Sicheres privates Netz in unsicheren Zeiten

- Richard Albrecht, Jahrgang 1949 http://rleofield.de/
  - Physiker, Uni Halle-Wittenberg
  - Bildverarbeitung, C/C++ Entwicklung
  - Middleware, Datenbanken, .NET, Webanwendungen
  - Senior in Görlitz mit Linux
  - Betreuung von Linux-Rechnern

- PC ist zur Privatsphäre geworden
  - private Sicherheit der Daten wird immer wichtiger
  - Bundesverfassungsgericht in DE, 27. Februar 2008
    - „Grundrecht auf Gewährleistung der Vertraulichkeit und Integrität informationstechnischer Systeme“

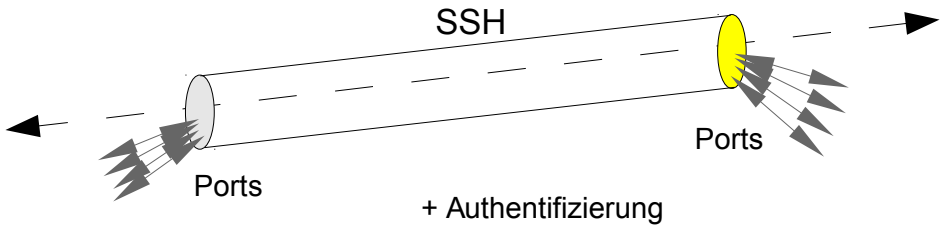
- Deshalb der Vortrag

  Es ist ein Designfehler, dass das Internet nicht „grundverschlüsselt" ist.

  - den Teil, den wir unter Kontrolle haben, verschlüsseln
  - Kryptographie dient der Durchsetzung eines Rechtes

Sicheres Netz für die Familie

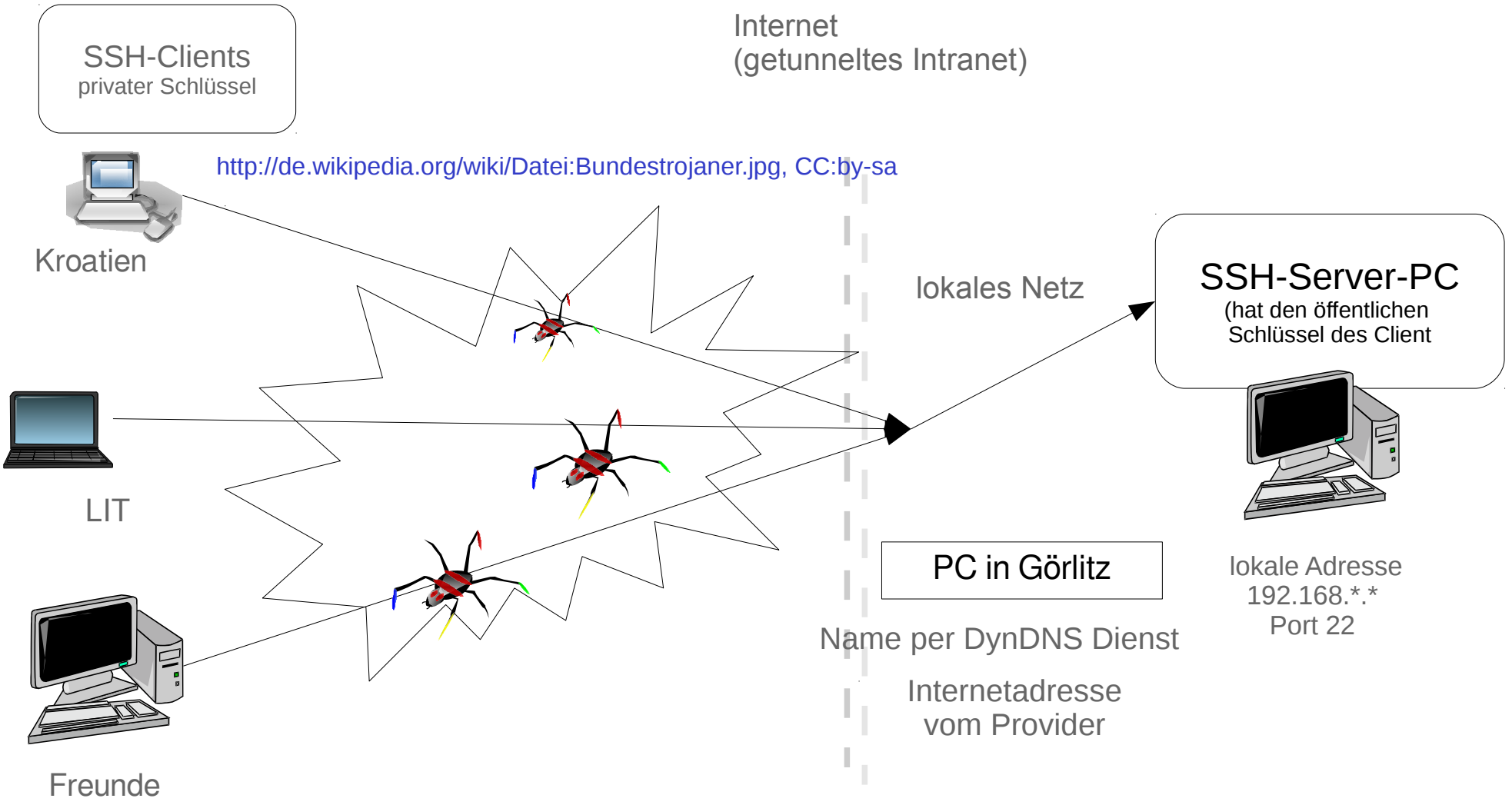
- SSH
  - universelle sichere Verbindung (verschlüsselt)



