Software-Defined Networking

Trennen von Hardware- und Softwarefunktionen

Kai-Oliver Detken,
Thomas Rix

Der Ansatz Software-Defined Networking (SDN) abstrahiert die Steuerungs- und Datenschicht in Computernetzen voneinander, um Netze einfacher verwalten zu können. Dies geschieht, indem die unteren Funktionsebenen in virtuelle Dienste umgesetzt werden, so dass man eine manuelle Konfiguration der Hardware nicht mehr vornehmen muss. Durch den Einzug der Virtualisierung in die Rechenzentren und Unternehmen wurde dieser Ansatz, der ursprünglich von der Stanford University (USA) im Jahr 2006 entwickelt wurde, immer notwendiger.

Prof. Dr.-Ing. Kai-Oliver Detken ist Dozent an der Hochschule Bremen im Fachbereich Informatik sowie Geschäftsführer der Decoit GmbH, Thomas Rix arbeitet als Softwareentwickler bei der Decoit GmbH in Bremen

SDN ermöglicht es, das Management einer großen Anzahl unterschiedlicher Ressourcen in höherer Skalierung vorzunehmen. Dabei wird die IT-Infrastruktur von einem SDN-Controller verwaltet und gesteuert, der logisch und physisch unabhängig von der Infrastruktur agiert. Die Steuerung wird daher in einer Softwareschicht durchgeführt, die auf einer separaten Hardware arbeitet. Dies wird durch die immer größere Verbreitung von Virtualisierungstechniken und Cloud-Lösungen auch immer wichtiger, da so das Netzmanagement zentralisiert vorgenommen werden kann und Konfigurationen nicht auf einzelne Netzkomponenten umgesetzt werden müssen. Zusätzlich erhält man eine größere Flexibilität, da die Softwaresteuerung an beliebigen Stellen aufgesetzt werden kann. Änderungen sind so schneller und fehlerfreier umzusetzen. Die Cloud-Nutzung bietet bereits die Möglichkeit, dynamisch auf Ressourcen zuzugreifen und diese unterschiedlich flexibel anordnen zu können. Der SDN-Ansatz erweitert diesen serverbasierten Ansatz auf die Netzumgebung, die dann ebenfalls virtuell zur Verfügung steht und gleichermaßen flexibel konfiguriert und ausgerollt werden kann. Es muss also nicht jeder Switch einzeln manuell konfiguriert werden, sondern es wird zentral eine Policy (Richtlinie) festgelegt und ausgerollt. Dies gilt für Firewall-Regeln, Bandbreitenzuteilung usw., so dass die Hardwarekomponenten nur noch das Forwarding von Paketen zum korrekten Port übernehmen.

Die Open Networking Foundation (ONF) widmet sich der Verbreitung und Implementierung von SDN inkl. der damit zusammenhängenden Protokolle und verwaltet auch den OpenFlow-Standard. Dieser bietet eine standardisierte Kommunikationsschnittstelle zwischen der Steuer- und Weiterleitungsebene einer SDN-Architektur an. OpenFlow ermöglicht den direkten Zugriff auf die Forwarding-Schicht eines Switches oder Routers – sowohl physisch als auch virtuell auf Basis eines Hypervisors. Der OpenFlow-Standard verfolgt daher den Ansatz, die Steuerung des Netzes aus den Geräten hinaus in eine logisch zentralisierte Software zu verlagern, d.h., der Weg der Pakete durch das Netz wird von einer Software bestimmt, die auf mehreren Routern läuft. Durch die Nutzung von Access Control Lists (ACL) und Routing-Protokollen wird die Trennung von Control- und Forwarding-Ebene ermöglicht. Es gibt bisher kein anderes Protokoll, das einen ähnlichen Ansatz verfolgt und somit SDN-basierte Umgebungen ermöglicht.

