Administration virtualisierter Infrastrukturen mit Archipel

Inselverwaltung

von Thorsten Scherf

Virtuelle Maschinen-Instanzen laufen heutzutage auf einer Vielzahl unterschiedlicher Hypervisor-Systeme. Ein einheitliches Management-Tool ist notwendig, um den Zoo an virtuellen Systemen und deren Hosts im Überblick zu behalten und entsprechend zu verwalten. Mit Archipel steht nun ein recht neues Tool dieser Gattung zur Verfügung, das durch seine administrationsfreundliche Oberfläche besticht. Dieser Workshop stellt die Fähigkeiten des Management-Werkzeugs vor und zeigt dessen Inbetriebnahme.

Neben dem Platzhirschen VMware hat auch das Open Source-Umfeld viele sehr elegante und leistungsstarke Virtualisierungswerkzeuge zu Tage gefördert. Die beiden Zugpferde sind hier sicherlich KVM (Kernel-based Virtual Machine) und XEN. Daneben existieren jedoch weitere Lösungen wie beispielsweise VirtualBox, OpenVZ oder auch Linux-Container (LXC). Setzt ein Unternehmen nun unterschiedliche Technologien ein, bedeutet dies im Umkehrschluss auch den Einsatz unterschiedlicher Management-Tools. Dass dies jedoch nicht zwangsläufig der Fall sein muss, ist spätestens seit Erscheinen des Virtualisierung-Frameworks libvirt bekannt. Das Framework bringt diverse Treiber für den Zugriff auf die unterschiedlichsten Virtualisierungstechnologien von Hause aus mit [1] und gestattet somit einen einheitlichen Zugriff auf virtuelle Maschinen unabhängig davon, auf welchem Hypervisor diese laufen.

Als Management-Tools stehen für libvirt auf der Kommandozeile *virsh* und *virt-install* zum Verwalten und Erzeugen von virtuellen Maschinen-Instanzen zur Verfügung und mit virt-manager existiert ein grafisches Frontend. Die Tools müssen dabei nicht zwingend auf dem jeweiligen Hypervisor installiert sein und ein Remote-Zugriff ist ohne weiteres möglich. Um die Daten zwischen der Workstation und dem Hypervisor zu übertragen, existieren diverse Möglichkeiten: So lässt sich beispielsweise der IP-Traffic sowohl über eine ungesicherte wie auch mit TLS geschützte TCP-Verbindung übertragen. Auch der Einsatz eines SSH-Tunnels ist von Haus aus denkbar. Eine Authentifizierung erfolgt wahlweise mittels eines passenden SASL-Plug-Ins, beispielsweise Kerberos oder aber auch mit Hilfe von X.509-Zertifikaten.

