UNIVERSITAT POLITÈCNICA DE CATALUNYA BARCELONATECH

SSH – Secure SHell

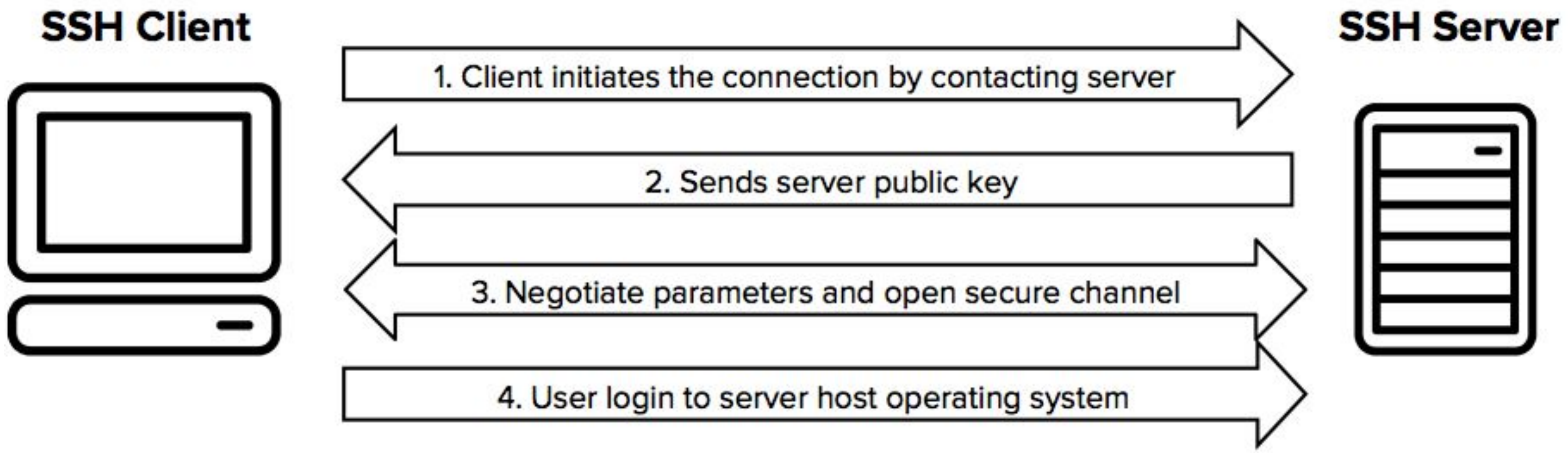
Why SSH?

- SSH is a de facto standard for remote logins and encrypted file transfers.
    - Founded in 1995 by Tatu Ylonen (Finland)
    - Earlier apps such as Telnet and FTP lack confidentiality and integrity (username and passwords are sent in cleartext).
- SSH provides encryption and authentication:
    - Host-based and (or) client-based authentication
    - Data integrity using MACs and hashes
- Runs at the application layer on port 22 (over TCP).
- It can also be used for port forwarding of arbitrary TCP/IP or X11 connections (interactive terminals)

