阶段 2：主要进行协商 SA 和协商用于保护用户数据的密钥协商（如 IPSec SA），阶段 2 的协商消息报文是在阶段 1 IKE SA 的保护下进行的。阶段 2 的模式为快速模式。

20.3　Kerberos

1．概述

Kerberos 身份认证服务是目前较为著名、也相对较为成熟的一种身份认证机制。Kerberos 协议是一种应用于开放式网络环境，基于可信任第三方和 TCP/IP 的网络安全认证协议，其认证模型基于 Needham-Schroeder，以加密为基础，对用户及网络连接进行认证，提供增强网络安全的服务。

该协议是美国麻省理工学院（MIT）“雅典娜项目”（Project Athena）的一部分，使用 DES 来进行加密和认证。至今，Kerberos 协议已经有了 5 个版本，Kerberos v4 是被公诸于众的第一个版本，它是分布式计算机环境框架的组成部分，已在一些 UNIX 系统中得到应用。

Kerberos 的设计是针对以下要求实现的：

① 安全性：网络中的窃听者不能获得必要的信息来假冒网络的用户。

② 可靠性：对基于 Kerberos 协议的所有服务来说，Kerberos 系统的瘫痪则意味着它所支持的所有服务的瘫痪。

③ 透明性：用户除了被要求输入密码外，不会觉察出认证的进行过程。

④ 可扩充性：系统应能够支持更多数据的用户和服务器。

Kerberos 的设计目的分为 3 个领域：认证、授权、记账，可用于构建安全网络，它提供了可用于安全网络环境的认证机制和加密工具。Kerberos 协议已成为业界的标准网络身份验证协议，得到了越来越广泛的应用。微软（Microsoft）公司操作系统 Windows 2000 中提供了对 Kerberos v5 的支持，Linux Redhat 环境下也可以使用 Kerberos 提供的 Ktelnetd、Krlogind、Krshd 来替代传统的 telnetd、rlogind、rshd 服务，Java 安全解决方案中的 Java 认证和权限服务（JAAS，Java Authentication and Authorization Service）也提供了对 Kerberos 的支持。

Kerberos 的目标是将认证从不安全的工作站转移到集中的认证服务器（Authentication Server），服务器在物理上是安全的，并且其可靠性也是可控制的。必须保证用户的密码不能以明文形式在网络中传输，每个用户和服务都是一个密码，只有认证服务器拥有所有用户和服务的密码。

2．基本原理

Kerberos 协议通过提供认证中心服务，并应用对称密码体制 DES，在客户机和服务器之间构造起了一个安全桥梁。针对用户向服务器提交的每个服务请求及其权限，要求用户必须预先经过认证中心服务器的认证合格后，才能被指定服务器所执行，这样在很大程度上消除了许多潜在的安全问题。

Kerberos 协议是以可信赖第三方为基础的认证协议，有域内认证和域间认证两种模式。域间认证模式的基本原理与域内认证类似。如果客户端要访问不在同一个域内的服务器，就必须从另一个域的 TGS 获得该服务器的票据。在两个 Kerberos 域之间建立信任关系，域之间通过这两个域间的密钥来加密之间传送的数据。以后的过程则与域内认证过程相同。

