Remote auf Rechnern am Informatikum arbeiten

Andreas Mäder

1 Remotezugang

Für spezielle, teilweise lizensierte, Software, die im Rahmen von Lehrveranstaltungen bei uns genutzt wird, kann es notwendig sein, sich direkt auf den TAMS-Rechnern einzuloggen. Dies geht natürlich auch von allen Linux-PCs des Fachbereichs (dann ohne VPN im Hintergrund), aber gerade für die Arbeit mit dem „eigenen Rechner“ sind hier die grundlegenden Schritte skiziert. Kontaktiert mich für Ergänzungen und Korrekturen! (A. Mäder)

- Speziell, wenn die benutzten Programme grafische Oberflächen haben, muss man die hier beschriebenen Mechanismen nutzen. Dazu noch eine Anmerkung zum technischen Hintergrund: je nach Art der Verbindung (s.u.) findet das Rendering der Anzeige auf dem eigenen Rechner (X-forwarding) oder auf dem TAMS-PC statt. Zusätzliche Einschränkungen gibt es beim 3D-Rendering und dem Zugriff auf die Grafikhardware. So kann zwar Remote auf den Grafikkarten gerechnet werden (OpenCL, CUDA etc.), die Grafik wird aber in Software erzeugt und nicht auf der installierten GPU, wenn man nicht an der Konsole eingeloggt ist. Deshalb kann es bei einigen Anwendungsszenarien zu Problemen bei der Grafikausgabe kommen: Fehlermeldungen, schwarze Bildschirme …

- Für Remote-Verbindungen sollten folgende Pool-PCs laufen: `tams191`, `tams192` … `tams199`

  Wegen Updates-/Sicherungen etc. kann es sein, dass einzelne Rechner kurzfristig nicht erreichbar sind, dann sollte man den Nächsten probieren. Tipp: nach dem Login wird angezeigt, welche Nutzer auf dem PC eingeloggt sind. Ansonsten kann man mit dem Befehl `who` nachsehen, wie viele andere Nutzer auf den Maschinen sind, so dass sich nicht alle auf einem PC „drängeln“.

- **Achtung:** die Rechner sind so konfiguriert, dass man nach dem Einloggen mit einem lokalen Verzeichnis arbeitet. Das ist deutlich schneller als die Dateien auf dem Dateiserver zu nutzen, hat aber auch zur Folge, dass
  - es keine Backups gibt!
  - beim nächsten Login auf einem anderen Rechner, die Daten dort nicht vorhanden sind!

  Wenn ihr euch eingeloggt habt, ist das Login-Verzeichnis des Informatik-Dateiservers unter `infhome` erreichbar. Für die praktische Arbeit solltet man also
  - entweder direkt auf dem Dateiserver arbeiten: `cd infhome`
  - oder die Daten (Verzeichnisse) vor und nach der Arbeit über das Netzwerk-Dateisystem synchronisieren. Befehle dazu sind beispielsweise: `rsync -av` ⟨*sourceDir*⟩`/` ⟨*destDir*⟩ oder `cp -pr` ⟨*sourceDir*⟩ ⟨*destDir*⟩

Betriebssysteme

- Hier sind erst einmal die Schritte für ein **Linux** (Debian, Ubuntu, openSuSE etc.) beschrieben. Dort sind in der Regel alle benötigten Programme installiert oder lassen sich über die Paketverwaltung „nachrüsten“. Die sind: *X-Server*, *ssh* und ggf. *VNC-Client Software*, beispielsweise `vncviewer`, `remmina` oder `krdc`.
- Da **macOS** historisch auf BSD-Unix aufbaut, sollten die entsprechenden Mechanismen der Linux Beschreibung auch funktionieren. Wie die Benutzung genau aussieht und welche Programme noch fehlen (VNC-Client?), konnte mangels Hardware nicht ausprobiert werden.
- Bei **Windows** Systemen muss man differenzieren, welche Services und Protokolle genutzt werden sollen.

  SSH-Verbindung: ist direkt mit „Bordmitteln“ von Windows möglich: *Eingabeaufforderung* oder *Power-Shell*

  VPN Setup: ist direkt bei den *Netzwerkeinstellungen* verfügbar.

  X-Forwarding: hier braucht man Software, die einen X-Server bereitstellt – Tipp: *SmarTTY* oder *MobaXterm*, beides sind kommerzielle Produkte, für die aber eine Privat-/Testnutzung möglich ist.

  VNC-Client: es gibt zahlreiche Programme für Windows, wie z.B.: *TightVNC*, *RealVNC*, *UltraVNC* ...

