Vom Dip zum Tipp: Guacamole

Wenn entfernte Anwendungen über ein Netzwerk angeboten werden müssen und dabei die Installation eines lokalen Clients unerwünscht oder gar unmöglich ist, kann der Einsatz der Web-Applikation Guacamole sinnvoll sein. Diese ist mit effizienten Mitteln in der Lage, die Kommunikation mit einem VNC-Server in ein Fenster eines HTML5-tauglichen Browsers zu verlagern. Harald Jele

Die Web-Applikation Guacamole läuft als Prozess in einer Apache Tomcat-Umgebung und ist dabei in der Lage, sich mit einem VNC-Server zu verbinden. Die graphische Ausgabe des VNC-Servers wird dabei durch Guacamole so aufbereitet, dass diese via HTTP mit einem HTML5-tauglichen Browser zur Anzeige gebracht werden kann. Die aktuelle Version 0.5 von Guacamole beherrscht ausschließlich das Weiterreichen von VNC-Verbindungen. Für zukünftige Ausgaben ist jedoch auch die Unterstützung des RDP-Protokolls sowie die Einbindung von SSH-Bibliotheken angekündigt [1].

Guacamole: Eigenschaften

- VNC-ähnliche Übertragungsraten
- HTML5 Client
- kein Plugin im Browser notwendig
- Unterstützung von VNC, weitere Protokolle in Entwicklung
- eingebaute Bildschirmtastatur
- Clipboardfunktionen
- Unterstützung von Touch Screens
- offenes Framework

VNC zuerst

Für einen reibungslosen Einsatz gilt daher, sich zuerst um die Art und Weise der VNC-Anbindung zu kümmern.
Der typische Anwendungsfall, in dem VNC-Verbindungen eine gewichtige Rolle spielen, ist die Fernwartung eines Computers. Ein Vorteil, der dabei auf der Hand liegt, ist, dass dies über (fast) alle Grenzen der üblichen Betriebssysteme hinweg funktioniert. Beim Einsatz des VNC-Protokolls ist zudem unerheblich, ob einzelne Anwendungen oder eine vollständige Desktop-Umgebung über das Netzwerk zugänglich gemacht werden sollen. Der VNC-Server liest in diesem Fall die entfernte graphische Ausgabe mit und sendet diese an einen lokalen VNC-Client. Die lokale Eingabe über Maus und Tastatur werden im Gegenzug durch den Client wiederum der entfernten Anwendung übergeben.
Technisch gesehen basiert die Kommunikation auf dem Protokoll „Remote framebuffer“ (RFB), das in einem offenen Standard spezifiziert ist [2].
Neben den typischen Anwendungsgebieten lassen sich über VNC-Verbindungen zudem mit wenigen Handgriffen einfache Szenarien zum Zweck der Aus- und Weiterbildung umsetzen: Ein Vortragender arbeitet aktiv an seinem Gerät mit einem Programm und gibt den Inhalt seines Bildschirms oder die graphische Ausgabe der betreffenden Anwendung an eine Gruppe von Teilnehmenden frei, die – sofern dies intendiert ist – unterdessen auf ihren Geräten allein das Geschehen verfolgen aber aktiv nicht eingreifen können.
Die Weiterentwicklung des VNC-Protokolls ist über die Jahre hinweg kontinuierlich fortgeschritten, sodass VNC-Verbindungen neben ihrer einfachen Konfiguration in stabilen Netzwerken als sehr zuverlässig angesehen werden können.
Die hier angedeutete Vielfältigkeit in den Anwendungsgebieten zählt sicher zu den großen Vorteilen, die der Einsatz des VNC-Protokolls mit sich bringt. Dass dem gegenüber auch Nachteile in Kauf genommen werden müssen, liegt auf der Hand bzw. zeigt sich beispielsweise auch in den Stärken der Konkurrenzmodelle:

