



Jakob Wittmann

## SSH 2

HTW Berlin

07.03.2021

# Gliederung

- Motivation
- SSH 2 Aufbau
- SSH 2 keys/Komponenten

# Motivation

- Easy and safe access to remote computers
- Prevents eavesdropping, spoofing, man in the middle attacks

Vgl.: Daniel J. Barrett, Richard E. Silverman (2001): The Secure Shell: The Definitive Guide  
3.10. Threats SSH Can Counter

# SSH 2 Aufbau

## SSH Transport Layer Protocol

- Server authentication
- Algorithm negotiation
- Session key exchange
- Privacy,integrity
- Session ID

## SSH User Authentication Protocol

- Client authentication (password,public key)
- Password changing

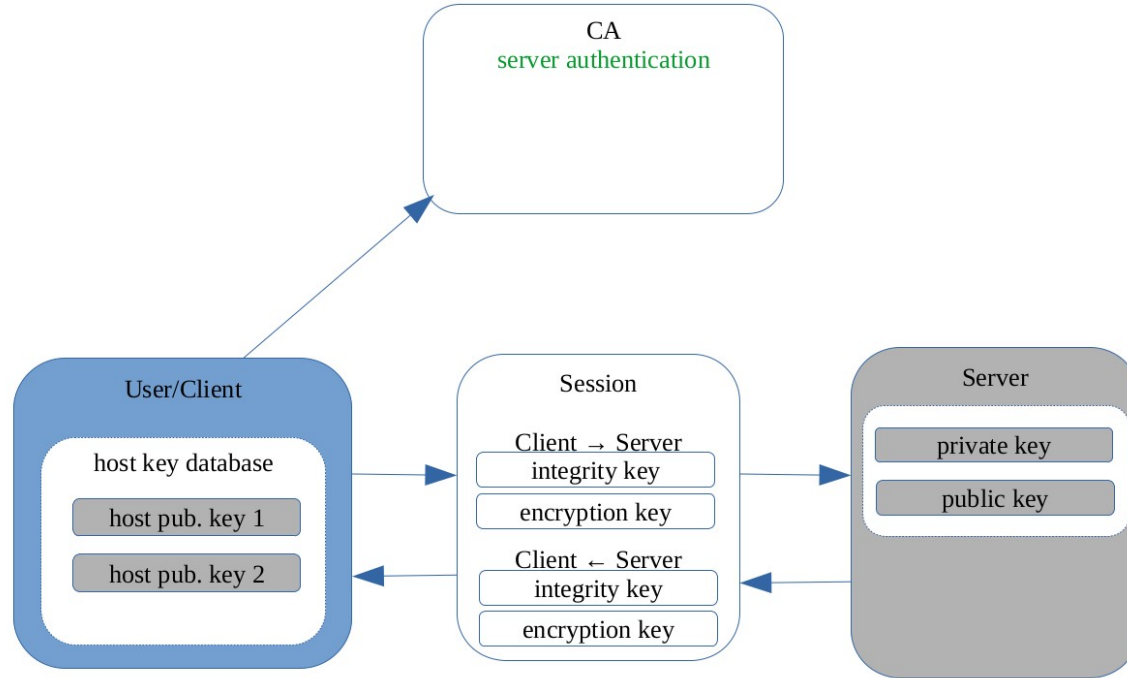
## SSH Connection Protocol

- Remote program execution
- Interactive session
- X forwarding
- ...

Vgl.: Daniel J. Barrett,Richard E. Silverman (2001): The Secure Shell: The Definitive Guide  
3.5. Inside SSH-2

