SSL技术白皮书

关键词：SSL，PKI，MAC

摘　　要：SSL利用数据加密、身份验证和消息完整性验证机制，为基于TCP等可靠连接的应用层协议提供安全性保证。本文介绍了SSL的产生背景、安全机制、工作过程及典型组网应用。

缩略语：

缩略语	英文全名	中文解释
AES	Advanced Encryption Standard	高级加密标准
CA	Certificate Authority	证书机构
DES	Data Encryption Standard	数据加密标准
HTTPS	Hypertext Transfer Protocol Secure	安全超文本传输协议
MAC	Message Authentication Code	消息验证码
MD5	Message Digest 5	消息摘要算法5
PKI	Public Key Infrastructure	公钥基础设施
RSA	Rivest Shamir and Adleman	非对称密钥算法的一种
SHA	Secure Hash Algorithm	安全散列算法
SSL	Secure Sockets Layer	安全套接层
VPN	Virtual Private Network	虚拟专有网络

目 录

1 概述

1.1 产生背景

基于万维网的电子商务和网上银行等新兴应用，极大地方便了人们的日常生活，受到人们的青睐。由于这些应用都需要在网络上进行在线交易，它们对网络通信的安全性提出了更高的要求。传统的万维网协议HTTP不具备安全机制——采用明文的形式传输数据、不能验证通信双方的身份、无法防止传输的数据被篡改等，导致HTTP无法满足电子商务和网上银行等应用的安全性要求。

Netscape公司提出的安全协议SSL，利用数据加密、身份验证和消息完整性验证机制，为网络上数据的传输提供安全性保证。SSL可以为HTTP提供安全连接，从而很大程度上改善了万维网的安全性问题。

1.2 技术优点

SSL具有如下优点：

- 提供较高的安全性保证。SSL 利用数据加密、身份验证和消息完整性验证机制，保证网络上数据传输的安全性。
- 支持各种应用层协议。虽然 SSL 设计的初衷是为了解决万维网安全性问题，但是由于 SSL 位于应用层和传输层之间，它可以为任何基于 TCP 等可靠连接的应用层协议提供安全性保证。
- 部署简单。目前 SSL 已经成为网络中用来鉴别网站和网页浏览者身份，在浏览器使用者及 Web 服务器之间进行加密通信的全球化标准。SSL 协议已被集成到大部分的浏览器中，如 IE、Netscape、Firefox 等。这就意味着几乎任意一台装有浏览器的计算机都支持 SSL 连接，不需要安装额外的客户端软件。

2 协议安全机制

SSL协议实现的安全机制包括：

- 数据传输的机密性：利用对称密钥算法对传输的数据进行加密。
- 身份验证机制：基于证书利用数字签名方法对服务器和客户端进行身份验证，其中客户端的身份验证是可选的。
- 消息完整性验证：消息传输过程中使用 MAC 算法来检验消息的完整性。

2.1 数据传输的机密性

网络上传输的数据很容易被非法用户窃取，SSL采用在通信双方之间建立加密通道的方法保证数据传输的机密性。

所谓加密通道，是指发送方在发送数据前，使用加密算法和加密密钥对数据进行加密，然后将数据发送给对方；接收方接收到数据后，利用解密算法和解密密钥从密文中获取明文。没有解密密钥的第三方，无法将密文恢复为明文，从而保证数据传输的机密性。

加解密算法分为两类：

- 对称密钥算法：数据加密和解密时使用相同的密钥。
- 非对称密钥算法：数据加密和解密时使用不同的密钥，一个是公开的公钥，一个是由用户秘密保存的私钥。利用公钥（或私钥）加密的数据只能用相应的私钥（或公钥）才能解密。

与非对称密钥算法相比，对称密钥算法具有计算速度快的优点，通常用于对大量信息进行加密（如对所有报文加密）；而非对称密钥算法，一般用于数字签名和对较少的信息进行加密。

SSL加密通道上的数据加解密使用对称密钥算法，目前主要支持的算法有DES、3DES、AES等，这些算法都可以有效地防止交互数据被窃听。

对称密钥算法要求解密密钥和加密密钥完全一致。因此，利用对称密钥算法加密传输数据之前，需要在通信两端部署相同的密钥。对称密钥的部署方法请参见“2.4 利用非对称密钥算法保证密钥本身的安全”。

