

基于 JAAS 的身份认证系统设计与实现

程俊波¹, 卫红春¹, 刘海峰², 肖 瑜³

(1 西安石油大学 计算机学院, 陕西 西安 710065; 2 陕西科技大学 理学院, 陕西 西安 710021;

3 中国人民解放军边防学院, 陕西 西安 710018)

摘 要: 介绍了 JAAS 安全模型的特点功能及其在身份认证系统中的应用, 并基于 JAAS 设计实现了一种安全的身份认证系统, 给出了登陆模块实现核心代码. 突出了 Java 作为一种安全开发语言在安全编程中的优点以及 JAAS 在实现安全登录模块过程中的优势, 为用户身份认证及授权服务管理提供了统一解决方案.

关键词: JAAS; 身份认证; 安全编程

中图分类号: TP393.08

文献标识码: A

文章编号: 1000-7180(2014)12-0161-03

Design and Implementation of Uthentication System Based on JAAS

CHENG Jun-bo¹, WEI Hong-chun¹, LIU Hai-feng², XIAO Yu³

(1 College of Computer, Xi'an Shiyou University, Xi'an 710065, China;

2 Faculty of Science, Shaanxi University of Science and Technology, Xi'an 710021, China;

3 Border Defence Academy of P. L. A., Xi'an 710018, China)

Abstract: Characteristic of JAAS safety model and it's application in authentication system are introduced in this paper, and the authentication system is designed in detail and implemented based on JAAS, the advantage of safety programming using java language and safety login module based on JAAS are highlighted, an unified solution of user authentication and management of authorization service is provided in the paper.

Key words: JAAS; user authentication; safety programming

1 引言

身份认证和授权是信息系统安全中很重要的部分,很多系统包括操作系统、应用系统等都离不开身份认证和授权^[1]. 身份认证及授权机制和系统本身的安全机制密切联系,例如:Linux 下,用户的基本信息保存于/etc/passwd.conf 中. 用户在登录 Linux 系统时需要输入用户 ID 和口令. 登陆过程就包括了身份认证和授权机制. 但就关于认证和授权的国内自主研究而言,目前仍处于理论研究状态,如文献[2]中对于单点登录系统的身份认证协议做了改进,通过线性回归、参数配准给出了用户信息与服务器数据快速匹配的方法,就编程接口而言这仍有不便之处.

目前的身份认证机制大都基于静态口令,静态

口令机制简单,能为用户维护一个二元组信息:用户 ID、口令,用户输入自己的 ID 和口令,系统根据用户当前输入的信息和用户曾维护的信息做比较,来判断用户身份的合法性. 这种方式极易受到网络不法分子攻击,造成机密资源的泄露. 用户信息被窃取大多数发生在登录阶段. 因此,在设计身份认证系统模块时针对登录阶段的安全问题进行防范措施设计尤为重要^[3].

JAAS(Java Authentication and Authorization Service, Java 验证和授权服务)是一种为进一步增强 Java2 安全框架功能而提供的编程接口,提供了基于代码运行者的存取控制能力,保证客户端或服务端端的 Java 程序可以仅被授权用户所执行. JAAS 主要由认证和授权两大部分构成^[4].

JAAS 提供了用于用户鉴别的类和接口,同时

也提供了用于用户授权的类和接口. JAAS 采用的是插件的运行方式, 可根据应用的需要, 配置 JAAS 的配置文件, 将相关组件即可包含在应用程序中. 使用 JAAS 包接口, 开发者和第三方可以开发一些组件或者 BEAN 来实现登陆认证, 或者通过与使用者或外部系统的交互来访问认证信息(当然也可以设计更为稳妥安全的密码学协议).

本文在 Java2 安全模型的基础上, 引入了 Java 认证和授权服务思想, 实现与平台无关的分布式身份信息安全认证系统.

2 基于 JAAS 的安全模型

2.1 JAAS 架构模型

JAAS 架构模型如图 1 所示.