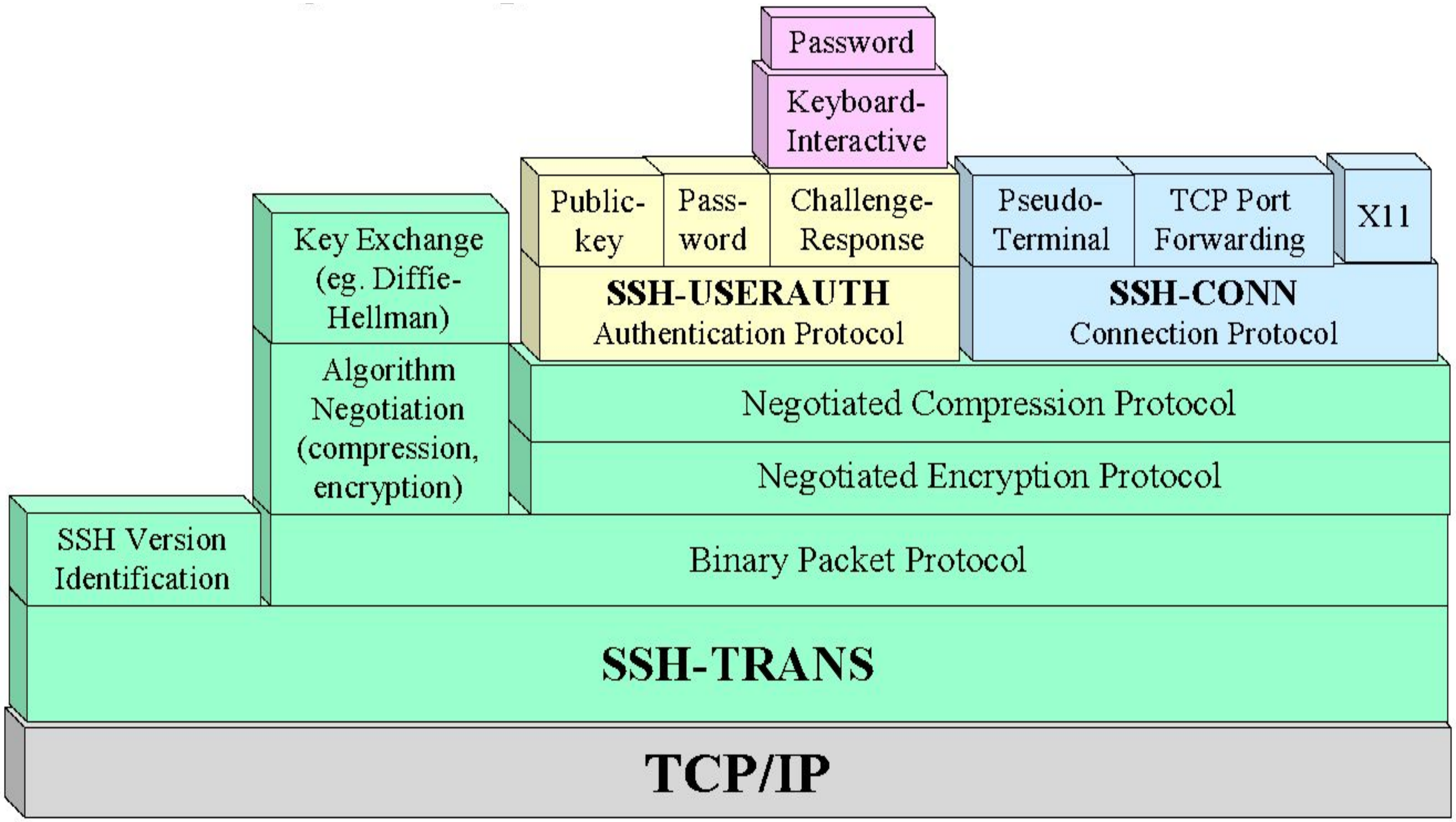
SSH tools

- There are several commercial and freeware SSH tools for UNIX and windows platforms
- Windows –
  - PuTTY
  - TeraTerm
  - SecureCRT
- UNIX –
  - The RedHat Linux package and OpenBSD come alongwith tools such as ssh, sshd(daemon), scp(secure copy), sftp(secure ftp), etc.
  - OpenSSH

How does SSH work?


SSH Client
SSH Server
1. Client initiates the connection by contacting server
2. Sends server public key
3. Negotiate parameters and open secure channel
4. User login to server host operating system

The SSH protocol



The SSH protocols

- SSH mainly consists of 3 protocols:
  - **SSH Transport layer protocol (SSH-TRANS)**
    - Provides the initial connection, packet protocol, server authentication and basic encryption and integrity services.
  - **SSH Authentication protocol (SSH-AUTH)**
    - Client authentication
  - **SSH Connection protocol (SSH-CONN)**:
    - It permits a number of different services to exchange data through the secure channel provided by SSH-TRANS.
      - Pseudo-terminal, X11, Port Forwarding

