基于混合加密的即时通讯系统设计与实现

景 旭[a,b],唐晶磊[a,b],何东健[a],吴 博[a],黄少立[a]
(西北农林科技大学 a. 信息工程学院;b. 机械与电子工程学院,陕西 杨凌 712100)

[摘 要] 针对即时通信中的安全问题,提出基于混合加密的即时通信方案,论述了混合加密方案和即时通信的工作流程,给出了基于 Java 平台的实现过程。测试结果表明,方案能够预防监听、重放等攻击,保障即时通信中的信息安全。

[关键词] 即时通信;混合加密;socket 通信;Java 平台
[中图分类号] TP309.2 [文献标识码] A [文章编号] 1671-9387(2007)09-0229-06

Design and implementation of instant message system based on mixed cryptography

JING Xu[a,b], HE Dong-jian[a,b], TANG Jing-lei[a], WU Bo[a], HUANG Shao-li[a]
(a. College of Information Engineering; b. College of Mechanical and Electnnics Engineering, Northwest A & F University, YangLing, Shaanxi 712100, China)

Abstract: To solve the safety problem of IM, a scheme of instant message is brought forward on the foundation of mixed cryptography. The scheme of the mixed cryptography of instant message flow is expounded. The implementation based on java platform is given. It is testified that the scheme can prevent from the attacking of listening and replaying, and strengthen the safety of instant message.

Key words: instant mssage; mixed cryptography; socket communications; jave platform

即时通信(Instant Message, IM)是基于 Internet 的网络应用,是继电话、E-mail 后最重大的通信方式变革[1]。据全球著名的市场调研公司 IDC(Internet Data Center)估计,到 2008 年全球 IM 用户将超过 5 亿。另一家市场调研公司 Radicatic 预测,企业 IM 用户届时将接近 8 000 万[2-3]。IM 虽有强劲的发展势头和前景,但其设计安全级别很低、用户缺乏安全防护意识与知识、应用广泛等原因导致 IM 系统存在大量的安全问题。目前,有很多研究分析了 IM 的安全缺陷和所面临的安全威胁[4-6],但是大多数研究仅简单罗列了这些安全攻击现象。针对 IM 系统的安全问题,本研究提出用混合加密方案确保 IM 中的信息安全,并介绍了基于 Java 平台的实现过程,以期促进 IM 的发展和应用。

1 混合加密方案

1.1 对称加密和公钥加密[7]

加密是从明文到密文的变换过程。计算机上数据的加、解密变换是由密钥控制实现的。根据加、解密过程使用的密钥是否相同,可将现代密码技术分为对称加密算法和公钥加密算法两类。对称加密算法中,加、解密使用的密钥相同,加、解密速度快,算法简便高效,密钥简短,但其安全性完全依赖于密钥。而密钥管理是对称密码的一个严重瓶颈。在公钥加密算法中,加、解密使用不同的密钥,几乎不可能从一个密钥推导出另一个密钥,它适应于网络开

*[收稿日期] 2006-08-29
[基金项目] 西北农林科技大学人才基金项目(01140401)
[作者简介] 景 旭(1971-),男,陕西礼泉人,博士,主要从事流媒体、MAS 的网络合作及网络安全研究。
[通讯作者] 何东健(1957-),男,陕西西乡人,教授,博士,主要从事图像分析与识别、智能化检测与控制、虚拟现实技术与应用研究。

放性要求,密钥管理简单,但算法复杂,加、解密效率低。

1.2 混合加密通信过程

混合加密方案利用公钥密码传输对称密码的密钥,用对称密码进行通信,解决了通信中的效率和安全问题。混合加密方案的通信过程可以分为公钥密码传输对称密钥和对称密码通信两个阶段,基于混合加密的安全通信模型如图 1 所示。

