NixOS – eine deklarative Linux-Distribution

Christian Kauhaus

NixOS bricht mit den Paketmanagement-Konventionen herkömmlicher Distributionen. Stattdessen beschreibt es das System umfassend mit Hilfe von Nix, einer eigens dafür entwickelten funktionalen Sprache. Dadurch sind bitgenau reproduzierbare Installationen, atomare Upgrades, Rollbacks und Mischungen beliebiger Softwareversionen möglich.

Der Vortrag gliedert sich in eine Einführung der zugrunde liegenden Konzepte, eine praktische Demonstration von NixOS-Anwendungsfällen sowie eine Übersicht über die NixOS-Community.

1 Anatomie einer NixOS-Installation

Die Unix-typischen Verzeichnisse */bin*, */usr/bin*, */usr/lib* oder */lib* sind unter NixOS fast leer oder komplett abhanden. Nachdem der Admin den ersten Schock überwunden hat, stellt er fest, dass sich ein Großteil der Installation in */nix/store* abspielt. Warum ist das so? Wie funktioniert der Nix-Store? Welche Vorteile ergeben sich daraus?

Eine NixOS-Installation wird durch eine deklarative Konfiguration vollständig und bitgenau definiert. Wir schauen uns eine beispielhafte *configuration.nix* an. Welche Elemente sind dort vorhanden? Wie werden sie in eine lauffähige Systemkonfiguration übersetzt? Wie kann ich herausfinden, welche Optionen zur Verfügung stehen?

Die Einführung schließt mit einer Übersicht über die wesentlichen Komponenten von NixOS: **nix** (Sprache), **nixpkgs** (Paketsammlung), **nixos** (Konfiguration).

2 Anwendungsfälle/Demos

2.1 Systemdefinition

In der ersten Demo zeige ich, wie ich eine einfache Konfiguration in einem laufenden System aktiviere und ggf. zurückrolle. Ebenso einfach ist es, Container- oder VM-Images aus derselben Konfiguration zu erzeugen.

2.2 Lokaler Paketmanager

Nix und nixpkgs lassen sich auch als lokale Paketmanager verwenden, ohne dass dafür eine vollständige NixOS-Installation aufgesetzt werden muss. Ich zeige, wie ich Software lokal aus NixOS-Channels installieren kann. Es ergeben sich ähnliche Vorteile wie bei der Verwendung von Snaps oder Flatpacks, nur viel einfacher.

2.3 Nix(OS) als Entwicklungs- und Deployment-Umgebung

Aufbauend auf den ersten beiden Anwendungen zeige ich, wie Nix und NixOS benutzt werden können, um einfach reproduzierbare und genau definierte Entwicklungsumgebungen für Softwareprojekte aufzusetzen. Dieselben Definitionen können dann in einem Continuous-Integration-System und später für das Deployment auf Servern genutzt werden.

3 Die NixOS-Community

Die NixOS-Community hat sich in den letzten Jahren sowohl zahlenmäßig als auch im Hinblick auf ihre Strukturen stark entwickelt. Ich stelle die vorrangigen Kommunikationskanäle vor. Weiterhin gehe ich auf das Entwicklungs- und Release-Modell ein. Zum Schluss zeige ich die wichtigsten Dokumentations-Ressourcen und wie man als Entwickler einsteigen kann.

Weitere Informationen: <https://nixos.org>

Referent: Christian Kauhaus <kc@flyingcircus.io>, Systems Engineer bei Flying Circus Internet Operations GmbH, Halle (Saale)