2.2 身份验证机制

电子商务和网上银行等应用中必须保证要登录的Web服务器是真实的，以免重要信息被非法窃取。SSL利用数字签名来验证通信对端的身份。

非对称密钥算法可以用来实现数字签名。由于通过私钥加密后的数据只能利用对应的公钥进行解密，因此根据解密是否成功，就可以判断发送者的身份，如同发送者对数据进行了“签名”。例如，Alice使用自己的私钥对一段固定的信息加密后发给Bob，Bob利用Alice的公钥解密，如果解密结果与固定信息相同，那么就能够确认信息的发送者为Alice，这个过程就称为数字签名。

SSL客户端必须验证SSL服务器的身份，SSL服务器是否验证SSL客户端的身份，则由SSL服务器决定。SSL客户端和SSL服务器的身份验证过程，请参见“3.2 SSL握手过程”。

使用数字签名验证身份时，需要确保被验证者的公钥是真实的，否则，非法用户可能会冒充被验证者与验证者通信。如**错误！未找到引用源。**所示，Cindy冒充Bob，将自己的公钥发给Alice，并利用自己的私钥计算出签名发送给Alice，Alice利用“Bob”的公钥（实际上为Cindy的公钥）成功验证该签名，则Alice认为Bob的身份验证成功，而实际上与Alice通信的是冒充Bob的Cindy。SSL利用PKI提供的机制保证公钥的真实性，详细介绍请参见“2.5 利用PKI保证公钥的真实性”。

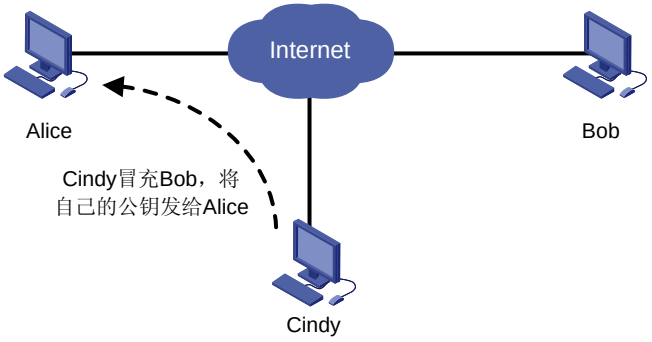

图1 伪造公钥

2.3 消息完整性验证

为了避免网络中传输的数据被非法篡改，SSL利用基于MD5或SHA的MAC算法来保证消息的完整性。

MAC算法是在密钥参与下的数据摘要算法，能将密钥和任意长度的数据转换为固定长度的数据。利用MAC算法验证消息完整性的过程如图2所示。发送者在密钥的参与下，利用MAC算法计算出消息的MAC值，并将其加在消息之后发送给接收者。接收者利用同样的密钥和MAC算法计算出消息的MAC值，并与接收到的MAC值比较。如果二者相同，则报文没有改变；否则，报文在传输过程中被修改，接收者将丢弃该报文。

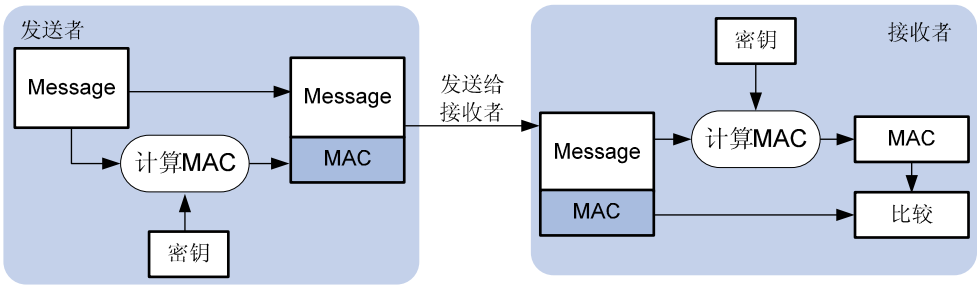

图2 MAC算法示意图

MAC算法具有如下特征，使其能够用来验证消息的完整性：

- 消息的任何改变，都会引起输出的固定长度数据产生变化。通过比较 MAC 值，可以保证接收者能够发现消息的改变。
- MAC 算法需要密钥的参与，因此没有密钥的非法用户在改变消息的内容后，无法添加正确的 MAC 值，从而保证非法用户无法随意修改消息内容。

