

《物联网安全技术》

课程设计文档

组 长： 程亚凡 组员： 龚子贤

组 员： 易长胜 组员： 王子通

院（系）： 计算机学院 专业： 网络工程

指导教师： 姚 宏 职称： 副教授

2017 年 6 月

**目录**

**[第一章 需求分析](#_Toc485363226)** [1](#_Toc485363226)

**[第二章 软硬件设备](#_Toc485363227)** [1](#_Toc485363227)

**[第三章 概要设计](#_Toc485363228)** [2](#_Toc485363228)

**[第四章 详细设计](#_Toc485363229)** [3](#_Toc485363229)

[4.1 客户端 3](#_Toc485363230)

[4.1.1 客户端职责 3](#_Toc485363231)

[4.1.2 客户端结构设计 4](#_Toc485363232)

[4.1.3客户端方法设计 4](#_Toc485363233)

[4.2 认证服务端（AS） 5](#_Toc485363234)

[4.2.1 认证服务器职责 5](#_Toc485363235)

[4.2.2 认证服务器方法设计 5](#_Toc485363236)

[4.3 票据授权服务器（TGS） 5](#_Toc485363237)

[4.3.1票据授权服务器的职责 5](#_Toc485363238)

[4.3.2 票据授权服务器的方法设计 5](#_Toc485363239)

[4.4应用服务器（V） 6](#_Toc485363240)

[4.4.1 应用服务器的职责 6](#_Toc485363241)

[4.4.2 应用服务器的方法设计 6](#_Toc485363242)

[4.5 包处理程序 6](#_Toc485363243)

[4.5.1 包处理程序的功能 6](#_Toc485363244)

[4.5.2 主要成员变量及方法设计 6](#_Toc485363245)

[4.5.3 包头部结构图 8](#_Toc485363246)

**第一章 需求分析**

Kerberos是一种网络认证协议，其设计目标是作为一个可信任的第三方，通过密钥系统为客户机/服务器应用程序提供强大的认证服务。即通过Kerberos对客户机进行Authentication，以判断客户机是否能够安全访问网络中的应用。

Kerberos需要三个子协议，即客户机Client与认证服务器As，客户机Client与授权服务器TGS，客户机Client与应用服务器Server。当Client想要访问一个特定网络中的应用服务器时，需要向AS发送一个Authentication Service Request数据包，来实现对Client身份的确认，AS中有与Client共享的一个对称密钥，数据包中大体包含Client的ID号，即将访问的TGS的ID号；AS收到该包并确认身份后，将一份Authentication Service Response数据包发送给Client，该数据包包括获得TGS票据的Ticket；然后Client向TGS发送Ticket Granting Service Request数据包，数据包的内容大体包含获得TGS票据的Ticket，Client的ID号，目的应用服务机的ID号；TGS在收到Client的数据包之前，与AS共同维护一个对称密钥，用此密钥对数据包进行解密认证，并发送一个由Server与TGS共同维护的密钥加密的Ticket给Client；Client收到该数据包后，发送上述收到的Ticket到Server，当收到Server的反馈信息后，就可以摆脱Kerberos，并和Server之间自由进行交互。

Kerberos通过第三方认证的方式，当某个Server收到Client的Ticket，该Server就可以根据Ticket实现对Client的验证，无需重复认证；同时Client与Server也实现了双向认证，即Client在访问Server的资源之前，可以要求对Server的身份执行认证。

该系统应满足安全性、可靠性、透明性以及可伸缩性四大性能。

**第二章 软硬件设备**

硬件：

至少4台PC机，分别充当客户端、认证服务器（AS）、票据授权服务器（TGS）、应用服务器（V）。

软件：

源代码管理工具：Git

编程环境：eclipse

**第三章 概要设计**

该系统包含四个部分：客户端、认证服务器（AS）、票据授权服务器（TGS）和应用服务器（V）。其相互之间的网络交互关系如图一所示。



图3- 1 系统结构

状态图：



图3- 2 状态机图示

时序图：



图3- 3 时序图

**第四章 详细设计**

**4.1 客户端**

4.1.1 客户端职责

客户端未获得认证ID之前，需要到认证服务器注册（包括用户名和用户口令）。

所以客户端承载这用户输入、发送、接收以及处理票据信息等多项职责。

客户端在整个认证过程中具有如下三个方面的职责：

（1）向认证服务器提请认证。客户端向认证服务器发送一条含有客户端ID和将要访问的票据服务器的ID的明文请求认证消息；然后等待认证服务器响应，并接收含有访问票据服务器的票据。

（2）向票据服务器申请访问应用服务器票据。将从认证服务器拿到的票据发送给票据服务器；然后等待票据服务器对票据的认证，之后接收票据服务器发放的访问应用服务器的票据。