- der Ansatz von NoMachine mit dem Produkt NX [3] bietet bessere Mechanismen zur Kompression der zu übertragenden Daten sowie effziente Proxy-Funktionen zum Ausgleich von mitunter schwankenden Latenzzeiten von Netzwerkverbindungen
- x2go [4] ist nicht für die Fernwartung gedacht, sondern bringt durch einen Terminalserver allein entfernte Anwendungen auf einen lokalen Computer. Dabei werden diese nahtlos (seamless) am lokalen Desktop angezeigt. Soundausgaben oder auch lokale Geräte wie Festplatten können dabei an den Server weitergereicht werden. x2go-Clients existieren für mehrere, aber – im Gegensatz zu VNC – nicht für alle gängigen Betriebssysteme.

Da Guacamole mit dem VNC-Server kommuniziert, sind dessen Fähigkeiten und funktionale Ein-

schränkungen auch für die Kommunikation mit der Web-Applikation bestimmend.
Die meisten Desktop-Umgebungen bringen „unter der Haube“ ohnehin einen VNC-Server mit, der über eine graphische Oberfläche einfach konfiguriert werden kann. Die Möglichkeiten, dabei besondere Einstellungen vornehmen zu können, fallen in der Regel jedoch eher bescheiden aus. Im System-Menü eines Gnome2 basierten Linux-Desktops findet sich im Zweig zu den „Einstellungen“ der Punkt „Entfernter Bildschirm“. Hier kann die Anzeige des eigenen Desktops über das VNC-Protokoll sehr einfach freigegeben werden.

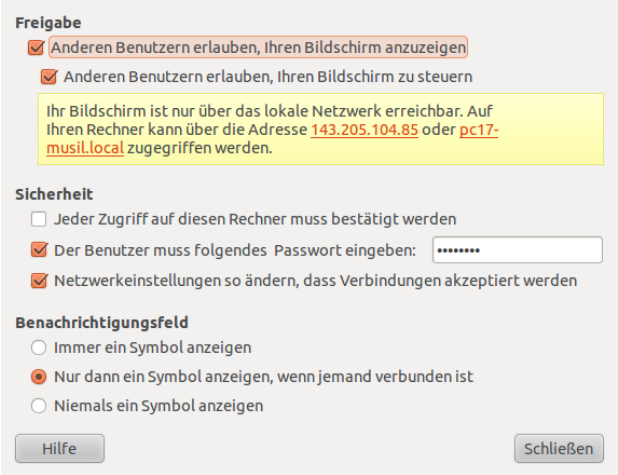

Abbildung 1: Einstellungen des Menüpunkts zum „Entfernten Bildschirm“ unter Gnome2.

Sind die Einstellungen entsprechend geglückt, kann über einen VNC-Client auf Port 5900 und entsprechend über Guacamole eine Verbindung aufgebaut werden. Zur Ansicht gelangt dabei der Inhalt des gesamten Desktops.

Sollen dagegen nur einzelne Anwendungen über das VNC-Protokoll nach außen weitergegeben werden oder sind mehrere Benutzer gleichzeitig an ein und demselben System zu bedienen, ist die Installation und Konfiguration eines eigenen VNC-Servers sinnvoll, um die Prozesse der einzelnen Benutzer weiteren Ports zuweisen zu können.
Im Repository eines Ubuntu-Systems sind folgende Anwendungen integriert:

```
apt-cache search vnc-server
[…]
linuxvnc - VNC-Server, ermöglicht den entfernten
Zugriff auf eine TTY-Schnittstelle
tightvncserver - VNC(Virtuelle(r)) Netzwerk Com-
puter) Server Software
vnc4server - Virtual Network Computing (VNC)-
Serveranwendung
[…]
x11vnc - VNC-Server gestattet Zugriff auf exis-
tierende X-Umgebung von außerhalb
[…]
```