MAC算法要求通信双方具有相同的密钥，否则MAC值验证将会失败。因此，利用MAC算法验证消息完整性之前，需要在通信两端部署相同的密钥。MAC密钥的部署方法请参见“2.4 利用非对称密钥算法保证密钥本身的安全”。

2.4 利用非对称密钥算法保证密钥本身的安全

对称密钥算法和MAC算法要求通信双方具有相同的密钥，否则解密或MAC值验证将失败。因此，要建立加密通道或验证消息完整性，必须先在通信双方部署一致的密钥。

SSL利用非对称密钥算法加密密钥的方法实现密钥交换，保证第三方无法获取该密钥。如图3所示，SSL客户端（如Web浏览器）利用SSL服务器（如Web服务器）的公钥加密密钥，将加密后的密钥发送给SSL服务器，只有拥有对应私钥的SSL服务器才能从密文中获取原始的密钥。SSL通常采用RSA算法加密传输密钥。

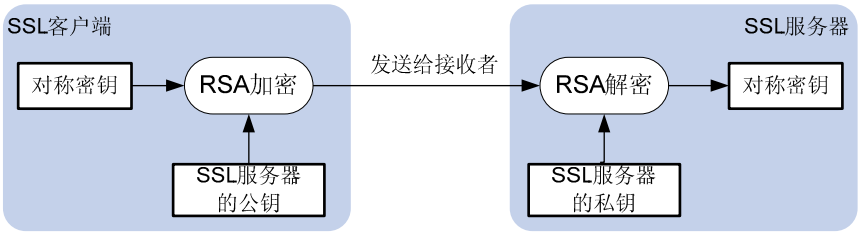

图3 密钥交换示意图

说明

- 实际上，SSL 客户端发送给 SSL 服务器的密钥不能直接用来加密数据或计算 MAC 值，该密钥是用来计算对称密钥和 MAC 密钥的信息，称为 premaster secret。SSL 客户端和 SSL 服务器利用 premaster secret 计算出相同的主密钥（master secret），再利用 master secret 生成用于对称密钥算法、MAC 算法等的密钥。premaster secret 是计算对称密钥、MAC 算法密钥的关键。
- 用来实现密钥交换的算法称为密钥交换算法。非对称密钥算法 RSA 用于密钥交换时，也可以称之为密钥交换算法。

利用非对称密钥算法加密密钥之前，发送者需要获取接收者的公钥，并保证该公钥确实属于接收者，否则，密钥可能会被非法用户窃取。如**错误！未找到引用源。**所示，Cindy冒充Bob，将自己的公钥发给Alice，Alice利用Cindy的公钥加密发送给Bob的数据，Bob由于没有对应的私钥无法解密该数据，而Cindy截取数据后，可以利用自己的私钥解密该数据。SSL利用PKI提供的机制保证公钥的真实性，详细介绍请参见“2.5 利用PKI保证公钥的真实性”。

2.5 利用PKI保证公钥的真实性

PKI通过数字证书来发布用户的公钥，并提供了验证公钥真实性的机制。数字证书（简称证书）是一个包含用户的公钥及其身份信息的文件，证明了用户与公钥的关联。数字证书由权威机构——CA签发，并由CA保证数字证书的真实性。

SSL客户端把密钥加密传递给SSL服务器之前，SSL服务器需要将从CA获取的证书发送给SSL客户端，SSL客户端通过PKI判断该证书的真实性。如果该证书确实属于SSL服务器，则利用该证书中的公钥加密密钥，发送给SSL服务器。

验证SSL服务器/SSL客户端的身份之前，SSL服务器/SSL客户端需要将从CA获取的证书发送给对端，对端通过PKI判断该证书的真实性。如果该证书确实属于SSL服务器/SSL客户端，则对端利用该证书中的公钥验证SSL服务器/SSL客户端的身份。