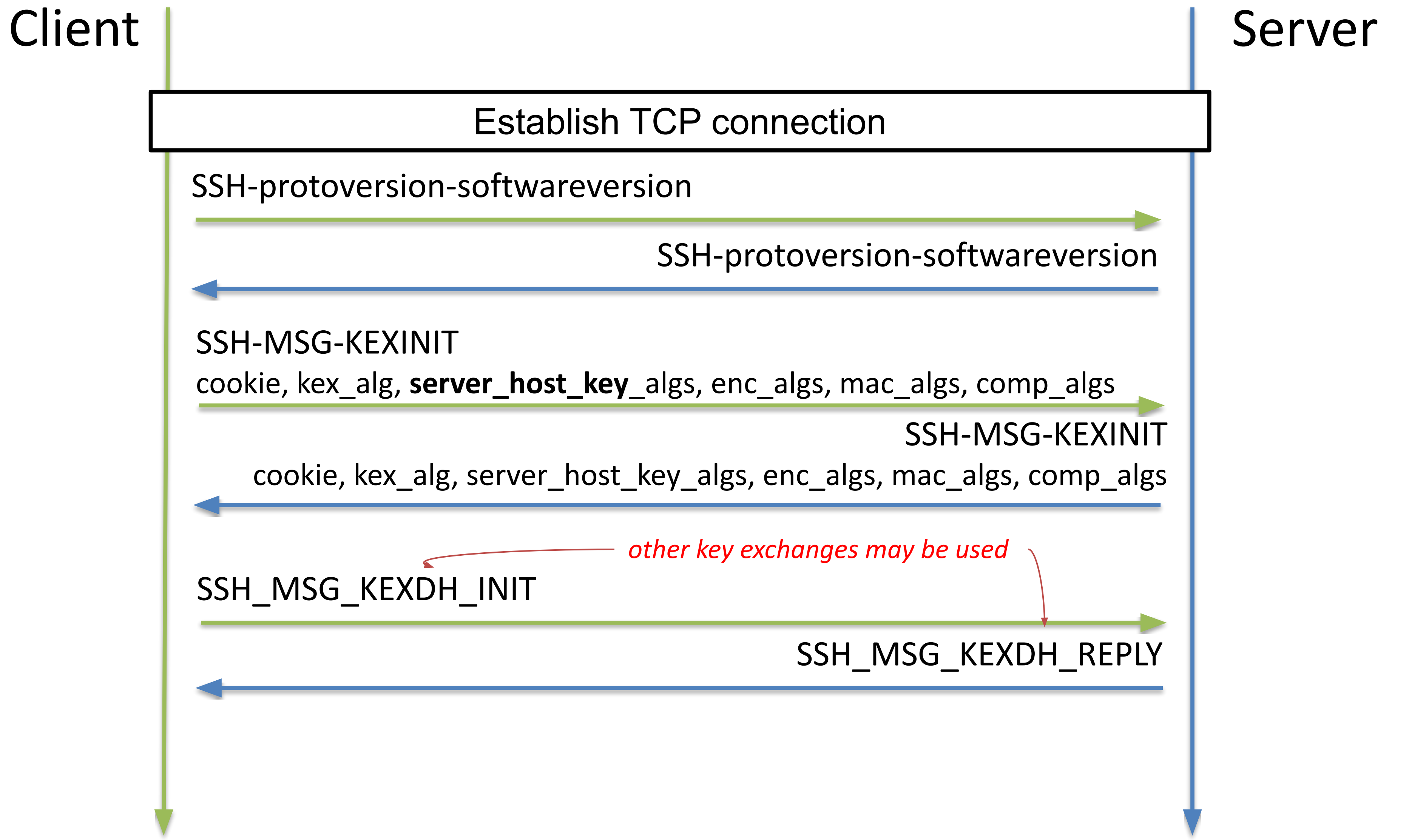
Host keys

- Host keys are key pairs (RSA, DSA, ECDSA) used to authenticate computers in SSH
  - Server authentication
    - Public host keys are stored in SSH clients, private keys are stored in SSH servers.
    - The server sends the fingerprint of its public key to authenticate
    - The first time we connect to a server we might see something like this:

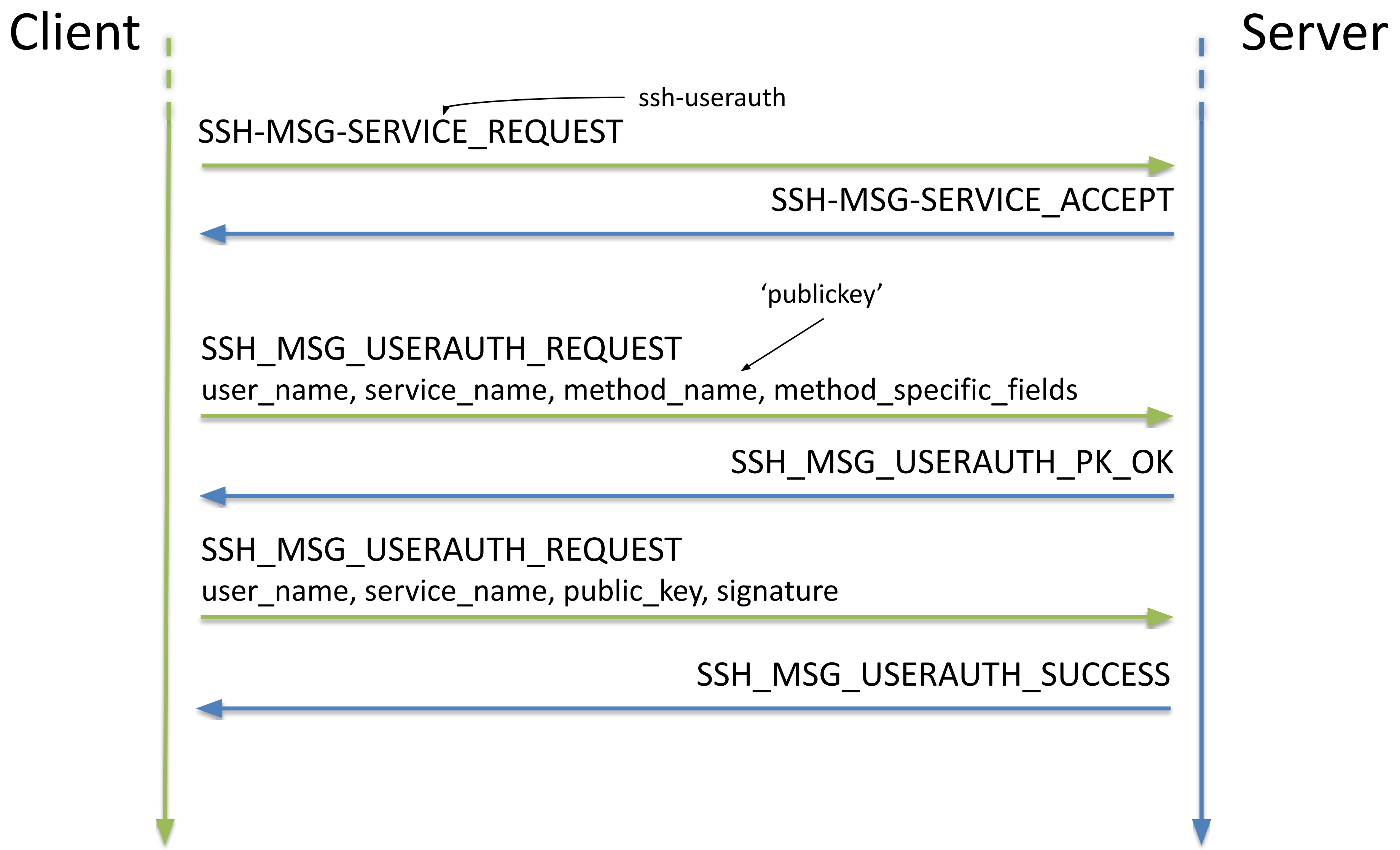
```
The authenticity of host '203.0.113.1 (203.0.113.1)' can't be established.
ECDSA key fingerprint is fd:fd:d4:f9:77:fe:73:84:e1:55:00:ad:d6:6d:22:fe.
Are you sure you want to continue connecting (yes/no)? yes
```