Zur Frage, welcher VNC-Server installiert werden soll, existieren im Umfeld von Guacamole einige Diskussionen. Die Entscheidung, die man hier trifft, ist im Wesentlichen wohl von den eigenen Vorlieben und Erfahrungen abhängig. Die Entwickler selbst äußern sich in dieser Sache eher zurückhaltend; schließlich sollten sich alle VNC-Server ja gleich oder sehr ähnlich verhalten.
Für das nachfolgende Beispiel wird das Paket „vnc4server“ installiert. Dabei gelangt jener Server ins System, der im Lauf seiner Geschichte auch als „RealVNC“ bekannt wurde und dessen Code inzwischen von Entwicklern der gleichnamigen Firma gepflegt wird.
Der Serverprozess kann zwar einfach auf der Kommandozeile gestartet werden; sinnvoll für die weitere Arbeit sind aber kleine Skripts zum Starten und Stoppen allemal. Mit „vncserver-start.sh“

```
#!/bin/bash

DIS=2

rm -f /tmp/.X${DIS}-lock
rm -f /tmp/.X11-unix/X${DIS}

unset XAUTHORITY
unset DISPLAY

killall ssh-agent

# start ssh-agent
ssh-agent > $HOME/ssh-agent.sh
source      $HOME/ssh-agent.sh
rm -f       $HOME/ssh-agent.sh

vncserver -geometry 1024x768 -depth 24 :$DIS
```

wird ein VNC-Server-Prozess gestartet, dessen graphische Ausgabe eine Geometrie von 1024x768 Bildpunkten mit einer Farbtiefe von 24 bit abbildet und diese auf Port 5902 (`DIS=2`) leitet. Im Vorfeld werden dabei eventuell übriggebliebene Lockfiles entfernt und der „ssh-agent“ für die Authentifizierung weiterer SSH-Verbindungen gestartet.

Das Stoppen des Serverprozesses kann beispielsweise mit „vncserver-stop.sh“ erfolgen:

```
#!/bin/bash
DP=2
vncserver -kill :$DP
```

Beim erstmaligen Start des Servers wird von diesem, nach dem Erfragen eines Passworts, im Homeverzeichnis des Benutzers ein Verzeichnis `.vnc` angelegt. In diesem ist neben einem PID-, einem LOG- und einem Passwortfile letztlich auch ein Startup-Skript „xstartup“ hinterlegt, das mit Standardwerten versehen ist. Dieses ist für die weitere Arbeit anzupassen. Im nachfolgenden Beispiel wird davon ausgegangen, dass beim Herstellen einer VNC-Verbindung ein einfacher Windowmanager („fvwm2“), zwei Anwendungen („gedit“ und „musil_start.sh“) sowie das Hilfsprogramm „vnc-

config“ (für das reibungslose Arbeiten mit der Zwischenablage) gestartet werden:

```
#!/bin/sh

# This is $HOME/.vnc/xstartup
# Uncomment the following two lines for normal
# desktop:
# unset SESSION_MANAGER
# exec /etc/X11/xinit/xinitrc

[ -x /etc/vnc/xstartup ] &&
    exec /etc/vnc/xstartup
[ -r $HOME/.Xresources ] &&
    xrdb $HOME/.Xresources
fvwm2 &
vncconfig -iconic &
gedit &
$HOME/musil2012/musil_start.sh &
```

Ein einfacher Test mit einem VNC-Client wie „gvncviewer rechnername:2“
bringt folgende Anzeige zur Ansicht:

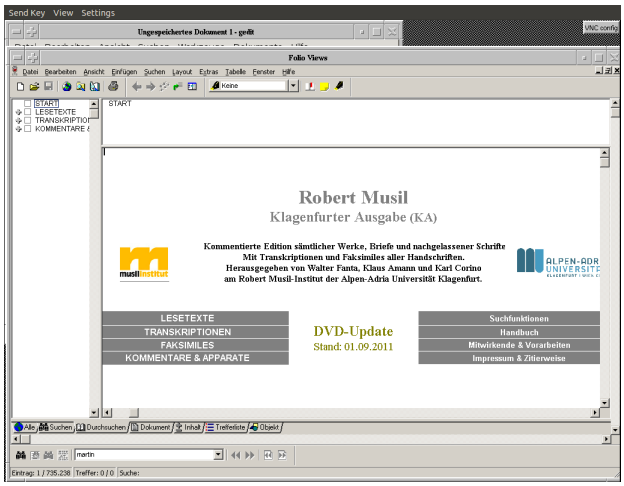

Abbildung 2: Anzeige der gestarteten Anwendungen im VNC-Client.

