但也包含一些功能测试工具的特性。

EAD多客户端模拟工具在H3C的应用非常广、服役时间也很长，除了工具本身功能强、能及时与EAD解决方案保持同步等原因外，还非常得益于测试人员不断提出细致的、贴近测试的工具需求，这推动了测试工具的不断改进和优化。

工具介绍

网络安全问题的解决，三分靠技术，七分靠管理，严格管理是企业、机构及用户免受网络安全问题威胁的重要措施。为了解决现有网络安全管理中存在的不足，应对网络安全威胁，H3C推出了EAD（Endpoint Admission Domination，终端准入控制）解决方案。该方案从用户终端准入控制入手，整合网络接入控制与终端安全产品，通过安全客户端、安全策略服务器、网络设备以及防病毒软件产品、系统补丁管理产品、桌面管理产品的联动，对接入网络的用户终端强制实施企业安全策略，严格控制终端用户的网络使用行为，可以加强用户终端的主动防御能力，大幅度提高网络安全。

很显然，EAD解决方案的管理对象是用户终端。由于用户终端的如下特点，EAD解决方案迫切需要一款测试工具来协助进行大用户量、高强度、长时间、测试参数灵活多变的测试。

A. 数量庞大；

B. 环境复杂，如终端安装的软件千差万别、运行的进程随时都可能变化；

C. 由于数量庞大，所以几乎任何时候都有终端在线；

D. 终端上线、下线等操作时机不固定，既有上下班时候的集中操作，也有平时的零星操作。

于是，EAD多客户端模拟工具应运而生，它可模拟上万个EAD客户端对SUT(包括设备和IMC)进行性能、规格、稳定性等测试，它的具体功能如下：

A. 模拟上万个数据和状态完全独立的客户端；

B. 可设置单一客户端的各种属性，如MAC/IP地址、客户端版本号、安装的软件列表等；

C. 客户端支持多种类型的用户接入认证：802.1x、Portal、L2TP VPN；

D. 每个客户端可进行EAD安全认证；

E. 支持EAD安全认证的有效性检查；

F. 支持动态改变客户端的安全状态；

G. 客户端收发的报文支持VLAN；

H. 支持802.11和LWAPP报文封装，以对无线AC进行测试。

同时，为了方便测试工具的部署，EAD多客户端测试工具可在Windows和Linux平台上运行。

工具的四大目标及其设计方案

四大目标

A. 工具要能产生大的负载压力，狂轰滥炸SUT。

B. 工具要能控制测试细节，精确打击SUT。

C. 提供一些管理功能，方便测试过程的监控和记录。

D. 提供一些附加功能，方便工具部署、扩展工具运用。

设计方案

1. 产生大的负载压力，狂轰滥炸SUT

在规格测试、性能测试、稳定性测试等测试活动中，就必须要负载压力。

EAD多客户端模拟工具主要通过如下4个手段产生大的负载压力：

A. 模拟多个客户端

通过一系列的专用技术，成功模拟了上万个客户端，这些客户端的数据和状态完全独立（如每个客户端都有自己的MAC/IP地址、定时器），以给SUT真实的多用户压力。

B. 多个模拟客户端并发上下线

EAD多客户端模拟工具通过多线程控制客户端的上下线，以给SUT并发压力。

C. 模拟客户端频繁上下线、长时间运行

模拟客户端可重复上千次“上线、下线”过程，认证失败后还可自动重试；测试工具可连续运行数月，以给SUT持续压力。

D. 认证通过后，发送大流量

认证通过后，测试工具可发送UDP流量（已测试到的最大流量是400M b/s），给SUT上层以压力。

2. 控制测试细节，精确打击SUT

在功能测试、协议一致性测试、异常测试等测试活动中，就常常需要对测试细节的可控。

EAD多客户端模拟工具主要通过如下3个方面实现对测试细节的可控：

A. 可设置单一客户端的参数，如MAC/IP地址、安装的软件等。

B. 设置客户端参数时，允许输入一些异常值，如用户名的长度超过定义、不正确的客户端版本等。

C. 可设置上下线参数（如在线时长、上线总次数）、并发参数、流量发送参数等进行设置。

D. 支持报文重传、认证失败后重试等。

3. 提供一些管理功能，方便测试过程的监控和记录。

管理功能主要包括日志记录和实时统计功能。日志记录能将包括交互报文在内的运行时信息显示给用户，也提供文件支持。实时统计功能是实时地对收发的报文、流量、客户端状态进行统计，可打印在工具界面上，也提供文件支持。