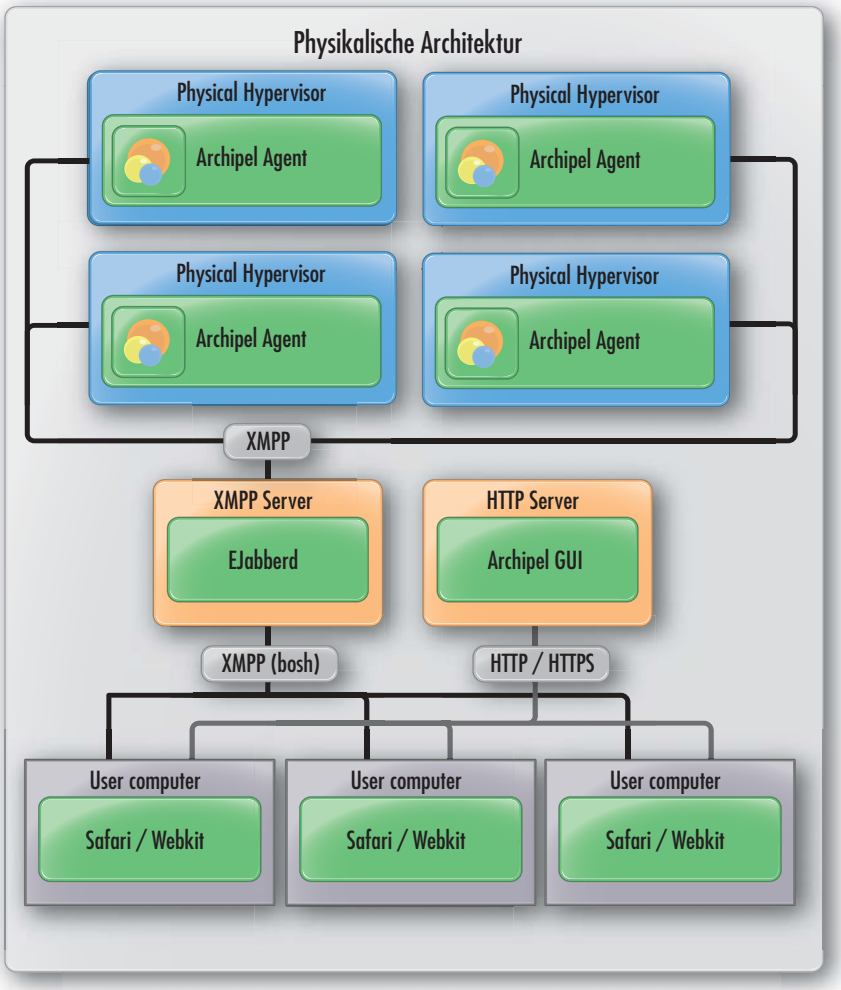

Bild 1: Bei Archipel kommen diverse OpenSource-Komponenten zum Einsatz. Die zentrale Rolle spielt dabei ein XMPP-Server, der für die Echtzeit-Kommunikation zwischen den einzelnen Komponenten verantwortlich ist.

Dank der großen Flexibilität von libvirt kommt es in diversen anderen Virtualisierungs-Projekten zum Einsatz, so beispielsweise in oVirt [2] oder aber auch in Archipel [3]. Archipel ersetzt dabei die etwas angestaubte GTK+-Oberfläche des virt-managers durch eine schicke und modere Cappuccino-basierte Oberfläche und fügt weitere Features, die beispielsweise virt-manager nicht besitzt, hinzu. Dank des JavaScript-basierten Frameworks, erfolgt der Zugriff auf Archipel über nahezu jeden beliebigen Webbrowser – eine zusätzliche Anwendung oder eine lokale libvirt-Installation sind dabei nicht notwendig.

XMPP-Server als zentrale Architektur-Komponente

Archipel besteht aus mehreren Komponenten die untereinander über das bekannte XMPP-Protokoll (ehemals Jabber) kommunizieren. Die Zustellung von Nachrichten findet somit nahezu in Echtzeit statt. Zentrale Komponente der gesamten Architekur ist ein XMPP-Server der als eine Art Nachrichtenzentrale agiert. Die Archipel-Entwickler empfehlen den Einsatz des ejabberd-Servers, jedoch sollte auch jeder andere XMPP-Server funktionieren. Die ejabberd-Software ist Bestandteil sämtlicher Linux-Distributionen und lässt sich somit recht einfach über den jeweiligen Paket-Manager installieren.

Über die Download-Seite von Archipel [4] stehen dann die beiden Komponenten Archipel-Agent und Archipel-Client zur Verfügung. Beim Agent handel es sich um die Komponente, die auf jedem Hypervisor-System zu installieren ist und als eine Art Bindeglied zwischen dem XMPP-Server und dem libvirt-Framework des Hypervisors dient. Der Agent selbst kommuniziert gar nicht mit dem Hypervisor sondern überlässt diese Aufgabe libvirt. Somit ist ein Zugriff auf sämtliche Hypervisoren möglich, die libvirt unterstützt. Der Archipel-Client ist dabei die Komponente, die – meistens zusammen mit dem XMPP-Server – auf einem zentralen Management-Server läuft und die jedem zugreifenden Client eine hübsche Weboberfläche präsentiert. Der komplette Client ist dabei in Javascript geschrieben und basiert auf dem Strophe.js-Framework. Archipel erzeugt zur Weiterverarbeitung Cappuchino-Objekte. Diese lassen sich dann mit nahezu jedem beliebigen Javascript fähigen Webbrower abrufen. Bild 1 zeigt die Architektur des Archipel-Frameworks.

```
cd /usr/local/src
wget http://ejabberd.jabber.ru/files/
   contributions/xmlrpc-1.13-ipr2.tgz
tar -xzf xmlrpc-1.13-ipr2.tgz
cd xmlrpc-1.13/src
make
cp - a ebin/*.beam /usr/lib/ejabberd/ebin/
```

