**Studienarbeit:**

# SSH in einem heterogenen Umfeld am Beispiel der Betriebssysteme Solaris, Irix und Windows NT/95

**Student:** **Peter Schmidt**
**Matrikelnummer:** **3316270**

# SSH in einem heterogenen Umfeld am Beispiel der Betriebssysteme Solaris, Irix und Windows NT/95

# 1 Überblick

## *1.1 Was ist SSH?*

SSH - ein Akronym für „Secure Shell“ -ermöglicht sicher authentifizierte, verschlüsselte, komprimierte Verbindungen über unsichere Kommunikationswege.

## *1.2 Entstehung von SSH*

SSH wurde von Tatu Ylonen (ylo@cs.hut.fi) geschrieben und wird nun von der Firma Data Fellows (http://www.datafellows.com) und der Firma SSH Communications Security (http://www.ssh.fi) weiterentwickelt.

## *1.3 Warum SSH?*

SSH ermöglicht das Einloggen in ein entferntes System (telnet- und rlogin-Ersatz), entfernte Kommandoausführung (rsh-Ersatz), Kopieren von Files (rcp-Ersatz) sowie das Weiterleiten von X-Verbindungen und TCP-Verbindungen wie SMTP, POP3 oder HTTP

### 1.3.1 SSH als BSD-‘r’-Kommandoersatz

Die Semantik der SSH-Kommandos ist eine Obermenge der BSD-‘r’-Kommandos und somit kann SSH als Ersatz für diese Kommandos genutzt werden.

- rsh
  Mit rsh werden Kommandos an eine entfernte Maschine abgesetzt. Standardein- und ausgabe und Standardfehlerausgabe werden dabei umgeleitet sowie Signalhandling durchgeführt.
  Die Kommandos werden serverseitig von rshd, der rsh-Serverapplikation bearbeitet, rshd wird von inetd (ein Verbindungsverwaltungsprozess) bei eingehenden Verbindungen von rsh-Clients gestartet.
  Die Authentifizierung bei Gleichheit des entfernten Benutzernamens basiert auf der IP-Adresse des Clients und der Verwendung von privilegierten Ports clientseitig. Bei abweichendem entferntem Benutzernamen vom lokalen werden Passwörter unverschlüsselt übertragen.
  Die Verwendung von rsh setzt Vertrauen des Servers gegenüber dem Client voraus.
  Die Daten (Standardein-/ausgabe und Standardfehlerausgabe) werden bei Verwendung von rsh unverschlüsselt über das Netzwerk übertragen.
- rlogin
  Bei rlogin wird rsh dazu benutzt eine interaktive Shell auf der Servermaschine zu starten.
- rcp
  Bei rcp wird mit Hilfe von rsh ein Dateikopiervorgang von einer zur anderen Maschine gestartet.

### 1.3.2 Nachteile der BSD-‘r’-Kommandos

Beim Einsatz der BSD-‘r’-Kommandos werden die Daten unverschlüsselt über das Netz geschickt und können mit geeigneten Programmen mitgelesen und protokolliert werden.
Ausserdem kann jeder Client, der in der Lage ist IP-Adressen zu fälschen, eine Verbindung zum Server aufbauen und wird vom Server als authentisch angesehen.

### 1.3.3 Sichere Kommunikation über TCP/IP-basierte Netze

#### 1.3.3.1 Authentizität der Benutzer und Rechner mittels RSA

RSA (benannt nach seinen Erfindern Rivest, Shamir und Adleman) ist der meisteingesetzte Public-Key-Algorithmus. Er basiert auf der Tatsache, das sich sehr große Zahlen nur mit derzeit nicht praktikablem Aufwand in Primzahlen faktorisieren lassen. Es werden Schlüsselpaare (Zahlenpaare) erzeugt, von denen der eine den öffentlichen Schlüssel darstellt, der bekanntgegeben wird und der andere den privaten Schlüssel, der geheimgehalten wird. Die Daten werden mit dem öffenlichen Schlüssel des Empfängers verschlüsselt und nur der Besitzer der privaten Schlüssels ist in der Lage mit seinem privaten Schlüssel die Daten zu entschlüsseln.

Bei Konnektieren des SSH-Servers durch den Client wird das zu benutzende Schlüsselpaar ausgehandelt. Danach sendet der Server eine mit dem serverseitig hinterlegten öffentlichen Schlüssel des Benutzers formulierte Frage (Challenge), der Client dechiffriert diese Frage mit seinem Schlüssel und gibt die Antwort an den Server zurück.

Die Identität der Server ist ebenfalls über solche Schlüsselpaare geregelt und somit ist die Authenizität der Server untereinander unabhängig vom darunterliegenden Transportsystem (IP-Adresse bzw. Auflösung durch Nameserver) gewährleistet.

#### 1.3.3.2 Verschlüsselung der Daten

Wahlweise können folgende Algorithmen zur Verschlüsselung der übertragenen Daten verwandt werden (als Standardalgorithmus wird bei SSH IDEA eingesetzt):

- BLOWFISH
  ist ein symmetrischer Blockchiffre und kann mit variabler Blocklänge von 32 bit bis 448 bit benutzt werden. Er wurde 1993 von Bruce Schneier entwickelt. Seine Vorteil sind seine Lizenz- und Patentfreiheit und seine höhere Geschwindigkeit gegenüber DES und IDEA.
- ARCFOUR/RC4
  ist ein schneller Streamchiffrierer, d.h. er arbeitet auf Bitebene. Er wurde 1987 von Ron Rivest, dem Miterfinder des RSA-Verfahrens, entwickelt. Er besitzt eine variable Schlüssellänge bis zu 2048 bit. Er ist sehr kompakt und 5-10 mal schneller als DES.
- IDEA
  steht für International Data Encryprion Algorithm. IDEA arbeitet mir einem 128 bit Blockchiffre und ist schneller und sicherer als DES. Was die Benutzung von IDEA problematisch machen kann, sind ggf. zu zahlende Lizenzgebühren an die Firma Ascom mit Sitz in der Schweiz, die ein Patent auf diesen Algorithmus hat.
- DES
  steht für Data Encryption Standard und ist ein sogenannter symmetrischer Blockchiffre mit 64 Bit Blöcken, die mit einem Schlüssel einer Länge von 56 Bit arbeiten. Er wurde in den USA standardisiert.
- 3DES
  steht für Triple-DES und ist eine Variation von DES, bei der mit drei Schlüsseln, die den Gesamtschlüssel bilden, nacheinander verschlüsselt, entschlüsselt und wieder verschlüsselt wird.
- TSS
  ist ein schneller Algorithmus der SSH-Entwickler

### 1.3.4 Kompression der Daten

Die übertragenen Daten werden mit dem LZL77 Lempel-Ziv Algorithmus komprimiert, der auch bei GNU ZIP zum Einsatz kommt. Dieses Kompressionsverfahren hat sich inzwischen als Standard etabliert. Bei SSH werden 9 Komressionstiefen zur Wahl gestellt: 1 bedeutet schnell und schlecht komprimiert, 9 bedeutet langsam und optimal komprimiert.

### 1.3.5 Weite Verfügbarkeit

SSH ist unter den verschiedensten UNIX-Varianten verfügbar, es werden ständig neue Portierungen vorgenommen. Ein Auszug aus der Portabilitätsliste (http://www.cs.hut.fi/ssh/#portability) verdeutlicht dieses:

386BSD 0.1; i386
AIX 3.2.5, 4.1, 4.2; RS6000, PowerPC
A/UX 3.1.1
BSD 4.4; several platforms
BSD/OS 1.1, 2.0.1, 3.0; i486
BSD/386 1.1; i386
BSDI 2.1, 3.0; x86 (using gnu make)
ConvexOS 10.1; Convex
Digital Unix 4.0, 4.0A, 4.0B; Alpha
DGUX 5.4R2.10; DGUX
DolphinOS 3.8
FreeBSD 1.x, 2.x, 3.0; Pentium

HPUX 7.x, 9.x, 10.0; HPPA
IRIX 5.2, 5.3, 6.2, 6.3; SGI Indy
IRIX 6.0.1; Mips-R8000
Linux 1.2.x, 2.0.x Slackware 2.x, 3.x, RedHat 2.1, 3.0; i486, Sparc
Linux 3.0.3, 4.0; Alpha
Linux/Mach3, Macintosh(PowerPC)
Linux/m68k (1.2.x, 2.0.x, 2.1.x)
Mach3; Mips
Mach3/Lites; i386
Machten 2.2VM (m68k); Macintosh
NCR Unix 3.00; NCR S40
NetBSD 1.0A, 1.1, 1.2, 1.3; Pentium, Sparc, Mac68k, Alpha
OpenBSD 2.0; x86, mac68k.
NextSTEP 3.3; 68040
OSF/1 3.0, 3.2, 3.2; Alpha
Sequent Dynix/ptx 3.2.0 V2.1.0; i386
SCO Unix; i386 (client only)
SINIX 5.42; Mips R4000
Solaris 2.3, 2.4, 2.5, 2.5.1, 2.6; Sparc, i386
Sony NEWS-OS 3.3 (BSD 4.3); m68k
SunOS 4.1.1, 4.1.2, 4.1.3, 4.1.4; Sparc, Sun3
SysV 4.x; several platforms
Ultrix 4.1; Mips
Unicos 8.0.3; Cray C90

Darüber hinaus sind für viele andere Betriebssysteme wie Windows 3.x/95/98/NT, Beos, OS/2, Macinthosh, OpenVMS, Palmpilot, Amiga und diverse Organzier wie PalmPilot Clients verfügbar.

Es existieren Client-Implementationen in Java, was bei Verfügbarkeit einer Java-Virtual-Machine für eine Platform auch die Verfügbarkeit eines SSH-Clients bedeutet.

# 2 SSH unter Unix

## *2.1 SSH unter Unix aus Benutzersicht*

### 2.1.1 Voraussetzungen

Um SSH benutzen zu können müssen folgende Voraussetzungen erfüllt sein:

- das Paket ist vom Systemverwalter installiert worden (das Paket enthält Client und Server): **2.2 SSH unter Unix aus Systemverwaltersicht** (Seite 28).
- Der Benutzer hat einen Shell-Account (eine Benutzerkennung) auf der Client-Maschine und der Server-Maschine

### 2.1.2 Erzeugen von Schlüssel-Paaren für Public-Key-Authentifizierung

Um Public-Key-Authentifizierung betreiben zu können, muß ein Schlüsselpaar erzeugt werden: ein privater Schlüssel für den Client und ein öffentlicher Schlüssel (Public-Key) für den Server.

#### 2.1.2.1 Erzeugen von Schlüssel-Paaren für SSH1

Zur Erzeugung eines Schlüsselpaares für SSH1 steht das Hilfsprogramm ssh-keygen1 zur Verfügung. Das Schlüsselpaar wird wie folgt erzeugt:

```
user@client $ ssh-keygen1 ↵
Initializing random number generator...
Generating p:  ..................................++ (distance 488)
Generating q:  ........................++ (distance 364)
Computing the keys...
Testing the keys...
Key generation complete.
Enter file in which to save the key (/home/user/.ssh/identity): ↵
Enter passphrase: <Passphrase> ↵
Enter the same passphrase again: <Passphrase> ↵
Your identification has been saved in /home/user/.ssh/identity.
Your public key is:
1024 37 134384290514104526517907785289933578837562030078052101714004520311883377414907133379457676713861060940132867067747309432102176969198828416047064747980499734876478211032231239600559538713047460303274469020601295238955889363923822576764454320895352058400413771098522851521717090588830623463747191468721380201017 user@client
Your public key has been saved in /home/user/.ssh/identity.pub
```

Bei dieser Schlüsselgenerierung ist zum Schutz des privaten Schlüssels eine Passphrase anzugeben, diese sollte nicht mit dem benutzten Passwort der Benutzerkennung identisch sein. Mit dieser Passphrase wird der private Schlüssel 3DES-verschlüsselt, um Mißbrauch durch Kopieren des privaten Schlüssels zu verhindern.
Nach erfolgter Schlüsselgenerierung wird der private Schlüssel nur für den Benutzer lesbar (Dateimodus 600) unter $HOME/.ssh/identity und der öffentliche Schlüssel für alle lesbar (Dateimodus 644) unter $HOME/.ssh/identity.pub abgelegt.
Bei der Schlüsselgenerierung können verschiedene Optionen angegeben werden:

- -b bits
  gibt die Schlüssellänge der erzeugten Schlüssel an; Standardwert ist 1024 Bits, Minimalwert ist 512 Bits
- -f file
  gibt den Dateinamen an, unter dem der Schlüssel abgelegt wird; Standardwert ist $HOME/.ssh/identify
- -N Passphrase
  Gibt die zu verwendende Passphrase zum Schutz des privaten Schlüssels an. Die Verwendung dieser Option ist nicht empfehlenswert, da die Passphrase im Klartext im Prozeßstatus sichtbar wird.
- -C Kommentar
  Vergibt einen Kommentar an das Schlüsselpaar; Standardwert ist user@client.

Ein bereits erzeugter Schlüssel kann mit folgenden Optionen verändert werden:

- -p
  ändert die Passphrase des Schlüssels (es muß die aktuelle Passphrase angegeben werden.
- -c
  ändert den Kommentar des Schlüssels (es muß die aktuelle Passphrase angegeben werden

#### 2.1.2.2 Erzeugen von Schlüssel-Paaren für SSH2

Zur Erzeugung eines Schlüsselpaares für SSH2 steht das Hilfsprogramm ssh-keygen2 zur Verfügung. Das Schlüsselpaar wird wie folgt erzeugt:

```
user@client $ ssh-keygen2 ↵
Generating 1024-bit dsa key pair
   1 oOo
Key generated.
1024-bit dsa, created by user@client Tue Feb 16 11:58:22 1999
Passphrase : <Passphrase> ↵
Again      : <Passphrase> ↵
Private key saved to /home/user/.ssh2/id_dsa_1024_a
Public key saved to /home/user/.ssh2/id_dsa_1024_a.pub
```

Bei dieser Schlüsselgenerierung ist zum Schutz des privaten Schlüssels eine Passphrase anzugeben, diese sollte nicht mit dem benutzten Passwort der Benutzerkennung identisch sein. Mit dieser Passphrase wird der private Schlüssel 3DES-verschlüsselt, um Mißbrauch durch Kopieren des privaten Schlüssels zu verhindern.
Nach erfolgter Schlüsselgenerierung wird der private Schlüssel nur für den Benutzer lesbar (Dateimodus 600) unter $HOME/.ssh2/id_dsa_1024_a und der öffentliche Schlüssel für alle lesbar (Dateimodus 644) unter $HOME/.ssh/id_dsa_1024_a.pub abgelegt.
Bei der Schlüsselgenerierung können verschiedene Optionen angegeben werden:

- -b bits
  gibt die Schlüssellänge der erzeugten Schlüssel an; Standardwert ist 1024 Bits, Minimalwert ist 512 Bits
- -o file
  gibt den Dateinamen an, unter dem der Schlüssel abgelegt wird; Standardwert ist $HOME/.ssh2/ id_dsa_1024_a (bei DSA-Schlüsseln mit 1024 Bit Länge)
- -t key_algorithm
  gibt den verwendeten Algorithmus bei der Schlüsselerzeugung an. DSS (Digital Signature Standard) und RSA sind möglich, RSA allerdings nur in der kommerziellen Version von SSH2.
- -c Kommentar
  Vergibt einen Kommentar an das Schlüsselpaar; Standardwert ist “1024-bit dsa, created by user@client“.
- -p Passphrase
  Gibt die zu verwendende Passphrase zum Schutz des privaten Schlüssels an. Die Verwendung dieser Option ist nicht empfehlenswert, da die Passphrase im Klartext im Prozeßstatus sichtbar wird.
- -q
  Bei Angabe dieser Option wird keine Fortschrittsanzeige dargestellt
- -r
  Diese Option veranlaßt ssh-keygen2 Zeichen vom Benutzer einzulesen, die in die Zufallszahlengenerierung einfließen

Ein bereits erzeugter Schlüssel kann mit folgender Option verändert werden:

- -e
  ändert die Passphrase und/oder den Kommentar des Schlüssels (es muß die aktuelle Passphrase angegeben werden.

### 2.1.3 Aktivieren der Public-Key-Schlüsselpaare

Der öffentliche Schlüssel des Benutzers muß auf den Server transportiert werden und dort als zugelassener Schlüssel eingetragen werden. Dies geschieht sinnvollerweise per Diskette oder mit PGP-verschlüsselter Email.

### 2.1.3.1 Aktivieren der Public-Key-Schlüsselpaare für SSH1

- Kopieren des öffentlichen Schlüssels auf Diskette:
  `user@client $` *cp $HOME/.ssh/identity.pub /floppy/floppy0/client1.pub* ↵
- Eintragen des öffentlichen Schlüssels als zugelassener Schlüssel:
  `user@server $` *cat /floppy/floppy0/client1.pub >> $HOME/.ssh/authorized_keys* ↵
- Setzen der Berechtigungen für die Authorisationsdatei:
  `user@server $` *chmod 644 $HOME/.ssh/authorized_keys* ↵

Weiter Informationen zum Aufbau der Authorisationsdatei sind zu finden in **2.1.4.1.9.1 Die Authorisationsdatei authorized_keys** (Seite 24)

### 2.1.3.2 Aktivieren der Public-Key-Schlüsselpaare für SSH2

- Eintragen des privaten Schlüssels als zu benutzenden privaten Schlüssel:
  `user@client $` *echo IdKey id_dsa_1024_a >> $HOME/.ssh2/identification* ↵
  Bemerkung: Es sind mehrere private Schlüssel möglich.
- Setzen der Berechtigungen für die Identifikationsdatei:
  `user@client $` *chmod 600 $HOME/.ssh2/identification* ↵
- Kopieren des öffentlichen Schlüssels auf Diskette:
  `user@client $` *cp $HOME/.ssh2/id_dsa_1024_a /floppy/floppy0/client2.pub* ↵
- Kopieren des öffentlichen Schlüssels von Diskette:
  `user@server $` *cp /floppy/floppy0/client2.pub $HOME/.ssh2/client2.pub* ↵
- Eintragen des öffentlichen Schlüssels als zugelassener Schlüssel:
  `user@server $` *echo Key client2.pub >> $HOME/.ssh2/authorization* ↵
  Bemerkung: Es sind mehrere öffentliche Schlüssel möglich.
- Setzen der Berechtigungen für die Authorisationsdatei:
  `user@server $` *chmod 644 $HOME/.ssh2/authorization* ↵

## 2.1.4 Benutzen von SSH

### 2.1.4.1 Benutzen von SSH1

#### 2.1.4.1.1 Interaktives Login

Es wird ermöglicht interaktive Kommandos wie z.B. den Editor vi oder Shellbefehle auf dem Server auszuführen, die Funktionalität ist eine Obermenge des Remote-Shell-Kommandos rsh bzw. rlogin.
Die erste interaktive Sitzung:

```
user@client $ ssh1 server ↵
Host key not found from the list of known hosts.
Are you sure you want to continue connecting (yes/no)? yes ↵
Host 'server' added to the list of known hosts.
Enter passphrase for RSA key 'user@client': <Passphrase> ↵
user@server $ vi ↵
...Arbeiten mit interaktiven Kommandos auf dem Server...
user@server $ exit ↵
Connection to server closed.
user@client $
```

Der SSH1-Client erstellt und aktualisiert automatisch eine Liste der öffentlichen Serverschlüssel in der Datei $HOME/.ssh/known_hosts. Damit läßt sich sicherstellen, daß der konnektierte Server auch der ist, der er vorgibt zu sein. Weitere Informationen zum Aufbau der Datei $HOME/.ssh/known_hosts sind zu finden in **2.1.4.1.9.2 Die öffentliche Serverschlüsseldatei known_hosts** (Seite 25). Die öffentlichen Serverschlüssel lassen sich auch manuell übertragen, um bei der ersten Verbindung ggf. Zweifel an der Authentizität des Servers auszuräumen:

- Kopieren des öffentlichen Schlüssels des Server auf Diskette:
  `user@server $` *cp /etc/ssh_host_key.pub /floppy/floppy0/hostkey1.pub* ↵
- Eintragen des öffentlichen Schlüssels des Server als authentisch:
  `user@client $` *cat /floppy/floppy0/hostkey1.pub >> $HOME/.ssh/known_hosts* ↵

- Setzen der Berechtigungen für die Datei:

```
user@client $ chmod 600 $HOME/.ssh/known_hosts ↵
```

Bei jeder folgenden Sitzung mit dem gleichen Server entfällt diese Vorgehensweise:

```
user@client $ ssh1 server ↵
Enter passphrase for RSA key 'user@client': <Passphrase> ↵
user@server $ vi ↵
...Arbeiten mit interaktiven Kommandos auf dem Server...
user@server $ exit ↵
Connection to server closed.
user@client $
```

Bei anderslautendem Benutzernamen auf dem Server sieht der Aufruf wie folgt aus:

```
user@client $ ssh1 –l serveruser server ↵
```

#### 2.1.4.1.2 entfernte Kommandoausführung

Nichtinteraktive Kommandos lassen sich direkt übergeben, z.B.:

```
user@client $ ssh1 server ls -la /tmp ↵
```

Dabei wird die Standardausgabe des Kommandos auf den Client umgeleitet.
Die Standardausgabe des Kommandos läßt sich auch in eine Datei umleiten; hierbei gibt es zwei Möglichkeiten:

- Ausgabe in eine Datei auf dem Client:

  ```
  user@client $ ssh1 server ls -la /tmp > ls.out ↵
  ```

  Hier wird das Ausgabeumleitungssymbol „>“ durch die Shell des Clients interpretiert und auf dem Client eine Datei namens ls.out erzeugt, die die Ausgabe des auf dem Server ausgeführten Kommandos enthält.
- Ausgabe in eine Datei auf dem Server:

  ```
  user@client $ ssh1 server ls -la /tmp “>“ ls.out ↵
  ```

  Hier wird das Ausgabeumleitungssymbol „>“ durch die Shell des Servers interpretiert und auf dem Server eine Datei namens ls.out erzeugt, die die Ausgabe des auf dem Server ausgeführten Kommandos enthält.

Es ist möglich den Prozeß in den Hintergrund zu stellen; dies geschieht nicht wie üblich durch Anhängen eines & an die Kommandozeile; in diesem Fall würde die Authentifizierung scheitern, da sie von Standardeingabe die Passphrase liest. Der SSH1-Client hat dazu die spezielle Kommandozeilenoption „-f“, der Aufruf sieht wie folgt aus:

```
user@client $ ssh1 –f server ls -la /tmp “>“ ls.out ↵
```

#### 2.1.4.1.3 Benutzen des Authentifizierungs-Agent

Um beim mehrmaligen Benutzen von SSH1 den Benutzer vom Eingeben der Passphrase des lokalen Schlüssels zu verschonen, existiert ein Programm, daß diese Informationen bei Bedarf liefert: ssh-agent1. Der Authentifizierungs-Agent wird zu Beginn einer Sitzung auf dem Client-Rechner gestartet und alle Kindprozesse des Authentifizierungs-Agent erben dessen Authentifizierungsinformationen.
Eine Beispielsitzung:

```
user@client $ ssh-agent1 $SHELL ↵
user@client $ ssh-add1 ↵
Need passphrase for /home/user/.ssh/identity (user@client).
Enter passphrase: <Passphrase> ↵
Identity added: /home/user/.ssh/identity (user@client)
user@client $ ssh1 server ↵
user@server $
```

Der Authentifizierungsagent wirkt über SSH1-Verbindungen hinweg, d.h. auch in der Sitzung auf dem Server besteht Zugriff auf den Agent:

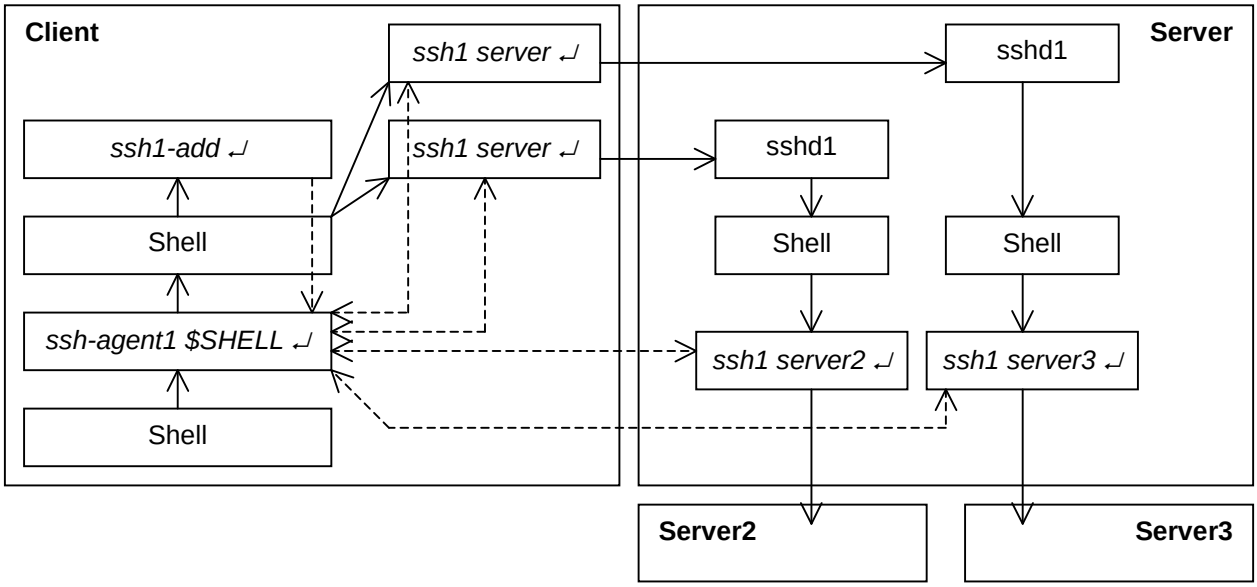



**Abbildung 2-1 der Authentifizierungs-Agent ssh-agent1**

Das Weiterleiten des Authentifizierungsagent über Verbindungen hinweg läßt sich mit der Kommandooption „–a“ von ssh1 deaktivieren.

Der Authentifizierungs-Agent ssh-agent1 kann auch ohne Subshell direkt in den Hintergrund gestellt werden:

- Die Syntax für den Aufruf aus einer Shell, die Environment-Variablen wie die Bourne-Shell exportiert lautet:

```
user@client $ eval `ssh-agent1 –s` ↵
Agent pid 1234
user@client $
```

  Das Entfernen des Agents geschieht wie folgt:

```
user@client $ eval ‘ssh-agent1 –k –s` ↵
Agent pid 1234 killed
user@client $
```

- Die Syntax für den Aufruf aus einer Shell, die Environment-Variablen wie die c-Shell exportiert lautet:

```
user@client % eval `ssh-agent1 –c` ↵
user@client %
```

  Das Entfernen des Agents geschieht wie folgt:

```
user@client % eval ‘ssh-agent1 –k –c` ↵
Agent pid 1234 killed
user@client %
```

Das Programm ssh-add1, mit dem Identitäten zum Authentifizierungs-Agent hinzugefügt werden können besitzt einige nützliche Optionen:

- Mit der Option „–l“ lassen sich alle derzeit im Agent gespeicherten Identitäten auflisten:

```
user@client $ ssh-add1 -l ↵
1024 35 100044540655809391053405092913118251256115814348514378570069818870335086668590361797185610507441524297595604120988670136946961370007687832211279130947040401025409240468419650274932288241125289559715988612503490933366739306129571251009557843537482488096477306725197022891122041074330001728792989708979878898541 user@client
user@client $
```

- Mit der Option „–p“ wird die Passphrase von der Standardeingabe gelesen und die Identität, die durch den nachfolgenden Dateinamen angegeben ist, hinzugefügt.

```
user@client $ ssh-add1 -p $HOME/.ssh/identity ↵
Need passphrase for /home/user/.ssh/identity (user@client).
```

```
Enter passphrase: <Passphrase> ↵
Identity added: /home/user/.ssh/identity (user@client)
user@client $
```

Diese Option ist die Standardoption, d.h. ohne Parameter an ssh-add1 zu übergeben wird diese Option benutzt.

- Mit der Option „–d“ läßt sich eine Identität aus dem Authentifizierungsagent löschen
- Mit der Option „–D“ lassen sich alle Identitäten aus dem Authentifizierungsagent löschen.

Wenn clientseitig ein X-Display gesetzt und ein X-Server verfügbar ist, läßt sich die Passphrase auch mit einem grafischen Benutzerinterface einlesen:

```
user@client $ ssh-add1 </dev/null ↵
```

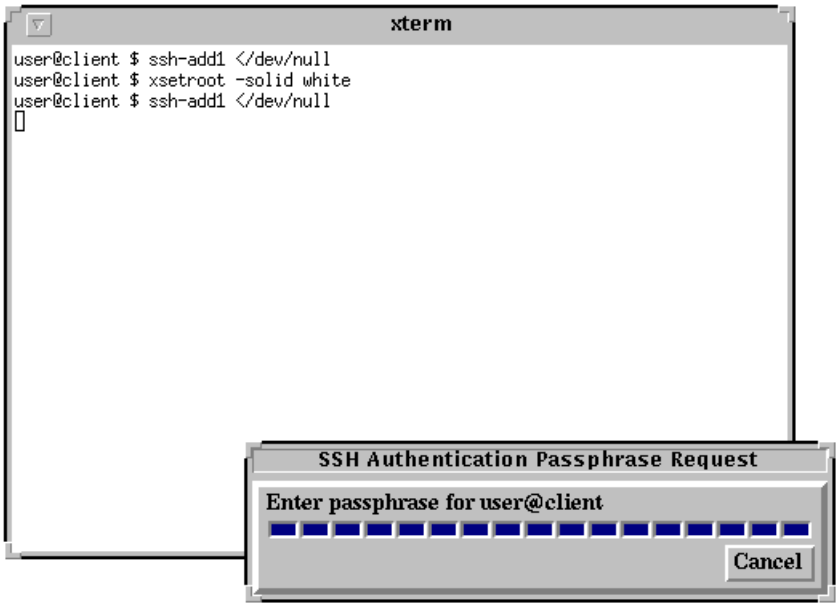



**Abbildung 2-2 ssh-add1 mit grafischem Benutzerinterface**

#### 2.1.4.1.4 Weiterleiten von X-Verbindungen

Wenn clientseitig ein X-Display gesetzt ist, leitet SSH1 automatisch X-Verbindungen durch den verschlüsselten SSH-Kanal weiter, serverseitig wird ein Pseudo-X-Display erzeugt, mit dem X-Clients kommunizieren können:

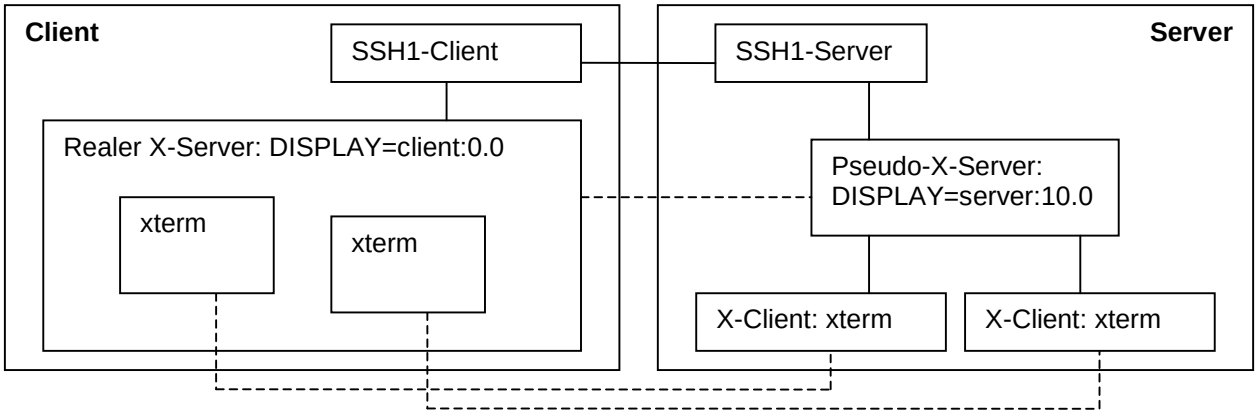



**Abbildung 2-3 Weiterleiten von X-Verbindungen**

X-spezifische Authorisierungs-Cookies (*man xauth ↵*) des Clients werden aus Sicherheitsgründen nicht serverseitig benutzt; serverseitig werden von SSH1-Server neue Cookies erzeugt und an den Client weitergereicht.

Ein typischer Aufruf eines Clients:

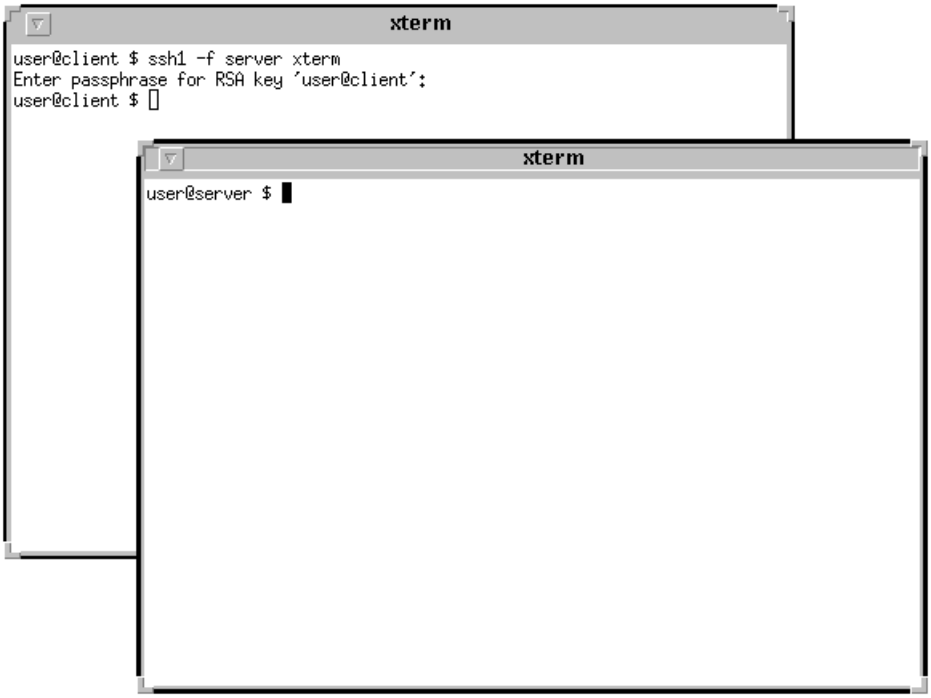



**Abbildung 2-4 Aufruf eines X-Clients über SSH1**

Der Parameter „–f“ weist den SSH1-Client an, nach erfolgreicher Authentifizierung als Hintergrundprozeß zu arbeiten.