4. 提供一些附加功能，方便工具部署、扩展工具运用

EAD多客户端模拟工具中的附加功能有跨平台和支持VLAN两个特性。

跨平台是指测试工具可运行在Windows平台上、也可运行在Linux平台上，为提供最大性能，测试工具还可运行在H3C的Ferrari测试仪器上。

测试工具收发的报文支持VLAN，也极大地方便了工具部署，扩展了工具应用。在支持VLAN之前，测试工具只能连接SUT的一个接口，进而导致流量只通过这一个端口进出SUT，这在一定程度上可以说是对单端口的测试，而不是对整个SUT的测试，如下图：

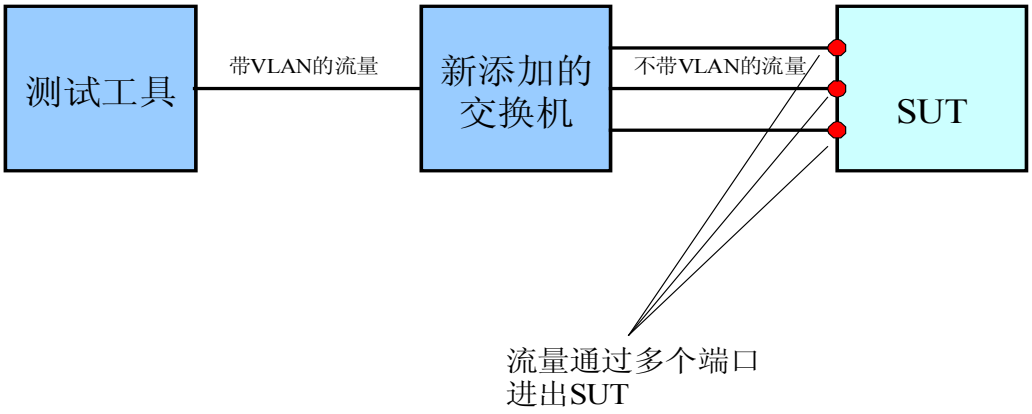


图2 测试工具和SUT的连接情况（支持VLAN之前）

在支持了VLAN之后，再外加一个交换机，所有流量就可以通过多个端口进出SUT，真正实现对整个SUT的测试，如下图：

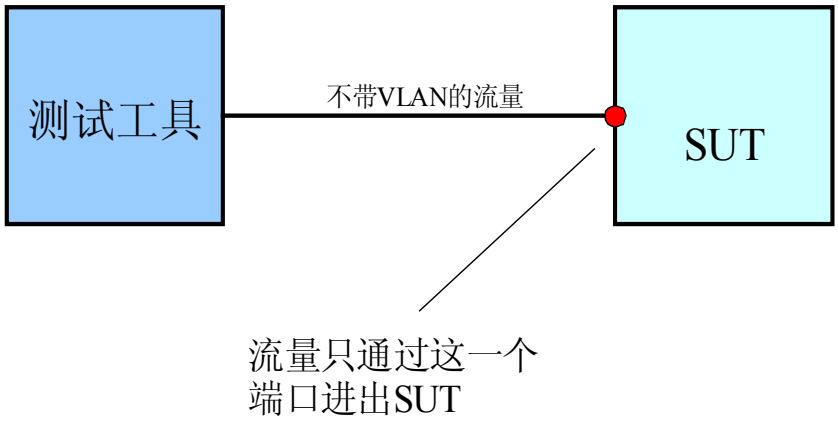


图3 测试工具和SUT的连接情况（支持VLAN之后）

测试工具发送携带VLAN Tag的流量，该流量被交换机转发时根据VLAN Tag找到对应的转发口，并且转发时去掉VLAN Tag。SUT发送不携带VLAN Tag的流量，该流量被交换机抓发时根据报文入端口找到对应的VLAN Tag，并在转发时给报文添加VLAN Tag。

总结

正如工具的使用是人类进化史上的重要一步，测试工具的开发和使用也是软件测试发展的重要一环，它有助于提高软件测试的深度和广度，提高软件测试的效率，体现了软件测试水平的高低。■

测试中心
TESTCENTER

网络之路-安全接入专题
ROUTE TO NETWORK

2011年第1季度，总第十五期

主办单位：H3C测试中心
策　　划：刘　宇 陈旭盛　杜祥宇
主　　编：陆宇翔
本期编委：徐庆伟　袁亚屏　李　丹
邓宏波　曾　劼　袁　江
盛冬旭　李朝辉　王　瑞
王君菠　饶　云

晚镜但愁云鬓改 夜吟应觉月光寒
——唐·李商隐