  Als weitere Alternative könnte das WSL (Windows-Subsystem für Linux) genutzt werden, auch hier wurden noch keine Tests durchgeführt...

2 Technik

Prinzipiell gibt es mehrere Möglichkeiten „Remote“ zu arbeiten und die Programme lokal anzuzeigen und zu bedienen. Dies betrifft sowohl den Zugang zu den Ziel-Rechnern, als auch die Methode der Grafikdarstellung.

2.1 SSH vs. VPN

Beim Login aus externen Netzen, also nicht von Informatik Pool-PCs aus, muss sich der eigene Rechner erst mit dem Informatik-Netz verbinden.

- Ein direkter Zugriff auf Informatik-Rechner ist von Außen ausschließlich auf den **SSH-Server**, die `rzssh1.informatik.uni-hamburg.de`, beschränkt. Um auf den TAMS-PCs zu arbeiten, müssen externe SSH-Verbindungen immer diesen Rechner nutzen. Da dort alle externen Verbindungen auflaufen, ist es aus Performancegründen nicht sinnvoll direkt dort zu arbeiten, ohnehin sind die für uns relevanten Programme dort nicht verfügbar! Man kann den Rechner aber als Relay nutzen und sich dann weiter mit den TAMS-PCs verbinden.
- Einfacher ist es, den eigenen Rechner per **VPN** (Virtual Private Network) in das Informatik-Netz einzubinden. Anleitungen dazu finden sich auf den Rechenzentrums-Seiten unter: `www.inf.uni-hamburg.de/inst/irz/it-services/private-devices/vpn-clients.html`. Danach ist der eigene Rechner in Informatik-internen Netz und man kann sich mit den TAMS-PCs verbinden.

2.2 X-forwarding vs. VNC

Technisch gesehen, gibt es zwei Szenarien, wie GUI-Elemente auf dem eigenen Rechner dargestellt werden.

- Das **X-forwarding** ist der ursprüngliche, alte Mechanismus, der netzwerktransparenten Grafikausgabe von Unix Systemen. Da aber alle grafischen Zeichenbefehle über das Netz übertragen werden und Client-seitig von einem X-Server verarbeitet werden müssen, ist eine flüssige Bedienung und Darstellung nur in lokalen Netzen sinnvoll oder wenn keine komplizierten GUIs vorhanden sind.
- Das **VNC-Protokoll** ist, ähnlich RDP bei Windows Systemen, eine Möglichkeit komplette Bildschirminhalte Remote darzustellen und den Rechner mit Tastatur und Maus fernzusteuern. In der Regel setzt man dabei eine eigene grafische Oberfläche auf, vergleichbar mit dem direkten Einloggen an der Konsole. Auch hier lassen sich, je nach Vorliebe, verschiedene Desktopumgebungen, bzw. Fenstermanager starten.

3 Vorgehen

Als „Rezept“ folgen jetzt die Schritte, die bei den verschiedenen Szenarien notwendig sind. Von den möglichen Alternativen ist meine Präferenz die zweite Variante: **VPN + VNC-Server**.

3.1 VPN + X-Forwarding

- VPN starten: nachfolgend wird vorausgesetzt, dass man erfolgreich ein VPN (Verbindung zu `fbivpn.informatik.unihamburg.de`) gestartet hat, siehe RZ-Seiten.
- auf dem eigenen Rechner: *Shell 1*

  1. `xhost tams⟨nnn⟩.informatik.uni-hamburg.de`

     Erlaubt dem TAMS-Rechner eine Verbindung zum eigenen X-Server.

  2. `ssh -Y -l ⟨username⟩ tams⟨nnn⟩.informatik.uni-hamburg.de`

     Öffnet SSH-Verbindung und man erhält eine Shell mit der man dann weiterarbeitet.

  3. ⟨*Kommandozeilenbefehle*⟩

     In der SSH-Shell aus 2. kann jetzt ganz normal gearbeitet werden, wobei GUI-Elemente von gestarteten Programmen zum eigenen Rechner übertragen und dort dargestellt werden. Damit ist auch schon der Nachteil dieser Methode beschrieben. Da die Zeichenbefehle des TAMS-Rechners übertragen und lokal umgesetzt werden, ist, gerade bei langsamen Netzwerkverbindungen, der Bildaufbau sehr zäh und man kann zusehen wie Buttons einzeln gezeichnet werden.

  4. … nach dem Ende der Arbeit beendet man die Remote-Shell ganz normal mit `exit`.

