The SSH Protocol

Network Security

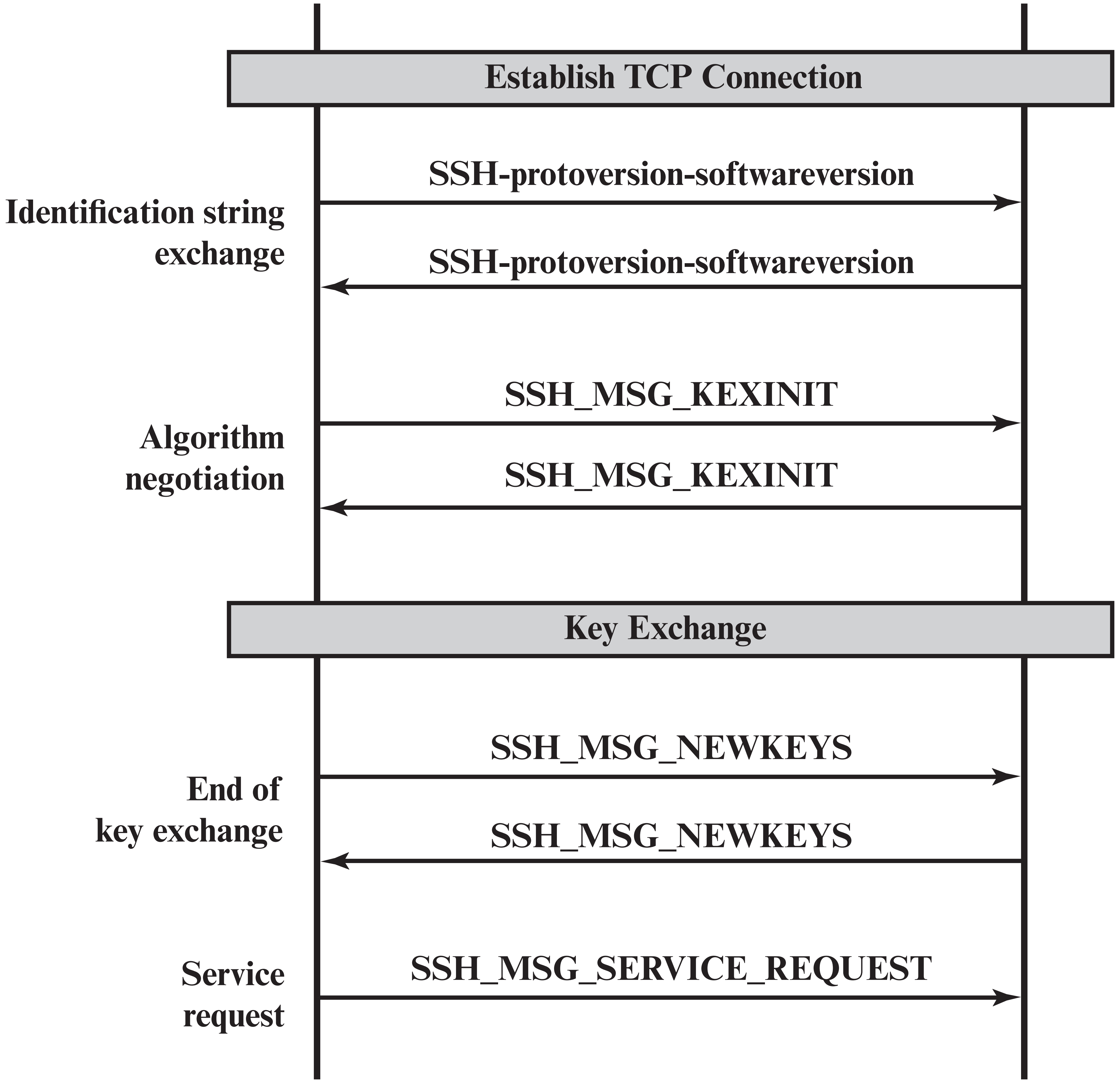
SSH Overview

- SSH = Secure Shell, mostly used for securely log onto remote servers (+ tunnelling, file transfer,…)
- Similar to TLS, it builds on top of the transport layer
- SSH achieves **authentication**, **data confidentiality**, **data integrity**

Server Authentication

- Again, similar to TLS, server authentication is done through public key cryptography.
- A server may own **host keys** for different asymmetric encryption schemes. A server host key is used during the key exchange to authenticate the identity of the host.
- There are two possibilities to establish trust in a host key:
    - The client has a local database that maps host names to public keys. (`~/.ssh/known_hosts` file; most commonly used method)
    - The mapping is certified by a trusted CA.

