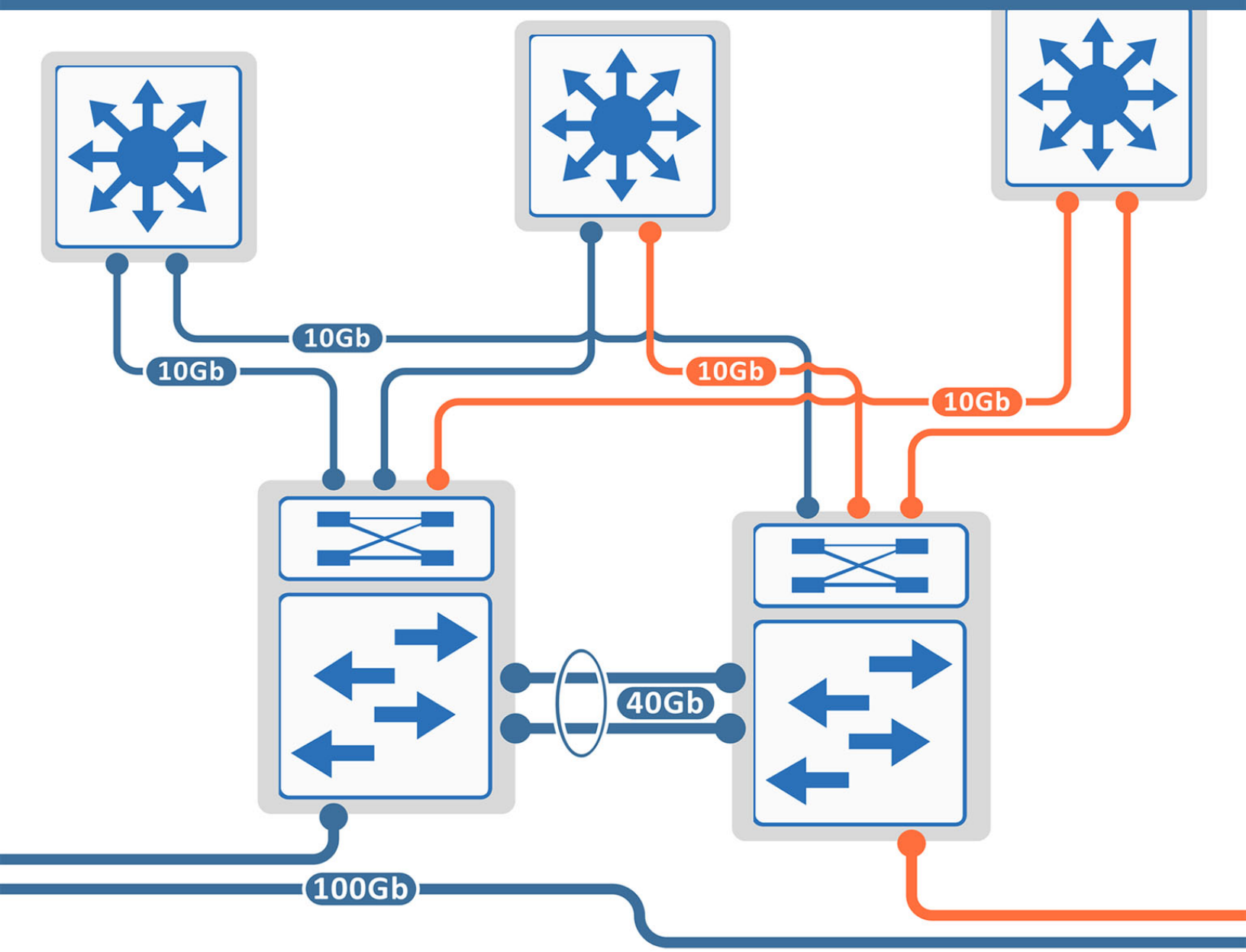

Der OpenSwitch-Praktiker

Datacenter-Switching mit Linux

Markus Stubbig

Der OpenSwitch-Praktiker

Markus Stubbig

Der OpenSwitch-Praktiker

Bibliografische Information der Deutschen Nationalbibliothek
Die Deutsche Nationalbibliothek verzeichnet diese Publikation in der Deutschen Nationalbibliografie; detaillierte bibliografische Daten sind im Internet über http://dnb.dnb.de abrufbar.

Herstellung und Verlag: BoD - Books on Demand, Norderstedt

1. Auflage 2019
ISBN: 978-3-7504-8008-7

Vorwort

Ein Open-Source-Betriebssystem für Switches? Wofür? Jeder Switch hat doch bereits ein Betriebssystem. In der Welt der Netzwerker scheint jeder Hersteller sein eigenes *Network-OS* zu kochen. Das Spektrum der Ergebnisse reicht von vielseitig über kurzlebig, verwaist bis zu stabil und weltbekannt. Manche Distributionen beginnen als Fork und enden nur Monate später.

Was ist also so toll an OpenSwitch? Es ist ein reguläres Linux, in dem sich die Entwickler und Administratoren austoben dürfen. Alles ist erlaubt und machbar. Damit lassen sich die Netzwerke im Rechenzentrum individuell zusammensetzen. Kurz: OpenSwitch ist der Maßanzug für das moderne Netzwerk.

Auf der anderen Seite ist OpenSwitch ein *Bring your own Device* für Netzwerkswitches. Hersteller-OS runter und OpenSwitch drauf. Letztendlich ist ein Switch auch nur ein kleiner Server mit erstaunlich vielen Netzadaptern.

Die Entwickler, Partner und Sponsoren fördern das Projekt mit Ideen, Weitsicht, Programmcode und Anleitungen. Das OpenSwitch-Team produziert keine Hardware, sondern arbeitet mit Netzwerkausrüstern zusammen. Damit der Deckel perfekt auf den Topf passt, erhält OpenSwitch Einblick in die Hardwarespezifikation. Mit diesen Infos kommen die ASICs so richtig in Fahrt und treiben die Durchsatzraten in die Höhe.

Das Konzept hat leider ein operatives Problem: Für Netzwerker ist OpenSwitch zu viel Linux, Konfigurationsdateien und Skripte. Für Linux-Admins ist OpenSwitch zu viel Netzwerkerei, Protokolle und Adressen.
Zum Glück schließt sich diese Lücke langsam durch Kompatibilitätstests, gute Dokumentation, Erfolgsrezepte und ein hervorragendes Buch.

Viel Spaß beim Ausprobieren, Staunen und Fluchen.

Ressourcen

https://www.openswitch.net
Die Homepage von OpenSwitch liefert einen guten Einstieg

ins Thema und verlinkt zur Dokumentation, zu Anwendungen und zum Download-Bereich.

https://github.com/open-switch
Die Entwickler hosten den Programmcode bei GitHub, wo jeder Einblick in den Fortschritt hat und sich an den Quellen bedienen kann. Daneben gibt es viele Demo-Projekte und Beispiele zum selber bauen.

https://chat.openswitch.net
Im offiziellen Chat sind Anwender und Entwickler vertreten und bereit für Ideen, Diskussionen und Support aus der Community.