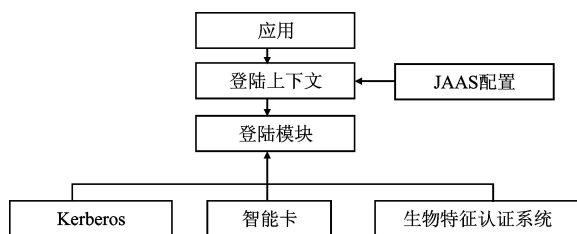


图 1 JAAS 架构模型和可插入认证

2.2 实现 JAAS 登录模块

JAAS API 中常用的一些类和接口说明如下所示.

(1) LoginModule: 确认用户的合法性, 并分配访问权限 principal 给 subject;

(2) LoginContext: 为了实现用户鉴别, 建立相应的环境, 从配置文件中导入规则;

(3) CallbackHandler: 回调处理器, 负责与用户交互, 确认其身份的合法性;

(4) Subject: 表示登陆处理的目标, 即一个被鉴别的用户, 并可关联一个或多个 principal;

(5) Principal: 表示具有访问权限的一个实体, 用来代表 Subject 的身份, 可以看作是可以执行某种操作的证件^[5].

2.3 JAAS 认证原理

(1) 设置 JAAS 配置文件;

(2) 根据 JAAS 配置文件的条目加载一个或者多个 LoginModule;

(3) 为了管理用户认证的有关过程, 将提供一个可选的 LoginModule 构造函数和一个回调处理器 CallbackHandler. 如果没有在构造函数中提供回调处理器, 系统采用默认设置;

(4) 初始并实例化 LoginContext(加载配置规则), 如果成功, 则调用 LoginContext 的 login 方法. 无论是否需要, LoginContext 都会去首先读取 JAAS 配置文件, 从中获得要加载的登陆模块信息, 其 initialize 方法将按照配置文件中的相关内容提供 LoginModule 运行所需要的信息^[1];

(5) LoginContext 的 login 方法将调用 LoginModule 的 login 方法, 确定用户身份. 该方法将设置相关的回调, 并由回调处理器 CallbackHandler 来管理登陆处理回调;

(6) LoginModule 的 login 方法将负责与用户进行交互(可能是人机交互, 也可能是机机交互), 如果用户输入信息无效, 则该方法返回 FALSE, 一次交互过程结束, 如果用户输入信息有效, 则该方法将设置 Principal 对象的 Subject 对象, 并返回 TRUE; 当然 LoginModule 也可以将与用户之间的所有交互过程全部委托给处理器 CallbackHandler 来处理. 如果登陆成功, LoginContext 将调用 LoginModule 的 commit 方法将结果提交给 LoginModule 实例的内部状态^[1].

为了满足可插拔性, JAAS 是可堆叠的. 在单一登录的情况下, 一组安全模块可以堆叠在一起, 然后被其他的安全机制按照堆叠的顺序被调用. JAAS 的实现者根据现在一些流行的安全结构模式和框架将 JAAS 模型化. 例如可堆叠的特性同 UNIX 下的可堆叠验证模块 (Pluggable Authentication Module, PAM) 就非常相似. 从事务的角度看, JAAS 类似于双步提交 (Two-Phase Commit, 2PC) 协议的行为. JAAS 中安全配置的概念, 包括策略文件 (Policy File) 和许可 (Permission) 来自于 J2SE. JAAS 还从其他成熟安全的框架中借鉴了许多思想^[6].

3 身份认证系统设计 with 实现

3.1 安全登录过程模型

安全登陆过程模型如图 2 所示.

基于 JAAS 身份认证来实现服务器登录验证模块, JAAS 的身份认证以插入式方式工作, 这样就使得应用程序独立于底层的认证技术, 新的或者更新的认证技术可以直接插在应用程序之下而不用对应用程序本身作任何更改, 只需要更改配置文件.