SDN-Einsatzmodelle und -Anwendungen

Es gibt verschiedene Einsatzmodelle für den SDN-Ansatz:

- Asymmetrisches/symmetrisches Modell: Die globale Information wird soweit es geht im asymmetrischen Modell zentralisiert, während der Betrieb der Switche so weit wie möglich verteilt wird. Die zentrale Ausrichtung bedingt aber eine höhere Störanfälligkeit sowie eine schlechtere Skalierbarkeit. Das symmetrische Modell ermöglicht es, dass jede Komponente auch alle für sie relevanten Kontrollschichtkonfigurationen kennt, so dass bei einem Teilausfall die verbleibende Infrastruktur weiter genutzt werden kann.
- Floodless/Flood-basiertes Modell: Das Floodless-Modell stellt die Funktion aller Komponenten sicher, indem lokale Caches von Lookup-Tabellen angelegt werden, die kontinuierlich miteinander synchronisiert werden. Das Flood-based-Modell verteilt Änderungen durch Broadcast- oder Multicast-Nachrichten,

wodurch ein SDN symmetrisch aufgebaut werden kann. Allerdings steigt die Netzlast pro Knoten, was die Skalierbarkeit einschränkt.

- Host-basiertes/netzzentriertes Modell: Das Host-basierte Modell nimmt die SDN-Verarbeitung auf einer virtuellen Maschine (Hypervisor-System) vor, während das netzzentrierte Modell z.B. die Routing-Funktionalität auf dem Router belässt.

Nicht immer lassen sich alle Modelle voneinander exakt abgrenzen. So kann man z.B. Performance-lastige Aufgaben dediziert vergeben, während andere Funktionen auf separaten SDN-Servern angesiedelt sind. Zusätzlich ergänzen sich die Modelle gegenseitig.

Der Cloud-Typ „Infrastructure as a Service" (IaaS) ist eine typische Anwendung von SDN. Hier wird das SDN mit virtuellen Systemen und virtuellem Speicher kombiniert, was eine sehr effiziente Ressourcenverteilung bewirkt. Werden weitere Ressourcen benötigt, können diese einfach hinzugefügt werden. Die virtuellen Systeme sind dabei vollständig von der Hardware (und dem Netz) entkoppelt worden.

Weitere SDN-Ansätze ermöglichen es, die Ressourcen von virtuellen Hosts besser sowie ungenutzte Rechenkapazitäten auszunutzen. Dies wird heute oftmals nur händisch vorgenommen. Auch die Verteilung der Lasten über Load Balancing auf verschiedene Netzverbindungen kann dabei Berücksichtigung finden. So lassen sich Latenzzeiten, Bandbreiten, Verfügbarkeiten und Sicherheitsmerkmale definieren und als Service Level Agreements (SLA) umsetzen.

Steuerung von Netztopologien

Das BMBF-Forschungsprojekt Visa (Virtual IT Security Architectures) hat sich mit virtuellen Netzinfrastrukturen beschäftigt und dabei die Infrastruktureigenschaften berücksichtigt. Hier stand im ersten Schritt die Simulation von physischen IT-Infrastrukturen im Vordergrund, die man virtuell nachbaute, um Fehlkonfigurationen oder Leistungsengpässe zu vermeiden. Im zweiten Schritt wurde auch eine komplette virtuelle Umgebung (Server und Netz) zur Verfügung gestellt, die in die Produktivumgebung übertragen werden konnte. Das heißt, auch hier wurden bereits die Soft- von der Hardware getrennt und bewusst der SDN-Ansatz gewählt, um eine fehlerfreie Konfiguration sowie einen höheren Sicherheitsgrad zu erhalten.