Rechtliches

Einleitung

OpenSwitch ist ein quelloffenes Netzwerk-Betriebssystem für Switches. Es basiert auf Debian GNU/Linux und vereint Techniken wie VXLAN, iptables, Kanalbündelung und Routing unter einer gewohnten Kommandozeile. OpenSwitch läuft auf physikalischer Hardware oder als virtuelle Maschine.
Jeder Ausrüster von Netzwerkkomponenten hat ein eigenes Betriebssystem im Angebot. OpenSwitch verkauft keine Switches. Die Grundidee ist: ein Linux-Betriebssystem für viele Hardwareplattformen anzubieten. OpenSwitch ersetzt auf *anderen* Switches das Betriebssystem und kann dann loslegen.
In dieser Nische hat sich OpenSwitch einen Namen gemacht. Dort punktet es in den Bereichen Funktionalität und Erweiterbarkeit. OpenSwitch verbindet die Flexibilität von Linux mit den Anforderungen eines Datacenter-Switches bei minimaler Budgetanforderung.

OpenSwitch ist:

Unvollkommen.

Und das ist positiv gemeint. Es gibt noch genug Raum zum Wachsen. Auch die Implementierung von Features hinkt etwas hinterher: bei der Multi-Chassis-Kanalbündelung und beim Paketfilter bestehen Nachholbedarf.

Open Source.

OpenSwitch setzt auf Linux als Betriebssystem und stellt viele seiner Eigenentwicklungen offen auf GitHub zur Schau [1]. Aber der Vorteil einer quelloffenen Lösung ist nicht immer ihr Preis. Denn kostenlos ist Open-Source-

Software nicht! Es fallen zwar keine Lizenzgebühren an, aber die Arbeitszeit der Netzwerk-Admins zum Einarbeiten in die Linux-Welt darf nicht unterschätzt werden.
OpenSwitch hält das Zusammenspiel mit dem Chipsatz der Switches unter dem Deckmantel des proprietären Codes. Und diese Geheimniskrämerei kommt vermutlich auf Wunsch der Hersteller, die alle Details ihrer Netzwerkprozessoren als Firmengeheimnis hüten.

Try before Buy.

Wie bei Shareware-Programmen kann (und sollte) OpenSwitch vor dem Einsatz getestet werden, bevor irgendwelche Investitionen in die Infrastruktur beginnen. Und wer freut sich über einen eingeschränkten Funktionsumfang, eine Evaluierungslizenz oder einen 30-Tage-Zeitraum?
In diesem Zusammenhang steht *Try* für Ausprobieren mit Beispielszenarien und *Buy* für den Einsatz in der eigenen Umgebung.

Hardware-frei.

OpenSwitch ist Software. Diese Software braucht eine Hardware. Die Antwort der Hardware-Frage liefert eine Kompatibilitätsliste aus etwa zwanzig Geräten verschiedener Hersteller und Leistungsklassen.
In der Vergangenheit gab es viele limitierende Gründe, warum eine softwarebasierte Lösung für Netzwerkinfrastruktur nicht an die Leistung der physikalischen Geräte herankam. Der Hauptgrund war das suboptimale Zusammenspiel von Software und Treiber mit der darunterliegenden Hardware. Bei der immens großen Auswahl von Netzwerkkarten, Mainboards, Prozessoren und Memory ist es für eine Software schwierig auf jede Kombination der Komponenten optimal vorbereitet zu sein.

Heutzutage sind normale Server oder eingebettete Systeme überraschend performant, sodass auch eine nicht-optimierte Software Bandbreiten jenseits von Gigabit durchbrechen kann.

Linux.

Unter OpenSwitch läuft ein unveränderter Linux-Kernel. Der Zugriff aufs Betriebssystem ist nicht gesperrt. Über das Konsolenmenü oder eine SSH-Verbindung und einem einfachen `sudo bash` liegt der Zugang offen.
Das bringt Möglichkeiten zum Anpassen, Verbessern und Nachinstallieren von Tools. Dagegen steht die Gefahr, dass die eigene Änderung ungewollte Instabilität mitbringt.

Best Of.

OpenSwitch erfindet an vielen Stellen das Rad nicht neu und bedient sich für seine Features an den vertrauten Linux-Diensten, die nach Jahren der Entwicklung eine hohe Stabilität erreicht haben. Die Implementierung der Routingprotokolle stammt von FRRouting, der SSH-Server gehört zu OpenSSH und beim Logging helfen Syslog und Journald.
Diebstahl? Keineswegs! Eher ein Nachweis, dass Open Source funktioniert. Solange Lizenzbedingungen eingehalten werden, darf Fremdsoftware beigemischt werden. Gerade im Security-Umfeld ist es höchst erwünscht, dass Anwendungsentwickler keine eigenen Implementierungen stricken, sondern sich an den freien und stabilen Bibliotheken bedienen.

White-Box-Switches

Ohne Hardware kann auch OpenSwitch nichts ausrichten. Was im Serverumfeld gängige Praxis ist, erscheint in der Netzwelt skurril: Denn *White-Box-Switches*, oder *Bare-Metal-Switches*, sind Netzwerkgeräte *ohne* Betriebssystem. Damit ist nicht gemeint, dass das Betriebssystem vergessen wurde. Vielmehr hat der Kunde die freie Wahl und kann sich sein Wunschmodell so zusammenstellen, dass es in die Infrastruktur von Rechenzentrum, Monitoring, Verwaltung und Automatisierung passt.
Diese Entkopplung von Hard- und Software hat auf beiden Seiten Vorteile. Egal welcher Hersteller von White-Box-Switches im Netzwerk mitspielt, das Betriebssystem sieht auf allen Boxen gleich aus. Das vereinfacht die Administration und den Lernaufwand, auch wenn die Architektur unterschiedlich ist.
Welches Betriebssystem darf es denn sein? Grundsätzlich läuft auf einem White-Box-Switch auch eine Distribution von Red Hat oder Ubuntu. Allerdings sind diese Anbieter nicht auf die Ausstattungen von Netzwerkswitches vorbereitet. OpenSwitch als Netzwerkbetriebssystem (*network operating system*, NOS) hat eine starke Ausrichtung auf Switchports, Buffer, SFPs, ASICs, CLI und die Überwachung von Temperatur und Lüfter.
Auf der Hardwareseite schonen White-Box-Switches das Budget. Diese Switches sind damit nicht billig, aber deutlich preisgünstiger als ein klassischer Switch von einem renommierten Netzwerkausrüster. Denn eine offene Preispolitik und das erklärte Ziel, die White-Box-Switches am Markt zu etablieren, bringen finanzielle Vorteile für die Käufer.

