SDN ONOS Produktion Installation

Wolfgang Hurst

1 Einleitung

1.1 Dieses Dokument und das Ziel

In diesem Dokument geht es darum eine produktive ONOS Installation auf einem Server zu erstellen.

In diesem Dokument geht es nicht darum wie man ONOS konfiguriert oder bedient. Sondern nur um die Installation.

Die produktive Variante hat den Vorteil dass sie dediziert auf einem Server (oder VM) läuft.

Als Spielwiese – so wie die Developer Version – ist diese Version ehr ungeeignet.

1.2 Voraussetzungen

Für die Installation sind zwei neue Maschinen nötig.

Der ONOS-Server - 8 GB Ram und 8 GB Festplatte. Hostname `sdn-onos-prod-01`.

Das Build-System - 8 GB Ram und 16 GB Festplatte. Hostname `sdn-onos-build-01`.

Als Referenz dient mein Dokument „*Installation Ubuntu 1604 Server*“.

Weiterhin wäre es schön wenn die Mininet Umgebung mit dem WCORD rennt.

Siehe Dokument „*SDN Mininet Topo WCORD*“

1.3 Einleitung

Möchte man ein ONOS in einer Produktionsumgebung betreiben, dabei ist es nicht wichtig ob es sich dabei um ein Testsystem oder ein Livesystem handelt, dann bekommt man mit einer Developer Installation Probleme. Sie ist nicht wirklich dafür geeignet.

Ein ONOS Produktionssystem besteht aus zwei Teilen. Ein Teil ist das sogenannte Build-System. Mit dem Build-System wird ein ONOS von den Quellen übersetzt und alle nötigen Teile werden auf dem Build-System zusammengesetzt. Das Ergebnis davon ist später ein etwa 100-300 MB großes TAR Archiv (Image) mit allen benötigten Teilen.

Dieses Archiv wird dann auf den zweiten Teil kopiert. Das ist unser ONOS-Server. Das Build-System kopiert das Archiv (Image) auf den Server, passt das System mit Start Scripts und Konfigurationen entsprechend an, so dass dieser ONOS-Server vollkommen eigenständig funktioniert.

Hat man auf dem Build-System ein solches Image liegen, kann man dieses vom Build-System gleich auf mehrere Server schieben. Der Vorteil dabei ist, dass auf allen Maschinen nachher exakt das gleiche ONOS verwendet wird. Damit lassen sich zum Beispiel Massenrollouts bewerkstelligen.

Wir benötigen also zwei virtuelle Maschinen, eine für das Build-System und eine für den ONOS-Server. Dabei ist zu beachten, das Build System sollte über mindestens 4 GB Ram verfügen. Es gibt zwei drei Codes die übersetzt werden, wo er tatsächlich sehr viel Speicher braucht. Eine VM mit nur 1 GB Ram und paar MB Swap knallt genau dort weg und man kommt nicht weiter.

2 ONOS-Server

2.1 Konfiguration

Wir installieren erst einmal den Server mit 8 GB Ram und 8 GB Festplatte.

Auf dem Server benötigen wir die JAVA Umgebung, Python und ZIP. Mehr ist dort nicht notwendig. Plus ein paar nützliche Tools

```
apt -y install mc vim sysstat wget curl iftop screen
apt -y install java-common java-wrappers default-jdk default-jre openjfx
apt -y install python zip
```

Hier das Protokoll

```
root@sdn-onos-prod-01:~# apt -y install mc vim sysstat wget curl iftop screen
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
»screen« ist bereits die neuste Version (4.3.1-2build1).
»iftop« ist bereits die neuste Version (1.0~pre4-3).
»mc« ist bereits die neuste Version (3:4.8.15-2).
»curl« ist bereits die neuste Version (7.47.0-1ubuntu2.2).
»sysstat« ist bereits die neuste Version (11.2.0-1ubuntu0.1).
»vim« ist bereits die neuste Version (2:7.4.1689-3ubuntu1.2).
»wget« ist bereits die neuste Version (1.17.1-1ubuntu1.1).
0 aktualisiert, 0 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.
root@sdn-onos-prod-01:~# apt -y install java-common java-wrappers default-jdk default-jre openjfx
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig

... ... blabla ... ...

openjfx (8u60-b27-4) wird eingerichtet ...
Trigger für libc-bin (2.23-0ubuntu7) werden verarbeitet ...
Trigger für systemd (229-4ubuntu16) werden verarbeitet ...
Trigger für ureadahead (0.100.0-19) werden verarbeitet ...
Trigger für ca-certificates (20160104ubuntu1) werden verarbeitet ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...

done.
done.
root@sdn-onos-prod-01:~# apt -y install python zip
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
»python« ist bereits die neuste Version (2.7.11-1).
python wurde als manuell installiert festgelegt.
Die folgenden NEUEN Pakete werden installiert:
  zip
0 aktualisiert, 1 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.
Es müssen 158 kB an Archiven heruntergeladen werden.
Nach dieser Operation werden 587 kB Plattenplatz zusätzlich benutzt.
Holen:1 http://de.archive.ubuntu.com/ubuntu xenial/main amd64 zip amd64 3.0-11 [158 kB]
Es wurden 158 kB in 0 s geholt (202 kB/s).
Vormals nicht ausgewähltes Paket zip wird gewählt.
(Lese Datenbank ... 99667 Dateien und Verzeichnisse sind derzeit installiert.)
Vorbereitung zum Entpacken von .../archives/zip_3.0-11_amd64.deb ...
Entpacken von zip (3.0-11) ...
Trigger für man-db (2.7.5-1) werden verarbeitet ...
zip (3.0-11) wird eingerichtet ...
root@sdn-onos-prod-01:~#
```

2.2 SDN User

Es wird ein Benutzer benötigt. Das Build-System benutzt diesen Benutzer um das Archiv per SCP zu kopieren. Weiterhin wird dieser Benutzer gebraucht um die Software zu installieren. Per Default in den ganzen ONOS Einstellungen ist das der Benutzer mit dem Namen `sdn`. Das Heimatverzeichnis des Users sollte unter `/home` sein. Die Software wird später unter `/opt` installiert, das darf nicht kollidieren, weil sonst knallt es.

```
useradd -c "ONOS Production" -d /home/sdn -m -s /bin/bash sdn
passwd sdn
```

Als Password habe ich `sdn` vergeben. Das Build-System benutzt diesen Benutzer aber auch um im Startvorgang entsprechende Module zu installieren. Der User muss also unbedingt Root werden dürfen. Und da die Scripte mehrfach Aktionen ausführen verwende ich `NOPASSWD` damit ich nicht ständig das Password eingeben muss.

```
echo 'sdn ALL=(ALL) NOPASSWD: ALL' >/etc/sudoers.d/sdn
```

Da das ganze über SSH mit Public-Key funktioniert, loggen wir uns jetzt kurz als `sdn` User ein und erstellen schon einmal ein SSH-Key

```
su - sdn
ssh-keygen -t rsa
```

Und wir testen als `sdn` User gleich mal ob wir ohne Password `root` werden dürfen.

```
sudo -s
```

Hier das Protokoll

```
root@sdn-onos-prod-01:~# useradd -c "ONOS Production" -d /home/sdn -m -s /bin/bash sdn
root@sdn-onos-prod-01:~# passwd sdn
Geben Sie ein neues UNIX-Passwort ein:
Geben Sie das neue UNIX-Passwort erneut ein:
passwd: password updated successfully
root@sdn-onos-prod-01:~# echo 'sdn ALL=(ALL) NOPASSWD: ALL' >/etc/sudoers.d/sdn
root@sdn-onos-prod-01:~# su - sdn
sdn@sdn-onos-prod-01:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/sdn/.ssh/id_rsa):
Created directory '/home/sdn/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/sdn/.ssh/id_rsa.
Your public key has been saved in /home/sdn/.ssh/id_rsa.pub.
.. bla ..
sdn@sdn-onos-prod-01:~$ sudo -s
root@sdn-onos-prod-01:~#
```

3 Build-System

3.1 Konfiguration

Wir installieren erst einmal den Server mit 8 GB Ram und 16 GB Festplatte.

