

SSL 安全代理的 Java 实现*

丁月华 熊前兴

(武汉理工大学 武汉 430063)

摘 要 文章介绍了 SSL 协议和 Java 语言所提供的安全平台,在此基础上介绍了 SSL 安全代理实现的基本原理。

关键词 SSL Java 代理

7P3 B

Abstract: This paper introduces SSL protocol and the security platform of Java. Based on the introduction, this paper also illustrates the theory of implementation of SSL proxy.

Key words: SSL; Java; Proxy

1 SSL 协议简介

安全套接层(Secure Socket Layer, SSL)是 Netscape 公司率先采用的一种网络安全协议,它可对网页与服务器之间传输的数据加密。SSL 协议提供的服务可归纳为 3 个方面:用户和服务器的合法性认证、加密数据和保护数据的完整性^[3]。因此采用 SSL 协议传输密码和信用卡号等敏感信息以及身份认证信息是一种较理想的选择。SSL 是介于 HTTP 协议与 TCP 协议之间的一个可选层。它在 OSI 模型中的位置如表 1 所示。

SSL 在 TCP 之上建立了一个加密通道,通过该通道的数据均经过了加密/解密过程。具体来讲,SSL 协议从结构上又可以分为两部分:握手协议(Handshake Protocol)和记录协议(Record Protocol)。其中握手协议用于协商密钥,协议的大部分内容都是描述通信双方如何安全地协商会话密

表 1 SSL 所在层次图

应用层	HTTP/LDAP/IMAP
表示层	SSL 握手协议
会话层	SSL 记录层
传输层	TCP
网络层	IP
链路层	Ethernet/Token Ring
物理层	LAN/MAN/WAN

钥。记录协议则定义了传输的格式。简单地讲,SSL 协议的工作流程是:SSL 客户端(也是 TCP 的客户端)在 TCP 连接建立后,发出一个消息(Client Hello),该消息中包含了 SSL 可实现的密码算法列表、压缩方法列表、SSL 版本号、产生的 32 字节随机数、会话 ID 和其它一些必要的消息。SSL 的服务器端将回应一个消息(Server Hello),其中确定了该次通信所要用的密码算法和压缩算法,然后发出服务器端的证书(其中包含了身份和公钥)。客户端在收到该消息后会生成一个秘密消

BulkTask 任务负责处理 Bulk-only 协议。在该传输方式下,有 3 种类型的数据在 PC 机和 U 盘之间传送:CBW, CSW 和普通数据。CBW(Command Block Wrapper,即命令块包)是从 PC 机发送到 U 盘的命令,这里为 SCSI 传输命令集。BulkTask 任务必须执行相应的命令,完成以后,向 PC 机发出反映当前命令执行状态的 CSW(Command Status Wrapper),PC 机根据 CSW 来决定是否继续发送下一个 CBW 或是数据。

BulkTask 任务的流程图如图 7 所示。

5 结 论

采用模块化方法,构造了一个符合 USB1.1 标准,使用灵活方便的 USB 接口。该设计已成功实现并被相关部门采用。

参考文献

- 1 李贵山,陈金鹏编. PCI 局部总线及其应用. 西安:西安电子科技大学出版,2003. 8~251
- 2 (美)阿多森(Adoron, D)著. 精英科技译. USB 系统体系. 北京:中国电力出版社,2000. 15~197

收稿日期:2003-10-08

*湖北省 2002 年自然科学基金项目资助(2002AB042)

息,并用 SSL 服务器的公钥加密后传回服务器。服务器用自己的私钥解密后,会话密钥协商成功,则双方可以用同一份会话密钥通信了。SSL 协议的握手阶段主要应用的是非对称密钥加密,一旦客户端认定了服务器端的身份,双方就开始应用事先商量好的对称密钥来进行加密通讯。图 1 简

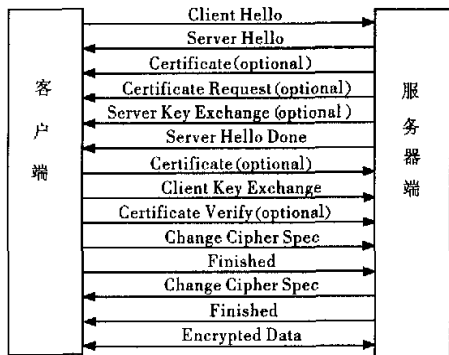


图 1 握手协议

要说明了 SSL 协议中的握手协议:

