Universität Rostock
Institut für Informatik
Studiengang: Technische Informatik (Master of Science)

Projektarbeit

Werkzeugkasten für PlanetLab

vorgelegt von:
Nico Neuwardt,
geboren am 16.09.1977
Matrikelnummer: 6256344

Betreuer:
Dipl.-Inf. Martin Garbe

Inhaltsverzeichnis

1 Einleitung

PlanetLab ist ein globales Netzwerk, welches hauptsächlich als Testumgebung für neue Netzwerkprotokolle und Netzwerkdienste konzipiert ist. Es besteht zum Zeitpunkt der Abgabe dieser Arbeit aus 1129 Nodes an 544 weltweit verteilten Orten [7].

Nutzer von PlanetLab können ein eigenes Overlay Netz innerhalb des PlanetLab Netzes anfordern. Diesem Overlay Netz können beliebig viele Nodes aus dem PlanetLab Verbund angehören. Hierzu wird das Konzept der verteilten Virtualisierung angewandt. Dem Nutzer wird eine sogenannte Slice (siehe Abschnitt 2.3) zur Verfügung gestellt, mit dem er unabhängig von anderen Slices innerhalb des PlanetLab Netzes arbeiten kann.

Um im PlanetLab Experimente durchführen zu können, ist es nötig auf die einzelnen virtuellen Komponenten - den Slivers (siehe Abschnitt 2.2) - zugreifen zu können. Im einfachsten Fall geschieht dies über eine Secure Shell (ssh). Für umfangreiche Experimente ist dies jedoch keine effiziente Lösung. Es stehen jedoch eine Reihe von Tools bereit, mit denen sich auch umfangreiche Experimente innerhalb des PlanetLab Netzes komfortabel steuern lassen.

Beim Einstieg in die Arbeit steht ein neuer Nutzer damit vor zwei wichtigen Problemen. Zum einen muss die ssh Konfiguration auf dem Steuerrechner bestimmten Anforderungen genügen, um auf die Sliver der PlanetLab Slice zugreifen zu können. Zum anderen ist es für ein umfangreiches Experiment nötig, sich für eines der zur Verfügung stehenden Tools zu entscheiden und dieses zu installieren und zu konfigurieren.

Ziel dieser Projektarbeit soll es sein, mehrere der vorhanden Tools zu vergleichen und eine Möglichkeit zum schnellen Einstieg in die Arbeit mit PlanetLab zu ermöglichen. Hierzu wurde eine Virtuelle Maschine mit einem installierten und fertig konfigurierten Betriebssystem erstellt, mit der der Zugriff auf die eigene PlanetLab Slice nach wenigen persönlichen Anpassungsschritten möglich ist. Zudem beinhaltet diese VM eines der verglichenen Tools für die Durchführung von größeren Experimenten, dessen Konfiguration ebenfalls nur noch um einige Nutzerdetails ergänzt werden muss.

Damit sollte es einem neuen PlanetLab Nutzer möglich sein innerhalb von kurzer Zeit die ersten PlanetLab Experimente durchzuführen.

In dieser Ausarbeitung sollen zunächst die technischen Grundlagen von PlanetLab kurz beleuchtet werden. Insbesondere die PlanetLab spezifischen Begrifflichkeiten sollen erklärt werden. Danach werden einige der vorhandenen Tools kurz vorgestellt.

Anschließend wird eine Einordnung der Tools in ein Kategorisierungsschema vorgenommen. Zum Abschuss dieser Ausarbeitung wird die erstellte Virtuelle Maschine kurz vorgestellt.

2 Technische Grundlagen

Das PlanetLab Netz ist aus einer Reihe von physikalischen und virtuellen Komponenten aufgebaut [5]. Aus Nutzersicht sind die wichtigsten davon die Nodes, Slices und Sliver, welche im Folgenden kurz erklärt werden.

2.1 Node

Eine Node ist ein physikalischer Server mit eigener dedizierter IP Adresse. Jede der am PlanetLab Netz beteiligten Organisationen sollte wenigstens zwei solcher Server für die exklusive Nutzung als PlanetLab Node bereitstellen [1]. Auf einer Node können mehrere virtuelle Maschinen erstellt werden, die dann als Sliver (siehe Abschnitt 2.2) zur Verfügung stehen. Um die Erstellung und die Verwaltung dieser virtuellen Maschinen kümmert sich ein auf der Node laufendes Programm, der Node Manager [8].