Auf dem Server benötigen wir ein paar Pakete.

```
apt -y install mc vim sysstat wget curl iftop screen
apt -y install git git-core
apt -y install java-common java-wrappers default-jdk default-jre openjfx
### !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! apt -y install
maven maven-ant-helper maven-debian-helper
### !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!! apt –y install
libproperties-maven-plugin-java-doc libproperties-maven-plugin-java
apt -y install python zip
```

Hier das Protokoll

```
root@sdn-onos-build-01:~# apt -y install mc vim sysstat wget curl iftop screen
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
»screen« ist bereits die neuste Version (4.3.1-2build1).
»iftop« ist bereits die neuste Version (1.0~pre4-3).
»mc« ist bereits die neuste Version (3:4.8.15-2).
»curl« ist bereits die neuste Version (7.47.0-1ubuntu2.2).
»sysstat« ist bereits die neuste Version (11.2.0-1ubuntu0.1).
»vim« ist bereits die neuste Version (2:7.4.1689-3ubuntu1.2).
»wget« ist bereits die neuste Version (1.17.1-1ubuntu1.1).
0 aktualisiert, 0 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.
root@sdn-onos-build-01:~# apt -y install git git-core
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
»git« ist bereits die neuste Version (1:2.7.4-0ubuntu1).
Die folgenden NEUEN Pakete werden installiert:
  git-core
0 aktualisiert, 1 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.
Es müssen 1.464 B an Archiven heruntergeladen werden.
Nach dieser Operation werden 8.192 B Plattenplatz zusätzlich benutzt.
Holen:1 http://de.archive.ubuntu.com/ubuntu xenial/main amd64 git-core all 1:2.7.4-0ubuntu1 [1.464 B]
Es wurden 1.464 B in 0 s geholt (5.666 B/s).
Vormals nicht ausgewähltes Paket git-core wird gewählt.
(Lese Datenbank ... 96220 Dateien und Verzeichnisse sind derzeit installiert.)
Vorbereitung zum Entpacken von .../git-core_1%3a2.7.4-0ubuntu1_all.deb ...
Entpacken von git-core (1:2.7.4-0ubuntu1) ...
git-core (1:2.7.4-0ubuntu1) wird eingerichtet ...
root@sdn-onos-build-01:~# apt -y install java-common java-wrappers default-jdk default-jre openjfx
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
.. blabla ..
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...

done.
done.
root@sdn-onos-build-01:~# apt -y install python zip
Paketlisten werden gelesen... Fertig
Abhängigkeitsbaum wird aufgebaut.
Statusinformationen werden eingelesen.... Fertig
»python« ist bereits die neuste Version (2.7.11-1).
python wurde als manuell installiert festgelegt.
Die folgenden NEUEN Pakete werden installiert:
  zip
0 aktualisiert, 1 neu installiert, 0 zu entfernen und 0 nicht aktualisiert.
Es müssen 158 kB an Archiven heruntergeladen werden.
Nach dieser Operation werden 587 kB Plattenplatz zusätzlich benutzt.
Holen:1 http://de.archive.ubuntu.com/ubuntu xenial/main amd64 zip amd64 3.0-11 [158 kB]
Es wurden 158 kB in 0 s geholt (0 B/s).
Vormals nicht ausgewähltes Paket zip wird gewählt.
(Lese Datenbank ... 99668 Dateien und Verzeichnisse sind derzeit installiert.)
Vorbereitung zum Entpacken von .../archives/zip_3.0-11_amd64.deb ...
Entpacken von zip (3.0-11) ...
Trigger für man-db (2.7.5-1) werden verarbeitet ...
zip (3.0-11) wird eingerichtet ...
root@sdn-onos-build-01:~#
```

3.2 ONOS User

Für die Übersetzung der Quellen benötigen wir einen Benutzer. In den ganzen Quellen wird der Benutzer `onos` verwendet. Den nehmen wir auch. Das Password setze ich auf `rocks`. Das ist Default in den ganzen Konfigurationsdateien.

```
useradd -c "ONOS Build User" -d /home/onos -m -s /bin/bash onos
passwd onos
```

Danach loggen wir uns als Benutzer `onos` ein und arbeiten jetzt nur noch mit diesem Benutzer. Weiterhin konfigurieren wir noch unseren Lieblings Editor

```
su - onos
select-editor
```

Hier das Protokoll

```
root@sdn-onos-build-01:~# useradd -c "ONOS Build User" -d /home/onos -m -s /bin/bash onos
root@sdn-onos-build-01:~# passwd onos
Geben Sie ein neues UNIX-Passwort ein:
Geben Sie das neue UNIX-Passwort erneut ein:
passwd: password updated successfully
root@sdn-onos-build-01:~# su - onos
onos@sdn-onos-build-01:~$ select-editor

Select an editor.  To change later, run 'select-editor'.
  1. /bin/ed
  2. /bin/nano        <---- easiest
  3. /usr/bin/mcedit
  4. /usr/bin/vim.basic
  5. /usr/bin/vim.tiny

Choose 1-5 [2]: 4
onos@sdn-onos-build-01:~$
```

3.3 SSH Key

Damit der `onos` Benutzer vom Build-System auf den Benutzer `sdn` des ONOS-Servers per SSH kommen kann, wird jetzt ein SSH Key benötigt, den wir auf den ONOS-Server kopieren müssen. Weiterhin müssen wir einmal manuell ein SSH zum Server machen, um die Abfrage des SSH Keys zu bestätigen. Tut man das nicht, geht die Installation schief.

```
ssh-keygen -t rsa

cat .ssh/id_rsa.pub

ssh sdn@sdn-onos-prod-01
cat >>.ssh/authorized_keys
### mit der maus in der konsole den key kopieren
### und CTRL-D drücken

# ssh vom Build-System zum sdn ONOS-Server testen (muss ohne password gehen)
exit
ssh sdn@sdn-onos-prod-01
```

Hier das Protokoll

```
onos@sdn-onos-build-01:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/onos/.ssh/id_rsa):
Created directory '/home/onos/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/onos/.ssh/id_rsa.
Your public key has been saved in /home/onos/.ssh/id_rsa.pub.
.. bla ..
onos@sdn-onos-build-01:~$ cat .ssh/id_rsa.pub
ssh-rsa AAAA-XXX-.../ onos@sdn-onos-build-01
onos@sdn-onos-build-01:~$ ssh sdn@sdn-onos-prod-01
The authenticity of host 'sdn-onos-prod-01 (10.0.0.196)' can't be established.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'sdn-onos-prod-01,10.0.0.196' (ECDSA) to the list of known hosts.
sdn@sdn-onos-prod-01's password:
sdn@sdn-onos-prod-01:~$ cat >>.ssh/authorized_keys
ssh-rsa AAAA-XXX-.../ onos@sdn-onos-build-01
sdn@sdn-onos-prod-01:~$ exit
Abgemeldet
Connection to sdn-onos-prod-01 closed.
onos@sdn-onos-build-01:~$ ssh sdn@sdn-onos-prod-01
sdn@sdn-onos-prod-01:~$ exit
```

3.4 Quellen holen

Wir holen uns den Quellcode vom GIT und checken die Version aus die wir brauchen

```
git clone https://gerrit.onosproject.org/onos
cd onos
git checkout tags/1.9.0
```

Hier das Protokoll

```
onos@sdn-onos-build-01:~$ git clone https://gerrit.onosproject.org/onos
Klone nach 'onos' ...
remote: Counting objects: 74054, done
remote: Finding sources: 100% (57589/57589)
remote: Total 295817 (delta 21608), reused 287285 (delta 21608)
Empfange Objekte: 100% (295817/295817), 97.58 MiB | 217.00 KiB/s, Fertig.
Löse Unterschiede auf: 100% (110530/110530), Fertig.
Prüfe Konnektivität ... Fertig.
onos@sdn-onos-build-01:~$ cd onos
onos@sdn-onos-build-01:~/onos$ git tag
1.0.0
1.0.1
1.1.0
1.1.0-rc1
1.1.0-rc2
1.2.0
1.2.0-rc1
1.2.0-rc2
1.2.1
1.2.2
1.3.0
1.3.0-rc1
1.3.0-rc2
1.3.0-rc3
1.4.0
1.4.0-rc1
1.4.0-rc2
1.4.0-rc3
1.5.0
1.5.0-rc2
1.5.0-rc3
1.5.1
1.6.0
1.6.0-rc1
1.6.0-rc2
1.6.0-rc3
1.6.0-rc4
1.6.0-rc5
1.7.0
1.7.0-rc1
1.7.0-rc2
1.7.0-rc3
1.7.1
1.8.0
1.8.0-rc1
1.8.0-rc3
1.8.0-rc4
1.8.0-rc5
1.8.0-rc6
1.8.1
1.8.2
1.8.3
1.8.4
1.9.0
1.9.0-b1b
1.9.0-b3
1.9.0-rc1
1.9.0-rc2
1.9.0-rc3
1.9.0-rc4
1.9.0-rc5
1.9.0-rc6
onos@sdn-onos-build-01:~/onos$ git checkout tags/1.9.0
Note: checking out 'tags/1.9.0'.

You are in 'detached HEAD' state. You can look around, make experimental
changes and commit them, and you can discard any commits you make in this
state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may
do so (now or later) by using -b with the checkout command again. Example:

  git checkout -b <new-branch-name>

HEAD ist jetzt bei 1bbaaaa... Tagging 1.9.0
onos@sdn-onos-build-01:~/onos$
```

3.5 Environment einrichten

Im Quellcode liegt eine Environment Datei die wir benutzen sollten. Dieses Script setzt eine ganze Handvoll von Umgebungsvariablen und Funktionen und Aliase die wir brauchen. Wir lesen das Environment in unserer `.bashrc` einfach mit ein und loggen uns danach neu ein, damit das Environment greift.

```
cd

echo '' >>.bashrc
echo '# ONOS environment' >>.bashrc
echo '. ~/onos/tools/dev/bash_profile' >>.bashrc

exit
su - onos
```