Eine bisher gängige Möglichkeit zur Netzkonfiguration ist die Erstellung virtueller Schnittstellen, mit denen man die virtuellen Machinen (VM) über Bridges verbinden kann. Bei diesen handelt es sich um eine direkte Weiterleitung des Datenverkehrs von einem Netzinterface zu einem anderen. Sie erfolgt über eine Konfiguration der Netzadapter des Betriebssystems und ist weit verbreitet. Mit Bridges und virtuellen Netzadaptern ist es somit möglich, ein beliebig großes Sternnetz aufzubauen. Dies spiegelt jedoch nur bedingt die Realität wieder, da z.B. keine Switche mit entsprechender Funktionalität abgebildet werden können. Auch lassen sich durch diese Art der Netzvirtualisierung nur perfekte (unmittelbare, verlust- und fehlerfreie) Verbindungen erstellen. Da virtuelle Netze auch zu Testzwecken benutzt werden, stellt dies eine Einschränkung der Funktionalität dar.

Grafische Tools zu Entwurf und Umsetzung von Netztopologien, die das Spektrum Layer 1 (Kabel) bis Layer 3 (Switche, Router) mitsamt VM abdecken sind heute hingegen kaum zu finden oder zu komplex in der Bedienung. Daher sind derzeit wenige Lösungen wie z.B. GNS3 zum Erstellen und Steuern von virtuellen Netztopologien vorhanden. Diese Software ist ein grafischer Netzsimulator, der auf dem Router-Emulator Dynamips und dessen Frontend Dynagen aufbaut. Dynagen ermöglicht, komplexe Router-Netze mithilfe einer einfachen Konfigurationsdatei zu erstellen. Sie ist als eine GUI für Dynagen zu verstehen. Durch die grafische Unterstützung lassen sich mit GNS3 sehr komfortabel komplexe virtuelle Netze erstellen. Allerdings ist diese Lösung auf Cisco-Router spezialisiert.

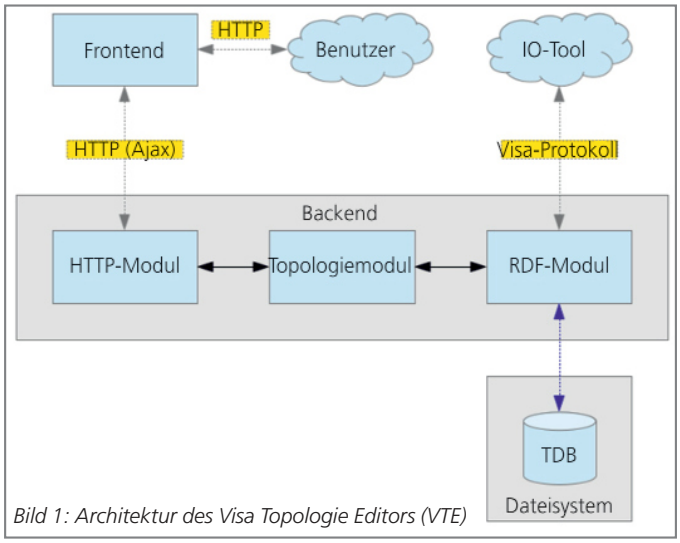

Bild 1: Architektur des Visa Topologie Editors (VTE)