应用程序认证流程如图 3 所示, 应用程序认证系统主要模块说明: User Request 表示用户访问请求; Login View 指用户登录视图; LoginAction 指用户登录动作; LoginModule 指用户登录模块; Pro-

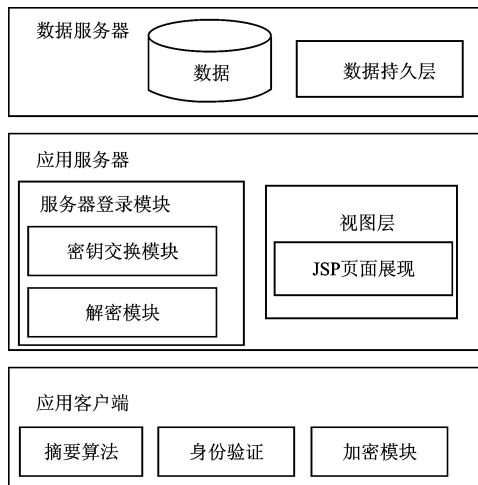


图2 安全登录过程模型

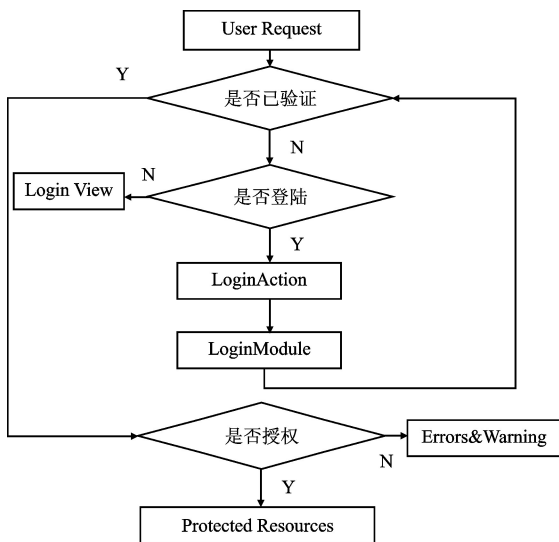


图3 应用程序认证流程

tected Resources 指服务器端机密资源,例如 JSP 资源文件,业务逻辑实现,业务功能服务等;Errors & Warnings 指错误警告信息视图。

3.2 模块详细实现

(1) 登录上下文模块:建立登陆上下文,并通过配置文件初始化,此处配置文件必须与程序同目录。

```
LoginContext loginContext = null;
loginContext = new LoginContext("simple", new SimpleCallbackHandle());
try {loginContext.login(); // 不抛出异常表示验证成功
```

```
} catch (LoginException e) { }
```

(2) 回调处理器模块实现:

```
public void handle(Callback[] callbacks) throws IOException,
    UnsupportedCallbackException
{ for (Callback callback : callbacks)
```

```
{ if (callback instanceof NameCallback)
{ NameCallback nc = (NameCallback) callback;
System.out.print(nc.getPrompt());
System.out.flush();
nc.setName((new BufferedReader(new InputStreamReader(System.in))).readLine()); }
else if (callback instanceof PasswordCallback)
{ PasswordCallback pcb = (PasswordCallback) callback;
System.out.print(pcb.getPrompt());
System.out.flush();
pcb.setPassword((new BufferedReader(new InputStreamReader(System.in))).readLine().toCharArray()); }
}
```

(3) 用户登录模块实现:

```
public boolean login() throws LoginException
{ callback[] callbacks = new Callback[2];
callbacks[0] = new NameCallback("用户名:");
callbacks[1] = new PasswordCallback("密码: ", false);
try { callbackHandler.handle(callbacks);
username = ((NameCallback) callbacks[0]).getName();
password = ((PasswordCallback) callbacks[1]).getPassword();
if (username.equals("your_username") && new String(password).equals("your_password"))
{ System.out.println("认证成功");
return true; }
else
{ System.out.println("认证失败"); }
} catch (IOException e) { }
catch (UnsupportedCallbackException e) { }
return false;
}
```

4 结束语

本文采用JAAS技术实现了用户身份认证,建立了应用系统中自己的登录机制,可以在客户端或服务端的应用程序上工作,可方便添加至新的登陆模块而无须修改之前的代码,可以简化涉及到Java Security包的程序开发。JAAS通过一个可扩展的框架SPI(服务者提供接口)来保证程序独立于安全机制,应用程序独立于底层认证技术,为用户身份认证及授权服务管理提供了统一解决方案。在实际应用系统中,还可根据现场情况提出更多的安全需求,进一步建立基于JAAS的安全性扩展。