Hier das Protokoll

```
onos@sdn-onos-build-01:~/onos$ cd
onos@sdn-onos-build-01:~$ echo '' >>.bashrc
onos@sdn-onos-build-01:~$ echo '# ONOS environment' >>.bashrc
onos@sdn-onos-build-01:~$ echo '. ~/onos/tools/dev/bash_profile' >>.bashrc
onos@sdn-onos-build-01:~$ exit
Abgemeldet
root@sdn-onos-build-01:~# su - onos
onos@sdn-onos-build-01:~$ set | grep ^ONOS_
ONOS_APPS=drivers,openflow,fwd,proxyarp,mobility
ONOS_CELL=local
ONOS_DEVICES=' of:0000000000000001 of:0000000000000002 of:0000000000000003 of:0000000000000004
of:0000000000000005 of:0000000000000006 of:0000000000000007 of:0000000000000008 of:0000000000000009
of:000000000000000a of:000000000000000b of:000000000000000c of:000000000000000d of:000000000000000e
of:000000000000000f of:0000000000000010 of:0000000000000011 of:0000000000000012 of:0000000000000013
of:0000000000000014 of:0000000000000015 of:0000000000000016 of:0000000000000017 of:0000000000000018
of:0000000000000019'
ONOS_GROUP=sdn
ONOS_HOSTS=' 00:00:00:00:00:01/-1 00:00:00:00:00:02/-1 00:00:00:00:00:03/-1 00:00:00:00:00:04/-1
00:00:00:00:00:05/-1 00:00:00:00:00:06/-1 00:00:00:00:00:07/-1 00:00:00:00:00:08/-1
00:00:00:00:00:09/-1 00:00:00:00:00:0a/-1 00:00:00:00:00:0b/-1 00:00:00:00:00:0c/-1
00:00:00:00:00:0d/-1 00:00:00:00:00:0e/-1 00:00:00:00:00:0f/-1 00:00:00:00:00:10/-1
00:00:00:00:00:11/-1 00:00:00:00:00:12/-1 00:00:00:00:00:13/-1 00:00:00:00:00:14/-1
00:00:00:00:00:15/-1 00:00:00:00:00:16/-1 00:00:00:00:00:17/-1 00:00:00:00:00:18/-1
00:00:00:00:00:19/-1'
ONOS_INSTANCES=$'192.168.56.101\n192.168.56.102'
ONOS_NIC='192.168.56.*'
ONOS_ROOT=/home/onos/onos
ONOS_SCENARIOS=/home/onos/onos/tools/test/scenarios
ONOS_TOPO=default
ONOS_USER=sdn
ONOS_WEB_PASS=rocks
ONOS_WEB_USER=onos
onos@sdn-onos-build-01:~$
```

3.6 ONOS Cell einrichten

ONOS arbeitet mit sogenannten Cells. Das sind effektiv kleine Scripte die Environment Variablen setzen um eine Umgebung zu definieren. Unter anderem stehen dort IP Adressen anderer ONOS Server drin, wenn man ONOS im Cluster betrieben will. Per Default ist die Cell `local` aktiviert. Dort stehen aber IP Adressen drin die man im Normalfall nicht hat. Man kann jetzt hingehen und die `local` Cell ändern, das ist aber eine ganz dumme Idee, spätestens bei einem Update wäre das verloren. Wir werden also unsere eigene Cell erstellen.

Um eine Cell zu erstellen kann man das Kommando `vicell` verwenden. Das startet den vi Editor und man kann fröhlich losklimpern, was wir auch tun. Der Name unserer Cell ist `sdn-onos-prod-01`

```
vicell -c -a sdn-onos-prod-01
i
# die ip addresse unseres produktiv onos system
export OC1="10.0.0.196"

# die default adresse fuer die kommnikation
export OCI="10.0.0.196"

# diese applikationen automatisch starten
export ONOS_APPS="drivers,openflow,fwd,proxyarp,mobility"

# die ip adresse unserer mininet umgebung (sofern vorhanden) [ sdn-mininet-01 ]
export OCN="10.0.0.193"

# bei cluster systemen kommt hier ein wildcard aller ip adressen der ONOS instanzen rein
# der parameter wird ignoriert, wenn kein Cluster konfiguriert ist
export ONOS_NIC="10.0.0.*"

:wq
```

Das `vicell` konfiguriert jetzt auch diese Zelle als Default. Jedoch ist diese Einstellung beim nächsten Login wieder weg. Wir müssen das also beim Login in unserer `.bashrc` mit hinzufügen

```
echo '' >>.bashrc
echo '# unserer ONOS cell setzten' >>.bashrc
echo 'ONOS_CELL=sdn-onos-prod-01' >>.bashrc
echo 'export ONOS_CELL' >>.bashrc
```

Nach dem Re-Login kann man das mit `onos-cell` überprüfen.

3.6.1 Hack

Wegen eines Fehlers irgendwo in den Quellen, bezüglich der Zellen, müssen wir das ONOS_CELL in der `.bashrc` zusätzlich noch VOR dem ONOS Environment setzen. Und weil ich damit kein Erfolg hatte, habe ich das `~/onos/tools/dev/bash_profile` editiert und mir beim Update gemerkt das ich darin rumgefummelt habe. Da sucht man ONOS_CELL und schreibt unserer rein.

Hier das Protokoll

```
onos@sdn-onos-build-01:~$ vicell -c -a sdn-onos-prod-01
i
# die ip addresse unseres produktiv onos system
export OC1="10.0.0.196"

# die default adresse fuer die kommnikation
export OCI="10.0.0.196"

# diese applikationen automatisch starten
export ONOS_APPS="drivers,openflow,fwd,proxyarp,mobility"

# die ip adresse unserer mininet umgebung (sofern vorhanden) [ sdn-mininet-01 ]
export OCN="10.0.0.193"

# bei cluster systemen kommt hier ein wildcard aller ip adressen der ONOS instanzen rein
# der parameter wird ignoriert, wenn kein Cluster konfiguriert ist
export ONOS_NIC="10.0.0.*"

<ESC>:wq
ONOS_CELL=sdn-onos-prod-01
OCI=10.0.0.196
OC1=10.0.0.196
OCN=10.0.0.193
ONOS_APPS=drivers,openflow,fwd,proxyarp,mobility
ONOS_GROUP=sdn
ONOS_NIC=10.0.0.*
ONOS_SCENARIOS=/home/onos/onos/tools/test/scenarios
ONOS_TOPO=default
ONOS_USER=sdn
ONOS_WEB_PASS=rocks
ONOS_WEB_USER=onos
onos@sdn-onos-build-01:~$ find . -name sdn-onos-prod-01\*
./onos/tools/test/cells/sdn-onos-prod-01
onos@sdn-onos-build-01:~$ cat ./onos/tools/test/cells/sdn-onos-prod-01
# die ip addresse unseres produktiv onos system
export OC1="10.0.0.196"

# die default adresse fuer die kommnikation
export OCI="10.0.0.196"

# diese applikationen automatisch starten
export ONOS_APPS="drivers,openflow,fwd,proxyarp,mobility"

# die ip adresse unserer mininet umgebung (sofern vorhanden) [ sdn-mininet-01 ]
export OCN="10.0.0.193"

# bei cluster systemen kommt hier ein wildcard aller ip adressen der ONOS instanzen rein
# der parameter wird ignoriert, wenn kein Cluster konfiguriert ist
export ONOS_NIC="10.0.0.*"

onos@sdn-onos-build-01:~$ echo '' >>.bashrc
onos@sdn-onos-build-01:~$ echo '# unserer ONOS cell setzten' >>.bashrc
onos@sdn-onos-build-01:~$ echo 'ONOS_CELL=sdn-onos-prod-01' >>.bashrc
onos@sdn-onos-build-01:~$ echo 'export ONOS_CELL' >>.bashrc
onos@sdn-onos-build-01:~$ exit
Abgemeldet
root@sdn-onos-build-01:~# su - onos
onos@sdn-onos-build-01:~$ onos-cell
ONOS_CELL=sdn-onos-prod-01
OCI=10.0.0.196
OC1=10.0.0.196
OCN=10.0.0.193
ONOS_APPS=drivers,openflow,fwd,proxyarp,mobility
ONOS_GROUP=sdn
ONOS_NIC=10.0.0.*
ONOS_SCENARIOS=/home/onos/onos/tools/test/scenarios
ONOS_TOPO=default
ONOS_USER=sdn
ONOS_WEB_PASS=rocks
ONOS_WEB_USER=onos
onos@sdn-onos-build-01:~$
```

3.7 Package erstellen

Nachdem nun alles soweit vorbereitet ist, erstellen wir dieses Image (Archiv). Dazu verwenden wir nicht `maven`, so wie es überall steht. Das `maven` brauchen wir effektiv nicht mehr. Wir verwenden das Buildsystem `buck`. Das `buck` kommt von so einer kleinen Garagenklitsche aus den USA und ist relativ schnell. Man weiß zwar nicht genau wie es funktioniert, aber am Ende zählt nur das Ergebnis ... haha

Mit dem Kommando `onos-package`, wird der `buck` Build Prozess angeworfen, alles wird Mundgerecht für den Server abgelegt. Der Vorgang dauert eine Weile.

Man sollte, sofern man ein schnelles Internet hat, immer das `buck-out` Verzeichnis löschen, damit löscht man alle temporären Dateien, runtergeladene Pakete und sonstigen Kram. Wenn ein Build schief geht, sollte man es immer löschen.

```
cd ~/onos
rm -rf buck-out
onos-package
```

Hier das Protokoll

```
onos@sdn-onos-build-01:~$ cd ~/onos
onos@sdn-onos-build-01:~/onos$ rm -rf buck-out
onos@sdn-onos-build-01:~/onos$ onos-package
Updating Buck...
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 43.8M  100 43.8M    0     0   207k      0  0:03:36  0:03:36 --:--:--  217k
Archive:  cache/buck-v2016.12.02.01.zip
  inflating: buck
 extracting: .buck_version
   creating: plugins/
  inflating: plugins/onos.jar
  inflating: plugins/yang.jar
Successfully updated Buck in /home/onos/onos/bin/buck to buck-v2016.12.02.01.zip

Not using buckd because NO_BUCKD is set.
[-] PROCESSING BUCK FILES...FINISHED 1,4s [100%] 🎬  New buck daemon
[+] DOWNLOADING... (0,00 B/S, TOTAL: 0,00 B, 0 Artifacts)
[+] BUILDING...8m49,5s [99%] (811/812 JOBS, 811 UPDATED, 811 [99,9%] CACHE MISS)
 |=> IDLE
 |=> //tools/package:onos-package...  8,1s (running dir_artifact_store[0,1s])
The outputs are:
//tools/package:onos-package buck-out/gen/tools/package/onos-package/onos.tar.gz
-rw-rw-r-- 1 onos onos 141349366 Mär 25 20:04 /tmp/onos-1.9.0.onos.tar.gz
1871588672 141349366 /tmp/onos-1.9.0.onos.tar.gz
onos@sdn-onos-build-01:~/onos$
```

3.8 Installation

Nach dem das Package auf dem Build-System bereit steht, kann man es nun beliebig oft auf beliebig vielen ONOS Servern installieren.