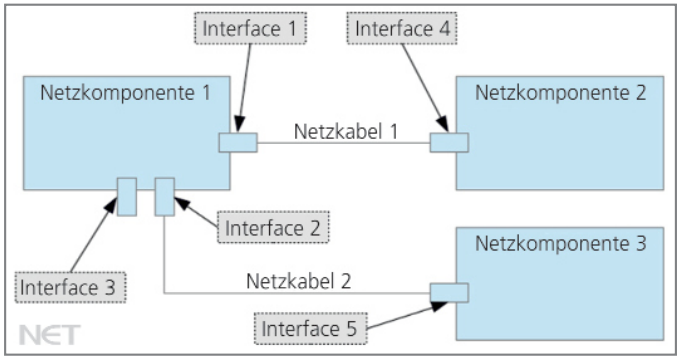

Bild 2: Schematische Darstellung der im VTE erzeugten Objektstruktur

Das Visa-Projekt entwickelte aufgrund fehlender herstellerübergreifender Lösungen den eigenen Visa Topologie Editor (VTE), der die grafische Konzeption von virtuellen IT-Infrastrukturen flexibel ermöglicht und damit einem SDN-Ansatz nahekommt. Er bietet dem Benutzer die Möglichkeit, eine bereits bestehende und vorher erfasste Netztopologie zu bearbeiten sowie neue Komponenten hinzuzufügen. Weiterhin kann auch eine neue bzw. bestehende Topologie von Hand nachmodelliert werden. Der VTE besteht aus zwei Kernkomponenten:

- Backend: in Java geschriebener Serverdienst;

- Frontend: webbasierte grafische Oberfläche zur Netzkonfiguration.

Visa Topologie Editor

Das Interconnected-Asset Ontology Tool (IO-Tool) ist in der Lage, bestehende IT-Infrastrukturen zu erfassen und über eine RDF-/XML-Datei oder durch eine TLS-gesicherte TCP-Verbindung dem VTE zum Import bereitzustellen. Selbst erstellte oder veränderte IT-Infrastrukturen des VTE können als RDF/XML-Datei gesichert oder in das IO-Tool übertragen werden. Das Backend besteht aus dem Topologie-, RDF- und HTTP-Modul (*Bild 1*).

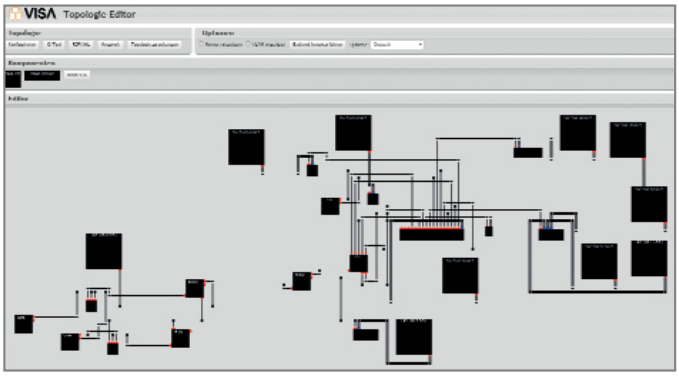

Bild 3: Web-Frontend des VISA Topologie Editors (VTE)

Das Topologiemodul stellt die Topologie, die entweder über das Webinterface oder aus RDF/XML-Daten importiert wurde, mithilfe von Java-Klassen dar. Das zentrale Element dieses Moduls ist die abstrakte Klasse „Network Component" (Netzkomponente), die als Basisklasse aller Komponenten in der Topologie dient. Als Komponente werden alle physischen und virtuellen Geräte bezeichnet, die sich wiederum in drei Unterklassen einteilen lassen. Daneben gibt es noch Klassen für Netzschnittstellen und -kabel. Die Klasse für Netzschnittstellen ist als innere Netzkomponentenklasse implementiert und heißt Interface. Sie enthält Eigenschaften, die für jedes Interface unterschiedlich sind wie z.B. die IP-Konfiguration.

Bild 2 zeigt die Struktur, die im Topologiemodul erzeugt wird. Jedes Objekt der Netzkomponentenklasse besitzt ein oder mehrere Interfaceobjekte, die die Schnittstellen dieser Komponente darstellen. Die Interfaceobjekte werden wiederum mit Objekten der Netzkabelklasse verknüpft, womit die Komponenten, zu denen die beiden Ports gehören, miteinander verbunden sind. Netzkomponente 1 könnte also ein Switch sein, der die Netzkomponenten 2 und 3 miteinander verbindet.