Das Weiterleiten von X-Verbindungen über SSH1 kann mit der Option „–x“ an ssh1 unterbunden werden.

#### 2.1.4.1.5 Weiterleiten von sonstigen TCP-Verbindungen

Es existieren zwei Möglichkeiten des Weiterleitens von TCP-Verbindungen:

##### *2.1.4.1.5.1 local forwarding*

Der Aufruf

`user@client $` *ssh1 –L port:host:hostport server* ↵

leitet Anfragen an den TCP-Port *port* der lokalen Maschine durch den gesicherten Kanal an den TCP-Port *hostport* des Servers *host* weiter:

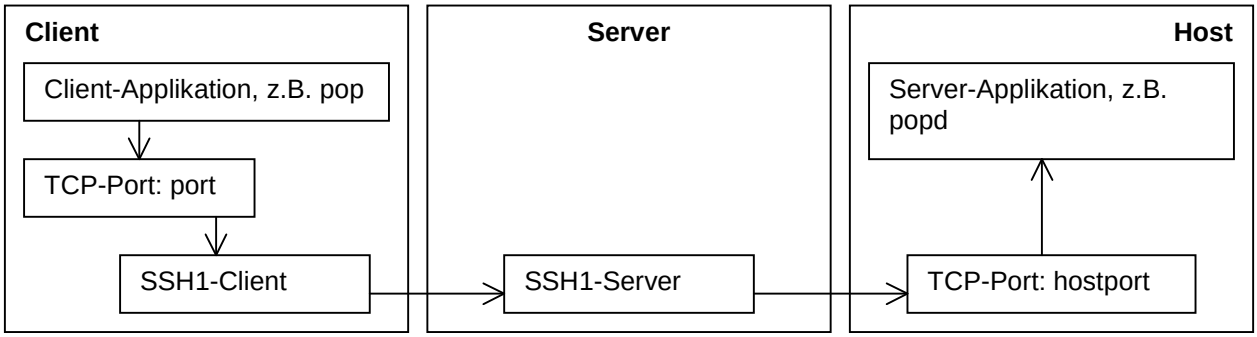



**Abbildung 2-5 Weiterleiten von TCP-Verbindungen: local forwarding I**

Hierbei ist die erzeugte TCP-Verbindung zwischen Server und Host nicht durch das SSH1-Protokoll gesichert; idealerweise sind Server und Applikationshost identisch:

`user@client $` *ssh1 –L port:server:serverport server* ↵

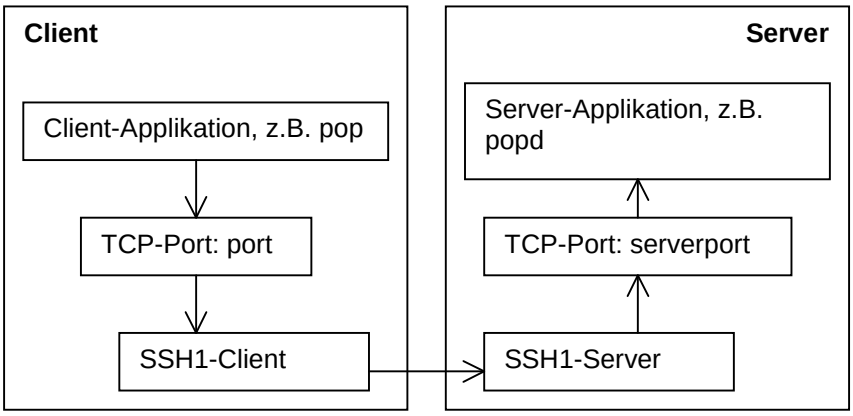



**Abbildung 2-6 Weiterleiten von TCP-Verbindungen: local forwarding II**

Um clientseitig einen privilegierten Port (TCP-Ports unter 1024) zu erzeugen, muß der Aufrufer des SSH1-Clients root oder ein root-äquivatlenter Benutzer sein.
Auf den auf dem Client erzeugten TCP-Port haben standardmäßig nur Applikationen des Clients Zugriff, diese Einschränkung läßt sich außer Kraft setzen durch die Option „–g“ des SSH1-Clients; dies ist jedoch mit einem Sicherheitsrisiko verbunden.

Anwendungsbeispiele für local forwarding:

- Senden von E-Mail mit dem SMTP-Protokoll
  Der Aufruf
  `user@client $` *ssh1 –L 25:server:25 server* ↵
  leitet Anfragen an den SMTP-Port des Clients auf den SMTP-Port des Servers weiter. Es wird vorausgesetzt, daß es sich bei *server* zugleich auch um die Maschine handelt, die ausgehende Mail verarbeitet.
  Im Mailprogramm (hier Netscape Navigator Version 3.01Gold) ist unter dem Menüpunkt „Options/Mail and News Preferences/Server“ als SMTP-Server „localhost“ anzugeben:


**Abbildung 2-7 Weiterleiten von TCP-Verbindungen: SMTP**

Bei Mailprogrammen, die die Angabe eines Ports beim SMTP-Server zulassen, kann der lokale Port auch anders gewählt werden.

- Abholen von E-Mail mit dem POP3-Protokoll
  Der Aufruf
  `user@client $` *ssh1 –L 110:server:110 server* ↵
  leitet Anfragen an den POP3-Port des Clients auf den POP3-Port des Servers weiter. Es wird vorausgesetzt, daß es sich bei *server* zugleich auch um die Maschine handelt, die eingehende Mail verarbeitet.
  Im Mailprogramm (hier Netscape Navigator Version 3.01Gold) ist unter dem Menüpunkt „Options/Mail and News Preferences/Server“ als POP3-Server „localhost“ anzugeben:


**Abbildung 2-8 Weiterleiten von TCP-Verbindungen: POP3**

Bei Mailprogrammen, die die Angabe eines Ports beim POP-Server zulassen, kann der lokale Port auch anders gewählt werden.

- Filetransfers mit FTP
  Der Aufruf

  `user@client $` *ssh1 –L 8021:server:21 server* ↵

  leitet Anfragen an den TCP-Port 8021 des Clients auf den ftp-Port des Servers weiter. Es wird vorausgesetzt, daß es sich bei *server* zugleich auch um die Maschine handelt, auf der ein ftp-Server läuft.
  Beim ftp-Client ist zu beachten, daß dieser den passive-Mode unterstützt; dieser Modus gewährleistet auch Datenrückverbindungen über den gleichen TCP-Port.
  Um einen Browser als ftp-Client einzusetzen, ist es notwendig den Proxy-Zugriff auf localhost zu unterbinden. Dies geschieht z.B. bei Netscape Navigator Version 3.01Gold unter dem Menüpunkt „Options/Network Preferences/Proxies/Manual Proxy Configuration“:

  **Abbildung 2-9 Weiterleiten von TCP-Verbindungen: FTP I**

  Der Zugriff auf den ftp-Server vollzieht sich wie folgt:

  **Abbildung 2-10 Weiterleiten von TCP-Verbindungen: FTP II**

  Nach erfolgreichem Konnektieren des ftp-Servers durch die gesicherte SSH1-Verbindung hindurch erwartet der ftp-Server das Passwort des Benutzers:

  **Abbildung 2-11 Weiterleiten von TCP-Verbindungen: FTP III**

  Das eingegebene Kennwort wird verschlüsselt über die SSH1-Verbindung übertragen.

  Auch ein kommandozeilenorientierter ftp-Client ist benutzbar, vorausgesetzt er unterstützt den passive-Mode:

  ```
  user@client $ ftp localhost 8021 ↵
  Connected to localhost.
  220 server FTP server ready.
  Name (localhost:user): user ↵
  331 Password required for user.
  Password: <Passwort> ↵
  230 User user logged in.
  ftp> passive ↵
  Passive mode on.
  ftp> binary ↵
  200 Type set to I.
  ftp> get file1.tgz ↵
  ...
  ftp> put file2.tgz ↵
  ...
  ftp> quit ↵
  221 Goodbye.
  user@client $
  ```

  Ein ftp-Client für Solaris, der den passive-Mode unterstützt, ist unter ftp://ftp.cert.dfn.de/pub/firewalls/software/ftp-PASV im Quellcode zu finden.

- Aktualisieren von html-Seiten
  Der Aufruf

  `user@client $` *ssh1 –L 8080:server:80 server* ↵

  leitet Anfragen an den TCP-Port 8080 des Clients auf den http-Port des Servers weiter. Es wird vorausgesetzt, daß es sich bei *server* zugleich auch um die Maschine handelt, auf der ein WWW-Server läuft. Dieser WWW-Server muß die put-Erweiterung beinhalten, um html-Seiten per http-Request aktualisieren zu können.
  Es ist notwendig den Proxy-Zugriff auf localhost zu unterbinden. Dies geschieht z.B. bei Netscape Navigator Version 3.01Gold unter dem Menüpunkt „Options/Network Preferences/Proxies/Manual Proxy Configuration“:


**Abbildung 2-12 Weiterleiten von TCP-Verbindungen: HTTP I**

  Wenn der Benutzer bislang durch die URL http://server/~user auf Dokumente zugriff, kann er es jetzt wie folgt tun:


**Abbildung 2-13 Weiterleiten von TCP-Verbindungen: HTTP II**

  Im html-Editor-Programm (hier Netscape Navigator Version 3.01 Gold) sind zum Publizieren von html-Seiten unter dem Menüpunkt „Options/Editor Preferences/Publish“ folgende Einstellungen vorzunehmen:


**Abbildung 2-14 Weiterleiten von TCP-Verbindungen: HTTP III**

### *2.1.4.1.5.2 remote forwarding*

Der Aufruf

`user@client $` *ssh1 –R port:host:hostport server* ↵

leitet Anfragen an den TCP-Port *port* des Servers durch den gesicherten Kanal an den TCP-Port *hostport* des Servers *host* weiter:

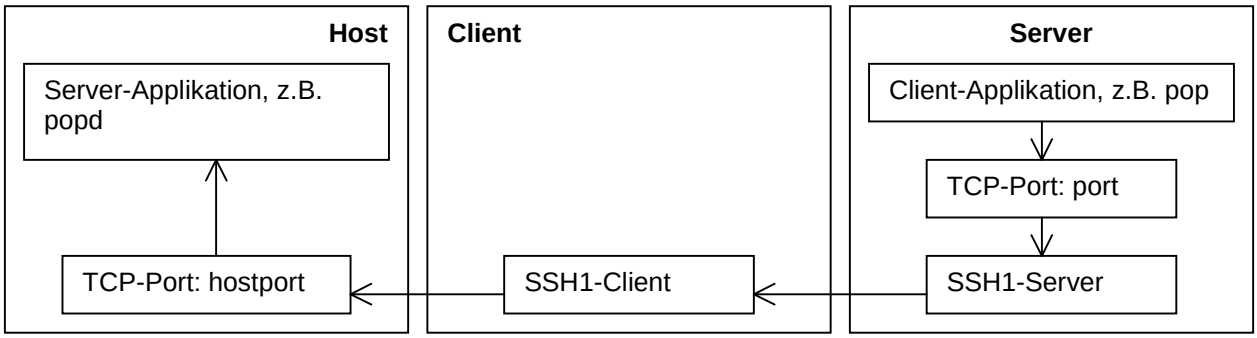



**Abbildung 2-15 Weiterleiten von TCP-Verbindungen: remote forwarding I**

Hierbei ist die erzeugte TCP-Verbindung zwischen Client und Host nicht durch das SSH1-Protokoll gesichert; idealerweise sind Client und Applikationshost identisch:

`user@client $` *ssh1 –R port:client:clientport server* ↵

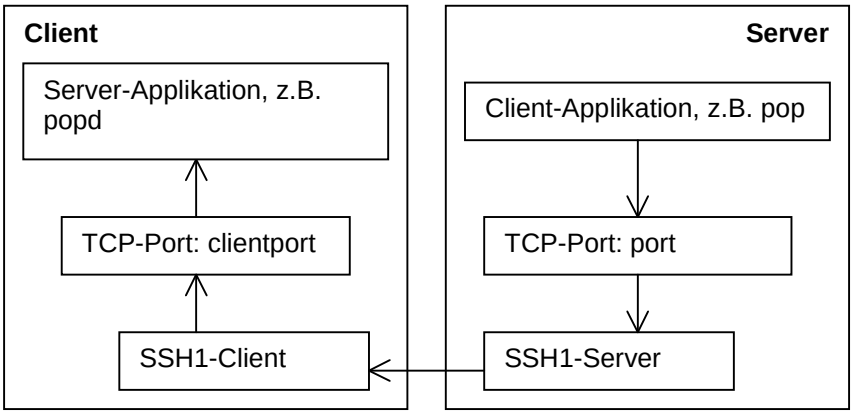



**Abbildung 2-16 Weiterleiten von TCP-Verbindungen: remote forwarding II**

Um serverseitig einen privilegierten Port (TCP-Ports unter 1024) zu erzeugen, muß der Aufrufer des SSH1-Clients serverseitig root oder root-äquivalent sein.

#### 2.1.4.1.6 Kommandooptionen von SSH1

Der SSH1-Client ssh1 besitzt einige nützliche Optionen:

- -a
  Deaktiviert das Weiterleiten des Authentifizierungsagent ssh-agent1 über Verbindungen hinweg, siehe auch: **2.1.4.1.3 Benutzen des Authentifizierungs-Agent** (Seite 10)
- -c [idea|des|3des|blowfish|arcfour|none]
  Gibt das zu benutzende Verschlüsselungsverfahren an, mit dem die SSH-Verbindung verschlüsselt wird; möglich sind: „idea“, „des“, „3des“, „blowfish“, „arcfour“ und „none“, wobei „none“ für keine Verschlüsselung steht.
- -e [ch|^ch|none]
  Gibt das Escape-Zeichen an, mit der sich in einer interaktiven SSH-Sitzung die Verbindung kontrollieren läßt. Standard-Escape-Zeichen ist die Tilde „~“. Das Escape-Zeichen wird nur am Anfang einer Zeile erkannt. Das Escape-Zeichen gefolgt von einem Punkt „.“ schließt die Verbindung, gefolgt von Control-Z hält die Verbindung und wechselt zur lokalen Shell, gefolgt von der Raute „#“ zeigt alle weitergeleiteten TCP-Verbindungen an, gefolgt vom Escape-Zeichen überträgt das Escape-Zeichen selbst.
  Das Setzen des Escape-Zeichens auf „none“ schaltet das Escape-Zeichen aus und gewährleistet eine volltransparente Verbindung.
  Ein Beispiel:

```
user@client $ ssh1 server ↵
Enter passphrase for RSA key 'user@client': <Passphrase> ↵
user@server $ ~^Z

[1]+  Stopped (user)          ssh1 server
user@client $ fg ↵
ssh1 server

user@server $ exit ↵
Connection to server closed.
user@client $
```

- -f
  Bestimmt, daß der SSH1-Client nach erfolgter Authentifizierung als Hintergrundprozeß arbeiten soll. Diese Option beinhaltet die Option „–n“. Mit der Option „–f“ werden üblicherweise X-Clients gestartet: **2.1.4.1.4 Weiterleiten von X-Verbindungen** (Seite 12).
- -i identity_file
  Gibt den zu benutzenden Private-Key bei Public-Key-Authentifizierung an. Standardwert ist hier $HOME/.ssh/identity. Es können mehrere Private-Key's angegeben werden, die nacheinander benutzt werden, bis eine erfolgreiche Authentifizierung erfolgt ist.

- -k
  Schaltet die Weiterleitung von Kerberos-Tickets aus. Diese Option ist nur bei Benutzung eines Kerberos-Servers zur Authentifizierung relevant.
- -l login_name
  Mit dieser Option läßt sich der Anmeldename an den Server übergeben; das ist dann sinnvoll, wenn der Anmeldename clientseitig nicht der Anmeldename serverseitig ist.
- -n
  Liest anstatt von Standardeingabe von /dev/null. Diese Option ist zwingend, wenn der SSH1-Client in den Hintergrund gestellt wird (siehe auch Option „–f“). Sinnvoll ist dies zum Ausführen von entfernten X-Clients (bei dieser Option wartet der Client nicht auf eine Passphrase oder ein Passwort).
  Ein Anwendungsbeispiel:

```
user@client $ eval `ssh-agent1 –s` ↵
Agent pid 1234
user@client $ ssh-add1 ↵
Need passphrase for /home/user/.ssh/identity (user@client).
Enter passphrase: <Passphrase> ↵
Identity added: /home/user/.ssh/identity (user@client)
user@client $ ssh1 –n server xterm ↵
user@server $
```

- -o ’option‘
  Hier können Optionen im Konfigurationsdatei-Format übergeben werden, für die keine Kommandozeilenoption existiert. Näheres zum Konfigurationsdatei-Format in **2.1.4.1.7 Hinterlegen von Standardeinstellungen für verschiedene Server** (Seite 19).

- -p port
  Gibt den TCP-Port an, auf dem der anzusprechende SSH1-Server auf eingehende Verbindungen wartet. Standardwert ist TCP-Port 22. Der Server kann bei Bedarf auch mit einem anderen TCP-Port betrieben werden: **2.2.8.1.1 Konfiguration des Servers** (Seite 35).
- -q
  Alle Warnungen werden unterdrückt, nur kritische Fehler werden angezeigt.
- -P
  Der Client benutzt keinen privilegierten TCP-Port (TCP-Port 1 bis 1024) zum Verbindungsaufbau. Bei Benutzung dieser Option kann rhosts- oder Public-Key-basierte rhosts-Authentifizierung nicht eingesetzt werden; manche Firewalls lassen jedoch ausgehend nur Pakete passieren, die nicht von einem privilegierten Port stammen.
- -t
  Erzwingt die Bereitsstellung eines Pseudo-tty’s zur interaktiven Nutzung auf dem Server auch wenn ein Kommando zur Ausführung an den Client übergeben wird.
- -v
  Bei Angabe dieser Option werden ausführliche Meldungen ausgegeben, dies ist nützlich zur Fehlererkennung und –behebung.
- -V
  Gibt die Versionsnummer und Protokollversion des Clients aus.
- -g
  Erlaubt es anderen Maschinen auf weitergeleitete Verbindungen des Clients zuzugreifen: **2.1.4.1.5.1 local forwarding** (Seite 13)
- -x
  Schaltet das Weiterleiten von X-Verbindungen aus: **2.1.4.1.4 Weiterleiten von X-Verbindungen** (Seite 12)
- -C
  Schaltet die Datenkomprimierung ein; die eingesetzte Datenkomprimierung ist GNU-zip kompatibel. Diese Option ist nützlich bei schmalbandigen Verbindungen, z.B. über Modems. Es werden auch weitergeleitete X-Verbindungen und weitergeleitete TCP-Verbindungen komprimiert.
- -L port:host:hostport
  leitet Anfragen an den TCP-Port *port* der lokalen Maschine durch den gesicherten Kanal an den TCP-Port *hostport* des Servers *host* weiter: **2.1.4.1.5.1 local forwarding** (Seite 13)
- -R port:host:hostport
  leitet Anfragen an den TCP-Port *port* des Servers durch den gesicherten Kanal an den TCP-Port *hostport* des Servers *host* weiter: **2.1.4.1.5.2 remote forwarding** (Seite 16)

#### 2.1.4.1.7 Hinterlegen von Standardeinstellungen für verschiedene Server

Der SSH1-Client bezieht seine Konfigurationsinformationen bezüglich der aufzubauenden Verbindung von drei Quellen (in dieser Reihenfolge):

- von der Kommandozeile,
- aus der Benutzerkonfigurationsdatei $HOME/.ssh/config
- aus der globalen Konfigurationsdatei /etc/ssh_config.

Der jeweils zuerst eingelesene Parameter ist dabei gültig.
Die Konfigurationsdatei hat folgendes Format:

- Leere Zeilen und Zeilen, die mit der Raute „#“ beginnen, werden ignoriert
- Zeilen haben grundsätzlich den Aufbau „Schlüsselwort = Parameter“ bzw. „Schlüsselwort Parameter“

Folgende Schlüsselwörter sind gültig:

- Host
  Zeilen, die mit dem Schlüsselwort Host beginnen, lassen nachfolgende Zeilen bis zur nächsten Zeile, die mit dem Schlüsselwort Host beginnt, nur für diese Hosts gelten, auf welche die Hosts-Zeile paßt. Als Parameter sind auch Einzel- und Allexistenzquantor („?“ und „*“) erlaubt. Der Allexistenzquantor „*“ alleine gibt an, daß es sich nachfolgend um globale Standardeinstellungen für den SSH1-Client handelt. Wichtig dabei ist, daß Einstellungen für spezifische Hosts am Anfang und globale Einstellungen am Ende der Konfigurationsdatei stehen müssen.
- Batchmode
  Wenn dieses Schlüsselwort vom Parameter „yes“ gefolgt ist, wird vom Client kein Passwort bzw. keine Passphrase eingelesen; die Authentifizierung muß also über den Authentifizierungs-Agent erfolgen. Gültige Parameter sind „yes“ und „no“.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-n“.
- Cipher
  Gibt das zu benutzende Verschlüsselungsverfahren an, mit dem die SSH-Verbindung verschlüsselt wird; möglich sind: „idea“, „des“, „3des“, „blowfish“, „arcfour“ und „none“, wobei „none“ für keine Verschlüsselung steht.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-c“.
- ClearAllForwardings
  Setzt alle Verbindungsweiterleitungen (X-Verbindungen, local forwarding, remote forwarding), die für den betreffenden Host auf der Kommandozeile oder in Konfigurationsdateien angegeben sind, zurück. Das ist sinnvoll, wenn eine zweite Verbindung zu dem betreffenden Host aufgebaut werden soll.
- Compression
  Wenn dieses Schlüsselwort vom Parameter „yes“ gefolgt ist, wird die Datenkomprimierung eingeschaltet; die eingesetzte Datenkomprimierung ist GNU-zip kompatibel. Dies ist nützlich bei schmalbandigen Verbindungen, z.B. über Modems. Es werden auch weitergeleitete X-Verbindungen und weitergeleitete TCP-Verbindungen komprimiert. Gültige Parameter sind „yes“ und „no“.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-C“.
- CompressionLevel
  Gibt bei eingeschalteter Datenkomprimierung die Kompressionsstufe an. Der Parameter muß eine Zahl zwischen 1 (schnell, aber wenig komprimiert) und 9 (langsam, aber sehr gut komprimiert) sein. Der Standardwert ist 6.
- ConnectionAttempts
  Gibt die Anzahl der Verbindungsversuche (einer pro Sekunde) an, die vom Client unternommen werden sollen bis ggf. ein Verbindungsversuch mit rsh unternommen werden soll (vgl. Schlüsselwort „FallBackToRsh“). Der Parameter muß ein ganzzahliger positiver Wert sein.
- EscapeChar
  Gibt das Escape-Zeichen an, mit der sich in einer interaktiven SSH-Sitzung die Verbindung kontrollieren läßt. Der Paramter sollte ein einzelnes Zeichen, „^“ gefolgt von einem einzelnen Zeichen oder „none“ sein.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-e“.
- FallBackToRsh
  Gibt an, ob nach erfolglosen Verbindungsversuchen (i.A. weil auf dem Server z.Zt. kein SSH1-Dämon auf eingehende Verbindungen wartet) automatisch ein Verbindungsaufbau mit rsh versucht werden soll. Gültige Parameter sind „yes“ und „no“.

- ForwardAgent
  Deaktiviert das Weiterleiten des Authentifizierungsagent ssh-agent1 über Verbindungen hinweg. Gültige Parameter sind „yes“ und „no“.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-a“.
- ForwardX11
  Bestimmt, ob X-Verbindungen weitergeleitet werden sollen oder nicht. Gültige Parameter sind „yes“ und „no“.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-a“.
- GatewayPorts
  Erlaubt es anderen Maschinen auf weitergeleitete Verbindungen des Clients zuzugreifen. Gültige Parameter sind „yes“ und „no“.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-g“.
- GlobalKnownHostsFile
  Gibt an, daß für die Überprüfung der öffentlichen Serverschlüssel anstatt der Datei /etc/ssh_known_hosts eine andere Datei verwendet werden soll.
- HostName
  Spezifiziert den echten Hostnamen zu dem die Verbindung aufgebaut werden soll. Diese Option ist nützlich zum Verwenden von Kürzeln für die Hostnamen. Der Standardwert ist der Hostname, der in der Kommandozeile verwendet wird. Numerische IP-Adressen (z.B. 192.168.25.2) sind auch erlaubt.
  Ein Beispiel für die Verwendung von Kürzeln für Hostnamen:
  Die Konfigurationsdatei $HOME/.ssh/config hat folgenden Inhalt:
  ```
  Host s1
    Hostname server1.subdomain.domain.net
  ```
  Dann kann auf diese Weise eine Verbingung aufgebaut werden:
  ```
  user@client $ ssh1 s1 ↵
  ```
- IdentityFile
  Gibt den zu benutzenden Private-Key bei Public-Key-Authentifizierung an. Standardwert ist hier $HOME/.ssh/identity. Es können mehrere Private-Key’s angegeben werden, die nacheinander benutzt werden, bis eine erfolgreiche Authentifizierung erfolgt ist. Der angegebene Dateiname kann das Tildezeichen „~“ zur Referenzierung des Benutzerverzeichnisses enthalten: „~user/.ssh/identity_host2“ ist also gleichbedeutend mit „$HOME/.ssh/identity_host2“
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-i“.
- KeepAlive
  Gibt an, ob der ssh-Client KeepAlive-Pakete an den Server schickt, d.h. Pakete zur Prüfung der Verbindung und Verfügbarkeit des Servers.
  Der Standardwert ist „yes“; der Client terminiert also Verbindungen mit nicht beantworteten KeepAlive-Paketen. Um KeepAlive-Pakete auszuschalten, muß dies sowohl in der Serverkonfigurationsdatei (/etc/sshd_config ) als auch in der Clientkonfigurationsdatei ($HOME/.ssh/config) geschehen. Gültige Parameter sind „yes“ und „no“.
- KerberosAuthentication
  Gibt an, ob Kerberos V5-Authentifizierung benutzt werden soll. Gültige Parameter sind „yes“ und „no“.
- KerberosTgtPassing
  Schaltet die Weiterleitung von Kerberos-Tickets ein oder aus. Diese Option ist nur bei Benutzung eines Kerberos-Servers zur Authentifizierung relevant. Gültige Parameter sind „yes“ und „no“.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-k“.
- LocalForward
  Die Zeile „LocalForward port host:hostport“ leitet Anfragen an den TCP-Port *port* der lokalen Maschine durch den gesicherten Kanal an den TCP-Port *hostport* des Servers *host* weiter. Mehrere LocalForward-Direktiven sind möglich.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-L“.
- NumberOfPasswordPrompts
  Gibt die Anzahl der möglichen Anmeldeversuche an, wenn Passwort-Authentifikation benutzt wird. Der Server begrenzt die Anzahl der möglichen Anmeldeversuche bei Passwort-Authentifikation standardmäßig auf 3 (dies ist konfigurierbar); Werte größer als die serverseitig definierten werden ignoriert. Der Parameter muß ein ganzzahliger positiver Wert sein. Standardwert ist 1.
- PasswordAuthentication
  Gibt an, ob Passwort-Authentifizierung genutzt wird oder nicht. Gültige Parameter sind „yes“ und „no“.
- PasswordPromptHost
  Gibt das Aussehen der Passwort-Eingabeaufforderung vor, wenn Passwort-Authentifikation benutzt wird. Wenn dieses Schlüsselwort vom Parameter „yes“ gefolgt ist, wird der Servername angezeigt. Gültige Parameter sind „yes“ und „no“.

Die Einstellung „PasswordPromptHost yes“ hat folgendes Aussehen:

```
user@client $ ssh1 server ↵
user@server's password:
```

Die Einstellung „PasswordPromptHost no“ hat folgendes Aussehen:

```
user@client $ ssh1 server ↵
users's password:
```

- PasswordPromptLogin
  Gibt das Aussehen der Passwort-Eingabeaufforderung vor, wenn Passwort-Authentifikation benutzt wird. Wenn dieses Schlüsselwort vom Parameter „yes“ gefolgt ist, wird der Serverbenutzername angezeigt. Gültige Parameter sind „yes“ und „no“.
  Die Einstellung „PasswordPromptLogin yes“ hat folgendes Aussehen:

```
user@client $ ssh1 server ↵
user@server's password:
```

  Die Einstellung „PasswordPromptLogin no“ hat folgendes Aussehen:

```
user@client $ ssh1 server ↵
server password:
```

  Die Einstellung „PasswordPromptLogin no“ in Verbindung mit „PasswordPromptHost no“ hat folgendes Aussehen:

```
user@client $ ssh1 server ↵
Password:
```

  Die beiden Schlüsselwörter „PasswordPromptLogin“ und „PasswordPromptHost“ sind nützlich, wenn von anderen Programmen (z.B. vom UUCP-Paket) bestimmte Login-Sequenzen vorgegeben sind.
- Port
  Gibt den TCP-Port an, an dem der SSH1-Server auf eingehende Verbindungen wartet. Standardwert ist 22.
- ProxyCommand
  Gibt ein Kommando an, daß vom Client zum Verbindungsaufbau zum Server benutzt wird. Das Kommando wird von /bin/sh ausgeführt. Im Kommando wird „%h“ durch den Servernamen und „%p“ durch die Portnummer des zu konnektierenden Servers ersetzt. Das Kommando sollte die Daten des Servers über Standardausgabe ausgeben und von Standardeingabe eingelesene Daten an den Server weiterleiten.
  Diese Option ist nützlich, wenn der Client über ein Applikationsgateway mit einem Telnetproxy kommunizieren muß.
  Ein Beispiel:
  ProxyCommand /usr/local/bin/telnetproxy %h %p

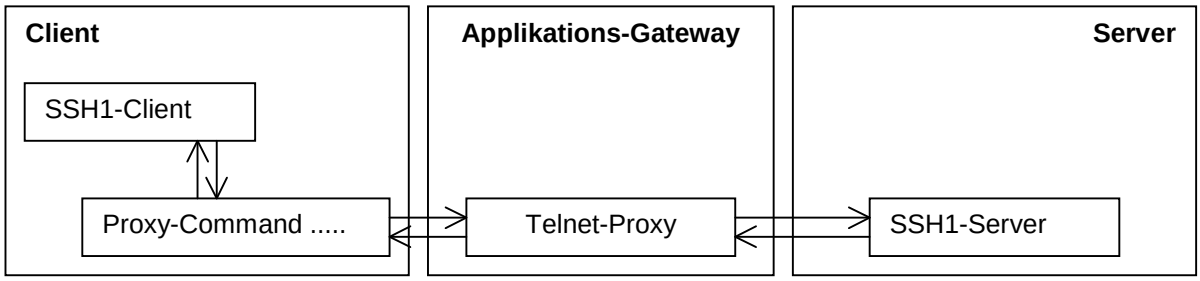



**Abbildung 17 Benutzung von Applikationsgateways**

- RemoteForward
  Die Zeile „RemoteForward port host:hostport“ leitet Anfragen an den TCP-Port *port* des Servers durch den gesicherten Kanal an den TCP-Port *hostport* des Servers *host* weiter. Mehrere RemoteForward-Direktiven sind möglich.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-R“.
- RhostsAuthentication
  Gibt an, ob rhosts-basierende Authentifizierung versucht werden soll. Der Server kann dies bei entsprechender Konfiguration ablehnen. Gültige Parameter sind „yes“ und „no“.
- RhostsRSAAuthentication
  Gibt an, ob rhosts-basierende Authentifizierung in Verbindung mit Public-Key-Host-Authentifizierung versucht werden soll. Der Server kann dies bei entsprechender Konfiguration ablehnen. Gültige Parameter sind „yes“ und „no“.

- RSAAuthentication
  Gibt an, ob Public-Key-Authentifizierung versucht werden soll. Der Server kann dies bei entsprechender Konfiguration ablehnen. Public-Key-Authentifizierung wird nur bei Vorhandensein eines privaten Schlüssels versucht (vgl. **2.1.2.1 Erzeugen von Schlüssel-Paaren für SSH1**, Seite 7). Gültige Parameter sind „yes“ und „no“.
- StrictHostKeyChecking
  Gibt an, wie bei Public-Key-Authentifizierung mit öffentlichen Serverschlüsseln verfahren wird. Wenn der Parameter auf „yes“ gesetzt ist, wird nie automatisch ein neuer öffentlicher Serverschlüssel in die Datei $HOME/.ssh/known_hosts aufgenommen und die Verbindung zum Server wird nicht aufrechterhalten. Ist der Parameter auf „no“ gesetzt wird automatisch ein neuer Serverschlüssel in die Datei $HOME/.ssh/known_hosts aufgenommen. Ist der Parameter auf „ask“ gesetzt wird der Benutzer fallweise gefragt, wie zu verfahren ist. Der Standardwert ist „ask“. Die größtmögliche Sicherheit bietet der Parameter „yes“, da der Benutzer hier gezwungen wird manuell den betreffenden öffentlichen Serverschlüssel einzutragen.
- TISAuthentication
  Gibt an, ob Authentifizierung durch den Authentifizierungsserver aus dem TIS-Firewallkit versucht werden soll. Diese Option ergibt nur Sinn, wenn der SSH1-Server mit Unterstützung dieser Option übersetzt wurde. Weitere Informationen über das TIS-Firewallkit sind unter der URL ftp://ftp.tis.com/pub/firewalls/toolkit/LICENSE verfügbar. Gültige Parameter sind „yes“ und „no“.
- UsePrivilegedPort
  Gibt an, ob der Client einen priveligierten TCP-Port zum Verbindungsaufbau nutzt. Der Standardwert ist „yes“, wenn rhosts-basierte Authentifizierung eingesetzt wird. Gültige Parameter sind „yes“ und „no“.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-P“.
- User
  Mit diesem Schlüsselwort läßt sich der Anmeldename an den Server übergeben, was nützlich ist, wenn auf verschiedenen Maschinen verschiedene Anmeldenamen benutzt werden.
  Dieses Schlüsselwort entspricht der Kommandozeilenoption „-l“.
- UserKnownHostsFile
  Gibt an, daß für die Überprüfung der öffentlichen Serverschlüssel anstatt der Datei $HOME/.ssh//known_hosts eine andere Datei verwendet werden soll.
- UseRsh
  Gibt an, daß bei diesem Server ausschließlich rsh oder rlogin verwandt werden soll. Alle anderen Schlüsselwörter außer „HostName“ werden ignoriert, wenn dieses Schlüsselwort verwandt wird. Gültige Parameter sind „yes“ und „no“.
- XAuthLocation
  Gibt den Pfad zum Programm „xauth“ an, daß beim Weiterleiten von X-Verbindungen benötigt wird. Der Standardwert wird beim Übersetzen des Clients festgelegt.