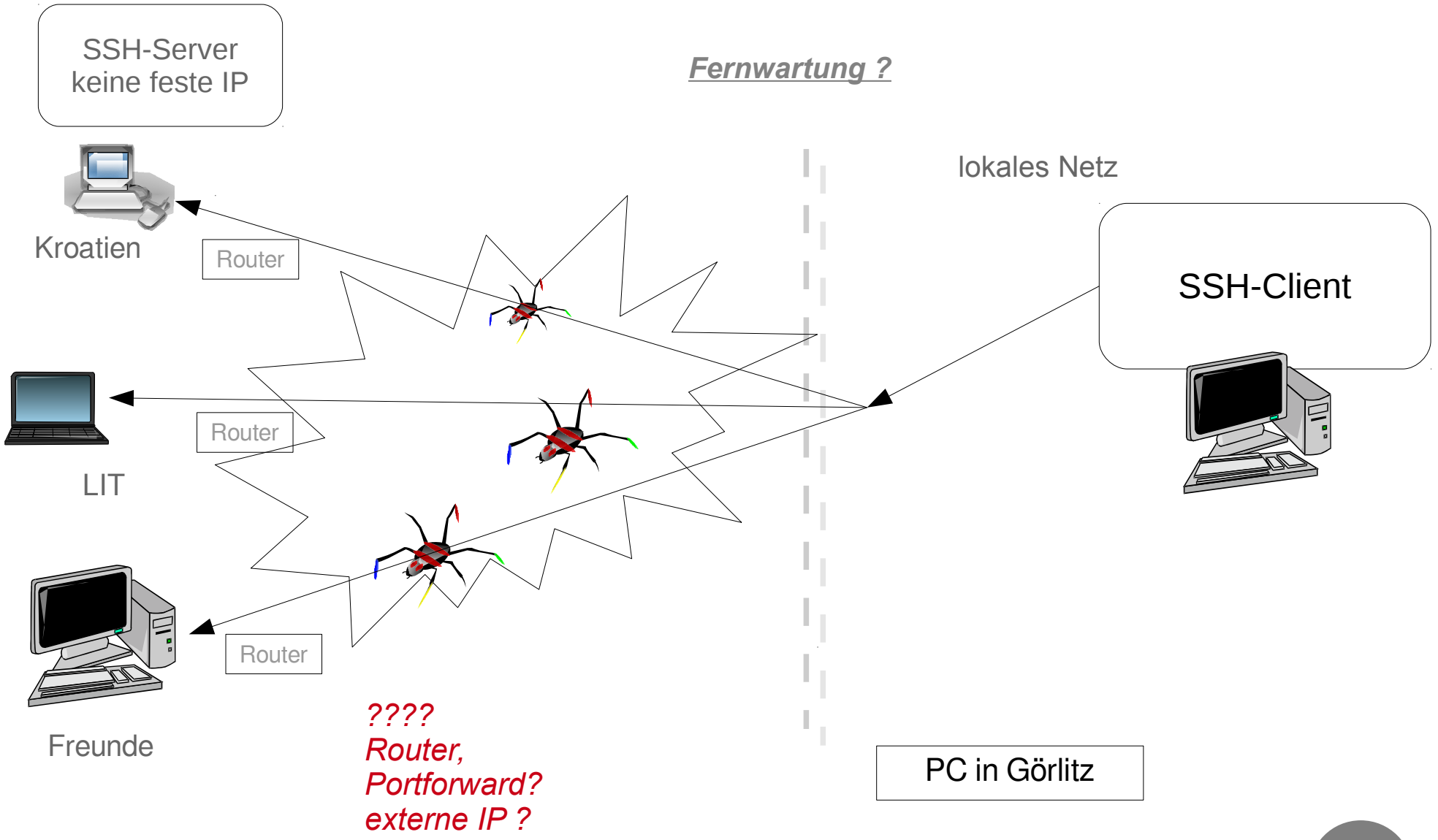
- Was kann ich damit tun
  - Terminal Verbindung
  - Ausgabe von grafischen Programmen umleiten
  - Filemanager verteilt verwenden
  - mit Tunnel beliebige Programme sicher durch das Netz bringen
  - Proxy