域内认证模型认证过程如图 20-9 所示。

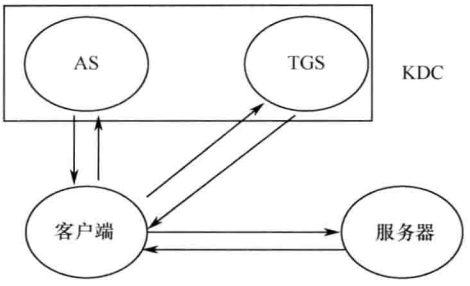

图 20-9　域内认证流程图

图中所用到的符号说明如下。

Client：客户端（C）。

KDC：密钥分配中心（Key Distribution Center），由 AS 和 TGS 组成。

AS：认证服务器（Authentication Server）。

TGS：票据授权服务器（Ticket Granting Server）。

Server：应用服务器（S）。

K_c：客户端 C 的密钥。

K_{tgs}：TGS 的密钥。

K_s：应用服务器 S 的密钥。

$K_{c,\ tgs}$：客户端 C 和 TGS 共享的会话密钥。

TGT：票据授权票，用于访问 TGS 的票据。

在 AS 服务器中，保存有 K_c、K_{tgs} 和 K_s 密钥，AS 服务器分别与客户、TGS 服务器和应用服务器共享这些密钥。

AS 用于在登录时验证用户身份，TGS 用于发放身份证明票据。

Kerberos 中使用两种凭证：票据（Ticket）和认证单（Authenticator）。将票据发送给服务器时，票据用来安全地传送客户的身份。票据中有一些信息，服务器可用这些信息保证使用票据的客户就是拥有票据的客户。认证单是与票据一起呈交的其他凭证。

Kerberos 协议在很大程度上依赖于共享密钥的认证技术。它的基本概念很简单：如果一个密码只有两方知道，那么双方都能通过证实另一方是否知道该密码来验证其身份。但这种方式存在一个问题，就是通信双方如何知道这一共享的密码，如果通过网络传输密码，则很容易被网络监听器所截获和解密。对于这个问题，Kerberos 协议利用密钥加密的方法来解决。通信双方不再共享口令，而是共享一个密钥，通信双方用这个密钥的有关信息互相证实身份，而密码本身不在网络中进行传输。

共享密钥解决了密码传输的问题，但另一个问题是通信中的双方如何获取密钥。在 Kerberos 中，密钥的分配是通过 KDC 的服务来统一管理的。KDC 是一个运行在域服务器上的服务，它维护着在其管辖的域范围内所有安全主体的账号信息数据。

客户端（Client）要访问应用服务器（Server），需要进行 6 次协议交换。具体认证过程如下：

第一阶段（认证服务交换）客户端从 AS 处获取 TGT。

（1）Client→AS

客户端向 AS 发出访问 TGS 的请求，请求以报文形式发送。请求报文包括客户的名称，TGS 的名称，客户端的 IP 地址，以及时戳。时戳用于向 AS 表示这一请求是新的。请求报文以明文方式发送。

（2）AS→Client

AS 收到客户请求报文后，在其数据库中查找客户的加密密钥 Kc，并产生随机会话密钥 $K_{c,\ tgs}$ 和 TGS 的票据 TGT 作为应答报文。会话密钥 $K_{c,\ tgs}$ 用于客户端与 TGS 进行加密通信，用 K_c 加密。TGT 的内容包括：TGS 的名字，客户名字，客户的 IP 地址，时戳，有效生存期限，以及会话密钥 $K_{c,\ tgs}$。这些数据使用 TGS 的密钥 K_{tgs} 进行加密，以保证只有 TGS 才能解密。

AS 向客户端发出应答。应答内容用客户的密钥 K_c 加密，使得只有客户端 Client 才能解密该报文的内容。如果客户端解不开该应答报文，则表明该客户的身份是假冒的，身份认证宣告失败；如果客户端能解开该报文，则证明该客户身份是正确的。

客户端收到 AS 返回的应答报文后，在本地输入口令生成 K_c，对报文进行解密，就得到 TGS 的票据 TGT，客户在下一步就可以把 TGT 发送给 TGS 来证明自己具有访问 TGS 的合法身份。客户端同时从 AS 处得到了与 TGS 的会话密钥 $K_{c,\ tgs}$，用它来与 TGS 进行加密通信。

第二阶段（授权服务交换）客户从 TGS 处获取访问应用服务器的票据 $T_{c,\ s}$。

（3）Client→TGS

客户端向 TGS 发送访问应用服务器 S 的请求报文，报文内容包括要访问的应用服务器 S 的名字，TGS 的票据 TGT 以及认证符 $A_{c,\ tgs}$。