The SSH Protocol



Source: Cryptography and Network Security

- Again, we find some similarities with TLS.
- SSH has a phase for algorithm negotiation before the key exchange:
    - The first algorithm on the client’s list that is also on the server’s list is chosen.

The SSH Key Exchange

C: client, S: server, V_C: client identity string, V_S: server identity string

p: prime, g: generator for subgroup of $\mathbb{F}_p$, q: order of the subgroup

K_S: server's public host key, I_C, I_S: client's/server's KEX_INIT message

<u>Client</u>

$e = g^x \mod p$ for random $1 < x < q$

$\xrightarrow{e}$

<u>Server</u>

$f = g^y \mod p$ for random $0 < y < q$

$K = e^y \mod p$

$H =$ hash$(V_C \,||\, V_S \,||\, I_C \,||\, I_S \,||\, K_S \,||\, e \,||\, f \,||\, K)$

$\xleftarrow{K_S, f, Sign_S(H)}$

Verify K_S is correct host key,
compute $K = f^x \mod p$,
calculate H and verify signature

The SSH Key Exchange

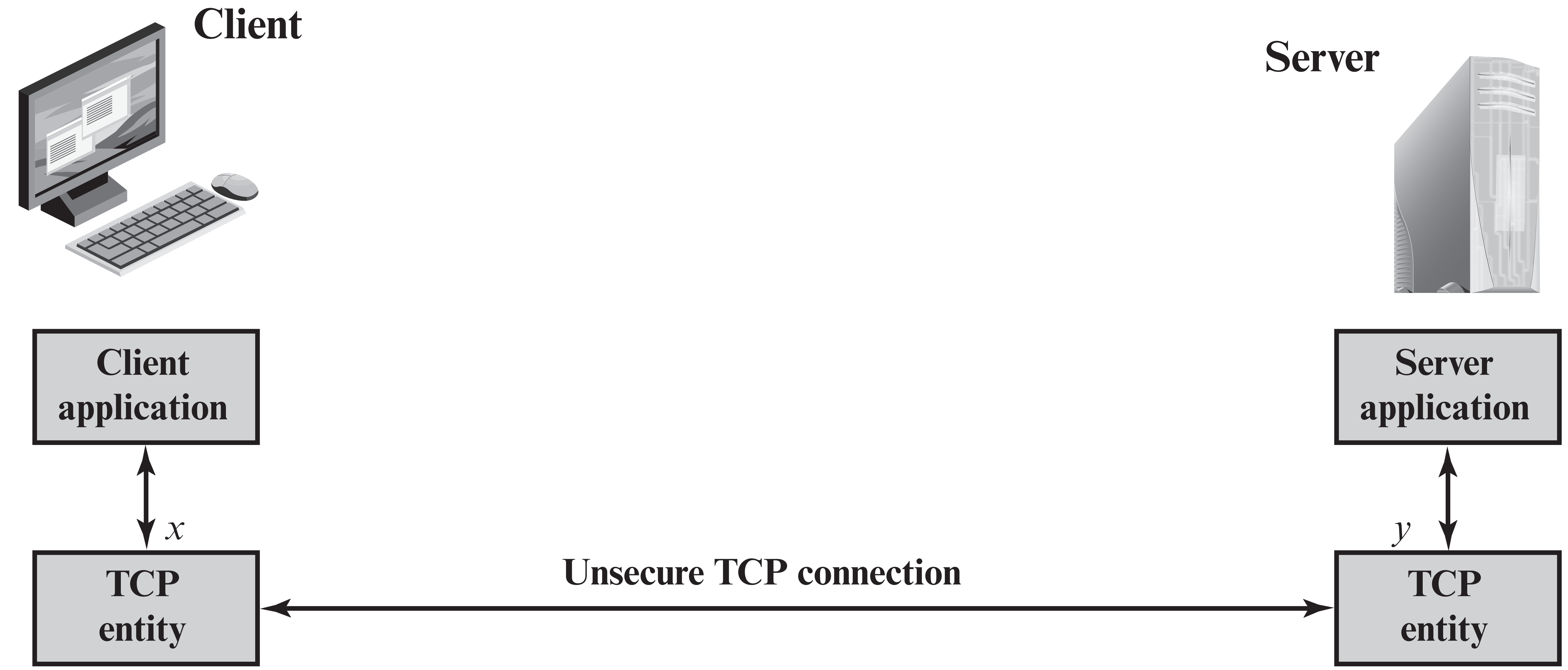
- The key exchange is: Diffie Hellman!
- All subsequent keys are generated from the master key K, e.g.,
    - Initial IV client to server: $\mathrm{hash}(K \mid\mid H \mid\mid \text{“A”} \mid\mid \mathrm{session\_id})$, where session_id is H most of the times
    - Encryption key server to client: $\mathrm{hash}(K \mid\mid H \mid\mid \text{“D”} \mid\mid \mathrm{session\_id})$, where session_id is H most of the times