- Familiennetzwerk mit SSH
  - Datenaustausch
    - SSH, statt Mail Anhang
  - Fernzugriff
    - zur Wartung bei anderen
    - zum Zugriff auf den eigenen PC
  - Netz zwischen Benutzern, die sich gegenseitig vertrauen
  - Intranet hat inzwischen jeder
    - Webserver für zu Hause
    - NAS
    - Router erlauben Exponierung zum Internet
    - Datenaustausch remote
  - Zugriff auf den eigenen Desktop mit X2Go

sicheres Netz in unsicheren Zeiten …



sicheres Netz in unsicheren Zeiten …



VPN

- Warum nicht VPN?
  - komplex
  - verschied. Protokolle
    - https://www.goldenfrog.com/de/vyprvpn/features/vpn-protocols
  - verbindet Netze, nicht nur Ports
  - Traffic geht u.U. über die Firma
    - Arbeitsplatz zu Hause

- VPN Anbieter mit Problemen
  - https://www.heise.de/newsticker/meldung/VPN-Anbieter-Aktivisten-beklagen-Datenmissbrauch-3795523.html
    - „Code-Analyse belegt Weitergabe von Daten“
  - https://netzpolitik.org/2017/facebook-spioniert-nutzer-seines-vpn-dienstes-aus/
  - uvam.

- SSH
  - einfach
  - Ports werden weitergereicht
  - Authentifizierung
    - Pw
    - Schlüssel
  - Verschlüsselung ausreichend
  - sicher
  - weniger Fehler durch Benutzer
  - durch Firewalls hindurch einsetzbar

Voraussetzungen für die folgenden Abschnitte

- Installieren von Programmen
  - Synaptic, apt-get
  - Hilfesystem (man, info, Wikis)
- Terminal
  - öffnen, einfache Kommandos absenden
  - Arbeiten als root, sudo -s
  - Editieren von Konfigurationen
  - einfache Skripte
- Netzwerk
  - Internetadressen, Namensauflösung,
  - Dienste, Ports (in /etc/services)
  - Router, Modem
  - lokales Netz
  - Rolle des Providers
    - Lieferant der Internetadresse
- alle Details lassen sich im Netz nachlesen
  - Video, Ubucon 2011: https://www.youtube.com/watch?v=Hxsl-jj2Bq0
  - https://www.lug-ottobrunn.de/wiki/SSH_Simple
  - https://wiki.x2go.org/doku.php
  - http://www.openssh.com/

Remote Zugriff mit SSH, Installation

- SSH installieren (auf allen beteiligten PCs)
  - # apt-get install **ssh**
    - **Server absichern**
    - **/etc/ssh/sshd_config editieren**
    - Passwort-Login für alle Benutzer sperren

in /etc/ssh/sshd_config:

PermitRootLogin no
PasswordAuthentication no

- Router u.U. freischalten
  - Port 22 (bzw. der für SSH gewählte Port) muss zum Server-PC weitergeleitet werden
  - Firewall im Router abschalten, bzw. den SSH Port freischalten