Ein Beispiel für den Einsatz von Konfigurationsdateien:

Man möchte auf die Maschine server1.another.net zugreifen, dabei soll der lokale TCP-Port 1025 auf den TCP-Port 25 des Servers wegen Mailauslieferung weitergeleitet und zur RSA-Authentifizierung die Datei /home/user/.ssh/identity_server1 benutzt werden.
Der Aufruf ohne spezielle Konfigurationsdatei würde wie folgt aussehen:

```
user@client $ ssh1 –L 1025:server1.another.net:25 –i /home/user/.ssh/identity_server1
                                                              server1.another.net ↵
```

Durch Eintragen der Optionen in die Konfigurationsdatei läßt sich die Kommandozeile beträchtlich verkürzen.
Inhalt der Konfigurationsdatei $HOME/.ssh/config:

```
Host s1
  HostName       server1.another.net
  IdentityFile   /home/user/.ssh/identity_server1
  LocalForward   1025 server1.another.net:25
```

Der Aufruf mit dieser Konfigurationsdatei würde wie folgt aussehen:

```
user@client $ ssh1 s1 ↵
```

#### 2.1.4.1.8 Kopieren von Dateien: scp1

Mit diesem Kommando wird es ermöglicht, Dateien zwischen Maschinen zu kopieren. Das Kommando setzt auf das SSH1-Protokoll auf, d.h. es gelten die gleichen Sicherheitsmechanismen wie bei der Benutzung des ssh1-Clients. Die Funktionalität ist eine Obermenge des Kommandos rcp.

Ein Anwendungsbeispiel:

```
user@client $ scp1 /home/user/datei1 user@server:/home/user/datei1 ↵
Enter passphrase for RSA key 'user@client': <Passphrase> ↵
user@server $
```

Die allgemeine Syntax des Kommandos lautet:

scp1 [Optionen] [[user@]host1:]filename1... [[user@]host2:]filename2

Es können mit scp1 sowohl mehrere lokale Dateien auf einen Server kopiert werden als auch mehrere Dateien des Servers auf die lokale Maschine. Des weiteren ist auch ein Kopieren von Dateien von einem Server auf einen anderen Server mit scp1 möglich.

Die verwendbaren Optionen von scp1 sind im Folgenden dargestellt:

- -a
  Schaltet die Ausgabe von Datendurchsatzstatistiken bei der Übertragung jeder Datei ein.
  Das Setzen der Umgebungsvariable SSH_ALL_SCP_STATS ist äquivalent zu dieser Option.
- -A
  Schaltet die Ausgabe von Datendurchsatzstatistiken bei der Übertragung jeder Datei aus.
  Das Setzen der Umgebungsvariable SSH_NO_ALL_SCP_STATS ist äquivalent zu dieser Option.
- -c [idea|des|3des|blowfish|arcfour|none]
  Gibt das zu benutzende Verschlüsselungsverfahren an, mit dem die zugrunde liegende SSH-Verbindung verschlüsselt wird; möglich sind: „idea“, „des“, „3des“, „blowfish“, „arcfour“ und „none“, wobei „none“ für keine Verschlüsselung steht.
  Diese Option wird direkt an SSH übergeben.
- -i identity_file
  Gibt den zu benutzenden Private-Key bei Public-Key-Authentifizierung an. Standardwert ist hier $HOME/.ssh/identity.
  Diese Option wird direkt an SSH übergeben.
- -L
  Bei Benutzung dieser Option benutzt scp1 keinen privilegierten TCP-Port (TCP-Port 1 bis 1024) zum Verbindungsaufbau. Bei Benutzung dieser Option kann rhosts- oder Public-Key-basierte rhosts-Authentifizierung nicht eingesetzt werden; manche Firewalls lassen jedoch ausgehend nur Pakete passieren, die nicht von einem privilegierten Port stammen.
- -o 'option'
  Hier können Optionen im Konfigurationsdatei-Format übergeben werden, für die keine SSH1-Kommandozeilenoption existiert.
  Näheres zum Konfigurationsdatei-Format in **2.1.4.1.7 Hinterlegen von Standardeinstellungen für verschiedene Server** (Seite 19).
  Diese Option wird direkt an SSH übergeben.
  Eine häufig benutzte Anwendung dieser Option besteht in der Übergabe der Kompressionsstufe der zu übertragenden Daten:
  *scp1 -o 'CompressionLevel 9' /home/user/datei1 user@server:/home/user/datei1 ↵*
  ermöglicht das Kopieren mit höchster Kompressionsstufe.
- -p
  Bei Benutzung dieser Option werden die letzte Zugriffszeit, die Änderungszeit sowie die Dateirechte der Originaldatei(en) auf die Kopie(n) übertragen.
- -q
  Schaltet die Ausgabe von Gesamtdatendurchsatzstatistiken nach der Übertragung aller Dateien aus.
  Das Setzen der Umgebungsvariable SSH_NO_SCP_STATS ist äquivalent zu dieser Option.
- -Q
  Schaltet die Ausgabe von Gesamtdatendurchsatzstatistiken nach der Übertragung aller Dateien ein.
  Das Setzen der Umgebungsvariable SSH_SCP_STATS ist äquivalent zu dieser Option.
- -r
  Kopiert rekursiv ganze Verzeichnisse
- -v
  Bei Angabe dieser Option sowohl von scp1 als auch von ssh1 ausführliche Meldungen ausgegeben, dies ist nützlich zur Fehlererkennung und –behebung.
- -B
  schaltet den Batchmodus ein. In diesem Modus fragt scp1 weder nach Passwörtern noch nach Passphrasen. Diese müssen ggf. vor dem Aufruf von scp1 durch den Authentifizierungs-Agent ssh-agent1 bereitgestellt werden.

- -C
  Schaltet die Datenkomprimierung ein; die eingesetzte Datenkomprimierung ist GNU-zip kompatibel. Diese Option ist nützlich bei schmalbandigen Verbindungen, z.B. über Modems.
  Diese Option wird direkt an SSH übergeben.
- -P port
  Gibt den TCP-Port an, auf dem der anzusprechende SSH1-Server auf eingehende Verbindungen wartet. Standardwert ist TCP-Port 22.
- -S SSH1-Pfad
  Hier ist der Pfad zum ssh1-Client anzugeben, wenn dessen Verzeichnis nicht im Suchpfad enthalten ist.

Das Kommando scp1 kann keine Gerätetreiberdateien kopieren, wie es z.B. notwendig ist, um eine Maschine komplett auf eine andere Maschine zu spiegeln.
Diese Einschränkung kann durch den Einsatz des Kommandos tar umgangen werden:

```
root@client $ tar –create –file - / | ssh1 server tar –extract –directory / --atime-preserve
                                                --same-owner –preserve –dereference –file - ↵
```

#### 2.1.4.1.9 Benutzte Dateiformate für Public-Key-Authentifizierung

##### *2.1.4.1.9.1 Die Authorisationsdatei authorized_keys*

Die Datei $HOME/.ssh/authorized_keys enthält alle öffentlichen Teile der RSA-Schlüssel, denen Zugriff erlaubt ist.
Die Datei hat folgendes Format:

- Leere Zeilen und Zeilen, die mit der Raute „#“ beginnen, werden ignoriert
- Zeilen haben beinhalten folgende Felder: Optionen, Länge des RSA-Schlüssels, RSA-Schlüssel-Exponent, RSA-Schlüssel-Modulus, Kommentar. Die Felder werden jeweils durch ein Leerzeichen getrennt. Die Angabe des Optionen-Feldes sowie des Komentar-Feldes ist optional.

Folgende Optionen sind möglich:

- from=“Muster-Liste“
  Mit dieser Option kann zusätzlich zur RSA-Authentifizierung ein Muster angegeben werden, auf daß der Hostname des Clients passen muß, Einzel- und Allexistenzquantor („?“ und „*“) sind erlaubt. Durch Voranstellen eines Ausrufungszeichens „!“ läßt sich ein Muster negieren. Clients, die auf ein negiertes Muster passen wird kein Zugriff erlaubt. Diese Option ist auch global für alle Benutzer verfügbar, vgl. **2.2.8.1.1 Konfiguration des Servers** (Seite 35), Option AllowHosts bzw. DenyHosts.
  Beispiel: from=“*.uni-stuttgart.de,!ppp*.dialin.uni-stuttgart.de“
- command=“Kommando“
  Gibt an, daß bei Benutzung dieses Schlüssels das Kommando ausgeführt wird, unerheblich welches Kommando der Client übergeben hat. Das ist nützlich, wenn man einem Client nur eine bestimmte Aktion ausführen lassen möchte.
  Beispiel: command=“/usr/local/sbin/backupstdin“
- environment=“NAME=Wert“
  Damit lassen sich Umgebungsvariablen definieren, die zur Umgebung hinzugefügt werden. Gleichnamige Umgebungsvariablen werden überschrieben. Diese Option kann mehrmals angegeben werden.
  Beispiel: environment=“XSERVERCLIENT=xwin“
- idle-timeout=Zeitspanne
  Legt fest, wie lange die Verbindung unbenutzt sein darf bevor sie geschlossen wird. Als Postfix hinter dem numerischen Wert sind „s“ für Sekunden, „m“ für Minuten, „h“ für Stunden, „d“ für Tage und „w“ für Wochen möglich, kein Postfix ist gleichbedeutend mit Sekunden.
  Diese Option ist auch global für alle Benutzer verfügbar, vgl. **2.2.8.1.1 Konfiguration des Servers** (Seite 35), Option IdleTimeout.
  Beispiel: idle-timeout=12h
- no-port-forwarding
  Verbietet das Weiterleiten von TCP-Verbindungen bei Benutzung dieses RSA-Schlüssels. Das ist z.B. sinnvoll in Verbindung mit der command-Option.
- no-X11-forwarding
  Verbietet das Weiterleiten von X11-Verbindungen bei Benutzung dieses RSA-Schlüssels. Das ist z.B. sinnvoll in Verbindung mit der command-Option.
- no-agent-forwarding
  Verbietet das Benutzen des Authentifizierungs-Agent bei Benutzung dieses RSA-Schlüssels.

- no-pty
  Verhindert die Bereitsstellung eines Pseudo-tty's zur interaktiven Nutzung auf dem Server bei Benutzung dieses RSA-Schlüssels. Das ist z.B. sinnvoll in Verbindung mit der command-Option.

Beispiele:
- uneingeschränke Nutzung des RSA-Schlüssels:
  1024 37 1343842...0201017 user@client
- Nutzung des RSA-Schlüssels nur in Abhängigkeit vom Hostnamen:
  from="*.uni-stuttgart.de,!ppp*.dialin.uni-stuttgart.de" 1024 37 1343842...0201017 user@client
- Nutzung des RSA-Schlüssels nur für Backups:
  command="/usr/local/sbin/backupstdin",no-port-forwarding,no-X11-forwarding, 1024 37 1343842...0201017 user@client
  Der Aufruf beim Client würde dann z.B. wie folgt aussehen:
  `user@client $` tar -cf - $HOME | *ssh1 server* ↵

Mit den verfügbaren Optionen lassen sich in Verbindung mit ppp und einem Pseudo-tty-Redirector mit wenig Aufwand VPN's (Virtual Private Networks) realisieren:

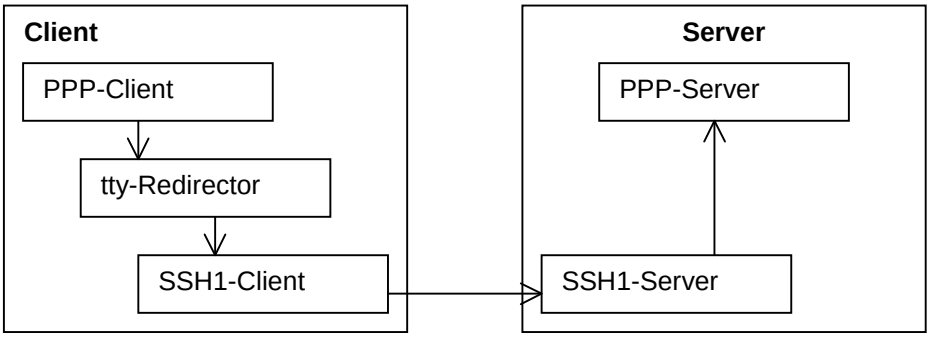



**Abbildung 18 Realisierung von VPN's mit SSH1**

Weiterführende Informationen dazu sind unter http://www.heimhardt.de/htdocs/ssh.html zu finden.

### *2.1.4.1.9.2 Die öffentliche Serverschlüsseldatei known_hosts*

Die Dateien $HOME/.ssh/known_hosts bzw. /etc/ssh_known_hosts enthalten die öffentlichen RSA-Schlüssel der Server, die konnektiert wurden bzw. die als authentisch bekannt sind. Die Dateien dienen dem Zweck Sicherheit über die Authentizität der zu konnektierenden Server zu haben.
Die Datei hat folgendes Format:
- Leere Zeilen und Zeilen, die mit der Raute „#" beginnen, werden ignoriert
- Zeilen haben beinhalten folgende Felder: Hostnamensliste, Länge des RSA-Schlüssels, RSA-Schlüssel-Exponent, RSA-Schlüssel-Modulus, Kommentar. Die Felder werden jeweils durch ein Leerzeichen getrennt. Die Angabe des Komentar-Feldes ist optional.

Die Hostnamensliste ist eine durch Kommata getrennte Liste von Mustern, auf daß der Hostname des Servers passen muß, Einzel- und Allexistenzquantor („?" und „*") sind erlaubt. Durch Voranstellen eines Ausrufungszeichens „!" läßt sich ein Muster negieren. Server, die auf ein negiertes Muster passen werden nicht durch diesen RSA-Schlüssel als authentisch erkannt.

Beispiele:
- ein Host mit einem Namen:
  server1.another.net,192.168.24.2 1024 33 1554504... 2800281 server1.another.net
- ein Host mit mehreren Namen:
  server1.another.net,www.another.net,192.168.24.2 1024 33 1554504... 2800281 server1.another.net

### 2.1.4.2 Benutzen von SSH2

#### 2.1.4.2.1 Interaktives Login

Wie auch beim SSH1-Client wird es beim SSH2-Client ermöglicht interaktive Kommandos wie z.B. den Editor vi oder Shellbefehle auf dem Server auszuführen, die Funktionalität ist eine Obermenge des Remote-Shell-Kommandos rsh bzw. rlogin.
Die erste interaktive Sitzung:

```
user@client $ ssh2 server ↵
Accepting host server without checking.
Passphrase for key "/home/user/.ssh2/id_dsa_1024_a" with comment "
1024-bit dsa, created by user@client Tue Jan 26 15:11:44 1999":
<Passphrase> ↵
user@server $ vi ↵
...Arbeiten mit interaktiven Kommandos auf dem Server...
user@server $ exit ↵
user@client $
```

Der SSH2-Client erstellt und aktualisiert automatisch eine Liste der öffentlichen Serverschlüssel im Verzeichnis Datei $HOME/.ssh2/hostkeys. Damit läßt sich sicherstellen, daß der konnektierte Server auch der ist, der er vorgibt zu sein. Die öffentlichen Serverschlüssel lassen sich auch manuell übertragen, um bei der ersten Verbindung ggf. Zweifel an der Authentizität des Servers auszuräumen:

- Kopieren des öffentlichen Schlüssels des Server auf Diskette:
  `user@server $ cp /etc/ssh2/hostkey.pub /floppy/floppy0/hostkey1.pub ↵`
- Eintragen des öffentlichen Schlüssels des Server als authentisch:
  `user@client $ cp /floppy/floppy0/hostkey1.pub $HOME/.ssh2/hostkeys/key_22_server.pub ↵`
  Der verwendete Dateiname setzt sich aus dem angeprochenen Server und dessen Port zusammen. Der Hostkey des Servers server2, dessen SSH2-Server auf TCP-Port 4022 wartet, würde also key_4022_server2.pub heissen.
- Setzen der Berechtigungen für die Datei:
  `user@client $ chmod 600 $HOME/.ssh2/hostkeys/key_22_server.pub ↵`

Bei jeder folgenden Sitzung mit dem gleichen Server entfällt diese Vorgehensweise:

```
user@client $ ssh2 server ↵
Passphrase for key "/home/user/.ssh2/id_dsa_1024_a" with comment "
1024-bit dsa, created by user@client Tue Jan 26 15:11:44 1999":
<Passphrase> ↵
user@server $ vi ↵
...Arbeiten mit interaktiven Kommandos auf dem Server...
user@server $ exit ↵
user@client $
```

Bei anderslautendem Benutzernamen auf dem Server sieht der Aufruf wie folgt aus:

```
user@client $ ssh2 –l serveruser server ↵
```

#### 2.1.4.2.2 entfernte Kommandoausführung

Nichtinteraktives Ausführen von Kommandos auf dem Server vollzieht sich bei Verwendung von SSH2 wie bei SSH1, vgl. **2.1.4.1.2 entfernte Kommandoausführung** (Seite 10)

#### 2.1.4.2.3 Benutzen des Authentifizierungs-Agent

Um beim mehrmaligen Benutzen von SSH2 den Benutzer vom Eingeben der Passphrase des lokalen Schlüssels zu verschonen, existiert ein Programm, daß diese Informationen bei Bedarf liefert: ssh-agent2. Der Authentifizierungs-Agent wird zu Beginn einer Sitzung auf dem Client-Rechner gestartet und alle Kindprozesse des Authentifizierungs-Agent erben dessen Authentifizierungsinformationen.
Eine Beispielsitzung:

```
user@client $ ssh-agent2 $SHELL ↵
user@client $ ssh-add2 ↵
Adding identity: /home/user/.ssh2/id_dsa_1024_a.pub
Need passphrase for /home/user/.ssh2/id_dsa_1024_a (1024-bit dsa,
created by user@client Tue Jan 26 15:11:44 1999).
```

```
Enter passphrase: <Passphrase> ↵
user@client $ ssh2 server ↵
user@server $
```

Der Authentifizierungsagent wirkt über SSH2-Verbindungen hinweg, d.h. auch in der Sitzung auf dem Server besteht Zugriff auf den Agent.

Das Weiterleiten des Authentifizierungsagent über Verbindungen hinweg läßt sich mit der Kommandooption „–a“ von ssh2 deaktivieren, mit der Kommandooption „+a“ von ssh2 wird das Weiterleiten aktiviert (dies ist die Standardeinstellung).

Der Authentifizierungs-Agent ssh-agent2 kann auch ohne Subshell direkt in den Hintergrund gestellt werden:

- Die Syntax für den Aufruf aus einer Shell, die Environment-Variablen wie die Bourne-Shell exportiert lautet:
  ```
  user@client $ eval `ssh-agent2 –s` ↵
  Agent pid 1234
  user@client $
  ```
  Das Entfernen des Agents geschieht wie folgt:
  ```
  user@client $ kill –SIGTERM 1234 ↵
  user@client $
  ```
- Die Syntax für den Aufruf aus einer Shell, die Environment-Variablen wie die c-Shell exportiert lautet:
  ```
  user@client % eval `ssh-agent2 –c` ↵
  user@client %
  ```
  Das Entfernen des Agents geschieht wie folgt:
  ```
  user@client % kill –SIGTERM 1234 ↵
  user@client %
  ```

Wenn der Authentifizierungsagent ssh-agent2 zusätzlich mit der Option „-1“ gestartet wird, befindet er sich im SSH1-Kompatibilitätsmodus, d.h. er kann auch SSH1-Schlüssel verwalten. Dies gilt jedoch nicht bei der nichtkommerziellen Version von SSH2, da diese keine RSA-Schlüssel unterstützt.

Das Programm ssh-add2, mit dem Identitäten zum Authentifizierungs-Agent hinzugefügt werden können besitzt einige nützliche Optionen:

- Mit der Option „–l“ lassen sich alle derzeit im Agent gespeicherten Identitäten auflisten.
- Mit der Option „–p“ wird die Passphrase von der Standardeingabe gelesen und die Identität, die durch den nachfolgenden Dateinamen angegeben ist, hinzugefügt.
  ```
  user@client $ ssh-add2 -p $HOME/.ssh2/id_dsa_1024_a ↵
  Adding identity: /home/user/.ssh2/id_dsa_1024_a.pub
  Need passphrase for /home/user/.ssh2/id_dsa_1024_a (1024-bit dsa,
  created by user@client Tue Jan 26 15:11:44 1999).
  Enter passphrase: <Passphrase> ↵
  user@client $
  ```
  Diese Option ist die Standardoption, d.h. ohne Parameter an ssh-add2 zu übergeben wird diese Option benutzt.
- Mit der Option „–d“ läßt sich eine Identität aus dem Authentifizierungsagent löschen
- Mit der Option „–D“ lassen sich alle Identitäten aus dem Authentifizierungsagent löschen.

- Mit der Option „–d“ läßt sich eine Identität aus dem Authentifizierungsagent löschen
- Mit der Option „–D“ lassen sich alle Identitäten aus dem Authentifizierungsagent löschen.
- Mit der Option „–L“ läßt sich der Authentifizierungsagent duch ein Passwort schützen
- die Option „–U“ hebt den Passwortschutz des Authentifizierungsagents wieder auf.
- Wenn eine Identität mit der Option „–1“ hinzugefügt wird, so wird diese Identität nicht im SSH1-Kompatibilitätsmodus genutzt.
- wird die Option „–u“ genutzt, so liest der Authentifizierungsagent die Identität nicht aus einer Datei, sondern aus der zu übergebenden URL. Dies ermöglicht z.B. das Benutzen von Schlüsseln, die durch SSL (also Secure-HTTP) von einem Server abgerufen werden oder auch die Benutzung von anderen Schlüsselquellen, z.B. SmartCards.

Wenn clientseitig ein X-Display gesetzt ist, läßt sich die Passphrase auch mit einem grafischen Benutzerinterface einlesen:

```
user@client $ ssh-add2 </dev/null ↵
```

#### 2.1.4.2.4 Weiterleiten von X-Verbindungen

Das Weiterleiten von X-Verbindungen geschieht auf die gleiche Weise wie in 2.1.4.1.4 Weiterleiten von X-Verbindungen (Seite 12) beschrieben.

#### 2.1.4.2.5 Weiterleiten von sonstigen TCP-Verbindungen

Das Weiterleiten von sonstigen TCP-Verbindungen geschieht auf die gleiche Weise wie in 2.1.4.1.5 Weiterleiten von sonstigen TCP-Verbindungen (Seite 13) beschrieben.

#### 2.1.4.2.6 Kommandooptionen von SSH2

Die Kommandooptionen von SSH2 sind ähnlich denen des SSH1-Clients. Einige zusätzliche Optionen wurden implementiert. Nähere Informationen liefert die Manualseite von SSH2 durch Eingabe des Kommandos *man ssh2* ↵.

#### 2.1.4.2.7 Hinterlegen von Standardeinstellungen für verschiedene Server

Die Vorgehensweise für das Hinterlegen von Standardeinstellungen für verschiedene Server bei SSH2 ist ähnlich der beim SSH1-Client. Nähere Informationen liefert die Manualseite von SSH2 durch Eingabe des Kommandos *man ssh2* ↵. Die Konfigurationsdateinamen sind sinngemäß gewählt: SSH1 benutzt die Dateien $HOME/.ssh/config und /etc/ssh_config, SSH2 benutzt die Dateien $HOME/.ssh2/ssh2config und /etc/ssh2/ssh2_config

#### 2.1.4.2.8 Kopieren von Dateien: scp2

Die Kommandooptionen von scp1 sind ähnlich denen des scp1-Clients. Einige zusätzliche Optionen wurden implementiert. Nähere Informationen liefert die Manualseite von scp2 durch Eingabe des Kommandos *man scp2* ↵.

#### 2.1.4.2.9 Benutzte Dateiformate für Public-Key-Authentifizierung

Die benutzten Datenformate zum Speichern der Public-Keys sind bei SSH2 andere wie bei SSH1, da sowohl RSA- als auch alternativ DSA-Schlüssel verwaltet werden können. Nähere Informationen sind auf dem WWW-Server des Herstellers verfügbar (http://www.ssh.fi).

## *2.2 SSH unter Unix aus Systemverwaltersicht*

### 2.2.1 Voraussetzungen

Um SSH unter Unix installieren zu können müssen folgende Voraussetzungen erfüllt sein:

- Verfügbarkeit von tar bzw. GNU-tar (tape-archiver) und GNU-zip
- Verfügbarkeit von PGP (Pretty Good Privacy)

#### 2.2.1.1 Verfügbarkeit von tar bzw. GNU-tar (tape-archiver) und GNU-zip

Diese beiden Programme werden benötigt, um das Programmpaket, das als tar-Archiv in komprimierter Form vorliegt, auszupacken.
Sie sind frei verfügbar nach den Bestimmungen der Free Software Foundation:
http://www.gnu.org/copyleft/gpl.html.

GNU-zip ist verfügbar unter ftp://ftp.cs.uni-sb.de/pub/gnu/gzip-1.2.4.shar.Z; GNU-tar ist verfügbar unter ftp://ftp.cs.uni-sb.de/pub/gnu/tar-1.12.shar.gz, ein zum Betriebssystem gehörendes tar funktioniert auch.

- Eine kurze Anleitung zum Installieren von GNU-zip:
  Entpacken mit compress (compress ist in jedem Unix enthalten):
  *uncompress gzip-1.2.4.shar.Z* ↵
  Es liegt nun das Shell-Archiv gzip-1.2.4.shar im aktuellen Verzeichnis.
  Entpacken des Shell-Archivs:
  *sh gzip-1.2.4.shar* ↵
  Das Verzeichnis gzip-1.2.4 enthält die zum Übersetzen von GNU-zip erforderlichen Dateien.
  Wechseln ins Verzeichnis:
  *cd gzip-1.2.4* ↵
  Sollten besondere Wünsche bzgl. der Zielpfade bestehen, so lassen sich diese als Parameter an configure übergeben, Hinweise dazu liefert ein Blick in configure.
  Ermitteln der Systemspezifika:
  *./configure* ↵
  Übersetzen des Pakets:
  *make* ↵
  Entfernen von symbolischen Debuginformationen aus dem ausführbaren Programm:
  *strip gzip* ↵
  Installation der Programme, Manual-Seiten und Info-Seiten:
  *make install* ↵
  GNU-zip ist nun installiert.

- Eine kurze Anleitung zum Installieren von GNU-tar:
  Entpacken mit GNU-zip:
  *gunzip tar-1.12.shar.gz* ↵
  Es liegt nun das Shell-Archiv tar-1.12.shar im aktuellen Verzeichnis.
  Entpacken des Shell-Archivs:
  *sh tar-1.12.shar* ↵
  Das Verzeichnis tar-1.12 enthält die zum Übersetzen von GNU-tar erforderlichen Dateien.
  Wechseln ins Verzeichnis:
  *cd tar-1.12* ↵
  Sollten besondere Wünsche bzgl. der Zielpfade bestehen, so lassen sich diese als Parameter an configure übergeben; näheres dazu erfährt man nach Eingabe von *./configure –help* ↵.
  Ermitteln der Systemspezifika:
  *./configure* ↵
  Übersetzen des Pakets:
  *make* ↵
  Entfernen von symbolischen Debuginformationen aus dem ausführbaren Programm:
  *strip src/tar* ↵
  Installation der Programme, Manual-Seiten und Info-Seiten:
  *make install* ↵
  GNU-tar ist nun installiert.

Den Ablauf des Entpackens veranschaulicht folgendes Diagramm:

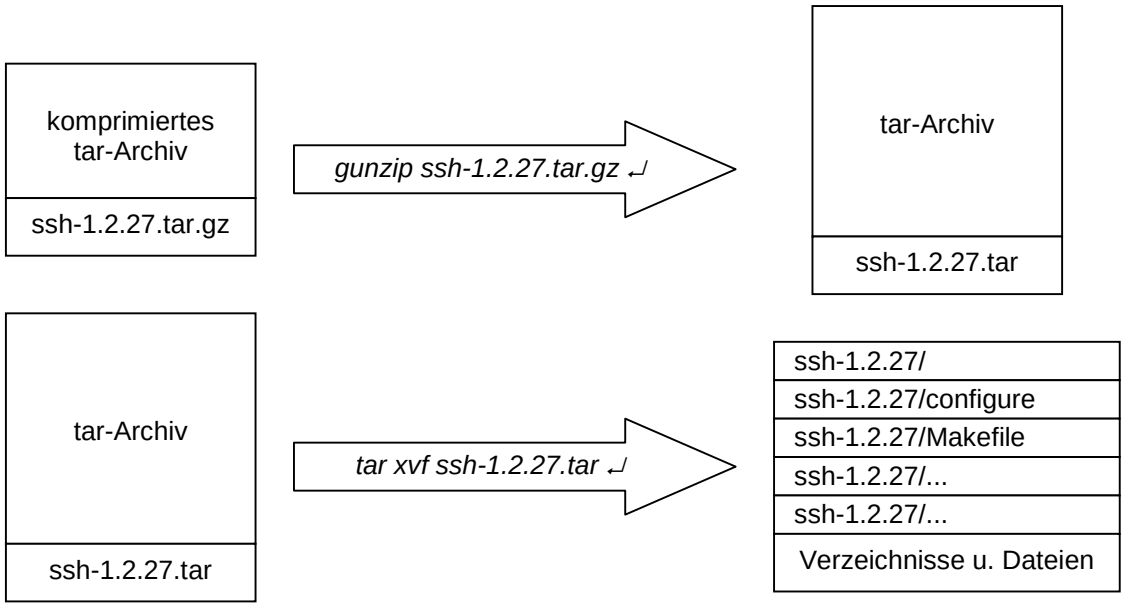



**Abbildung 2-19 Entpacken von Archiven mit GNU-zip und GNU-tar**

### 2.2.1.2 Verfügbarkeit von PGP (Pretty Good Privacy)

PGP wird benötigt, um die Distribution von SSH auf Echtheit zu prüfen.
Es ist als Paket mit Quelltexten verfügbar unter ftp://ftp.cert.dfn.de/pub/pgp/pgpi/2.x/src/pgp263is.tar.gz.
PGP darf von nicht-kommerziellen Anwendern kostenlos genutzt werden:
http://www.pgpi.com/faq/pgpi.shtml#License.

Eine kurze Anleitung zum Installieren von PGP:
Entpacken mit GNU-zip:

*gunzip pgp263is.tar.gz ↵*

Es liegt nun das tar-Archiv pgp263is.tar im aktuellen Verzeichnis.
Entpacken des tar-Archivs:

*tar xvf pgp263is.tar ↵*

Es liegen nun das eigentliche tar-Archiv pgp263ii.tar, die zum tar-Archiv gehörende Signatur pgp263ii.asc sowie mehrere Dokumentationsdateien (readme.usa, readme.1st und setup.doc) im aktuellen Verzeichnis.
Entpacken des tar-Archivs:

*tar xvf pgp263ii.tar ↵*

Das Verzeichnis src enthält die zum Übersetzen von PGP erforderlichen Dateien.
Wechseln ins Verzeichnis:

*cd src ↵*

Übersetzen des Pakets (für Solaris 2.x mit GNU-c-Compiler):

*make sun4sunos5gcc ↵*

Entfernen von symbolischen Debuginformationen aus dem ausführbaren Programm:

*strip pgp ↵*

Installation der Programme und Manualseiten:

*cp pgp /usr/local/bin ↵*
*cp ../doc/pgp.1 /usr/local/man/man1/ ↵*

PGP ist nun installiert.
Es gilt nun das Paket PGP auf seine Echtheit zu überprüfen. Für die internationale Version von PGP ist Stale Schumacher (mailto:stale@hypnotech.com) verantwortlich. Sein öffentlicher Schlüssel, der zur Echtheitsüberprüfung notwendig ist, ist über verschiedene PGP-Keyserver erhältlich, z.B. von http://www.surfnet.nl/pgp/pks-commands.html#extract. Der Schlüssel hat die ID 0xCCEF447D, er ist auch im Anhang (Seite 52) aufgeführt.
Den Schlüssel fügt man in eine Datei namens stale_schumacher.asc ein.
Das Aufnehmen der Keys in den eigenen Schlüsselring geschieht durch folgendes Kommando:

*pgp -ka stale_schumacher.asc ↵*

Das Überprüfen des Pakets geschieht durch folgendes Kommando:

*pgp pgp263ii.asc pgp263ii.tar* ↵

In der Ausgabe von pgp sollte folgender Text erscheinen:

```
File has signature.  Public key is required to check signature.

File 'pgp263ii.$00' has signature, but with no text.
Text is assumed to be in file 'pgp263ii.tar'.
.
Good signature from user "Stale Schumacher <stale@hypnotech.com>".
Signature made 1996/04/24 17:36 GMT using 1024-bit key, key ID CCEF447D
```

Nach dieser Ausgabe kann man sicher sein, daß das Paket pgp263ii.tar von Stale Schumacher stammt.

### 2.2.2 Lizensierung

SSH1 darf von Privatpersonen und zum nicht-kommerziellen Gebrauch genutzt werden (vgl. Datei COPYING im Archiv ssh-1.2.27.tar.gz), kommerzielle Versionen können von Data Fellows Ltd. http://www.datafellows.com bezogen werden. Nichtkommerzielle Version und kommerzielle Version sind vom Funktionsumfang identisch.