Das RDF-Modul sorgt für die Verwaltung der RDF-Informationen. Dazu baut es auf dem Open Source Framework Jena auf. Die zentrale Klasse des Moduls ist der RDF-Manager. Wird ein Objekt dieser Klasse erzeugt, wird ein sog. Dataset erstellt, das die RDF-Informationen speichert und verwaltet. Ein Dataset enthält mindestens ein RDF-Modell (Default Model). Neben diesem können noch beliebig viele „Named Models" in einem Dataset enthalten sein. Das im RDF-Manager verwendete Dataset verwendet das Datenbanksystem TDB als Speichermedium, das die Verwendung von Transaktionen für jeglichen Zugriff auf die gespeicherten Modelle erlaubt und im Fehlerfall einen dauerhaften Schaden an den Daten verhindert.

Das HTTP-Modul stellt einen einfachen HTTP-Server zur Verfügung, der die vom Frontend abgesetzten Ajax-Requests verarbeiten und beantworten kann. Die zentrale Klasse dieses Moduls ist der Ajax-Server, der eine Instanz des HTTP-Servers startet und die HTTP-Handler, die die einzelnen Ajax-Requests bearbeiten, definiert.

Das Frontend des VTE wurde als Weboberfläche größtenteils in JavaScript entwickelt. Im oberen Bereich der Oberfläche befindet sich die Optionsleiste (*Bild 3*). Sie erlaubt verschiedene Einstellungen und gibt Zugriff auf diverse Funktionen des Editors. Die erste Option erlaubt das Einblenden der vollen Namen aller Komponenten auf dem Raster, die zweite das Einfärben der Kabel entsprechend des VLAN, zu dem sie gehören. Beide Optionen können zeitgleich aktiviert werden und blockieren jeweils die Drag-and-Drop-Funktionalität des Editors, um Anzeigeprobleme zu verhindern. Aktive Komponenten werden als schwarze Boxen dargestellt. Beim Hinzufügen neuer Komponenten stehen dem Benutzer die Optionen Name, Breite/Höhe, Anzahl der Netzschnittstellen, Position der Schnittstellen an der dargestellten Box zur Verfügung.

Die Möglichkeit, mehrere Komponenten zu einer Gruppe zusammenzufassen, ist eine der zentralen Funktionen des VTE. Da das Raster, auf dem die Komponenten abgebildet werden, relativ klein ist, reicht der Platz höchstens für ca. 20 Geräte. Danach wird die Darstellung unübersichtlich oder ist überhaupt nicht mehr möglich. Um trotzdem größere Topologien darstellen zu können, werden Komponenten, die bestimmte Eigenschaften teilen, zusammengefasst. Die Gruppen werden im Editor als Objekte dargestellt und nehmen dadurch relativ wenig Platz ein. Ein Klick auf die Gruppe erlaubt die Einsicht ihrer Inhalte.

Fazit

SDN-Ansätze und -Modell befinden sich noch in der Anfangsphase ihrer Entwicklung. Durch den Einzug der Virtualisierung wird es aber immer notwendiger, dass die Ressourcen, die auf den physischen Host-Systemen frei sind bzw. werden, auch entsprechenden Anwendungen und Diensten zur Verfügung gestellt werden können. Diese Ressourcenverteilung wird momentan häufig manuell vorgenommen und ist daher noch nicht ausreichend effizient. Ebenso wird die Netzebene dabei kaum einbezogen. So lassen sich zwar die Serversysteme heute einfach virtualisieren, aber nicht die darunterliegende IT-Infrastruktur. SDN-Modelle und -Anwendungen ermöglichen hier eine deutliche Verbesserung, da sie zum einen die Entkopplung der Hardware von der Software vornehmen und zum anderen eine effizientere Aufteilung der Ressourcen unter Beibehaltung der Skalierbarkeit ermöglichen. Das Forschungsprojekt Visa zeigt, wohin der Weg gehen kann. Softwarelösungen wie OpenStack, auf dem auch die Visa-Lösungen basieren, zeigen einen neuen Weg in eine flexible SDN-Umgebung auf. *(bk)*