User Authentication

- The server authentication happens during the key exchange.
  The user authentication is handled separately and several methods are possible:
  - **Public key:** The client sends a message to the server containing its public key and a signed message. The server checks whether the key is acceptable (`~/.ssh/authorized_keys` file) and whether the signature is valid.
  - **Password:** The password is sent in **plaintext** over the encrypted channel.
  - **Host based:** The host machine does the authentication. The client sends a message signed by the host private key. The server trusts the host.

SSH Port Forwarding

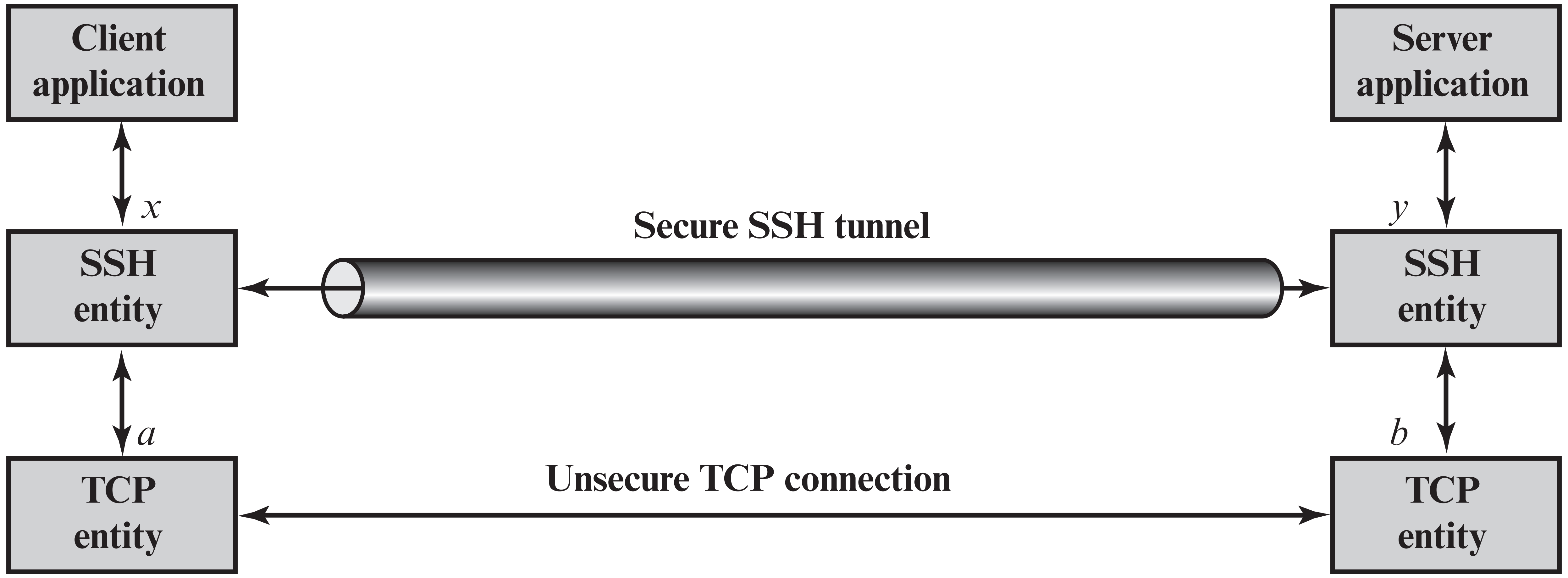
- Port Forwarding is also called Tunneling.
- A port is a number associated with a transport protocol like TCP.
  It identifies where packets should be handled.
  An application may listen on a port (e.g., HTTPS often listens on port 443) and this is where data is sent to.