Listing 1: Installation des ejabberd_xmlrpc Moduls

```
cd /usr/local/src
svn checkout http://svn.process-one.net/
   ejabberd-modules/
cd mod_admin_extra/trunk/
./build.sh
cp ebin/mod_admin_extra.beam
   /usr/lib/ejabberd/ebin/
```

Listing 2: Installation des ejabberd_xmlrpc Moduls

Installation

Kommt der ejabberd-Server als XMPP-Komponente zum Einsatz, so ist darauf zu achten, dass dieser die beiden Module "mod_admin_extra" und "ejabberd_xmlrpc" enthält. Ist dies nicht der Fall, müssen Sie diese manuell nachliefern. Hierzu verwenden Sie die Sourcen der ejabberd-Module und das Erlang-XMLRPC-Archiv und übersetzen diese. Unter Umständen sind noch die Dateisystem-Pfade im Makefile anzupassen, dies hängt von der eingesetzten Linux-Distribution ab. Die resultierenden beam-Dateien gehören dann in das ebin-Verzeichnis des ejabberd-Servers (Listing 1 und Listing 2).

Hat dies soweit funktioniert, müssen Sie schließlich noch die Konfigurationsdatei des XMPP-Servers anpassen. Hierbei handelt es sich um die Datei */etc/ejabberd/ejabberd.cfg*. Wichtig ist hierbei, dass Sie den lokalen Rechnernamen und die zusätzlich installierten ejabberd-Module korrekt angeben. Listing 3 zeigt eine beispielhafte Konfiguration (die ejabberd Default-Konfiguration wollte in unserem Praxis-Test nicht mit Archipel zusammenarbeiten).

Der Aufruf von */etc/init.d/ejabberd start* aktiviert schließlich den Server. Sollten Fehler auftreten, so finden Sie die entsprechenden Logeinträge in der Datei */var/log/ejabberd/*

```
# /etc/ejabberd/ejabberd.cfg
{loglevel, 4}.
{hosts, ["rawhide.tuxgeek.de","tuxgeek.de",
   "localhost"]}.
{listen,
 [
  {5222, ejabberd_c2s, [
           {access, c2s},
           {shaper, c2s_shaper},
           {max_stanza_size, 65536}
            ]},
  {5269, ejabberd_s2s_in, [
             {shaper, s2s_shaper},
             {max_stanza_size, 131072}
            ]},
  {5280, ejabberd_http, [
            captcha,
            http_bind,
            http_poll,
            web_admin
           ]}
 ]}.
{auth_method, internal}.
{shaper, normal, {maxrate, 1000}}.
{shaper, fast, {maxrate, 50000}}.
{max_fsm_queue, 1000}.
{acl, local, {user_regexp, ""}}.
{access, max_user_sessions, [{10, all}]}.
{access, max_user_offline_messages, [{5000,
   admin}, {100, all}]}.
{access, local, [{allow, local}]}.
{access, c2s, [{deny, blocked},
          {allow, all}]}.
{access, c2s_shaper, [{none, admin},
            {normal, all}]}.
{access, s2s_shaper, [{fast, all}]}.
{access, announce, [{allow, admin}]}.
{access, configure, [{allow, admin}]}.
{access, muc_admin, [{allow, admin}]}.
{access, muc_create, [{allow, local}]}.
{access, muc, [{allow, all}]}.
{access, pubsub_createnode, [{allow, local}]}.
{access, register, [{allow, all}]}.
{language, "en"}.
{modules,
 [
  {mod_adhoc,    []},
  {mod_caps,     []},
  {mod_disco,    []},
  {mod_irc,      []},
  {mod_http_bind, []},
  {mod_last,     []},
  {mod_muc,      [
          {access, muc},
          {access_create, muc_create},
          {access_persistent, muc_create},
          {access_admin, muc_admin}
         ]},
  {mod_offline,  [{access_max_user_messages,
   max_user_offline_messages}]},
  {mod_ping,     []},
  {mod_privacy,  []},
  {mod_private,  []},
  {mod_pubsub,   [
          {access_createnode, pubsub_createnode},
          {last_item_cache, false},
         ]},
  {mod_register, [
          {welcome_message, {"Welcome!",
                     "Hi.\nWelcome to this XMPP
   server."}},
          {ip_access, [{allow, "127.0.0.0/8"},
                    {deny, "0.0.0.0/0"}]},
          {access, register}
         ]},
  {mod_roster,   []},
  {mod_shared_roster,[]},
  {mod_stats,    []},
  {mod_time,     []},
  {mod_vcard,    []},
  {mod_version,  []}
 ]}.
```