2 Java 语言的安全体系

Java 平台已经内置了 Java 安全体系结构核心和 Java 加密体系结构(JCA)^[2],两者构成 Java 所带的安全平台,如图 2 所示。

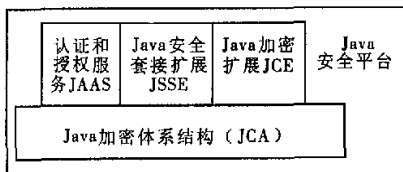


图 2 Java 安全体系结构的标准组件

2.1 JCA

JCA(JavaCryptography Architecture)提供了用 Java 平台实现基本加密功能的工具。加密功能的范围包括用基本加密函数与算法保护数据完整性,防止破坏数据。JCA 中还有标识数据与代码源的加密签名生成算法。用 JCA 提供的基本加密功能接口可以开发实现含消息摘要、数字签名、密钥生成、密钥转换、密钥库管理、证书管理和使用等功能的应用程序。但大多数情况下 JCA 只是定义了这些接口,并没有有效地实现这些功能,这些功能往往由加密服务提供者(CSP)来实现。CSP 遵照 JCA 中定义的加密接口实现一个或几个加密功能,CSP 实现的加密功能封装成扩展 JCA 库为数据一些标准加密服务提供者接口而成的加密

引擎类的形式提供,因此,应用程序员可以通过 JCA 定义的 API 来使用这些功能,这样对加解密功能的调用和具体 CSP 无关。加密引擎表示特定加密算法和一组调用算法所用参数,完成一些加密功能。具体加密算法通过加密引擎提供的静态 getInstance() 方法调用,其 algorithm 参数表示特定算法名,如 DES,该方法返回加密引擎的具体对象实例。

2.2 JSSE

JSSE(Java Secure Socket Extension)提供了 SSL 的加密功能^[1]。如果需要与 SSL 服务器或 SSL 客户通信,可以使用此扩展包中的 API。如果服务器和客户的实现中都需要增加加密功能,则不仅可以使使用 JSSE,也可以使用 JCE 的密码功能。JSSE 提供了实现 SSL 通信的标准 Java API。JSSEv1.0 结构包括下列包:

javax.net.ssl; 包含 JSSE API 的一组核心类和接口。

javax.net; 支持基本客户机套接与服务器套接工厂功能所必需的。

javax.security.cert; 支持基本证书管理功能所必需的。

在 JDK1.4 后,JSSE 已整合到 JDK 中了。利用 JSSE 可象处理协议一样创建和使用 SSLSocket,从程序设计的角度来看,SSL 协议中的握手协议都是透明的,开发人员申请一个 SSLSocket,当创建了 SSLSocket 后,协议的握手也随之完成。

2.3 JCE

CE Java Cryptography Extension 是一组包,它为加密、密钥生成、密钥一致和消息认证代码(Message Authentication Code,MAC)算法提供了一个框架和实现。JCE 还支持安全流和封装的对象。在 JCE 框架内,可把其他配备了密码系统的库作为服务提供者插入。JCE 的下载包中包含 Sun JCE 服务提供者。其他供应商(如 IBM)也提供他们自己的 JCE 服务,开发者可把这些实现插入到 JCE 框架中。Java 密码架构(JCA)和 Java 密码扩展(JCE)的设计目的是为 Java 提供与实现无关的加密函数 API。JCA 是 Java2 运行环境的一部分,而 JCE(在 jdk1.4 之前)是对没包含在 JDK 中 JCA 的扩展。JCE 为 JCA 增加了简单的加密和解密 API。在 JCA 的基础上提供加密算法、密钥交换、密钥产生和消息鉴别服务等扩展接口。

2.4 JAAS

JAAS (Java Authentication and Authorization Service) 可以在 Java 平台上提供用户身份鉴别。它提供了灵活和可伸缩的机制来保证客户端或服务端的 Java 程序。Java 早期的安全框架强调的是通过验证代码的来源和作者,保护用户避免受到下载下来的代码的攻击。JAAS 强调的是通过验证谁在运行代码以及他/她的权限来保护系统免受用户的攻击,允许开发者根据用户提供的鉴别信任状,准许用户对程序进行访问。