3 协议工作过程

3.1 SSL的分层结构

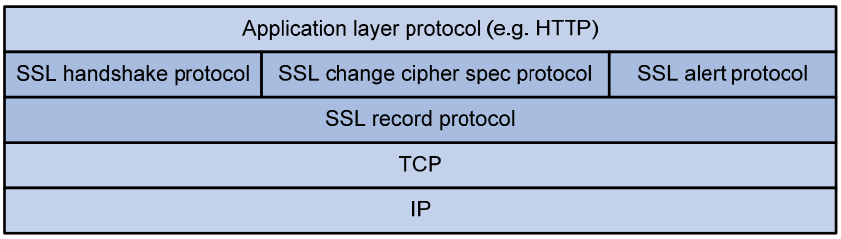

图4 SSL协议分层

如图4所示，SSL位于应用层和传输层之间，它可以为任何基于TCP等可靠连接的应用层协议提供安全性保证。SSL协议本身分为两层：

- 上层为 SSL 握手协议（SSL handshake protocol）、SSL 密码变化协议（SSL change cipher spec protocol）和 SSL 警告协议（SSL alert protocol）；
- 底层为 SSL 记录协议（SSL record protocol）。

其中：

- SSL 握手协议：是 SSL 协议非常重要的组成部分，用来协商通信过程中使用的加密套件（加密算法、密钥交换算法和 MAC 算法等）、在服务器和客户端之间安全地交换密钥、实现服务器和客户端的身份验证。
- SSL 密码变化协议：客户端和服务器端通过密码变化协议通知对端，随后的报文都将使用新协商的加密套件和密钥进行保护和传输。
- SSL 警告协议：用来向通信对端报告告警信息，消息中包含告警的严重级别和描述。
- SSL 记录协议：主要负责对上层的数据（SSL 握手协议、SSL 密码变化协议、SSL 警告协议和应用层协议报文）进行分块、计算并添加 MAC 值、加密，并把处理后的记录块传输给对端。

3.2 SSL握手过程

SSL通过握手过程在客户端和服务器之间协商会话参数，并建立会话。会话包含的主要参数有会话ID、对方的证书、加密套件（密钥交换算法、数据加密算法和MAC算法等）以及主密钥（master secret）。通过SSL会话传输的数据，都将采