Listing 3: Die Konfigurationsdatei des XMPP-Servers

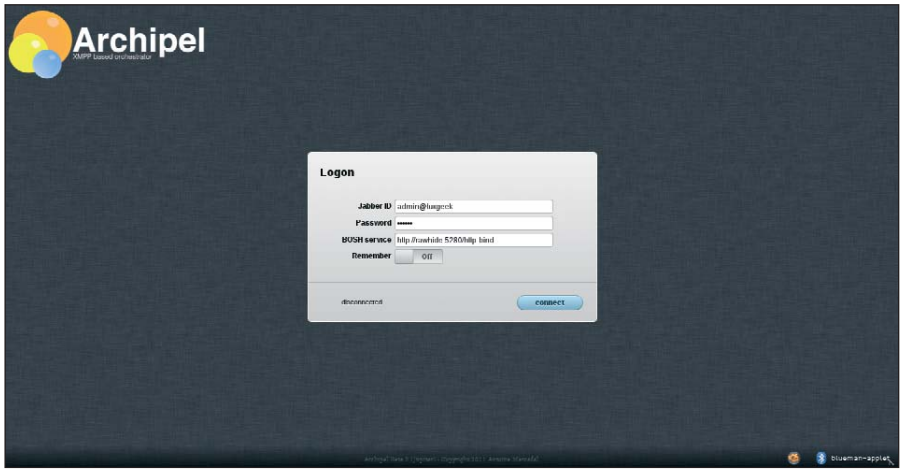

Bild 2: Nach der Installation des Management-Systems steht dieses für erste Zugriffe bereit

ejabberd.log. Um die Installation des Servers abzuschließen legen Sie noch ein entsprechendes admin-Konto an:

```
# ejabberdctl register admin
  {rawhide.tuxgeek.de} {password}
```

Den Rechnername und das Passwort passen Sie dabei natürlich entsprechend an.

Die Archipel-Client-Installation gestaltet sich um einiges einfacher. Hier laden Sie lediglich die gewünschte Version [4] herunter und entpacken diese in ein beliebiges Verzeichnis. Wichtig ist, dass der Webserver Zugriff auf dieses Verzeichnis erhält. Sollte auf dem Management-System ein aktiviertes SELinux zum Einsatz kommen, so ist nach dem Entpacken des Client-Archives zwingend der richtige SELinux Context auf den Ordner zu setzen, ansonsten kann der Webserver nicht auf die Dateien zugreifen und erzeugt eine Vielzahl von AVC-Deny-Meldungen. Den richtigen SELinux Context setzen Sie wie folgt:

```
# chcon -R -t public_content_r
  /var/www/html/Archipel/
```

Hiermit ist die Installation des Management-Servers beendet. Ein Zugriff mittels eines Webbrowsers sollte zu diesem Zeitpunkt bereits möglich sein. Hierzu bauen Sie einfach eine HTTP-Verbindung zum Management-Server auf. Das Archipel-Client-Installationsverzeichnis geben Sie dabei in der URL an – also beispielsweise *http://rawhide.tuxgeek.de/Archipel/* für eine Installation des Clients auf dem Rechner "rawhide.tuxgeek.de" im Verzeichnis "/var/www/html/". Das DocumentRoot des Webservers kann hier natürlich auch wieder zwischen den einzelnen Linux-Distributionen variieren. Alle in diesem Artikel vorgestellten Beispiele basieren auf einer Fedora 15-Installation. Zur Anmeldung am Archipel-Management-System geben Sie den oben erzeugten Admin-Account an. Wichtig dabei ist, dass Sie den Account mit Domänenname verwenden, also beispielsweise admin@tuxgeek.de. Als Boch-Server nutzen Sie die URL *http://Management-Server:5280/http-bind/*.

Einbindung der Hypervisoren

Ein Log-In in das System ist zu diesem Zeitpunkt jedoch noch nicht sehr hilfreich, da natürlich noch die Hypervisor-Systeme fehlen. Ohne diese Systeme lassen sich natürlich auch keine virtuellen Maschinen erzeugen. Für den Betrieb eines Hypervisors kommt üblicher weise ebenfalls ein Linux mit libvirt-Installation zum Einsatz. Der gewünschte Hypervisor ist dann entsprechend hinzuzufügen, also beispielsweise XEN, KVM, QEMU oder LXC. Die einzelnen Abhängigkeiten sollte der Paket-Manager der eingesetzten Linux-Distribution selbstständig auflösen. So benötigt ein KVM beispielsweise ein QEMU als Hardware-Emulator sowie einige weitere Python-Tools.