Das in Abbildung 2 rechts oben als Icon angezeigte „vncconfig“ sollte in diesem Installationsvorschlag immer mitgestartet werden. Anderenfalls ist das Arbeiten mit dem Clipboard auch in Guacamole einigermaßen gestört oder gar unmöglich.

Guacamole danach

Ist das Unterfangen bis hierher geglückt, sollten die nachfolgenden Schritte vergleichsweise einfach gelingen.
In den üblichen Software-Repositories ist Guacamole (noch) nicht vorhanden. Das Installieren und vor allem das Einspielen von Updates gelingt daher über die Paketmanager der Distributionen nicht. Pakete für Debian, Ubuntu und Fedora finden sich neben Anweisungen zur Installation im Download-Bereich der Homepage [1].
Die Entwickler bieten für Ubuntu eine tar.gz-Datei an, die die entsprechenden Ubuntu-Pakete, passend zur Version des Systems, beinhaltet. Aktuell ist dies die Datei

```
guacamole-0.5.0-ubuntu-11.10-i586.tar.gz
```

Nach deren Auspacken mit

```
tar xfvz guacamole-0.5.0-ubuntu-11.10-
        i586.tar.gz
```

beinhaltet das angelegte Unterverzeichnis folgende Dateien:

```
guacamole_0.5.0_all.deb
guacd_0.5.0_i386.deb
libguac2_0.5.0_i386.deb
libguac2-dev_0.5.0_i386.deb
libguac-client-vnc0_0.5.0_i386.deb
```

Diese werden durch folgenden Aufruf dem System hinzugefügt:

```
sudo dpkg -i guacd_*.deb guacamole_*.deb lib-
guac2_*.deb libguac-client-vnc0_*.deb
```

Da Guacamole als Applikation in einer Apache Tomcat-Umgebung läuft, ist als nächstes diese zu installieren. Für die Kommunikation mit dem VNC-Server benötigt man zusätzlich die entsprechende VNC-Library.

```
sudo apt-get install tomcat6 libvncserver0
```

Die notwendigen symbolischen Links der Applikation zur Tomcat-Umgebung werden bei der Installation nicht angelegt. Damit Tomcat Guacamole „findet“, sind diese händisch anzulegen:

```
sudo ln -s /var/lib/guacamole/guacamole.war
           /var/lib/tomcat6/webapps
sudo ln -s /etc/guacamole/guacamole.properties
           /usr/share/tomcat6/lib
```

Durch einen Neustart des Serverprozesses von Tomcat, werden die gelinkten Files eingelesen – und im Prinzip steht damit die graphische Ausgabe des VNC-Servers der Guacamole-Anwendung zur Verfügung.

```
sudo /etc/init.d/tomcat6 restart
```

Für die nachfolgende Anmeldung an der Web-Applikation werden eigene Benutzernamen-Passwort-Kombinationen in einem einfach strukturierten XML-File verwaltet. Dieses ist entsprechend anzupassen. Die betreffende Datei, die durch den Installationsvorgang angelegt wird, ist

```
/etc/guacamole/user-mapping.xml
```

Die Defaulteinträge sollten aus Sicherheitsgründen auskommentiert oder gelöscht werden. Eigene Einträge sind, je nach Passwortmethode, ent-

sprechend anzulegen. Im einfachen Fall genügt das Bearbeiten folgenden Abschnitts:

```
<authorize username="name" password="passwort">
    <protocol>vnc</protocol>
    <param name="hostname">localhost</param>
    <param name="port">5902</param>
    <param name="password">vncpasswort</param>
</authorize>
```

Mit diesen Einträgen kann sich ein Benutzer mit dem Namen „name“ und dem zugehörigen Passwort „passwort“ mit seinem Browser über Port 8080 die graphische Ausgabe seines VNC-Serverprozesses ansehen, die über einen VNC-Client am selben Rechner auf Port 5902 mit dem VNC-Passwort „vncpasswort“ angezeigt wird.

```
http://rechnername:8080/guacamole/
```

Der Anmeldebildschirm dazu ist entsprechend einfach gehalten.