Hier ist aber Vorsicht angesagt. Ein ONOS-Server der eine IP Adresse hat, die nicht in der Cell Konfiguration ist, fährt nicht hoch! Das muss man dann manuell ändern, siehe dazu das Kapitel cluster.json auf Seite 16

Die Installation übernimmt das Programm `onos-install`. Das rufen wir mit der Option `-f` auf, diese Option sorgt dafür dass eine alte Installation deinstalliert wird.

```
cd ~/onos
onos-install -f sdn-onos-prod-01
```

Das `onos-install` startet auch noch den ONOS auf dem Server automatisch. Nach einer kleinen Weile kann man darauf zugreifen.

Achtung. Das die OCI Variable wird irgendwo in dem Wust von Skripten wieder auf die `local` Cell gestellt. Grund unbekannt. Gibt man es nicht an kommt es zur Fehlermeldung das der Server die IP nicht kennt.

Hier das Protokoll

```
onos@sdn-onos-build-01:~/onos$ cd ~/onos
onos@sdn-onos-build-01:~/onos$ OCI=10.0.0.196 onos-install -f sdn-onos-prod-01
-rw-rw-r-- 1 onos onos 141349366 Mär 25 20:04 /tmp/onos-1.9.0.onos.tar.gz
1871588672 141349366 /tmp/onos-1.9.0.onos.tar.gz
Connection to sdn-onos-prod-01 closed.
Using scp
Connection to sdn-onos-prod-01 closed.
Connection to sdn-onos-prod-01 closed.
Connection to sdn-onos-prod-01 closed.
onos@sdn-onos-build-01:~/onos$ ssh sdn@sdn-onos-prod-01
sdn@sdn-onos-prod-01:~$ ps ax | grep java
10217 ?        Sl     0:09 /usr/bin/java -
agentlib:jdwp=transport=dt_socket,server=y,suspend=n,address=5005 -server -Xms128M -Xmx512M -
XX:+UnlockDiagnosticVMOptions -XX:+UnsyncloadClass -Dcom.sun.management.jmxremote -
Djava.endorsed.dirs=/usr/lib/jvm/java-8-openjdk-amd64/jre/jre/lib/endorsed:/usr/lib/jvm/java-8-
openjdk-amd64/jre/lib/endorsed:/opt/onos/apache-karaf-3.0.8/lib/endorsed -
Djava.ext.dirs=/usr/lib/jvm/java-8-openjdk-amd64/jre/jre/lib/ext:/usr/lib/jvm/java-8-openjdk-
amd64/jre/lib/ext:/opt/onos/apache-karaf-3.0.8/lib/ext -Dkaraf.instances=/opt/onos/apache-karaf-
3.0.8/instances -Dkaraf.home=/opt/onos/apache-karaf-3.0.8 -Dkaraf.base=/opt/onos/apache-karaf-3.0.8 -
Dkaraf.data=/opt/onos/apache-karaf-3.0.8/data -Dkaraf.etc=/opt/onos/apache-karaf-3.0.8/etc -
Djava.io.tmpdir=/opt/onos/apache-karaf-3.0.8/data/tmp -
Djava.util.logging.config.file=/opt/onos/apache-karaf-3.0.8/etc/java.util.logging.properties -
Dkaraf.startLocalConsole=true -Dkaraf.startRemoteShell=true -classpath /opt/onos/apache-karaf-
3.0.8/lib/karaf-jaas-boot.jar:/opt/onos/apache-karaf-3.0.8/lib/karaf-
org.osgi.core.jar:/opt/onos/apache-karaf-3.0.8/lib/karaf.jar org.apache.karaf.main.Main
10463 pts/1    S+     0:00 grep --color=auto java
sdn@sdn-onos-prod-01:~$
```

3.9 Zugriff auf ONOS

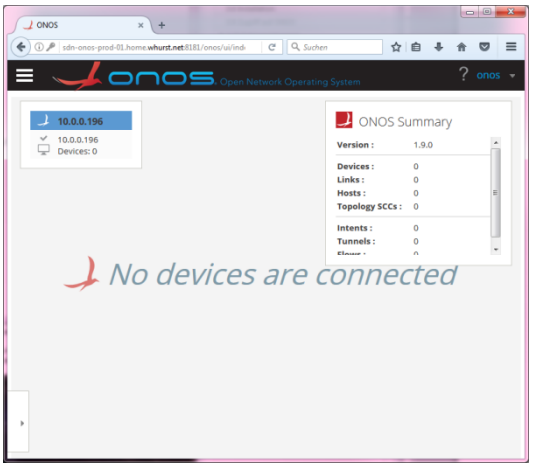
Wenn das alles funktioniert hat, kann man nun auf den ONOS zugreifen. Einmal mit einem Webbrowser und einmal mit SSH.

Der Benutzername ist `onos` und das Password ist `rocks`. Das wird aus dem Environment gelesen, oder kann auch in einer Cell Konfiguration geändert werden.

```
# webbrowser
http://sdn-onos-prod-01:8181/onos/ui/login.html

# console
onos sdn-onos-prod-01
```

Hier das Protokoll

```
onos@sdn-onos-build-01:~/onos$ onos sdn-onos-prod-01
Warning: Permanently added '[sdn-onos-prod-01]:8101,[10.0.0.196]:8101' (RSA) to the list of known
hosts.
Password authentication
Password:
Welcome to Open Network Operating System (ONOS)!
    ____  _  ______  ____
   / __ \/ |/ / __ \/ __/
  / /_/ /    / /_/ /\ \
  \____/_/|_/\____/___/

Documentation: wiki.onosproject.org
Tutorials:     tutorials.onosproject.org
Mailing lists: lists.onosproject.org

Come help out! Find out how at: contribute.onosproject.org

Hit '<tab>' for a list of available commands
and '[cmd] --help' for help on a specific command.
Hit '<ctrl-d>' or type 'system:shutdown' or 'logout' to shutdown ONOS.

onos> apps -a -s
*   5 org.onosproject.drivers              1.9.0    Default device drivers
*   6 org.onosproject.optical-model        1.9.0    Optical information model
*  20 org.onosproject.mobility             1.9.0    Host Mobility App
*  28 org.onosproject.openflow-base        1.9.0    OpenFlow Provider
*  47 org.onosproject.hostprovider         1.9.0    Host Location Provider
*  48 org.onosproject.lldpprovider         1.9.0    LLDP Link Provider
*  49 org.onosproject.openflow             1.9.0    OpenFlow Meta App
*  69 org.onosproject.proxyarp             1.9.0    Proxy ARP/NDP App
*  81 org.onosproject.fwd                  1.9.0    Reactive Forwarding App
onos>
```



3.10 Fehlermeldungen

3.10.1 ONOS GUI not ready yet... please stand by...

Bitte auf den Server einloggen und im `/opt/onos/log/karaf.log` schauen was da los ist. Bekanntes Problem kann zum Beispiel der Fehler `java.lang.IllegalStateException: Unable to determine local ip` sein.

Dann ist irgendwas mit der cluster.json nicht korrekt. Siehe Kapitel cluster.json auf Seite 16

Bitte unbedingt den „Hack“ mit der `ONOS_CELL` beachten.

Siehe Kapitel Environment einrichten auf Seite 9

4 ONOS Tuning

4.1 Konfiguration

Die ONOS Installation auf dem Server befindet sich unter `/opt/onos`. Dort findet man auch ein Verzeichnis mit dem Namen `config`. In diesem Verzeichnis kann man Startup Konfigurationen ablegen die ONOS benutzen soll wenn er hochfährt.

4.2 cluster.json

Die `cluster.json` sagt dem ONOS welche IP Adresse und Ports er verwenden soll. Wenn man mehrere ONOS Server mit einem Build installiert, hätten alle ONOS Instanzen in dieser Datei die gleichen IP Adressen konfiguriert. Die muss man dann hier ändern damit der ONOS starten kann.

Bei der Installation wird die konfigurierte Cell konvertiert und auf dem Build-System wird eine temporäre `cluster.json` erstellt. Zum Beispiel unter `/tmp/sdn@sdn-onos-prod-01.cluster.json` liegt die Datei die später auf das Produktionssystem unter `/onos/config/cluster.json` kopiert wird.

Wer also die cluster.json bearbeitet und vom Build-System eine neue Installation macht, darf damit rechnen dass die Datei dann weg ist.

4.3 network-cfg.json

Die `network-cfg.json` gibt es anfänglich nicht. Die `network-cfg.json` ist eine Konfiguration in Form von NETCFG. In der Dokumentation zum WCORD haben wir im Mininet Script eine NETCFG erstellt, die wir per REST API dem ONOS ge-`curl`-t haben. Genau dieses JSON kann man nun hier als `network-cfg.json` erstellen. Natürlich ohne den `curl` Aufruf, nur das reine JSON. Wenn ONOS dann hochfährt, lädt er automatisch das NETCFG und unser Netzwerk ist in ONOS bekannt.