2.2 Sliver

Eine einzelne auf dem Node Server laufende virtuelle Maschine, welche dem PlanetLab Nutzer zur Verfügung gestellt wird, wird als Sliver bezeichnet [6]. Die einzelnen Sliver sind voneinander isoliert. Es ist nicht möglich von einem Sliver aus auf Ressourcen eines anderen Slivers (wie beispielsweise Dateien, Prozesse oder Ports) zuzugreifen. Auch ein Zugriff auf den Netzwerk-Traffic eines anderen Slivers ist nicht möglich.

2.3 Slice

Ein Verbund von Slivern, welcher über mehrere Nodes innerhalb des PlanetLab Netzes verteilt ist, wird als Slice bezeichnet. Für den Nutzer ist eine Slice ein Overlay Netz innerhalb des PlanetLab Netzes, welches er für seine Experimente benutzen kann. Eine Slice kann von einem Principal Investigator (PI) [2] einer PlanetLab Site erstellt werden. Der PI weist dieser Slice auch ihre Nutzer zu. Ein PlanetLab

Nutzer ist dazu autorisiert, der Slice Nodes hinzufügen. Wird eine Node zur Slice hinzugefügt, wird dort ein Sliver (siehe Abschnitt 2.2) erstellt.

3 Übersicht über vorhandene Tools

Damit der Nutzer seine Slice nutzen kann, reicht es im einfachsten Fall aus, mittels Secure Shell (ssh) auf die einzelnen Sliver zuzugreifen. Bei Experimenten, die auf einer Vielzahl von Knoten ausgeführt werden, ist dieses Verfahren jedoch nicht effizient.

Für die Arbeit mit der Slice stehen einige Tools zur Verfügung. Mit diesen lassen sich sowohl einzelne Verwaltungsaufgaben (wie z.B. die Softwareverteilung) als auch komplette Experimente wesentlich komfortabler durchführen. Im Folgenden werden einige dieser Werkzeuge vorgestellt.

3.1 Stork

Stork [12] ist eine Entwicklung der University of Arizona. Es ist ein Toolkit, welches zur Softwareverteilung im PlanetLab Netzwerk dient. Als solches stellt es ein Ersatz für die klassischen Distributionstools, wie z.B. yum oder apt dar.

Bei den klassischen Distributionstools kommt es innerhalb des PlanetLab Netzes zu dem Problem der doppelten Datenhaltung. Auf den einzelnen PlanetLab Nodes wird für jede Slice eine eigene virtuelle Maschine bereitgestellt (Sliver). Ein Software-Paket, welches auf einem Sliver mittels apt oder yum installiert wird, steht nur innerhalb des Dateisystems der VM zur Verfügung. Wenn mehrere Sliver der gleichen Node dieselben Software-Pakete benötigen, müssen sie so in jedem einzelnen Sliver installiert werden.

Stork behebt dieses Problem, indem auf den Nodes eine eigene virtuelle Maschine zur Datenhaltung bereitgestellt wird. In der Stork Terminologie wird diese spezielle VM mit Nest betitelt. Das Nest ist in der Lage, Dateien für die Slivers der Node bereitzustellen, ohne dass diese Dateien in das Dateisystem der jeweiligen Sliver-VM kopiert werden müssen. Wenn ein Sliver eine Paketinstallation mittels Stork verlangt, wird erst überprüft ob das Paket bereits im Nest vorhanden ist. Ist dies der Fall, muss das Paket nicht erneut heruntergeladen und im Dateisystem des Sliver installiert werden. Das Nest stellt dann lediglich den Zugriff auf die bestehenden Dateien zur Verfügung.

Zum Stork Projekt gehört der Stork Slice Manager. Das ist eine GUI für Stork, welches die Installation von Stork und die Paketverwaltung in einer kompletten Slice vereinfacht.

3.2 CoDeploy

CoDeploy [10] wurde an der Princeton University entwickelt. Es dient ebenfalls der Verteilung von Dateien innerhalb des PlanetLab Netzwerkes, verfolgt jedoch einen anderen technischen Ansatz. Während Stork vor allem die mehrfache Datenhaltung auf den Nodes entgegenwirken will, soll CoDeploy dafür sorgen, dass der Netzwerktraffic während des Deployments gering gehalten wird.