Geschichte

Die Historie von OpenSwitch ist noch relativ kurz. Es beginnt im Oktober 2015: HP stellt OpenSwitch als Community-

Projekt vor. Das Ziel ist ein offenes Betriebssystem für Netzwerkswitches, das auf Datacenter-Switches von namhaften Herstellern operiert und erweiterbar ist. Die Software unterstützt Switches von Edgecore und HP. Intern heißt das Projekt "OPS".
Im Juni 2016 wechselt OpenSwitch zur Linux Foundation und wird damit Hersteller-neutral. Die Linux Foundation ist eine gemeinnützige Organisation mit den Zielen, Linux bekannter zu machen und Open-Source-Software zu fördern. Das Projekt heißt fortan intern "OPX". Seit dem Wechsel hält sich HP mit Entwicklung und Support zurück. Dell/EMC wird die treibende Kraft.
Die Entwicklung zieht sich hin, aber im Februar 2017 veröffentlicht die Linux Foundation ihr neues "OpenSwitch" und beginnt mit der Versionsnummer 2.0.
In den folgenden Jahren beschäftigt sich das OpenSwitch-Projekt mit Entwickeln und veröffentlicht in unregelmäßigen Abständen ihre Ergebnisse. Parallel dazu portieren die Entwickler ihre Software auf weitere Switches und vergrößern damit die Kompatibilitätsliste. Im September 2018 erreicht OpenSwitch die Version 3.0 und bringt mit diesem Meilenstein zahlreiche Features.
OpenSwitch hat zum Entwicklungsbeginn bei Versionsnummer 0.1.0 gestartet und ist bisher (2019) bei Version 3.2.0 angekommen. Für eine Webrecherche sind die Schlagwörter "openswitch -vswitch" am aussagestärksten.

Kapitel 1: Quickstart

Dieses Kapitel ist eine Kurzanleitung zum ersten Umgang mit OpenSwitch und gibt eine Übersicht der Version, Netzadapter und die Gesundheit der Hardware.

Was ist OpenSwitch?

OpenSwitch OPX ist ein Betriebssystem für Netzwerkswitches. Es basiert auf Debian Linux und bringt viele Erweiterungen mit, welche die Hardwarekomponenten korrekt ansteuern. Die verwendete Linux-Distribution ist weitgehend im Urzustand. Damit lässt sich ein Switch mit OpenSwitch ähnlich verwalten und konfigurieren wie ein Server - mal abgesehen von netzwerkspezifischen Einstellungen wie Fan-Out, ACLs oder Spanning-Tree.

Zugang

Ein OPX-kompatibler Switch hat stets einen seriellen Konsolenport, einen Switchport für Management und viele Switchports für Server oder Uplinks. Der serielle Zugang erfolgt über einen RJ45-Port mit RS232-Signalisierung. Ein handelsübliches Konsolenkabel und eine Terminalsoftware öffnen ein Konsolenfenster für das erste Login.
Der Managementadapter erwartet eine Ethernetverbindung und einen aktiven DHCP-Server. Sobald sich der Switch erfolgreich eine IP-Adresse geholt hat, darf die SSH-Software ein Login wagen.
In beiden Fällen meldet sich der Switch mit der Linux-typischen Begrüßung und einem Login-Prompt. Die Anmeldung mit dem Benutzernamen *admin* und Kennwort *admin* öffnet die Tore und präsentiert die Linux-Shell Bash.

Wenn höhere Berechtigungen benötigt werden, wechselt `sudo bash` das Benutzerkonto und die Shell läuft anschließend als `root`-User. Abbildung 1.1 zeigt das SSH-Login auf den ersten Switch im Labornetz, das in den folgenden Kapiteln vorgestellt wird.

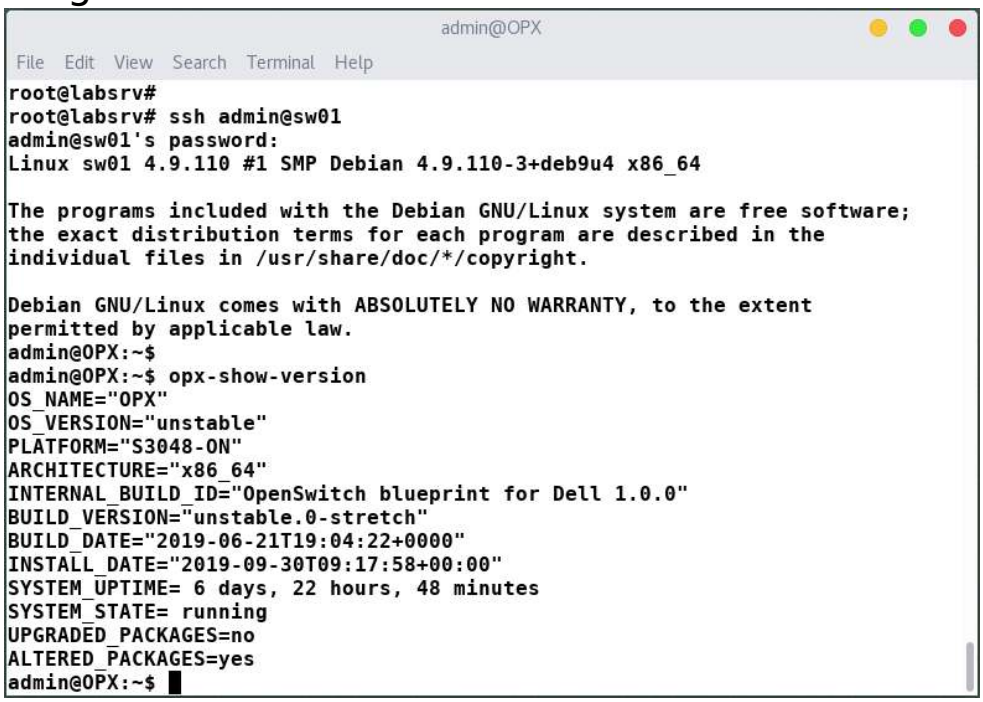

Abbildung 1.1: SSH-Login in einen OPX-Switch

IP-Adresse

Wenn sich der Switch keine IP-Adresse per DHCP beschaffen konnte, erledigt dies das `ip`-Kommando per seriellem Zugang. Die Syntax ist identisch mit jedem anderen Linux-Server, da es sich um dieselben Befehle handelt. Das folgende Beispiel vergibt dem Verwaltungszugang die IPv4-Adresse 10.5.1.1 und nutzt 10.5.1.250 als Gateway.

```
p address add 10.5.1.1/24 dev eth0
p route del default
p route add default via 10.5.1.250
```

Bei passender Infrastruktur ist jetzt ein Login via SSH möglich. Der SSH-Dienst ist in OpenSwitch automatisch aktiv und benötigt keine Ersteinrichtung.

Übersicht

Der Switch begrüßt den eingeloggten Admin schlicht mit `admin@OPX` und wartet auf Befehle. Welche Switchports hat das Gerät und in welchem Gesamtzustand befindet sich das System?
Zuerst stellt sich der neue Switch vor und informiert über seinen Hersteller und die Modellbezeichnung:

```
admin@OPX:~$ sudo opx-show-version
OS_NAME="OPX"
OS_VERSION="3.1.0"
PLATFORM="S3048-ON"
ARCHITECTURE="x86_64"
INTERNAL_BUILD_ID="OpenSwitch blueprint for Dell 1.0.0"
BUILD_VERSION="3.1.0.0-rc1"
BUILD_DATE="2018-12-19T12:31:44-0800"
INSTALL_DATE="2019-08-23T12:49:56+00:00"
SYSTEM_UPTIME= 1 minute
SYSTEM_STATE= running
UPGRADED_PACKAGES=no
ALTERED_PACKAGES=no
```

