

[Get unlimited access](#)[Open in app](#)

Nimit Aggarwal

[Follow](#)

Jul 22, 2019 · 4 min read



## SOCKS 5 — A Proxy Protocol

Before reading this article it is highly recommended to be familiar with the basic proxy concepts. To brush up on the proxy concepts and to know how to make a Simple HTTP/HTTPS proxy in node, read [this](#).

---

*SOCKS 5 basically is a **Framework for a General Proxy Protocol**.*

---

What I mean by a general proxy protocol is that for each application layer protocol like HTTP, HTTPS(controversial), FTP and many more, you would need to set up a different proxy server. For example, for HTTP you would have something like squid, for HTTPS Browsermob proxy and so on so forth. Now a SOCKS 5 proxy provides a general framework for these protocols to transparently and securely traverse the data.

Another use of *SOCKS* is as a **circumvention tool**, allowing traffic to bypass Internet filtering to access content otherwise blocked, e.g., by governments, workplaces, schools, and country-specific web service

SOCKS 5 additionally provides **authentication** so only authorized users may access a server.

### Real-World Application

1. Mostly all torrent clients have an option to configure a SOCKS 5 proxy to bypass the firewall if any.
2. It is faster than VPN, so in many places where the speed matter it is used over VPN.
3. The [Tor](#) onion proxy software presents a SOCKS interface to its clients.

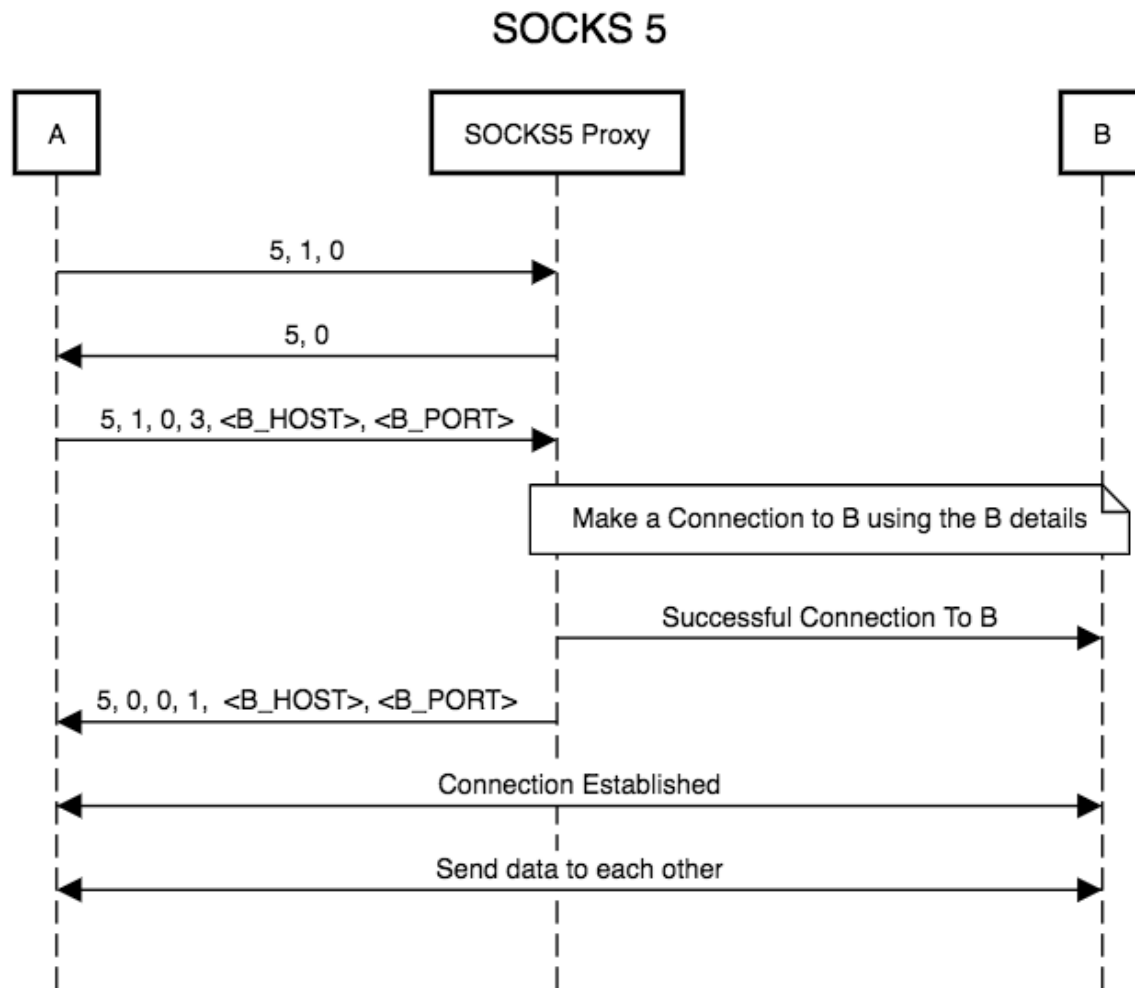
### Basic Working

A wishes to communicate to B over the internet, but a firewall exists between those two, so A



SOCKS 5 protocol, see below section.