Auf diesem System ist nun der Archipel-Agent als eine Art Proxy zwischen dem XMPP-Server und der lokalen libvirt-Installation zu installieren. Auch hier geht die Installation recht einfach vonstatten: Je nach Vorliebe und Experimentierfreude nutzen Sie entweder den letzten NightlyBuild oder den letzten StableBuild [4]. Noch einfacher gelingt die Installation über den zentralen Python Package Index (PyPi) – dies setzt jedoch eine Internet-Verbindung und das Paket python-setuptools auf den Hypervisor-Systemen voraus. Über den Aufruf von *easy_install archipel-agent* installieren Sie dann den Archipel-Agent auf dem lokalen System.

In der Konfigurationsdatei */etc/archipel/archipel.conf* müssen Sie wieder darauf achten, dass der richtige Rechnername für den lokalen Hypervisor und den zentralen XMPP-Server zum Einsatz kommen. Auch das zuvor eingerichtete admin-Konto des XMPP-Servers geben Sie in der Datei an. Für das Berechtigungs- und Tagging-System auf dem XMPP-Server richten Sie noch zwei PubSub-Nodes ein. Dies geschieht auf dem Hypervisor-Systemen durch den Aufruf von:

```
# archipel-tagnode –jid=admin@xmpp-
  server –password=password –create
# archipel-rolesnode
  –jid=admin@xmpp-server
  –password=password –create
```

Ein */etc/init.d/archipel* startet schließlich den Agent-Dienst. Mögliche Fehlermeldungen finden Sie der Logdatei */var/log/archipel/archipel.log*.

Hypervisor-Management

Nachdem die Installation sämtlicher Archipel-Komponenten abgeschlossen ist, besteht die nächste Aufgabe darin, die Hypervisor-Systeme auf dem Management-Node bekannt zu machen. Da bei Archipel alles ein XMPP-Objekt (oder Jabber- Objekt) ist, sind natürlich auch die Hypervisoren so zu behandeln, indem diese als reguläre XMPP-Kontakte dem Manage-

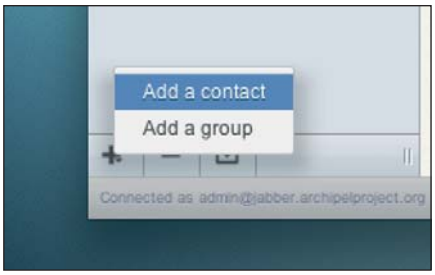

Bild 3: Hypervisor-Systeme werden in Archipel als reguläre XMPP-Kontakte behandelt

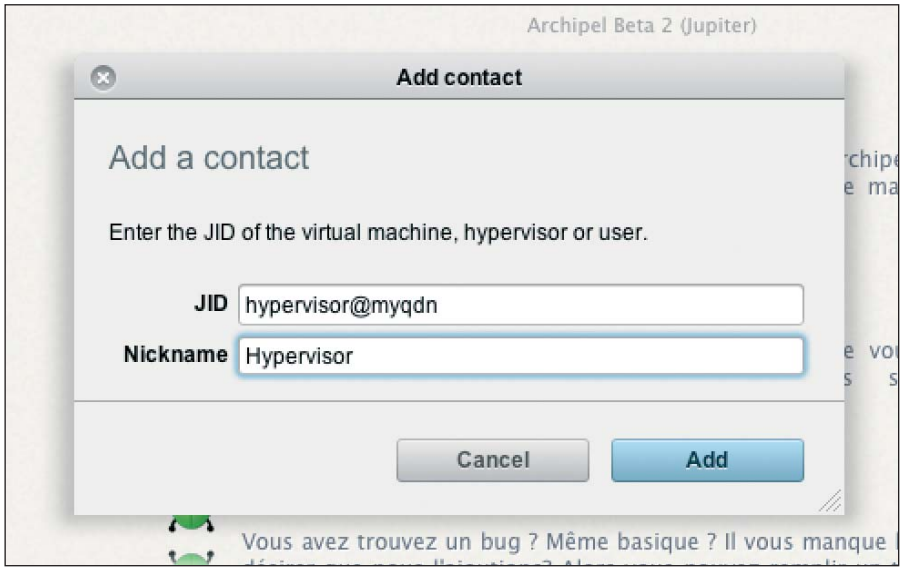

Bild 4: Als JIT-Name ist zwingend der String "Hypervisor" gefolgt vom Domänennamen zu verwenden