Die Führung geht weiter zu den Netzadaptern, die tabellenförmig aufgelistet werden. Erwartungsgemäß sind alle Anschlüsse (mit Ausnahme des Managementadapters) vorhanden und abgeschaltet:

```
admin@OPX:~$ sudo opx-show-interface --summary
Port       | Enabled | Operational status | Supported speed
-------------------------------------------------------------
e101-001-0 | no      | down               | 10M 100M 1G
e101-002-0 | no      | down               | 10M 100M 1G
e101-003-0 | no      | down               | 10M 100M 1G
e101-004-0 | no      | down               | 10M 100M 1G
e101-005-0 | no      | down               | 10M 100M 1G
[...]
```

```
e101-049-0 | no      | down                | 100M 1G 10G
e101-050-0 | no      | down                | 100M 1G 10G
e101-051-0 | no      | down                | 100M 1G 10G
e101-052-0 | no      | down                | 100M 1G 10G
eth0       | yes     | UNKNOWN             | UNKNOWN
```

Die Liste passt zu einem Modell mit 48 Gigabit-Adaptern und vier TenGigabit-Anschlüssen. Der Verwaltungszugang `eth0` ist separat aufgeführt und beendet die Liste.
OpenSwitch benennt seine Netzadapter abhängig von ihrer Position an der Vorderseite des Gehäuses. Das Präfix *e101* und die Fan-Out-Nummer werden in Kapitel 3 beschrieben.

Zuletzt gibt der Switch in zwei knappen Kommandos Auskunft über seinen Gesundheitszustand.

```
admin@OPX:~$ sudo opx-show-system-status
System State:  running
No Failed Service
No Modified Package
admin@OPX:~$ sudo opx-show-alms
2019-08-23 14:52:51.395658      PSU 1 absent
```

Die Befehlsausgabe bestätigt, dass die Hardware schnurrt und lediglich das redundante Netzteil (PSU 1) vermisst.

Zusammenfassung

OpenSwitch lässt sich nicht in zehn Minuten erklären, aber für einen kurzen Einstieg reicht es. Hinter den blinkenden LEDs läuft ein Debian Linux, welches eine IP-Adresse im Netz benötigt und einen regulären SSH-Zugang anbietet. Nach einem erfolgreichen Login enthüllen OPX-spezifische Befehle eine Liste der Netzadapter, den Systemstatus und eventuelle Probleme.

Kapitel 2: Installation

Wenn der vorliegende Switch das Betriebssystem seines Herstellers mitbringt, gibt es mehrere Wege dies zu ändern. In allen Fällen wird die OPX-Software auf der Flashkarte installiert und davon gebootet. Danach startet der Switch, als hätte er nie ein anderes Betriebssystem gesehen.
Das Ziel dieses Kapitels sind viele installierte Switches - physisch oder virtuell. Zuletzt soll ein kleines Demo-Netz mit sechs Switches entstehen, welches im Verlauf des Buchs für Beispiele und Szenarien herhält. Abschnitt *Labor* liefert die Details.

Das geplante Labor benutzt OPX-Version 3.2.0 mit der Installationsdatei:

```
PKGS_OPX-3.2.0-installer-x86_64.bin
```

USB-Stick

Für einen White-Box-Switch oder ein kompatibles Modell stellt OpenSwitch ein schlüsselfertiges Installationsimage bereit [2]. Dieses ersetzt das eventuell vorhandene Betriebssystem. Die Kunst liegt darin, es auf die vorhandene Hardware zu kopieren und zu starten.
Genau für diesen Zweck benutzt OpenSwitch das *Open Network Install Environment* (ONIE). Diese Installationsumgebung hat den Zweck, unterschiedliche Betriebssysteme auf einem Switch zu starten. Damit hat der Anwender die Wahl zwischen mehreren Betriebssystemen oder der Installation eines neuen Netzwerk-OS, wie z. B. OpenSwitch OPX.
Ein genauer Blick auf ONIE enthüllt einen GRUB-Bootloader,

der einen Linux-Kernel startet und eine Busybox-Umgebung bereitstellt. Dort werden vordefinierte Installationsquellen abgefragt und im Erfolgsfall beginnt die Installation. Die Suchfunktion durchstöbert den lokalen USB-Stick und gräbt anschließend neugierig im Netzwerk durch HTTP- und FTP-Server. Sobald ein passendes Installationsimage gefunden ist, endet die Suche und der Installer übernimmt die Leitung.

Vorbereitung

Damit dieser scheinbar einfache Plan Erfolg haben kann, muss der USB-Stick korrekt vorbereitet sein. Dazu gehören das passende Dateisystem und das Installationsimage mit einem Dateinamen, den die Suchfunktion erwartet. Die Installation über das Netzwerk erfordert serverseitige Vorbereitung und eignet sich für die massenhafte Installation von Switches. Dazu liefert Kapitel 17 die nötigen Handgriffe.
Für kleine Umgebungen ist der USB-Stick die schnellere Variante. Dieser benötigt das Dateisystem VFAT oder EXT3. Beim Dateinamen ist ONIE tolerant und probiert verschiedene Kombinationen aus Plattform, Hersteller, Modell und Architektur aus. Im einfachsten Fall heißt die Datei *onie-installer*.

Die Vorbereitung des USB-Sticks für eine lokale Installation von OpenSwitch zeigen die folgenden Kommandos.
Der Inhalt des USB-Sticks wird dadurch überschrieben.

```
arted --script /dev/sdb mkpart primary 2048s 100%
kfs.ext3 /dev/sdb1
ount /dev/sdb1 /mnt
p PKGS_OPX-3.2.0-installer-x86_64.bin /mnt/onie-installer
mount /mnt
```

Der USB-Stick wird in diesem Beispiel als `sdb` bereitgestellt. Das eigene Linux-System könnte auch eine andere Gerätedatei verwenden, z. B. `sda` oder `sdc`. Unbedingt vorab mit `dmesg` oder `fdisk -l` validieren.

Die `parted`-Kommandos beginnen das Fundament in Form einer Partition, die den gesamten USB-Stick einnimmt. Anschließend wird in Zeile 2 das Dateisystem gegossen. Wenn das erfolgreich war, lässt sich das Ergebnis in Zeile 3 in den lokalen Dateibaum einhängen und beschreiben. Danach macht sich `cp` ans Werk und setzt das Installationsimage oben drauf. Kollege `umount` schließt in Zeile 5 den Rohbau ab und übergibt den USB-Stick schlüsselfertig.

Installation

Der Bootloader von ONIE meldet sich nur beim Systemstart, also muss der Switch erst mal neu gebootet werden. Einblick in den Bootloader geben die Switches über einen Konsolenanschluss. Und der USB-Stick muss natürlich auch seinen Weg in einen freien USB-Anschluss des Geräts finden. Die Maßnahmen machen deutlich, dass diese Form der Installation für große Umgebungen zu aufwendig ist.