SSH2 darf von Privatpersonen und zum nicht-kommerziellen Gebrauch genutzt werden (vgl. Datei LICENSING im Archiv ssh-2.0.13.tar.gz), kommerzielle Versionen können von Data Fellows Ltd. http://www.datafellows.com bezogen werden. Nichkommerzielle Version und kommerzielle Version unterscheiden sich im Funktionsumfang:
Die kommerzielle Version unterstützt an Verschlüsselungsverfahren des, 3des, blowfish, idea und arcfour; die nichtkommerzielle Version unterstützt nur des, 3des und arcfour.
Die kommerzielle Version unterstützt an Public-Key-Verfahren DSS (Digital Signature Standard) und RSA; die nichtkommerzielle Version unterstützt nur DSS.

### 2.2.3 Beschaffen von SSH

Derzeit aktuelle Version für SSH1 ist 1.2.27, für SSH2 ist die Version 2.0.13 aktuell (Stand: 30.09.1999). Beide Pakete sind verfügbar unter ftp://ftp.cs.hut.fi/pub/ssh, es liegen jeweils Signaturdateien für die Echtheitsüberprüfung bei.
Folgende Dateien werden für SSH1 und SSH2 benötigt:

ssh-1.2.27.tar.gz
ssh-1.2.27.tar.gz.sig
ssh-2.0.13.tar.gz
ssh-2.0.13.tar.gz.sig

Gegebenenfalls werden Patches zur Behebung von Fehlern zur Verfügung gestellt, z.B.:

patch-ssh-2.0.x
patch-ssh-2.0.x.sig

Die Pakete werden auf mehreren anderen Servern gespiegelt, um weltweit eine schnelle Verfügbarkeit gewährleisten zu können, ein Spiegelserver für die Pakete ist: ftp://ftp.cert.dfn.de/pub/tools/net/ssh.

### 2.2.4 Überprüfen der SSH-Pakete auf Echtheit

Drei PGP-Schlüssel sind zur Echtheitsüberprüfung notwendig: der öffentliche Schlüssel der SSH1-Distribution (ylo@cs.hut.fi, ID: 0xDCB9AE01), der öffentliche Schlüssel der SSH2-Distribution (ssh2@ssh.fi, ID: 0xAFCA7459) und der öffentliche Schlüssel von Tatu Ylonen, dem Author (ylo@ssh.fi, ID: 0x961F4A35).
Die Schlüssel sind über verschiedene PGP-Keyserver erhältlich, z.B. von http://www.surfnet.nl/pgp/pks-commands.html#extract. Diese drei Schlüssel sind auch im Anhang aufgeführt (Seite 56).
Den Schlüssel der Distribution fügt man in eine Datei namens ssh1-key.asc bzw. ssh2-key.asc ein, den Schlüssel von Tatu Ylonen fügt man in eine Datei namens ylo-key.asc ein.
Die drei Schlüssel werden in den eigenen Schlüsselring aufgenommen:

*pgp -ka ssh1-key.asc* ↵
*pgp -ka ssh2-key.asc* ↵
*pgp -ka ylo-key.asc* ↵

Nun erfolgt das Überprüfen der beiden Pakete:

*pgp ssh-1.2.27.tar.gz.sig ssh-1.2.27.tar.gz* ↵

*pgp ssh-2.0.13.tar.gz.sig ssh-2.0.13.tar.gz* ↵

Als Ausgabe sollte folgender Text erscheinen:

```
File has signature.  Public key is required to check signature.

File 'ssh-1.2.27.tar.gz.sig' has signature, but with no text.
Text is assumed to be in file 'ssh-1.2.27.tar.gz'.
.
Good signature from user "Ssh distribution key <ylo@cs.hut.fi>".
Signature made 1998/07/08 16:43 GMT using 1024-bit key, key ID DCB9AE01
```

bzw. für ssh2:

```
File has signature.  Public key is required to check signature.

File 'ssh-2.0.13.tar.gz.sig' has signature, but with no text.
Text is assumed to be in file 'ssh-2.0.13.tar.gz'.
.
Good signature from user "Ssh 2 Distribution Key <ssh2@ssh.fi>".
Signature made 1998/11/16 14:15 GMT using 2048-bit key, key ID AFCA7459
```

Nach diesen Ausgaben kann man sicher sein, daß die beiden Pakete authentisch sind.

### 2.2.5 Entpacken der Pakete

Entpacken mit GNU-zip:

*gunzip ssh-1.2.27.tar.gz* ↵

*gunzip ssh-2.0.13.tar.gz* ↵

Im aktuellen Verzeichnis liegen nun 2 tar-Archive namens ssh-1.2.27.tar und ssh-2.0.13.tar.
Entpacken der tar-Archive:

*tar xvf ssh-1.2.27.tar* ↵

*tar xvf ssh-2.0.13.tar* ↵

Es liegen nun Verzeichnisse mit den Paketen vor: ssh-1.2.27 für SSH1, ssh-2.0.13 für SSH2.

### 2.2.6 Anwenden von Patches

Ein Patch bereinigt im Allgemeinen Fehler, die erst nach der Distribution der Version bereinigt wurden. Patches werden wie folgt angewandt:

*cd ssh-2.0.x* ↵

*patch -p1 -l <../patch-ssh-2.0.x* ↵

### 2.2.7 Übersetzen der Pakete

Vor der Übersetzung der Pakete muß bereits bekannt sein, wo die Programme, Konfigurationsdateien und Manualseiten letztendlich zur Ablage kommen, da diese Informationen in den Programmen enthalten sein müssen.

Bei der Ablagesystematik kommen mehrere Varianten in Frage:

1. Lokale Ablage aller Komponenten:
2. Lokale Ablage von Serverkonfigurationsdateien und Serverprogramm und gemeinsam genutzte Ablage von Clientprogrammen und Manualseiten
3. Gemeinsam genutzte Ablage von allen Komponenten bis auf die Serverkonfigurationsdateien

Die zweite Variante gewährleistet eine hohe Stabilität des Serverdienstes auch bei Ausfall des NFS-Servers verbunden mit geringem Installationsaufwand für die Clientprogramme und Dokumentationen.

Ablageort für Serverkonfigurationsdateien:

| | |
|---|---|
| /etc | (lokal) |

Ablageort für das Serverprogramm:

| | |
|---|---|
| /usr/local/sbin | (lokal) |

Ablageort für variable Serverdateien:

| | |
|---|---|
| /var/run | (lokal) |

Ablageort für Clientprogrammdateien:

| | | |
|---|---|---|
| /export/pub/IRIX/bin | (nfs-gemountet) | für IRIX |
| /export/pub/SOLARIS/bin | (nfs-gemountet) | für SOLARIS |

Ablageort für Manualseiten:

| | | |
|---|---|---|
| /export/pub/IRIX/man | (nfs-gemountet) | für IRIX |
| /export/pub/SOLARIS/man | (nfs-gemountet) | für SOLARIS |

Anmerkung zum Ablageort für Clientprogrammdateien:
Bei dieser Art der Installation ist eine rhosts-/shosts-Authentifizierung nicht möglich, da der Client nicht in der Lage ist von einem tcp-Port unter 1024 Verbindungen aufzubauen. Um dies zu erreichen müßte das Clientprogramm das s-Bit gesetzt haben, was sich bei nfs-gemounteten Verzeichnissen aus Sicherheitsgründen verbietet.

### 2.2.7.1 Übersetzen von SSH1

Besonderheit beim Übersetzen unter IRIX:

- Beim Übersetzen von SSH1 unter IRIX 6.2 ist zu beachten, daß der GNU-C-Compiler gcc (Version 2.8.0) das Paket nicht sauber übersetzt und der mitgelieferte Compiler cc benutzt werden muß; der GNU-C-Compiler darf also nicht im Pfad stehen.

Die oben festgelegten Zielpfade lassen sich an configure übergeben; configure ist ein Programm, dass die Systemvoraussetzungen und - besonderheiten für die Paketübersetzung prüft. Mit configure läßt sich auch festlegen, welche Verschlüsselungsprotokolle in den Programmen später verfügbar sind, näheres dazu erfährt man nach Eingabe von *./configure –help* ↵.

Anlegen des Verzeichnisses für variable Serverdateien (wird von configure auf Existenz geprüft):
*mkdir /var/run* ↵
Zum Übersetzen wechselt man ins Verzeichnis:
*cd ssh-1.2.27* ↵
Ermitteln der Systemvoraussetzungen unter Berücksichtigung oben festgelegter Pfade:
(Es handelt sich dabei jeweils um EINE Zeile.)
für IRIX:
*./configure --bindir=/export/pub/IRIX/bin --sbindir=/usr/local/sbin --sysconfdir=/etc --mandir=/export/pub/IRIX/man* ↵
für SOLARIS:
*./configure --bindir=/export/pub/SOLARIS/bin --sbindir=/usr/local/sbin --sysconfdir=/etc --mandir=/export/pub/SOLARIS/man* ↵

Besonderheit beim Übersetzen unter IRIX:

- Unter IRIX 6.2 wird Projektaccounting unterstützt, d.h. Benutzer können Projekten kostenstellenmäßig angehören. Von SSH1 wird dies nur bei lokaler Administration der Projektzuordnungstabellen /etc/project und /etc/projid unterstützt (vgl. *man project* ↵), nicht bei Benutzung von NIS bzw. NIS+.
  Zum Deaktivieren ändert man in der Datei ssh-1.2.27/config.h
  die Zeile:
  #define HAVE_SGI_PROJ_H 1
  in
  /* #define HAVE_SGI_PROJ_H 1 */

Übersetzen des Pakets:
*make* ↵
Entfernen von symbolischen Debuginformationen aus den ausführbaren Programmen:
*strip scp ssh ssh-add ssh-agent ssh-askpass ssh-keygen sshd* ↵

### 2.2.7.2 Übersetzen von SSH2

Besonderheit beim Übersetzen unter IRIX:

- Beim Übersetzen von SSH2 unter IRIX 6.2 ist zu beachten, daß der GNU-C-Compiler gcc (Version 2.8.0) das Paket nicht sauber übersetzt und der mitgelieferte Compiler cc benutzt werden muß; der GNU-C-Compiler darf also nicht im Pfad stehen.

Die in Kapitel **2.2.7 Übersetzen der Pakete** (Seite 32) festgelegten Zielpfade lassen sich an configure übergeben; configure ist ein Programm, dass die Systemvoraussetzungen und - besonderheiten für die Paketübersetzung prüft. Mit configure läßt sich auch festlegen, welche Verschlüsselungsprotokolle in den Programmen später verfügbar sind, näheres dazu erfährt man nach Eingabe von *./configure –help* ↵.

Zum Übersetzen wechselt man ins Verzeichnis:
*cd ssh-2.0.13* ↵
Ermitteln der Systemvoraussetzungen unter Berücksichtigung oben festgelegter Pfade:
(Es handelt sich dabei jeweils um EINE Zeile.)
für IRIX:
*./configure --bindir=/export/pub/IRIX/bin --sbindir=/usr/local/sbin --sysconfdir=/etc --mandir=/export/pub/IRIX/man* ↵
für SOLARIS:
*./configure --bindir=/export/pub/SOLARIS/bin --sbindir=/usr/local/sbin --sysconfdir=/etc --mandir=/export/pub/SOLARIS/man* ↵
Übersetzen des Pakets:
*make* ↵
Entfernen von symbolischen Debuginformationen aus den ausführbaren Programmen:
*cd apps/ssh* ↵
*strip scp2 sftp-server2 sftp2 ssh2 ssh-add2 ssh-agent2 ssh-askpass2 ssh-keygen2 sshd2* ↵

## 2.2.8 Installieren der Pakete

### 2.2.8.1 Installieren von SSH1

Zum Installieren wechselt man ins Verzeichnis:
*cd ssh-1.2.27* ↵
Installation der Programme, Manual-Seiten und Info-Seiten:
*make install* ↵

Was bewirkt dieses make install?

- Erzeugen eines Hostkeys:
  *./ssh-keygen -b 1024 –f /etc/ssh_host_key –N ‘‘* ↵
  Der private Hostkey wird dabei unter /etc/ssh_host_key nur für den Systemverwalter lesbar (Dateimodus 600) und der öffentliche Hostkey wird unter /etc/ssh_host_key.pub für alle lesbar (Dateimodus 644) abgelegt. Bei manchen Konfigurationen kann es vorkommen, daß der Kommentar des Hostkeys nur den Hostnamen (root@host) und nicht den vollqualifizierten Rechnernamen (root@host.dom.ain) enthält.
  Der Kommentar des Hostkeys läßt sich wie folgt ändern:
  *./ssh-keygen -f /etc/ssh_host_key –c* ↵
- Kopieren von Programmen, Konfigurationsbeispieldateien und Manualseiten:

| **Von** | **Nach** | **Modus** |
|---|---|---|
| ./ssh | /export/pub/IRIX/bin/ssh1 | 755 (4711) |
| ./ssh-keygen | /export/pub/IRIX/bin/ssh-keygen1 | 755 |
| ./ssh-agent | /export/pub/IRIX/bin/ssh-agent1 | 755 |
| ./ssh-add | /export/pub/IRIX/bin/ssh-add1 | 755 |
| ./scp | /export/pub/IRIX/bin/scp1 | 755 |
| ./ssh-askpass | /export/pub/IRIX/bin/ssh-askpass1 | 755 |
| ./make-ssh-known-hosts | /export/pub/IRIX/bin/make-ssh-known-hosts1 | 755 |
| ./sshd | /usr/local/sbin/sshd1 | 755 |
| ./ssh-keygen.1 | /export/pub/IRIX/man/man1/ssh-keygen1.1 | 644 |
| ./ssh-agent.1 | /export/pub/IRIX/man/man1/ssh-agent1.1 | 644 |
| ./ssh-add.1 | /export/pub/IRIX/man/man1/ssh-add1.1 | 644 |
| ./scp.1 | /export/pub/IRIX/man/man1/scp1.1 | 644 |
| ./ssh.1 | /export/pub/IRIX/man/man1/ssh1.1 | 644 |
| ./make-ssh-known-hosts.1 | /export/pub/IRIX/man/man1/make-ssh-known-hosts1.1 | 644 |
| ./sshd.8 | /export/pub/IRIX/man/man8/sshd1.8 | 644 |
| ./host_config.sample | /etc/ssh_config | 644 |
| ./server_config.sample | /etc/sshd_config | 644 |

  (Die Pfade für Solaris lauten sinngemäß)

- Erzeugen von symbolischen Links:

| Von | Nach |
|---|---|
| /export/pub/IRIX/bin/ssh1 | /export/pub/IRIX/bin/ssh |
| /export/pub/IRIX/bin/ssh | /export/pub/IRIX/bin/slogin |
| /export/pub/IRIX/bin/ssh-keygen1 | /export/pub/IRIX/bin/ssh-keygen |
| /export/pub/IRIX/bin/ssh-agent1 | /export/pub/IRIX/bin/ssh-agent |
| /export/pub/IRIX/bin/ssh-add1 | /export/pub/IRIX/bin/ssh-add |
| /export/pub/IRIX/bin/scp1 | /export/pub/IRIX/bin/scp |
| /export/pub/IRIX/bin/ssh-askpass1 | /export/pub/IRIX/bin/ssh-askpass |
| /export/pub/IRIX/bin/make-ssh-known-hosts1 | /export/pub/IRIX/bin/make-ssh-known-hosts |
| /usr/local/sbin/sshd1 | /usr/local/sbin/sshd |
| /export/pub/IRIX/man/man1/ssh-keygen1.1 | /export/pub/IRIX/man/man1/ssh-keygen.1 |
| /export/pub/IRIX/man/man1/ssh-agent1.1 | /export/pub/IRIX/man/man1/ssh-agent.1 |
| /export/pub/IRIX/man/man1/ssh-add1.1 | /export/pub/IRIX/man/man1/ssh-add.1 |
| /export/pub/IRIX/man/man1/scp1.1 | /export/pub/IRIX/man/man1/scp.1 |
| /export/pub/IRIX/man/man1/ssh1.1 | /export/pub/IRIX/man/man1/ssh.1 |
| /export/pub/IRIX/man/man1/ssh1.1 | /export/pub/IRIX/man/man1/slogin1.1 |
| /export/pub/IRIX/man/man1/ssh.1 | /export/pub/IRIX/man/man1/slogin.1 |
| /export/pub/IRIX/man/man1/make-ssh-known-hosts.1 | /export/pub/IRIX/man/man1/make-ssh-known-hosts.1 |
| /export/pub/IRIX/man/man8/sshd1.8 | /export/pub/IRIX/man/man8/sshd.8 |

(Die Pfade für Solaris lauten sinngemäß)

#### 2.2.8.1.1 Konfiguration des Servers

Der Server sshd bezieht seine Konfigurationsinformationen aus der Datei /etc/sshd_config. Diese Einstellung wurde durch die Option „--sysconfdir=/etc“ beim Aufruf von configure festgelegt und ist im Programm festgeschrieben. Durch Übergabe des Parameters „-f configfile“ an sshd kann diese Information übersteuert werden, um z.B. mehrere ssh-Server auf einer Maschine zu ermöglichen

Die im Paket mitgelieferte Datei kann und sollte den lokalen Sicherheitsrichtlinien angepaßt werden. Die verfügbaren Optionen sind vollständig in den Manualseiten des Servers aufgelistet (*man sshd1 ↵*).
Die Datei ist zeilenweise aufgebaut, Zeilen mit der Raute „#“ beginnend dienen als Kommentar und werden ignoriert.
Nachfolgend die mitgelieferte Datei /etc/sshd_config:

```
# This is ssh server systemwide configuration file.

Port 22
```
Gibt den TCP-Port an, an dem der sshd-Server auf eingehende Verbindungen wartet. Standardwert ist Port 22.

```
ListenAddress 0.0.0.0
```
An diese IP-Adresse bindet sich der sshd-Server und wartet auf eingehende Verbindungen. „0.0.0.0“ bedeutet jede servereigene Interface-Adresse; dieser Wert ist bei Multihomed-Systemen von Interesse, also bei Hosts, die mehrere IP-Adressen besitzen. Standardwert ist „0.0.0.0“.

```
HostKey /etc/ssh_host_key
```
Gibt an, wo der private Hostkey der Maschine abgelegt ist.
Der Standardwert ist „/etc/ssh_host_key“, was jedoch abhängig von den Festlegungen bei configure ist (./configure --sysconfdir=/etc ...).

```
RandomSeed /var/run/ssh_random_seed
```
Gibt an, wo die Datei ssh_random_seed sich befindet. /etc ist m.E. kein geeigneter Platz, da diese Datei regelmäßig vom sshd-Server geändert wird.
Standardwert ist „/etc/ssh_random_seed“.

```
ServerKeyBits 768
```
Gibt die Länge des dynamischen Serverkeys an; der minimale Wert ist 512. Bei rechenperformanten Maschinen ist 1152 angemessen, um eine hohe Sicherheit zu gewährleisten.
Standardwert ist 768.

`LoginGraceTime 600`
: Nach dieser Zeit trennt der Server die Verbindung, wenn sich der Benutzer nicht erfolgreich anmelden konnte. Ist der Wert 0, so existiert kein Zeitlimit.
Standardwert ist 600 (Sekunden).

`KeyRegenerationInterval 3600`
: Der Serverkey wird automatisch nach [Wert] Sekunden neu erzeugt, dieser Serverkey wird nur im Speicher gehalten. Ist der Wert 0, so wird der Serverkey nie neu erzeugt.
Standardwert ist 3600 (Sekunden).

`PermitRootLogin yes`
: Spezifiziert, ob root über ssh authentifiziert wird. Mögliche Werte sind „yes“, „nopwd“ oder „no“; „yes“ bedeutet, daß root-Logins über jede Authentifizierungsmöglichkeit, die auch anderen Benutzern zur Verfügung steht, möglich ist; „no“ bedeutet, daß root-Logins über ssh nicht möglich sind; „nopwd“ bedeutet, daß für root-Logins Passwort-Authentifizierung abgeschaltet ist.
Standardwert ist „yes“.

`IgnoreRhosts no`
: Gibt an, daß rhosts und shosts-Dateien nicht für die Authentifizierung genutzt werden; /etc/hosts.equiv und /etc/shosts.equiv werden trotzdem genutzt.
Standardwert ist „no“.

`StrictModes yes`
: Gibt an, ob der sshd-Server die Zugriffsrechte und Eigentümer von Benutzerverzeichnis und rhosts-Dateien prüfen soll bevor Anmeldungen erfolgen können.
Standardwert ist „yes“.

`QuietMode no`
: Ist QuietMode auf „yes“ gesetzt, werden nur fatale Fehler ans Systemlog gemeldet, ansonsten übliche Aktivitäten wie Logins und Logoffs ebenfalls.
Standardwert ist „no“.

`X11Forwarding yes`
: Gibt an, ob Weiterleitung von X11-Verbindungen erlaubt ist.
Der Standardwert ist „yes“. Durch Ausschalten dieser Option wird die Sicherheit nicht erhöht, da es jedem Benutzer freisteht ein eigenes X11-Weiterleitungsprogramm zu installieren.

`X11DisplayOffset 10`
: Dieser Wert gibt an, ab welcher Displaynummer sshd mit der Erzeugung von X11-Pseudodisplays beginnen soll. Ein Wert von 10 läßt 10 echte X-Server (von localhost:0 bis localhost:9) auf der Maschine zu, danach werden X11-Pseudodisplays für die X11-Weiterleitung von Remote-Benutzern erzeugt (localhost:10 ...).

`FascistLogging no`
: Gibt an, ob Aktivitäten ausführlich mitprotokolliert werden, was zum Testen sinnvoll ist.
Standardwert ist „no“.

`PrintMotd yes`
: Gibt an, ob der Server bei interaktivem Login eines Benutzers /etc/motd ausgeben soll oder nicht.
Standardwert ist „yes“.

`KeepAlive yes`
: Gibt an, ob der ssh-Server KeepAlive-Pakete an den Client schickt, d.h. Pakete zur Prüfung der Verbindung und Verfügbarkeit des Clients.
Der Standardwert ist „yes“; der Server terminiert also Verbindungen mit nicht beantworteten KeepAlive-Paketen. Um KeepAlive-Pakete auszuschalten, muß dies sowohl in der Serverkonfigurationsdatei als auch in der Clientkonfigurationsdatei ($HOME/.ssh/config) geschehen.

`SyslogFacility DAEMON`
: Gibt an, welchen Typ die Meldungen haben werden, die der sshd-Server ans Log-System syslogd sendet. Mögliche Typen sind „DAEMON“, „USER“, „AUTH“, „LOCAL0“, „LOCAL1“, „LOCAL2“, „LOCAL3“, „LOCAL4“, „LOCAL5“, „LOCAL6“, „LOCAL7“. Diese Option dient der Vorbereitung des Trennens von Meldungen im Log-System.
Standardwert ist „DAEMON“.

`RhostsAuthentication no`
: Gibt an, ob Authentifizierung über rhosts oder /etc/hosts.equiv zugelassen ist. Diese Methode sollte nicht zulässig sein, da sie ausschliesslich über die IP-Adresse authentifiziert.
Standardwert ist „no“.

`RhostsRSAAuthentication yes`
: Gibt an, ob Authentifizierung über rhosts oder /etc/hosts.equiv in Verbindung mit einer erfolgreichen RSA-Host-Authentifizierung zugelassen ist.
Standardwert ist „yes“.

`RSAAuthentication yes`
: Gibt an, ob reine RSA-Benutzerauthentifizierung zugelassen ist.
Standardwert ist „yes“.

`PasswordAuthentication yes`
: Gibt an, ob Passwort-Authentifizierung erlaubt ist.
Standardwert ist „yes“.

`PermitEmptyPasswords yes`
: Wenn Passwort-Authentifizierung erlaubt ist gibt dieser Wert an, ob der ssh-Server Zugang zu Accounts, die kein Passwort haben, erlaubt.
Standardwert ist „yes“.

`UseLogin no`

`# CheckMail no`
: Gibt an, ob bei interaktivem Login eine Meldung über neue Mail ausgegeben wird oder nicht.
Standardwert ist „yes“.

`# PidFile /u/zappa/.ssh/pid`
: Gibt den Ablageort des Files an, in dem die Prozeß-ID des sshd-Servers abgespeichert ist.
Standardwert ist „/var/run/sshd.pid“ bzw. bei Nichtexistenz des Verzeichnisses /var/run ist der Standardwert „/etc/sshd.pid“.

`# AllowHosts *.our.com friend.other.com`
: Durch dieses Schlüsselwort wird ein Login nur von Hosts erlaubt, die der nachfolgenden IP-Nummern- und/oder Hostnamensmaske entsprechen; Einzel- und Allexistenzquantor („?“ und „*“) sind erlaubt.
Standardmäßig darf die Verbindung von beliebigen Hosts initiiert werden.
Beispiel: `AllowHosts 134.96.*.* *.uni-stuttgart.de`
Alternativ kann ssh auch mit tcp-Wrapper-Unterstützung übersetzt werden, d.h. bei jeder Verbindung werden die Dateien /etc/hosts.allow und /etc/hosts.deny zum Überprüfen der Zulässigkeit der Verbindung durchsucht.

`# DenyHosts lowsecurity.theirs.com *.evil.org evil.org`
: Durch dieses Schlüsselwort wird ein Login nur von Hosts erlaubt, die der nachfolgenden IP-Nummern- und/oder Hostnamensmaske nicht entsprechen; Einzel- und Allexistenzquantor („?“ und „*“) sind erlaubt.
Standardmäßig darf die Verbindung von beliebigen Hosts initiiert werden.
Beispiel: `DenyHosts 194.25.2.* *.t-online.de`

`# Umask 022`
: Gibt die Rechtemaske an, mit der sshd und seine Nachkommen in der Prozeßhierarchie Dateien anlegen.
Standardmßig ist keine Rechtemaske gesetzt.

`# SilentDeny yes`
: Gibt an, ob erfolglose Anmeldeversuche mitprotokolliert werden oder nicht.
Standardwert ist „no“.

Zusätzliche Optionen in der Datei /etc/sshd_config sind:

`AllowGroups`
: Der Benutzer muß zusätzlich zur Authentifizierung einer der hier genannten Gruppen als primärer Benutzer angehören; Einzel- und Allexistenzquantor („?“ und „*“) sind erlaubt.
Standardwert ist keine Einschränkung nach Gruppenzugehörigkeiten.
Beispiel: `AllowGroups adm user`

`AccountExpireWarningDays`
: Gibt an ab welcher Anzahl von Tagen vor ungültigwerden des Accounts eine Warnung ausgegeben wird. Beim Wert von 0 wird keine Warnung ausgegeben.
Standardwert ist 14 Tage.
Beispiel: `AccountExpireWarningDays 30`

`AllowSHosts`
: Durch dieses Schlüsselwort wird ein Login mit Rhosts-RSA-Authentication (durch .shosts, .rhosts und /etc/hosts.equiv) nur von Hosts erlaubt, die der nachfolgenden IP-Nummern-

und/oder Hostnamensmaske entsprechen; Einzel- und Allexistenzquantor („?“ und „*“) sind erlaubt.
Standardmäßig darf die Verbindung von beliebigen Hosts initiiert werden.
Beispiel: `AllowSHosts 134.96.*.* *.uni-stuttgart.de`

`AllowTcpForwarding`
Gibt an ob Weiterleitung von TCP-Verbindungen erlaubt ist.
Der Standardwert ist yes. Durch Ausschalten dieser Option wird die Sicherheit nicht erhöht, da es jedem Benutzer freisteht ein eigenes TCP-Weiterleitungsprogramm zu installieren.

`AllowUsers`
Der Benutzer muß zusätzlich zur Authentifizierung auf ein Benutzernamen-Muster oder ein Benutzernamen-/Hostnamen-Muster passen; Einzel- und Allexistenzquantor („?“ und „*“) sind erlaubt.
Standardwert ist keine Einschränkung nach Benutzernamen-/Hostnamen-Mustern.
Beispiel: `AllowUsers betty jane@134.96.*.* uucp@*.uni-stuttgart.de`

`DenyGroups`
Der Benutzer darf zusätzlich zur Authentifizierung keiner der hier genannten Gruppen als primärer Benutzer angehören; Einzel- und Allexistenzquantor („?“ und „*“) sind erlaubt.
Standardwert ist keine Einschränkung nach Gruppenzugehörigkeiten.
Beispiel: `DenyGroups mail uucp`

`DenySHosts`
Durch dieses Schlüsselwort wird ein Login mit Rhosts-RSA-Authentication (durch .shosts, .rhosts und /etc/hosts.equiv) nur von Hosts erlaubt, die der nachfolgenden IP-Nummern- und/oder Hostnamensmaske nicht entsprechen; Einzel- und Allexistenzquantor („?“ und „*“) sind erlaubt.
Standardmäßig darf die Verbindung von beliebigen Hosts initiiert werden.
Beispiel: `DenySHosts 194.25.2.* *.t-online.de`

`DenyUsers`
Der Benutzer darf zusätzlich zur Authentifizierung nicht auf ein Benutzernamen-Muster und nicht auf ein Benutzernamen-/Hostnamen-Muster passen; Einzel- und Allexistenzquantor („?“ und „*“) sind erlaubt.
Standardwert ist keine Einschränkung nach Benutzernamen-/Hostnamen-Mustern.
Beispiel: `DenyUsers guest uucp@194.25.2.*.*`

`ForcedEmptyPasswdChange`
Gibt an, ob der Benutzer bei leerem Passwort zum Ändern des Passworts gezwungen wird.
Standardwert ist „no“.

`ForcedPasswdChange`
Gibt an, ob der Benutzer zur Passwortänderung gezwungen wird, wenn das Passwort ungültig geworden ist.
Standardwert ist „yes“.

`IdleTimeout`
Legt fest, wie lange die Verbindung unbenutzt sein darf bevor sie geschlossen wird. Als Postfix hinter dem numerischen Wert sind „s“ für Sekunden, „m“ für Minuten, „h“ für Stunden, „d“ für Tage und „w“ für Wochen möglich, kein Postfix ist gleichbedeutend mit Sekunden.
Standardwert ist kein IdleTimeout.
Beispiel: `IdleTimeout 7d`

`IgnoreRootRhosts`
Gibt an, daß rhosts und shosts-Dateien nicht für die Authentifizierung von root genutzt werden.
Standardwert ist der Wert von IgnoreRhosts.

`KerberosAuthentication`
Gibt an, ob Kerberos-V5-Authentifizierung zugelassen ist.
Standardwert ist „yes“

`KerberosOrLocalPasswd`
Wenn gesetzt kann sich der Benutzer bei mißlungener Kerberos-Authentifizierung zusätzlich über lokale Mechanismen wie Passwort-Authentifizierung anmelden.
Standardwert ist „no“.

`TISAuthentication`
: Gibt an, ob Authentifizierung durch den Authentifizierungsserver des TIS-Firewallkits zugelassen ist. Weitere Informationen über das TIS-Firewallkit sind unter der URL ftp://ftp.tis.com/pub/firewalls/toolkit/LICENSE verfügbar.
Standardwert ist „no“

`PasswordExpireWarningDays`
: Gibt an ab welcher Anzahl von Tagen vor Ungültigwerden des Passworts eine Warnung ausgegeben wird. Beim Wert von 0 wird keine Warnung ausgegeben.
Standardwert ist 14 Tage.
Beispiel: `PasswordExpireWarningDays 30`

`XAuthLocation`
: Gibt den Pfad zum xauth-Programm an. Der Standardwert des Pfads wird bereits von configure ermittelt und in den ssh-Daemon eincompiliert.

Nachfolgend eine restriktive Serverkonfigurationsdatei /etc/sshd_config, die ausschließlich RSA-basierende Authentifikation und kein root-Login zuläßt:

```
# This is ssh server systemwide configuration file.

Port                    22
ListenAddress           0.0.0.0
HostKey                 /etc/ssh_host_key
RandomSeed              /var/run/ssh_random_seed
ServerKeyBits           1152
LoginGraceTime          600
KeyRegenerationInterval 3600
PermitRootLogin         no
IgnoreRhosts            no
StrictModes             yes
QuietMode               no
X11Forwarding           yes
X11DisplayOffset        10
FascistLogging          yes
PrintMotd               yes
KeepAlive               yes
SyslogFacility          DAEMON
RhostsAuthentication    no
RhostsRSAAuthentication yes
RSAAuthentication       yes
PasswordAuthentication  no
PermitEmptyPasswords    no
UseLogin                no
CheckMail               yes
PidFile                 /var/run/sshd.pid
SilentDeny              no
```

#### 2.2.8.1.2 Aktualisieren der Servicenamen-Datenbank

Die Datei /etc/services wird um den TCP-Port, den SSH verwendet, ergänzt.
An die Datei fügt man folgende Zeile an:

```
ssh             22/tcp                          # SecureShell Protocol
```

Anfügen der Zeile:

*echo "ssh 22/tcp # SecureShell Protocol " >> /etc/services* ↵

Eine einer Übersicht der Netzwerkverbindungen wird jetzt anstatt der Zahl 22 für den TCP-Port das Postfix ssh verwendet.
Beispiel:

Die Eingabe von

*/usr/etc/netstat -a | grep .ssh* ↵

unter IRIX

bzw. von

*/bin/netstat –a | grep .ssh* ↵

unter SOLARIS

zeigt alle z.Zt. etablierten SSH-Verbindungen:

```
tcp   0   0  0.0.0.0.ssh      0.0.0.0.*              LISTEN
tcp   0   0  servereth0.ssh   client..1015           ESTABLISHED
```

#### 2.2.8.1.3 Aufnehmen von SSH in die Systemstartdateien

Es gibt zwei Möglichkeiten des Startens von sshd:

- Jede eingehende Verbindung auf TCP-Port 22 (den sshd-Port) wird von inetd, dem Internet-Services-Daemon entgegengenommen und es wird jeweils ein sshd-Prozeß gestartet.
  Das Einbinden geschieht durch Einfügen folgender Zeile in die Datei /etc/inetd.conf:
  `ssh stream tcp nowait root /usr/local/sbin/sshd sshd -i`
  inetd wird zum Neueinlesen seiner Konfigurationsdatei das Signal SIGHUP gesendet:
  *kill -HUP ‘/usr/bin/ps -e | /usr/bin/grep -w inetd | /usr/bin/sed -e ’s/^ */’ -e ’s/ .*//‘* ↵
  Nachteil dieser ersten Möglichkeit ist, daß für jede eingehende Verbindung ein neuer Serverkey erzeugt werden muß, was zusätzliche Rechnerbelastung und Zeitverlust bei jeder eingehenden Verbindung bedeutet.
- Ein verwaltender sshd-Prozeß wird via init-Runlevel, also über Systemstartdateien, gestartet, nimmt eingehende Verbindungen entgegen und erzeugt für jede Verbindung einen eigenen sshd-Prozeß.