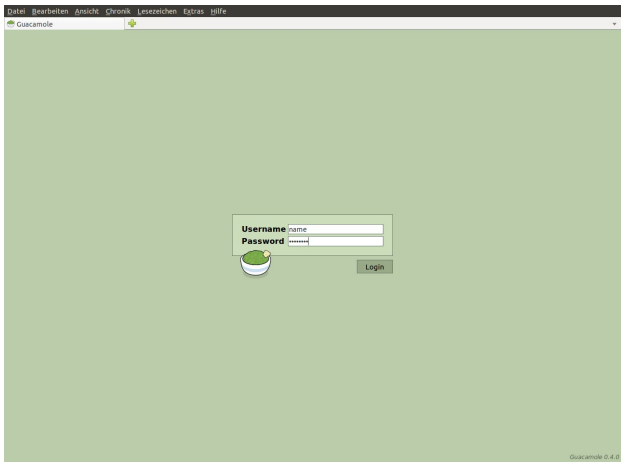

Abbildung 3: Der Anmeldebildschirm von Guacamole.

Nach erfolgreicher Anmeldung zeigt der Browser im günstigen Fall die Ansicht, die bereits aus dem VNC-Client bekannt ist.

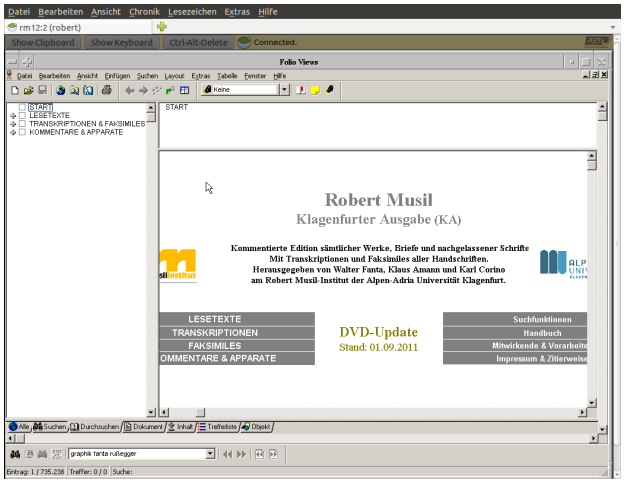

Abbildung 4: Die über den VNC-Server gestarteten Anwendungen im Fenster des Browsers.

Durch entsprechendes Editieren des Eintrags

```
<param name="hostname">localhost</param>
```

in der Datei „user-mapping.xml“ kann erreicht werden, dass die Tomcat-Umgebung und der VNC-Server nicht am selben Rechner installiert und betrieben werden müssen. Sicherzustellen ist dann jedoch, dass die Anwendung Guacamole den VNC-Server am entsprechenden Port erreichen kann.

Bei der Installation von Guacamole werden alle Dateien im Verzeichnis „/etc/guacamole“ mit dem Owner und der Gruppenzugehörigkeit von „root“ angelegt und die Dateirechte so vergeben, dass der Guacamole-Prozess, der mit den Rechten des Users „tomcat6“ läuft, die Dateien lesen kann. Diese Konfiguration bedingt jedoch auch, dass eventuell angemeldete Benutzer diese Dateien lesen und aus „user-mapping.xml“ die Anmeldedaten sowohl für Guacamole als auch für den VNC-Server entnehmen können. Daher sollten die Rechte sowie die Zugehörigkeit zumindest dieser Datei verändert werden.

```
sudo chown tomcat6.tomcat6 user-mapping.xml
sudo chmod 640 user-mapping.xml
```