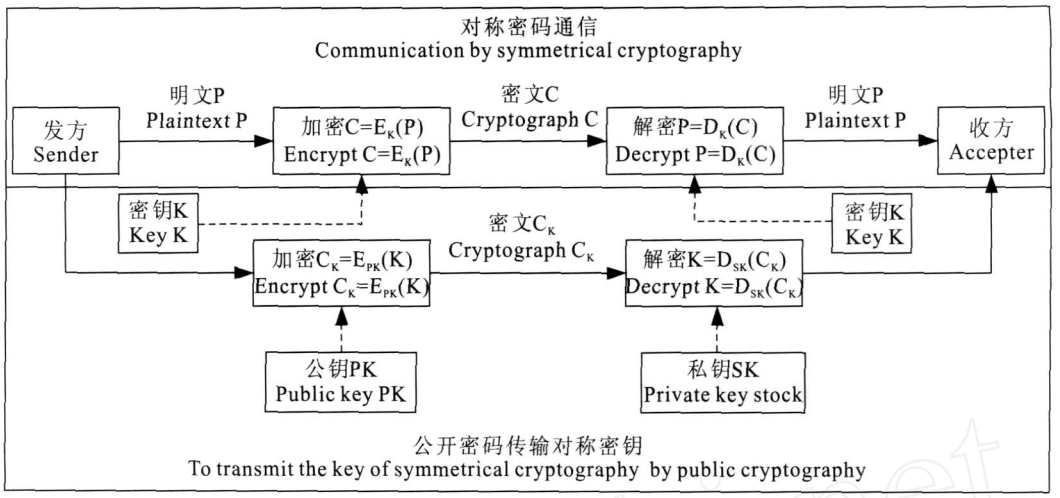

图 1 基于混合加密的安全通信模型

Fig. 1 Model of secure communication based on mixed cryptography

(1) 公钥密码传输对称密钥。公钥密码传输对称密钥的通信流程如下:

第一步,发方和收方分别在本地生成自己的密钥对(公钥 P_K 和私钥 S_K),安全保存 S_K,公开发布 P_K;

第二步,通过检索公钥目录,发方获得收方的 P_K;

第三步,发方生成对称密钥 K;

第四步,发方用收方的 P_K 加密 K 得到 C_k,将 C_k 发给收方;

第五步,收方用自己 S_K 解密 C_k,获得密钥 K。

这样,收发双方拥有共享密钥 K,安全地完成了密钥的分发。

(2) 对称密码通信。收发双方利用共享密钥 K 通信的流程如下:

第一步,发方用 K 加密消息 P 得到密文 C;

第二步,发方将 C 发送给收方;

第三步,收方用 K 解密 C 获得消息 P,即可进行保密通信;

当一个密钥 K 被安全分发后,在安全期内,混合加密系统利用密钥 K 进行对称密码通信。当由于 K 被泄露或过了 K 的有效期需要更换密钥时,才再次利用公钥密码分发新的密钥 K。

2 即时通信的工作流程

基于混合加密的 IM 系统由用户端、服务器端和数据库 3 部分组成(图 2)。

由图 2 可知,即时通信的工作流程如下:

第一步,用户生成公钥对(公钥 P_K 和私钥 S_K),安全保存 S_K,将 P_K 保存在文件中;

第二步,用户登陆到服务器 S,将 P_K 和注册信息封装为一个数据包发送给 S;

第三步,S 将注册信息及 P_K 存放在 XML 文件中,用户的 Socket 信息被保存到 S 的 Socket 池中;

第四步,S 将 Socket 池的在线用户列表以 Hashtable 的形式分发给在线用户,为了及时更新用户端的在线用户,S 以一定的时间间隔(100 ms)向用户端分发最新的在线用户列表。

第五步,用户 A 从在线用户中选择会话用户 B,向 S 请求 B 的 P_K;

第六步,B 同意 A 的会话请求后,S 将 B、A 的 P_K 分发给 A、B;

第七步,A、B 分别用对方的 P_K 和自己的 S_K 生成密钥 K,由于只有用户知道 S_K,所以除通信双方外,包括 S 在内的任何第三方都无法了解 K;

第八步,A、B 间用 K 安全通信;

第九步,通信完成后,A、B 中任何一方可以中止会话,S 更新 Socket 池在线用户列表。

3 即时通信系统的实现

本研究基于 Java 平台实现了一个安全即时通信系统。系统中用户端和服务器端的连接采用 TCP Socket,数据库访问采用 XML,公钥密码采用 Diffie-