Hat man eine Konfiguration erstellt und ONOS neu gestartet kann man sich das Ergebnis mittels `netcfg` auf der ONOS Konsole bestätigen lassen.

```
$ onos sdn-onos-prod-01
> netcfg
```

4.4 Autostartup

Wenn man den ONOS Server rebootet, wird das ONOS nicht gestartet. Das `onos-install` vom Build-System jedoch installiert ein `systemctl` Modul. Damit ONOS beim Booten gestartet wird muss man dieses Modul einfach nur noch aktivieren, mehr nicht.

```
systemctl enable onos
```

4.5 Build-System

Das Build-System kann nun runtergefahren werden. Solange man nicht ein neues Images installieren und bauen will, braucht man diese Maschine nicht mehr.

4.6 Integration WCORD

Wir erst einmal auf die WCORD Mininet Installation und beenden das laufende Mininet.

Wir ändern die IP Adresse (es muss die IP sein – kein Hostname) des ONOS Servers.

Wir starten das Mininet in einem sehr großen Terminalfenster und markieren und den NETCFG Teil.

Danach gehen wir auf den ONOS-Server und stoppen diesen

Wenn es ruhig geworden ist, erstellen wir dort eine network-cfg.json

Wir kopieren unseren Test in die Datei und speichern das

Danach starten wir ONOS wieder

Protokoll Mininet (den Teil den wir brauchen ist unterstrichen)