用该会话的主密钥和加密套件进行加密、计算MAC等处理。

不同情况下，SSL握手过程存在差异。下面将分别描述以下三种情况下的握手过程：

- 只验证服务器的 SSL 握手过程
- 验证服务器和客户端的 SSL 握手过程
- 恢复原有会话的 SSL 握手过程

3.2.1 只验证服务器的SSL握手过程

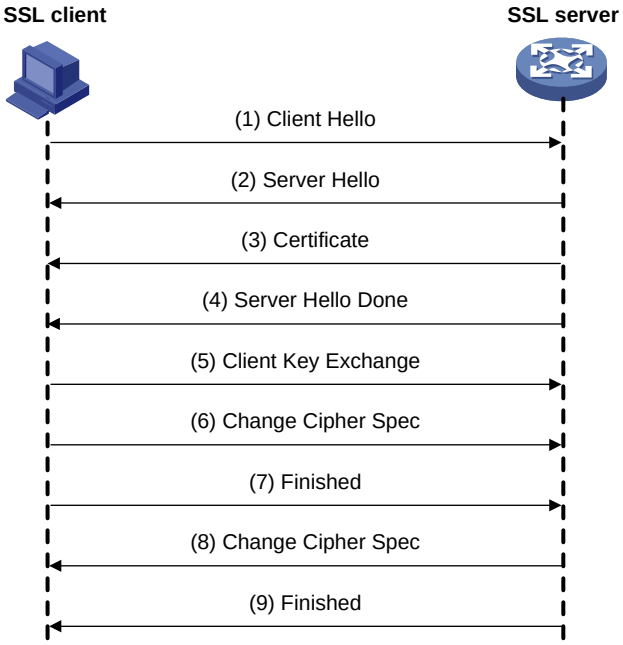

图5 只验证服务器的SSL握手过程

如图5所示，只需要验证SSL服务器身份，不需要验证SSL客户端身份时，SSL的握手过程为：

(1) SSL 客户端通过 Client Hello 消息将它支持的 SSL 版本、加密算法、密钥交换算法、MAC 算法等信息发送给 SSL 服务器。

(2) SSL 服务器确定本次通信采用的 SSL 版本和加密套件，并通过 Server Hello 消息通知给 SSL 客户端。如果 SSL 服务器允许 SSL 客户端在以后的通信中重用本次会话，则 SSL 服务器会为本次会话分配会话 ID，并通过 Server Hello 消息发送给 SSL 客户端。

(3) SSL 服务器将携带自己公钥信息的数字证书通过 Certificate 消息发送给 SSL 客户端。

(4) SSL 服务器发送 Server Hello Done 消息，通知 SSL 客户端版本和加密套件协商结束，开始进行密钥交换。

(5) SSL 客户端验证 SSL 服务器的证书合法后，利用证书中的公钥加密 SSL 客户端随机生成的 premaster secret，并通过 Client Key Exchange 消息发送给 SSL 服务器。

(6) SSL 客户端发送 Change Cipher Spec 消息，通知 SSL 服务器后续报文将采用协商好的密钥和加密套件进行加密和 MAC 计算。

(7) SSL 客户端计算已交互的握手消息（除 Change Cipher Spec 消息外所有已交互的消息）的 Hash 值，利用协商好的密钥和加密套件处理 Hash 值（计算并添加 MAC 值、加密等），并通过 Finished 消息发送给 SSL 服务器。SSL 服务器利用同样的方法计算已交互的握手消息的 Hash 值，并与 Finished 消息的解密结果比较，如果二者相同，且 MAC 值验证成功，则证明密钥和加密套件协商成功。

(8) 同样地，SSL 服务器发送 Change Cipher Spec 消息，通知 SSL 客户端后续报文将采用协商好的密钥和加密套件进行加密和 MAC 计算。

(9) SSL 服务器计算已交互的握手消息的 Hash 值，利用协商好的密钥和加密套件处理 Hash 值（计算并添加 MAC 值、加密等），并通过 Finished 消息发送给 SSL 客户端。SSL 客户端利用同样的方法计算已交互的握手消息的 Hash 值，并与 Finished 消息的解密结果比较，如果二者相同，且 MAC 值验证成功，则证明密钥和加密套件协商成功。

SSL客户端接收到SSL服务器发送的Finished消息后，如果解密成功，则可以判断SSL服务器是数字证书的拥有者，即SSL服务器身份验证成功，因为只有拥有私钥的SSL服务器才能从Client Key Exchange消息中解密得到premaster secret，从而间接地实现了SSL客户端对SSL服务器的身份验证。

📖 **说明：**

- Change Cipher Spec 消息属于 SSL 密码变化协议，其他握手过程交互的消息均属于 SSL 握手协议，统称为 SSL 握手消息。
- 计算 Hash 值，指的是利用 Hash 算法（MD5 或 SHA）将任意长度的数据转换为固定长度的数据。