3 SSL 安全代理的实现原理

3.1 SSL 安全代理的基本原理

根据 SSL 协议规范,SSL 安全代理由服务器端和客户端两部分组成。SSL 安全代理的客户端和浏览器在同一台机器上。SSL 安全代理的服务器端和 Web 服务器在同一台机器上。SSL 安全代理客户端可以接收网络浏览器发出的 HTTP 请求,SSL 安全代理客户端对这个请求进行分析,如果请求通过,SSL 安全代理客户端根据系统配置决定明文转发或者是加密后转发。如 SSL 采用加密,系统对该请求和传输的数据提供 128 位强度 SSL 的加密并将该加密的信息与指定的 SSL 安全代理服务器端发送,服务器端支持 128 位强度的 SSL,并将该请求转换成明文发送给客户请求的目的服务器。SSL 安全代理服务端的工作过程与客户端的类似,处理来自 SSL 安全代理客户端的请求,对 Web 资源进行访问控制。SSL 安全代理的原理图如图 3 所示。

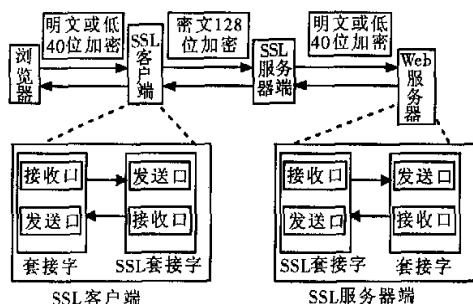


图 3 SSL 安全代理原理

SSL 由下面几个功能部分组成:

1) SSL Proxy 客户端代理。客户代理接收所有来自内部的普通 Socket 连接,然后将这些连接以转发给服务器代理,Socket SSL 客户端同时可对服务器进行认证。

2) SSL 服务器端代理。服务器代理需要一个

特殊的客户端建立连接。它只接收的 SSL Socket 连接请求,然后用 128 位的服务器证书提供服务器认证。它可以有选择地对客户端进行认证,同时也可以对要求连接的客户端地址进行限制。

3) 在客户端和服务端都提供本地的属性和各项参数的设置功能。这些参数包括:是否启用加密;是否对客户进行认证;代理监听端口;SSL 代理连接端口;提供可供选择的加密算法的设置以及是否启用父代理;父代理主机名;父代理连接端口。

4) 日志记录。对代理所处理的任何请求和转发的地址和连接端口及发生时间都写入到系统日志 IP 志文件当中,可供安全审计使用。

5) 支持 HTTP 协议的 GET 和 POST 请求方式。SSL 安全代理 Proxy 的两端的配合使用,用户可以用 SSLProxy 安全代理对任何需要的连接进行加密。而用户只要开启两端的软件,并在浏览器的连接属性中设定代理服务器的 IP 地址和连接端口号之后,用户完全可以同平常一样地使用和浏览网络,而所有对信息保密等功能的实现对用户都是透明的。实现对 Web 资源的访问控制和其它安全功能只需要设置简单的配置,而不需要改动原有 Web 应用的核心内容与运作方式,无需重新编写服务器端的应用程序,具有高度的透明性。

我们可以这样理解,SSL 客户端和 SSL 服务器端是两个刚好相反的过程,SSL 服务器端从 web 服务器得到页面内容(明文或者是低 40 位的加密),然后加密成 128 位的高加密信息。而 SSL 客户端将从信道上得到的加密信息解密成 web 服务器上的页面内容(明文或者是低 40 位的加密)。这样我们就保证了在信道上传输的信息是有高可靠安全保障的加密信息。

3.2 SSL 客户端与 SSL 服务器端模拟

下面说明一下 SSL 客户端的原理:首先我们只考虑最简单的情况(暂时不考虑用户对网站的提交),如果用户只接受来自网页的信息,当用户在 IE 地址栏键入网页地址后,SSL 客户端会自动捕获这样一个地址,然后去代替用户获取网页的内容,SSL 客户端从 SSL 服务器端获取了被 SSL 服务器加密的信息后并传送解密到自己的 serverSocket,用户可以从这个 ServerSocket 读取被解密的信息。我们可以这样说,SSL 客户端一方面是 SSL 服务器端的客户端(SSLSocket),另一方

面它又是我们普通用户的服务器(ServerSocket),起到转传兼解密的作用。

前面介绍了SSL客户端,而SSL服务器端可以理解为SSL客户端的逆过程,SSL服务器端接收到来自于SSL客户端的请求,会去web服务器获取客户所需信息,获取后传递到自己的SSLServerSocket,将信息加密成密文,然后将信息传递给SSL客户端。和SSL客户端一样,SSL服务器端一方面是web服务器的客户端,一方面又是SSL客户端的服务器,起到转传兼加密的作用。