TGT 的内容是用 TGS 的密钥 K_{tgs} 加密的，只有 TGS 才能解开。认证符的内容包括客户的名字、客户的 IP 地址以及时戳，认证符的内容用客户和 TGS 的会话密钥进行加密，以保证只有 TGS 才能解开。票据 TGT 可以重复使用且有效期较长，而认证符只能使用一次且有效期很短。

TGS 收到客户端发来的请求报文后，用自己的密钥 K_{tgs} 对票据 TGT 进行解密处理，得知客户已经从 AS 处得到与自己的会话密钥 $K_{c,\ tgs}$，此处票据 TGT 的含义为“使用 $K_{c,\ tgs}$ 的客户是 C”。TGS 用 $K_{c,\ tgs}$ 解密认证符，并将认证符中的数据与 TGT 中的数据进行比较，从而可以相信 TGT 的发送者 C 就是 TGT 的实际持有者。

此处的票据 TGT 并不能证明任何人的身份，只是用来安全地分配密钥，而认证符则用来证明客户身份。因为认证符只能被使用一次且其有效期很短，所以可以防止对票据和认证符的盗窃使用。

（4）TGS→Client

TGS 检验认为客户身份合法以后，产生随机会话密钥 $K_{c,\ s}$，该密钥用于客户端 C 和应用服务器 S 进行加密通信，同时产生用于访问应用服务器 S 的票据 $T_{c,\ s}$。$T_{c,\ s}$ 的内容包括：应用服务器的名字，客户的名字，客户的 IP 地址，时戳，有效生存期和会话密钥 $K_{c,\ s}$。$T_{c,\ s}$ 的内容用应用服务器 S 的密钥 K_s 加密，以保证只有 S 才能解开。会话密钥 $K_{c,\ s}$ 和票据 $T_{c,\ s}$ 组成 TGS 的应答报文，该应答报文用客户 C 和 TGS 的会话密钥 $K_{c,\ tgs}$ 加密，以保证只有客户端 C 才能解开。TGS 将该应答报文发送给客户端 C。

客户端 C 收到 TGS 的应答报文后，用会话密钥 $K_{c,tgs}$ 对报文进行解密，可以得到访问应用服务器 S 的票据 $T_{c,s}$，以及与 S 进行加密通信的会话密钥 $K_{c,s}$。只有合法的用户 C 才能解开给报文的内容。

第三阶段（客户端/应用服务器交换）客户端与服务器相互验证身份。

（5）Client→S

客户端 C 向应用服务器 S 发送请求报文，报文的内容包括应用服务器的名字，用于访问应用服务器 S 的票据 $T_{c,s}$，以及认证符。

$T_{c,s}$ 的内容是用应用服务器 S 的密钥 K_s 加密的，只有 S 才能解开。认证符的内容包括客户的名字，客户的 IP 地址，时戳。认证符的内容用客户和服务器的会话密钥加密，以保证只有应用服务器 S 才能解开。票据 $T_{c,s}$ 可以重复使用且有效期较长，而认证符只能使用一次且有效期很短。

应用服务器 S 收到客户端发来的请求报文后，用自己的密钥 K_s 对票据 $T_{c,s}$ 进行解密处理，得知客户 C 已经从 TGS 处得到与自己的会话密钥 $K_{c,s}$。此处票据 $T_{c,s}$ 的含义为“使用 $K_{c,s}$ 的客户端是 C”。S 用 $K_{c,s}$ 解密认证符，并将认证符中的数据与 $T_{c,s}$ 中的数据进行比较，从而可以相信 $T_{c,s}$ 的发送者 C 就是 $T_{c,s}$ 的实际持有者，客户端 C 的身份得到验证。

（6）S→Client

应用服务器 S 检验认为客户端 C 身份合法后，对从认证符中得到的时戳加 1，然后用与客户端共享的会话密钥 $K_{c,s}$ 加密后作为应答报文发给客户。该应答报文只有客户 C 才能解开。