Entscheidet man sich für den Einsatz von Remote-Applikationen, bleiben im Grunde dann noch einige Fragen zu klären, denn der eigentliche Prozess der Anwendung läuft auf einem entfernten System. Für das Drucken können die Netzwerkfähigkeiten von CUPS herangezogen werden, findet ein Datenexport oder -import in den einzelnen Anwendungen statt, helfen eingerichtete Freigaben von Verzeichnissen im (lokalen) Netz, die sowohl dem lokalen als auch dem entfernten System zugänglich sind. Spezielle Eingabegeräte wie Lesegeräte von Barcodes sind häufig in der Lage, ihre Datenausgabe über die Tastatur abzuwickeln und können daher auch in solchen Umgebungen eingesetzt werden. Die Integration weiterer Geräte erfordert, wenn diese allein technisch überhaupt gelingen kann, zumindest einen erhöhten Aufwand in der Planung.

Als einfache Hilfsmittel zur Dateneinn- und ausgabe bringt Guacamole zudem eine Bildschirmtastatur sowie ein eigenes Clipboard mit. Diese können mit unter hilfreich sein, wenn die zugehörigen Ausgabefunktionen des VNC-Servers Probleme bereiten.

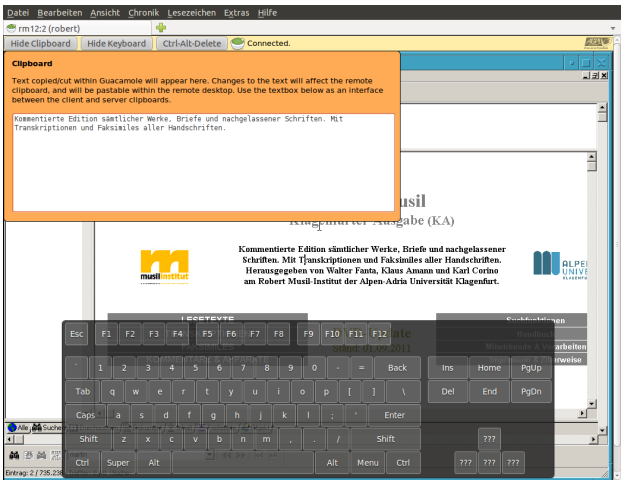

Abbildung 6: Das eingeblendete Clipboard sowie die Bildschirmtastatur von Guacamole.

Fazit

Guacamole erweist sich im täglichen Einsatz als ein sehr interessantes Projekt aus dem Umfeld der Open Source-Entwicklung, das bereits in einer frühen Version Potential zeigt, die Administration ganzer Desktop-Umgebungen und die Integration von Anwendungen in ein Fenster eines Browsers zu verlagern.
Die Vorzüge liegen in der einfachen Installation und Konfiguration sowie in der guten Ausnutzung vorhandener Übertragungskanäle. Eine Komprimierung der zu übertragenden Daten wäre vor allem dann ein großer Vorteil, wenn die Anwendung zur Fernwartung eingesetzt wird und vollständige Desktops angezeigt werden müssen. Eine Dekomprimierung im Browser sollte mit den modernen Verfahren, Code im Browser auszuführen, durchaus realistisch erscheinen.

Quellen

[1] Guacamole: HTML5 Clientless Remote Desktop: [http://guac-dev.org/]

[2] Das Protokoll „Remote framebuffer“ für VNC-Verbindungen: [http://datatracker.ietf.org/doc/draft-levine-rfb/]

[3] NoMachine NX – Desktop Virtualization and Remote Access Management: [http://nomachine.com]

[4] x2go – Open source terminal server project: [http://x2go.org]

Der Autor

Dr. Harald Jele ist Mitarbeiter an der Universität Klagenfurt und beschäftigt sich zur Zeit (auch) mit den technischen Aspekten digitaler Werkeditionen.