Zur Implementierung der zweiten Methode:
Das Einbinden geschieht über dieses Script, daß unter /etc/init.d/sshd abgelegt wird:

```
#!/bin/sh
#

case "$1" in
’start’)
        if [ -f /etc/sshd_config -a -f /usr/local/sbin/sshd ]; then
                echo "sshd service starting."
                /usr/local/sbin/sshd
        fi
        ;;
’stop’)
        [ ! -f /var/run/sshd.pid ] && exit 0
        sshdpid=‘cat /var/run/sshd.pid‘
        if [ "$sshdpid" -gt 0 ]; then
                echo "Stopping the sshd service."
                /usr/bin/kill $sshdpid
        fi
        ;;
*)
        echo "Usage: /etc/init.d/sshd { start | stop }"
        ;;
esac
exit 0
```

Das Script wird mit den korrekten Attributen versehen:

*chown root.sys /etc/init.d/sshd* ↵
*chmod 744 /etc/init.d/sshd* ↵

Dann wird das Script in die jeweiligen init-Runlevel eingebunden:

*cd /etc/rc0.d ; ln –s ../init.d/sshd K11sshd* ↵
*cd /etc/rc1.d ; ln –s ../init.d/sshd K11sshd* ↵
*cd /etc/rc2.d ; ln –s ../init.d/sshd S98sshd* ↵

Der sshd-Server ist nun in die Systemstartdateien eingebunden und startet bei jedem Systemstart automatisch.
Um den Server manuell zu starten (z.B. um jetzt einen Systemneustart zu vermeiden) gibt man ein:

*/etc/init.d/sshd start* ↵

Sinngemäß läßt sich der Server durch Eingabe von

*/etc/init.d/sshd stop* ↵

manuell stoppen, um beispielsweise eine neue Version einzuspielen.

#### 2.2.8.1.4 Aufnehmen von SSH in die Benutzerprofile

Um den Benutzern die Nutzung von SSH zu ermöglichen, muß der Suchpfad für Programme (PATH-Variable) und der Suchpfad für Manualseiten (MANPATH-Variable) um die betreffenden Pfade, in die SSH installiert wurde, erweitert werden.
Die Änderungen müssen nur zentral in der Datei /export/home_ch/bin/profile.student.basis.97 vorgenommen werden, da in jedem Anwenderprofile diese Datei ausgeführt wird.
Folgende Änderungen sind notwendig:

```
if [ ‘uname‘ = "SunOS" ]
then
...
      PATH=$PATH:/export/pub/SOLARIS/bin
      MANPATH=$MANPATH:/export/pub/SOLARIS/man
...
else
...
      PATH=$PATH:/export/pub/IRIX/bin
      MANPATH=$MANPATH:/export/pub/IRIX/man
...
fi
```

#### 2.2.8.1.5 Abgleich der öffentlichen Schlüssel der Installationen

Um eine sichere Hostidentifizierung unabhängig von Nameservice und IP-Adresse zwischen einzelnen ssh-Installationen zu ermöglichen, werden auf jedem in Frage kommendem Host die öffentlichen Schlüssel anderer Hosts zentral in der Datei /etc/ssh_known_hosts gespeichert. Um die öffentlichen Hostkeys einer ganzen Domain einzusammeln, liegt der ssh-Distribution ein Script namens make-ssh-known-hosts bei, das diese Aufgabe übernimmt:

*make-ssh-known-hosts htw.uni-sb.de >/etc/ssh_known_hosts* ↵

Dieses Script braucht nur auf einer Maschine ausgeführt zu werden; von dieser Maschine kann dann die erzeugte Datei auf die anderen Maschinen kopiert werden:

scp */etc/ssh_known_hosts andere_maschine:/etc/ssh_known_hosts* ↵

Weitere Informationen zum Aufbau der Datei /etc/ssh_known_hosts sind zu finden in **2.1.4.1.9.2 Die öffentliche Serverschlüsseldatei known_hosts** (Seite 25)

### 2.2.8.2 Installieren von SSH2

Zum Installieren wechselt man ins Verzeichnis:

*cd ssh-2.0.13* ↵

Installation der Programme, Manual-Seiten und Info-Seiten:

*make install* ↵

Was bewirkt dieses make install?

- Erzeugen eines Hostkeys:

  *./ssh-keygen2 -P -b 1024 -t dsa -c "1024-bit dsa hostkey" -o /etc/ssh2/hostkey* ↵

  Der private Hostkey wird dabei unter /etc/ssh2/host_key nur für den Systemverwalter lesbar (Dateimodus 600) und der öffentliche Hostkey wird unter /etc/ssh2/host_key.pub für alle lesbar (Dateimodus 644) abgelegt.
- Kopieren von Programmen, Konfigurationsbeispieldateien und Manualseiten:

| **Von** | **Nach** | **Modus** |
|---|---|---|
| ./apps/ssh/ssh2 | /export/pub/IRIX/bin/ssh2 | 755 (4711) |
| ./apps/ssh/ssh-keygen2 | /export/pub/IRIX/bin/ssh-keygen2 | 755 |
| ./apps/ssh/ssh-agent2 | /export/pub/IRIX/bin/ssh-agent2 | 755 |
| ./apps/ssh/ssh-add2 | /export/pub/IRIX/bin/ssh-add2 | 755 |
| ./apps/ssh/scp2 | /export/pub/IRIX/bin/scp2 | 755 |
| ./apps/ssh/ssh-askpass2 | /export/pub/IRIX/bin/ssh-askpass2 | 755 |
| ./apps/ssh/sftp-server2 | /export/pub/IRIX/bin/sftp-server2 | 755 |
| ./apps/ssh/sftp2 | /export/pub/IRIX/bin/sftp2 | 755 |
| ./apps/ssh/sshd2 | /usr/local/sbin/sshd2 | 755 |
| ./apps/ssh/ssh-keygen2.1 | /export/pub/IRIX/man/man1/ssh-keygen2.1 | 644 |
| ./apps/ssh/ssh-agent2.1 | /export/pub/IRIX/man/man1/ssh-agent2.1 | 644 |

| | | |
|---|---|---|
| ./apps/ssh/ssh-add2.1 | /export/pub/IRIX/man/man1/ssh-add2.1 | 644 |
| ./apps/ssh/scp2.1 | /export/pub/IRIX/man/man1/scp2.1 | 644 |
| ./apps/ssh/ssh2.1 | /export/pub/IRIX/man/man1/ssh2.1 | 644 |
| ./apps/ssh/sftp2.1 | /export/pub/IRIX/man/man1/sftp2.1 | 644 |
| ./apps/ssh/sshd2.8 | /export/pub/IRIX/man/man8/sshd2.8 | 644 |
| ./apps/ssh/ssh2_config | /etc/ssh2/ssh2_config | 644 |
| ./apps/ssh/sshd2_config | /etc/ssh2/sshd2_config | 644 |

(Die Pfade für Solaris lauten sinngemäß)

- Erzeugen von symbolischen Links:

| **Von** | **Nach** |
|---|---|
| /export/pub/IRIX/bin/ssh2 | /export/pub/IRIX/bin/ssh |
| /export/pub/IRIX/bin/ssh-keygen2 | /export/pub/IRIX/bin/ssh-keygen |
| /export/pub/IRIX/bin/ssh-agent2 | /export/pub/IRIX/bin/ssh-agent |
| /export/pub/IRIX/bin/ssh-add2 | /export/pub/IRIX/bin/ssh-add |
| /export/pub/IRIX/bin/scp2 | /export/pub/IRIX/bin/scp |
| /export/pub/IRIX/bin/ssh-askpass2 | /export/pub/IRIX/bin/ssh-askpass |
| /export/pub/IRIX/bin/sftp2 | /export/pub/IRIX/bin/sftp |
| /export/pub/IRIX/bin/sftp-server2 | /export/pub/IRIX/bin/sftp-server |
| /usr/local/sbin/sshd2 | /usr/local/sbin/sshd |
| /export/pub/IRIX/man/man1/ssh-keygen2.1 | /export/pub/IRIX/man/man1/ssh-keygen.1 |
| /export/pub/IRIX/man/man1/ssh-agent2.1 | /export/pub/IRIX/man/man1/ssh-agent.1 |
| /export/pub/IRIX/man/man1/ssh-add2.1 | /export/pub/IRIX/man/man1/ssh-add.1 |
| /export/pub/IRIX/man/man1/scp2.1 | /export/pub/IRIX/man/man1/scp.1 |
| /export/pub/IRIX/man/man1/ssh2.1 | /export/pub/IRIX/man/man1/ssh.1 |
| /export/pub/IRIX/man/man1/ssh.1 | /export/pub/IRIX/man/man1/slogin.1 |
| /export/pub/IRIX/man/man1/sftp2.1 | /export/pub/IRIX/man/man1/sftp.1 |
| /export/pub/IRIX/man/man8/sshd2.8 | /export/pub/IRIX/man/man8/sshd.8 |

(Die Pfade für Solaris lauten sinngemäß)

#### 2.2.8.2.1 Konfiguration des Servers

Der Server sshd bezieht seine Konfigurationsinformationen aus der Datei /etc/ssh2/sshd2_config. Diese Einstellung wurde durch die Option „--sysconfdir=/etc“ beim Aufruf von configure festgelegt und ist im Programm festgeschrieben. Durch Übergabe des Parameters „-f configfile“ an sshd kann diese Information übersteuert werden, um z.B. mehrere ssh-Server auf einer Maschine zu ermöglichen

Die im Paket mitgelieferte Datei kann und sollte den lokalen Sicherheitsrichtlinien angepaßt werden. Die verfügbaren Optionen sind vollständig in den Manualseiten des Servers aufgelistet (*man sshd2 ↵*).
Die Datei ist zeilenweise aufgebaut, Zeilen mit Raute „#“ beginnend dienen als Kommentar und werden ignoriert.
Nachfolgend die mitgelieferte Datei /etc/ssh2/sshd2_config:

```
# sshd2_config
# SSH 2.0 Server Configuration File

*:
   Port                         22
```

Gibt den TCP-Port an, an dem der sshd-Server auf eingehende Verbindungen wartet. Standardwert ist Port 22.

```
   ListenAddress                0.0.0.0
```

An diese IP-Adresse bindet sich der sshd-Server und wartet auf eingehende Verbindungen. „0.0.0.0“ bedeutet jede servereigene Interface-Adresse; dieser Wert ist bei Multihomed-Systemen von Interesse, also bei Hosts, die mehrere IP-Adressen besitzen. Standardwert ist „0.0.0.0“.

```
   Ciphers                      AnyStd
#  Ciphers                      AnyCipher
#  Ciphers                      AnyStdCipher
#  Ciphers                      3des
```

Gibt an, welche Verschlüsselungsverfahren für die Verbindung angewandt werden, mehrere Verfahren können durch Komma getrennt aufgeführt werden.
Unterstützt werden derzeit „des“, „3des“, „blowfish“, „idea“ und „arcfour“, davon „des“, „3des“ und „arcfour“ in der nichtkommerziellen Version von SSH2.

`IdentityFile identification`
Gibt den Namen der Benutzer-Identifikationsdatei an, in dieser Datei sind die privaten Schlüssel des Benutzers aufgezählt.
Standardwert ist „identification“.

`AuthorizationFile authorization`
Gibt den Namen der Benutzer-Authorisations-Datei an; in dieser Datei gibt der Benutzer die öffentlichen Schlüssel an, die zur Anmeldung authorisiert sind.
Standardwert ist „authorization“.

`HostKeyFile hostkey`
Gibt an, wo der private Hostkey der Maschine abgelegt ist.
Der Standardwert ist „/etc/ssh2/host_key“, was jedoch abhängig von den Festlegungen bei configure ist (./configure --sysconfdir=/etc ...).

`PublicHostKeyFile hostkey.pub`
Gibt an, wo der öffentliche Hostkey der Maschine abgelegt ist.
Der Standardwert ist „/etc/ssh2/host_key.pub“, was jedoch abhängig von den Festlegungen bei configure ist (./configure --sysconfdir=/etc ...).

`RandomSeedFile random_seed`
Gibt an, wo die Datei ssh_random_seed sich befindet. /etc ist m.E. kein geeigneter Platz, da diese Datei regelmäßig vom sshd-Server geändert wird.
Standardwert ist „/etc/ssh_random_seed“.

`ForwardAgent yes`
Gibt an, ob die Verbindung zu einem Authentifikations-Agent zu einer entfernten Maschine weitergeleitet wird.
Standardwert ist „yes“.

`ForwardX11 yes`
Gibt an, ob Weiterleitung von X11-Verbindungen erlaubt ist.
Standardwert ist „yes“.

`PasswordAuthentication yes`
Gibt an, ob Passwort-Authentifizierung erlaubt ist.
Standardwert ist „yes“.

`PasswordGuesses 3`
Gibt die Anzahl der möglichen Anmeldeversuche an, wenn Passwort-Authentifikation benutzt wird.
Standardwert ist 3.

`PermitRootLogin yes`
Spezifiziert, ob root über ssh authentifiziert wird; „yes“ bedeutet, daß root-Logins über jede Authentifizierungsmöglichkeit, die auch anderen Benutzern zur Verfügung steht, möglich ist; „no“ bedeutet, daß root-Logins über ssh nicht möglich sind.
Standardwert ist „yes“.

`PubkeyAuthentication yes`
Gibt an, ob reine Public-Key-Benutzerauthentifizierung zugelassen ist, ein Synonym für dieses Schlüsselwort ist `RSAAuthentication`.
Standardwert ist „yes“.

`ForcePTTYAllocation no`
Gibt an, ob auch dann ein tty vom Server angefordert wird, wenn ein Kommando ausgeführt wird, daß kein tty braucht.
Standardwert ist „no“; die Funktionalität ist jedoch in SSH2 noch nicht implementiert.

`VerboseMode no`
Gibt an, ob sshd2 ausführliche Meldungen an syslogd ausgibt, die zum Debuggen dienen. Entspricht der Option „–d 2“ als Kommandozeilenoption.
Standardwert ist „no“.

`PrintMotd yes`
Gibt an, ob der Server bei interaktivem Login eines Benutzers die Datei /etc/motd ausgeben soll oder nicht.
Standardwert ist „yes“.

`UserConfigDirectory "%D/.ssh2"`

`# UserConfigDirectory "/etc/ssh2/auth/%U"`

Gibt das Directory für ssh2-relevante Benutzerinformationen an. „%D“ wird durch das Home-Directory des Benutzers ersetzt, „%U“ wird durch den Benutzernamen ersetzt.
Standardwert ist „%D/.sshd“, also $HOME/.ssh2.

```
# subsystem definitions

subsystem-sftp              sftp-server
```

Von sshd2 können beliebige Subsysteme aktiviert werden, die mit den Rechten des aufrufenden Clients gestartet werden.
Beispiel:

```
subsystem-oink          /bin/echo „oink-oink“
```

würde von einem Client durch folgende Zeile zur Ausgabe von “oink-oink“ aktiviert:

*ssh2 server –s oink ↵*

Zusätzliche Optionen in der Datei /etc/ssh2/sshd2_config sind:

`RhostsAuthentication`
Gibt an, ob Authentifizierung über rhosts oder /etc/hosts.equiv zugelassen ist. Diese Methode sollte nicht zulässig sein, da sie ausschliesslich über die IP-Adresse authentifiziert.
Standardwert ist „yes“; die Funktionalität ist jedoch in SSH2 noch nicht implementiert.

`RhostsPubKeyAuthentication` bzw.
`RHostsRSAAuthentication`
Gibt an, ob Authentifizierung über rhosts oder /etc/hosts.equiv in Verbindung mit einer erfolgreichen RSA-Host-Authentifizierung zugelassen ist.
Standardwert ist „yes“; die Funktionalität ist jedoch in SSH2 noch nicht implementiert.

`QuietMode`
Ist QuietMode auf yes gesetzt, werden nur fatale Fehler ans Systemlog gemeldet, ansonsten übliche Aktivitäten wie Logins und Logoffs ebenfalls.
Standardwert ist „no“.

`KeepAlive`
Gibt an, ob der ssh-Server KeepAlive-Pakete an den Client schickt, d.h. Pakete zur Prüfung der Verbindung und Verfügbarkeit des Clients.
Der Standardwert ist „yes“; der Server terminiert also Verbindungen mit nicht beantworteten KeepAlive-Paketen. Um KeepAlive-Pakete auszuschalten, muß dies sowohl in der Serverkonfigurationsdatei als auch in der Clientkonfigurationsdatei ($HOME/.ssh2/ssh2_config) geschehen.
Standardwert ist „yes“; die Funktionalität ist jedoch in SSH2 noch nicht implementiert.

`IgnoreRhosts`
Gibt an, daß rhosts- und shosts-Dateien nicht für die Authentifizierung genutzt werden; /etc/hosts.equiv und /etc/shosts.equiv werden trotzdem genutzt.
Standardwert ist „no“; die Funktionalität ist jedoch in SSH2 noch nicht implementiert.

`PermitEmptyPasswords`
Wenn Passwort-Authentifizierung erlaubt ist gibt dieser Wert an, ob der ssh-Server Zugang zu Accounts, die kein Passwort haben, erlaubt.
Standardwert ist „no“.

`StrictModes`
Gibt an, ob der sshd2-Server die Zugriffsrechte und Eigentümer von Benutzerverzeichnis und rhosts-Dateien prüfen soll bevor Anmeldungen erfolgen können.
Standardwert ist „yes“.

`LoginGraceTime`
Nach dieser Zeit trennt der Server die Verbindung, wenn sich der Benutzer nicht erfolgreich anmelden konnte. Ist der Wert 0, so existiert kein Zeitlimit.
Standardwert ist 600 (Sekunden).

`Ssh1Compatibility`
Gibt an, ob sshd2 im ssh1-Kompatibilitätsmodus arbeitet, d.h. ein Client, der nur ssh1-Protokolle unterstützt wird an sshd1 weitergereicht.
Standardwert ist „no“.

`Sshd1Path`
Hier ist der Pfad zum sshd1-Daemon anzugeben, der für den ssh1-Kompatibilitätsmodus benötigt wird.

`FascistLogging`
Gibt an, ob Aktivitäten ausführlich mitprotokolliert werden, was zum Testen sinnvoll ist.
Standardwert ist „no“, diese Option ist undokumentiert.

Nachfolgend eine restriktive Serverkonfigurationsdatei /etc/ssh2/sshd2_config, die ausschließlich Public-Key-basierende Authentifikation und kein root-Login zuläßt:

```
# sshd2_config
# SSH 2.0 Server Configuration File

*:
   Port                        22
   ListenAddress               0.0.0.0
   Ciphers                     des,3des,arcfour
   IdentityFile                identification
   AuthorizationFile           authorization
   HostKeyFile                 /etc/ssh2/hostkey
   PublicHostKeyFile           /etc/ssh2/hostkey.pub
   RandomSeedFile              /var/run/ssh2_random_seed
   ForwardAgent                yes
   ForwardX11                  yes
   PasswordAuthentication      no
   PasswordGuesses             3
   PermitEmptyPasswords        no
   PermitRootLogin             no
   PubkeyAuthentication        yes
   RhostsAuthentication        no
   IgnoreRhosts                no
   RhostsPubKeyAuthentication yes
   ForcePTTYAllocation         no
   StrictModes                 yes
   VerboseMode                 no
   QuietMode                   no
   KeepAlive                   yes
   LoginGraceTime              600
   PrintMotd                   yes
   UserConfigDirectory         "%D/.ssh2"
   Ssh1Compatibility           yes
   Sshd1Path                   /usr/local/sbin/sshd1
   FascistLogging              yes

# subsystem definitions

   subsystem-sftp              /export/pub/IRIX/bin/sftp-server
```

#### 2.2.8.2.2 Aktualisieren der Servicenamen-Datenbank

Hier wird vorgegangen wie bei beim ssh1-Daemon, da der gleiche TCP-Port verwandt wird: **2.2.8.1.2 Aktualisieren der Servicenamen-Datenbank** (Seite 39)

#### 2.2.8.2.3 Aufnehmen von SSH2 in die Systemstartdateien

Hier wird bis auf das Startscript vorgegangen wie beim ssh1-Daemon: **2.2.8.1.3 Aufnehmen von SSH in die Systemstartdateien** (Seite 40).
Das Startscript für sshd2 in /etc/init.d/sshd ist wegen eines anderen Namens für die Prozeß-ID-Datei folgendes:

```
#!/bin/sh
#

case "$1" in
'start')
        if [ -f /etc/sshd_config -a -f /usr/local/sbin/sshd ]; then
                echo "sshd service starting."
                /usr/local/sbin/sshd
        fi
        ;;
'stop')
```

```
        [ ! -f /var/run/sshd2_22.pid ] && exit 0
        sshdpid=‘cat /var/run/sshd2_22.pid‘
        if [ "$sshdpid" -gt 0 ]; then
                echo "Stopping the sshd service."
                /usr/bin/kill $sshdpid
        fi
        ;;
*)
        echo "Usage: /etc/init.d/sshd { start | stop }"
        ;;
esac
exit 0
```

#### 2.2.8.2.4 Aufnehmen von SSH2 in die Benutzerprofile

Hier wird vorgegangen wie bei bei SSH1, da die gleichen Pfade bei configure verwandt wurden:
**2.2.8.1.4Aufnehmen von SSH in die Benutzerprofile** (Seite 41).

### 2.2.9 Kompatibilität von SSH1 und SSH2

SSH1 und SSH2 sind aufeinander abgestimmt.
Vier Fälle sind möglich:

- SSH1-Client und SSH1-Server
- SSH2-Client und SSH2-Server
- SSH1-Client und SSH2-Server
  Sobald ein SSH2-Server einen SSH1-Client erkennt startet er einen SSH1-Server, vorausgesetzt dieser ist installiert.
- SSH2-Client und SSH1-Server
  Sobald ein SSH2-Client einen SSH1-Server erkennt, startet er einen SSH1-Client, vorausgesetzt dieser ist installiert.

# 3 SSH unter Windows

## *3.1 Kommerzielle Implementationen*

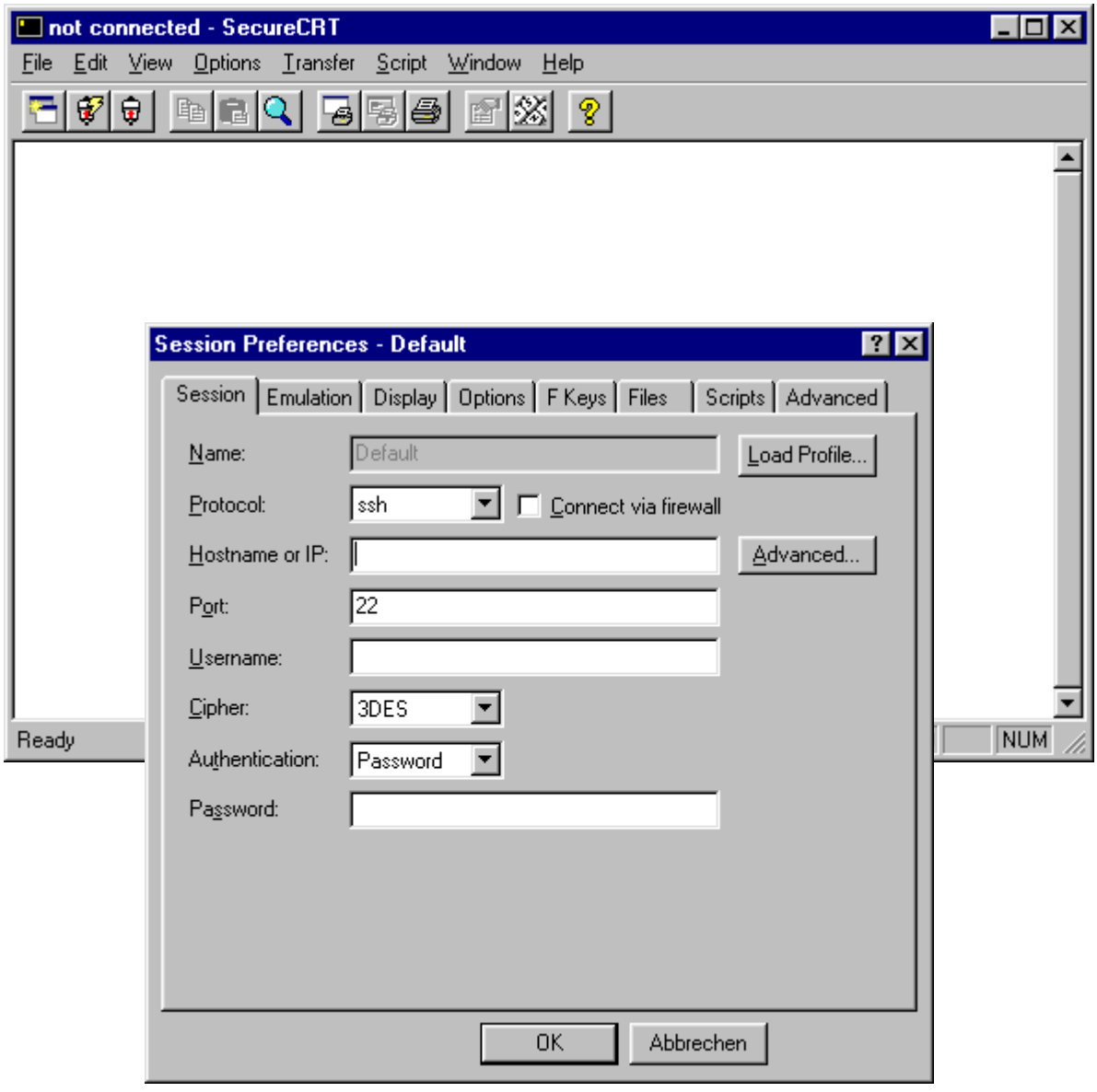
Es sind derzeit zwei kommerzielle Implementationen erhältlich.

- F-Secure von Data-Fellows

Von Data Fellows ist eine Implementation namens F-Secure for Windows erhältlich. Der Preis pro Client beträgt 99 US-$; Bildungseinrichtungen erhalten 50% Nachlass. Unter http://www.datafellows.com/f-secure sind weitere Bezugskonditionen ersichtlich.
Diese SSH-Client-Implementierung ist die vollständigste und sie beherrscht sowohl das SSH1 als auch das SSH2-Protokoll.

- SecureCRT von VanDyke Technologies, Inc.

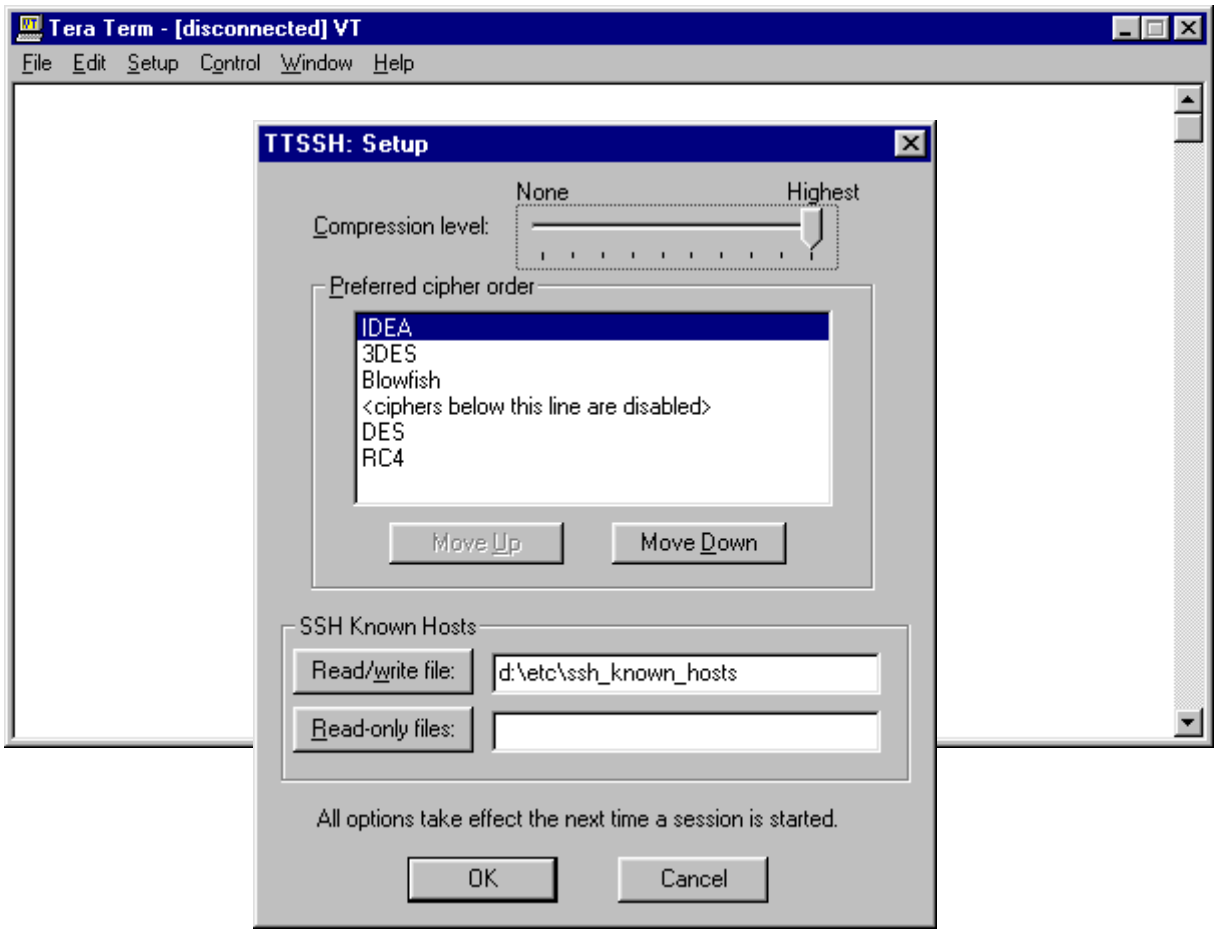
VanDyke Technologies hat seinen Firmensitz in den USA und ist aus diesem Grund mit seinem Produkt von den Beschränkungen des Waffenexportgesetzes betroffen. SecureCRT darf ausserhalb der USA nicht genutzt werden.
Dieser Client unterstützt ausschliesslich das SSH1-Protokoll. Implementiert sind interaktives Login, X-Forwarding, Port-Forwarding und sämtliche im SSH1-Protokoll definierten Authentifizierungs- und Verschlüsselungsprotokolle. Die Terminalemulation von SecureCRT ist sehr flexibel.
Weitere Informationen zum Client sind verfügbar unter http://www.vandyke.com/products/securecrt.

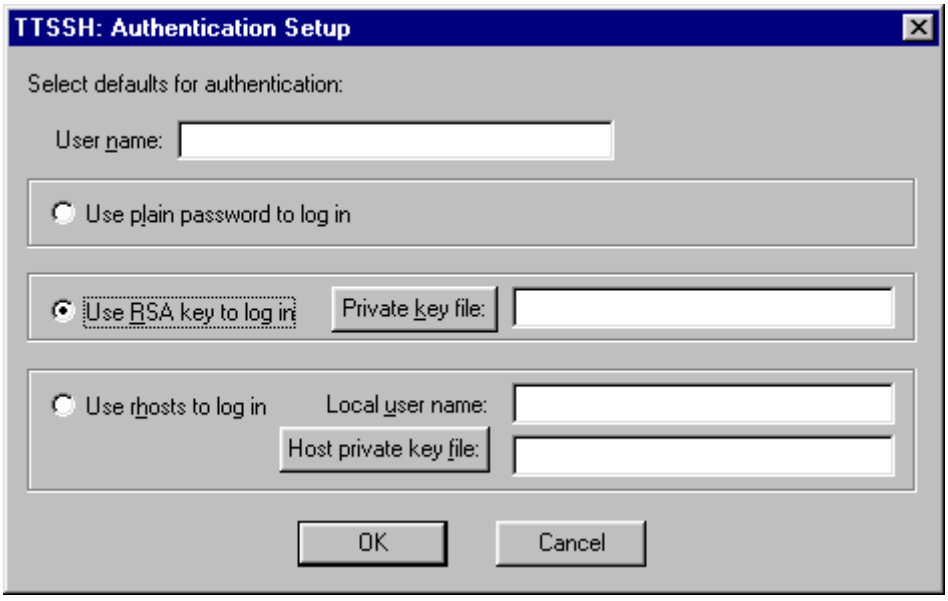


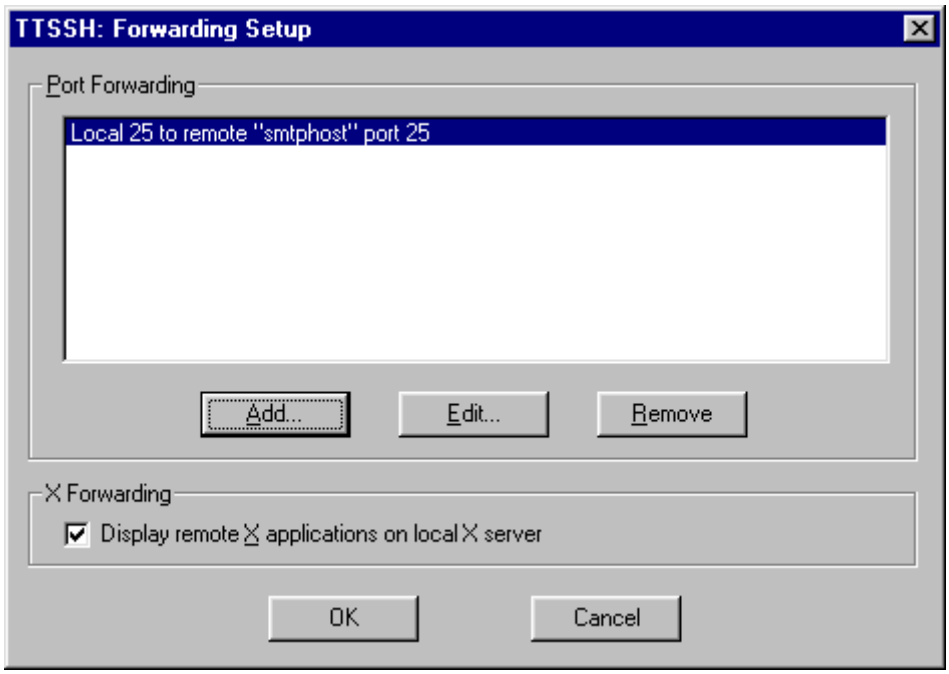
## 3.2 Frei verfügbare Implementationen

- TTSSH von O'Callahan

Von Robert O'Callahan stammt ein Add-In für das telnetfähige Terminalprogramm Teraterm Pro von T. Teranishi; es kann über http://www.zip.com.au/~roca/ttssh.html bezogen werden. Dieses Add-In erscheint ausgereift zu sein und stabil zu laufen, es ermöglicht ausschliesslich interaktive Client-Nutzung als SSH1-Client. Sowohl X-Forwarding als auch generell Port-Forwarding steht zur Verfügung. Authentifizierung ist über Passwort und durch RSA-Schlüssel möglich, alle Verschlüsselungsverfahren werden unterstützt.