  - Client authentication
    - Can also use password authentication, host-based and certificate-based.

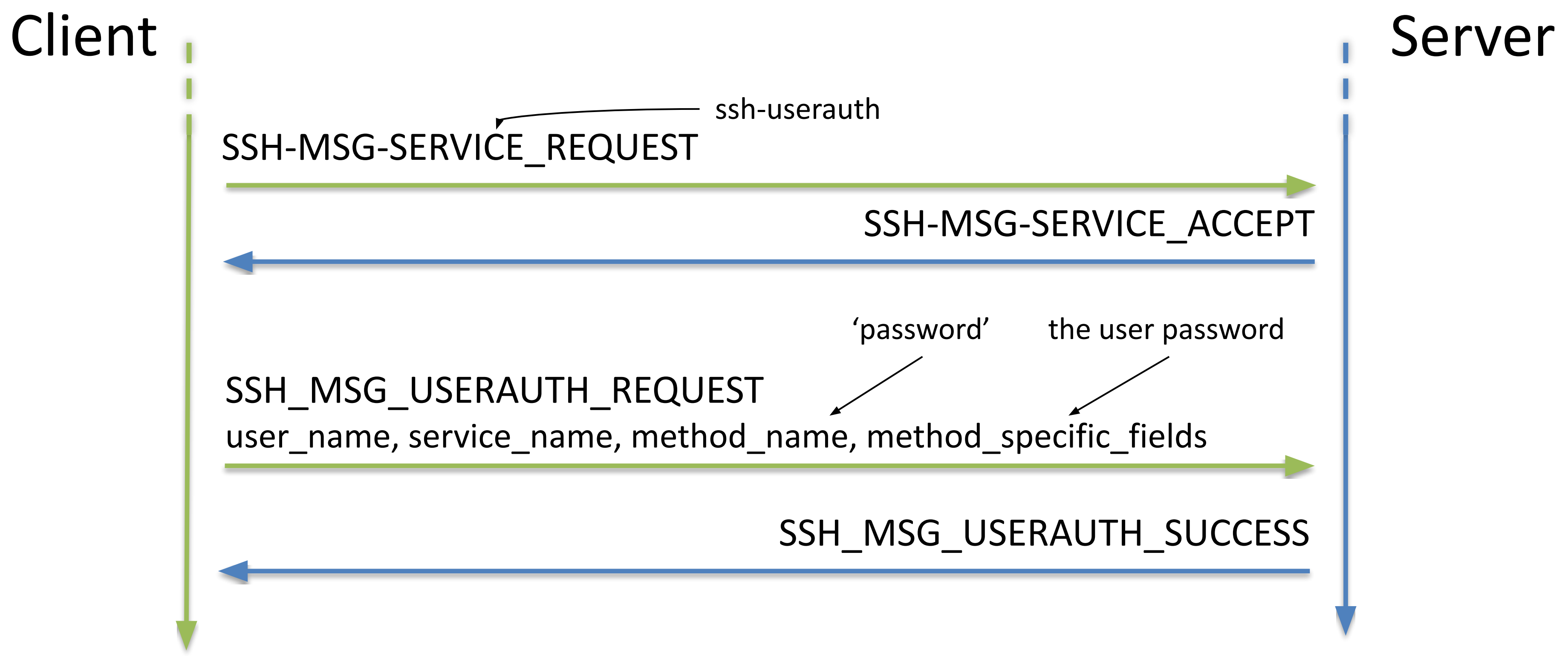
SSH protocol: SSH-TRANS (simplified)