Neben generellen Techniken zur Minimierung des Netzwerktraffics (wie z.B. Synchronisationstechniken um nur geänderte Dateibestandteile zu übertragen) nutzt CoDeploy vor allem das CoDeeN Netzwerk [9] um dieses Ziel zu erreichen. Dabei werden Proxy Server des CoDeeN Netzes mittels eines Peer Reviews ausgewählt und als Caching Server verwendet.

3.3 CoTop

CoDeploy [11] wurde ebenfalls im Rahmen des CoDeeN Projektes an der Princeton University entwickelt. Es ist ein einfaches Tool, welches nach dem Vorbild von top die Auslastung einer PlanetLab Node anzeigt. Dabei wird angegeben wie stark die einzelnen Slices der Node die Ressourcen der Node in Anspruch nehmen.

CoTop ist trotz des Namens und der Projektzugehörigkeit ein eigenständiges Tool, welches nicht auf das CoDeeN Netzwerk aufbaut. Es verwertet die Daten von slicestat, einem Sensor Daemon, welcher auf jeder PlanetLab Node installiert ist.

3.4 pssh

Die pssh Tools (Parallel ssh) [3] wurden im Intel Research Institute in Berkeley entwickelt. Es handelt sich hierbei nicht um eine PlanetLab-spezifische Toolsammlung. Pssh ist eine Sammlung von Tools, welche nach Vorbildern aus den openssh Tools aufgebaut sind. Im Unterschied zu ihren openssh Pendants, sind die pssh Tools dafür bestimmt, Operationen auf mehreren entfernten Maschinen gleichzeitig auszulösen.

Die pssh Tools kann man somit als parallele Versionen von bestimmten openssh Tools ansehen. Im Einzelnen sind das:

- Parallel ssh (pssh) zum Ausführen von Kommandozeilenbefehlen
- Parallel scp (pscp) zum Kopieren von Dateien
- Parallel rsync (prsync) zum Synchronisieren von Verzeichnissen
- Parallel nuke (pnuke) zum Beenden von Prozessen
- Parallel slurp (pslurp) zum Kopieren von gleichnamigen Dateien von mehreren Hosts zum lokalen Rechner

Bei allen Tools handelt es sich um Kommandozeilenwerkzeuge.

3.5 pShell

pShell [4] ist eine Entwicklung eines einzelnen Programmierers an der McGill University in Montreal. Ähnlich wie pssh stellt es verschiedene Funktionen für das Absetzen paralleler Kommandos an unterschiedliche Sliver bereit.

Im Gegensatz zu pssh ist pShell ein Tool, welches speziell für PlanetLab entwickelt wurde. Ein weiterer Unterschied ist, dass es eine eigenständige Shell bereitstellt, über die die Befehlseingabe erfolgt.

Auf dieser Shell stehen dann Kommandos bereit, über die sich z.B. Kommandos auf den Slivern ausführen lassen oder mit denen man Dateitransfers zu und von den Slivern durchführt. Ebenso lässt sich eine Liste von PlanetLab Nodes anzeigen und es werden Funktionen zur Slice-Verwaltung bereitgestellt.

3.6 Plush

Plush (Planet Lab User Shell) [14] ist eine Entwicklung des Williams College in Massachusetts. Es ist eines der komplexesten Tools für die Durchführung von PlanetLab Experimenten und vor allem für die Durchführung von aufwendigen Experimenten geeignet.

Plush besteht aus einem leichgewichtigen Daemon, der auf allen Slivern der PlanetLab Slice zu installieren ist und dem Application Controller, der üblicherweise auf dem Rechner des Nutzers läuft. Der Application Controller ist in der Lage, Operationen auf den Slivern mit Hilfe der Daemons auszuführen.

Experimente werden in Plush mittels XML Dateien beschrieben. Hierbei lassen sich auch sehr komplexe Vorbereitungsschritte (wie z.B. Softwareverteilung) und Nachbereitungsschritte (wie z.B. Einsammeln von Logfiles) definieren. Zur Unterstützung der Experimentedurchführung stellt Plush außerdem noch einige Monitoring Funktionen bereit.

Die Verwendungsmöglichkeiten von Plush beschränken sich nicht nur auf die Experimentedurchführung. Mit Plush kann man auch Verwaltungsaufgaben (wie z.B. das Hinzufügen und Löschen von Nodes zu einer Slice) durchführen.