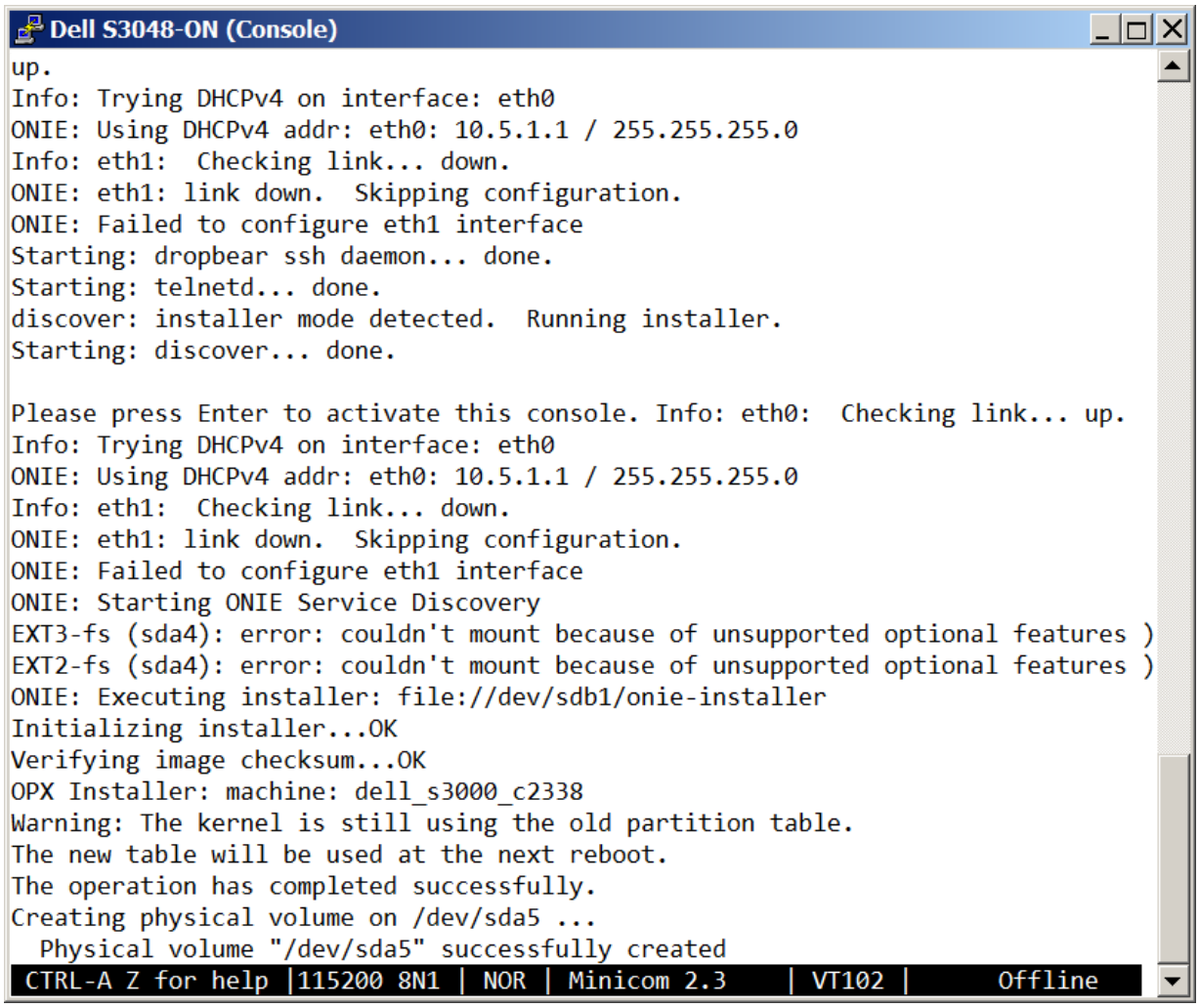

Abbildung 2.1: Bootloader und ONIE installieren OpenSwitch vom USB-Stick

Nach einem Neustart präsentiert sich der Bootloader des vorhandenen Betriebssystems. Neben den normalen Bootmethoden sollte ein Eintrag mit *ONIE* beschriftet sein. Dieser führt zum Auswahlmenü des ONIE-Bootloaders. Neben der Installation sind auch die Deinstallation, Updates und eine Recovery-Option verfügbar. Der gewünschte Weg führt über den Eintrag *ONIE: Install OS*.

Unmittelbar danach beginnt der Discovery-Prozess mit der Suche nach dem Image (Abbildung 2.1). Das Installationsimage ist eine selbst-extrahierende Datei. Sie

besteht aus ein paar Zeilen Shell-Skript mit Installationsanweisungen, gefolgt von binärem Programmcode für das spätere Betriebssystem. Der anfängliche Skriptcode enthält die hardwarespezifischen Anweisungen, um den Binärcode an der richtigen Stelle zu platzieren.
Danach startet OpenSwitch zum ersten Mal auf der neuen Hardware und begrüßt den Admin mit einem Login-Prompt.

Virtuelle Maschine

Grundsätzlich ist OpenSwitch ein Betriebssystem für physikalische Switches. Aber OpenSwitch kann auch als virtuelle Maschine laufen. In der virtuellen Welt lassen sich damit Features untersuchen und Netzdesigns durchspielen.
Dieses Kapitel kann kein Fachbuch über VirtualBox oder VMware ersetzen! Die Installation der VMs setzt Grundwissen in den jeweiligen Produkten voraus. Die Beschreibungen behandeln nur den Aufbau der neuen VM und nicht, warum die einzelnen Schritte notwendig oder vorteilhaft sind.

VirtualBox

VirtualBox ist eine Applikation für Windows, Linux und macOS, mit der virtuelle Maschinen erstellt und gehostet werden.
Es gibt mehrere Konfigurationsmethoden: Bei der normalen Installation ist der *Oracle VM VirtualBox Manager* mit im Boot. Er ist leicht zu bedienen, verlangt aber nach einer X11-Oberfläche. Alternativ (oder ergänzend) hilft *phpVirtualBox* [3], ein webbasierter Manager, der das Look-and-Feel des Oracle-Managers als Webseite bereitstellt.
Fans der Kommandozeile bekommen ebenfalls etwas geboten, denn der Laboraufbau lässt sich unter VirtualBox

komplett skripten.

Das Download-Portal von OpenSwitch bietet leider kein Maschinen-Template (OVA) oder eine fertige VM an. Stattdessen gibt es ein Installationskommando, welches eine neue VM erstellt, bootet, installiert und zuletzt Konsolenzugriff anbietet. Fertig ist die OpenSwitch-VM.
Das folgende Beispiel installiert OpenSwitch in VirtualBox unter CentOS, wobei auch Windows und macOS unterstützt werden. Der Linux-Installer `lvm` benötigt die Binärdatei für OpenSwitch und die Installations-CD von ONIE.

```
wget https://archive.openswitch.net/installers/3.2.0/Dell-EMC/
\
  PKGS_OPX-3.2.0-installer-x86_64.bin
wget https://archive.openswitch.net/onie/onie-kvm_x86_64-r0.iso
wget http://archive.openswitch.net/vm-tools/lvm
chmod +x lvm
./lvm create openswitch --iso onie-kvm_x86_64-r0.iso \
  --bin PKGS_OPX-3.2.0-installer-x86_64.bin
```

Erst in der letzten Zeile beginnt `lvm` mit der Arbeit. Im VirtualBox-Manager taucht plötzlich eine neue VM auf, die scheinbar selbstständig ihr Betriebssystem installiert (Abbildung 2.2). In der Kommandozeile berichtet `lvm` neutral über seinen Fortschritt:

```
OS-installer file: PKGS_OPX-3.2.0-installer-x86_64.bin
ONIE-Recovery file: onie-kvm_x86_64-r0.iso
Deleting VM name: "openswitch"
Initial boot in
progress.....................................................OK
ONIE is self-
embedding.....................................................OK
Wait for VM to
boot.....................................................OK
ONIE stop
discovery.....................................................OK
Load OS-installer. This operation may take a few minutes.
Please wait..OK
Installing
```

```
OS.....................................................OK
Configure
OS.....................................................OK
Final Restart of
VM.....................................................OK
Setup complete! You can now connect using: ssh -p 2222
opxUser@127.0.0.1
```

Falls mehrere OPX-Switches benötigt werden, lässt sich die fertige VM klonen, oder der Installer `lvm` wird erneut gestartet.