3.2 VPN + VNC-Server/-Client

- VPN starten: nachfolgend wird vorausgesetzt, dass man erfolgreich ein VPN (Verbindung zu `fbivpn.informatik.unihamburg.de`) gestartet hat, siehe RZ-Seiten.
- auf dem eigenen Rechner: *Shell 1*
  1. `ssh -Y -l ⟨username⟩ tams⟨nnn⟩.informatik.uni-hamburg.de`

     Öffnet die SSH-Verbindung und man erhält eine Shell mit der man dann weiterarbeitet.

  Bevor man das erste Mal den VNC-Server auf einem TAMS-Rechner startet, kann (sollte) man noch zwei Konfigurationsschritte vornehmen. Später, wenn man erneut auf diesem Rechner arbeitet, ist das natürlich nicht mehr notwendig.

  2. `vncpasswd`

     Vor den Start des VNC-Servers wird ein Passwort gesetzt. Dieses wird nur für die Verbindung zum VNC-Server benutzt und hat nichts mit den Informatik- oder Uni-Accounts zu tun.
  3. ⟨*Konfiguration des VNC-Servers*⟩

     Der VNC-Server startet eine eigene X-Session, die individuell angepasst werden kann. Voreingestellt startet der Ubuntu-Desktop (Gnome). Wenn man für die Remote-Arbeit ein einfacheres Setup bevorzugt, dann kann man eine entsprechende Konfigurationsdatei `$HOME/.vnc/Xvnc-session` erstellen, die die Desktopoberfläche festlegt und ggf. weitere Programme startet.[1] Als Beispiel, meine `Xvnc-session` für einen LXDE-Desktop:

     ```
     #!/bin/sh
     unset SESSION_MANAGER
     exec /usr/bin/startlxde
     ```

  Anschließend kann man den VNC-Server starten.

  4. `vncserver -localhost no -geometry 1920x1080`

     Auf dem Remote-Rechner wird ein VNC-Server gestartet, mit dem sich der eigene Rechner (im nächsten Schritt) verbinden kann. Beim Start wird die Nummer des X-Displays angezeigt. Wenn kein weiterer X-Server läuft `:1`, sonst `:2` etc. Diese Nummer (⟨*X*⟩) braucht man für den nächsten Schritt und am Ende für das Beenden (`kill`)!

- auf dem eigenen Rechner: *Shell 2*
  1. `vncviewer`

     Nach Eingabe des Rechnernamens `tams`⟨*nnn*⟩`.informatik.uni-hamburg.de:`⟨*X*⟩ und des Passworts öffnet sich ein Fenster, der eingestellte Desktop (s.o.) startet.

- auf dem VNC-Desktop
  1. Man arbeitet auf dem Desktop, wie bei einer lokalen Session...
  2. Zum Beenden loggt man sich an der Desktopumgebung aus -oder- schließt das Fenster -oder- beendet den zuvor gestarteten `vncviewer`-Prozess.

[1] Vergleiche dazu: `/usr/share/xsessions/...desktop`

- in der SSH-Session: *Shell 1*

  5. `vncserver -kill :⟨X⟩`

     Hier muss die Display-Nummer von Start angegeben werden, in der Regel `:1`. Mit dem Kill wird der immer noch laufende Server beendet.

  6. ... anschließend beendet man die Remote-Shell ganz normal mit `exit`.

3.3 SSH-Tunnel + VNC-Server (oder X-Forwarding)

Teilweise gab es Probleme mit der oben beschriebenen Verbindung zum VNC-Server (*VPN + VNC-Server/-Client*), da viele verschiedene Faktoren das Systemverhalten beeinflussen: Linux-/Windows-Versionen, Firewallkonfigurationen etc. Als Alternative zu obiger Methode mit VPN, kann man auch einen SSH-Tunnel über die `rzssh1.informatik.uni-hamburg.de` laufen lassen. Hier benötigt man mehr Terminals (Shells), da zusätzliche Protokoll- und Port-Forward Einstellungen notwendig sind — man beachte die Eingaben in die unterschiedlichen *Shell* ⟨*n*⟩**!**

- auf dem eigenen Rechner: *Shell 1* – SSH-Tunnel einrichten

  1. `ssh -TNL 1337:tams⟨nnn⟩.informatik.uni-hamburg.de:22`
     `⟨username⟩@rzssh1.informatik.unihamburg.de`

     Der remote Port 22 (SSH) des TAMS-PCs wird über die `rzssh1` auf den lokalen Port 1337 getunnelt. Port 1337 ist zwar ein registrierter Port, kann aber auf den meisten Systemen von normalen Nutzern ohne besondere Rechte verwendet werden.