Client
Server
Establish TCP connection
SSH-protoversion-softwareversion
SSH-protoversion-softwareversion
SSH-MSG-KEXINIT
cookie, kex_alg, **server_host_key**_algs, enc_algs, mac_algs, comp_algs
SSH-MSG-KEXINIT
cookie, kex_alg, server_host_key_algs, enc_algs, mac_algs, comp_algs
other key exchanges may be used
SSH_MSG_KEXDH_INIT
SSH_MSG_KEXDH_REPLY

SSH protocol: SSH-AUTH: public key

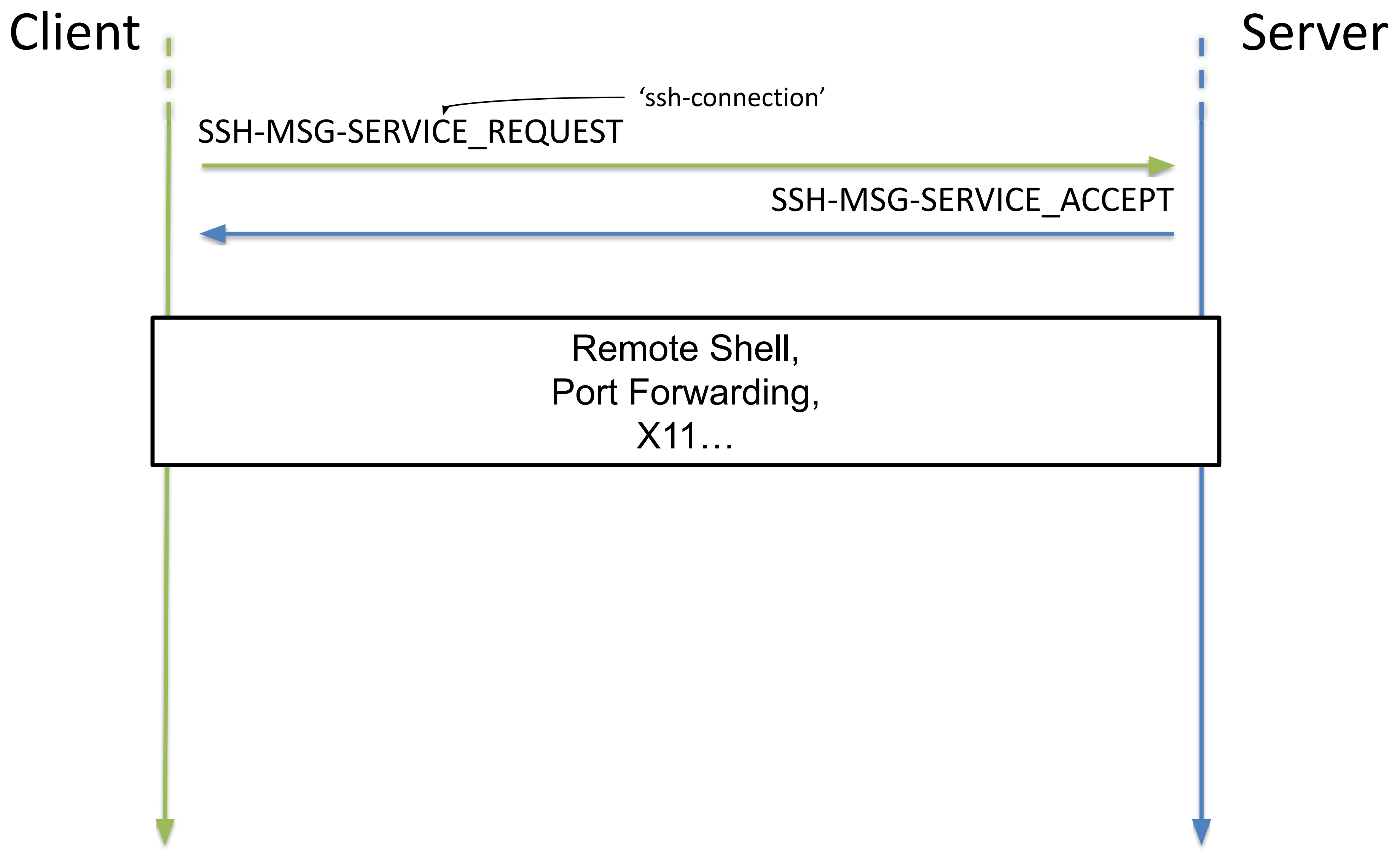


SSH protocol: SSH-AUTH: password



Besides publickey and password, SSH method_name can also be: 'host-based' and 'none'

SSH protocol: SSH-CONN



SSH Forwarding

- SSH forwarding, also called SSH tunnelling, allows to tunnel application ports from a ssh client to a ssh server, or viceversa
- 3 types:
  - Local Forwarding
  - Remote Forwarding
  - Dynamic Forwarding