Hellman(DH)协议,对称密码采用 DES 加密标准。

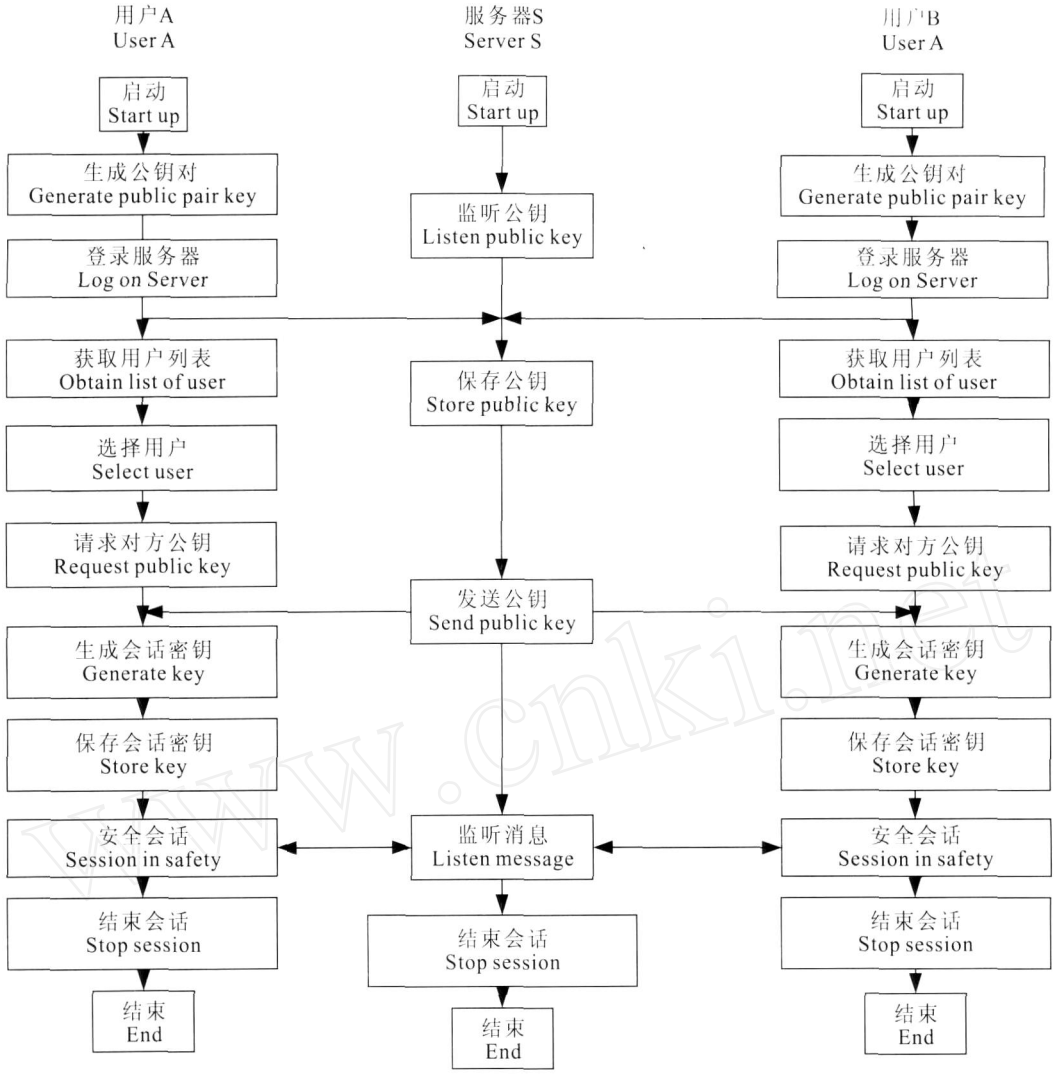

图 2　基于混合加密的即时通信系统流程图

Fig. 2　System flow chart based on mixed cryptography

3.1　Socket 连接的实现

在建立 Socket 连接的过程中,客户端向服务器端发出连接请求,服务器端监听到请求后向客户端返回接受消息。连接建立后,客户端和服务器端就可通过 Socket 连接双向通信。Socket 连接的具体实现过程如下:

(1)创建 Socket 服务器。ServerSocket 类支持创建 Socket 服务器。其代码为:

```
ServerSocket server = new ServerSocket(2136);
/ * Socket 服务器端口 2136 监听。端口的分配必须是唯一的,端口范围一般是 1024～65535。 */
```