(下转第167页)

比标准 DES 算法节约了 2/3 的时间,在 50 MHz 时钟频率下,改进的 DES 算法的处理性能可达到 533 Mb/s(表 1)。

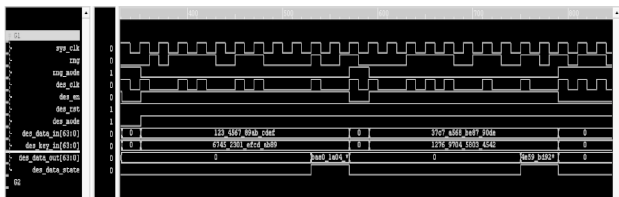


图 5 DES 算法使用随机时钟仿真波形

表 1 标准 DES 与改进 DES 算法面积与性能比较

	面积/ μm^2	处理性能/ (Mb/s)	加解密所需 时钟周期
标准 DES 算法	17 917	178	18
改进 DES 算法	36 151	533	6(最快)/Random

5 结束语

本文在 DES 算法基础上提出一种改进的 DES 算法设计,既优化了 DES 算法的执行速度又提高了算法的安全性。仿真及综合结果表明采用改进方法实现的 DES 算法满足深亚微米条件下集成电路设计对性能和面积的要求,适用于在速度与安全性都有较高要求的加密芯片应用领域。

参考文献:

- [1] 汪翼,沈海斌,何乐年,等. DES 算法的高速流水线实现[J]. 微电子学与计算机,2003,20(8):158-160.
- [2] Stinson DR. 密码学原理与实践[M]. 北京:电子工业出版社,2009.
- [3] Zodpe H D, Wani P W, Mehta R R. Design and implementation of algorithm for DES cryptanalysis[C] // 2012 12th International Conference on Hybrid Intelligent Systems (HIS). Pune: IEEE, 2012: 278-282.
- [4] 蒋惠萍,毛志刚. 一种抗差分功耗攻击的改进 DES 算法及其硬件实现[J]. 计算机学报,2004,27(3):334-338.
- [5] Li J, Lv Y X, Sun H F, et al. A power analysis resistant DES cryptographic algorithm and Its hardware design[C] // 2012 Third International Conference on Digital Manufacturing & Automation. GuiLin: IEEE, 2012: 121-124.
- [6] 李杰,单伟伟,吕宇翔,等. 一种抗相关功耗攻击 DES 算法及 FPGA 电路实现[J]. 东南大学学报:自然科学版,2012,42(6):1063-1068.

作者简介:

温泉 男,(1983-),硕士研究生.研究方向为数字集成电路设计,通信与网络技术. E-mail: wunquan@outlook.com.
沈海鸿 女,(1970-),博士,副教授,硕士生导师.研究方向为通信网络技术研究。

(上接第 163 页)

参考文献:

- [1] 斯蒂尔. 安全模式: J2EE、Web 服务和身份认证管理最佳实践和策略[M]. 北京:机械工业出版社,2006.
- [2] 马蕾,杨洪雪,朱青松. 基于改进身份认证协议的单点登录系统研究[J]. 微电子学与计算机,2012,29(7):182-183.
- [3] 王建雄. 网站注册与登录安全设计[J]. 大众科技,2010(4):11-12.
- [4] 郑爱媛. 在 JAAS 基础上的 Java 安全编程[J]. 电脑编程技巧与维护,2008(14):59-67.

- [5] 王敏,吉逸. Java2 环境下身份认证和授权机制的研究[J]. 微机发展,2003,13(5):40-42.
- [6] 刘晓华. 精通 Java 核心技术[M]. 北京:电子工业出版社,2003.

作者简介:

程俊波 男,(1964-),硕士,高级工程师.研究方向为计算机网络与信息安全研究. E-mail: cjbcs1@126.com.
肖瑜 男,(1964-),教授.研究方向为作战仿真与虚拟现实研究。