Plush ist ein Kommandozeilenwerkzeug. Es steht jedoch auch die Nebula GUI für Plush zur Verfügung. Hiermit können Experimente geplant, durchgeführt und visualisiert werden, ohne sich mit den XML basierten Experimentenbeschreibungen von Plush auseinanderzusetzen.

3.7 PlMan

PlMan (PlanetLab Experiment Manager) [13] wurde an der University of Washington entwickelt. Es ist ein Tool, welches die wichtigsten Funktionen für die Durchführung einfacher Experimente in PlanetLab bereitstellt. Bei der Entwicklung von PlMan wurde vor allem das Ziel verfolgt, die Durchführung von PlanetLab Experimenten zu einem einfachen Prozess zu machen, der keine speziellen Kenntnisse voraussetzt.

PlMan bietet in seiner GUI die Möglicheit die folgenden Aufgaben mittels einfachen Point-and-Click Operationen durchzuführen:

- einfache Verwaltungsaufgaben (wie. z.B. das Hinzufügen von Nodes zur Slice)
- Dateitransfer zu und von den Slivern
- Ausführen von Kommandos auf den Slivern

PlMan fungiert dabei als grafisches Frontend für die openssh Tools. Da PlMan nur ssh Funktionen für die Kommunikation mit den Slivern nutzt, wird keine weitere Softwareinstallation auf den Slivern benötigt.

4 Kategorisierung der Tools

4.1 Kategorisierungsschema

Die oben vorgestellten Tools lassen sich nach ihren Funktionen und nach ihrem Bedienkonzept kategorisieren.

Einige der Tools sind für die Erfüllung einer ganz spezifischen Aufgabe vorgesehen, andere Tools decken mehrere Aufgaben ab und einige erheben für sich den Anspruch, komplette Experimentierumgebungen zu sein.

Grob lassen sich die gebotenen Funktionen in die Kategorien Softwaremanagement und Ressourcenmanagement einteilen. Dabei lassen sich innerhalb dieser beiden Kategorien noch folgende Einzelaufgaben unterscheiden:

Softwaremanagement

- Package Distribution
- Remote Execution
- Monitoring

Ressourcenmanagement

- Data Distribution
- Slice Management

Ein zusätzliches Unterscheidungsmerkmal ist, ob es sich bei den Tools um Kommandozeilenwerkzeuge, GUI Programme oder einer Kombination von beidem handelt.

Im Anhang A sind die in Abschnitt 3 vorgestellten Tools in dieses Kategorisierungsschema eingeordnet.

4.2 Auswahl eines geeigneten Tools für die PlanetLab VM

Bei der Auswahl eines geeigneten Tools für die PlanetLab VM, war vor allem der Funktionsumfang das entscheidende Kriterium. Ob das Tool ein Kommandozeilenwerkzeug oder ein Programm mit GUI ist, sollte eher zweitrangig sein.

Die PlanetLab VM soll es einem Nutzer ermöglichen schnell in die ersten Experimente mit PlanetLab einzusteigen. Um überhaupt die Durchführung von Experimenten zu ermöglichen, muss dass Tool die Übertragung der Experiment-Dateien

auf die Sliver ermöglichen (Data Distribution) und es muss mit ihm möglich sein, Programme auf den Slivern auszuführen (Remote Execution).

Ein weiterer häufig benötigter Schritt zur Durchführung eines Experimentes ist das Hinzufügen von Nodes zu einer Slice. Dies kann zwar prinzipiell ohne Zuhilfenahme eines Tools über das PlanetLab Webinterface geschehen, bei einer großen Anzahl Nodes ist dies jedoch nicht besonders effizient. Daher sollte das ausgewählte Tool auch diese Funktionalität bereitstellen. Von den vorgestellten Tools sind das alle Tools, die im Anhang A mit der Funktionalität Slice Management markiert sind.

Nach den oben genannten Ausführungen sollte das Tool demnach mindestens die Funktionen Data Distribution, Remote Execution und Slice Management erfüllen. Nach der im Anhang A aufgeführten Tabelle kamen daher die Tools pShell, Plush und PlMan in die engere Auswahl.

Plush ist von diesen drei Programmen ohne Zweifel das Funktionsreichste. Es erfordert allerdings auch einiges an Einarbeitungszeit. Da die PlanetLab VM das Hauptziel hat, dem Nutzer einen schnellen Einstieg zu ermöglichen, konnte dieses Tool für die Verwendung ausgeschlossen werden.