- ohne Portforward
  - root Server im Netz mieten (ca. 80€/Jahr als VM)
  - SSH Traffic darüber leiten
  - 24/7 online
  - mit SSH Keys absichern, fast wartungsfrei

	#	Service Name	External Start Port	External End Port	Internal Start Port	Internal End Port	Internal IP address
○	1	ssh pi2					192.168.10.19
○	2	ssh veserver at					192.168.10.35
○	3	ssh 3 veserver					192.168.10.35
○	4	SSH22	22	22	22	22	192.168.10.232
○	5	SSH pi					192.168.10.29

Remote Zugriff mit SSH, Installation

- Schlüsselpaar erzeugen und sichern
  - $ ssh-keygen -t rsa -b 4096
  - für jeden Benutzer auf dem Client

```
-----BEGIN RSA PRIVATE KEY-----
MIIEowIBAAKCAQEAxR6YbXtsxpyyOIV+rcQv9KQRhSCNBbDCWey/0uMLYyJrUpqM
CrOBhlN1Ea7Oig1jdxOeSsTsyk3IscTr8OdnMd5QiIxyeaxqN81KJwxB14CYn0Sq
…
DcjtAoGBAOEoPou4Iyd3ArwMFQf8sT6ZHUuURmqFaJt5Doo4/EeDzFbUKFryRNd6
moFOlhIsI4XqGVVQL4OJ1vXpTz5sSIW9w199a1SGjF6ExgjZBMVoE4MnfeonqmwI
cY240i70VfH5FKfF/UCs95zhWAD+76CV2MRva9+9xD0lLoevvxKO
-----END RSA PRIVATE KEY-----
```

```
public Key:
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDFHphte2zGnLI4hX6t
…
ZIuCHv1XIGv2Wg2zooIxVMtnrNIMOyHN/bts4e17HmTz5 rleo@ls5
```

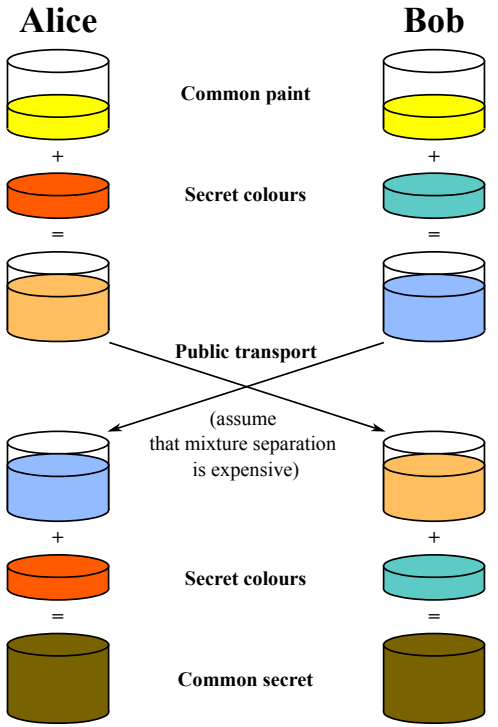
- Kopieren von id_rsa.pub auf den Server
  - mit PW anmelden, erst abmelden,wenn die Schlüssel funktionieren
  - id_rsa.pub mit copy/paste im Terminal in ~/.ssh/authorized_keys auf dem Server eintragen
  - sshd_config umstellen, **PasswordAuthentication no**
  - mit anderem Login testen !
  - erst dann 1. Login abmelden
  -

Sicherheit

- SSH unsicher?
  - NSA in den Medien,
    - irgendwann ist jeder Schlüssel knackbar …
    - man braucht nur teure Computer
    - Quantencomputer
  - hoher Aufwand durch Regierungen
    - http://www.zeit.de/digital/internet/2016-07/cyberangriffe-hacker-innenministerium-thomas-de-maiziere
    - „Gleichzeitig plant das Ministerium jedoch abseits der hier formulierten Strategie, eine weitere Behörde aufzubauen. Deren einziges Ziel: **Verschlüsselte Daten zu knacken**, damit Dienste und Behörden sie trotzdem lesen können. „
    - „ Die Verschlüsselung privater Kommunikation müsse zum Standard werden und nicht mehr nur die Ausnahme sein. ¨
- Stimmt da was nicht?
  - schauen wir, wie SSH arbeitet
- Kryptographie
  - Verschlüsselung einfach
  - Entschlüsselung auch einfach,wenn man das ‚Geheimnis' kennt
  - **gilt als sicher, wenn der Aufwand zur Entschlüsselung wirtschaftlich nicht machbar ist**
  - mit Computern kann der Aufwand so hoch getrieben werden, dass eine Entschlüsselung ausgeschlossen ist
  - Verfahren ist nicht geheim, Schlüssel sind geheim,