Local port forwarding

- Initiated by a SSH client.
- Connections from the SSH client are forwarded via the SSH server, then to a destination server

$$\texttt{ssh -L }\underbrace{\texttt{local\_port:}}_{\substack{\text{local tcp port}\\ \text{on SSH client}}}\underbrace{\texttt{dst\_server:dst\_port}}_{\substack{\text{hostname or IP address and port of}\\ \text{the destination server, as seen from}\\ \text{the remote SSH server.}\\ \text{Notice that it could be a private IP}\\ \text{that is available from the SSH server}}}\ \underbrace{\texttt{ssh\_server}}_{\substack{\text{the hostname or IP}\\ \text{address of the remote}\\ \text{SSH server}}}$$

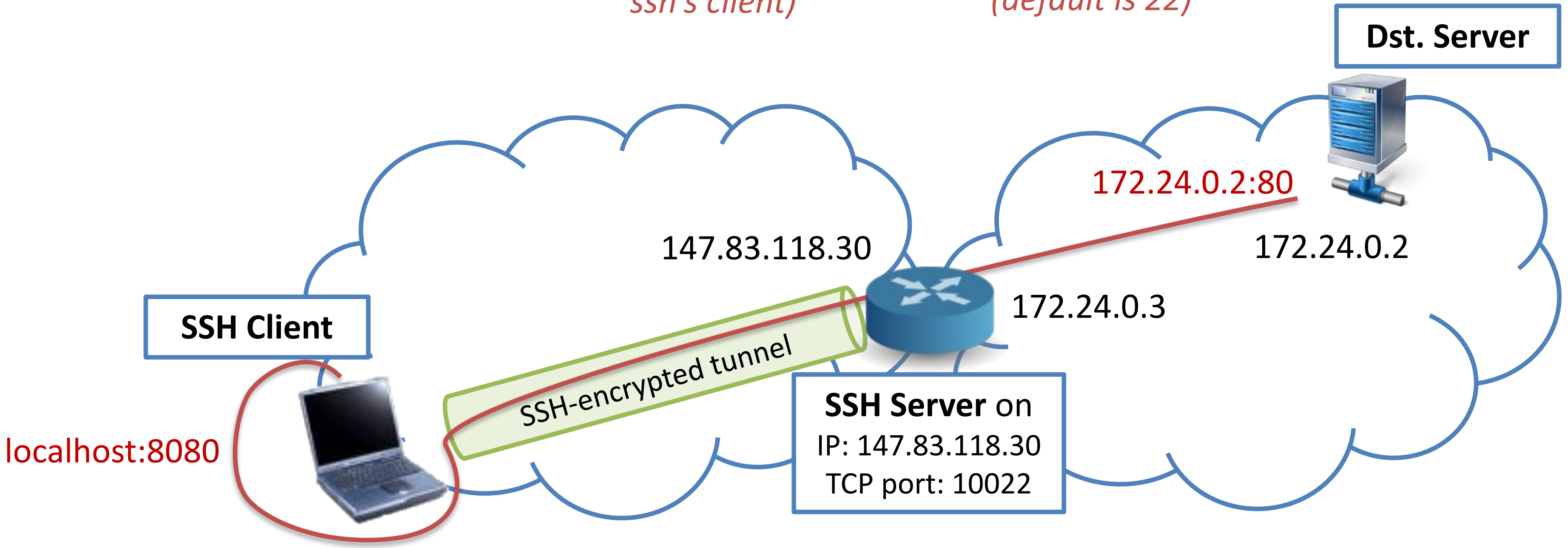
Local port forwarding: Example

- Access intranet's http server on remote tcp port

```
[user@sshclient ~]$ ssh -L 8080:172.24.0.2:80 -p 10022 user@147.83.118.30
```

Local port (on ssh's client) — 8080

ssh port (default is 22) — -p 10022



Remote port forwarding

- Initiated by a SSH client.
- Connections from the SSH server are forwarded via the SSH client, then to a destination server