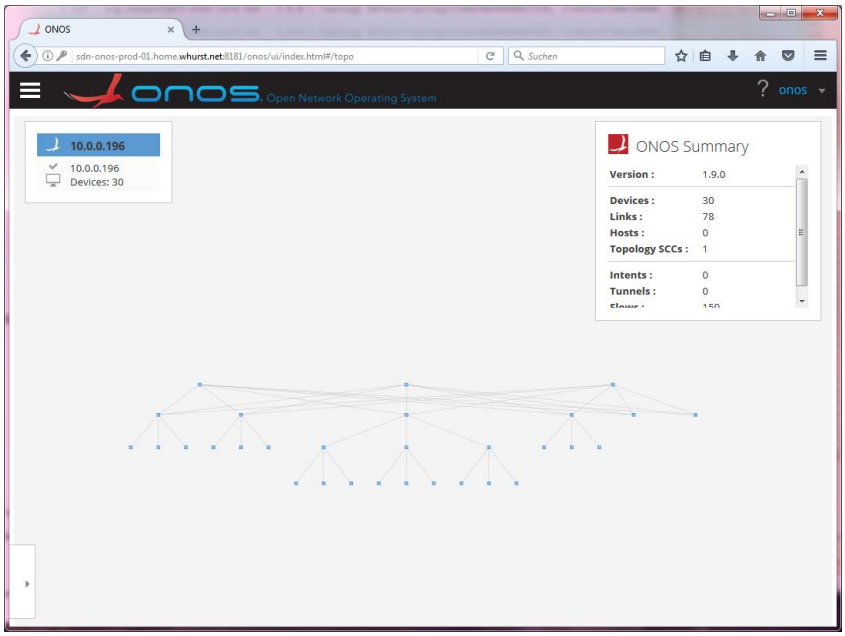
```
whurst@sdn-mininet-01:~$ screen -r
mininet> exit
*** Stopping 1 controllers
c0
*** Stopping 112 links
.....
*** Stopping 30 switches
leaf1 leaf2 leaf3 leaf4 leaf5 leaf6 menodeb1 menodeb2 menodeb3 rdslam1 rdslam2 rdslam3 rgf11 rgf12 rgf13 rgf21 rgf22 rgf23 rgf31 rgf32 rgf33 rolt1 rolt2 rolt3 roltgf1 roltgf2
roltgf3 spine1 spine2 spine3
*** Stopping 73 hosts
cp11 cp12 cp13 cp21 cp22 cp23 cp31 cp32 cp33 cr11 cr12 cr21 cr22 cr31 cr32 ds111 ds112 ds113 ds121 ds122 ds123 ds131 ds132 ds133 dslgf111 dslgf112 dslgf113 dslgf121 dslgf122
dslgf123 dslgf131 dslgf132 dslgf133 dslgf211 dslgf212 dslgf213 dslgf221 dslgf222 dslgf223 dslgf231 dslgf232 dslgf233 dslgf311 dslgf312 dslgf313 dslgf321 dslgf322 dslgf323
dslgf331 dslgf332 dslgf333 ecpe11 ecpe12 ecpe13 ont11 ont12 ont13 ont21 ont22 ont23 ont31 ont32 ont33 saaa11 saaa21 sbng11 sbng21 scache11 scache21 sdhcp11 sdhcp21 sdns11 sdns21
*** Done
completed in 1158825.557 seconds
whurst@sdn-mininet-01:~$ sudo mn --custom=wcord.py --topo=WCORD --controller=remote,ip=10.0.0.196
curl --user onos:rocks -X POST -H "Content-Type: application/json" http://sdn-onos-dev-
01:8181/onos/v1/network/configuration/ -d '
{
"devices" : {
"of:0000000000001011" : { "basic": { "name":"SPINE 1", "latitude":-20, "longitude":26 } },"of:0000000000001012" : { "basic": { "name":"SPINE 2", "latitude":-20, "longitude":86 }
},"of:0000000000001013" : { "basic": { "name":"SPINE 3", "latitude":-20, "longitude":146 } },"of:0000000000002021" : { "basic": { "name":"LEAF 1", "latitude":-28, "longitude":14
} },"of:0000000000002022" : { "basic": { "name":"LEAF 2", "latitude":-28, "longitude":38 } },"of:0000000000002023" : { "basic": { "name":"LEAF 3", "latitude":-28, "longitude":86
} },"of:0000000000002024" : { "basic": { "name":"LEAF 4", "latitude":-28, "longitude":134 } },"of:0000000000002025" : { "basic": { "name":"LEAF 5", "latitude":-28,
"longitude":152 } },"of:0000000000002026" : { "basic": { "name":"LEAF 6", "latitude":-28, "longitude":170 } },"of:0000000000021011" : { "basic": { "name":"DSLAM 1", "latitude":-
36, "longitude":6 } },"of:0000000000021012" : { "basic": { "name":"DSLAM 2", "latitude":-36, "longitude":14 } },"of:0000000000021013" : { "basic": { "name":"DSLAM 3",
"latitude":-36, "longitude":22 } },"of:0000000000021021" : { "basic": { "name":"OLT 1", "latitude":-36, "longitude":30 } },"of:0000000000021022" : { "basic": { "name":"OLT 2",
"latitude":-36, "longitude":38 } },"of:0000000000021023" : { "basic": { "name":"OLT 3", "latitude":-36, "longitude":46 } },"of:0000000000021031" : { "basic": { "name":"OLT GF
1", "latitude":-36, "longitude":62 } },"of:0000000000021032" : { "basic": { "name":"OLT GF 2", "latitude":-36, "longitude":86 } },"of:0000000000021033" : { "basic": {
"name":"OLT GF 3", "latitude":-36, "longitude":110 } },"of:0000000000031011" : { "basic": { "name":"GF 11", "latitude":-44, "longitude":54 } },"of:0000000000031012" : { "basic":
{ "name":"GF 12", "latitude":-44, "longitude":62 } },"of:0000000000031013" : { "basic": { "name":"GF 13", "latitude":-44, "longitude":70 } },"of:0000000000031021" : { "basic": {
"name":"GF 21", "latitude":-44, "longitude":78 } },"of:0000000000031022" : { "basic": { "name":"GF 22", "latitude":-44, "longitude":86 } },"of:0000000000031023" : { "basic": {
"name":"GF 23", "latitude":-44, "longitude":94 } },"of:0000000000031031" : { "basic": { "name":"GF 33", "latitude":-44, "longitude":102 } },"of:0000000000031032" : { "basic": {
"name":"GF 33", "latitude":-44, "longitude":110 } },"of:0000000000031033" : { "basic": { "name":"GF 33", "latitude":-44, "longitude":118 } },"of:0000000000041011" : { "basic": {
"name":"eNodeB 1", "latitude":-36, "longitude":126 } },"of:0000000000041012" : { "basic": { "name":"eNodeB 2", "latitude":-36, "longitude":134 } },"of:0000000000041013" : {
"basic": { "name":"eNodeB 3", "latitude":-36, "longitude":142 } },
"of:0000000000000000" : { "basic": { "name":"NOT EXISTENT" } }
},
"hosts" : {
"00:00:00:01:01:69/-1" : { "basic": { "name":"CR 11", "latitude":-12, "longitude":24 } },"00:00:00:02:01:69/-1" : { "basic": { "name":"CR 12", "latitude":-12, "longitude":28 }
},"00:00:00:01:02:69/-1" : { "basic": { "name":"CR 21", "latitude":-12, "longitude":84 } },"00:00:00:02:02:69/-1" : { "basic": { "name":"CR 22", "latitude":-12, "longitude":88 }
},"00:00:00:01:03:69/-1" : { "basic": { "name":"CR 31", "latitude":-12, "longitude":144 } },"00:00:00:02:03:69/-1" : { "basic": { "name":"CR 32", "latitude":-12, "longitude":148
} },"00:00:00:01:01:01/-1" : { "basic": { "name":"ds111", "latitude":-44, "longitude":4 } },"00:00:00:02:01:01/-1" : { "basic": { "name":"ds112", "latitude":-44, "longitude":6 }
},"00:00:00:03:01:01/-1" : { "basic": { "name":"ds113", "latitude":-44, "longitude":8 } },"00:00:00:01:02:01/-1" : { "basic": { "name":"ds121", "latitude":-44, "longitude":12 }
},"00:00:00:02:02:01/-1" : { "basic": { "name":"ds122", "latitude":-44, "longitude":14 } },"00:00:00:03:02:01/-1" : { "basic": { "name":"ds123", "latitude":-44, "longitude":16 }
},"00:00:00:01:03:01/-1" : { "basic": { "name":"ds131", "latitude":-44, "longitude":20 } },"00:00:00:02:03:01/-1" : { "basic": { "name":"ds132", "latitude":-44, "longitude":22 }
},"00:00:00:03:03:01/-1" : { "basic": { "name":"ds133", "latitude":-44, "longitude":24 } },"00:00:00:01:01:02/-1" : { "basic": { "name":"ont11", "latitude":-44, "longitude":28 }
},"00:00:00:02:01:02/-1" : { "basic": { "name":"ont12", "latitude":-44, "longitude":30 } },"00:00:00:03:01:02/-1" : { "basic": { "name":"ont13", "latitude":-44, "longitude":32 }
},"00:00:00:01:02:02/-1" : { "basic": { "name":"ont21", "latitude":-44, "longitude":36 } },"00:00:00:02:02:02/-1" : { "basic": { "name":"ont22", "latitude":-44, "longitude":38 }
},"00:00:00:03:02:02/-1" : { "basic": { "name":"ont23", "latitude":-44, "longitude":40 } },"00:00:00:01:03:02/-1" : { "basic": { "name":"ont31", "latitude":-44, "longitude":44 }
},"00:00:00:02:03:02/-1" : { "basic": { "name":"ont32", "latitude":-44, "longitude":46 } },"00:00:00:03:03:02/-1" : { "basic": { "name":"ont33", "latitude":-44, "longitude":48 }
},"00:01:01:01:01:03/-1" : { "basic": { "name":"dslgf111", "latitude":-52, "longitude":52 } },"00:00:02:01:01:03/-1" : { "basic": { "name":"dslgf112", "latitude":-52,
"longitude":54 } },"00:00:03:01:01:03/-1" : { "basic": { "name":"dslgf113", "latitude":-52, "longitude":56 } },"00:00:01:02:01:03/-1" : { "basic": { "name":"dslgf121",
"latitude":-52, "longitude":60 } },"00:00:02:02:01:03/-1" : { "basic": { "name":"dslgf122", "latitude":-52, "longitude":62 } },"00:00:03:02:01:03/-1" : { "basic": {
"name":"dslgf123", "latitude":-52, "longitude":64 } },"00:00:01:03:01:03/-1" : { "basic": { "name":"dslgf131", "latitude":-52, "longitude":68 } },"00:00:02:03:01:03/-1" : {
"basic": { "name":"dslgf132", "latitude":-52, "longitude":70 } },"00:00:03:03:01:03/-1" : { "basic": { "name":"dslgf133", "latitude":-52, "longitude":72 } },"00:00:01:01:02:03/-
1" : { "basic": { "name":"dslgf211", "latitude":-52, "longitude":76 } },"00:00:02:01:02:03/-1" : { "basic": { "name":"dslgf212", "latitude":-52, "longitude":78 }