# SSH Transport Layer Protocol

Optional ■



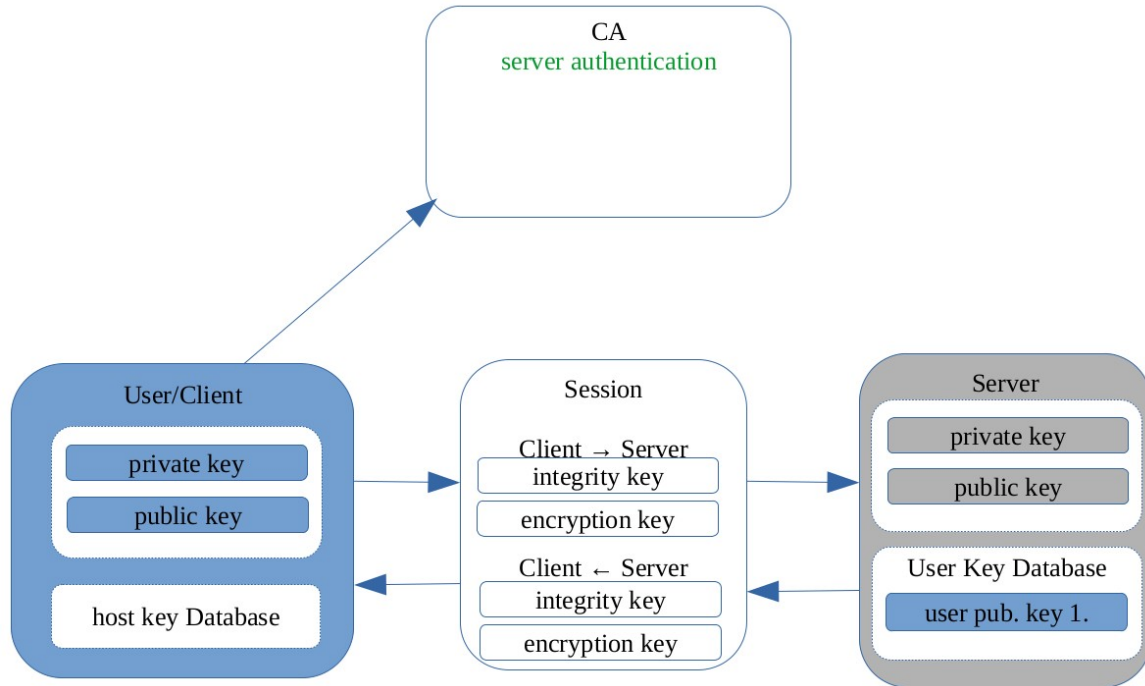
## Connection establishment

- Tcp connection
- Algorithm negotiation
- Client: (number)
- Server: (number, public key, signature)
- Both calculate session keys

Vgl.: Daniel J. Barrett, Richard E. Silverman (2001): The Secure Shell: The Definitive Guide 3.5. Inside SSH-2

# SSH User Authentication Protocol

Optional ■



## User authentication

- Client signs data with private key
- Server checks if a authorized public key exists

Vgl.: Daniel J. Barrett, Richard E. Silverman (2001): The Secure Shell: The Definitive Guide 3.5. Inside SSH-2  
O.V.(O.J.): OpenSSH: Key Management Needs Attention

# Deffie Hellmann Key Exchange

Grey: Server  
 Blue: Client  
 p: prime  
 g: generator for subgroup  
 q: order of subgroup  
 V\_S: Server identification string  
 V\_C: Clients identification string  
 K\_S: Server public host key  
 I\_C: Client SSH\_MSG\_KEXINIT  
 I\_S: Server SSH\_MSG\_KEXINIT

TCP connection is established	
SSH-protoversion-softwareversion SP comments CR LF	
SSH-protoversion-softwareversion SP comments CR LF	
Algorithm negotiation	
Algorithm negotiation	
beispielhaft Deffie Hellman Key Exchange	
Random number x $e = g^x \text{ mod } p$ <b>send e</b>	
	Random number y $f = g^y \text{ mod } p$ $K = e^y \text{ mod } p$ $H = \text{hash}(V_C    V_S    I_C    I_S    K_S    e    f    K)$ $s = K_S(H)$ <b>send (K_S    f    s)</b>
<b>Verify K_S</b> $K = f^x \text{ mod } p$ $H = \text{hash}(V_C    V_S    I_C    I_S    K_S    e    f    K)$	
verify s	
Both calculate Client → Server Encryption: $\text{HASH}(K    H    "C"    \text{session\_id})$ Server → Client Encryption: $\text{HASH}(K    H    "D"    \text{session\_id})$ Client → Server Integrity: $\text{HASH}(K    H    "E"    \text{session\_id})$ Server → Client Integrity: $\text{HASH}(K    H    "F"    \text{session\_id})$	
Repeat after one hour or 1Gb of data send	

Vgl.: RFC 4253

# Quellen

Daniel J. Barrett, Richard E. Silverman (2001): The Secure Shell: The Definitive Guide, First edition, O`Rilley, URL: [https://docstore.mik.ua/oreilly/networking\\_2ndEd/ssh/index.htm](https://docstore.mik.ua/oreilly/networking_2ndEd/ssh/index.htm)

SSH-Arch: Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Protocol Architecture", RFC 4251, January 2006

SSH-Userauth: Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Authentication Protocol", RFC 4252, January 2006

SSH-Trans: Ylonen, T. and C. Lonvick, Ed., "The Secure Shell (SSH) Transport Layer Protocol", RFC 4253, January 2006

O.V.(O.J.): SSH Protocol, SSH.com, URL: <https://www.ssh.com/ssh/protocol/>

O.V.(O.J.): OpenSSH: Key Management Needs Attention, SSH.com, URL: <https://www.ssh.com/ssh/openssh/>





**Hochschule für Technik  
und Wirtschaft Berlin**

University of Applied Sciences

[www.htw-berlin.de](http://www.htw-berlin.de)