（3）向应用服务器提请认证。将从票据服务器获得的票据发送给应用服务器；接受应用服务器认证结果。认证成功就可以通信了。

4.1.2 客户端结构设计

为了降低程序的耦合程度，将客户端按照功能不同划分为3个部分：

（1）控制台程序。实现信息的发送和接收，并将相关参数传递给处理程序。

（2）处理程序。根据自定义应用层协议，处理相应信息。主要功能为票据的处理。

（3）登录程序。接受用户信息的输入，并将参数传递给控制台。

客户端的整体流程如下：



图4- 1 客户端处理流程

4.1.3客户端方法设计

客户端的控制台的功能比较单一，都是将接收的信息传递给处理程序处理。接口参数需要包括一下字段：

用户名【userName】

用户口令【userPassword】

服务器ID【serverID】

服务器地址【serverIP】

函数接口的详细设计：

public client()

void login()：登录系统，建立连接

Send(String msg，String serverIP)：向serverIP发送消息msg。

String receive(String serverIP)：从serverIP接收消息，并将消息返回。

详细设计：

运行客户端程序，首先显示登陆界面，将四个参数：用户ID、用户口令、服务器ID号和服务器IP地址输入，与AS端的数据库中的数据进行匹配，完成登陆操作，改变状态机的模式。建立与AS的Socket连接，准备消息发送。将C的ID号、TGS的ID号和当前的时间戳TS1进行拼接，再加上首部，完成对消息的封装，调用Send()函数发送给AS。

C收到AS发送的数据包，先把首部切分保留，用C与AS共同维护的密钥K（C，AS）进行解密，再根据首部将数据部分进行切分得到Ticket（TGS）和C与TGS共享的会话密钥K（C，TGS）。

C将V的ID号、Ticket（TGS）和Authenticator进行拼接，再加上首部发送到TGS。

C收到TGS发送的数据包，先把首部切分保留，用C与TGS共同维护的密钥K（C，TGS）进行解密，再根据首部将数据部分进行切分得到Ticket（V）和C与TGS共享的会话密钥K（C，V）。

C将Ticket（V）和Authenticator进行拼接，再加上首部发送到TGS。当收到V发送的确认后，完成认证过程。

C将矩阵发送到V，V进行矩阵计算并将结果返回给C，完成云计算应用操作。

**4.2 认证服务端（AS）**

4.2.1 认证服务器职责

认证服务器主要负责下面三个部分的职责：

（1）接收客户端发送的认证请求。当客户端发送认证请求，认证服务器对请求信息处理，并获取客户端的ID号。

（2）将认证请求中的相应信息到数据库中匹配。将从认证消息中那个获取的ID号与数据库中的合法用户ID进行匹配。

（3）给客户端返回授权票据。当与数据库中的合法用户匹配上后，认证服务端封装票据，将票据发送回客户端。

4.2.2 认证服务器方法设计

函数接口的详细设计：

Register()：客户注册

Send(String msg，String serverIP)：向serverIP发送消息msg。

String receive(String serverIP)：从serverIP接收消息，并将消息返回。

Boolean Find(userName)：在数据库中匹配相应的userName。返回是否匹配成功，如果匹配成功返回true，匹配失败返回false。

详细设计：

执行AS端的程序，首先要进行服务认证注册，即对C的注册认证，输入C的ID号、C的口令并生成一个C与AS共同维护的K（C），写入到数据库中。

在AS上运行多线程，实现程序的并发执行，持续监听。

同时AS与TGS共同维护一个K（TGS）密钥。

当AS收到C发送的数据包，根据首部将数据包进行切分，得到C的ID号和TGS的ID号，与AS的本地数据库中的信息进行匹配，完成验证。AS生成一个由C与TGS共享的会话密钥K（C，TGS），将K（C,TGS）、C的ID号、AD、TGS的ID、时间戳TS2和生存时间Lifetime2，利用利用AS与TGS共同维护的密钥进行加密得到Ticket（TGS）；AS将C与TGS共享的会话密钥K（C，TGS）、TGS的ID、时间戳TS2、生存时间Lifetime2和Ticket（TGS）进行拼接，再加上首部，封装成数据包并发送给C。