Sicherheit

- wie arbeitet SSH
  - https://en.wikipedia.org/wiki/Diffie%E2%80%93Hellman_key_exchange#General_overview
  - https://www.digitalocean.com/community/tutorials/understanding-the-ssh-encryption-and-connection-process
  - http://morknet.de/smork/entry/sichere-konfiguration-fuer-openssh/
  - https://www.heise.de/security/artikel/Diffie-Hellman-Verfahren-270980.html
  -
- sichererer Kanal mit beiden Schlüsseln, meist 4096 Bit, asymm. Verschlüsselung
  - Hostkey → Prüfung korrekter Server
  - eigene Keys → Prüfung korrekter User
  - Generierung eines weiteren kurzen Schlüssels
  - für symmetrische. Verschlüsselung
    - dieser Schlüssel wird nicht ausgetauscht
    - auf beiden Seiten identisch generiert
  - dieser Schlüsseltausch gilt als angreifbar (?), siehe Heise.de



- Datenaustausch ab dann mit symmetrischem Schlüssel, meist 128 Bit
  - schnell
  - nicht knackbar
  - wird in best. Abständen geändert.

Sicherheit, Aufwand

- Turm zu Hanoi
  - 2 Scheiben gehen schnell
  - Was ist mit 64 Scheiben?
  - 30 Scheiben 34 Jahre, 60 Scheiben 36,6 Milliarden Jahre

- mit Binärzahlen https://micahflee.com/2013/01/no-really-the-nsa-cant-break-your-crypto/
  - 128 Bit Key erraten
  - 100010100110110111000100100111000001100001010100001110101101011000010000000001101000101101110010010100001110010111111011111011100
  - 184.003.411.495.303.822.475.448.869.063.584.250.844
  - 2000 Milliarden Computer, jeder 100000 Tests/Sekunde (100 Milliarden $)
  - 1,701,411,834,604,692,317,316 Sekunden.
  - 28,356,863,910,078,205,288 Minuten
  - 472,614,398,501,303,421 Stunden
  - 19,692,266,604,220,975 Tage
  - 53,951,415,354,030 Jahre
  - **53,951,415,354 Jahrtausende !**

  - oder mit Physik, für 256 Bit Keys https://www.schneier.com/blog/archives/2009/09/the_doghouse_cr.html
  - alle Energie der Sonne über 32 Jahre reicht, um einen 192 Bit Key zu knacken
  - Solange wir Computer aus normaler Materie bauen, sind 256 Bit Keys sicher
  - in der Realität sind 128 Bit mehr als genug, s.o.
  - d.h. auch mit Quantencomputern ist symm. Verschlüsselung sicher

  - das gilt nicht für die public/private Keys
  - Hier kommt es auf die Zerlegung in Primfaktoren an,
    und die ist mit Quantencomputern schneller lösbar, wenn es mal welche geben sollte
  - es gibt Ansätze den 1. Schlüsseltausch bei SSH dagegen zu sichern.

Sicherheit

- SSH unsicher?
- Angriffsszenarien
  - SSH Software kompromittiert
    - nur signierte Pakete installieren
    - Vertrauen in die Entwickler nötig
    - gilt in Linux immer
  - Server/Client kompromittiert
    - unter eigener Kontrolle halten
    - Logfiles anschauen, /var/log/auth.log
    - Login Versuche feststellen
    - Passwort Login abschalten
    - ~ /.ssh/authorized_keys schützen
    - Login Restriktionen verwenden
  - social engineering
    - schwer absicherbar
    - menschl. Schwächen werden ausgenutzt
- Server absichern
  - kein root login
  - Updates aktuell einspielen

