Design and Implementation of a SSH Proxy System Based on B/S Architecture

Wenchao Cui¹, Minghao Gou²

¹School of Control and Computer Engineering, North China Electric Power University
²School of Control and Computer Engineering, North China Electric Power University

Abstract. The SSH protocol is one of the most commonly used protocols in the current operation and maintenance process, but the commonly used SSH clients are all implemented based on the C/S architecture, which causes the operation and maintenance personnel to face cross-platform problems, and when accessing from the external network it is also necessary to develop corresponding VPN according to different platforms. This paper proposes a SSH proxy system based on B/S architecture. The client of the system is based on a browser, which solves the cross-platform problem of different clients and facilitates the use of remote operation and maintenance personnel.

1. Introduction
With the increasing role of information systems in enterprises, the construction and operation management of information systems have become accelerators of business efficiency at the same time. A wide range of network equipment, servers, middleware or business systems make information systems more and more complex[1]. How to conveniently and effectively carry out remote operation and maintenance operations has become more and more important. At present, the SSH protocol is the most commonly used remote operation and maintenance protocol in information systems. It provides transport layer support for encryption and compression during data transmission, and can ensure the safe and efficient transmission of data[2]. Common SSH client implementations are implemented in the C/S architecture, that is, users use special client software to connect remotely, but they will face the following problems: some require a special hardware platform at a cost, and clients have compatibility issues on different operating systems, so different client software needs to be developed for different system platforms (Windows, Linux, Mac operating systems, etc.).

This article discusses and designs a B/S architecture SSH proxy system. The system is proposed for the following reasons and significance: browsers on various operating system platforms and even mobile terminal platforms have already supported the HTML5 standard quite well. Although the browser is also a form of client, compared with the general dedicated client, browser support can be more flexible and changeable; without platform restrictions, users can realize remote control access more flexibly; and general implementation Compared with the method, the maintenance and upgrade of the Web-based remote access control system does not need to consider the impact of the client, and the cost is lower. It is a more centralized management method, which can further reduce the overall system management overhead for enterprises, and is more cost-effective; communication based on HTTPS and WSS protocols can also ensure the safety and efficiency of the transmission process[3].
2. Related technology introduction

2.1. SSH
SSH is the abbreviation of Secure Shell Protocol, developed by the IETF Network Working Group. SSH is a security protocol created based on the application layer and the transport layer. It aims to provide a security protocol for remote login sessions and other network services.

2.2. SSL protocol
The SSL protocol is located between the TCP/IP protocol and various application layer protocols to provide security support for data communication. The SSL protocol can be divided into two layers: SSL Record Protocol: it is built on a reliable transmission protocol (such as TCP) and provides support for basic functions such as data encapsulation, compression, and encryption for high-level protocols. SSL Handshake Protocol (SSL Handshake Protocol): It is built on top of the SSL record protocol and is used to authenticate the identity of the communicating parties, negotiate encryption algorithms, and exchange encryption keys before the actual data transmission starts[4].

2.3. WebSocket protocol
The WebSocket protocol is defined in the HTML5 standard. It is a protocol for full-duplex communication on a single TCP connection. WebSocket makes the data exchange between the client and the server easier, allowing the server to actively push data to the client. In the WebSocket API, the browser and the server only need to complete a handshake, and a persistent connection can be created directly between the two, and two-way data transmission can be carried out[5].

3. Analysis and design
The SSH proxy system mainly includes two functions: on the one hand, the proxy system establishes an HTTPS/WSS protocol connection with the front end to obtain static resources, transmit configuration information, and exchange remote operation information; On the other hand, the agent system establishes an SSH connection with the remote server, maintains the connection status and exchanges remote operation information. The deployment structure is shown in the figure 1:

![Deployment structure](image)

Figure 1. Deployment structure.

3.1. Communication process
The SSH proxy server is the core of the entire system. When users want to establish an SSH connection with a remote server, they need to build a connection channel through the proxy server. As shown in Figure 2, the user-proxy communication framework is mainly composed of two layers:
SSL protocol layer: Implement server authentication function based on SSL protocol and ensure the confidentiality, integrity and authenticity of communication data[6].

HTTP/WS protocol layer: Based on the HTTP/WS protocol, it provides functions such as user authentication, static resource acquisition, and long connection, and uses the SSL protocol layer as the basis to ensure information security[7]. On the basis of the communication framework, the user interacts with the proxy server.

When the user accesses the proxy system, the system requires the user to enter the relevant configuration information of the target remote server, including IP address, port number, user name, password, etc. After the proxy server receives the configuration information, it establishes an SSH channel with the remote server according to the configuration information, and the information at both ends is forwarded, and the overall communication process of the system is shown in Figure 3:
3.2. Proxy server architecture

The overall system architecture is shown in Figure 4. The system adopts the EventLoop thread model[8], including three main modules, namely BossEventLoopGroup, WorkEventLoopGroup and ListenerThreadPool modules.

**Figure 4. The overall structure of the system.**

- **BossEventLoopGroup module:** This module contains one or more NioEventLoops dedicated to listening to Accept, that is, front-end connection events. When polling receives a new connection request, it creates a Channel and delivers the Channel to WorkEventLoopGroup. Subsequent read and write events are registered by WorkEventLoopGroup to its management. NioEventLoop.