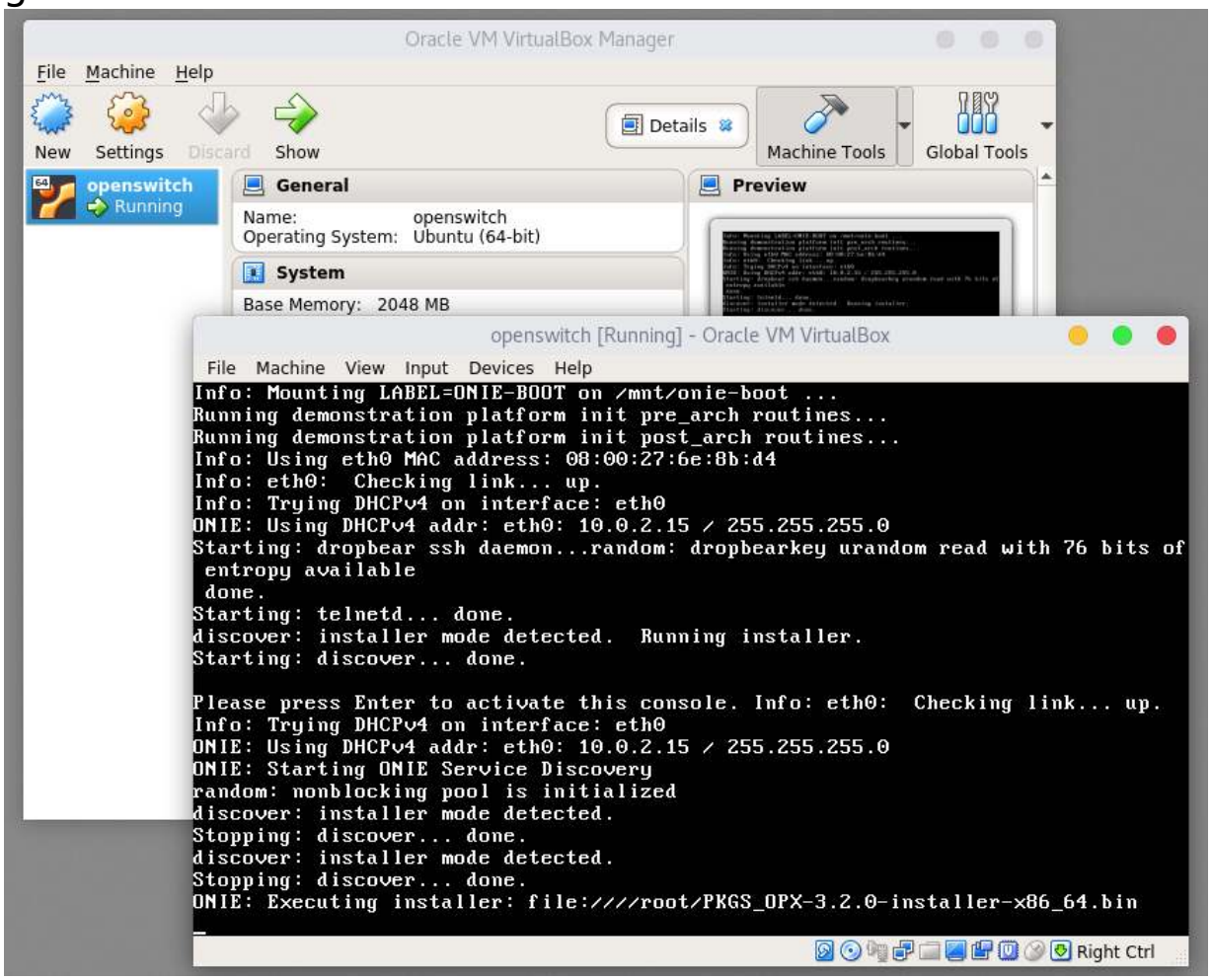

Abbildung 2.2: Das Werkzeug lvm installiert OpenSwitch in VirtualBox

Verbindungskabel

Die einfachste Methode zum Nachbilden eines Verbindungskabels ist der virtuelle Netzwerkadapter vom Typ *Internes Netzwerk*. Dieses Netz muss nicht vorab angelegt werden, sondern entsteht mit der Angabe eines Namens. Sobald zwei Netzadapter in einem internen Netz mit demselben Namen sind, haben sie eine Verbindung miteinander.
Vorteilhaft sind hier sprechende Namen, die das virtuelle "Kabel" gut beschreiben. Die beispielhafte Bezeichnung *kabel-sw01-sw12* kündigt eine Verbindung zwischen Switch sw01 und sw12 an.
OpenSwitch hat verschiedene Features, die im Hostsystem von VirtualBox den promiskuitiven Modus erfordern. Ein Netzadapter im *promiscuous mode* kann mehrere MAC-Adressen haben. Ohne diesen Modus wird beispielsweise die Netzbrücke in OPX keine Daten transportieren.
Der zweite Netzadapter vom Demo-Switch sw13 wechselt den Modus auf der Kommandozeile von VirtualBox:

```
VBoxManage modifyvm sw13 --nicpromisc2 allow-all
```

VMware

Offiziell unterstützt OpenSwitch nur VirtualBox, sodass die Installation unter VMware nur mit Umwegen möglich ist. Als Grundlage benötigt VMware die Festplattendatei, die nach der Installation unter VirtualBox auf der lokalen Festplatte liegt. Freundlicherweise kann VirtualBox über die Kommandozeile die HDD-Datei in das VMware-eigene Format umwandeln:

```
vboxmanage clonehd openswitch.vdi openswitch.vmdk --format vmdk
```

Mit der resultierenden Datei `openswitch.vmdk` lässt sich eine VM in VMware erstellen und starten.
Die Produktpalette von VMware ist groß, aber für das Lab eignen sich hauptsächlich ESXi, Player und Workstation.

In VMware ist die Nummerierung der Netzadapter bei OpenSwitch nicht einheitlich.
Damit kann es vorkommen, dass der *dritte* konfigurierte Netzadapter in der Verwaltungsoberfläche von VMware Workstation in OpenSwitch als *e101-008-0* auftaucht. Aus diesem Grund eignet sich VMware nur bedingt für die Virtualisierung von OpenSwitch. Falls ein Produkt von VMware dennoch für die Virtualisierung eingesetzt wird, sind die folgenden Hinweise zu beachten.