},"00:00:03:01:02:03/-1" : { "basic": { "name":"dslgf213", "latitude":-52, "longitude":80 } },"00:00:01:02:02:03/-1" : { "basic": { "name":"dslgf221", "latitude":-52,
"longitude":84 } },"00:00:02:02:02:03/-1" : { "basic": { "name":"dslgf222", "latitude":-52, "longitude":86 } },"00:00:03:02:02:03/-1" : { "basic": { "name":"dslgf223",
"latitude":-52, "longitude":88 } },"00:00:01:03:02:03/-1" : { "basic": { "name":"dslgf231", "latitude":-52, "longitude":92 } },"00:00:02:03:02:03/-1" : { "basic": {
"name":"dslgf232", "latitude":-52, "longitude":94 } },"00:00:03:02:03/-1" : { "basic": { "name":"dslgf233", "latitude":-52, "longitude":96 } },"00:00:01:01:03:03/-1" : {
"basic": { "name":"dslgf311", "latitude":-52, "longitude":100 } },"00:00:02:01:03:03/-1" : { "basic": { "name":"dslgf312", "latitude":-52, "longitude":102 }
},"00:00:03:01:03:03/-1" : { "basic": { "name":"dslgf313", "latitude":-52, "longitude":104 } },"00:00:01:02:03:03/-1" : { "basic": { "name":"dslgf321", "latitude":-52,
"longitude":108 } },"00:00:02:02:03:03/-1" : { "basic": { "name":"dslgf322", "latitude":-52, "longitude":110 } },"00:00:03:02:03:03/-1" : { "basic": { "name":"dslgf323",
"latitude":-52, "longitude":112 } },"00:00:01:03:03:03/-1" : { "basic": { "name":"dslgf331", "latitude":-52, "longitude":116 } },"00:00:02:03:03:03/-1" : { "basic": {
"name":"dslgf332", "latitude":-52, "longitude":118 } },"00:00:03:03:03:03/-1" : { "basic": { "name":"dslgf333", "latitude":-52, "longitude":120 } },"00:00:00:01:01:04/-1" : {
"basic": { "name":"cp11", "latitude":-44, "longitude":124 } },"00:00:00:02:01:04/-1" : { "basic": { "name":"cp12", "latitude":-44, "longitude":126 } },"00:00:00:03:01:04/-1" : {
"basic": { "name":"cp13", "latitude":-44, "longitude":128 } },"00:00:00:01:02:04/-1" : { "basic": { "name":"cp21", "latitude":-44, "longitude":132 } },"00:00:00:02:02:04/-1" : {
"basic": { "name":"cp22", "latitude":-44, "longitude":134 } },"00:00:00:03:02:04/-1" : { "basic": { "name":"cp23", "latitude":-44, "longitude":136 } },"00:00:00:01:03:04/-1" : {
"basic": { "name":"cp31", "latitude":-44, "longitude":140 } },"00:00:00:02:03:04/-1" : { "basic": { "name":"cp32", "latitude":-44, "longitude":142 } },"00:00:00:03:03:04/-1" : {
"basic": { "name":"cp33", "latitude":-44, "longitude":144 } },"00:00:00:01:01:05/-1" : { "basic": { "name":"ecpe11", "latitude":-36, "longitude":148 } },"00:00:00:02:01:05/-1" :
{ "basic": { "name":"ecpe12", "latitude":-36, "longitude":152 } },"00:00:00:03:01:05/-1" : { "basic": { "name":"ecpe13", "latitude":-36, "longitude":156 } },"00:00:00:01:01:06/-
1" : { "basic": { "name":"sbng11", "latitude":-36, "longitude":160 } },"00:00:00:02:01:06/-1" : { "basic": { "name":"sbng21", "latitude":-36, "longitude":162 }
},"00:00:00:01:02:06/-1" : { "basic": { "name":"sdhcp11", "latitude":-44, "longitude":164 } },"00:00:00:02:02:06/-1" : { "basic": { "name":"sdhcp21", "latitude":-44,
"longitude":166 } },"00:00:00:01:03:06/-1" : { "basic": { "name":"sdns11", "latitude":-52, "longitude":168 } },"00:00:00:02:03:06/-1" : { "basic": { "name":"sdns21",
"latitude":-52, "longitude":172 } },"00:00:00:01:04:06/-1" : { "basic": { "name":"saaa11", "latitude":-44, "longitude":174 } },"00:00:00:02:04:06/-1" : { "basic": {
"name":"saaa21", "latitude":-44, "longitude":176 } },"00:00:00:01:05:06/-1" : { "basic": { "name":"scache11", "latitude":-36, "longitude":178 } },"00:00:00:02:05:06/-1" : {
"basic": { "name":"scache21", "latitude":-36, "longitude":180 } },
"00:00:00:00:00:00/-1" : { "basic": { "name":"NOT EXISTENT" } }
}
}
'

*** Creating network
*** Adding controller
*** Adding hosts:
cp11 cp12 cp13 cp21 cp22 cp23 cp31 cp32 cp33 cr11 cr12 cr21 cr22 cr31 cr32 ds111 ds112 ds113 ds121 ds122 ds123 ds131 ds132 ds133 dslgf111 dslgf112 dslgf113 dslgf121 dslgf122
dslgf123 dslgf131 dslgf132 dslgf133 dslgf211 dslgf212 dslgf213 dslgf221 dslgf222 dslgf223 dslgf231 dslgf232 dslgf233 dslgf311 dslgf312 dslgf313 dslgf321 dslgf322 dslgf323
dslgf331 dslgf332 dslgf333 ecpe11 ecpe12 ecpe13 ont11 ont12 ont13 ont21 ont22 ont23 ont31 ont32 ont33 saaa11 saaa21 sbng11 sbng21 scache11 scache21 sdhcp11 sdhcp21 sdns11 sdns21
*** Adding switches:
leaf1 leaf2 leaf3 leaf4 leaf5 leaf6 menodeb1 menodeb2 menodeb3 rdslam1 rdslam2 rdslam3 rgf11 rgf12 rgf13 rgf21 rgf22 rgf23 rgf31 rgf32 rgf33 rolt1 rolt2 rolt3 roltgf1 roltgf2
roltgf3 spine1 spine2 spine3
*** Adding links:
(cp11, menodeb1) (cp12, menodeb1) (cp13, menodeb1) (cp21, menodeb2) (cp22, menodeb2) (cp23, menodeb2) (cp31, menodeb3) (cp32, menodeb3) (cp33, menodeb3) (ds111, rdslam1) (ds112,
rdslam1) (ds113, rdslam1) (ds121, rdslam2) (ds122, rdslam2) (ds123, rdslam2) (ds131, rdslam3) (ds132, rdslam3) (ds133, rdslam3) (dslgf111, rgf11) (dslgf112, rgf11) (dslgf113,
```

```
rgf11) (dslgf121, rgf12) (dslgf122, rgf12) (dslgf123, rgf12) (dslgf131, rgf13) (dslgf132, rgf13) (dslgf133, rgf13) (dslgf211, rgf21) (dslgf212, rgf21) (dslgf213, rgf21)
(dslgf221, rgf22) (dslgf222, rgf22) (dslgf223, rgf22) (dslgf231, rgf23) (dslgf232, rgf23) (dslgf233, rgf23) (dslgf311, rgf31) (dslgf312, rgf31) (dslgf313, rgf31) (dslgf321,
rgf32) (dslgf322, rgf32) (dslgf323, rgf32) (dslgf331, rgf33) (dslgf332, rgf33) (dslgf333, rgf33) (ecpe11, leaf5) (ecpe12, leaf5) (ecpe13, leaf5) (menodeb1, leaf4) (menodeb2,
leaf4) (menodeb3, leaf4) (ont11, rolt1) (ont12, rolt1) (ont13, rolt1) (ont21, rolt2) (ont22, rolt2) (ont23, rolt2) (ont31, rolt3) (ont32, rolt3) (ont33, rolt3) (rdslam1, leaf1)
(rdslam2, leaf1) (rdslam3, leaf1) (rgf11, roltgf1) (rgf12, roltgf1) (rgf13, roltgf1) (rgf21, roltgf2) (rgf22, roltgf2) (rgf23, roltgf2) (rgf31, roltgf3) (rgf32, roltgf3) (rgf33,
roltgf3) (rolt1, leaf2) (rolt2, leaf2) (rolt3, leaf2) (roltgf1, leaf3) (roltgf2, leaf3) (roltgf3, leaf3) (saaa11, leaf6) (saaa21, leaf6) (sbng11, leaf6) (sbng21, leaf6)
(scache11, leaf6) (scache21, leaf6) (sdhcp11, leaf6) (sdhcp21, leaf6) (sdns11, leaf6) (sdns21, leaf6) (spine1, cr11) (spine1, cr12) (spine1, leaf1) (spine1, leaf2) (spine1,
leaf3) (spine1, leaf4) (spine1, leaf5) (spine1, leaf6) (spine2, cr21) (spine2, cr22) (spine2, leaf1) (spine2, leaf2) (spine2, leaf3) (spine2, leaf4) (spine2, leaf5) (spine2,
leaf6) (spine3, cr31) (spine3, cr32) (spine3, leaf1) (spine3, leaf2) (spine3, leaf3) (spine3, leaf4) (spine3, leaf5) (spine3, leaf6)
*** Configuring hosts
cp11 cp12 cp13 cp21 cp22 cp23 cp31 cp32 cp33 cr11 cr12 cr21 cr22 cr31 cr32 dsl11 dsl12 dsl13 dsl21 dsl22 dsl23 dsl31 dsl32 dsl33 dslgf111 dslgf112 dslgf113 dslgf121 dslgf122
dslgf123 dslgf131 dslgf132 dslgf133 dslgf211 dslgf212 dslgf213 dslgf221 dslgf222 dslgf223 dslgf231 dslgf232 dslgf233 dslgf311 dslgf312 dslgf313 dslgf321 dslgf322 dslgf323
dslgf331 dslgf332 dslgf333 ecpe11 ecpe12 ecpe13 ont11 ont12 ont13 ont21 ont22 ont23 ont31 ont32 ont33 saaa11 saaa21 sbng11 sbng21 scache11 scache21 sdhcp11 sdhcp21 sdns11 sdns21
*** Starting controller
c0
*** Starting 30 switches
leaf1 leaf2 leaf3 leaf4 leaf5 leaf6 menodeb1 menodeb2 menodeb3 rdslam1 rdslam2 rdslam3 rgf11 rgf12 rgf13 rgf21 rgf22 rgf23 rgf31 rgf32 rgf33 rolt1 rolt2 rolt3 roltgf1 roltgf2
roltgf3 spine1 spine2 spine3 ...
*** Starting CLI:
mininet>
<<CTRL-A>>D
exit
```

