文章编号：2096-1472(2020)-08-25-03
DOI:10.19644/j.cnki.issn2096-1472.2020.08.007

一种内外网数据交互系统的设计与实现

吴英宾

(聊城职业技术学院，山东 聊城 252000)
✉93913440@qq.com

摘　要：随着互联网的全面普及，手机、电脑等海量智能终端不断接入网络，在公网资源有限的情况下，绝大部分终端设备均运行于内网，外网一般无法直接访问内网数据。本文针对WebSocket技术的特征，设计并实现了一种基于WebSocket的内外网数据交互系统，系统通过WebSocket建立内外网数据通道，实现了数据交互共享，为内外网数据交互应用场景提供了一种有效地解决方案。

关键词：WebSocket；内外网数据交互；SpringBoot
中图分类号：TP311　**文献标识码**：A

Design and Implementation of a Data Interaction System for Intranet and Extranet

WU Yingbin

(*Liaocheng Vocational and Technical College*, *Liaocheng* 252000, *China*)
✉93913440@qq.com

Abstract: With the popularity of the Internet, massive smart terminals such as mobile phones and computers have a constant access to the network. However, in the case of limited public network resources, a large majority of terminal devices are operated on the Intranet, the data of which cannot be directly accessible from the Extranet. This paper designs and implements a WebSocket-based internal and external network data interaction system that realizes data interaction sharing by establishing internal and external network data channels through WebSocket. This system provides an effective solution to data exchange application scenarios in Intranet and Extranet.

Keywords: WebSocket; intranet data exchange; SpringBoot

1 引言(Introduction)

随着移动通信技术的迅速发展，互联网基本实现了全面普及，手机、电脑等海量智能终端均已接入了网络。在公网资源有限的情况下，绝大部分终端设备均运行于内网，外网一般无法直接访问内网数据，因此在保证安全的前提下做到外网下的应用与内网环境下的业务数据进行交互[1]，实现内外网数据交互信息共享显得尤为重要。

内网通常指局域网，外网是相对内网而言，是面向Internet部署的网络[2]。传统的内外网数据交互方案主要有NAT映射和安装反向代理软件，实现NAT映射需要网络管理员介入进行手工配置，在内网设备动态接入、数量多的情况下实施起来非常困难。安装反向代理软件虽然简便快捷，但第三方软件的介入存在数据安全隐患。基于WebSocket的内外网数据交互系统充分利用了WebSocket技术安全、高效、全双工通信的特征，提供了一种易于部署和管理的内外网数据交互方案。

2 WebSocket技术特征(WebSocket technical features)

WebSocket协议是HTML5中提出的一种基TCP的新通信协议[3],其工作过程为通过一次“握手”建立一条长连接，实现了浏览器端与服务器端之间的全双工(full-duplex)通信[4]。在基于WebSocket的通讯过程中，服务器端与客户端完全对等，双方可以主动互发消息，解决了HTTP协议下服务器端不能主动发消息给客户端的问题，真正实现了二者之间快速、

基金项目：聊城职业技术学院重点科技项目(2017LZYK03).

高效、全双工数据交互。WebSocket建立连接及通信过程如图1所示。

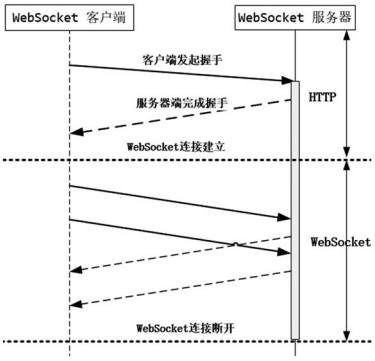

图1 WebSocket连接及通信过程

Fig.1 WebSocket connection and communication process

WebSocket作为HTML5的标准通信协议，广泛应用于B/S架构的应用中，其在建立连接之初使用HTTP协议发起“握手”请求，客户端与服务器端“握手”完成后便可以建立一条高效、持久、实时的双向数据通道，除非客户端或服务器端主动断开，这条数据通道会以长连接形式持久保持，实现生产数据实时推送，有效降低了网络吞吐量，提高了通信效率[5]。