**4.3 票据授权服务器（TGS）**

4.3.1票据授权服务器的职责

票据授权服务器主要负责下面三个部分的职责：

（1）与AS共同连接并维护一个中央数据库存放用户口令、标识等重要信息；

（2）实现应用服务器的注册；

（3）通过Authenticator验证客户端呈现的票据有效，与客户端进行服务器授权票据交换，将应用服务器的许可票据发给客户端。

4.3.2 票据授权服务器的方法设计

函数接口的详细设计：

void Register()：应用服务器注册

Send(String msg，String serverIP)：向serverIP发送消息msg。

String receive(String serverIP)：从serverIP接收消息，并将消息返回。

Boolean Find(userName)：在数据库中匹配相应的userName。返回是否匹配成功，如果匹配成功返回true，匹配失败返回false。

详细设计：

运行TGS端的程序，首先需要完成服务器的注册，输入V的ID号、V的密码并生成一个由TGS和V共同维护的密钥K（V）。

在TGS上运行多线程，实现程序的并发执行，持续监听。

当TGS收到C发送的数据包。根据首部进行切分得到V的ID号和Ticket（V），Ticket（V）利用AS与TGS共同维护的密钥进行解密；

TGS生成一个由C与V共享的会话密钥K（C，V），将K（C，V）、C的ID号、AD、TGS的ID、时间戳TS2和生存时间Lifetime2，利用V与TGS共同维护的密钥进行加密得到Ticket（V）；AS将C与V共享的会话密钥K（V，TGS）、V的ID、时间戳TS4、生存时间Lifetime4和Ticket（V）进行拼接，再加上首部，封装成数据包并发送给C。

**4.4应用服务器（V）**

4.4.1 应用服务器的职责

（1）与Kerberos服务器共享一个特定的密钥；

（2）通过Authenticator验证客户端呈现的票据有效，并与客户端进行相互认证，认证成功则提供后续服务。

4.4.2 应用服务器的方法设计

Send(String msg，String serverIP)：向serverIP发送消息msg。

String receive(String serverIP)：从serverIP接收消息，并将消息返回。

详细设计：

运行V端的程序，（有没有界面？？）

在V上运行多线程，实现程序的并发执行，持续监听。

收到C发送的数据包，根据首部进行切分，得到Ticket（V），利用TGS与V共同维护的密钥K（V）进行解密，得到由TGS生成的C与V共享的会话密钥、C的ID号、AD、V的ID号、时间戳TS4和生存时间。

用C与V共同维护的密钥K（C，V）对当前的时间戳TS5+1进行加密发送给C，进行相互认证。

认证完成后，接收C发送的计算请求，在V端进行计算操作后，将结果返回到C，完成云计算操作。

**4.5 包处理程序**

4.5.1 包处理程序的功能

将控制台提交的信息进行处理，按照规定的应用层协议，将消息进行分解，得到需要的字段。

4.5.2 主要成员变量及方法设计

主要成员变量：

Private：

int src\_id\_mark； //原id标志位

int des\_id\_mark； //目的id标志位

int timestamp\_mark； //时间戳标志位

int lifetime\_mark； //生命周期标志位

int ticket\_mark； //票据标志位

int authenticator\_mark；//授权认证标志位

int key\_mark； //session key标志位

String src\_id； //原id

String des\_id； //目的id

Data timestamp； //时间戳

int lifetime； //生命周期

String ticket； //票据

String authenticator；//授权认证

String key； //session key

String dataPacket；

final int SRC\_ID\_LENGTH=8；

final int DES\_ID\_LENGTH=8；

final int TIMESTAMP\_LENGTH=6；

final int LIFETIME\_LENGTH=4；

final int AUTHENTICATOR\_LENGTH=29；

final int KEY\_LENGTH=8；

public：

void setSrc\_id\_mark(int src\_id\_mark) 设置原id标志位

int getSrc\_id\_mark() 获取原id标志位

void setDes\_id\_mark(int Des\_id\_mark) 设置目的id标志位

int getDes\_id\_mark() 获取目的id标志位

void setTimeStamp\_mark(int timestamp\_mark) 设置时间戳标志位

int getTimeStamp\_mark() 获取时间戳标志位

void setLifeTime\_mark(int lifetime\_mark) 设置生命周期标志位

int getLifeTime\_mark() 获取生命周期标志位

void setTicket\_mark(int ticket\_mark) 设置票据标志位

int getTicket\_mark() 获取票据标志位

void set Authenticator\_mark(int authenticator\_mark)

设置授权认证标志位

int getAuthenticator\_mark() 获取授权认证标志位

void setKey\_mark(int key\_mark) 设置session key标志位

int getKey\_mark() 获取session key标志位

String getSourceID(String msg)：获取包中发送方的ID。

String getDesID(String msg)：获取包中目的方的ID。

String getKey(String msg)：获取包中的密钥。

String getTimestamp(String msg)：获取包中的时间戳。

String getLifeTime(String msg)：获取此包的生命周期。

String getTicket(msg)：获取包中的票据。

String getAuthenticator(msg)：获取Authenticator的内容。

4.5.3 包头部结构图



图4- 2 包头结构

当src\_id\_mark为1时，表示数据段有源ID字段；为0，则表示无；

当des\_id\_mark为1时，表示数据段有目的ID字段；为0，则表示无；

当timestamp\_mark为1时，表示数据段有时间戳字段；为0，则表示无；

当lifetime\_mark为1时，表示数据段有生命周期字段；为0，则表示无；

当ticket\_mark为1时，表示数据段有票据字段；为0，则表示无；

当authenticator\_mark为1时，表示数据段有授权认证；为0，则表示无；

当key\_mark为1时，表示数据段有session key；为0，则表示无；

当头部序列为1110000时，此包则为客户端C发给AS的数据包；