ment-System hinzugefügt werden. Nach dem Login klicken Sie dazu in der linken unteren Ecke auf das "+"-Zeichen um den ersten Hypervisor hinzuzufügen. Als JIT-Name verwenden Sie hier Hypervisor@hypervisor-fqdn (also etwa hypervisor@hv01.tuxgeek.de), zusätzlich ist die Angabe eines Alias-Namens möglich. Durch einen Klick auf das Symbol "New VM" in der Icon-Leiste erzeugen Sie eine neue virtuelle Maschine. Hierzu definieren Sie alle notwendigen Eigenschaften entsprechend, also beispielsweise die Größe der virtuellen Disk, der Arbeitsspeicher, Boot-Medium und Netzwerkkonfiguration. Libvirt stellt standardmäßig ein NAT-Gerät zur Konfiguration zur Verfügung, aber natürlich besteht auch die Möglichkeit, eine Bridge einzurichten, sodass Sie die virtuellen Gäste direkt erreichen können.

Zur eigentlichen Installation des Betriebssystems kommt üblicher Weise eine virtuelle CD zum Einsatz, oft in Form einer Image-Datei. Diese sucht Archipel im Verzeichnis "/vm/iso/". Das Verzeichnis "/vm" sollte deshalb allen Hypervisor-Systemen zur Verfügung stehen. Dies ist auch deshalb notwendig, da im Verzeichnis "/vm/drives/" die Speicher-Backends der virtuellen Systeme liegen, bei einer Live-Migration zwischen zwei Hypervisor-Systemen müssen diese natürlich Zugriff hierauf haben.

Sollten die hinzugefügten Hypervisor-Systeme bereits über virtuelle System-Instanzen verfügen, so erkennt Archipel diese leider nicht automatisch. Glücklicherweise bringt es aber ein Import-Tool für diese Aufgabe mit. Dieses benötigt zum einen die Deskriptor-Datei der SQLite-Datenbank des Hypervisors (üblicher weise */var/ lib/archipel/hypervisor.sqlite3*) und zum anderen die UUIDs der zu importierenden virtuellen Systeme. Diese ermitteln Sie durch eine einfache libvirt-Abfrage. Mittels *virsh dumpxml {VMName}* präsentiert libvirt sämtliche Konfigurationsdaten der angegebenen VM. Da an dieser Stelle jedoch nur die uuid interessant ist, filtert grep diese entsprechend heraus:

```
# virsh dumpxml webserver01|grep -i
  uuid
<uuid>7f0bbaae-8ac2-11e0-be38-
  00216ab8187e</uuid>
```

Nach einem Stop des Archipel-Dienstes lässt sich die gewünschte VM nun in die Archipel-Datenbank importieren und taucht danach, neben dem Hypervisor, als regulärer XMPP-Kontakt in der Archipel-Weboberfläche auf:

```
# /etc/init.d/archipel stop
* Stopping Archipel: [OK]
# archipel-importvirtualmachine
  –file /var/lib/archipel/hypervi-
  sor.sqlite3 \
 –uuid 7f0bbaae-8ac2-11e0-be38-
  00216ab8187e –
  xmppserver=rawhide.tuxgeek.de \
 –name webserver01
SUCCESS: Virtual machine webserver01
  has been inserted with JID \
 7f0bbaae-8ac2-11e0-be38-
  00216ab8187e@rawhide.tuxgeek.de
# /etc/init.d/archipel start
* Starting Archipel: [OK]
```

Da als Echtzeit-Protokoll zwischen allen beteiligten Archipel-Objekten das XMPP-Protokoll zum Einsatz kommt (Bild 6), lässt sich jeder XMPP-fähige IM-Client zur Kommunikation mit den einzelnen Objekten verwenden. Bild 7 zeigt ein Beispiel mit einem konfigurierten Empathy-Client, der hier einfache Status-Abfragen an die VM "webserver01" sendet. Das Tool vnc-viewer stellt schließlich eine Verbindung zum VNC-Server des Hypervisors her, um so einen grafischen Zugriff auf die virtuelle Maschine zu bekommen. Beide Funktionen, also VNC-Viewer und Chat-Client, sind jedoch auch im Archipel-Client selbst integriert. Besonders der Javascript-ba-