Source: Cryptography and Network Security

SSH Port Forwarding

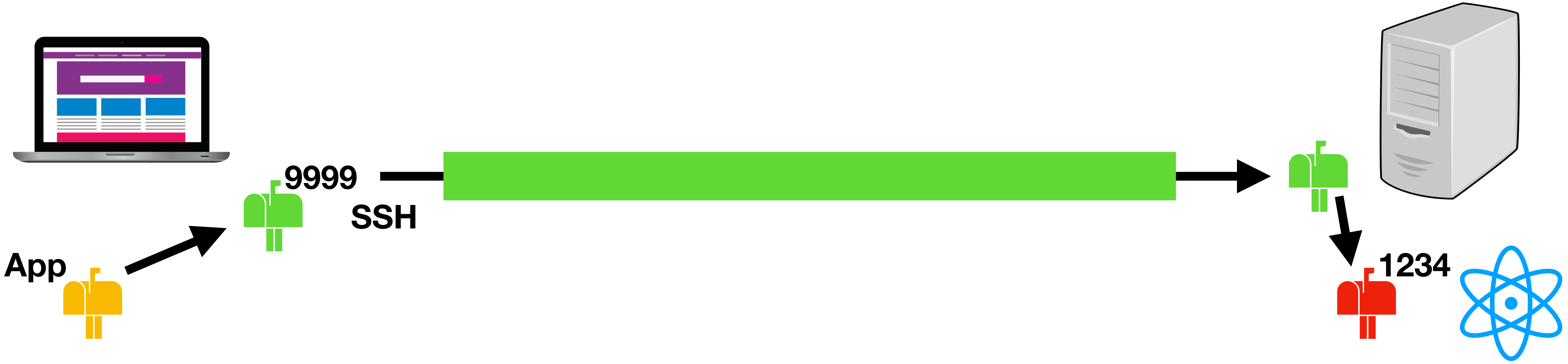
- Port Forwarding is also called Tunneling.
- A port is a number associated with a transport protocol like TCP.
  It identifies where packets should be handled.
  An application may listen on a port (e.g., HTTPS often listens on port 443) and this is where data is sent to.



Source: Cryptography and Network Security

SSH Port Forwarding Example

- Server runs an application listening on port 1234.
- Client sets up a SSH connection, forwarding local traffic to 9999 to the server on port 1234.
- Client connects to the local port 9999 and is able to talk to the server application.
- Essentially, SSH takes any traffic received on 9999, encrypts it, sends it to the server, decrypts it, and forwards it to port 1234.



SSH Potential Issues

- **Machine-in-the-Middle** attacks:
  - Upon first connection, a user is presented with the **host key** and is required to ensure its validity
  - If not properly checked, a MITM attack is possible and an attacker could retrieve the plaintext password
- **Brute force** attacks:
  - If a server is not configured to take appropriate actions, brute forcing passwords is an option