3.2.2 验证服务器和客户端的SSL握手过程

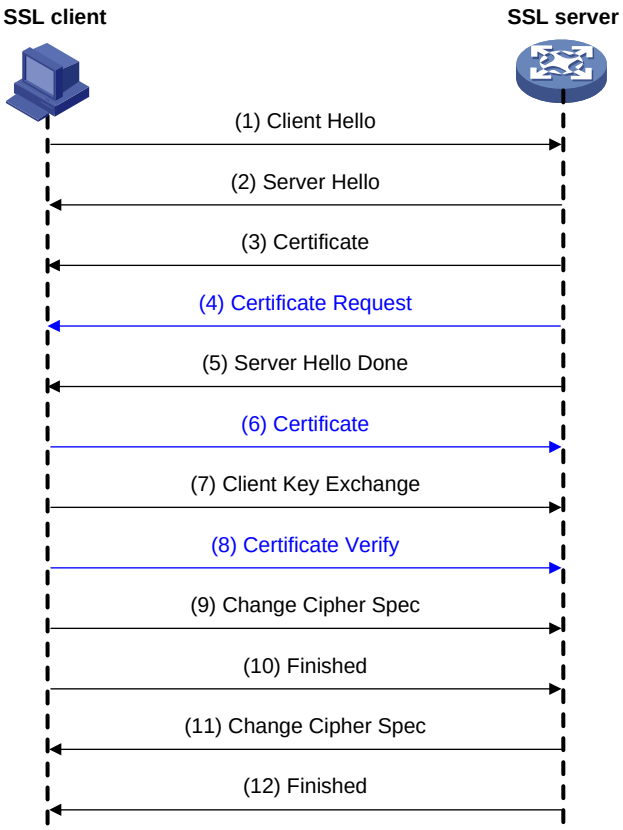

图6 验证服务器和客户端的SSL握手过程

SSL客户端的身份验证是可选的，由SSL服务器决定是否验证SSL客户端的身份。如图6中蓝色部分标识的内容所示，如果SSL服务器验证SSL客户端身份，则SSL服务器和SSL客户端除了交互“3.2.1 只验证服务器的SSL握手过程”中的消息协商密钥和加密套件外，还需要进行以下操作：

(1) SSL 服务器发送 Certificate Request 消息，请求 SSL 客户端将其证书发送给 SSL 服务器。

(2) SSL 客户端通过 Certificate 消息将携带自己公钥的证书发送给 SSL 服务器。SSL 服务器验证该证书的合法性。

(3) SSL 客户端计算已交互的握手消息、主密钥的 Hash 值，利用自己的私钥对其进行加密，并通过 Certificate Verify 消息发送给 SSL 服务器。

(4) SSL 服务器计算已交互的握手消息、主密钥的 Hash 值，利用 SSL 客户端证书中的公钥解密 Certificate Verify 消息，并将解密结果与计算出的 Hash 值比较。如果二者相同，则 SSL 客户端身份验证成功。

3.2.3 恢复原有会话的SSL握手过程

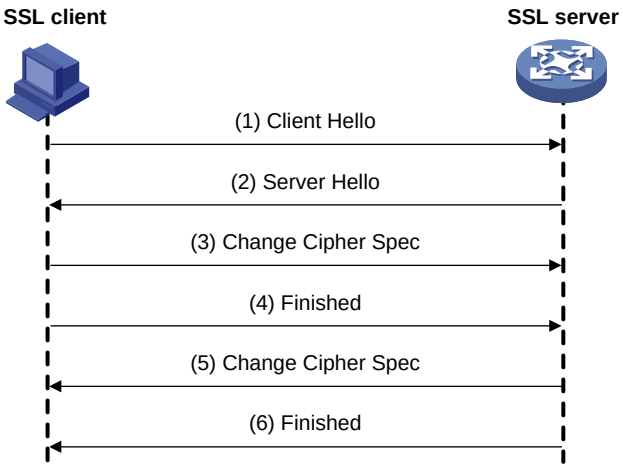

图7 恢复原有会话的SSL握手过程

协商会话参数、建立会话的过程中，需要使用非对称密钥算法来加密密钥、验证通信对端的身份，计算量较大，占用了大量的系统资源。为了简化SSL握手过程，SSL允许重用已经协商过的会话，具体过程为：

(1) SSL 客户端发送 Client Hello 消息，消息中的会话 ID 设置为计划重用的会话的 ID。

(2) SSL 服务器如果允许重用该会话，则通过在 Server Hello 消息中设置相同的会话 ID 来应答。这样，SSL 客户端和 SSL 服务器就可以利用原有会话的密钥和加密套件，不必重新协商。

(3) SSL 客户端发送 Change Cipher Spec 消息，通知 SSL 服务器后续报文将采用原有会话的密钥和加密套件进行加密和 MAC 计算。

(4) SSL 客户端计算已交互的握手消息的 Hash 值，利用原有会话的密钥和加密套件处理 Hash 值，并通过 Finished 消息发送给 SSL 服务器，以便 SSL 服务器判断密钥和加密套件是否正确。