- **WorkEventLoopGroup module:** The module maintains multiple NioEventLoops and is responsible for registering read and write events to each NioEventLoop. Each NioEventLoop has a ChannelPipeline, and ChannelPipeline uses the responsibility chain mode to register various Handlers and execute related processing logic, including: HTTP protocol analysis and packaging, WebSocket protocol analysis and encapsulation, SSL protocol processing, user authentication, heartbeat detection, user behavior log, SSH connection, etc.

- **ListenerThreadPool module:** This module maintains a thread pool with a configurable number of threads, which is used to maintain the SSH channel between the listener and the remote server, receive character information from the remote server, and write the information to the corresponding Channel and transmit it back to the front end.

4. System implementation

We developed a proxy server named WebSSH based on the Netty framework to implement the proposed architecture[9]. The function of the proxy server can be extended by registering the required Handler to WebSSH. Currently, WebSSH supports the following functions:

4.1. User authentication.

Realize the verification of the front-end operation and maintenance personnel's identity, including the verification of user names, passwords, IP addresses and other related information. Users who are verified correctly can use the WebSSH server normally.

When operation and maintenance users request static resources through the HTTPS interface, they need to provide their user name and password in the request. The server verifies the user name and password. When the user name and password are consistent with the database, the authentication is passed. After the authentication is successful, the server generates a TOKEN string and sends it as a response to the front end for the subsequent server to confirm its identity.

4.2. Log records throughout the process.

Record the user's identity information when logging in, including log-in time, IP address, user name, access duration, operation content and other related information.

Log recording is divided into four levels: DEBUG, INFO, WARN, and ERROR. The system can be configured to achieve different output methods of different levels of logs, including log file generation, database entry, discard, and control panel output.
4.3 Heartbeat detection.
Realize configurable traffic idle time detection. If there is no traffic on the channel within the set time, the connection will be closed forcibly.

The system starts heartbeat detection after the successful HTTPS handshake between the operation and maintenance user and the proxy server. When the channel between the user and the server has no traffic in any direction within a specified period of time, the channel is forcibly closed, and the corresponding SSH connection is closed at the same time.

4.4 Session management.
Manage the session life cycle of a SSH connection with a remote server, establish the same session with the same server, create different channels based on the session when multiple connections are made, and destroy the session when all channels are destroyed.

The proxy server attempts to establish an SSH connection with the corresponding remote server based on the configuration information carried in the first WebSocket request of the operation and maintenance user. After the connection is successful, a session is created to manage all connections with the target server. The session management function is based on the typical SSH session management method. During the process of establishing a connection with the proxy server, the remote server does not perceive the existence of operation and maintenance personnel, and its sessions are maintained by the proxy server.

In addition to the above functions, the proxy system also supports SSL processing, HTTP protocol encode and decode, static resource response, WS protocol encode and decode and other functions, the overall processing flow of the system for read and write events is shown in Figure 5:

![Figure 5. I/O event processing flow.](image)

The user first initiates an HTTPS request that is processed by the SSL processor and then registers for the heartbeat detection event, and then analyzes the HTTP protocol. After the analysis is completed, the user's identity is verified, and various static resources are encapsulated into HTTP responses, which are processed by the SSL processor and sent back to user.

After the user receives the static resource, the protocol is upgraded, and then the communication is based on the WSS protocol. The communication adopts the JSON format, including operation type, connection or command information, etc. When the first WSS protocol connection is made, connect information is sent, including information such as target ip address, port number, user name, and password. The proxy server sequentially attempts to establish an SSH connection with the target remote server. After the channel is established successfully, it starts to send response information to the user in real time. The command information in JSON format can be sent, and the implementation interface is shown in Figure 6:

![Figure 6. User interface.](image)
5. Summary
In this article, we propose an SSH proxy system called WebSSH, which uses the B/S architecture to use the browser as the client without a special client, which solves the cross-platform problem of the SSH client and is convenient for remote operation and maintenance personnel. The communication process of the system is based on the SSL protocol to ensure the security and reliability of the transmitted information. The application layer protocol is based on HTTP/WebSocket protocol to ensure the realization of various business functions and provide compression transmission functions to improve system performance. The proxy server was developed based on the Netty framework and the EventLoop threading model was adopted to improve the performance of the system. The system provides an extended interface, and similar processing can be done for character-type protocols, which has good scalability. In addition, the system provides a log record of the whole process, ensuring the function of auditing the behavior of operation and maintenance personnel.

References
[1] Zheng,L.(2013)The SSH protocol audit system based on proxy technology.In:International Conference on Computational and Information Sciences.Shiyan, Hubei, China.1489-1492.
[2] RFC 4252.(2006)The Secure Shell (SSH) Authentication Protocol.https://datatracker.ietf.org/doc/rfc4252.
[3] RFC 6101.(2011)The Secure Sockets Layer (SSL) Protocol Version 3.0.https://datatracker.ietf.org/doc/html/rfc6101.
[4] Yan,X.F.(2021)Information platform construction based on SSL VPN technology.Railway Signalling & Communication.2021(01):62-64.
[5] RFC 7118.(2014)The WebSocket Protocol as a Transport for the Session Initiation Protocol (SIP).https://datatracker.ietf.org/doc/html/rfc7118.
[6] Dai,R F.(2020)Construction of University Network Security System Model Based on SSL Virtual Technology.Computer and modernization.2020(08):122-126.
[7] Tong,H.(2020)A design model of WebSocket performance test tool.Electronic Technology & Software Engineering.2020(19):49-50.
[8] Netty Project.(2021)Netty Project.https://netty.io.
[9] Zhen,K,C.(2020)Internet of Things data access system based on Netty and Kafka.Computer Engineering and Applications.2020(05):135-140.