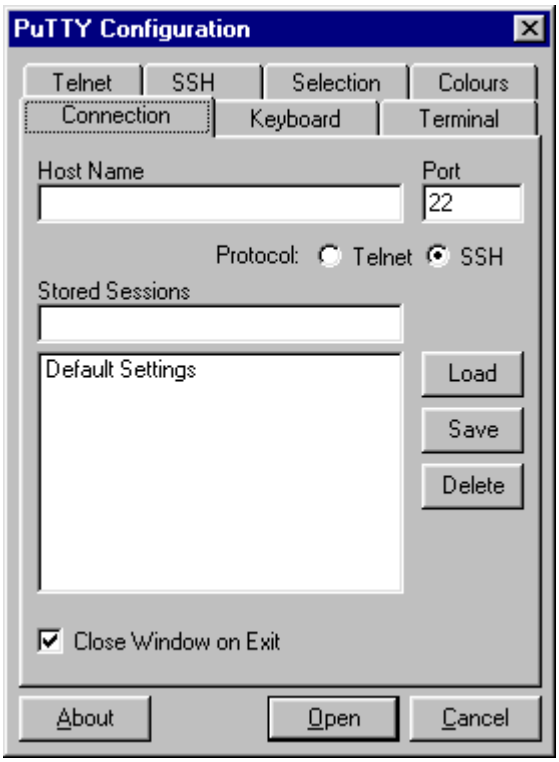




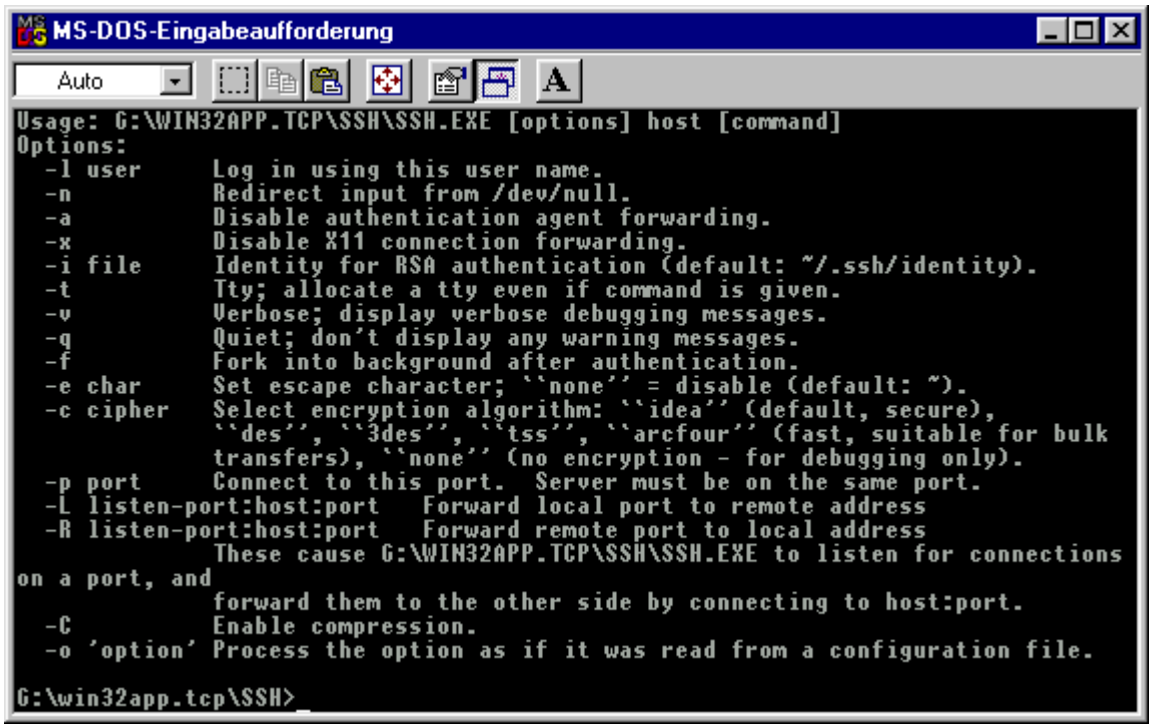

- PuTTY

PuTTY ist ein SSH1-kompatibler Client für interaktives Login. Er unterstützt kein Portforwarding sowie keine Public-Key-Authentifizierung. Verfügbar ist PuTTY unter http://www.chiark.greenend.org.uk/~sgtatham/putty.html.



- Win32-Portierung der Unix-SSH1-Clients

Von Sergey Okhapkin stammt eine Portierung der SSH1-Distribution (derzeit leider nur die Client-Seite). Die Portierung baut auf der Cygnus-Win32-Bibliotek auf, welche Unix-Systemaufrufe in Win32-Systemaufrufe umsetzt. Die Portierung ist verfügbar unter http://miracle.geol.msu.ru/sos. Die Funktionalität ist identisch mit dem Unix-Client.




- MindTerm, ein Java-basierter SSH1-Client

Mindterm ist sowohl als Java-Applikation als auch als Applet verfügbar, unterstützt interaktives Login mit Terminalemulation, X-Forwarding und Port-Forwarding, RSA- und Passwort-Authentifizierung und alle in der SSH1-Spezifikation enthaltenen Verschlüsselungsverfahren. MindTerm ist verfügbar unter http://www.mindbright.se/mindterm, ein experimenteller Java-basierter SSH-Server existiert ebenfalls.

# 4 Anhang

## *4.1 Abbildungsverzeichnis*

## *4.2 Öffentliche PGP-Schlüssel*

- Der öffentliche PGP-Schlüssel von Stale Schumacher, dem Verantwortlichen für internationale Versionen von PGP.

pub 1024/CCEF447D 1994/07/05 Stale Schumacher stale@hypnotech.com

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 5.0
Comment: PGP Key Server 0.9.3-db2test

mQCNAi4ZLEQAAAEEAM30ebZnCM+tHtWwqA2CIBs607FxOYnU1yyk3mUD5WfbzrzP
O+VPxICI323n/wUunuJdYg67HlrAvvgj9pJi8ImNFkzROSR4J1vR/zH02LNpVPzA
gb1CIaCK0Q1RL7ynWPnxdWOM1gNoGZJ5S3YdUhxuK5pK2oeKjbCfd7bM70R9AAdD
tCZTdGFsZSBTY2h1bWFjaGVyIDxzdGFsZUBoeXBub3RlY2guY29tPokARgQQEQIA
BgUCNhz2HgAKCRAAF/8DxLpKtn0zAJ9N3zxnUxuKzmh+yIPPdRmpvu67gwCfWM56
5ydEcHNDpO/4cYrLjgIcfX+JARUDBRAz8eN6AfnWgrAfLuUBARFeCACYfP0O1HMm
GJiCsyu+2D4ZpMARLy/pZ8eT79YObCLRhuC0XPsMO4+ny96mVbNAyc9LnAXsCiIN
k+eEtdvihuzeDHCrnGaRpUgFr7kbeFlEzJn/zYleTp/hgAItuYnQcwv/+brq1y0m
bve2hgod8c2Jo+D74y82LF0q98CP1zxWlxmzMEaleSUNqjBF0zxvTU9NiCIKPYJx
dmetkTYdneoX3DBSfwQAsLnBrXAbxDT397xdNA7EmAfJei3U/d061YqyvPhvgguc
9Dhxu/SDtZUHcZB5poM7KWy+Uvfk75MIJkL6MOfhQPOV2uLY7MiQHelAXV/JUhJh
55x7ajnKXSCYiQCVAgUQML+U0w9HL1s0103BAQHo5QP9HFtIyRwt2t7O6tbSLdBc
n2ZrYg2QkPfVUi3ndYgi9eyjj22IKyFGJwdyrDZNVdAtrDXgBZDvQkloeXIBuxsJ
kKFiEhSTtv0kYSSCbBNtWz/Q9Pt42rDKR8lhxA+4WKedUYGxnv5nB+f/6Elrs3Zl
4f+mnCbjswbQs3gFMuI3/sSJARUDBRAz1s5dEkNUiIKqs/EBAQ3TB/97jKVLkYhh
tLFCc2cR8mlwkLm+hjpwt0MxRAo4sUwJ8eLU25rrbb6usCW12gK01/3AO2ZX4cWF
D5eMOwFY6kiHDhPyU8coxrUoNP85q7x9+wOc2ttG5CJmSvWuOBMYuI0AMxJTl/C/
zQk7PUoY6eiAe7iq7wKBx4wzEdZcHZ4EDL7UdGt/WpnxV6cwdHvUsad4ednk0Vbt
ozqy8Xw+W8TF46Ap4X81qcOgtzlPe68spjXckanvJgdrR9Fc1Ldqt4u+Q0PTALBH
nLISD6LIioCz0RFzQYSldCKyV8PsRX8CnEByP6nnmgt4CSSfIjEUi7dT9RZBGV5Y
UxhAWoxLSrtBiQEVAwUQM95PbxRVSSgCwG/BAQEdvwf/TiRKyK3gdA5fxyYKkk3g
L4ZEvkLRqw+ZxJHKmG4acFjcF37i16gyz90wydJiK91OtOGKOzdE8pmLzTYQj/57
OAp30uk6+OBdYLwT++xXQK9jKtDM55BOv2rQ3D9ZX58kSlCU+ibtWiyD60PYvDLM
1ZEjwezyn2TvAWctvfN44bkdnz8DRwDQM/Oi74yXlL8QgjdtCeN8OrsilqGMCC3r
IEDnECVVTTyRmozkoZhbUJJuXjsk4zTyqdqCtE9SDsAoo24FFhUUPvVqw7LRDD6/
dMmeEM7ni5SqmlVMv3hzIfOgqtxyubgaGqik/VuWrLC+82En9dXu61120FBUpCxf
yYkARgQQEQIABgUCNk7AGgAKCRAazW/lPKCnPMyTAJ4ldG+7N2tJFo6gjbOOrLzw
Jjg3VwCcDFRUGpzV6qKnnMFjlYrmfnQTlsiJAD8DBRA09XVfHawFpUaEpi0RAnzM
AKDx+noSZRLCpWWs8eEE0vam43SPxgCcDtKIMv23InqhXi0GnOG9MC5zD4CJAEUC
BRAwcwVKJTGTqV6s1UEBAeTXAX4giIW9FbsI1jzC9SWXNkhMK99Q6IhW9HSIeRIU
dSNBOWu5H7qAy5JvepizrmVAtBmJAJUDBRA0x1AVMEMvArn3c+kBAYtKA/90EORb
z564Wql0rdTDx7RQQ2wPXAryD4lOgnxjcsnJd8dgZAvQsjzxGLYeRGhKLofwp0lP
9hB465QzYrI9WbySWJhgIvwHK3e5Kzx1de1XVY8jSOvYYUB3h4s18rBUxbyTgi7q
YrcJ1u0SspLXV+/ibLmtfG567BzCXdZ7DfPHiYkBFQMFEDXNLws6Oy8Deyj/aQEB
FocH/ij6XIiYSuJ+h3GRucR/KkvIi9S+GCbWmCd51u5cbDwamUg8gali45akSDYG
/gYDMINAaPqefz157qZWrPRw+LUuoMXIms3KjRWs3oeAumRGd+Ivw0lywnaIVq3w
u4dCcCd0Ksv2+1OzfEBrKe9P15+8EndsahwUV6PojU7iXp1Fjm2gQ6D7ozzNgKVq
DL/5SeWFeUssX7o1Em1X7PkNoUkqpieFv+S99KVr7a2IcX9/nVEXZEOimlaNYUcd
WHP5ushqk3KWrSSFpZ7gLd783BUWh9HNRIO8L6bGcDBM+F1efb4yGblGOFEIGuqs
LDQd8+Y5V0JmJD4BADx8wcinw5GJARUDBRAxJR6vReOW9jvHGZEBAXiFCACy3yFL
UnhCEpaQ7EIb2x04lHcAOLQkpzzFX621LK8J96DIaHT+NWcgt8/Vw/55TZ5CaS+X
qWeJxXaKFKEu0WMXo+hzJtSf0LJZZPSOv+xeFdDoErSEJwDnxXWA+tAEoqCUxzbB
VYlB6aKiSsJZN6Ye1VGC44IGP3oohNc+7EcRYvXCbl0Z5CVZt+UXJa1Oi79bwM6U
R/Wuo0+TEkO0OKYD8zzP+xZKTpEqZhmVSfSgLtfQJ1kjJlrhphNzIkpx3Fy8EToG
gspuQE9iCzs0trmOVL3dt96sfREoz9gV2cF7k9WVGRotnPbMUL5m6bQYtp+Mh4KW
Aksnba6YXVt4JfsmiQCVAwUQNOLibmHFpqmvPNa1AQFxgAP6AngYXnl5vlLI20L1
Qslw3CThU751FEaZHYyPkHEMMbCCLsKuWA+xkuWyCEmPoDCeVUalRCHt4liPiy/U
Sh/XN4ZC4VVKp0TUgXSMcOY8AbnJwA1Y9H+IAHI/ls8TGB/IjVbV1bbRh97FoRqw
8ssyvRBvCoJwL9BGtp7MeeAnuG2JAEYEEBECAAYFAjZYqcsACgkQYv34Zug1LdaS
JgCg8xtl39HsZyhEkOu910W/yBr/CboAoN72naLaGlS8JGKYELCnvhl/aWNRiQEV
AwUQM8aBFXx38cgUULmNAQFnWwf+ONeKwZJaa8jEILnPUaPh7+D/xY+zwG0AFTbU
```

jCz3VTKZncV7Lnfjc3+YYAc4R50InEDC8TwMM98VYmrW+KYjThqJSfo9yVkOp3Lk
UzEbFyy+PwawGuOHfAjAYWrU+SDYG0mf6wok3O8/2ukCMVWiMJb1Zh5L8GeTru3e
glzAXvvMY8CBS4Ez3vUAtfBmbyUAM6mIOsTT2M2FFwTpu2CE/7HkGJO5X/E7lAB5
v3XdIUVmf9pTkONkvPSvj922MSw9v5noJL0O+518jwn1pWS8NMAwkXnt9VADbAmp
HK9uERt2CJuyTTERHwPvM1xtFNVWcazw0gar3uzNAT6NneItRIkARgQQEQIABgUC
NhEwDQAKCRCKGSmqg8nAEAmzAKDOlGsHCX+qT08cQ59rzjYfu5se3QCfcoZJOABF
uD6OZ2CJLlizC1H1WsaJAHUDBRA1ERXykK+C1sFczFEBAapaAv9jMf1A6sbgUAyR
r/j/SJLs6YgG+c64QEFyIaP1esEdfdnm11uqO0qxsdBqrRdsdd3rJn44A4ue8Ol0
mxpGiZzT1V7zmnGYEAPULzD3y2Wn5hrTVDHO7n+q0mWd4A2b0dGJAEYEEBECAAYF
AjW/tNcACgkQkdEmVgshTY84+QCeOVYViY+nhbGDmd7ZgqSHLSY+eTsAoIKoxN8J
Stxmv9x1mN8zlFb53SN4iQEVAwUQNYFpcpTOrDOV3sFJAQHRyQf8CGduKRBxR5JC
UmO0H/EOvFzcbkO1Z8n3J/T5FU7Ne+HcYf0JaGh+h9XRHCDV8+8t433r+Pv3oLaJ
QI5zUnMR9YzC99f7po8WN+HVBoEcU4kWUoexcRUJna0rL1aMloAssGsdTQRoGqEw
X9MY5tEcv0hou5r+WTYnKl071D4T0ZKbtfBwu0xJ5glMEw98+3kH3YTC6hzBNPng
XrPnRkmR3SEe/LsCrLpdkvmu4Tp0cpvmtREXbXuvOMB+GSC9/mmDXN4GOaJV7uSv
u1dvCTfXA/0mjkMa6ldFH31shVINGLLi+Zp4YwuSunwdy8oKaYOgnLAEk2ODJYYf
P6ILCC2H1okAPwMFEDTgtGmWHyC53RjwChECIskAnAkZkwNSV5izV7XbcaUQ3AEY
C3pWAKCeSuW/OQRPIcbMxrZmJCmdtfTZwIkAlQMFEDQVvl+ciidMRsSaTwEBGQ0D
/jUR+yqAABQk1I0j6Awv+K7CHUtG0zUNxvd+nEst7Cvlxin/KbHLF30CHUwZ8Eqf
t3NovVzvOq7LrtWelf+TuKWTdAu5YKP75xMM+0TBpkCX67ieS89tjgzTpRGNKI+Q
jMRY+AqFCBpVnbbSkXQ/EfUUdeUNIJoUnm+EyasL8LgYiQCVAwUQM9eXTqU6p5xt
SWApAQGgxQQAt/gnpAm/dzezZoVA3BDV+Qr/Srf0MZU+aParS2M22kyC7vJY1avb
sF995Kxx4LwH9ClLeof/6YpE4LRZ8W2JSYob0JRCAb5ite/D++39wSIvqAdWR3oj
dibE3q1pAVXZn6ArVj/xOfZnVeAE9zYjU+LpVCFB6xnDgu5LjnaMcUKJAEYEEBEC
AAYFAjXCWxMACgkQqSYi8234KTkjwwCgsdqhjW4rFZL5w7ZDRubY3+Pw0boAoMHC
dUHkhZy+rmXJufmKXKtdQ+KJiQEVAwUQNXqOmalbHIcGwhexAQEHcAgAtCDpdBDC
yCCn831uzYQGvDLZGQrDbvrwrZhcfGVPAxZXDAfwvfYNLxv9N75XtCMQ0rxd/gta
L3CGM0Ro1oQSSLWLvk4/MbZ4jfziHz+I6ySXpcvuyZfGlS+rdhydMCvMzk4G8J6f
4qmC+Bg8Euv1aEbFf/zJQDbq7POfdc5c/VQb44De/aC76aRIhRc8BMCRE4PtRw2W
FXDN+yA4rJ/vNaJ9pVaCQWH+FoDg5sY117Iau6XduY6G/TDXLe5+QYkv1UOpuIEE
iHmnfrGDnbMDOUzGGeZl6nXI9bh0ikspxzY1H4SG9SqQ5GhfTJzkMqmVgCsG1SlC
6lKbsp+NRfOPyYkBFQMFEDPl6+CvvFuXXZ4U8wEBdLgH/3L13G9KHSkNfGUcY05E
Qg0/yDUcaeNlVKySggNnRjhkf7LO1Bqm5Anh2DqBb6T2zssZZ9ov7EnZYHixOStJ
h5lWRXJseD/SlRfEQy3adMlziGPEzyUW8RqL/JYvb35Q5g3gjMDkChSEj1m174PN
y81e/QCL4KNoShv9DwSclEjRM3jIs0ZEwJHnR/Jv8KgsXinnUhMlHBAp/FELEL8s
38cE5vwOrx2XOiFuKd9D2o0OsoA1UX3wQxG9NjdqW8QEWsLEioDj8PBTmiq1i2bS
Ehx9pQkxeytWU1TjSbE3jPQvu9FRZ+sB7RUwk952VpnE5XvrwJkPemuplXovSSiH
TsuJARUDBRA2UCursI5kAUbVbjkBASUoB/wK5ozXRUi7HhTqNz8JQdMyGooIV0of
jMgmSoCuSDCsvpAtLlO4N19FtSBAbEPJiTdsepl8p5sY2HzzE3FeV+nrCoFbCv9g
8jxKqm4SN3wDdg6G/IY0L0YrPQZkQg6nZNMfm5LGJGp4LlRseH/iz5mh9NCsIi2d
w1MdOkIIX2bVfv3MJXelbCPoDunc91kIq0JGXWI46bQWCbs1tjFAp9LEF5ha/spi
o2zCNtUaupHINELPMWo+r5+s+IWzKgon3oufgz53NGTrgDIjITuQTyb90W2J/b8U
Hh1e0geZH8T/kTVtCVuc2QZH+B+LCVwNcx6z1EwP8426wcoIwlMJfNToiQCVAgUQ
MHHQ67Cfd7bM70R9AQGJ1wP/S/Vn6R9r4Y6XKHC6pMWZo3Z6faUK4Fa9JdukeAxO
KN/Jbwjj3L/NL9Rr9h5QRokBf2i5ysIDRMBDTTk5IdUonGBlOtijifTy2qQWtlT6
NarB/g31QdayATjE4j77u28glc1SQohkkJk7basbTk1YE9EbRVwz9qz48d6uCQIi
yciJAJUDBRAxS17KsTAYeqTHYN0BAW8NA/9HT8R9l1bqg1u92EFhCtekQtJrk8A9
KLBPOCPxby8fPGWpj3qnZggy1QYj9+1NV5+8f+teIiV25djBPWBGUxHzuBiDcc8O
eZVxRp11UNlWwuY2vPS09nEteguEV3xReY/KEfTBvFzV1kqvv7C87lTK/iFEx0EP
8mNUY2m3MtTA1IkAPwMFEDRDB86zJ+d9ZDd8NBECOI8An0d9Sy7CPRnmOEMhMSF1
aO6+OUEtAKDLJBlkfR4m9QHB5+ZUuXqdmLlrsIkBFQMFEDFLXrS3A6DBk/WgvQEB
TU4H/AjX3rU2x+XnzbTwr8kKPXWpaQeLnztfOCZAxdOttV5XyejwesDbalQO+owE
pZN1K6utOHEreb2076sNr/yaJge+sy6YVmXkE5AM/4zL8B5zuFIv5tX3E9rZj4go
3CixqsIH1eywsEzkGuUcrWn/fUdXWGfNIU4nz1vGvJxOgIk0yyGksNYIZSli8LQt
+fE2Ifpm/E/9ndfs0DtTa8pL0VPEatrQ787kD1YRVVv3hpDhrM8lW6uiuyd2m4Y/
GDcIazDHkb0PgQ8paxod8s2j7WN5JDa1RdaU5wUfvYZC+acylwpRqM1+0FyNOjQV
2NewIQ8TP1wFsPL81nl7nGxUf7CJARUDBRAxS1661LJXwV4dg7kBAaXxB/0f/P0V
gfteMvpyaybwkSDsZwXxJuvCduA+UuQ84fb1ACAyyNqhBVhL3hLlatbg2/LpnJrl
meooRknAKO/2MI8PVPN/Q+5mdl4Ey+nWC0RZV7aMFjlBb+DGD8vvyVaWTUfTINpA
p8r4fal9nfJ/n0CcWOtKAtHvBeRHy6EaN5hb6WZTKvxk86/rNSWfFBSF+QMfHJyp
snRdBITX7taQnMHqB+WQlDf6sK5lG2uIOV+qSqliqYOloIHp2t4on3hNNN2G/Wyo
JR3UReoPyCgHzxrGRdCc5udNgNr4NX5sJIgw/o5klpMwNvR/WnOO/9fBTm6E7cg1
axgLQDbWMBM91E7NiQEVAgUQNEn4otUpBE2jJN6NAQHGkAf+JofqiorfWNj2JARS

SJYlE9OgQt7qMJbOF3Q3U1OaXBe8U33ExuEuZJDlOaUe2QWb1L9b1ztRq/SQDjp5
ivinsZAXd/aiNxUsfcKJZT3UmOu09ztUAvClnlvFh11OB14eD/VQw160ENms6u9g
CW4bg2Q0Nv+3plRQiy7PoJJE7yj+BuZvnBdxmiIwzTIKWJ5w98nRTzHrhFS7XEd+
xj7yXcebvw5b3eStWyydU84Km3PZA5mjYw5gS0/UGlZeyNsDvdGbc2VN1AFYw13p
FMbtBCInmUpyXTvZWXh0M0prxdl0OjfY7bPGjGx6U0u7KY8CNGHMLKE8sHdqH88u
geQ45YkBFQMFEDPGgOnfV4Tyfr6dQQEBMx0IAImNcO3Cn5TK+3dyda/jwJQNnYxp
umuk33NOy69h5Kmz+rB0qgdbZ/FWW/pamz8PZei5W6Hw6PtbKBs2TzL9Haj06ERe
esMeecTQaF5HNIZX4/4Zd+ZwTYV2Z2LdEVqIyUeAGWq+u2YPJUSrPrTIqLwIP7lA
O1jtOCCjggl5pZMuiJHEfznSRdp2TRyN4nbHdDB9bzVPK7XQVci3GUM2fuu5oRkk
1e+AAdGKxcbf+J9w8VsfY16z3jaCRAqfqTE9TBzBj6dZancGZRrT9DBPUVRqv17g
5hbWQRIAdaRQ5N49dqOMnPHvNUdALuHuL4O43lb5/4zkidjrYgw2C7OLIQaJARUD
BRA0aIr64XrSg7CHEc0BAUtWB/9g44SKPvcK1BI//647+enopMTUlR/SdHTLrX2U
4wyDV9fBvuZCmllbRLI6MZSISZqUg+vk4qglftzRhMCnNfaYhlqtdc5+K2oJ2bz7
+0GT1g+S8Ho2aFkhEbF5zAB5BhTh4Rl6VrDx/vJ56N57QNJnB7Th5fhGHxIAHn2X
A1Z+HaoCqRMxtY6cVBiFoLoetmpGQKvAgHvZqMUVR0nG3ebUdpDb6JxhR2hgdSik
iCNFSUa9ztlghs4AWys611mevzECTUyCHVdS5FTctRMgx2V4SNQOAUDF0MlrLhXb
4aJblvHLlQrsEU/Nq/DjOjc9K5KGrW6BBaxwBFU7aBao3rABiQA/AwUQM/+e1uLc
1W6tP2paEQKaCQCfSvzAAjU/CjOo980u/ImmaJ1DKXwAn330AqBr2GhwEMV95xYh
UYsJTyeoiQBGBBARAgAGBQI1OfHIAAoJEONc+rtO8h9kNQUAoKCutyXMSpAKOzpt
/tMu6S9gK6eqAJ9J6gtuw6c/KVm5ss20XkYnuicxxokAPwMFEDW68lHxWNBs+r3a
5RECB8MAniu7gxRCsr22S9tYepg9/c3dUSiQAJ949/FLQFGr7vmQaxUHIFLMkINT
bIkARgQQEQIABgUCNNoYvAAKCRD2OV1lrEl/2QAnAKDyEBVQ5rnj6XcFb5TCpIT+
pBGNVACgs8jRO99uRJ796UPr/HFYvKIhLS6JARUDBRAxS17A+40/h3clR4kBAbfy
B/wOgP+KdqTQdPIBwbSAF+l2+yfUC7fkZuVSemaBauJ3/TsXp+HtJJM8ax4RKUY/
y/1DrWJN3gned2YCHCYGR2mfaxJ/V4IJDCxJvtY5BCAkRJ0s6liw3p1wW9zHOzbB
UaJvn8uYFLFMtCWKwzSIZRk02Euz/pxzXpgAKNm1EAANdNvpm/3PMduMVwq0zrOD
59M/orNPUzj+xr4YRhYPQptlHDqCy1zEcF1mO+IKaaq2V8hXTjnE950ByVBIw78z
VBOmiXjrbR+mcOvhm53gbW8aJFh3h5pRBGmmig1fh8qCk73b9yDoTptEtU3ilJQu
Q6PGrAEfuquBDRdFTk9z7q0QtCZTdGFsZSBTY2h1bWFjaGVyIDxzdGFhbGVzY0Bp
ZmkudWlvLm5vPokAlQMFEDPuALYD5JjlocK9FQEBO6gEAJ9gB3KtsAsUC0CHM5pI
xm+UDBCYcP/F6DOQ2vWCxOJeQH6UbTajT+92URZE3dIl43Tk8LCs3eakzRA3+Eep
XglQTaEqcumWIV2GVgwTAG0T2ejW3sj/XPmtaKJE5uKKcQnfKjv8QjqWA4bbBEiU
BrO+LtRKq2eExaTuRheYjXzgiQCVAgUQLx3tYRxiuZNyWu6VAQGZ9gQAtmfTFV4x
7Uj5gdOuL15DARGIjcyISsCka12405M+BvOoTeZpRJk1t2jQC88Y7rQcKszSMSFT
XmZuB3v5hFe9KVPnKQURc483hvcNaq11vtNX+0Qnaa54u2maXsJiKIq4D9N8zXu9
g67spDk7c1G5e/sYGJfwtKE4UKDFttOsk6eJAFUDBRAzRVj9HtCXpnpKVQUBAYPP
Af9vX5tZYCQp+Ayhz5YmIukji0EoMDWkqT5ipbwUIk7cvg4YsUJa7baShDKIkFAw
wJVan5jvDRfT1D4d7I9RR5hjiQB5AgUQLhmFgh9of+z+sPvbAQErBQMgpjJPLC/V
er/qZzlSIGhIbvZF5lGJIxfv3gRtLIlzsrcEjgUOFP/q8wzUynISCDuhaw3XF1L0
vPRLR075bzVDwkOTN6dJiPxnTkBdmfl583ZISQB6c9Q5wTMj4c8CCZlEVgi5KokA
lQMFEDBXm6YwpdMBYpnk9QEB/cAEAMDjqio9ExTjbu1pL+uFg/mlj71QAI44otE3
T04wSjbQWhpTsMJt8ovip0oj5wk6jZjpPYhatW/aNp94txAsyEIRee1pvYv0ds8O
nmoax72G0MaZ52uz4NeaMTBD2GNsK6p+MQtqrUouMwbTrGuX/HxSV4Kf+7oRAWqC
sjUrfABWiQEVAwUQNc0vCzo7LwN7KP9pAQFRrQf9GtN0sgsEjGa5NJ6KvxmwjYIY
PV/9UP+VnLa//hG7eEOikQFWnYb6LLT6e6U08TH1oy3eMLubdnf/e7/htdS+cyjW
yCvQviK8PiGaesIhvYjQ6+1BHhzKmjqtKndp3znmQqqwM8buqfzRZRVhJzDuEnwg
WnDhKoMli8IUftgA1TKKBlERuYbNYdRQFAnJN42dECZ9w5C/dhJgJRvPg0j7Bx7V
Yx1RxyYzTgt84I4xWwahEJpbdYhv7WOQD/EXacIpD/idIPc13rNhBFLtizRpdwDN
2I8thmQrZAVOWY5ZkL5jSopX5aB47d9Q54dIPakP1AVw2WNT0zOWAOomMp2224kA
gwIFEC4yilRGOV7kLHFqsQEB5yADaQGDwVqoqg5kwj0deNn5t/dBHQkoxS9dlY+s
kT1vVesvC+tyLZ4U9knVL2H6+694DvnCkSmvXUYhfdQChHFFBBK0l7xqIYazk5Qp
mjG3SSSUrrpGwWJ8d4Yi5e+wGdQ0H1GkK8WqTeq5XnPVjS1tiQA/AwUQM8TFzExo
aoAIxZMVEQJTAgCbBC65Z5ZmV/+zjV7f0qP4dqa9IG0An2wkKE6+IvM/IxzAaRTN
Of1V5OsZiQCVAgUQLlkaL18k3sEYI56RAQHzmgP/Z6X2oEXaCxSeuqFJgh4eXS/+
/FK1/AYRt1xn0v68431Tm6F5mwE1z/LVFY74HhYHs6cNCxO5AJfWtO+af7ux1Ig2
lat5t7J733HAOQjaB2XyEDxczJomLuwyR83k1gOXrrmm+OGvbLU1uXkam6tqydre
fEXQ4ObMX2Upfz0yStuJAJUCBRAuugoeX8szs78XIEUBAR3pA/98c8cXxdgxcRXH
RgZE7qHp2TCFpbCLrpUYXQ0l5ts+O2yH5uvi01bpebpnSdvVVu02rhaLSWkOpkbQ
TU/GemzlAiqtNZk2hfals0FSR38yNyhxD4J3UEO2LB5GVBagfEmrw2uhTLJg5mrD
CO+npbmLY7vl9zMvv7sUqnsz3LjuE4kAlQMFEDRGH+VsodCHBnntkQEBI8ID/2Uo
bbG80tsHBlxjJliYQe/Sdb0oIsPplUbiJz8ihmd2ZBszuAKnXYMfcP4KsxxnrTcX
WC768OxH1niWdMXl/v2JTO77blmA8Tc8RfQNCqsCjt3Br9B02n4UARFC0KrjTHF2
WnQBi7dRorcCeZeZcOAOe5TN+hLAW+DTbHE29KXIiQEVAwUQL0DSP3bPiYHEALhF

AQF+cgf8DhdJOdgYtac+srheMl46mA2HM3ZKBE0SZn1JF+cdoABi9XhJP07Zo36Z
BmDo4EqnWBLzqJzGUMBH8TScUt9CNpWl2eyRofBbDr0dOAtetvcIwa5/j0x56FY6
xMxBReT7NoVSjwAC3d0XeWSsbBubIwz/57nmeNTgJTD+1m7/hlzGMh9OLYQwd2xZ
gdD1ySF6jvqq9zbr+HVo/9nZ72Q4Y14T/97B3jJsTKyeCu3WJOartK28PjERJdy4
hTaM+VxvZrJ0oyQ2XAUZBxnhNbiqYcMnZ5ePYf3oD86mPnSCYIi6vFBiIuL1XB7T
qiHwXHFe59yq7hb2Smar44ftIW0AjIkBFQMFEDPGgXt8d/HIFFC5jQEB1qQH+gKS
aOvUXyE1IKcshE3rf7MVrMdMrPTBWLfMYOkvG5I+cCNnplQiR6dwBZKHANoLW7hn
JfWVR+BP4X5zOlx3USmQGvl8yXdFVBelmt7mSKMkVAaB9La63PsxZmJtckSP6YYe
pY7/1/QQQzG2erDJjV6ApK+C7P9bauIPmEIseLaaUgLYjqBeRVNGQC/kchQMVZui
Fon3BjpUbNoMeqXh1//RfmdFAHrDlkDaxtXXLhWiw8+L1sk56uh9+FNJkY/sPq6C
HJHSJtMINhKOyhXQT39PEzSg3+elVT+TPsRYyWdGUkIsrV+EENfBZqsJC1Noi55E
7UxLJOn6jtXgEKYq7CmJARUCBRAvHew5f+LLZh7yXLUBAZ1iB/0cmc/OC1KC+fKy
SqQOnrRBaoKjy+nM+qVfEM4cyyW1a+eCHlXZVLLQt/fYb0rcKfaJ17hIMglf4koD
pHhHLvGfsA7bJKHJFT362ttAdzvfdCLLzLqh7tJBA7XAvTpKDQ3gf/5Qctoih62G
JuA0w/oa9IxS95taf0XDnoe6dHEo3pd58apZLAHSwEdhs8hcV0No0aBMJtWqG4vU
KYoow+dORsq2PwWOxZdHwLGIvpoVt1i6A840wsIE0baThkT40Ecn69AESoYl3K6y
UvIf6R0YAI1Xw5J6tKgEVVdAOoyInmqFUmogI+XkyTx0lEkwxXMvoTAELh81Jq50
xK8vEeEJiQCVAgUQLjNxPYjOE7fdHoSFAQGeHgQAnkpuURUrYPq63GdnnlVJJZqB
LGppC8706LcVI7LJYH+ySapWq2mcIRjdfh+b5Je/g4lZA+GODajRmZ7KAS6k2vFz
sDOjagPhKppqm2eX7stmASgd4m1ZpvWjv18AVVLmiL/ZZvOnNTyu560YKvL3sopq
SSd74jx7yfuWe+P+JSWJAJUCBRAuG//9jYMcr6dwrnEBAdi6BACVcAArEq3xnQJB
X2W/v2FWbxfuuoclkI+Gj+C8H7XsSy+Uv1ATROlwuRBJShJOnfkhSXRvHRP5PWyt
qH+ZieO2qSAYBEVQASBW5yCrtFqtfngdHdVBzBrFY7Bb/gjxfMJ5Vlpd7Skgu6ff
pR7t60faovflSjvZz4NCUAy2blRT1IkAlQMFEC+YQ7mRdmVrA2lSyQEBY5wEAIcr
oAi610y4a2zon9dd8hDNpyaMiN2QK9qVWhCfq6N5VRTiPrdkFu8hpVcS4GdxKDBk
H4Ef/sqVHJy41qZTqiq6ZC6qMbCwFdFMYPJ4a+kszC2BZUvWP68WlbqFCBoivjYy
zZe6FbN0TZJMiwugaqA+naEAa2GtnmSV5n02zNosiQCVAwUQM5u8uZOlvCueyEEp
AQE8LAP/ROrffrjcDMwPFr6nSnr6RpzZadDKpr8GW855N0ZCA1AW2dfq93tzrrCR
gx5PncZtHepjXJi42qw8O5YwS117S173/AgMR1PL+3FW1f6XDV1p+8aOjT4ZKc/A
Xm+jB1fnGKnPERy8NsuNyozzVyofjA/YmnL3+QDYD20imZgdu9aJARUDBRA1gW4j
lM6sM5XewUkBAfJRCAC3MtiZ1Aq6/ClmXZ4J0HfKWj6MMMua+Y/Vw+kO52WsmmKG
dJfE+vT8LHLUEEyRyWRN85PuBGXESnqthW3mgQrOHU2GbDaLCj7JUM0gjJIAfLAM
u9nIS/Jqm5dpTOvduhIZkjbo0f/UFME8mI/fhCSSmL0SE7jA6vroqa+Wj2cltmHA
QpDPWBTJm0jgY9jpVk1EPBDwtPU5KjfOLuUD/XwrQ09paSPAjsvLcqdrUMxZ/oiB
xthtQ/D2U347EJ/wtvMlwdNiUkgMO9o7VLuvMI+i1X8eGXbmuDoOYSzDc8nsndkB
L4iS6Y8H5Ji9HM80tZJUHL1t15Oy/AEFPy+XvvM6iQBFAgUQLx3tupaoh6PXN119
AQFE2AF/WdnCJ2DX+Selpg6nv8nWx31tn8e11ymxlkguHybC6TN1Qc8jog2QhaNz
tDMm/qWxiQBVAgUQLx3s6Jhky0UpTcvRAQH3tQH/TD4kkfHT4fZnpMobfya18oSl
6JiO4Udc/wUjef7HRLepz1vLRWlRK32/CYAOE3biUqMM5mepC3kirUFk7a0vYYkA
lQIFEC/wiEGlZOCuUMiZpQEBom4D/RaZAzqZAz/UTbQR+T+UVibNJu6UQtGKTfM1
d4xIyNkPz58OF8UaiXSv95YrA/RKDctEcTNvfZgrI/DEV7+YVZn6e/YKyiMrtIP5
sCW3+JjaDoDfHuPm2aZCDNe5wT4Xazbi4foQC4GxcwGIck3fY741cBqnT/aYlE7D
voOD46qqiQBzAgUQLi3Tr6rbQUK9OQ+TAQFaPwLqA/A4/4fxAzyqEfrBxJOtwDSO
BdXcCL6YN2wzrHbP1N9Kc7paNECWeay1N4WrSopUfl493LxYIpKfA9jUJJgRP7Kx
yxk0i+L5AyZXU1aNBC1993KDSMbbohVUldIPWokAlQIFEC4ZhVSwn3e2zO9EfQEB
0AQD/1cad3kplfEJ77Oh2bGXW4SWBpI5+ZD/jPK8kEGGxfJvy2uS0rSdGuOKhXWd
bAG+QL62rkNONhz8Bu+yFv13vvIQArZfJEBq+F9NWFA2qVkoWa3KRhmh9SU0DOMf
/1gM9HY4xt19Kv90+NzU0vnP5qm4oSkVZHbnlHuLBz8hZL6ViQCVAgUQLx3uM7Ew
GHqkx2DdAQGPYgP+Lo9nj831gnzfgg7qknreUG0B7ybHxFMkxV1WRJjZp2xH0h8A
lvjqOetyMoerLUQo1bwCywqU1Q5R+1IoeKMV5SFBeoAau0JcSPVHOITEPCK8MfQh
v4fQ+JHjrZ6Cu3XdXsJHJxD9sw+h/QoVr+49Jje0yfB4uSOIEaBJeL1lLMyJAJUC
BRAvHfB9sn/I8+ucfk0BAWFeA/wOanwUZ3zJhbQDYebGXqC+NFh9+Mi4OYt5gXJa
hGS0YUCxzHAdtdOuSCdpAqCTrHYWwO07cd824dkPHiHOlT3pVvuYmKQASfYOzf+F
Y0bKfOJa6hC8rIBOVEtKDiLkiVg3dFn2m7qTq4iuefpkiuqJLVBND+oNqJzNtSta
6fBdM4kAlQIFEC66G3mzREz3W/N2pQEBy7ID/2lomXcsGybpOAALubPgLxHVP96S
ptielKX7Ifp/Ot9TxztY3iC3gtZHIDGnhnnXKzVmOTjW+le4J5z6FmRuDYaMNbw2
elcs/ILefINVLfqT1WK+TJOg3CZH5bJCisegA19ztrX4JAitVAD0yK1NfZ3jYTmr
sxI061eud9/Up16piQCVAwUQMAOq9bWlSk526nORAQESQgP6A5HyUgVO7iwKH4zO
QHI816d1oYNzlny/l+KBZkpQqsEMVwt+/4eLVgGmhBeU8CkoEgZmM3HzC69XiiLN
0Wpjxpbd5J79n4gPHdjdIv6LFKbePp2kRI49nmtHE18D7TVdMRGBwJE1G0V5/Gxp
mWmI0YgfHCzWPnnlsYRYzCGxAu6JAJUCBRAuwiwLvUIenE6I+2EBAd9fBAC0wWa9
DX3WorrA+xve/sito+GpwKgBTkjRD1DZz3bWejiUpPP1If/LniNDOYm0oM4QmRg0
xMWC9bEapt7kJgwqm0gecPwAXNa+g7sYLAY/51yxtMAGG4IUDg204ZvqFtS0a5/a