ESXi

In VMware ESXi wird die Festplattendatei beim Start der VM eventuell mit der Fehlermeldung "Unsupported or invalid disk type 2" zurückgewiesen. Der ungültige Plattentyp lässt sich mit Bordmitteln auf der Kommandozeile von ESXi korrigieren:

```
vmkfstools -i openswitch.vmdk openswitch-esxi.vmdk
```

OpenSwitch ist kein *vSwitch* oder *Distributed Switch*.
Aus der Sicht von VMware ESXi ist OpenSwitch eine reguläre virtuelle Maschine mit vielen Netzadaptern. Die Verbindung von zwei Netzadaptern verschiedener OPX-Switches erfolgt über einen *Virtuellen Standard-Switch* (vSwitch) oder den *Distributed Switch*, den ESXi bereitstellt. Innerhalb des vSwitch entstehen Portgruppen, welche die Netzadapter der OPX-Switches gegeneinander abschotten. Dieses Durcheinander von Switches und Portgruppen ist eine Besonderheit von ESXi und eignet sich lediglich für eine Laborumgebung.

VMware-Tools

Die Linux-Distribution von OpenSwitch hat ein Softwarepaket für die VMware-Tools im Repository, welches

für den Betrieb als virtuelle Maschine stark empfohlen wird. Der Paketmanager holt die Tools und startet sie:

```
apt install open-vm-tools
```

Labor

Ein einzelner Switch ohne umgebendes Netzwerk ist wenig beeindruckend. Für den praxisnahen Einstieg gesellen sich die OPX-Switches in einem konstruierten Labornetz zum Leben. In dieser Umgebung kann OpenSwitch Kapitel für Kapitel mit seinen Fähigkeiten glänzen.
Die folgenden Kapitel basieren alle auf demselben Netzaufbau. Es stellt ein kleines Rechenzentrumsnetz mit redundanten Verbindungen dar. Je nach Komplexität eines Themas reicht ein Teil des Labornetzwerks aus, um die Kernaussage zu beschreiben.
Wenn ein Abschnitt einen gesonderten Aufbau benötigt oder ein weiteres Gerät untersucht werden soll, gibt es am Anfang der Lektion einen entsprechenden Hinweis mit Erklärung.

Ressourcen

Der stets unveränderte Aufbau des Labornetzes hat den charmanten Vorteil, dass zwischen den Kapiteln nicht umgebaut werden muss. Kein Umverkabeln der Geräte oder Umkonfigurieren der virtuellen Umgebung. Das spart Zeit und verhindert Fehler. Und nach ein paar Kapiteln wird das Labornetz zum vertrauten Begleiter, denn die Namen der Switches, Server, Netzschnittstellen und IP-Adressen bleiben unverändert.
Das vollständige Labornetz ist als Netzdiagramm in Abbildung 2.3 dargestellt. Es ist als Grundlage für die nachfolgenden Kapitel konzipiert und orientiert sich an der Spine-Leaf-Architektur (vgl. Kap. 13).

Da ein händischer Eingriff nach dem ersten Aufbau nicht mehr notwendig ist, kann das Lab auch "aus der Ferne" betrieben werden - Remotezugriff vorausgesetzt.
Die erforderliche Hardware ist stets abhängig vom geplanten Durchsatz. Für die Laborgeräte eignet sich jedes Gerät von der Kompatibilitätsliste [4]. Für virtuelle Umgebungen stellt OpenSwitch ein Installationsskript für VirtualBox zur Verfügung.
Manche Kapitel arbeiten isoliert, andere benötigen Internetzugriff. Der Zugang zum Internet läuft stets über das Managementinterface *eth0*. Hier reicht ein Uplink zum DSL-Router, aber grundsätzlich ist alles möglich, was letztendlich ins Internet führt.

Virtualisierung

Alle Geräte im Lab können vollständig virtualisiert werden. Jeder Switch im Labornetz ist dann eine eigene virtuelle Maschine (VM) mit virtuellen Netzwerkkabeln zu den benachbarten VMs. Das Verbindungsnetz zwischen zwei VMs ist ein *internes Netzwerk* (bei VirtualBox) oder *LAN Segment* (bei VMware). Eine physikalische Netzwerkkarte im Hostsystem ist nötig, wenn mit echter Hardware gemischt wird.
Welches Interface in welchem virtuellen Netz Zuhause ist zeigt Tabelle 2.1. Getestet und geprüft sind die Labs mit VirtualBox 5.2.

Hardware

OpenSwitch läuft auf physikalischen Switches mit x86_64-Prozessor. Bei der Auswahl von passender Hardware lohnt sich ein Blick in die Kompatibilitätsmatrix von OpenSwitch [4]. Für die Laborumgebung sind Hersteller und Modell zweitrangig, da das Beispielnetz Verständnis bieten soll und nicht Höchstleistung.

Die folgenden Kapitel verwenden Switches vom Hersteller Dell.

Netze

Die Netze zwischen den Switches basieren auf Ethernet. Jede Verbindung zwischen zwei Geräten besteht aus einem Kabel ohne weitere Teilnehmer. Das Übertragungsmedium und die Bandbreite spielen keine Rolle. Kupferkabel sind ebenso willkommen wie Glasfaserkabel.
Im Beispielnetz sind alle Netzverbindungen Kupferkabel mit einer Übertragungsrate von einem Gbit/s.

Switches

Die OPX-Switches verwenden die aktuelle Version 3.2.0. Wenn andere Versionen oder zusätzliche Switches mitspielen, wird das entsprechende Gerät ersetzt oder das Lab ergänzt.
Die beste Version eines Betriebssystems ist nicht immer die Aktuellste. Die Release Notes und ein Softwaretest in einer realistischen Umgebung liefern Entscheidungsgrundlagen für eine stabile Version.
Jeder Switch hat einen separaten Netzanschluss für den Managementzugriff. Darüber erreicht der SSH-Client sein Ziel und kann Konfigurationsänderungen unabhängig von der Topologie umsetzen.
Die Labor-Switches sind durchnummeriert. Diese Geräte-Nummer findet sich in den IPv4- und IPv6-Adressen wieder. Damit sind Adressen in einer Kommandoausgabe leichter dem passenden Gerät zuzuordnen. Die Nummer des Switchports ist stets am Gerätesymbol angeschlagen.

Adressierung

Die Subnetze für Server und Transitbereiche bauen auf private IPv4-Adressen bzw. Unique-Local IPv6-Adressen. Jeder Leaf-Switch verbindet mehrere Server, die meist nur zum Prüfen von Features oder zum Erzeugen von Datenverkehr benutzt werden. Mehr als ping, traceroute, netstat oder ein Webbrowser wird nicht gefordert.
Die Wahl des Betriebssystems der Server ist für die Szenarien nebensächlich; im Demo-Lab finden aus Popularitätsgründen VMware ESXi, Linux und Windows Server Verwendung.
Wenn das Labornetz zwischen internen und externen Netzbereichen unterscheidet, dann bedienen sich die externen Geräte aus den Adressblöcken für Dokumentation (RFC 5737): 192.0.2.0/24 und 198.51.100.0/24.
Die IPv6-Adressen stammen ebenfalls aus mehreren Bereichen, um eine Unterscheidung optisch zu vereinfachen: fd00::/16 für die internen LANs und 2001:db8::/32 für die äußeren Bereiche. Diese Unterscheidung soll die verschiedenen Netzbereiche verdeutlichen - funktionell ist sie nicht nötig.
Die Adressen sind genau dafür vorgesehen und kollidieren nicht mit einem öffentlichen Bereich. Weiterhin ist die Adressierung bewusst einfach gehalten: Die Adressbereiche sind einheitlich strukturiert und haben nur "normale" Netzmasken von /24 (IPv4) oder /64 (IPv6).