在安全性方面，WebSocket协议提供了两种URI方案供用户选择，其中前缀为WS的URI为未加密的一般连接，前缀为WSS的URI为加密的高安全性连接。其中“WSS”连接基于安全传输层协议(Transport Layer Security Protocol，TLSP)确保WebSocket连接的保密性和数据完整性[6]。

3 系统设计(System design)

3.1 系统总体架构设计

为便捷的实现内外网数据交互共享，系统采用B/S模式，在总体架构上分为外网服务管理端、内网数据服务端和通用数据调用客户端三个部分。充分利用外网WebSocket服务作为“桥梁”打通调用客户端与内网客户端的数据通道，从而 “穿透”内网数据的访问，最终实现数据交互和共享。系统体系结构图如图2所示。

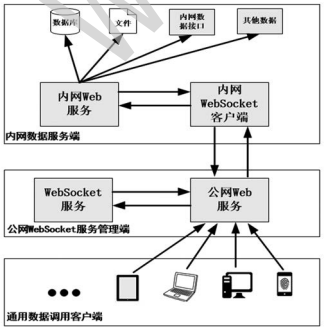

图2 系统体系结构图

Fig.2 System architecture diagram

3.2 系统功能设计

系统在功能上划分为外网WebSocket服务管理子系统和内网数据服务子系统，外网WebSocket服务管理子系统提供系统的核心服务，实现了内外网数据“桥梁”并提供统一Web服务；内网数据服务子系统整合了内网WebSocket客户端和内网Web服务，提供内网统一数据传输业务接口。系统功能结构图如图3所示。

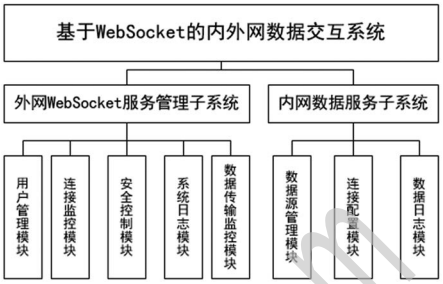

图3 系统功能结构图

Fig.3 System functional structure diagram

系统主要功能模块如下：

(1)用户管理模块。实现用户信息、用户角色和访问权限的管理功能，通过用户鉴权方可接入系统。

(2)连接监控模块。实现对已接入系统进行数据交互的WebSocket连接进行动态监控和管理。

(3)安全控制模块。实现对调用客户端的IP、访问频次等限制性规则的管理。

(4)系统日志模块。记录系统用户访问、操作、数据连接与传输等各种吸入事件记录。

(5)数据传输监控模块。实现对系统中所有数据传输类型、数据大小等信息进行监控。

(6)数据源管理模块。实现对内网可访问数据源的配置和设置，未经配置的数据源用户无法进行访问。

(7)连接配置模块。实现与外网WebSocket服务管理子系统的对接的服务地址、登录信息等配置。

(8)数据日志模块。实现内网数据传输日志记录和管理。

4 系统实现(System implementation)

4.1 系统技术选型

系统采用SpringBoot框架进行实现，通过SpringBoot整合Web服务和WebSocket服务。SpringBoot是目前业界最为流行的Java EE一站式解决方案，具有快速构建、自动配置等特点，使Spring更加易于开发和维护[7]。

SpringBoot对WebSocket服务端提供了良好的支持，只需要在Maven配置文件中加入WebSocket依赖即可，开发者在实现过程中只需关注业务需求的实现，可有效降低系统实现难度和提升开发效率。

4.2 系统鉴权的实现

为更好的保障数据安全，系统采用Token机制来实现WebSoket连接安全认证，即用户在登录成功后会得到一个Token，每次建立WebSocket连接必须在首次握手中对Token进行验证，无Token或无效Token将导致服务端直接断开，以