```
NQoECQiiG+0TdfJmRnT4ZpOSSOcYj+lYB3+ljokBEAMFEDAB1PHOXvQ5gbFDBQEB
boAH0QFcnO6zATTntYvNhlJvVIKwdCSGeT2UeU9xonDo/hKUfj9gyB2Wh4uoYvij
h3AmudDbwD6iQt+NV7doRI7aYzG8HIZvl2ZNxvjLOu+7kmpm1uusLRo2Pkc2UNqx
sEzhnPDo9PCoRMiXlWf0WexbOAoV5SLeuoKLv5je+W75we2QLxizxdtjnajr80Co
NYBWqM/62F553gs2icii5QFKy62VuGMMd31PphSxZFRB0pLvzuiJvscv7X2PqcWD
DipjYpM4ZTFzGVDOQigliyQOfOXv6y6tN5UDORuWg1PTO6JP2rIFAEPbc6Ve8r1B
Nf8E6u8JDsfUa/HtbQ5liQA/AwUQM+paDNjA6VC6sIc2EQKihgCfe/npL1i0IaHB
oKHZVF1OpbPLQnEAoN7cF+10ta1RKOJ7UVnC2eg2pJ8EiQCVAwUQLm5Gsd4nNf3a
h8DHAQF9lwP/Yx7Vls4PaEjUWe+d4ZUCyutvcgPoDh+wNP0jlFx5aroUBSQFSoo0
VtOrO8zUYX2dRIJfUw+aAYrpwVijcHB/Vbaxvu1yBtUzifTwKTejLaS3bmhdbCwV
9x+oXuAQQyPtSWvoF3+3zmPO2jKRRK3Q20Bipy3AfhPmwAUAKOAuZzOJARUDBRAz
xoFn31eE8n6+nUEBAd/1B/9s4y8IQIsZZcdaY/YuNGRDqcwXcwx3U/x7dCgBGjhG
Y2G+dART4dtXC+QjRhPJYXq3Jse3StGNT2O+Z9SVSKm45IQHoer7tVomE6Z9VVYs
nJ9brncuFv2DbIgwdKd063tIOps5A/gg4xda7vcYwciKwSY4vQYSZWoVxQ3ByklY
xeB5MrI4WCDNHVQ/+8RlojgMlCf0wz1WYDcP9VX7CocPfSwe5u9I8SAbJP5ym4VL
oWuYm+v8pzr4jt4nX4i2/ql0iMGcd7dUjOCy4UzBXnXUXofdxEEL9ZibKxRiGvm4
Kc4Y8zsOK92GJdYrkRqazjXR7F5wb2F1+yRBNcvFMo+liQEVAgUQL/CIjuYzJ6XO
mlylAQERuQf+KTp2JLAnO1GWiB/IJm0RXq9KD5P58KSa4pUPGFmHYMZRUhcXgXeq
9N1qk5cWi4bkddR/4ioaEF5tySgeP7xyLuYMmSxC2Y9UdfNh0b/qkYHUlO/6sRPj
I1LQWfMchYQwyCmUYgtS9YXkjUgyDfjEFbb2qszD1rtf9wemYDYGL6aQdUgStsHJ
+ifLwqvpOOmKyJaFtTzeqWyAiyNQFvyU9DFkMYANjBdhIVwdVvAd+5LrJdSF5Nf1
QFcbiA1WOmmcjjbX8TmKT7YIYV1UJDjdW6JwSVJjf7+GY7WbW/a1MvJdfGyN+AAr
rikvccoNx+yIHI+Xe1FeJ7m45sEtu1MsDIkBFQIFEC8d7/zpm6yJEeuOjQEBse0H
/iRQ/RkIKyRD4wQqyMYZui4ydv+9EEVsDw4klIaa76U/HCq/NATtDomE2Y3EPLNx
r2NuhNK6sl3J9pCBSE/H4g+MiKeXrV2QgQq6l0FguFTJe/O+ZcwSggsT5OLC9Dy7
FSVdF+4nyOqMm+DCVvK5Pfzz63RmqR+EB/WfEZLx8zjwV8JbqE+JyHNCzgi9Uqq4
+9rbw+T/nOanM4NnUxRrC6rNFQmJgtEz1vN4/Hfh7ziD+vPAS6+jhaPkSIOMLHCJ
8dRVYp9HAxO+iHKnpyO7k/MlyeGI4H0oafMsL10EO2mscgkmZfp7UYOwYZ5d+ZU/
IQCvAKdi6L78ZQq/AfJH+w+JAEYEEBECAAYFAjTaGR4ACgkQ9jldZaxJf9lgKwCg
lOX5krnLK9XhL/DUv5+BVVXSmCMAn3cKIgAzJkNP87hDwG67OxBfmBYAiQCVAgUQ
LnT0uPoRqumdmX1HAQG1LgQAtU0AhfZHc/x4tx6AcT/1dyHrJ+kerUBFkzhHqm48
8MW5wYnsdnmiUogPMOYbNA1ExZzrJ3+VkY6DPqn4xFW/JUZJNSS089JfZt2+EkBg
Yj0gBkzwytKb0JTsGLzjxnMpknh+HheCymcGPMqEc1kAderkQF4mc7+mrnptSPqb
eMc=
=ix6L
-----END PGP PUBLIC KEY BLOCK-----
```

- der öffentliche PGP-Schlüssel der SSH1-Distribution

  pub  1024/DCB9AE01 1995/04/24 Ssh distribution key ylo@cs.hut.fi

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.i

mQCNAi+btRkAAAEEAKxQ9HwqfsQc9apOIQmFTo2wqbCL6Q1xlvN6CjxkBbtviaLq
EgmVPnb/FGD5wwxDMjCCJDwBFfLLRwASQAyyy5RjukkZx1Gn8qHzmoyIOVTFOIJI
TFDWyVjMSSvUKACDqXv/xVFunsPlPc7d6f4MwxD1kw2BBpoV7k64di/cua4BAAUR
tCRTc2ggZGlzdHJpYnV0aW9uIGtleSA8eWxvQGNzLmh1dC5maT6JAJUCBRAvm7Vv
qRnF8ZYfSjUBAW7pBACQ7G2pYStkBM5aOK2udb/m/YAAZ/NlY2emSgEJfYrAysSY
0yfbhKGt0K59fGSotmSRcMOpq0tgTMm7lQjsUr5ez1Ra/0Dv7e3xoGQYJ8764X9w
popC+u9JuxLeGTtgWYwPUZIHFcQanZslUmCDr36kvesx/2wXBf8+StghMbA3vw==
=aGik
-----END PGP PUBLIC KEY BLOCK-----
```

  oder alternativ:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 5.0
Comment: PGP Key Server 0.9.3-db2test

mQCNAi+btRkAAAEEAKxQ9HwqfsQc9apOIQmFTo2wqbCL6Q1xlvN6CjxkBbtviaLq
EgmVPnb/FGD5wwxDMjCCJDwBFfLLRwASQAyyy5RjukkZx1Gn8qHzmoyIOVTFOIJI
TFDWyVjMSSvUKACDqXv/xVFunsPlPc7d6f4MwxD1kw2BBpoV7k64di/cua4BAAUR
```

```
tCRTc2ggZGlzdHJpYnV0aW9uIGtleSA8eWxvQGNzLmh1dC5maT6JAJUDBRA0dOAJ
BTYJQI+2vvEBAYRiBAClTijfBQe1sQ8x7eOPHpRGesI3WSRHn1HnYKzZSz57JeaH
d3R/kTg0tRuAiD3YVcs+SKToCcKoo6tLotoYczlwZuq6p8NiLi03DUahIinzl3hk
qmCf0tbg3stCgbFUrvOr/Vozt+Q+MaGXR+EWukGOE5zAQf/gtHk8+ygPBJIva4kA
lQMFEDNtE+EGcW8JGaIDQQEBKa4D/10HS0xkLH/B2BiJTaj12Wi81jnqNBXZJyqh
cPG+EWltL49+14puuEoVg+25zTNp61JWQuY36Z9QnCCefvelE5oR7IzrfD1KTdMt
TbZqOl1nW3FnazKc7/tZRipaVHFWjeLjNh+3vUvHYbH4t/YhO2IZ+6Xk29jlhsLR
A69ZXYBjiQEVAwUQMolOqkfSDRgzv/VdAQECpggAma2GOfOteg76+gyHRGMcaZWF
RUyUTDMFTvvxbxuanNoYhDXy5glW/umLXfIk9Hok/0HQg3lWRHRweOAFqNkmf1kH
0FcNFg0Q1dYtuwsvqQR6ePkrGcFDte0L1jP+J+BJ9Q5p3IlUIbgwa/+agxaLe9+H
pIJH8zdmHpI1rOlD2BY4dUhsx+/+k4qkDAFJnJbAQ829+Bsfmo0+OVbFf3/mwZ6y
B/ZkPEztmGTbzHgEcbDtfOOzMQ/Xfsy0AkcwcKN3w2yLUcaMwX5qXyidfNYaApK5
V4bS2R/m8kvbJ8K06ivUf90w8XdWAaWl5JEyD8AKUW5PFsfgQakph6/7ppn4K4kA
lQMFEDNtFBZOuHYv3LmuAQEBD50D/Auro2k2ub5tWYrK7XFgaa68W5DjnN3/E8+C
832HcGOkVCy/USHSJCSP76DXdREQbFrpe6Ai79Op8ErZn5phDpLT8sS4ma6uXtfC
fDE9UOOmxOO2eMN2DNBPzXgg69e8X/RGiF2AnQq1ljDtg+cfhYo5eS+jT/D+2zHh
Z5pRPxQ4iQEVAwUQNOXar1xNwteQARzFAQHKbwgAg3vPNG4uLrVi98xcLEfubNIT
UAT3jBrlmDNIL2c6j6QJSAE8Tz6g3D5fUBxnceXvXjBJiIfP03zoACnga4FlahfI
PVQNEt+PgJd0ePI2t+SWZ2F6cgvyB8DFb3L7pzqePoPAOCUZtXtNh5Gp1OHCPA4f
jjvOIrxxREknXmAJJmuISJMgkf35iz8H5I2d5twDRq2eMN0BiOj1xAZDBlGUkUJj
j5TqyCoRfqTp3uwG2WX1BMKcECQ5vEz+WbMlfGXBoonKNyIGMlhkIrk9rGObXK1Y
PiqLPUwZceGTJOtN9BsvPXhPVYK5rP0mXkn53b6tmHVbXjG4zHE6WIaPA3+XJ4kA
lQIFEC+btW+pGcXxlh9KNQEBbukEAJDsbalhK2QEzlo4ra51v+b9gABn82VjZ6ZK
AQl9isDKxJjTJ9uEoa3Qrn18ZKi2ZJFww6mrS2BMybuVCOxSvl7PVFr/QO/t7fGg
ZBgnzvrhf3CmikL670m7Et4ZO2BZjA9RkgcVxBqdmyVSYIOvfqS96zH/bBcF/z5K
2CExsDe/iQEVAwUQM20T867PD8B1UMKBAQE4QQgAqFwurz7Ltp7weTLYnP0/jdee
aOcrFViMOEgaAkiDFfXytQVz7d3Y58QNJKlXvmVzpwxRE+2zhm7T6smOwIBxzMMu
VEvvAjUl3XGAF9THZxeDNnQVRbU33G00yNKbQtYGIxRyJYbXlbHbK4zYziC7JZIG
qBjDzRN7gJv84OYH/3p6KQPcAeqm1qnZXLBMmxOPHMRFhHlMWiqeJi1xU8TuIbew
d080cKprVA0FjKclYRRzmrgYd5rfoJ6DE9z7Ecg4RsfL/AV9Xn3FsjkGIuJfEBPz
VwnRUxR43Wc5ZWL6MlGfk1OIH3+zLyBgXMi9tFA8pzEGWEglqT4DOsCy3oIA7w==
=nNA5
-----END PGP PUBLIC KEY BLOCK-----
```

- der öffentliche PGP-Schlüssel des SSH-Autors, Tatu Ylonen

  pub  1024/961F4A35 1995/01/23 Tatu Ylonen ylo@ssh.fi, Tatu Ylonen ylo@cs.hut.fi

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.i

mQCNAi8jGEgAAAEEAL67DAiiwUrJ2cxpMru5Sh/Tnkg25bd3X+p3zx0iv2GJXKeo
TLP/agMrd3aEAJzzSOX2ojKSHjTnYUcFjMeU1qsr4ximDti1A4SarhQ0bpoax+EH
nP1f4imRb+88z6akXXysIv2N9t7lNTJq2ScEQy/2eW94VhOLrqkZxfGWH0o1AAUR
tBtUYXR1IFlsb25lbiA8eWxvQGNzLmh1dC5maT6JAJUDBRAw1FwqUnJ6D/U/JbUB
AcKaA/0R9Zt/ZnYmLaErYFubTFUMj04PvjMGoLXBV+GHAUWXlrF5ScWw9oDrZUlV
a9pJ0hVSMCoczmmOxJzz6u+MkH9Dplj5AcXMXmBsOTquIq12psg955cK95AAMjwO
kc+nfxBhHbG6P0I+7USsSx0guGhPzAmuepsXa3d2LoDnx3Po2YkAlQMFEDDYIXrw
bvnGyXCglQEBs1oD/ixOUVgk1kVYD9M+QSdOSZ4ibhoyGzyERDWIZkQH0nv+fEpO
brPPz7IY9XDWDqnXYqScOrpJUQHw4KOuFFpANWUJ/9ohjFsmNhlts5n92ui4CZEx
Oa0Mb4Asd6wEFQ2/xudOF1c/BZq8Rkb3kfeJu4ExJ43cc6W5+pkmiAJrVtLTiQCV
AwUQMM/OD/RQjGvpqXoZAQEcPwP/dgPTYkLe1YAMUJrTCoH4WeBivX0sRLIUiiDu
9v6XVc3BoE2SGe+3gTERRSBzVSNyMLOOF3DWIxwhU4NsMyJSKO1rKxcDYEn9bVGx
RDEQNr6YAYFL8hepuVS6HAGI5iZkSFa3pL5c5gWg4DFoisbCzP4RrSYcmagJFN/h
T416SWGJAJUDBRAw1eGBe8C+byiGw00BARneA/94LSdrgrTwMoqoSlKUxAZtJKB8
fIe54jwKheEON9xuOavv3K4uX3v0fYc96dEKkHI0bjeW9JhIPU0+zDU729vtPHU+
/jFQ1DrkD+nsKZCtYdBM9dz4sVlbtjM2wOc5rcG0pTw8FieNTSpl4+0BbcRRydyw
zL2s/rLdK6aaGsizDIkAlAMFEDDQHDS1NvCWDnhfxQEB3RID9RAz1To/XfYOtasU
/2NjLPF6+ldqD3l8vszBt5jWwCAMr1B9F/4NpdaI89TCNXhJLcKF3KI6+0B5H37B
a8KU8FXuJTnQRARq+OP0i01GAvvdKwLt8jK+YH1pAaaRnk49/bHZe+3Rs88I1QMq
W+QaNQfuXcUyH5Vb2Gwh3culMXCJAHUDBRAw0DRqkJCLOFRne7UBARFjAv96K1Bn
eD0EXreylUmlwaUaNwvZ/wYeuioEE6VQivAIqm2rOAxS4dYZKgI82QYbVgS9iDwZ
hyyRgs1QjpDs+Eyq4KWlRC39X4vbSdTKKBbardSAtNdZbLtQuPMw8SXnTDiJARUD
```

```
BRAwz+7ZHBwt1lghBYkBAT2AB/0UYpOENT8Koy9huUXJG03ydcC9UXCkMohhRgjs
pDQxciPOTBeXFurR3x6xWwaI+aZC38vAP3Ca6na1XPX+tVQ1XN18WcCGyBsxxEw+
RU3K1TG12Tikj4VL+lwxRCqaV2JVmGaVidWteenmjFrUCOkoV7ePZHjqrc7awRW9
oAMpuGtmE6905CYLkXb8JIiAdDUK5eHNTV/q7GaWhfxBzTfNHijQNJXSlzkO/CM6
PngGTs4OUvvw7SpWIuHIDBA94kWjS0pd59UCAW/Y7nhu543pL9WybM/xG8HyXqPS
Of2vOtvccyJYgJ7+zXOg13UbkX3DnguijB/01PshA9Oh0uKgiQEVAwUQMNRuc27j
mhkvZ2ThAQFIuwf+OwL3CKo3+ggqtCqWxRcHGMpIpEALSzSlEi8XC0Pcgd807iIg
thE1ElMwjpZCKM8BOeWC2wfEdCyxSCXa7k2O5laQrFNZLemfPr7kWirAiLUpNbJt
OfXzlGL68bUPbltPu9cQBzbk9UJlFLOEj5Se5JPtonqzb+IIuX/8Miby47l/FJuU
mb7UEx6cw6SR0hvMWi48qxgN0pSMxWA2bxopaaZJkuwbkgW9Hz6MiYQ9Srez/ENo
J2oppK3WSu1EMkI3cp1umQF6Rah8UQUMhHdbYZLmKyEANMeIYuCTxn20qgyiquLQ
ThL4mzZqmtXc0wncQR71p+HXhuItUOoPgdGSHIkAlQMFEDDP0VGtKmT3ql3AjQEB
5pcD/0LL7IaY8zXzM0MarHY1wxE+CZ0pvWpf2yLZP0IIkmBpGuSQZL885ePSDElv
cjEW6ERLBWTrOB657kdJArSGdYySgcfkQY++ifIipMsBJFpnaOTTLve5LJz3ms2D
UPrY8dx7Ujvkl1U1rJ0niYV/zb5n1N9EJrfZycQGLJgxbo+IiQCVAgUQMM+IZ9y5
iExUDHZRAQHMIQP+MXLm0OJ/BtuPOxfxxFls8Nyi28TN8956CuIeDZ0IdAl4ELFB
Z0xMHEg+Yds2cZ2kpodqfQJIz26+CIv2v4bAqxOU5/ptBzLOpZW3Mp2WDBmSk28G
jQQqSfCe6+muRFsM2vgBfmVnQ1+ks0Yce8wj6hyetLjS6cqofIfstq6WH5iJAJUD
BRAw0Acn7zPwJffxYVUBAaN5A/9FTJ2e+3F0PIGlkuZKT4uj4EnHWV7W7uFbDpaa
a/cMhcq7nF51lSSYvRQkJ0F6XP2DVz868l0sAWK4/uzqjy52CokM5HWc2DTfUVQf
AhjYxVpqtPAV1pw7sWiRN6jbsnP+HYfDQp5DgTZ+hmdt1zb+LOgc+4fTC61Xz5py
ryhnD4kAlQMFEDDN74eEEZZeTLqS0QEBx+ID/Rgm81WZkt6w7LPrm2BSjwzDo9mf
QBDcLLaUgw2ouRpKdc7Xxbax6OjxVF6zj1ironYI2g9aomSIF4ZfGSJ11AFLrJZY
2aroa1ULGmbS6qPMydHu4GkNdd7J8XriwAIiAsB9PRZQDhFxdotw31C3dZL7RTjt
w6Lwc1PL0PIedTNziQCVAwUQMGe+/2ZacKQwIslRAQHgCgQA4N80P+5NneVmAcbO
TK4HOl5VxZ6zpWyI4XcGECAV6tJ7uloH6UG1ZqQoADADGEBWme80vele1Hw7EYWH
XdMwrshFdavIJ1dxX7FqGDAL+X5HOro1lVWF4Hr6Xywtdlz8b71MnKzu6B3v+eW8
r7hroTmAgM0SWIMWERUAFPJhA1iJAJUCBRAwZ0M5SJ+gxtw6gZEBAWgnBACL/vqV
f2McoBkJEx5+XWZjcYUZMxkOFeBXPXPvGKzale7gmU6702EdAWQ1yVTZVHl59A60
qIaisCdiIXulkCvVUq3Y8IQTGZU66tMcAkXilH8UfM55WqjEHU38IiogNipFBewe
v+SKFn1GbMEjemGgev1EayudZSYQNpxBK3qhlIkAlQIFEDBnQtoGcW8JGaIDQQEB
DMAD/Ajrjjl/WvwZWhGKSSEiCdVO0HS8zGsQ68QGEWQFKB/84a7NDPEEG1cDVcUM
ziwH/kukGdD6QfRm/RC0HBWXaUPjZQ11fi3xhS9mf/rMh/2ojpZsodVNWT9Mjyri
hsziJYobBrBDD4358lNOUnbSJQqS3BVMRM2gAwwjHNGoBaAQiQCVAgUQMGaTg6kZ
xfGWH0o1AQG7UwP6A9/4N73/TD99UD0XgWxn4GZwIQauUVnvFSxyxgxZa5+7ES76
qIo7z30sZGfUOdteBI1HquNfjuiXKxvAtiiFj8Wg5Ej+Jop6AvuXatoMyv21A/rg
bdb9jJBYVeDjjua0pIohSzgj5dvi0ny7M8cuv/cP7MWK5x4ZF5RdENjVFwKJAJUD
BRAwGr9dxS1HbQ2/kG0BAW0YA/9FqZ2esQjRWzjPEuOwQeNJi8ipdds3wCAetQlg
f+mPt4XJ4teVWakELO93B22KzGvf7FiBl+XxlMN31iw5NT8yyGphZrozHh3QrHsd
l6FgIMJXkpBUOUj23aMgqCkGa4DJKQdKP55itVj2cYfO0CbTYbJDbhNVmw1E9ud5
Npm1AokAlQIFEC9GalD+SsgEEr2s0QEBCnUD/RPnKKz0S7nUnK9vzaof4o73uI6V
M9csrluYwY5qS8qixnAdBntZRs9xzl1JqZ+G3MMXqiSYRu0hNIaZ+kXq/6J0QOaJ
ePBN7Hd+FFWrN+ku8lFyY1gMBEgpsDfL86Qg2m+beu18GlRj79dNgDpBYwqfXO8s
YS2/H5uckm/0DZMGiQCVAgUQLzsVUN023rbQXPl5AQHjowP+KgljROUcfGJEC6FR
mwq9DqpSKkR7KT6Tib7pAT/Fd7218R+UEwjKrd7a0kIukWnCmGFVP35X0njPqgfH
NWMDw6+9DR8Q9TjuOrpWeaQOMo/Sb0o2DbznVC0SIHAMcoSTlljfgC1lkjHyKUxY
Xe5BkOy6puQb2dqBy2gYhuuTw6M=
=ixuR
-----END PGP PUBLIC KEY BLOCK-----
```

oder alternativ:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 5.0
Comment: PGP Key Server 0.9.3-db2test