SSH-Netz

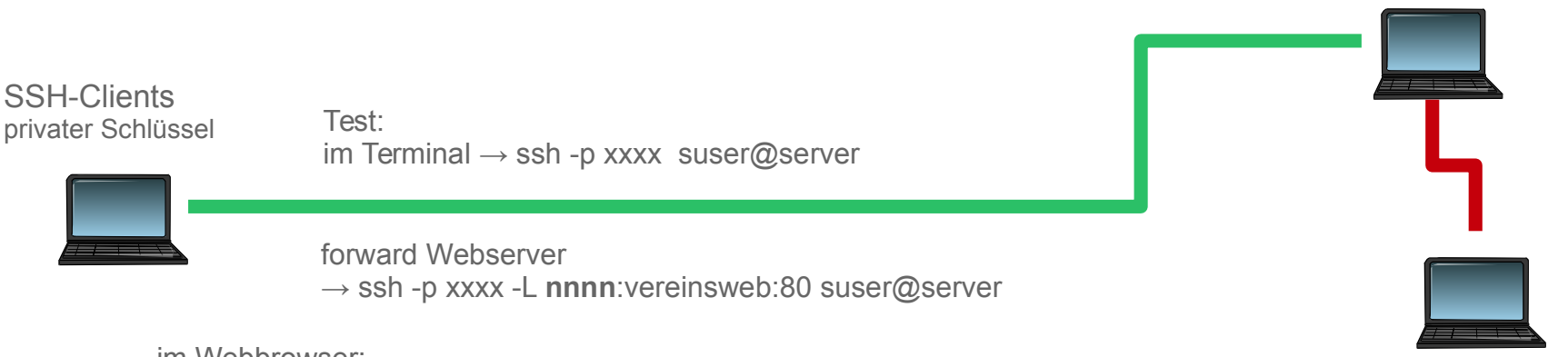
- Client-Server Struktur
  - jeder PC kann gleichzeitig Client und Server sein
  - Client-Benutzer hat beide Schlüssel
  - Server-Benutzer hat den öffentlichen Schlüssel des Client

- Wer → Wohin ?
  - Client initiiert Verbindung zu einem Benutzer auf dem Server
  - ***ssh benutzer@server_IP_Adresse***
  - Client bekommt die Rechte von ***'benutzer'*** auf dem Server
  - d.h. der '***benutzer***' am Server stellt seinen Account zur Verfügung
  - Vertrauen untereinander nötig (Familie,Freunde)
  - oder sicheren Account anlegen

- Anwendungen
  - Terminal, Filemanager, Desktop, Tunnel
  - je eine Demo

SSH Tunnel, lokaler Webserver

- Durchleitung von Ports
  - vom PC to PC
  - PC muss erreichbar sein, im Netz der Gegenseite
  - PC braucht kein SSH
  - es muss ein PC im Netz der Gegenseite SSH können
    - Client – Server-PC verschlüsselt
    - Server-PC anderer PC unverschlüsselt
- Beispiel:
  - Vereinszeitung auf einem Webserver
  - früher: lokale Zeitung, ca. 20 Leser, **Informationen lokal**
  - heute: FB Gruppe 1 Milliarde Leser (?), **Informationen global**
  - mit SSH: wieder die 20 Leser, **Informationen lokal**



SFTP, eigene Cloud

- gemeinsame Daten an sicherer Stelle:
  - auf externem PC, gemietet,
  - sicher, da **root-Jail**
  - kein SSH Zugriff, nur SFTP
  - mit LUKS Container als Backend

- in sshd_config
  - sftp Subsystem
  - sftp User in eigene Gruppe
  - **root-Jail**
  - LUKS-Container

