

Inhaltsverzeichnis

- [1 Ubuntu](#)
 - [1.1 Authentifizierung mit Hilfe von Schlüsseln](#)
- [2 Datei über SSH \(SCP\) kopieren](#)
- [3 JAVA](#)
 - [3.1 Einen Tunnel Herstellen mit JSch](#)
- [4 Putty und das Public-Key verfahren](#)

Ubuntu

Authentifizierung mit Hilfe von Schlüsseln

```
ssh-keygen -t  
rsassh-copy-id -i ~/.ssh/id_rsa.pub user@server
```

Datei über SSH (SCP) kopieren

siehe: [Ubuntu](#)

JAVA

Einen Tunnel Herstellen mit JSch

Hier gehts zu [JSch](#)

```

...JSch jsch = new JSch();
session = jsch.getSession(<ssh_user>, <ssh_host>, <ssh_port>);
UserInfo ui = new MyUserInfo();
session.setUserInfo(ui);
session.connect();
int assinged_port;
assinged_port = session.setPortForwardingL(<localport>, <tunnelziel>, <port>
);
System.out.println("localhost:" +assinged_port+
" -> <tunnelziel>:<tunnelport>");
...
public static class MyUserInfo implements UserInfo, UIKeyboardInteractive {
public String getPassword() {
return passwd;
}
public boolean promptYesNo(String str) {
/*
* Object[] options={ "yes", "no" }; int
* foo=JOptionPane.showOptionDialog(null, str, "Warning",
* JOptionPane.DEFAULT_OPTION, JOptionPane.WARNING_MESSAGE, null,
* options, options[0]);
*
* return foo==0; */
return true;
}
String passwd;
JTextField passwordField = (JTextField) new JPasswordField(20);
public String getPassphrase() {
return null;
}
public boolean promptPassphrase(String message) {
return true;
}
public boolean promptPassword(String message) {
Object[] ob = { passwordField };
int result = JOptionPane.showConfirmDialog(null
, ob, message, JOptionPane.OK_CANCEL_OPTION);
if (result == JOptionPane.OK_OPTION) {
passwd=passwordField.getText();
return true;
} else {
// wenn Abbrechen beim PW-Feld gedrückt beenden wir
disconnectAndExit();
return false;
}
}
public void showMessage(String message) {
JOptionPane.showMessageDialog(null, message);
}

```

```

}
final GridBagConstraints gbc = new GridBagConstraints(0, 0, 1, 1, 1, 1
,GridBagConstraints.NORTHWEST, GridBagConstraints.NONE
,new Insets(0, 0, 0, 0), 0, 0);
private Container panel;
public String[] promptKeyboardInteractive(String
destination,String name, String instruction, String[] prompt, boolean[]
echo) {
    pane=new JPanel();
    pane.setLayout(new GridBagLayout());
    gbweightx = 1.0;
    gbgridwidth = GridBagConstraints.REMAINDER;
    gbgridx = 0;
    pane.add(new JLabel(instruction), gbc);
    gbgridy++;
    gbgridwidth = GridBagConstraints.RELATIVE;
    JTextField[] texts = new JTextField[prompt.length];
    for (int i = 0; i < prompt.length; i++) {
        gbfill = GridBagConstraints.NONE;
        gbgridx = 0;
        gbweightx = 1;
        pane.add(new JLabel(prompt[i]), gbc);
        gbgridx = 1;
        gbfill = GridBagConstraints.HORIZONTAL;
        gbweighty = 1;
        if (echo[i]) {
            texts[i]= new JTextField(20);
        } else {
            texts[i]= new JPasswordField(20);
        }
        pane.add(texts[i], gbc);
        gbgridy++;
    }
    if (JOptionPane.showConfirmDialog(null, panel, destination + ": "
+ name, JOptionPane.OK_CANCEL_OPTION
,JOptionPane.QUESTION_MESSAGE) == JOptionPane.OK_OPTION) {
        String[] response = new String[prompt.length];
        for (int i = 0; i < prompt.length; i++) {
            response[i]= texts[i].getText();
        }
        return response;
    } else {
        return null; // cancel
    }
}
}
}

```

Putty und das Public-Key verfahren

Mit Hilfe des Public-Key verfahrens aus Putty auf einem Server anmelden:

#Schlüsselpaar erzeugen

ssh-keygen -t

dsacat .ssh/id_dsa.pub >> .ssh/authorized_keys

- Transferieren des Schlüssels (id_dsa) zum Putty-System mit SFTP o.ä.
- Download PuttyGen <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>
- Starten von PuttyGen Load Klicken und unten "all files" wählen id_dsa öffnen
- Save private key
- OK
- Start Putty
- Navigiere zu SSH->Auth
- Die erzeugte ppk Datei einfügen
- Speichen auf der Hauptseite
- Verbindung Öffnen