Zwischen den beiden verbliebenen Alternativen pShell und PlMan fiel die Wahl auf PlMan, da hier der Einstieg für den Nutzer durch die GUI noch vereinfacht wird und es außerdem noch ein Monitoring der laufenden Experimente ermöglicht.

5 Die PlanetLab VM

Ziel dieser Projektarbeit war die Erstellung eines Werkzeugkastens für PlanetLab. Dieses Ziel wurde mit der Zusammenstellung einer virtuellen Maschine verwirklicht. Zudem wurde der VM ein Tutorial hinzugefügt, welches einen neuen PlanetLab Nutzer von der Einrichtung seines Accounts bis zur Durchführung eines ersten Experimentes führt.

Die erstellte VM basiert auf dem Ubuntu 12.04 LTS Betriebssystem. Dieses wurde ausgewählt, da hierfür seitens des Herstellers Support bis zum April 2017 angekündigt wurde.

Das System wurde zunächst so konfiguriert, das nach kurzer persönlicher Anpassung (Einspielung bzw. Erstellung eines persönlichen ssh Schlüsselpaares) ein direkter Zugriff mittels ssh auf eine vorhandene PlanetLab Slice erfolgen kann.

In der VM sind ebenfalls zwei Programmdateien für die Durchführung eines ersten kleinen Experimentes vorhanden. Dabei handelt es sich um die in GNU-C geschriebenen Programme HelloSender und HelloReceiver. Das Programm HelloSender kann auf einer oder mehreren PlanetLab Slivern ausgeführt werden um dem Programm HelloReceiver eine festgelegte Anzahl von UDP Nachrichten in Zufallsintervallen zu senden.

Außerdem ist in der VM das PlanetLab Tool PlMan vorinstalliert, welches der Nutzer nur noch mit seinen persönlichen Zugangsdaten für PlanetLab konfigurieren muss. Danach kann er - durch das Tutorial geführt - das oben beschriebene Experiment leicht auf mehreren 100 Slivern gleichzeitig durchführen.

Dem Nutzer wird es somit ermöglicht innerhalb kurzer Zeit sein erstes PlanetLab Experiment durchzuführen. Mit Hilfe der dann für ihn fertig konfigurierten VM kann er danach leicht in seine eigenen Experimente einsteigen.

Literatur

[1] Huang, Mark L. ; Fiuczynski, Marc E. ; Klingaman, Aaron ; Soltesz, Stephen: *PlanetLab Technical Contact's Guide*. 2007

[2] Huang, Mark L. ; Fiuczynski, Marc E. ; Soltesz, Stephen: *PlanetLab Principal Investigator Guide*. 2008

[3] Intel Research Berkeley: *pssh Website*. `http://code.google.com/p/parallel-ssh/`

[4] McGill University: *pShell Website*. `cgi.cs.mcgill.ca/~anrl/projects/pShell`

[5] Peterson, Larry ; Muir, Steve ; Roscoey, Timothy ; Klingaman, Aaron: *PlanetLab Architecture: An Overview*. 2006

[6] Peterson, Larry ; Roscoe, Timothy: *The Design Principles of PlanetLab*. 2006

[7] PlanetLab: *PlanetLab Website*. `https://www.planet-lab.org`

[8] PlanetLab: *PlanetLab Node Manager API Documentation*. 2007

[9] Princeton University: *CoDeeN Website*. `http://codeen.cs.princeton.edu/`

[10] Princeton University: *CoDeploy Website*. `http://codeen.cs.princeton.edu/codeploy/`

[11] Princeton University: *CoTop Website*. `http://codeen.cs.princeton.edu/cotop/`

[12] University of Arizona: *Stork Website*. `http://www.cs.arizona.edu/stork`

[13] University of Washington: *PlMan Website*. `http://www.cs.washington.edu/research/networking/cplane/`

[14] Williams College: *Plush Website*. `http://plush.cs.williams.edu/`

Anhang A
Kategorisierung der vorgestellten Tools

	Software-management			Ressourcen-management		Bedien-konzept	
	Package Distribution	Remote Execution	Monitoring	Data Distribution	Slice Management	Command Line	GUI
Stork	●					●	●
CoDeploy				●		●	
CoTop			●			●	
Pssh		●		●		●	
pShell		●		●	●	●	
Plush		●	●	●	●	●	●
PlMan		●	●	●	●		●