Verteilte Systeme

...für C++ Programmierer

SSH

by

Dr. Günter Kolousek

SSH – Secure Shell

- Protokoll & Anwendungen
- Sicherheitsdienste
  - Geheimhaltung ✔
  - Integrität ✔
  - Authentifizierung ✔
  - Zugriffskontrolle (✔)
    - indirekt über Zugriffskontrolle des Servers
  - Nicht-Zurückweisung ✗
    - einseitig und indirekt über Logging-Mechanismen des Servers (falls vorhanden)

SSH - OpenSSH

Programme: umfangreiche Konfiguration → `man ssh`, `man sshd`,...

- Clients: `ssh`, `scp`, `sftp`, `sshfs`
  1. Kommandozeilenargumente
  2. `~/.ssh/config`
  3. `/etc/ssh/ssh_config`
- Server: `sshd`
  1. Kommandozeilenargumente
  2. `/etc/ssh/sshd_config`
- Unix: OpenSSH 7 (Protokoll Version 2)!
- Windows: `putty`, `winscp`

Geheimhaltung

- symmetrische Verschlüsselung
  - Schlüsselaustausch mittels Diffie-Hellman → Session Key → perfect forward security (PFS)
    - PFS → Session Key nicht aus Langzeitschlüssel regenerierbar!
  - verschiedene Algorithmen: Server bietet an, Client wählt!

```
ssh -Q cipher
```

```
3des-cbc
aes128-cbc
aes192-cbc
aes256-cbc
...
```

- Optionen (von `ssh`-Clients)
  - `-c` … Liste von Cipher-Spezifikationen (getrennt durch =,=)

Integrität

- Message Authentication Code (MAC)
  - verschiedene Algorithmen

```
ssh -Q mac
```

```
...
hmac-sha1
hmac-sha1-96
hmac-sha2-256
hmac-sha2-512
hmac-md5
...
umac-128-etm@openssh.com
```

- Optionen (von `ssh`-Clients)
  - `-m` … Liste von MACs

Authentifizierung

1. GSSAPI
   - Generic Security Service API (IETF Standard)
   - Schnittstelle zu Sicherheitsdienst
     - $\rightarrow$ Kerberos (verteilte Authentifizierung)
2. Host-basiert
   - `/etc/hosts.equiv`, Benutzernamen am Client und Server gleich
   - `~/.rhosts`: host + username

   \+ Hostkey vom Client in `~/.ssh/known_hosts`
3. Public Key
4. Challenge-Response
   - z.B. mit Google Authenticator
5. Passwort

Authentifizierung - 2

- Account muss zugreifbar sein
  - nicht gesperrt (`/etc/passwd`)
  - Benutzer nicht in `DenyUsers` ($\rightarrow$ `/etc/ssh/sshd_config`)
  - Gruppe nicht in `DenyGroups`

Login Vorgang

1. Ausgabe des letzten Login-Vorganges und `/etc/motd`
2. Zugriffszeit loggen
3. kein Zugriff wenn `/etc/nologin` (außer `root`) $\rightarrow$ Inhalt wird ausgegeben
4. Wechsel auf **normale** Benuzterprivilegien!
5. *Umgebungsvariable* setzen
   - wenn erlaubt (`PermitUserEnvironment` in `/etc/sshd_config`), dann Umgebungsvariable gemäß `~/.ssh/environment`!
6. Wechsel ins Home-Verzeichnis
7. `~/.ssh/rc` ausführen
   - wenn erlaubt $\rightarrow$ `PermitUserRC`
8. Shell oder Kommando ausführen

known_hosts - Datei

- Server: jeder Client, der sich mittels host-basierter Authentifizierung authentifizieren will benötigt einen Hostkey.
  - $\rightarrow K_{pub}$ in:
    - `/etc/ssh/known_hosts`
- Client: jeder Server besitzt Hostkey.
  - $\rightarrow K_{pub}$ in:
    - `/etc/ssh/known_hosts`
    - `~/.ssh/known_hosts`
- Abfrage eines K_{pub}
  - `ssh-keyscan host`
    - `ssh-keyscan ifssh.htlwrn.ac.at`

known_hosts - Client

1. Server überträgt Host-Key an Client
2. Client überprüft known_hosts
   - nicht enthalten → Benutzer kontaktieren!
   - enthalten
     - gleich → OK
     - nicht gleich → Benutzer kontaktieren! (möglicherweise Server spoofing)

→ `ssh-keygen -lv -f ~/.ssh/known_hosts`

- -l … zeigt Fingerprint
- -v … verbose → ASCII art des Keys
- -f … file → Angabe der Datei

Anwendungen

- Entfernter Zugriff
- Entfernte Ausführung
- Entferntes Kopieren
- Dateiserver
- Entferntes Dateisystem
- Port forwarding
- VPN
  - Layer 2 & Layer 3
  - für temporäre VPNs (da overhead)