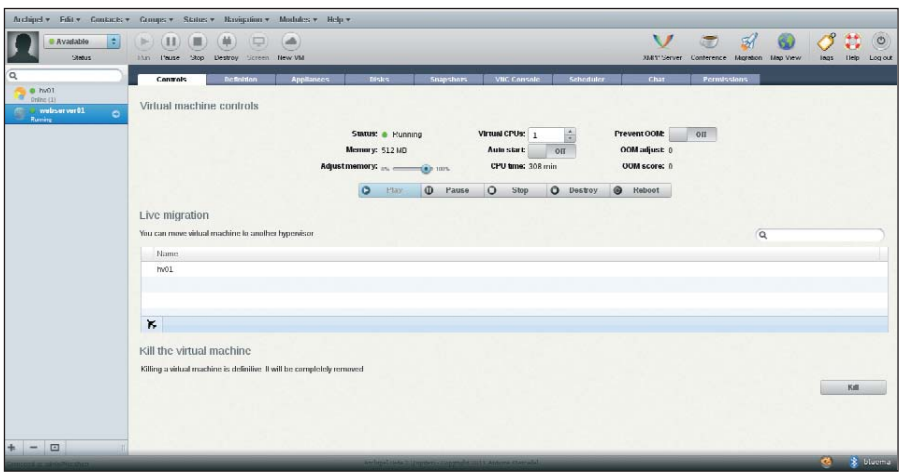

Bild 5: Nach erfolgreichem Import der virtuellen Maschine lassen sich sämtliche Eigenschaften nun über Archipel definieren

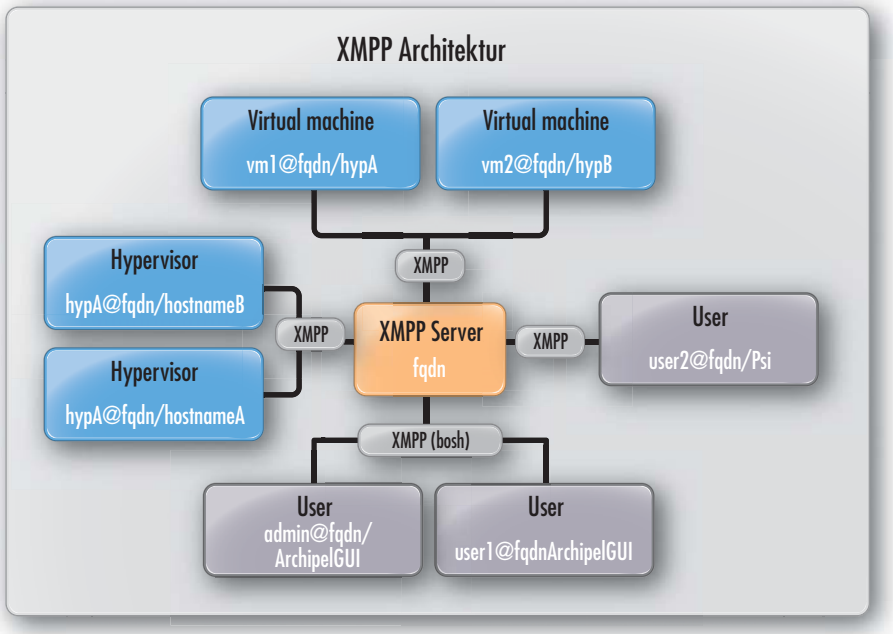

Bild 6: Alle Archipel-Komponenten kommunizieren über das Protokoll XMPP miteinander...

sierte VNC-Viewer innerhalb des Webbrowsers sieht sehr schick aus.

Der integrierte IM-Client ist dabei nicht nur in der Lage, mit dem internen XMPP-Server zu sprechen, sondern unterstützt auch externe Server. Dies kann recht hilfreich sein, falls Sie einmal einen Ratschlag der Archipel-Entwickler benötigen, die immer hilfsbereit Fragen auf dem IRC Freennode Kanal #archipel beantworten.