下面这段程序模拟了SSL服务器端最简单的功能,运行该程序后,可以看到原来没有加密的网页变成了加密的网页。

```
package mainp;
import java.io.*;
import java.net.*;
import javax.net.ssl.*;
import java.security.cert.*;
import javax.net.ssl.*;

public class https{
    public static void main(String[] args) throws IOException{
        SSLServerSocketFactory ssf = (SSLServerSocketFactory)
        SSLServerSocketFactory.getDefault();
        SSLServerSocket ss = (SSLServerSocket) ssf.createServerSocket(8080);
        int id = 1;
        while(true){ } //在8080端口监听
        try {
            SSLSocket s = (SSLSocket) ss.accept();
            OutputStream outstream = s.getOutputStream();
            PrintWriter out = new PrintWriter(outstream);
            URL url = new URL("http://www.whut.edu.cn"); //这里只是简单的示例,默认用户均可访问这个网站
            BufferedReader in = new BufferedReader(new InputStreamReader(url.openStream()));
            String line;
            while ((line = in.readLine()) != null) {
                System.out.println(line);
                out.write(line); //每当从http://www.whut.edu.cn读取一行就写入sslserversocket一行
                out.flush();
            }
            out.close();
            in.close();
            s.close();
            String[] tempcipher = ss.getEnabledCipherSuites(); //这里可知道ssl通道支持的ssl加密算法
            for (int i = 0; i < tempcipher.length; i++)
                System.out.print(tempcipher[i]);
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

```
e.printStackTrace();
}
```

在该程序运行前还需要加入参数(此处需要利用java的keytool命令来生成所需的密钥库和信任库,有关keytool详细使用可参考其它资料),命令行如下:

```
java -Djavax.net.ssl.keyStore = $home/.keystore -
Djavax.net.ssl.keyStorePassword = * * * * * https
```

在IE地址栏中键入https://localhost:8080/就可以接收到来自http://www.whut.edu.cn网页的内容,可以看到原来没有被加密的网页已经变成了加密的网页,可以看到浏览器任务栏出现一个小锁,表明信息已被加密。

3.3 SSL安全代理的关键技术

1) JSSE。在前面的SSL服务器端的模拟程序就用到了JSSE。

2) 密钥管理。密钥是很重要的,从数字签名到数据流的加密都属于此范畴。密钥库是一个文件,用来保存一组密钥和证书。根据约定,此文件称为.keystore。在前面的SSL服务器模拟程序中就用到了密钥库.keystore。Keytool是JRE所提供的工具(有关keytool的详细信息,可以到Sun公司的网站去查找http://java.sun.com),利用此工具,你可以创建新的密钥,导入数字证书,输出现有密钥,而且与密钥管理系统的交互也往往需要此工具完成。


4 总 结

随着Internet尤其是电子商务的迅速发展,网络信息安全得到人们的空前重视。SSL(Secure Sockets Layer)协议已成为Internet上进行信息发布和电子商务活动所使用的最主要的安全通讯协议,安全通讯代理(Proxy)为Web浏览器、服务器和其它应用系统提供高强度安全数据通道,可作为电子商务系统信息安全支持平台。

参考文献

- 1 谭毓安. 在Java中实现端到端的加密. 计算机应用研究, 2002(8):149~150
- 2 张峰岭. 基于Java2的身份认证数字签名和SSL实现技术. 现代计算机, 2002(2):27~31
- 3 Steve Burnett Stephen Paine 著. 密码工程实践指南. 北京:清华大学出版社, 2001. 206~220

SSL安全代理的Java实现

作者: [丁月华](#), [熊前兴](#)
作者单位: [武汉理工大学, 武汉, 430063](#)
刊名: [交通与计算机](#) 
英文刊名: [COMPUTER AND COMMUNICATIONS](#)
年, 卷(期): 2003, 21 (6)
被引用次数: 2次

参考文献(3条)

1. [Steve Burnett Stephen Paine](#) [密码工程—实践指南](#) 2001
2. [张峰岭](#) [基于Java2的身份认证数字签名和SSL实现技术](#)[期刊论文]-[现代计算机:下半月刊](#) 2002 (04)
3. [谭毓安](#) [在Java中实现SSL端到端加密](#)[期刊论文]-[计算机应用研究](#) 2002 (08)

引证文献(2条)

1. [顾成威](#) [SSL技术研究及其安全代理的设计](#)[学位论文]硕士 2006
2. [杨国建](#) [J2EE连接件技术研究与应用](#)[学位论文]硕士 2005

本文链接: http://d.g.wanfangdata.com.cn/Periodical_jtyjsj200306031.aspx