Entfernter Zugriff

- → Systemadministration!
- `ssh username@hostname`
  - `ssh ko@ifssh.htlwrn.ac.at`
- Escape-Kommandos: `<Enter>~` gefolgt von
  - `.` … beendent Sitzung (inkl. aller Tunnels)
  - `&` … versetzt ssh in den Hintergrund
  - `<Ctrl>-z` … zeitweiliges Aussetzen (suspend) (→ `fg`)
  - `#` … zeigt alle Verbindungen über die gerade getunnelt wird
  - `?` … Hilfemenü
- Optionen
  - `-p PORT` … anstatt defaultmäßig 22

Entfernte Ausführung

- Starten von Anwendungen auf entfernten Hosts
- `ssh username@hostname command arguments`
  - `ssh ko@ifssh.htlwrn.ac.at ls public/nvs5`
  - d.h. wie entfernter Zugriff, also mit Shell als Kommando
- keine Benutzerangabe $\rightarrow$ lokaler Benutzer
- noch nie mit Server verbunden $\rightarrow$ `known_hosts`
- Optionen
  - `-t` ... Pseudo-Terminal $\rightarrow$ interaktive Programme
    - `ssh -t ko@ifssh.htlwrn.ac.at vim test.txt`

Entferntes Kopieren

- `scp source dest`
- zwischen localhost und remote host
  - `scp local.txt ko@ifssh.htlwrn.ac.at:`
  - `scp ko@ifssh...at:public/remote.txt .`
- zwischen zwei remote hosts
  - `scp ko@ifssh...at:public/nvs5/remote.txt ko@ifssh...at:remote.txt`
- Optionen
  - `-r` … rekursives Kopieren
  - `-p` … (preserve) gleiche Änderungszeit, Zugriffszeit und Modus wie Quelldatei
  - `-C` … Komprimierung (alle Anwendungen!)

Funktionalität eines Dateiservers

- interaktiv
- Verzeichnisse auflisten, navigieren, Dateien löschen, umbenennen (verschieben) Verzeichnisse anlegen, zum/vom Server kopieren (auch mit fortsetzen der Übertragung), Besitzer und Rechte ändern
- `sftp username@hostname`
  - `sftp ko@ifssh.htlwrn.ac.at`
  - `ls`, `cd`, `get`, `put`,..., `help`
- Optionen
  - `-a` ... unterbrochene Übertragung fortsetzen
  - `-b file` ... führt Inhalt der Datei im Batch aus
  - `-p` und `-r` ... wie bei `scp`

Entferntes Dateisystem

- Einbinden eines entfernten Verzeichnisses in den lokalen Verzeichnisbaum (engl. mount)
- `sshfs username@hostname:dir mountpoint`
  - `sshfs ko@ifssh.htlwrn.ac.at:public/nvs5 nvs5`
  - lokales Verzeichnis muss existieren
- unmounting: `fusermount -u mountpoint`
  - `fusermount -u nvs5`

Port forwarding

- lokalen Port forwarden
- dynamischen Port forwarden
- entfernten Port forwarden

Lokaler Port

- lokalen Port zu entfernten Host über Gateway per SSH
- `ssh -L lport:host:hostport sshgateway`
  - `ssh -L 1234:www.htlwrn.ac.at:80 ko@ifssh.htlwrn.ac.at`
    - Namensauflösung von `www.htlwrn.ac.at` am `ifssh.htlwrn.ac.at`!
- Optionen
  - `-g` … entfernte Rechner können auf lokalen Port des lokalen Rechners zugreifen
  - `-N` … führt keine entfernte Kommandos aus, d.h. auch keine Shell
- Anwendung: Zugriff auf entfernten Host
  - mittels sicherer Kommunikation
  - zu Umgehung einer Beschränkung (Firewall)

Tunnel verschachteln

- Tunnel von `hostB` → `hostC`
  - auf `hostB`
    - `ssh -gL 2222:hostD:22 hostC`
- Tunnel von `hostA` → `hostD` über `hostB` & `hostC`
  - auf `hostA`
    - `ssh -p 2222 -gL 3333:server:3333 hostB`
- `client` verbindet sich zu `hostA`, Port 3333
  - und erlangt Verbindung zu `server`, Port 3333
- Anwendung: Kein direkter Tunnel möglich, daher Umweg über anderen Tunnel

Dynamischer Port

- vom lokalen Port zu beliebigen Hosts über Gateway per SSH
  - wird zu SOCKS Proxy
- `ssh -D lport sshgateway`
  1. `ssh -C -D 9999 ko@ifssh.htlwrn.ac.at`
  2. Firefox: Preferences → Advanced → Connection → Settings
     - *Manual proxy configuration* → checked
     - *Use this proxy server for all protocols* → unchecked
     - alle bis auf SOCKS leeren
     - SOCKS: *localhost* und *9999*
  3. Firefox: DNS ebenfalls über Proxy
     - `about:config` → `network.proxy.socks_remove_dns` → `true`