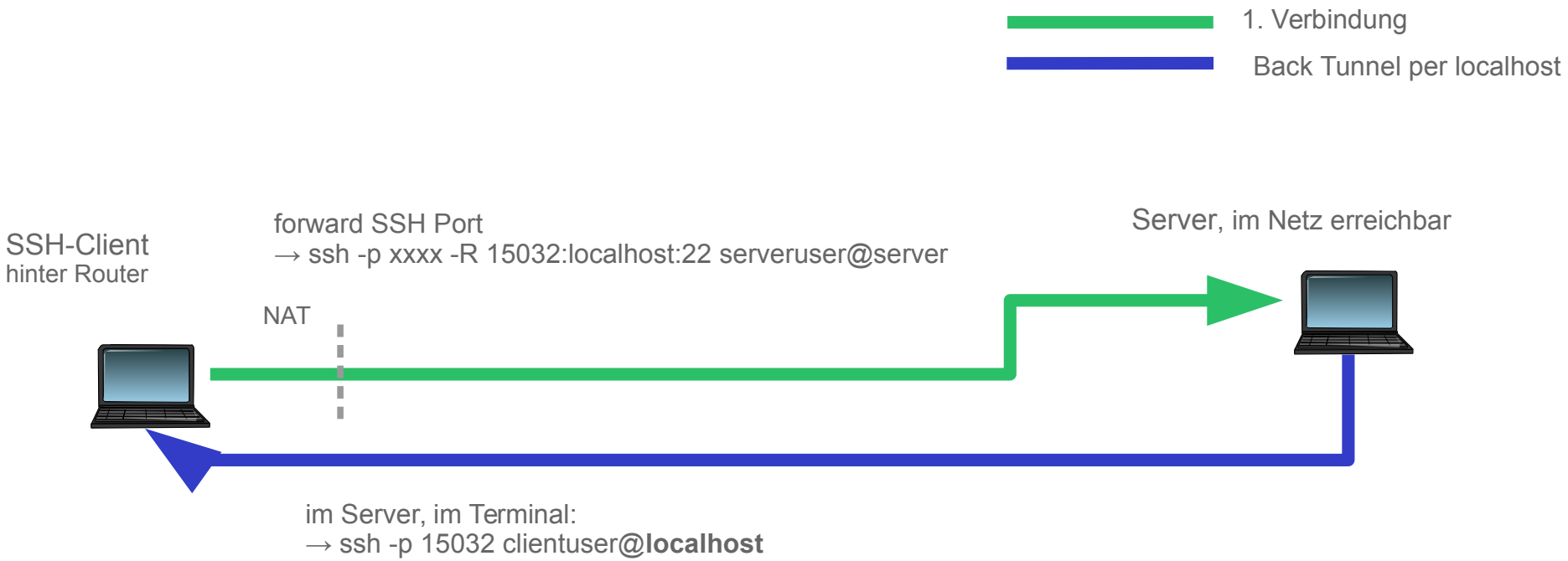
in sshd_config

```
Subsystem sftp internal-sftp

Match Group sftponly
        ChrootDirectory /mnt/cont/sftp/%u
        X11Forwarding no
        AllowTcpForwarding no
        ForceCommand internal-sftp
        PasswordAuthentication yes
```

SSH Tunnel, PC hinter Router

- PC Client ist hinter Router
  - vom Client auf den Server verbinden
  - mit SSH -R
  - dann mit localhost vom Server auf den Client verbinden