(2)建立 Socket 连接。Socket 类支持创建 Socket 对象。其代码为:

```
Socket client = new
Socket(InetAddress.getLocalHost(),2136);
/ * InetAddress 类描述了 IP 地址格式 */
```

(3)数据传输。I/O 操作提供了对字节流、Unicode 的读者(Reader)和写者(Writer)。其代码为:

```
BufferedReader in = new BufferedReader(new
InputStreamReader(server.getInputStream()));PrintWriter out = new
PrintWriter(server.getOutputStream());
```

3.2　数据的封装

系统中需要传输的数据有消息/命令和公钥文件二进制两类。不同的数据格式其使用要求不同。本系统采用两个 Socket 通道分别处理一类数据格式,在端口 2136 监听并处理消息/命令,在端口 3247 监听并处理二进制数据。

(1)消息/命令数据。系统中采用 Hashtable 处理消息/命令数据。消息/命令数据包的封装格式

如图 3 所示。

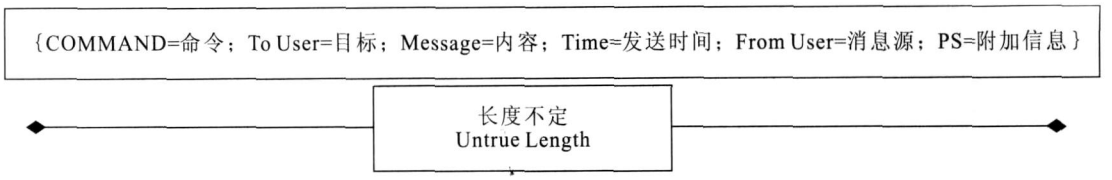

图 3 消息/命令数据包的封装格式

Fig. 3 Format of messgae/order package

消息/命令数据封装的实现代码为:

```
Public void sendPbk(String command,String toUser,String msg,String fromUser,Socket socket,Object ps)throws IOException{
ObjectOutputStream out = new
ObjectOutputStream(socket.getOutputStream());
Date time = new Date();
Hashtable hashkey = new Hashtable();
hashkey.put("COMMAND",command);
hashkey.put("ToUser",toUser);
hashkey.put("Message",msg);
hashkey.put("Time",time);
hashkey.put("FromUser",fromUser);
hashkey.put("PS",ps);
out.writeObject(hashkey.toString());
out.flush();
}
```

(2) 二进制数据。系统中采用 Data 流处理二进制数据。二进制数据包的封装格式如图 4 所示。

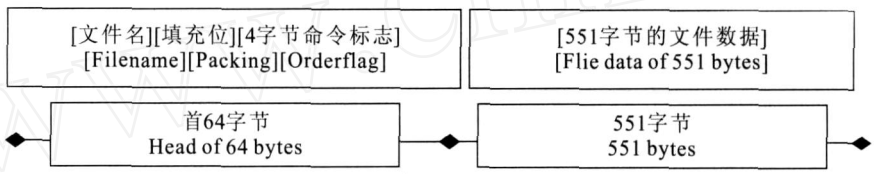

图 4 二进制数据包的封装格式

Fig. 4 Format of binary data package

二进制数据流封装的实现代码为:

```
InputStream ips = socket.getInputStream();
BufferedInputStream br2 = new BufferedInputStream(ips);
DataInputStream dis = new
DataInputStream(br2);
byte[]buf = new byte[551];/*缓冲大小*/
byte[]bufName = new byte[64];
if(dis.available()>=64){
dis.read(bufName);
}
StringBuffer filenameBF = new StringBuffer();
/*读取首 64 位的前 60 位提取文件名*/
for(inti=0;i<bufName.length-4;i++){
if(bufName[i]!=0){
filenameBF.append((char)bufName[i]);
}}
/*提取首 64 位的后四位*/
Boolean sflag = true;
for(inti=0;i<4;i++){
if(bufName[60+i]!=1){/*如果后四位有一位不是 1 则表示不是请求*/
sflag = false;
}}
String fn = filenameBF.toString();
flag = true;
if(sflag){
requester = "KeyServer:\用户请求获取"+fn;
sendDat(fn+"pbk.dat");/*把公钥文件发送回去*/
}else{
dos = new FileOutputStream(fn);
if((dis.read(buf))!=-1){
dos.write(buf,0,buf.length);
}
```

```
dos.flush();
```

