Proxy SOCKSv5 configuration protocol

November 2020

1. Status of Memo

This RFC describes the details of our Proxy SOCKSv5 server configuration protocol.

The reader is especially cautioned not to depend on the values which appear in examples to be current or complete, since their purpose is primarily pedagogical. Distribution of this memo is unlimited.

Table of Content

2. Introduction

2.1. Overview

This protocol was designed for the server administrator to communicate with our Proxy SOCKSv5 server. Our server is a Proxy SOCKSv5, with added capabilities, like DNS over HTTP queries and logging. This protocol allows the administrator to change the server settings, such as buffer size, without turning off the server, and to retrieve metrics such as historical and concurrent connections.

This protocol is an application connection-oriented protocol over STCP. It also counts with authentication. The client must provide a user and password to be able to start a connection with the server.

2.1.1. Methods

The following methods are the ones currently supported by this protocol version:

1. Get the number of historical connections
2. Get the number of concurrent connections
3. Get the number of bytes transferred
4. Set a new buffer size

3. Authentication

After the SCTP connection is established, the user must authenticate itself. To do so, it must send a package with the following format:

3.1. Package to send format

VER	ULEN	UNAME	PLEN	PASSWD
1	1	1 to 255	1	1 to 255

VER	Protocol version. 1 byte long. Current supported version: 1.
ULEN	Length of the username. 1 byte long.
UNAME	Username. Max. 255 bytes.
PLEN	Length of the password. 1 byte long.
PASSWD	Password. Max. 255 bytes.

After receiving the authentication package, the server will process the request and reply with a package with the following format:

3.2. Package to receive format

VER	CODE
1	1

VER	Protocol version. 1 byte long. Current supported version: 1.
CODE	Status of the authentication. 1 byte long. 0 Authentication successful 1 Invalid username or password 2 Invalid protocol 3 Internal server error

4. Sending and receiving requests

After the user is validated and the connection is established, the user can send its request. Keep in mind that the server will close the connection after receiving a request and sending its response. To send a request, the user must send a package with the following format:

4.1. Package to send format

CODE	NPARAM	LPARAM1	PARAM1	LPARAM2	PARAM2
1	1	1	1 to 255	1	1 to 255

CODE	Method code. 1 byte long.	
	0	Get the number of historical connections.
	1	Get the number of concurrent connections.
	2	Get the number of bytes transferred.
	3	Set a new buffer size.
	4-253	Reserved for future methods.
	254-255	Reserved for reply errors.
NPARAM	Amount of parameters sent. 1 byte long.	
LPARAM1	Parameter length in bytes. 1 byte long.	
PARAM1	First parameter. Max. 255 bytes.	
LPARAM2	Parameter length in bytes. 1 byte long.	
PARAM2	Second parameter. Max, 255 bytes.	

For methods 0, 1 and 2, no parameters are needed, so NPARAM must be 0 and LPARAM1, PARAM1, LPARAM2, PARAM2 are ignored. For method 3, the new buffer size must be sent in ASCII format. LPARAM2 and PARAM2 were included for future versions, where new methods which would require 2 params, such as adding users.

When setting a new buffer size using method 3, the server’s response data will be the new buffer size if the operation was successful, in ASCII format.

4.2. Package to receive format

CODE	LEN	DATA
1	1	1 to 255

CODE	Method code received	Request successfully resolved
	254	Invalid param
	255	Invalid method

LEN	Length of the response. 1 byte long.
DATA	Body of the response. Response is in ASCII format.

5. Bibliography

All of the information presented above was written by Griggio, Juan Gabriel; Méndez, Ignacio Alberto; Villanueva, Ignacio & Zuberbuhler, Ximena.