此来保证系统的安全性。系统鉴权流程如图4所示。

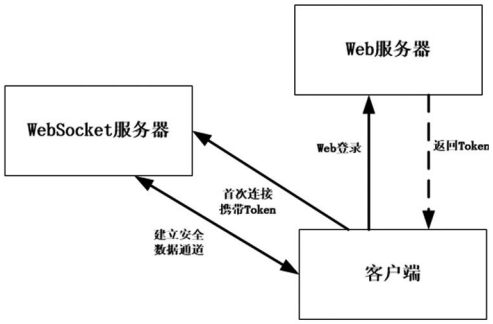

图4 系统鉴权流程

Fig.4 System authentication process

4.3 外网子系统核心功能实现

外网子系统的核心功能为WebSocket的连接、管理和数据收发。系统过添加spring-boot-starter-websocket依赖来获得WebSocket服务端的支持，通过Configuration注解轻松实现配置。利用ServerEndpoint注解可以轻松实现WebScoketServer类。用户首先通过Web登录并获得授权Token，然后只需通过路径参数中携带的用户ID即可通过WebSocketServer轻松与该ID的客户端进行通信。WebSocketServer类主要实现代码如下：

```
@ServerEndpoint("/websocket/{userId}")
@Component
public class WebSocketServer {
//存放每个客户端对应的WebSocketServer对象
private static CopyOnWriteArraySet<WebSocketServer> webSocketSet = new CopyOnWriteArraySet<WebSocketServer>();
//建立连接时验证Token
@OnOpen
public void onOpen(Session session,@PathParam("userId") String sid) {
if(verifyToken(session)){ //成功建立连接}
}
//接受数据并转发给指定客户端
@OnMessage
public void onMessage(String message, Session session) {
//根据用 户ID进行收发数据
}
```

为实现对用户连接的监控，可以通过管理连接对象、Session或Token的方式进行实现连接的手动管理，另外在onMessage及sendMessage方法中对数据进行分析和记录。

4.4 内网子系统核心功能实现

内网子系统的核心功能为与外网建立WebSocket连接和提供数据交互业务接口。与服务器端建立WebSocket连接一般用于前端网页，本系统为更好的实现对内网所有本地、设备等信息进行读取，采用Java WebSocket实现的WebSocket客户端，需要在项目的Maven配置中添加org.java-websocket依赖，只需通过继承WebSocketClient类实现一个WebSocket客户端，其主要方法代码如下：

```
public class DataTransWebSocketClient extends WebSocketClient{
//首次握手回调
public void onOpen(ServerHandshake arg0) {//握手处理}
//收到并处理数据
public void onMessage(String message) {//处理收到数据}
//发送数据到WebSocket服务器
public void send(String data) {//发送数据}
}
```

数据交互业务接口采用SpringBoot Web技术和Spring Data JPA技术进行构建和实现。

5 结论(Conclusion)

随着互联网的全面普及，信息共享变得无处不在，在保证安全的前提下进行内外网数据交互共享有着广泛的应用场景。本文通过对WebSocket技术特征进行分析，设计了一种兼具安全性、拓展性和灵活性的内外网数据交互系统，采用成熟的Java EE技术加以实现。系统以公网WebSocket服务为桥梁，使用登录和Token机制进行鉴权，内网通过配置便可与外网服务连接进行数据双向交互共享，有效降低了内外网数据交互的使用难度和部署成本，同时保障了系统的数据安全。

参考文献(References)

[1] 孟威,乔林,刘颖,等.基于电力企业移动办公的内外网数据交互[J].计算机科学与探索,2017(11):480-482.

[2] 何钰,李瑞祥.实现内外网数据交互安全[J].网络安全和信息化,2017(9):127-129.

[3] 万可达.基WebSocket的水泥厂动设备的全平台状态监测系统的研究[D].杭州:浙江大学,2018.

[4] 潘峰,王笑天.基于Redis与WebSocket的战场态势实时推送方案设计及实现[J].软件导刊,2018(7):143-146.

[5] 曹文彬,谭新明,刘备,等.基于事件驱动的高性能WebSocket服务器的设计与实现[J].计算机应用与软件,2018(1):20-27.

[6] 刘栋,黄斌,王锋,等.WebSocket技术在信息安全系统中的应用实现[J].信息安全与通信保密,2016(5):92-94.

[7] 张峰.应用SpringBoot改变Web应用开发模式[J].科技创新与应用,2017(23):30-31.

作者简介：

吴英宾(1983-)，男，硕士，讲师.研究领域：软件开发.