Labor-Server

Alle zentralen Funktionen übernimmt der Labor-Server, der ebenfalls physikalisch oder virtuell integriert wird. Wenn die OPX-Switches auf ein Client/Server-Protokoll getestet werden, übernimmt der Labserver stets die Rolle des Gegenstücks. Er akzeptiert Anfragen zu NTP, DNS, Syslog, FTP/TFTP, sFlow und HTTP. Die Kommandobeispiele in den folgenden Kapiteln beziehen sich auf Debian 10.

Zusammenfassung

Mit dem vorgestellten Labornetz lassen sich die Features von OpenSwitch und die Themen im Buch praktisch nachbauen. Dabei sind die verwendeten Geräte, Adressen und der Netzaufbau nur ein Beispiel, welches sich beliebig variieren lässt.
Das beschriebene Netzdesign stellt für die Server redundante Switches bereit. Die Switches sind durch weitere Switches im Backbone verbunden. Der Aufbau entspricht dem Spine-Leaf-Prinzip (vgl. Kap. 13) und ist die praktische Basis für die weiteren Kapitel.

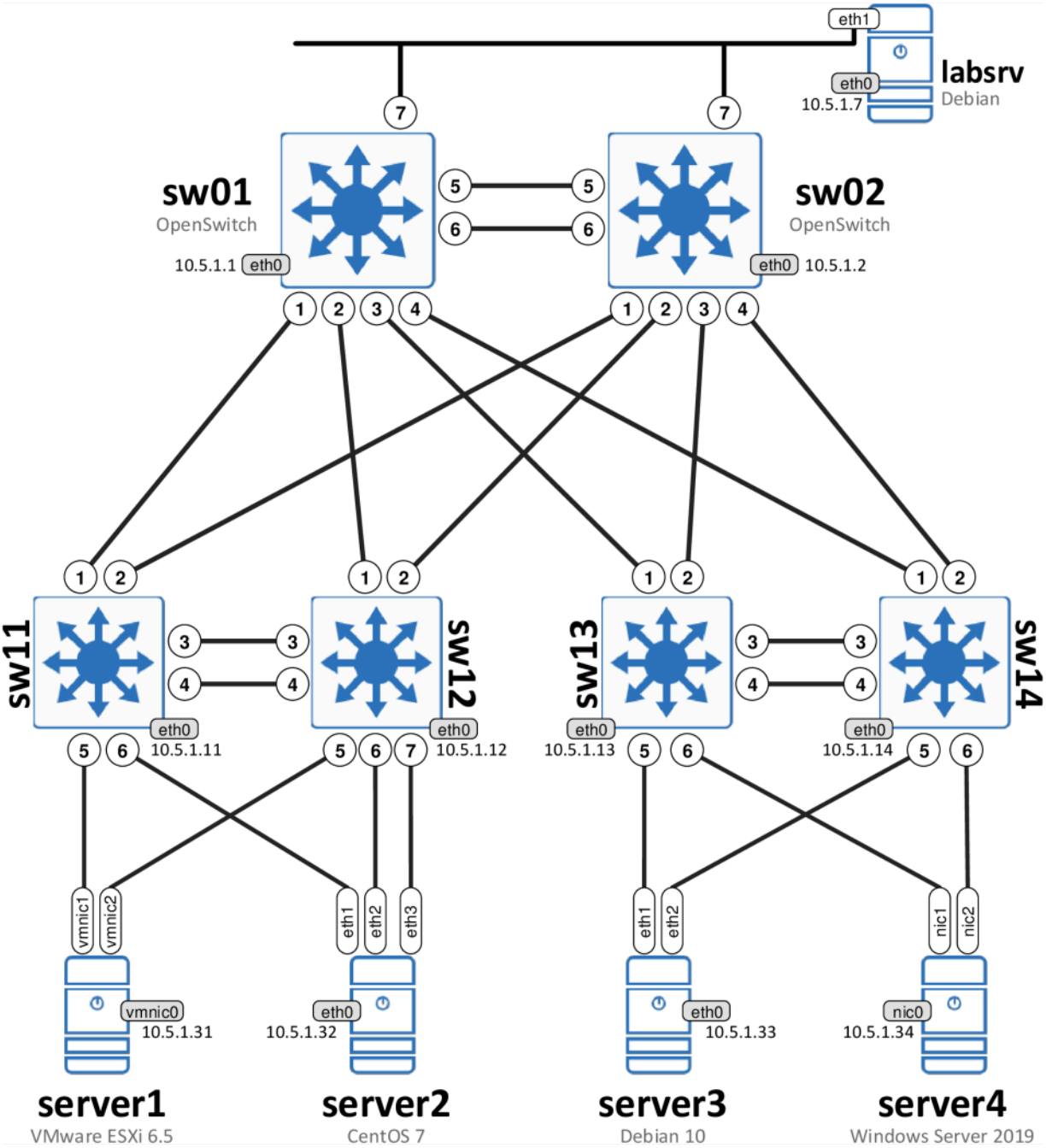

Abbildung 2.3: Das Labornetzwerk als Vorlage für alle Kapitel

Tabelle 2.1: Alle Geräte mit Netzadaptern, Funktion und Managementadresse

Gerät	**Interface**	**Funktion/Netz**	**IPv4**	**IPv6**
sw01	eth0	Management	10.5.1.1	fd00:5::1

Gerät	Interface	Funktion/Netz	IPv4	IPv6
	e101-001-4	Spine-Leaf		
	e101-005,6	Spine-Spine		
	e101-007	vnet4	10.4.1.1	fd00:4::1
sw02	eth0	Management	10.5.1.2	fd00:5::2
	e101-001-4	Spine-Leaf		
	e101-005,6	Spine-Spine		
	e101-007	vnet4	10.4.1.2	fd00:4::2
sw11	eth0	Management	10.5.1.11	fd00:5::11
	e101-001,2	Leaf-Spine		
	e101-003,4	Leaf-Leaf		
	e101-005,6	Leaf-Server		
sw12	eth0	Management	10.5.1.12	fd00:5::12
	e101-001,2	Leaf-Spine		
	e101-003,4	Leaf-Leaf		
	e101-005-7	Leaf-Server		
sw13	eth0	Management	10.5.1.13	fd00:5::13
	e101-001,2	Leaf-Spine		
	e101-003,4	Leaf-Leaf		
	e101-005,6	Leaf-Server		
sw14	eth0	Management	10.5.1.14	fd00:5::14
	e101-001,2	Leaf-Spine		
	e101-003,4	Leaf-Leaf		
	e101-005,6	Leaf-Server		
labsrv	eth0	Management	10.5.1.7	fd00:5::7
	eth1	vnet4	10.4.1.7	fd00:4::7