客户 C 收到应用服务器 S 发来的应答报文后，用会话密钥 $K_{c,s}$ 进行解密后，对应答报文中增加的时戳进行验证，通过比较时间戳的有效性实现对 S 的认证。验证正确则相信增加时戳的确实是应用服务器 S，从而应用服务器 S 的身份得到验证。

整个协议交换过程结束后，客户和应用服务器之间就拥有了共享的会话密钥，双方以后就可以用该会话密钥进行加密通信了。

20.4 TSP

1. 简介

在书面合同中，与手写签名一样，签署日期也是一项关键性内容，用于防止文件被伪造和篡改。在电子交易时代，如何确定电子文件的签署日期并防止被篡改成为一个难题。时间戳技术（TSP，Time-Stamp Protocol）能有效解决电子文件的签署日期问题，通过数字签名技术将电子数据和特定时间绑定在一起，既能准确确定电子文件的签署日期，又能有效防止该签署日期和文件内容被伪造或修改，实质上是数字签名技术的一种特殊应用。

通过数字签名技术将电子数据和特定时间绑定后的结果称作时间戳（time-stamp），它主要由 3 部分内容组成：电子数据的摘要值、特定时间、数字签名。

书面合同的时间是签署人自己写上的，但时间戳则不然，其中的特定时间和数字签名

均由特定机构 TSA（Time Stamping Authority）签署。TSA 可理解为可信第三方 TTP（Trusted Third Party）服务机构，具有权威性，专门为用户数据签发时间戳。

时间戳技术总体结构如图 20-10 所示。

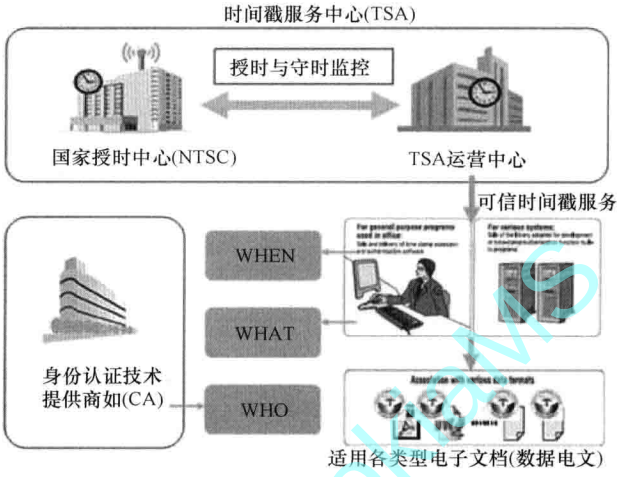

图 20-10　时间戳总体结构

时间戳产生的过程主要包括以下几个步骤：

① 用户（时间戳需求方）将电子数据（文件）使用摘要算法计算出摘要值，然后组成时间戳请求包 TimeStampReq。

② 用户将请求包 TimeStampReq 发送给 TSA。

③ TSA 接收到请求包 TimeStampReq。

④ TSA 需要验证 TimeStampReq 的时效性，通过判断 nonce 是否重复来防止重放攻击。TSA 使用私钥对请求包中的摘要进行数字签名后，组成时间戳响应包 TimeStampResp。TSA 可以拥有多个私钥，针对不同策略、不同算法等，可使用不同的私钥。TSA 数字证书的 extendedKeyUsage 扩展项必须设置为关键扩展项，且必须包含 id-kp-timeStamping 扩展密钥用途。

⑤ TSA 将响应包 TimeStampResp 发送给用户。

⑥ 用户接收到响应包 TimeStampResp。

⑦ 用户首先判断 TimeStampResp 中的状态信息，如果为错误状态，则表示本次时间戳申请失败；如果为正确状态，则验证响应包中各种字段信息和 TSA 签名是否正确；如果字段信息或签名不正确，则拒绝该响应包。用户需要通过 OCSP 或 CRL 验证 TSA 证书是否作废或有效。

2. 请求包格式

时间戳请求包格式用 ASN.1 描述如下：

```
TimeStampReq ::= SEQUENCE  {
    version                  INTEGER  { v1(1) },
    messageImprint           MessageImprint,
```