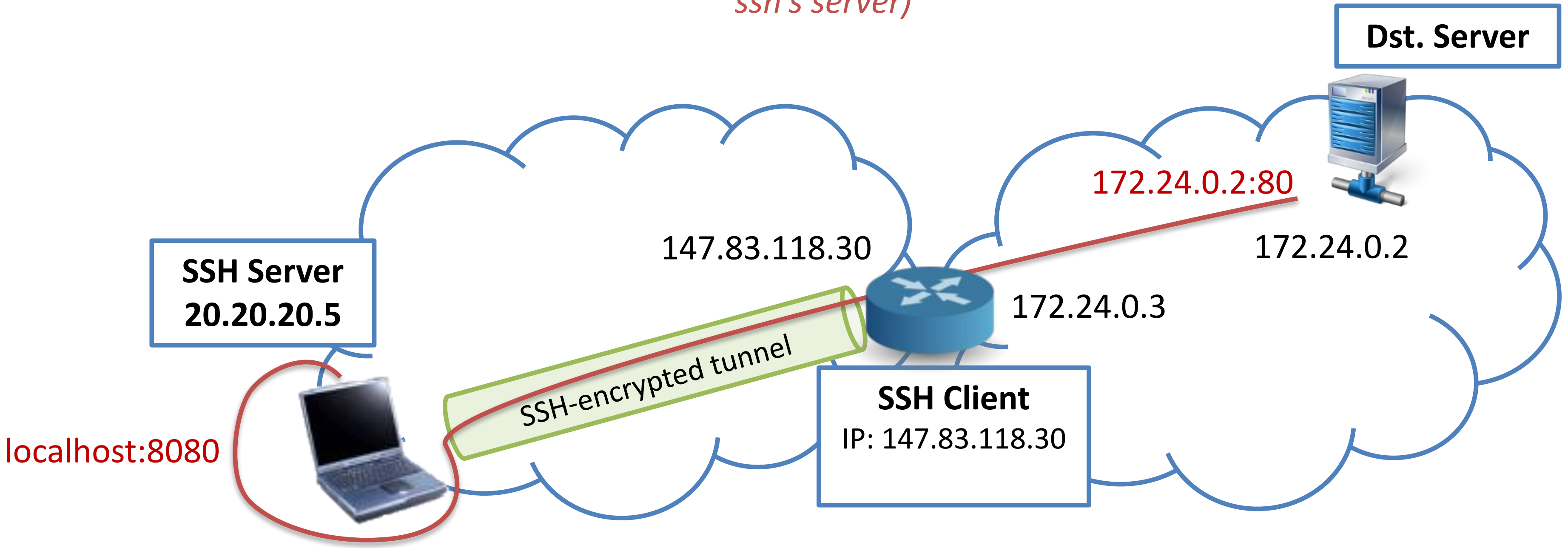
$$\texttt{ssh -R }\underbrace{\texttt{remote\_port:}}_{\substack{\text{tcp port on}\\ \text{remote SSH}\\ \text{server}}}\underbrace{\texttt{dst\_server:dst\_port}}_{\substack{\text{hostname or IP address and port of}\\ \text{the destination server, as seen from}\\ \text{the SSH client.}\\ \text{Notice that it could be a private IP}\\ \text{that is available from the SSH client}}}\ \underbrace{\texttt{ssh\_server}}_{\substack{\text{the hostname or IP}\\ \text{address of the remote}\\ \text{SSH server}}}$$

Remote port forwarding: Example

- Access intranet's http server on local tcp port

```
[user@sshclient ~]$ ssh -R 8080:172.24.0.2:80 user@20.20.20.5
```

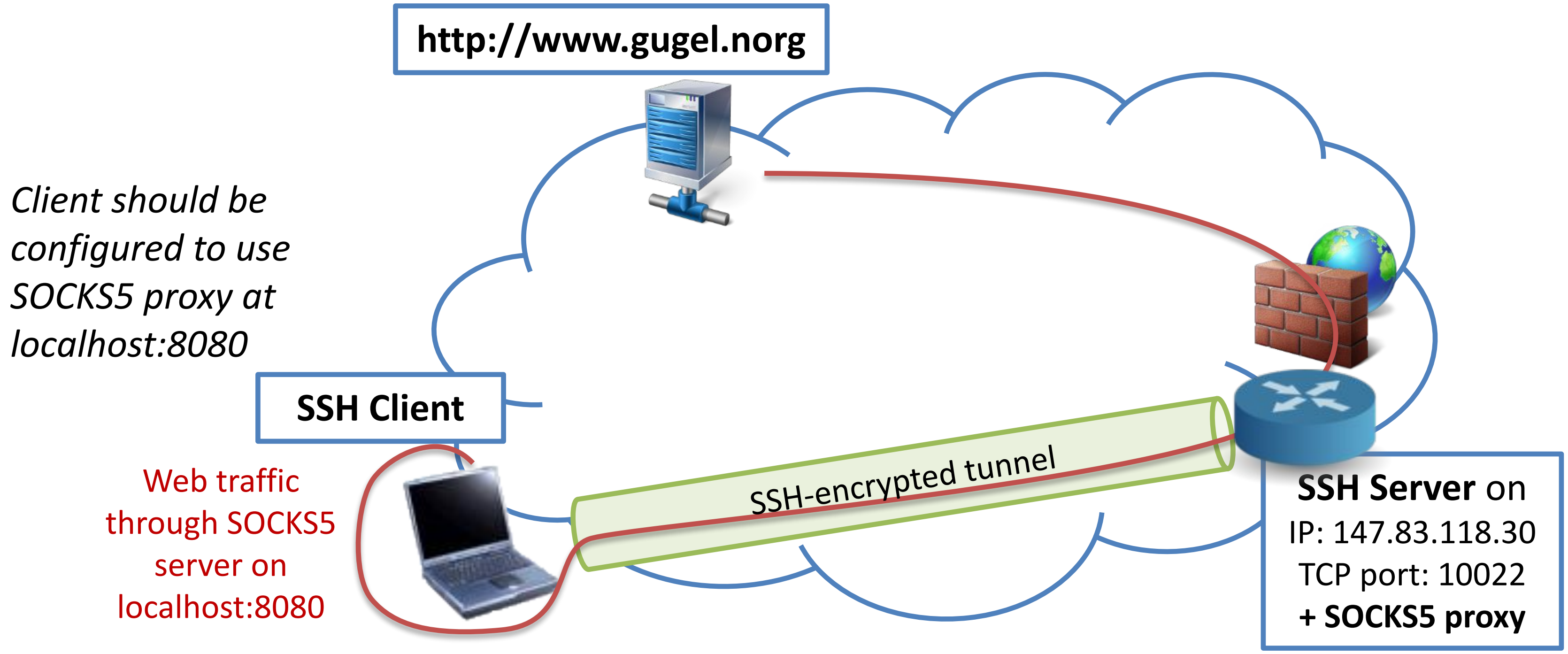
Remote port (on ssh's server)



