# SOCKS5 protocol (rfc1928)

| App |
|---|
| SOCKS5 |
| TCP/UDP |
| IP |
| … |

(Originally designed for firewall traversal, i.e. SOCKS server negotiating a service (bnd.addr, bnd.port) behind firewall for connection to (dst.addr.dst.port)

(as such, layer 3 ICMP ping would not go through a SOCKS5 proxy)

**TCP client** — firewall — **SOCKS server (def: 1080)** — **(bnd.addr, bnd.port)**

TCP handshake →

auth request (plain)

| 0x05 (ver) | #methods | method_1… |
|---|---|---|
| 1B | 1B | #methods B |

auth select →

auth response (plain)

| 0x05 (ver) | selected_method |
|---|---|
| 1B | 1B |

← auth select resp

← method-dependent auth negotiation →

connect request (optionally encrypted)

| 0x05 (ver) | cmd | reserved | addr type | dst.addr | dst.port |
|---|---|---|---|---|---|
| 1B | 1B | 1B | 1B | var | 2B |

cmd: connect/bind/udp associate
addr type: ipv4/ipv6/domain name
dst.addr: when domain name, first byte is length

connect request →

← connect response

connect response (optionally encrypted)

| 0x05 (ver) | rep | reserved | addr type | bnd.addr | bnd.port |
|---|---|---|---|---|---|
| 1B | 1B | 1B | 1B | var | 2B |

rep: succeed/general failure/network unreachable/ttl expired/connection not allowed by ruleset
addr type: ipv4/ipv6/domain name

In response to
- "connect": (bnd.addr, bnd.port) the server connection client can use to talk to (dst.addr, dst.port) (server might be multi homed)
- "bind": when client need to accept connection from the server (e.g. FTP) (comes after connect, two responses)
- "UDP associate": (bnd.addr, bnd.port) means client's UDP requests gets relayed through this pair

← TCP: (dst.addr, dst.port) req/res (encrypted) →

UDP relay (encrypted)

| reserved | fragment | addr type | dst.addr | dst.port | data |
|---|---|---|---|---|---|
| 2B | 1B | 1B | var | 2B | var |

← UDP: (dst.addr, dst.port) request (encrypted) →

UDP relay to (bnd.addr, bnd.port) is encapsulated in this header since UDP dst.addr, dst.port may not be known at the time of "udp associate" request in the previous step.

## shadowsocks

https://shadowsocks.org/en/wiki/Protocol.html
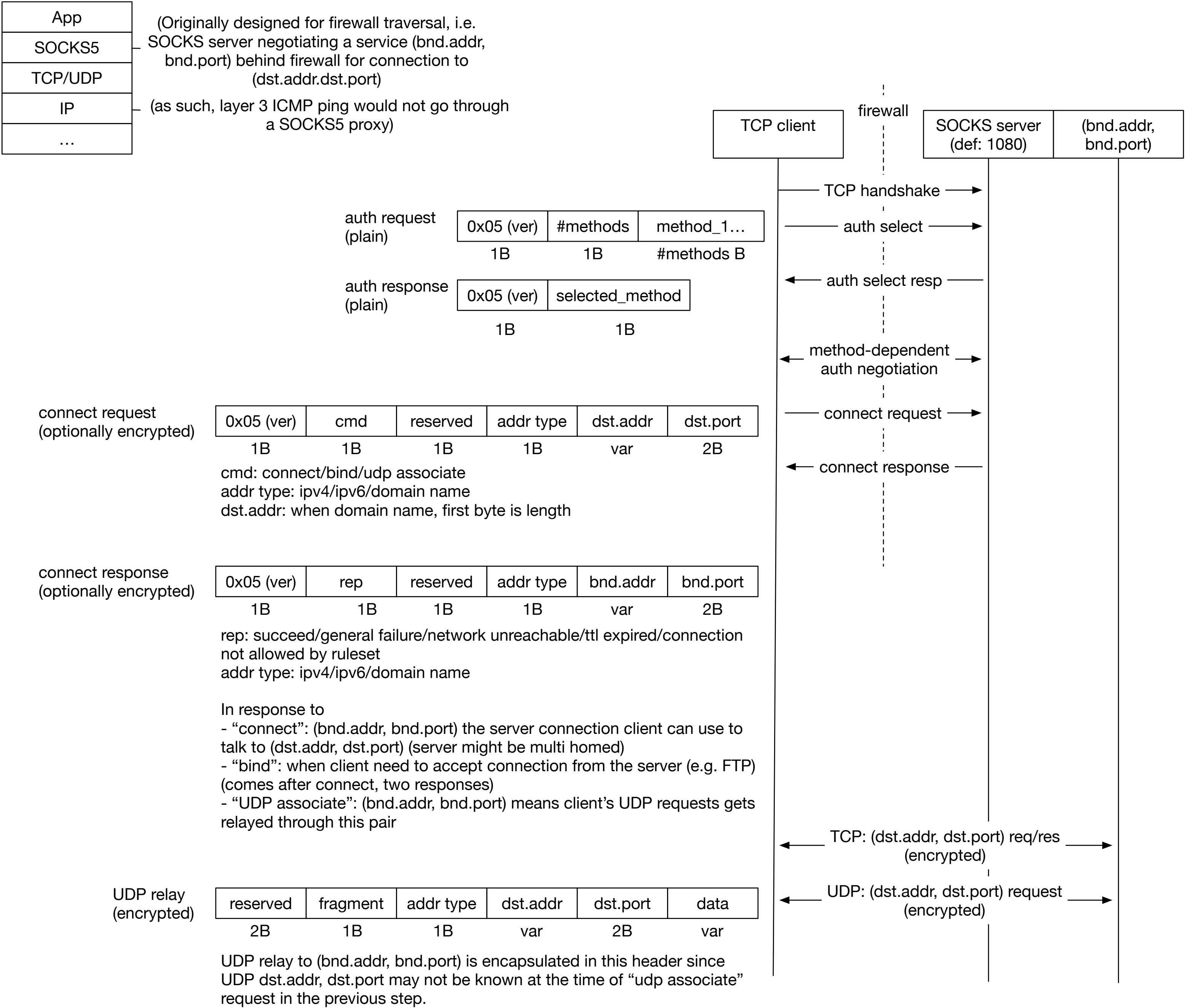
Note that SOCKS5 server is running locally on the client machine in shadowsocks protocol: no plain text auth request / response should leave the client machine. All the exchange between client / server machine should just be TCP packets with encrypted payload.

The likes of shadowsocks-NG would include a SOCKS server and an http proxy?