Zusätzliche Features

Neben dieser netten Spielerei bietet Archipel natürlich auch Funktionen wie das Clustering der Umgebung. Leider bezieht sich dieser Punkt bisher lediglich auf den XMPP-Server, die Hypervisor-Systeme bilden somit aktuell noch einen Single-Point-of-Failure, der sich aber durch den Einsatz weiterer Tools, wie beispielsweise der Red Hat Cluster Suite, beheben lässt. VMCasts erlauben es dem Administrator, vorgefertigte Appliances zu erstellen und zu konfigurieren. Diese stellt Archipel dann über RSS-Feeds den Hypervisoren zum Download zur Verfügung. Dies, und auch der Einsatz von Templates, beschleunigt das Bereitstellen von virtuellen Maschinen enorm.

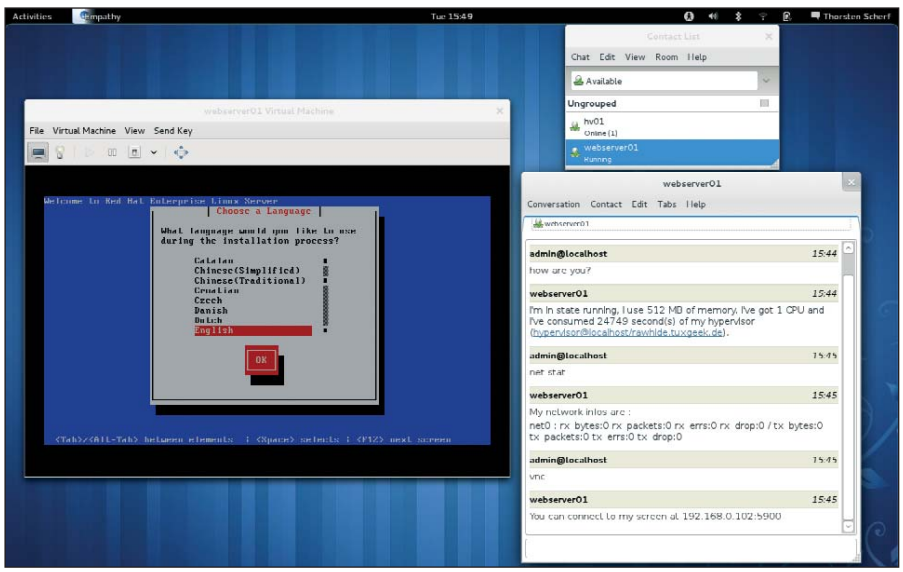

Bild 7: ...dies ermöglicht die Kommunikation mit den einzelnen Komponenten sowie mit einem regulären XMPP-/Jabber-Client, wie beispielsweise Empathy oder Pidgin

Über eine eingebaute Google Maps-Karte lassen sich die Archipel-Objekte geographisch positionieren. Eine Migration von Objekten soll dann auch über das Verschieben auf der Landkarte möglich sein, in der aktuellen Version hat diese Funktion aber leider noch nicht funktioniert. Ein weiteres nützliches Feature von Archipel ist die Möglichkeit, ein einzelnes oder auch mehrere Kommandos an eine Vielzahl von Systemen abzusetzen. Diese lassen sich nämlich in bestimmte Gruppen eingliedern, etwa in die Gruppe Webserver. Kommandos, wie beispielsweise das Starten, Stoppen oder eine Migration, lassen dann auf sämtliche Objekte dieser Gruppe anwenden.

Fazit

Abschließend lässt sich festhalten, dass Archipel ein interessantes Tool ist, wenn es um die Verwaltung von vielen virtuellen Maschinen und deren Hypervisor-Systemen geht. Die schicke und ansprechende Weboberfläche bietet keines der anderen Tools, die in dieser Liga spielen. Auf der anderen Seite hat Archipel noch einige technische Mängel und bestimmte Funktionen sind noch nicht implementiert. Auch fehlende Software-Pakete und die mangelhafte Dokumentation machen die Installation und den Betrieb des Frameworks nicht unbedingt einfacher. Schließlich muss sich der verantwortliche Administrator fragen, ob die genannten Features einen zusätzlichen Komplexitätslevel durch den Einsatz von Archipel rechtfertigen. Die meisten grundlegenden Funktionen, die Archipel bietet, sind bereits von Tools wie virt-manager [5] bekannt. *(jp)*

Link-Codes

[1] libvirt-Projektseite
B8P41

[2] oVirt-Projektseite
B8P42

[3] Archipel-Projektseite
B8P43

[4] Archipel-Download
B8P44

[5] virt-manager-Projektseite
B8P45