Entfernter Port

- entfernten Port vom Gateway über lokalen Host auf anderen Host und Port weiterleiten
- `ssh -R rport:host:hostport sshgateway`
  - `ssh -R 8080:edvoexam.htlwrn.ac.at:80 ko@ifssh.htlwrn.ac.at`
    - → Webbrowser auf `ifssh.htlwrn.ac.at` starten und `localhost:8080` → Intranetwebsite zugreifbar
  - `ssh -R 2222:edvoexam.htlwrn.ac.at:22 ko@ifssh.htlwrn.ac.at`
    - → `ssh -p 2222 localhost` auf `ifssh.htlwrn.ac.at` → SSH Zugriff auf `edvoexam`!
- Anwendung: Zugriff auf Intranet obwohl kein Zugriff von außen erlaubt!
  - *wenn du gerne entlassen werden willst…*

Passwörter vs. Keys

- Passwörter
  - brechbar, wenn Server kompromittiert (Zugriff auf Hashes)
  - ”sniffable”
  - etwas was man weiß: werden oft wiederverwendet, oft schwach, schlecht verwaltet (Zettel, einfache Textdateien)
- Keys
  - nicht brechbar
  - nicht ”sniffable”
  - etwas was man hat: gut zu verwalten (mit Passwort)
    - je ein Key je Server!

Public/Private Keypaar

- Key-Paar generieren
  - $\rightarrow$ `ssh-keygen`
  - Server generiert ein Key-Paar bei Installation
    - und wenn Sysadmin…
- K_{pub} von Server $\rightarrow$ `~/.ssh/known_hosts` am Client
  - $\rightarrow$ `known_hosts - Client`
- K_{pub} von Benutzer $\rightarrow$ `~/.ssh/authorized_keys` am Server
  - $\rightarrow$ Public Key-basierte Authentifizierung

Public/Private Key generieren

- `ssh-keygen`
  - → public und private nach `~/.ssh`
    - z.B. `id_rsa` und `id_rsa.pub`
  - Typ `-t rsa`
  - Länge `-b 2048`
- RSA mit 4096 Bits und Kommentar zum Identifizieren des Schlüssels
  - `ssh-keygen -C ko@htlwrn.ac.at -b 4096 -f id_rsa`

Public Key zum Server

- Key muss in `~/.ssh/authorized_keys`
- Manuell
  1. `scp ~/.ssh/id_rsa.pub ko@ifssh.htlwrn.a.cat:`
  2. `ssh ko@ifssh.htlwrn.ac.at`
     - 2.1 `mkdir .ssh`
     - 2.2 `chmod 700 .ssh`
     - 2.3 `cat id_rsa.pub » ~/.ssh/authorized_keys`
     - 2.4 `rm id_rsa.pub`
     - 2.5 `chmod 600 ~/.ssh/authorized_keys`
- `ssh-copy-id`
  - `ssh-copy-id ko@ifssh.htlwrn.ac.at`
  - `ssh-copy-id -i ~/.ssh/id_rsa.pub ko@ifssh.htlwrn.ac.at`

Key verwenden

- `ssh ko@ifssh.htlwrn.ac.at`
  - $\rightarrow$ kein Login mehr notwendig
- Mehrere Keys…
  - jeweils generieren und…
  - verwenden: `ssh -i ~/.ssh/id_rsa ko@htlwrn.ac.at`

Passphrase für Key

- `ssh-keygen` mit Passphrase
  - sicherer, aber…
  - jedes Mal ist die Passphrase notwendig!
- `ssh-agent`
  - speichert *entschlüsselten* K_{priv} im Speicher
  - `ssh-agent`
    - manuell starten
    - in Einstellungen wie XFCE, Gnome,…
  - Privaten Key zum `ssh-agent` hinzufügen:
    - `ssh-add ~/.ssh/id_rsa`
  - Anzeigen der privaten Keys vom Agent
    - `ssh-add -L`

Anwendungen von `ssh-keygen`

- Fingerprint von Public Key
  - `ssh-keygen -l -f id_rsa.pub`
- Public Key regenerieren
  - wenn ”verloren”
  - `ssh-keygen -y -f id_rsa > id_rsa.pub`
- Passphrase ändern
  - `ssh-keygen -p -f id_rsa`

Lokale Datei `config`

- im `.ssh` Verzeichnis
- Zum Konfigurieren des Clients (vielfältige Möglichkeiten)
- Beispiel

```
Host ifssh
    HostName ifssh.htlwrn.ac.at
    User ko
```

Danach anstatt

```
ssh ko@ifssh.htlwrn.ac.at
```

folgende Möglichkeit

```
ssh ifssh
```

Lokale Datei `config` – 2

- auch andere Möglichkeiten
- z.B. Tunnel

```
Host tunnel_to_www_htlwrn
    HostName ifssh.htlwrn.ac.at
    User ko
    LocalForward 1234 www.htlwrn.ac.at:80
```