3.3　公钥密码分发对称密钥的实现

系统中采用 DH 分发对称密钥,公钥密码分发对称密钥的实现过程如下:

(1) DH 密钥对的生成。实现代码为:

```
DHParameterSpec DHP = new
DHParameterSpec(skip1024Modulus,skip1024Base);
/ * skip1024Modulus 指定 DH 的模,skip1024Base
指定 DH 的基数 */
KeyPairGenerator
KpairGen = KeyPairGenerator.getInstance("DH");/
*创建密钥对生成器 */
KpairGen.initialize(DHP);/ *初始化密钥生
成器 */
KeyPair Kpair = KpairGen.generateKeyPair();/ *
生成密钥对 */
PublicKey pbk = Kpair.getPublic();/ *获得
公钥 */
PrivateKey prk = Kpair.getPrivate();/ *获得
私钥 */
```

(2) 创建共享密钥。实现代码为:

```
/ *读取自己的私钥和对方的公钥 */
FileInputStream fis = new
FileInputStream(args[0]);
ObjectInputStream obj = new
ObjectInputStream(fis);
PublicKeypbk = (PublicKey)obj.readObject();
FileInputStream f = new FileInputStream(args
[1]);
ObjectInputStream ob = new ObjectInputStream
(f);
PrivateKey prk = (PrivateKey)ob.readObject
();
KeyAgreement
ka = KeyAgreement.getInstance("DH");/ *创建
密钥协定对象 */
ka.init(prk);/ *初始化密钥协定对象 */
ka.doPhase(pbk,true);/ *执行密钥协定 */
byte[]b = ka.generateSecret();/ *生成密钥
*/
```

3.4　对称密码通信的流程

利用公钥密码传输的对称密钥就可进行保密通信。对称密码通信的具体流程如下:

```
Ciphercp = Cipher.getInstance("DES");/ *创建密
码器 */
cp.init(Cipher.ENCRYPT_MODE,dd.desKey());/
*初始化密码器 */
FileInputStreamsin = newFileInputStream(args[0]);/
*获取要加密的内容 */
FileOutputStreamfos = newFileOutputStream(args
[1]);/ *创建加密的输出流 */
CipherOutputStreamcout = newCipherOutputStream
(fos,cp);/ *创建 CipherOutputStream 对象 */
while((b=in.read())!=-1){
cout.write(b);
}/ *写输出流 */
```

4　系统测试与分析

本系统是基于 Java 平台实现的,具有与平台无关性。选用 2136 端口作为消息监听端口,3247 端口作为二进制文本传输端口,进行模拟通信,测试效果如图 5～10 所示。从图 5～10 可以看出,本系统具有安全、方便的特点。

图 5　服务器启动窗体

Fig. 5　Windows of server startup

图 6　服务器监视窗体

Fig. 6　Windows of server monitor

图 7　用户登录窗体

Fig. 7　Windows of user login

图 8 公钥窗体

Fig. 8 Windows of public key

图 9 消息发送窗体

Fig. 9 Windows to send message

图 10 密文接收窗体

Fig. 10 Windows to accept message

5 结 论

本研究分析了即时通信中的安全问题,提出基于混合加密的即时通信方案,在 Java 平台上实现了 Socket 通信的建立、公钥的管理、对称密钥的分发及对称密码通信等。测试结果表明,采用混合加密不仅能够解决 IM 中的信息安全问题,而且能防止窃听、重放等攻击。

[参考文献]

[1] 代印唐,张世永. 即时通信安全研究[J]. 电信科学,2006(4):10-16.

[2] Steven J. Presence technology more than just instant messaging [J]. IEEE Internet Computing,2003,7:1211-1222.

[3] Chery S M. M means bussing[J]. IEEE Spectrum,2002,39:1010-1018.

[4] Joe L. Best practices for instant messaging in business[J]. Network Secturity,2005,5:4-7.

[5] Harlan C. Instant messaging investigation on a live windows xp system[J]. Digital investigation,2004,1:8-12.

[6] Retuers. IM service shut down by worm[J]. computer fraud & Security,2005,4:102-110.

[7] Stallings W. 密码编码学与网络安全[M]. 3 版. 刘玉珍,王丽娜,傅建明,等译. 北京:电子工业出版社,2004.