(5) 同样地，SSL 服务器发送 Change Cipher Spec 消息，通知 SSL 客户端后续报文将采用原有会话的密钥和加密套件进行加密和 MAC 计算。

(6) SSL 服务器计算已交互的握手消息的 Hash 值，利用原有会话的密钥和加密套件处理 Hash 值，并通过 Finished 消息发送给 SSL 客户端，以便 SSL 客户端判断密钥和加密套件是否正确。

4 典型组网应用

4.1 HTTPS

HTTPS是基于SSL安全连接的HTTP协议。HTTPS通过SSL提供的数据加密、身份验证和消息完整性验证等安全机制，为Web访问提供了安全性保证，广泛应用于网上银行、电子商务等领域。

图8为HTTPS在网上银行中的应用。某银行为了方便客户，提供了网上银行业务，客户可以通过访问银行的Web服务器进行帐户查询、转帐等。通过在客户和银行的Web服务器之间建立SSL连接，可以保证客户的信息不被非法窃取。

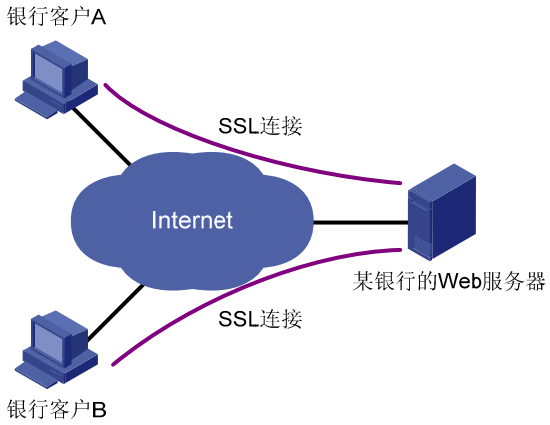

图8 HTTPS在网上银行中的应用

4.2 SSL VPN

SSL VPN是以SSL为基础的VPN技术，利用SSL提供的安全机制，为用户远程访问公司内部网络提供了安全保证。如图9所示，SSL VPN通过在远程接入用户和SSL VPN网关之间建立SSL安全连接，允许用户通过各种Web浏览器，各种网络接入方式，在任何地方远程访问企业网络资源，并能够保证企业网络的安全，保护企业内部信息不被窃取。

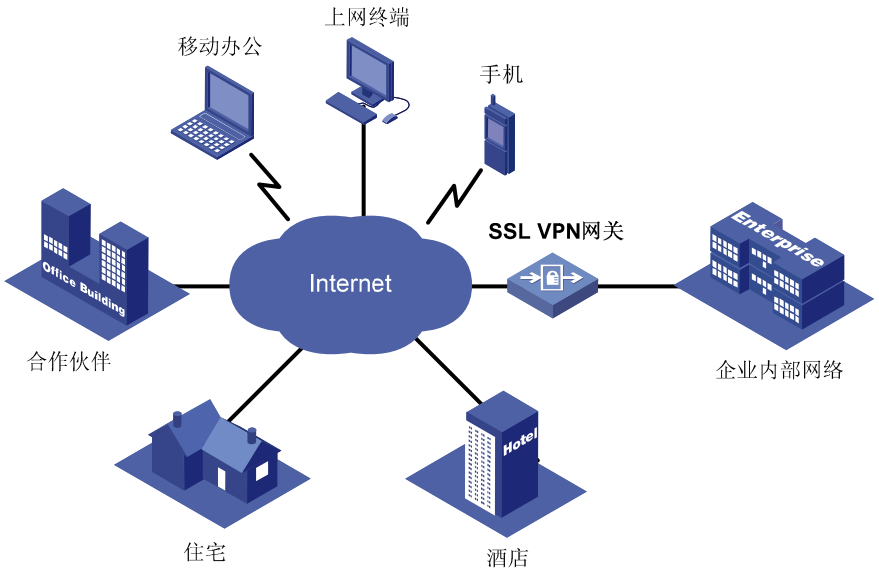

图9 SSL VPN的典型组网环境

5 参考文献

- draft-freier-ssl-version3-02：The SSL Protocol Version 3.0