The SOCKS5 protocol is defined in [RFC 1928](#). Here is a brief introduction of the protocol in the diagram. For detailed protocol features please refer to [RFC 1928](#)



SOCKS 5 HANDSHAKE[NOTE: 5, 1, 0 are actual bytes (0x05, 0x01, 0x00)]

## Handshake

A(Client) sends the initiation packet(0x05, 0x01, 0x00) to the SOCKS5 proxy. The breakdown of the first packet is :-

1. The **First Byte 0x05** is for the version of the SOCKS, in this case, it is SOCKS 5. This remains common for all the SOCKS 5 packets. Won't mention this byte in next sections.
2. The **Second Byte 0x01** is for authentication purposes. Precisely, the number of



3. The **Third Byte 0x00** is for the authentication methods. There are authentication methods, variable length, 1 byte per method supported. The authentication methods supported are numbered as follows:

*0x00: No authentication*

*0x01: GSSAPI*

*0x02: Username/password*

*0x03–0x7F: methods assigned by IANA*

*0x80–0xFE: methods reserved for private use*

SOCKS Proxy sends back to A(**0x05, 0x00**). :-

1. The **Second Byte 0x00** is the chosen authentication method by the proxy. In this case, it was the only method provided by the client hence this. The subsequent authentication is method-dependent. Username and password authentication(method 0x02) is described in [RFC 1929](#). Here no authentication dependent steps need to take place further. If the negotiated method includes encapsulation for purposes of integrity checking and/or confidentiality, further requests **MUST** be encapsulated in the method-dependent encapsulation.

A(Client) sends the request packet (**0x05, 0x01, 0x00, 0x03, <B\_HOST>, <B\_PORT>**). Once the method-dependent subnegotiation has completed, the client sends the request details to SOCKS proxy:-

1. The **Second Byte 0x01** is for the command code. It is one byte

*0x01: establish a TCP/IP stream connection*

*0x02: establish a TCP/IP port binding*

*0x03: associate a UDP port*

2. The **Third Byte 0x00** is a reserved byte. It **must** be 0x00 and 1 byte.

3. The **Fourth Byte 0x03** is the address type of desired HOST and 1 byte. The options are as

[Get unlimited access](#)[Open in app](#)

0x03: Domain name, 1 byte for name length, followed by host name

0x04: IPv6 address, followed by 16 bytes IP

4. The last Byte is port number in a network byte order, 2 bytes

SOCKS proxy sends back the request packet (0x05, 0x00, 0x00, 0x01, <B\_HOST>, <B\_PORT>). This is for the status of the request by the client to the proxy:-

1. The **Second Byte 0x00** is the status field. It is one byte. Meaning the request was granted.
2. The **Third Byte 0x00** is a reserved byte. It **must** be 0x00 and 1 byte.
3. The **Fourth Byte 0x01** is the address type of desired HOST and 1 byte. In case of CONNECT, this is followed by the binded IP address for the desired HOST, to give the client the detail of the DNS resolution.
4. The last Byte is port number in a network byte order, 2 bytes

After this, the connection takes place all the coming data from client A is transferred to client B and vice versa. This way the SOCKS proxy works as a general framework proxy and handle most PROTOCOL with its security features.

There is a great node library which can handle all the details for a socks server. It transforms the incoming SOCKS5 stream into its respective protocol stream.

<https://www.npmjs.com/package/socks5-stream>



--





Get unlimited access

Open in app

Love podcasts or audiobooks? Learn on the go with our new app.

Try Knowable

## Recommended from Medium

 CHAINsofBLOCKS

**Decentraland L1 Wearables | Voting Power**



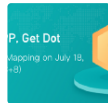
 Omayr Zanata in The Startup


**Abusing Microsoft Teams rate limiting for DDoS**



 ZB.com

**ZB supports mapping Dot mainnet tokens on July 18 and launches special purchase!**



 Sabbir Mollah in NSUACMSC

**Hack into a program**



 Ax Sharma in The Startup

**What is cybersquatting? And why is it a BIG deal?**



 Scott J Roberts

**My Favorite Open Source Security Tools**

 Samantha

**TryHackMe: Advent of Cyber [Day 4] Training**



 Vishal Gupta

**What is Amazon Cognito?**



[About](#) [Help](#) [Terms](#) [Privacy](#)

Get the Medium app





Get unlimited access

Open in app