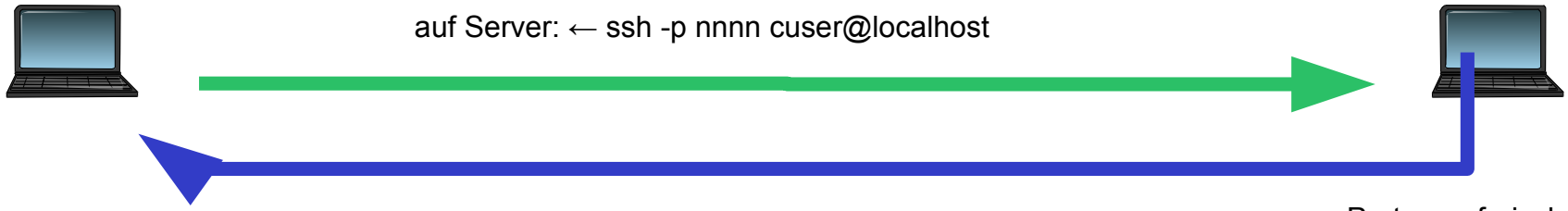
SSH Tunnel, noch mehr

SSH-Clients
irgendwo in der Welt

Server:
Port xxxx

ssh -p xxxx -R nnnn:localhost:22 suser@server

auf Server: ← ssh -p nnnn cuser@localhost



Port xxxx freischalten
im Router
offen zum Internet

Brückenserver, Port xxxx
keine privaten Keys
gut gesichert

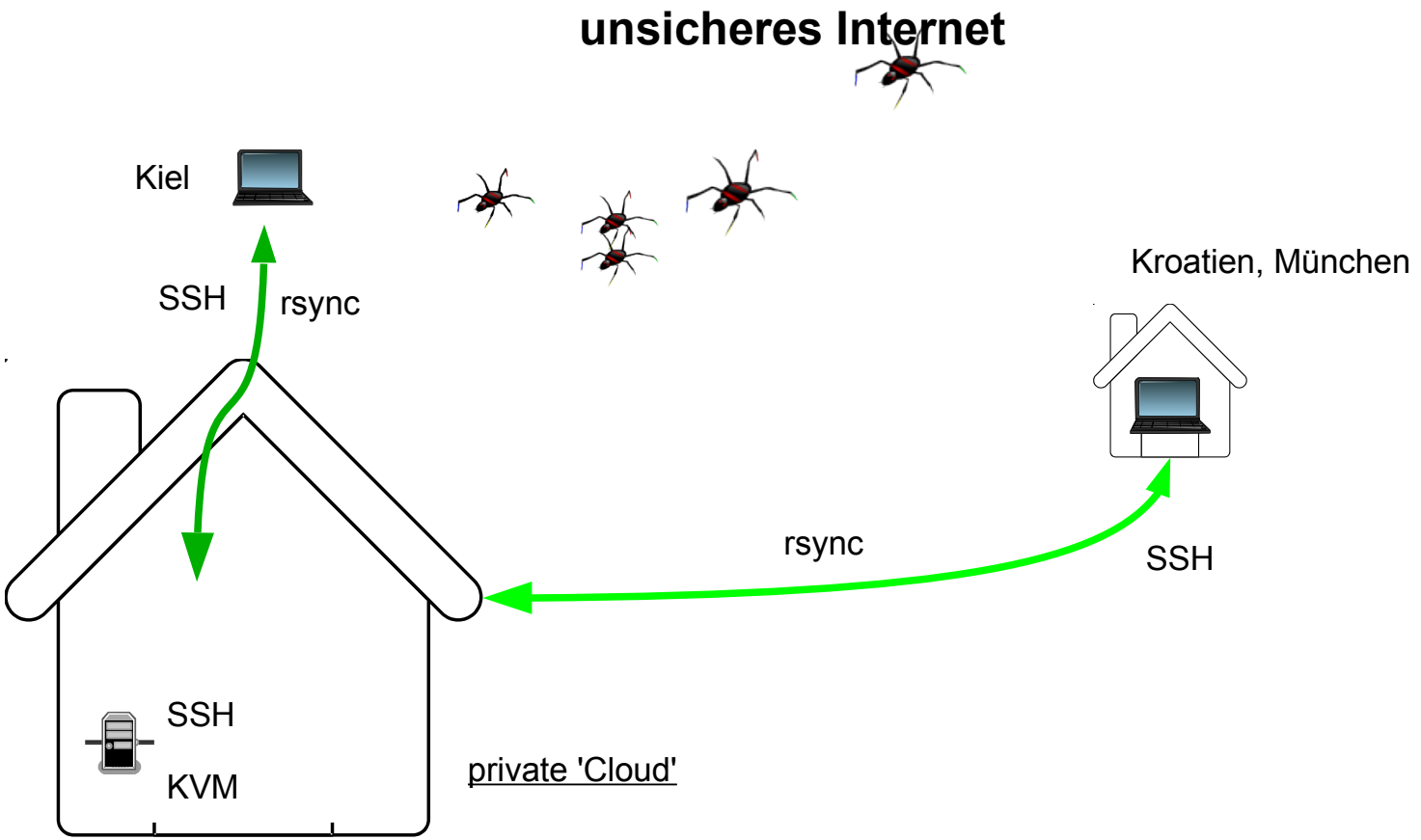
Test: ssh -p xxxx buser@brückenserver

← ssh -p xxxx -R **nnnn**:localhost:22 buser@brückenserver

auf PC: ssh -p xxxx **-A** -t buser@brückenserver ssh -p nnnn cuser@localhost

Demos

- Demonstration
  - mit X2Go nach Hause
  - Terminal nach Kroatien, mit -R
  - mit Bridge



Danke

- Take Home Message

  - eine sicheres Netz mit Linux Bordmitteln ist nicht schwer
  - mit **SSH -R** kann man jeden (konfigurierten) Rechner remote erreichen

Vielen Dank für Eure Aufmerksamkeit.

Richard Albrecht

LUG-Ottobrunn

Richard Albrecht
Linux in Görlitz