     Nach erfolgreichem Login sind keine weiteren Eingaben mehr möglich. Der SSH-Tunnel muss später (s.u.) mit *Ctrl-C* beendet werden!

- auf dem eigenen Rechner: *Shell 2* – SSH Verbindung (durch den Tunnel) zum Ziel-PC

  1. `ssh ⟨username⟩@localhost -p 1337`

     Login auf dem TAMS-Rechner, um dort den VNC-Server aufzusetzen und zu starten.

     Anmerkung: es kann sein, dass SSH so konfiguriert ist, dass es eine Fehlermeldung wegen der SSH-Schlüssel gibt. In diesem Fall muss man den falschen Schlüssel mit `ssh-keygen -R ...` löschen.

  Bevor man das erste Mal den VNC-Server auf einem TAMS-Rechner startet, kann (sollte) man noch zwei Konfigurationsschritte vornehmen. Später, wenn man erneut auf diesem Rechner arbeitet, ist das natürlich nicht mehr notwendig.

  2. `vncpasswd`

     Vor den Start des VNC-Servers wird ein Passwort gesetzt. Dieses wird nur für die Verbindung zum VNC-Server benutzt und hat nichts mit den Informatik- oder Uni-Accounts zu tun.

  3. ⟨*Konfiguration des VNC-Servers*⟩

     Der VNC-Server startet eine eigene X-Session, die individuell angepasst werden kann. Voreingestellt startet der Ubuntu-Desktop (Gnome). Wenn man für die Remote-Arbeit

ein einfacheres Setup bevorzugt, dann kann man eine entsprechende Konfigurationsdatei `$HOME/.vnc/Xvnc-session` erstellen, die die Desktopoberfläche festlegt und ggf. weitere Programme startet.[2] Als Beispiel, meine `Xvnc-session` für einen LXDE-Desktop:

```
#!/bin/sh
unset SESSION_MANAGER
exec /usr/bin/startlxde
```

Anschließend kann man den VNC-Server starten.

4. `vncserver -geometry 1920x1080`

   Auf dem Remote-Rechner wird ein VNC-Server gestartet, mit dem sich der eigene Rechner (im nächsten Schritt) verbinden kann. Beim Start wird die Nummer des X-Displays angezeigt. Wenn kein weiterer X-Server läuft `:1`, sonst `:2` etc. Diese Nummer (⟨*X*⟩) braucht man im nächsten Schritt für den zweiten SSH-Tunnel und am Ende für das Beenden (`kill`)!

- auf dem eigenen Rechner: *Shell 3* – SSH-Tunnel für den VNC-Desktop

  1. `ssh -L 9900:localhost:590`⟨*X*⟩ ⟨*username*⟩`@127.0.0.1 -p 1337`

     Mit diesem SSH-Tunnel wird die Ausgabe des VNC-Servers (Port `590`⟨*X*⟩) lokal auf Port `9900` umgeleitet.

     Anmerkung: eine Fehlermeldung wegen der SSH-Schlüssel ist auch hier möglich (s.o.).

  2. Man erhält eine SSH-Shell auf dem TAMS-PC, die bis zum Beenden des VNC-Servers aktiv bleiben muss.

- auf dem eigenen Rechner: *Shell 4*

  1. `vncviewer localhost:9900`

     Der VNC-Client wird direkt mit dem zuvor geöffneten Tunnel verbunden.

- auf dem VNC-Desktop

  1. Man arbeitet auf dem Desktop, wie bei einer lokalen Session…
  2. Zum Beenden loggt man sich an der Desktopumgebung aus -oder- schließt das Fenster -oder- beendet den zuvor gestarteten `vncviewer`-Prozess.

- in der SSH-Session: *Shell 3*

  3. Jetzt wird die Remote-Shell beendet: `exit`

- in der SSH-Session: *Shell 2*

  5. `vncserver -kill :`⟨*X*⟩

     Hier muss die Display-Nummer von Start angegeben werden, in der Regel `:1`. Mit dem Kill wird der immer noch laufende Server beendet.

  6. … anschließend beendet man die Remote-Shell ganz normal mit `exit`.

- in der SSH-Session: *Shell 1*

  2. … der ursprüngliche Tunnel über die `rzssh1` kann jetzt durch Eingabe von *CTRL-C* beendet werden.

[2] Vergleiche dazu: `/usr/share/xsessions/...desktop`