mQCNAi8jGEgAAAEEAL67DAiiwUrJ2cxpMru5Sh/Tnkg25bd3X+p3zx0iv2GJXKeo
TLP/agMrd3aEAJzzSOX2ojKSHjTnYUcFjMeU1qsr4ximDti1A4SarhQ0bpoax+EH
nP1f4imRb+88z6akXXysIv2N9t7lNTJq2ScEQy/2eW94VhOLrqkZxfGWH0o1AAUR
tBhUYXR1IFlsb25lbiA8eWxvQHNzaC5maT6JAFUDBRAz99c/HtCXpnpKVQUBAb0P
AgCfeKPHK1ZrgwGwExFOweK1Car8hb+Kkg9Ca4Up6UHuvBBvnAFfvcNjOOs3dCCQ
MmUMePsAC3cM6Th5IjtzQPBqiQA/AwUQM/tSACHQTFe9I2YCEQItPgCgwlb1M0Bf
Q+Mc+W67dN8R2M6dEtkAn16Qon2x9JrYofKJHntO89GrtXnCiQB1AwUQM/RgWSfJ
```

eCVLKSvFAQHFTAL+Lhq9KFSM9BsdMyB630kd1LADs4RsFXzt/ldy87n5xQfDuWPC
nUHSGESf5MP2qyslolan1jTiOvwuY+VSBOa1BpSUHZ61OchWFM0rhkUMRUAv5oTe
M6IxpqUKn4yrfP99iQA/AwUQNAyrFE3LZlzJbHHXEQLcBQCgz5LILUzS4TDtlAN9
kdImWD5XkxoAoIt5r9eQeBmt73bdkCjahbrc9pXciQCVAwUQM6m/hVQXJENzYr45
AQGmWwQAhJVXa/uKsKXXFMTanCVVxmJHo7guKtq5s7Q7XMQlY6LyJlM1h2yUqmEH
UWkEEuy+ngUj7f7TTuDwEjhCzDrvJ1jTRiEwiqUTja+bs65e6Fi0DHmWF4Ix2hkP
K/L8JB7V7j1hclo+v83GIw7NuKpzXqNBci0g4Xln/ut9qbFhUeiJARUDBRA09JOz
buOaGS9nZOEBAVpbB/96N8DCCqTa/cAQ9byNFmo1hwyV8Q3bijioG2hgjRR3zorF
den+EbyvaP9TZj1urCuxBAobnTFkwhAnvJh+Xl7rT7p8OGsYqHpFJQ5kNPLSN8yk
xQ+1J2m9hUlVo4JSDN5Or8qi2Ch2+9iR8+uAmGaRr1HofG7lRkYdJnIhT2RqDov0
Z9agHg8F1DeAbbypmX+iXWk7tl7oLrTJaaoMgjeoGFjVlb+1+dtLT93WzcaPgRuD
AVkFeXqeEdCG75A+1Hf/YrakTXXaNnz0Ue5ywPvtVNw2hArdZG7V6MmSuMArSkhL
t74RlZZVPuGFIx2o7dgUAb9+ohciJbJX6jTlgdRWiQEVAwUQMq+7YncrsxJuc7vB
AQEoCAf/cf/RNXPXZ4Ih8X9tYV5h4wGaW7hQeoKgbqz5jEolccwnxLyy1h7323Z6
yyBzGVQQWfi6qBkWRZGG2pH2VvgX63y9UqUueiecL+QyTPowE/K8bn4z0kgan3eI
xtIGRyhM3p0dffLQNyKJgE00E2IRkL+BdPB94JFCtJ9Jl2zokmxxHYWHZu6Cdi1c
QGSU1EK13wil0kM0f9ZZT0FV5/z5Q5V9k6ZzsA2kFhLYXblwVhomhgSx8LwBSuQh
YeXFSc9VNolvkgZVVhEJ5GrVZAW8x4BZq32JRsqBklR8t66grTzoALGmuA0EjV1d
s8h4j6EkspvGlm6ZR1+DVa/qoLbh3IkAlQMFEDQAc46ohJICDMdH1QEBSmcEAKNn
7pvLDWeBJpHIaYEeKxvxLEprP7nuc/D0BVU32cbPMUqG9FZZh9Z2GCC797nzqez8
hru2M8NdhEyC2jesp1E7Lul/aQdynTp9BTIoXxEkKtTgFI5+YJaNDWNem8LFyp9L
889H36a1Sk5BLzpjlsnSyf4m3o9FXJzid6ZoIAlRiQCVAwUQMYh+iKkZxfGWH0o1
AQHFygP/XiV6XuPiyuX7Decr1Guoj2+gZ1xGxv1OFD+D9BX67IEMXBAh21hHN5SZ
KQMLCJ/F1jwJ0z3uK7aUIPyu+WxztQO7XU4d4dXV4Crq238bpifTSxVTxC1yGFBg
ECbbZ8QvxERtjED/+UA3pdJN5nSRX0UN3ZoPPDC2pfFJ3L1BVuiJAJUDBRAyr7ts
vSkGOHaHWQUBARUcA/4yGet0Lrwfdj7WPX54Vikf6628MMoOVGf/CUKPWtY0+KHx
XFbwZCAOF6aC6TkEo19EADIkDDdErCaptaHl6opvrnTm6JKhsQDeyiJbbgR06nNk
rmDK3aE5zcE937p+lD9ajeU2yQDcyh4FfXkkZEKl45dSwxZklf81m0fCkigrrokA
lQMFEDQWCaDZv6jI+kfrMQEBIGoEAIjdq8hpIZWXgZm+HA9AAzAE07HzOs+SAC5U
U1pBo8uynaMgnU/HcwfPiLKUu/wns+D2x9YDsg/gnPrgTiD4WULZ4aTVQj9RMRsP
a3X4WHpH4+Vd76LZTYTA2i8L5Phn1jigCQlRPCjT8HVlwwCwVexSbXN+h5hdadBM
YrD0OL6xiQA/AwUQM/tR1d0suEaAeEvdEQJ6JACcCgAYZFeUW820q+UzEshdnfqm
3kkAn1e3z/a3FV1ix57fYLbDXO92rtzOiQCVAwUQM1anNe9tgkHwgRldAQGdQQP/
Y/g1a1BvGpqMq8sHkdPNnTBR4PeFOSKlReP0y0g+rlNLKlxovvOhDm84wmbjP7fX
P6Qn2rhxt6isx/g6yu4KT4K06eVsgdClSwF8op2Mk16aj/9xzc0Okk+VdTs/zZaD
TjKLZy6nAMBSKXZxfVB6xGXZpuReZXwfZZbQ3Nq5CfmJAD8DBRA0BF6A/FZSIdBw
IB4RAn+eAJwOkNuIgjnqrCUtzRl/1epiVbqxwQCgsTiK/aW7SU4THWmyg1yRzmHl
WqG0G1RhdHUgWWxvbmVuIDx5bG9AY3MuaHV0LmZpPokAlQMFEDR038IFNglAj7a+
8QEB6bgEAIC/iMbDc769VxFjLZDWpsZ92foSmrfQFfr2jabIzUPKzPrgcOYS3p2O
BM8uaMR5KitAhbHEZ89JX5MrWpelVaweD7tpxzSIYReP2sj+7EVqzsbirQiBlGXI
h4ZUR4VxeJpCmHBBnLu5cKlrYh1G6epohgecyz+AygcBFdRJ2ihjiQCVAgUQMViY
PgV7LMKx0XT5AQFwjAP9E8RPu3yyWlBV59N6YD+CBvLkWnnsedbt+h/Rv3vTuWbS
awiTl3KpLK4+68X/Hy7YLBXEvVIK6hfbQwNfgsoyLv+0dRjL/BBGUypvWMx822pd
TQItH2uSqkgPKNKwsqQuqVPt41vjQX8G93IjgXcg2u1MoyzN/wO7Ppyit5CYAg2J
AJUCBRAwZ0LaBnFvCRmiA0EBAQzAA/wI6445f1r8GVoRikkhIgnVTtB0vMxrEOvE
BhFkBSgf/OGuzQzxBBtXA1XFDM4sB/5LpBnQ+kH0Zv0QtBwVl2lD42UNdX4t8YUv
Zn/6zIf9qI6WbKHVTVk/TI8q4obM4iWKGwawQw+N+fJTTlJ20iUKktwVTETNoAMM
IxzRqAWgEIkBFQMFEDDP7tkcHC3WWCEFiQEBPYAH/RRik4Q1PwqjL2G5RckbTfJ1
wL1RcKQyiGFGCOykNDFyI85MF5cW6tHfHrFbBoj5pkLfy8A/cJrqdrVc9f61VDVc
3XxZwIbIGzHETD5FTcrVMbXZOKSPhUv6XDFEKppXYlWYZpWJ1a156eaMWtQI6ShX
t49keOqtztrBFb2gAym4a2YTr3TkJguRdvwkiIB0NQrl4c1NX+rsZpaF/EHNN80e
KNA0ldKXOQ78Izo+eAZOzg5S+/DtKlYi4cgMED3iRaNLSl3n1QIBb9jueG7njekv
1bJsz/EbwfJeo9I5/a8629xzIliAnv7Nc6DXdRuRfcOeC6KMH/TU+yED06HS4qCJ
AD8DBRAz+0v5IdBMV70jZgIRAnY3AKDD2s2DwdKrlpa2wByLgopG30+hwACfSwc8
Td/ps0hNG9oCSZ6RbB60eDqJAJUDBRAx5n7oJBb4eZqNKQUBAZfMBACjfj6YU9tE
+QoymlcgdRNEhcNOzKqZCLdwL742xPM65nodXugPcNrgdwWydjaX2TqsMQmJwOMS
RTFnrCWPSB1JkTXkFARxi45/CcS5+cAg4yLJxryvhTYlfpqugH+FCB/spTL7qUEH
t8kkLfuShUHmCsJ5MN15GO9sDN/SpyKtA4kAlQIFEDBnQzlIn6DG3DqBkQEBaCcE
AIv++pV/YxygGQkTHn5dZmNxhRkzGQ4V4Fc9c+8YrNqV7uCZTrvTYR0BZDXJVNlU
eXn0DrSohqKwJ2Ihe6WQK9VSrdjwhBMZlTrq0xwCReKUfxR8znlaqMQdTfwiKiA2
KkUF7B6/5IoWfUZswSN6YaB6/URrK51lJhA2nEEreqGUiQCVAwUQMNRcKlJyeg/1
PyW1AQHCmgP9EfWbf2Z2Ji2hK2Bbm0xVDI9OD74zBqC1wVfhhwFFl5axeUnFsPaA
62VJVWvaSdIVUjAqHM5pjsSc8+rvjJB/Q6ZY+QHFzF5gbDk6riKtdqbIPeeXCveQ

```
ADI8DpHPp38QYR2xuj9CPu1ErEsdILhoT8wJrnqbF2t3di6A58dz6NmJAJUDBRAz
qb9kVBckQ3NivjkBAc09BACj6R0/Wsa4AmWd9wRtCtPhThDKiAxXKJQ7hZzZkDeX
c8rqpNb2mOV53xIqkIfgdSv6zyJB4OlUa0Zeo/Wsj58ePpaXAA4nmw5AtQCBUKBG
8AynBKQp6SrbQpullFrQme6fX0AO+7gSqMjOnVBCPgjS9X8oQJ08xCawz1ullMeJ
JYkBFQMFEDTl2u5cTcLXkAEcxQEBCJQH/RAk8Nt/D9yzSMZ+v2ALa0MhRgnvB4Mo
VXRjkahxc7/W3BUnCGEKRfCHjjE9xZl0KtAxWeJkV9/NdkNiMoVnxywRzxrakFfa
me6DOH0mwhf7nD+2XnGBLKddcbJOggLP5qTt1L+rOnLyWvZRO/U3tpcYMGB5DeuW
dhzF+bkcxjqCEJ8ZjP7n8wX+xr6MpWZnVZQqPqZ35EHOIgy7C7t+JhfW4phNwqJK
yHcWnLcX+IG1B4fgGW6uhppa4evFuNjePiIf/XVM1pVo0ADmSQxJjCJ60dEMQ88Y
iiAYUu7U7aq4cLiuDrJL49qP0iL5pxwSTAHVN2HS9oi3isc4k3Swp2yJAJUDBRAw
Z77/ZlpwpDAiyVEBAeAKBADg3zQ/7k2d5WYBxs5Mrgc6XlXFnrOlbIjhdwYQIBXq
0nu6WgfpQbVmpCgAMAMYQFaZ7zS96V7UfDsRhYdd0zCuyEV1q8gnV3FfsWoYMAv5
fkc6ujWVVYXgevpfLC12XPxvvUycrO7oHe/55byvuGuhOYCAzRJYgxYRFQAU8mED
WIkAdQMFEDFpXyhqmwnIWCbPjQEBFH0C/2tVgC3pRPe1EiI/EAJWJqDAErJLcwx2
mUOtOoor3J5QmrI5SG4h/k8QO8OMhX0iqebjwLELUCGkUaZg+zwXz5HHUYePKICP
EMJGMrwNqCRPvAg8OCwHopcocYr51faMGIkBFQMFEDDUbnNu45oZL2dk4QEBSLsH
/jsC9wiqN/oIKrQqlsUXBxjKSKRAC0s0pRIvFwtD3IHfNO4iILYRNRJTMI6WQijP
ATnlgtsHxHQssUgl2u5NjuZWkKxTWS3pnz6+5FoqwIi1KTWybTn185Ri+vG1D25b
T7vXEAc25PVCZRSzhI+UnuST7aJ6s2/iCLl//DIm8uO5fxSblJm+1BMenMOkkdIb
zFouPKsYDdKUjMVgNm8aKWmmSZLsG5IFvR8+jImEPUq3s/xDaCdqKaSt1krtRDJC
N3KdbpkBekWofFEFDIR3W2GS5ishADTHiGLgk8Z9tKoMoqri0E4S+Js2aprV3NMJ
3EEe9afh14biLVDqD4HRkhyJARUDBRAyr7tOdyuzEm5zu8EBARU/CACNjyWpOvFK
Nbji2BEqNUFU9D4+5gxQWQfx+eUcA3UkH0F3Frw2y/atcv4wjUisbnnunyo7ptWN
PjPVCtLJ1UF6d7TN35vv2J9kdDRDSCDdg/Hlc92ced2sbMLfEQWqpR240IB0VzRN
rZCY/6DEcAqlWr5IGIIVYwpi8Km02XvYPSY5nDDrE1/TvrmlN6nG02nhEF+8zdbk
hrqAbo4/GQ+0xm5HvN9t3mOezRp3IFufHXOXKzhu1QO7QTVUPz95kT2INxNocDqZ
l650aLjxlJDEzIsYjLIft7ZyZwy3tpfyyObDSjhdTsLG3EyHFE3AQOt1IlZ5WWrz
hbMzIFrBgt6EiQCVAgUQMUjqi3fSqyWwyHE9AQEYdwP+NINISbQNyjLojxHi6t+t
HIvQxTAuAyH1RhPzFM3xQGvZuuNV2B9mjC1Kg49UsI2/V86tkQS7oZlW0f/VkWoa
VayO1oDhWdJgB75/TKI8FfKsMawOAqOgv/U03iTZCarc7hK586KoUYS6oaZ/p7Es
pq7M+E05/6G4JdPqkXFF2cCJAJUDBRAw1eGBe8C+byiGw00BARneA/94LSdrgrTw
MoqoSlKUxAZtJKB8fIe54jwKheEON9xuOavv3K4uX3v0fYc96dEKkHI0bjeW9JhI
PU0+zDU729vtPHU+/jFQ1DrkD+nsKZCtYdBM9dz4sVlbtjM2wOc5rcG0pTw8FieN
TSpl4+0BbcRRydywzL2s/rLdK6aaGsizDIkAPwMFEDPTnKyD3wujOcdIpBECnYwA
njlRhIy5Ihjfagz75oCluomRI+IJAJ93AFBA1HG0rqMpqhUkJNa5V6+ayokAlQMF
EDDN74eEEZZeTLqS0QEBx+ID/Rgm81WZkt6w7LPrm2BSjwzDo9mfQBDcLLaUgw2o
uRpKdc7Xxbax6OjxVF6zj1ironYI2g9aomSIF4ZfGSJ11AFLrJZY2aroa1ULGmbS
6qPMydHu4GkNdd7J8XriwAIiAsB9PRZQDhFxdotw31C3dZL7RTjtw6Lwc1PL0PIe
dTNziQB1AwUQMP+YK42OFcG2AMe5AQHbsAL/dHggy0Gxa+O0lb0Iz8PhKQIK9BXs
r+BeSGApYSz2S4B8ur0jUCSWY2MreUxr5yxGFkyeE+x1MyVgm3EKqAO7OmF820YT
2vfBY0yp85LHSJgAwcFqigQ65no8JOThHyTDiQA/AwUQM/KkFI+pUU+WDilyEQL2
uwCg0DhcBCnFEu1ZMYWoFw/wy1eWIz8AoIZWTcoYLJJw5KpD8LC0mpKenW7hiQB1
AwUQMNA0apCQizhUZ3u1AQERYwL/eitQZ3g9BF63spVJpcGlGjcL2f8GHroqBBOl
UIrwCKptqzgMUuHWGSoCPNkGG1YEvYg8GYcskYLNUI6Q7PhMquClpUQt/V+L20nU
yigW2q3UgLTXWWy7ULjzMPEl50w4iQA/AwUQM/MneJWQJPlv1wT4EQI9rgCgo32A
oHUdHhYwr+CL7jLxGkCLmxAAoMH/eb1QHxfzoqO/oigFIAyMSej8iQCVAwUQM2q9
NqHBOF9KrwDlAQFCJQP/f9B8o9Yuo7xgw0bMOChXaWmmmAEhVwwPGAD5aVRUlrtk
LlLO8g3vaDNUpyc2gkjsIbyhUmfW0cEfyVqEQme8nn6gZ57JjhhpEi2GV4E5E604
HAUGXmsg3n5j1bhcOrLq6Zdxg1thLVCiqcCnR6qZHq7uU5C35Sx2IHULszNDXlOJ
ARUDBRAxPnU1ofwRupVewsEBAe0YB/4r7EILhMRxMf203gGW2Yw4pfkHygRSjI2O
5eEbFEjtw54CD8Cmk5bX6cjInMIND958KZmB/nh9qlVo9Qi+XxK7f/oJksMlGysu
sgpeCb9PUqZnfw6UuI3hIbL/hmYffa8vT2122d0aSYLeRQCO1BYEurOBhsNKaReu
quG/rEedCaZV33ttOVqLjoRCdXljV6tC+6NFGtAXlt07l+hiP2HvxgGDU+x5jLlq
OWt3WkO43bbYct78x/07qOqOzBfbUd2c8/ryVlpjteqPM+KN5Poejc0O4v6eteqL
xMRfGTTaNaqArTk5Z2IIPQbzUzseqdXlfDccoW56KDsO7xwCbpYLiQCVAgUQMGaT
g6kZxfGWH0o1AQG7UwP6A9/4N73/TD99UD0XgWxn4GZwIQauUVnvFSxyxgxZa5+7
ES76qIo7z30sZGfUOdteBI1HquNfjuiXKxvAtiiFj8Wg5Ej+Jop6AvuXatoMyv21
A/rgbdb9jJBYVeDjjua0pIohSzgj5dvi0ny7M8cuv/cP7MWK5x4ZF5RdENjVFwKJ
AJUDBRAwz9FRrSpk96pdwI0BAeaXA/9Cy+yGmPM18zNDGqx2NcMRPgmdKb1qX9si
2T9CCJJgaRrkkGS/POXj0gxJb3IxFuhESwVk6zgeue5HSQK0hnWMkoHH5EGPvony
IqTLASRaZ2jk0y73uSyc95rNg1D62PHce1I75JdVNaydJ4mFf82+Z9TfRCa32cnE
BiyYMW6PiIkAlQIFEDDP22CxaklAbeqf1QEB0PED/1k0GGvHCz6Rys+Lf07bru2F
4FNH4kawUUhdqNUeTGxxbGkTGVUFT9vQr7rXMsss/7y+ss0IRn4ehjn7tWC/Kdk3
```

```
wE+eBvHkN+9/EmeWEdGSrd2+8fkobGO91A9FXpfhWb9TBZ7U14H4uwJbNCAyPikG
hIRqgrcm8LYnANqGTddmiQCUAwUQMNAcNLU28JYOeF/FAQHdEgP1EDPVOj9d9g61
qxT/Y2Ms8Xr6V2oPeXy+zMG3mNbAIAyvUH0X/g2l1ojz1MI1eEktwoXcojr7QHkf
fsFrwpTwVe4lOdBEBGr44/SLTUYC+90rAu3yMr5gfWkBppGeTj39sdl77dGzzwjV
Aypb5Bo1B+5dxTIflVvYbCHdy6UxcIkAlQMFEDKvu1e9KQY4dodZBQEB7+8D/2IZ
74HgkXCiBMTcGGXpcr1nr4P0ILQegwXXru7Y2ClM/GARcIweP+vkgofJCXWvIEhK
LUAqKDKydBV/vL2siA6d24AJfUVkQE1cdGrDAQp816LXrAhmxXMW/QRyng/wjg/D
V6QOjIs1xIa4Fph/6kpMdMwAVIfca2MUvNTXQ5ayiQCVAwUQMVfu5b4Z393fEYrx
AQFxaAP/ZkO0SB9JDb1gmIX2AMh1LZvshVIe2Ld64gG2kDIlyRji3Oud0GtfAmBv
7wk8Ni5SVrNemdWdcaSYY7+BhAN8B2ThGjrOgd1XIReFc4mAAr8Lnv5pyL9gUbtj
9ck6L2llacfu/7uXj3T5/4j0lBUvr4IfV1hPrQVJtkUV6DZ+bWWJAJUDBRAwGr9d
xS1HbQ2/kG0BAW0YA/9FqZ2esQjRWzjPEuOwQeNJi8ipdds3wCAetQlgf+mPt4XJ
4teVWakELO93B22KzGvf7FiBl+XxlMN31iw5NT8yyGphZrozHh3QrHsdl6FgIMJX
kpBUOUj23aMgqCkGa4DJKQdKP55itVj2cYfO0CbTYbJDbhNVmw1E9ud5Npm1AokA
lQMFEDFfWaXZv6jI+kfrMQEBIF0D/3tcAZhfV7rUv/9P8GsmgXlgHdEU8mfzdpzb
51RZ3sVnGiDLsOArxLr6cSn4ZMSKqGCElo6eQyRTK0MjwuVhLw20+DVtCEpNMHux
dXsc65RnV/Ijq7jozrHjlJV+cDmhacSW+rh41j7Nt9OBQyW7e/6G12bto4apKO4z
wUswUGK9iQCVAgUQMM+IZ9y5iExUDHZRAQHMIQP+MXLm0OJ/BtuPOxfxxFls8Nyi
28TN8956CuIeDZ0IdAl4ELFBZ0xMHEg+Yds2cZ2kpodqfQJIz26+CIv2v4bAqxOU
5/ptBzLOpZW3Mp2WDBmSk28GjQQqSfCe6+muRFsM2vgBfmVnQ1+ks0Yce8wj6hye
tLjS6cqofIfstq6WH5iJAD8DBRAz+0vQ3Sy4RoB4S90RAgT3AKDTuDIw86P3eKS+
+3ZEXr/VoXTGmQCgxMVmBSUDkSjRg18Ytz7MLu0QWhGJAJUCBRAvOxVQ3TbettBc
+XkBAeOjA/4qCWNE5Rx8YkQLoVGbCr0OqlIqRHspPpOJvukBP8V3vbXxH5QTCMqt
3trSQi6RacKYYVU/flfSeM+qB8c1YwPDr70NHxD1OO46ulZ5pA4yj9JvSjYNvOdU
LRIgcAxyhJOWWN+ALWWSMfIpTFhd7kGQ7Lqm5BvZ2oHLaBiG65PDo4kAlQMFEDDQ
ByfvM/Al9/FhVQEBo3kD/0VMnZ77cXQ8gaWS5kpPi6PgScdZXtbu4VsOlppr9wyF
yrucXnWVJJi9FCQnQXpc/YNXPzryXSwBYrj+7OqPLnYKiQzkdZzYNN9RVB8CGNjF
Wmq08BXWnDuxaJE3qNuyc/4dh8NCnkOBNn6GZ23XNv4s6Bz7h9MLrVfPmnKvKGcP
iQCVAwUQMiHH5e9tgkHwgRldAQEFPQP9Gj6SR1S5SFN2fbtAGpXQ9eFw59Ar1XUf
ga9/6MpDmLuo1nNFk9izqCsLrjzAHyFNkrHx3lbWMQUM78J3rUZX2wZXoYPsUNpl
ji+zkk3KlcX/4Nh/6DGTiv5CRaC845pfz1ftptlbU3XaN4/81r3tFqteqbPhJyhe
9Rsg4XqjybSJAJUDBRAw2CF68G75xslwoJUBAbNaA/4sTlFYJNZFWA/TPkEnTkme
Im4aMhs8hEQ1iGZEB9J7/nxKTm6zz8+yGPVw1g6p12KknDq6SVEB8OCjrhRaQDVl
Cf/aIYxbJjYZbbOZ/drouAmRMTmtDG+ALHesBBUNv8bnThdXPwWavEZG95H3ibuB
MSeN3HOlufqZJogCa1bS04kBFQMFEDO3koz0Qjh5QeuduQEBNIIH/1xBJ3st4V1A
s5dwC9ezGptz79mFY5Ai417azu62tlGwKzqaiiw/OIalyN4OY1KYPEFvhiGzRVhI
w+t+IyT8z2zqxW7QdkYYv1gy4QXovB5zCPg97+K7tb1RvmQoz+v+3gxV5Z2Q5pti
j9JYy5Pn1PHmKqGGvvx53lTr9eHQKe8s7pTEs3+9/S/mxh753hy8De5kP10LqB3i
US5y0fEFCCR6gJ7HAkUYlFJe2z8+McePqsnFm+3WNiVBjMLZAdRo8hMROwLBQl9b
LUCnTWVid7yw9lFnBADmKSnQmFdsIwbx4JzbMKw08QNXGB77jpGhKT97WptMkyPc
S7+jPBMIkK+JAJUDBRAwz84P9FCMa+mpehkBARw/A/92A9NiQt7VgAxQmtMKgfhZ
4GK9fSxEshSKIO72/pdVzcGgTZIZ77eBMRFFIHNVI3Iws44XcNYjHCFTg2wzIlIo
7WsrFwNgSf1tUbFEMRA2vpgBgUvyF6m5VLocAYjmJmRIVrekvlzmBaDgMWiKxsLM
/hGtJhyZqAkU3+FPjXpJYYkAlQIFEC9GalD+SsgEEr2s0QEBCnUD/RPnKKz0S7nU
nK9vzaof4o73uI6VM9csrluYwY5qS8qixnAdBntZRs9xzl1JqZ+G3MMXqiSYRu0h
NIaZ+kXq/6J0QOaJePBN7Hd+FFWrN+ku8lFyY1gMBEgpsDfL86Qg2m+beu18GlRj
79dNgDpBYwqfXO8sYS2/H5uckm/0DZMG
=Ottj
-----END PGP PUBLIC KEY BLOCK-----
```

- der öffentliche PGP-Schlüssel der SSH2-Distribution

  pub  2048/AFCA7459 1998/07/11 Ssh 2 Distribution Key ssh2@ssh.fi

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 2.6.3i

mQENAjWneSMAAAEIALcvz7kKUx1ev4pTXGplG1pWuAxSgCMj4/IM43hHHw1awci/
6tumJW7SkMW3IFuBVRBBR1818ff+2CF346lLbn2QutJTkeY0CRDVGsjKh0EeRjrr
TtatTaPlzk6Qdo03Rye5ISi2OS/XFTxVbwiX8lr0tIqofHBbod9g20HWLXE/Gqfa
tG1+OCD7aw2Z0chaqQ21vdk94Xf4Ud3V1CFmm4yZ2Y865nfrq+MPVrkvOXIoxG4W
6PKrrFBOvmoPqdNyO/JcLYQ8A66r1uRCw9vfLuDPFhHlZH8TxbMFpH9QDVOrhcA0
5daJvwdKRNcAzUnAClR/kx/L+ji3NC1GZK/KdFkABRG0JFNzaCAyIERpc3RyaWJ1
```

```
dGlvbiBLZXkgPHNzaDJAc3NoLmZpPokA1QIFEDWnfKM/Kxi2RxUWEQEB8HsF/inK
hfiY+ygcoVmw0g7ahgkaXi1ieEtfS/mHLiqCje2nMFRGZHU4er0FlpP36rnFcayg
8ARl8nfbHhhGkni2hs89rvrHuNGrLgvjAusavY3xZ6W2ITgrGrrUDdnMtnLStx4C
FnqJYlhSL3uJfvPe/Yy01X8BlgNCHMmC+IWzloB/bsbX7mHjX+Ckomw8/Y/OL/hW
aMhm1OAFaJSH5XyYdyvIGD/Kz5jkh/CqeKmJCfoiXOObIAybGdW2RYmEC+dytokA
lQMFEDWne+JOuHYv3LmuAQEBY38EAICoxKUghTrdNYj88wcbtIdIVkW+/wYOC5g0
MuB7fV12zkI/S79iJ/fEvt33xaB33YuZCT2VvXf+tJayyN3byfpUQB3XLNCTKRuk
TB7SRQ9di0n19dPd/exEDLcJGVgF8jW/HPnZ2dwtnAiAA+O6iJ6/xMZgE4xCKKO1
/AWQRX6ciQCVAwUQNad7qQZxbwkZogNBAQFdqAP8C94QGzwDmlnsVs9BtXGJivSe
zTZXUsQ8zVKVuywKWIDPr+h/JY3m7jye78A+7/Cr+hT3TB/fnAKvMZdY5lG0/oDI
TllhnGfLqQvD1An5YfihBnf3VnW4eh6IiC/jmg1MC2Xvk0Gf/CLCv7H1cTE/pVh6
SMHkro3zuo4KDgYJiNKJARUDBRA1p3uOrs8PwHVQwoEBAb33CADGQjN9snqQ1LKY
MW3SGxUr7WKoXOW9si2q6c6NkfYc8I1D/LXRqKGJQ2RoVH4Mq3R9Gitx1Qs8EI9m
8knUGc9UsJdJ+TqZKHjP+h3cLLdgvrdybttPwOBFUSvH8JYufpnJuNHFp4K5J9SO
8SeE2TjicnW9G08REKrHXshgxy8dIgZhbS6sq19A3edXAIY0JpRZP0jmFR4P5jNo
ZSgu4KoyzYp+mgLKMWBRGTiPN3i5Twx4xvCQAH0VlOUq34GpRyF+Vkq+NvK/UMa8
cFS1nO91RinezIm/t11vBNl07Vv5RAq6MgHXgzJNwbne+CDVxioEdpgDbkNiv85S
rUHJwy1ViQDVAgUQNad7chM/FXvA9CSpAQGnRQX/SztmPU5woFmsL19Wh5Hk6ERb
untgXoAESSlAemL0VxZrmu1g2XRBWaMv0IeNzpqjYbD+PsmSSkX1Zw8K7uH3t2hM
U1qtLL56K6ytmRJVRoHAaKscBwPGacMYXNB6lFYbld+Jp3nc1eQZiT3RUuw5z3W7
UPIAwvWdoChLaXV56xZA8ZdBFMtyBDgkC6U7OcKDi/ttAjtUOatTE5LhWIBQyzB8
6xx4SVg9xHQ+6XDXue9FhojdU+66L4wg/g/zcbJgiQEVAgUQNad5IzQtRmSvynRZ
AQGwFwf9G8gwwpPvX5xjiW9W7AgEum+/WTQjs3Ej+Q3Cnu78srZPVm0Y9bkq70ok
ca1g9N+WDoVy+PZ3gkZhJI1pHsCQA0fcnIJaR6aUHJb2rJ36XkgGmpDj5AlRkHpW
nGZY96hRrOYK4z7FPtyndxVK7sp0Et7wusYEoc5ZbQVbyt8cK6etO+dEnC5TqkyY
yzpLXMybq1x7Tt4B69mIIG4prxkkvWBYvKEjY7Ikea3fRqBcmUuSrKnmiEa1ElT9
ZtvqW7WHmnSKkTHblarD/L75NQxvrakVCVpmFvbcjjy3Gmsu/+ON3w8ev6HIbj2i
/qbb9IIXwZm2uQ3h5iAQc4Y8t+1r2A==
=oFyR
-----END PGP PUBLIC KEY BLOCK-----
```

oder alternativ:

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: 5.0
Comment: PGP Key Server 0.9.3-db2test

mQENAjWneSMAAAEIALcvz7kKUx1ev4pTXGplG1pWuAxSgCMj4/IM43hHHw1awci/
6tumJW7SkMW3IFuBVRBBR1818ff+2CF346lLbn2QutJTkeY0CRDVGsjKh0EeRjrr
TtatTaPlzk6Qdo03Rye5ISi2OS/XFTxVbwiX8lr0tIqofHBbod9g20HWLXE/Gqfa
tG1+OCD7aw2Z0chaqQ21vdk94Xf4Ud3V1CFmm4yZ2Y865nfrq+MPVrkvOXIoxG4W
6PKrrFBOvmoPqdNyO/JcLYQ8A66r1uRCw9vfLuDPFhHlZH8TxbMFpH9QDVOrhcA0
5daJvwdKRNcAzUnAClR/kx/L+ji3NC1GZK/KdFkABRG0JFNzaCAyIERpc3RyaWJ1
dGlvbiBLZXkgPHNzaDJAc3NoLmZpPokAlQMFEDWne6kGcW8JGaIDQQEBXagD/Ave
EBs8A5pZ7FbPQbVxiYr0ns02V1LEPM1SlbssCliAz6/ofyWN5u48nu/APu/wq/oU
90wf35wCrzGXWOZRtP6AyE5ZYZxny6kLw9QJ+WH4oQZ391Z1uHoeiIgv45oNTAtl
75NBn/wiwr+x9XExP6VYekjB5K6N87qOCg4GCYjSiQDVAgUQNad7chM/FXvA9CSp
AQGnRQX/SztmPU5woFmsL19Wh5Hk6ERbuntgXoAESSlAemL0VxZrmu1g2XRBWaMv
0IeNzpqjYbD+PsmSSkX1Zw8K7uH3t2hMU1qtLL56K6ytmRJVRoHAaKscBwPGacMY
XNB6lFYbld+Jp3nc1eQZiT3RUuw5z3W7UPIAwvWdoChLaXV56xZA8ZdBFMtyBDgk
C6U7OcKDi/ttAjtUOatTE5LhWIBQyzB86xx4SVg9xHQ+6XDXue9FhojdU+66L4wg
/g/zcbJgiQEVAgUQNad5IzQtRmSvynRZAQGwFwf9G8gwwpPvX5xjiW9W7AgEum+/
WTQjs3Ej+Q3Cnu78srZPVm0Y9bkq70okca1g9N+WDoVy+PZ3gkZhJI1pHsCQA0fc
nIJaR6aUHJb2rJ36XkgGmpDj5AlRkHpWnGZY96hRrOYK4z7FPtyndxVK7sp0Et7w
usYEoc5ZbQVbyt8cK6etO+dEnC5TqkyYyzpLXMybq1x7Tt4B69mIIG4prxkkvWBY
vKEjY7Ikea3fRqBcmUuSrKnmiEa1ElT9ZtvqW7WHmnSKkTHblarD/L75NQxvrakV
CVpmFvbcjjy3Gmsu/+ON3w8ev6HIbj2i/qbb9IIXwZm2uQ3h5iAQc4Y8t+1r2IkA
1QIFEDWnfKM/Kxi2RxUWEQEB8HsF/inKhfiY+ygcoVmw0g7ahgkaXi1ieEtfS/mH
LiqCje2nMFRGZHU4er0FlpP36rnFcayg8ARl8nfbHhhGkni2hs89rvrHuNGrLgvj
AusavY3xZ6W2ITgrGrrUDdnMtnLStx4CFnqJYlhSL3uJfvPe/Yy01X8BlgNCHMmC
+IWzloB/bsbX7mHjX+Ckomw8/Y/OL/hWaMhm1OAFaJSH5XyYdyvIGD/Kz5jkh/Cq
eKmJCfoiXOObIAybGdW2RYmEC+dytokAlQMFEDWne+JOuHYv3LmuAQEBY38EAICo
xKUghTrdNYj88wcbtIdIVkW+/wYOC5g0MuB7fV12zkI/S79iJ/fEvt33xaB33YuZ
```

```
CT2VvXf+tJayyN3byfpUQB3XLNCTKRukTB7SRQ9di0n19dPd/exEDLcJGVgF8jW/
HPnZ2dwtnAiAA+O6iJ6/xMZgE4xCKKO1/AWQRX6ciQEVAwUQNad7jq7PD8B1UMKB
AQG99wgAxkIzfbJ6kNSymDFt0hsVK+1iqFzlvbItqunOjZH2HPCNQ/y10aihiUNk
aFR+DKt0fRorcdULPBCPZvJJ1BnPVLCXSfk6mSh4z/od3Cy3YL63cm7bT8DgRVEr
x/CWLn6ZybjRxaeCuSfUjvEnhNk44nJ1vRtPERCqx17IYMcvHSIGYW0urKtfQN3n
VwCGNCaUWT9I5hUeD+YzaGUoLuCqMs2KfpoCyjFgURk4jzd4uU8MeMbwkAB9FZTl
Kt+BqUchflZKvjbyv1DGvHBUtZzvdUYp3syJv7ddbwTZdO1b+UQKujIB14MyTcG5
3vgg1cYqBHaYA25DYr/OUq1BycMtVQ==
=w1CM
-----END PGP PUBLIC KEY BLOCK-----
```