Dynamic port forwarding

- A local forward to a SOCKS5 proxy on the SSH server, then access to the net from that server

```
[user@sshclient ~]$ ssh -D 8080 -p 10022 user@147.83.118.30
```

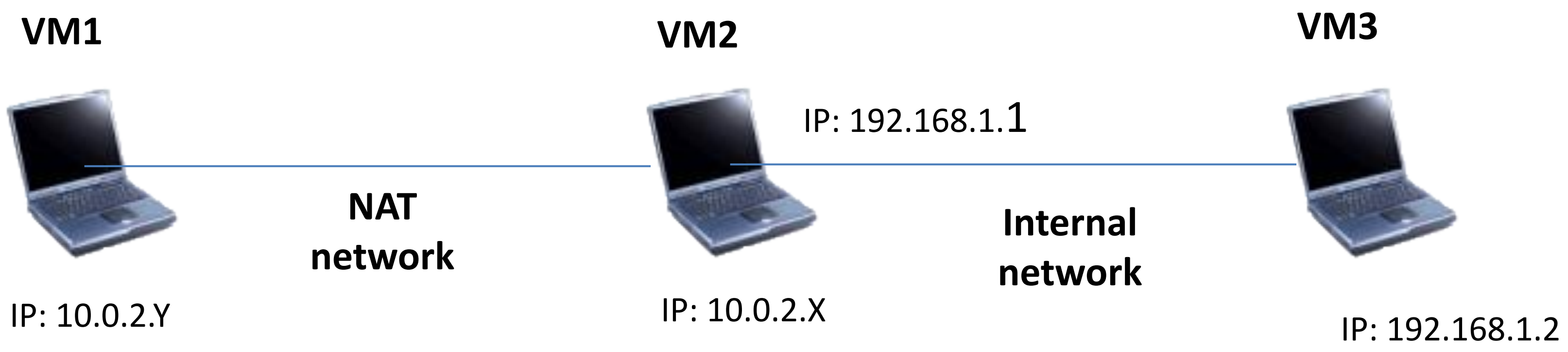

http://www.gugel.norg
Client should be configured to use SOCKS5 proxy at localhost:8080
SSH Client
Web traffic through SOCKS5 server on localhost:8080
SSH-encrypted tunnel
SSH Server on
IP: 147.83.118.30
TCP port: 10022
+ SOCKS5 proxy

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

SSH – LAB

Escenari

1. Necessitareu una tercera VM. Cloneu una de les dues MVs que teniu.
2. Abans de conectar la VM3 a la xarxa interna, comproveu si hi teniu instal·lat apache2 amb la comanda `systemctl status apache2 (*)`
3. Un cop instal·lat, connecteu VM2 i VM1 a través d'una xarxa interna



`(*) Si no teniu Apache instal·lat, connecteu primer la VM3 a NAT per tenir Internet, i descarregueu i instal·leu apache amb la comanda  apt install apache2`

Accés SSH

- Fent servir VM2 com a servidor ssh:
  - Configureu el fitxer `/etc/ssh/sshd_config` de tal manera que:
    - Es permeti l'accés mitjançant usuari root amb password i contrassenya.
    - Es permeti l'accés de l'usuari kali amb autenticació mitjançant clau pública
  - Reinicieu el servei `systemctl restart sshd`
  - Comproveu que us podeu connectar des del client ssh (VM1) al servidor (VM2) amb els dos usuaris: root i Kali.
    - `ssh root@10.0.2x`
    - `ssh kali @10.0.2.x.`

Port Forwarding

- Per poder configurar túnels SSH, és necessari canviar la configuración del fitxer `/etc/ssh/sshd_config`
  - Haureu d'habilitar els paràmetres AllowTcpForwarding, X11Forwading i GatewayPorts (han d'estar a "yes")
  - Reiniciar el servei
- Implementeu un Local Port Forwarding i un Remote Port Forwarding utilitzant l'escenari amb 3 VMs.
  - VM1 I VM2 són client I servidor SSH (o viceversa, depenent del tipus de túnel)
  - VM3 és el destination server