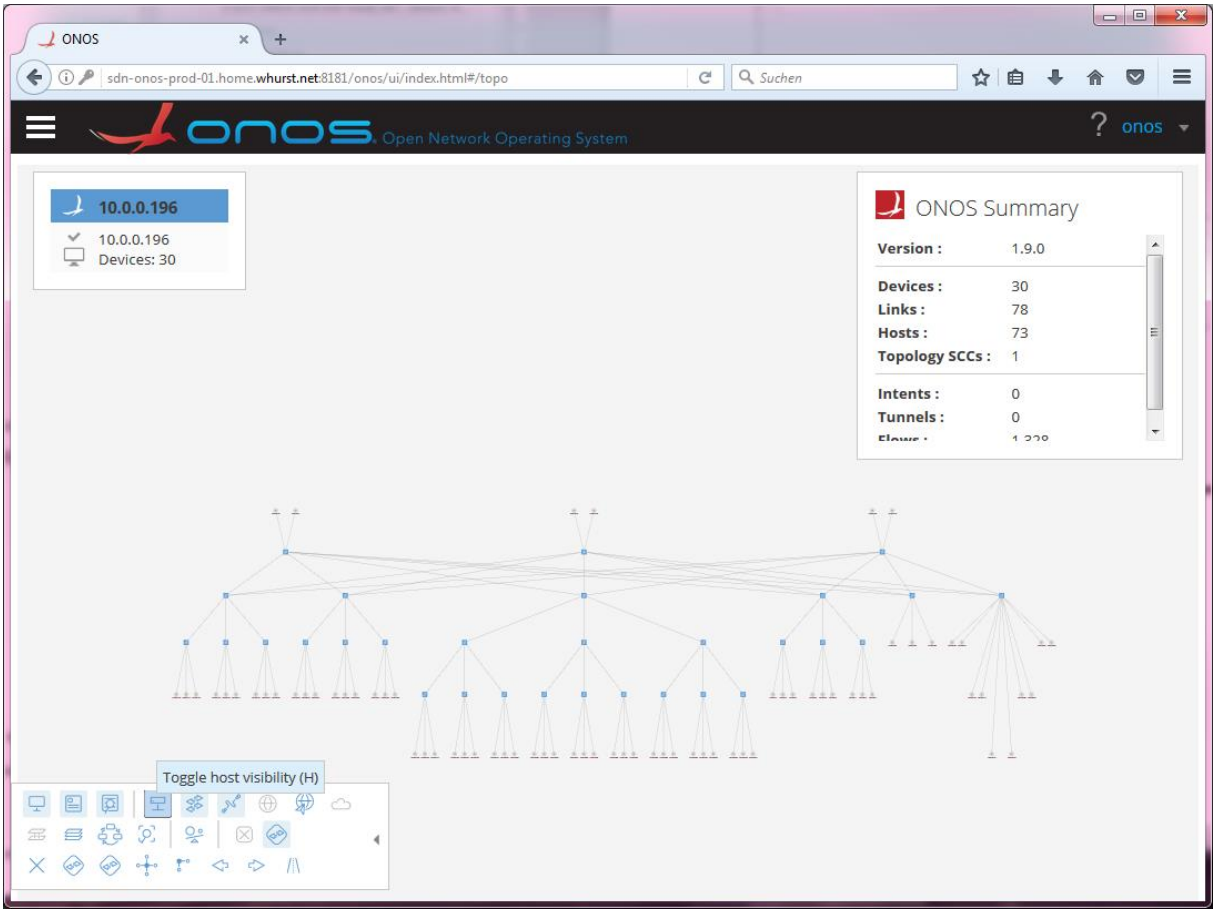
Protokoll ONOS-Server

```
sdn@sdn-onos-prod-01:~$ sudo -s
root@sdn-onos-prod-01:~# systemctl stop onos
root@sdn-onos-prod-01:~# ps ax | grep java
19325 pts/0    S+     0:00 grep --color=auto java
root@sdn-onos-prod-01:~# cat >/opt/onos/config/network-cfg.json
{
"devices" : {
"of:0000000000001011" : { "basic": { "name":"SPINE 1", "latitude":-20, "longitude":26 } },"of:0000000000001012" : { "basic": { "name":"SPINE 2", "latitude":-20, "longitude":86 }
},"of:0000000000001013" : { "basic": { "name":"SPINE 3", "latitude":-20, "longitude":146 } },"of:0000000000002021" : { "basic": { "name":"LEAF 1", "latitude":-28, "longitude":14
} },"of:0000000000002022" : { "basic": { "name":"LEAF 2", "latitude":-28, "longitude":38 } },"of:0000000000002023" : { "basic": { "name":"LEAF 3", "latitude":-28, "longitude":86
} },"of:0000000000002024" : { "basic": { "name":"LEAF 4", "latitude":-28, "longitude":134 } },"of:0000000000002025" : { "basic": { "name":"LEAF 5", "latitude":-28,
"longitude":152 } },"of:0000000000002026" : { "basic": { "name":"LEAF 6", "latitude":-28, "longitude":170 } },"of:0000000000021011" : { "basic": { "name":"DSLAM 1", "latitude":-
36, "longitude":6 } },"of:0000000000021012" : { "basic": { "name":"DSLAM 2", "latitude":-36, "longitude":14 } },"of:0000000000021013" : { "basic": { "name":"DSLAM 3",
"latitude":-36, "longitude":22 } },"of:0000000000021021" : { "basic": { "name":"OLT 1", "latitude":-36, "longitude":30 } },"of:0000000000021022" : { "basic": { "name":"OLT 2",
"latitude":-36, "longitude":38 } },"of:0000000000021023" : { "basic": { "name":"OLT 3", "latitude":-36, "longitude":46 } },"of:0000000000021031" : { "basic": { "name":"OLT GF
1", "latitude":-36, "longitude":62 } },"of:0000000000021032" : { "basic": { "name":"OLT GF 2", "latitude":-36, "longitude":86 } },"of:0000000000021033" : { "basic": {
"name":"OLT GF 3", "latitude":-36, "longitude":110 } },"of:0000000000031011" : { "basic": { "name":"GF 11", "latitude":-44, "longitude":54 } },"of:0000000000031012" : { "basic":
{ "name":"GF 12", "latitude":-44, "longitude":62 } },"of:0000000000031013" : { "basic": { "name":"GF 13", "latitude":-44, "longitude":70 } },"of:0000000000031021" : { "basic": {
"name":"GF 21", "latitude":-44, "longitude":78 } },"of:0000000000031022" : { "basic": { "name":"GF 22", "latitude":-44, "longitude":86 } },"of:0000000000031023" : { "basic": {
"name":"GF 23", "latitude":-44, "longitude":94 } },"of:0000000000031031" : { "basic": { "name":"GF 33", "latitude":-44, "longitude":102 } },"of:0000000000031032" : { "basic": {
"name":"GF 33", "latitude":-44, "longitude":110 } },"of:0000000000031033" : { "basic": { "name":"GF 33", "latitude":-44, "longitude":118 } },"of:0000000000041011" : { "basic": {
"name":"eNodeB 1", "latitude":-36, "longitude":126 } },"of:0000000000041012" : { "basic": { "name":"eNodeB 2", "latitude":-36, "longitude":134 } },"of:0000000000041013" : {
"basic": { "name":"eNodeB 3", "latitude":-36, "longitude":142 } },
"of:0000000000000000" : { "basic": { "name":"NOT_EXISTENT" } }
},
"hosts" : {
"00:00:00:01:01:69/-1" : { "basic": { "name":"CR 11", "latitude":-12, "longitude":24 } },"00:00:00:02:01:69/-1" : { "basic": { "name":"CR 12", "latitude":-12, "longitude":28 }
},"00:00:00:01:02:69/-1" : { "basic": { "name":"CR 21", "latitude":-12, "longitude":84 } },"00:00:00:02:02:69/-1" : { "basic": { "name":"CR 22", "latitude":-12, "longitude":88 }
},"00:00:00:01:03:69/-1" : { "basic": { "name":"CR 31", "latitude":-12, "longitude":144 } },"00:00:00:02:03:69/-1" : { "basic": { "name":"CR 32", "latitude":-12, "longitude":148
} },"00:00:00:01:01:01/-1" : { "basic": { "name":"dsl11", "latitude":-44, "longitude":4 } },"00:00:00:02:01:01/-1" : { "basic": { "name":"dsl12", "latitude":-44, "longitude":6 }
},"00:00:00:03:01:01/-1" : { "basic": { "name":"dsl13", "latitude":-44, "longitude":8 } },"00:00:00:01:02:01/-1" : { "basic": { "name":"dsl121", "latitude":-44, "longitude":12 }
},"00:00:00:02:02:01/-1" : { "basic": { "name":"dsl22", "latitude":-44, "longitude":14 } },"00:00:00:03:02:01/-1" : { "basic": { "name":"dsl23", "latitude":-44, "longitude":16 }
},"00:00:00:01:03:01/-1" : { "basic": { "name":"dsl31", "latitude":-44, "longitude":20 } },"00:00:00:02:03:01/-1" : { "basic": { "name":"dsl32", "latitude":-44, "longitude":22 }
},"00:00:00:03:03:01/-1" : { "basic": { "name":"dsl33", "latitude":-44, "longitude":24 } },"00:00:00:01:01:02/-1" : { "basic": { "name":"ont11", "latitude":-44, "longitude":28 }
},"00:00:00:02:01:02/-1" : { "basic": { "name":"ont12", "latitude":-44, "longitude":30 } },"00:00:00:03:01:02/-1" : { "basic": { "name":"ont13", "latitude":-44, "longitude":32 }
},"00:00:00:01:02:02/-1" : { "basic": { "name":"ont21", "latitude":-44, "longitude":36 } },"00:00:00:02:02:02/-1" : { "basic": { "name":"ont22", "latitude":-44, "longitude":38 }
},"00:00:00:03:02:02/-1" : { "basic": { "name":"ont23", "latitude":-44, "longitude":40 } },"00:00:00:01:03:02/-1" : { "basic": { "name":"ont31", "latitude":-44, "longitude":44 }
},"00:00:00:02:03:02/-1" : { "basic": { "name":"ont32", "latitude":-44, "longitude":46 } },"00:00:00:03:03:02/-1" : { "basic": { "name":"ont33", "latitude":-44, "longitude":48 }
},"00:00:01:01:01:03/-1" : { "basic": { "name":"dslgf111", "latitude":-52, "longitude":52 } },"00:00:02:01:01:03/-1" : { "basic": { "name":"dslgf112", "latitude":-52,
"longitude":54 } },"00:00:03:01:01:03/-1" : { "basic": { "name":"dslgf113", "latitude":-52, "longitude":56 } },"00:00:01:02:01:03/-1" : { "basic": { "name":"dslgf121",
"latitude":-52, "longitude":60 } },"00:00:02:02:01:03/-1" : { "basic": { "name":"dslgf122", "latitude":-52, "longitude":62 } },"00:00:03:02:01:03/-1" : { "basic": {
"name":"dslgf123", "latitude":-52, "longitude":64 } },"00:00:01:03:01:03/-1" : { "basic": { "name":"dslgf131", "latitude":-52, "longitude":68 } },"00:00:02:03:01:03/-1" : {
"basic": { "name":"dslgf132", "latitude":-52, "longitude":70 } },"00:00:03:03:01:03/-1" : { "basic": { "name":"dslgf133", "latitude":-52, "longitude":72 } },"00:00:01:01:02:03/-
1" : { "basic": { "name":"dslgf211", "latitude":-52, "longitude":76 } },"00:00:02:01:02:03/-1" : { "basic": { "name":"dslgf212", "latitude":-52, "longitude":78 }
},"00:00:03:01:02:03/-1" : { "basic": { "name":"dslgf213", "latitude":-52, "longitude":80 } },"00:00:01:02:02:03/-1" : { "basic": { "name":"dslgf221", "latitude":-52,
"longitude":84 } },"00:00:02:02:02:03/-1" : { "basic": { "name":"dslgf222", "latitude":-52, "longitude":86 } },"00:00:03:02:02:03/-1" : { "basic": { "name":"dslgf223",
"latitude":-52, "longitude":88 } },"00:00:01:03:02:03/-1" : { "basic": { "name":"dslgf231", "latitude":-52, "longitude":92 } },"00:00:02:03:02:03/-1" : { "basic": {
"name":"dslgf232", "latitude":-52, "longitude":94 } },"00:00:03:03:02:03/-1" : { "basic": { "name":"dslgf233", "latitude":-52, "longitude":96 } },"00:00:01:01:03:03/-1" : {
"basic": { "name":"dslgf311", "latitude":-52, "longitude":100 } },"00:00:02:01:03:03/-1" : { "basic": { "name":"dslgf312", "latitude":-52, "longitude":102 }
},"00:00:03:01:03:03/-1" : { "basic": { "name":"dslgf313", "latitude":-52, "longitude":104 } },"00:00:01:02:03:03/-1" : { "basic": { "name":"dslgf321", "latitude":-52,
"longitude":108 } },"00:00:02:02:03:03/-1" : { "basic": { "name":"dslgf322", "latitude":-52, "longitude":110 } },"00:00:03:02:03:03/-1" : { "basic": { "name":"dslgf323",
"latitude":-52, "longitude":112 } },"00:00:01:03:03:03/-1" : { "basic": { "name":"dslgf331", "latitude":-52, "longitude":116 } },"00:00:02:03:03:03/-1" : { "basic": {
"name":"dslgf332", "latitude":-52, "longitude":118 } },"00:00:03:03:03:03/-1" : { "basic": { "name":"dslgf333", "latitude":-52, "longitude":120 } },"00:00:00:01:01:04/-1" : {
"basic": { "name":"cp11", "latitude":-44, "longitude":124 } },"00:00:00:02:01:04/-1" : { "basic": { "name":"cp12", "latitude":-44, "longitude":126 } },"00:00:00:03:01:04/-1" : {
"basic": { "name":"cp13", "latitude":-44, "longitude":128 } },"00:00:00:01:02:04/-1" : { "basic": { "name":"cp21", "latitude":-44, "longitude":132 } },"00:00:00:02:02:04/-1" : {
"basic": { "name":"cp22", "latitude":-44, "longitude":134 } },"00:00:00:03:02:04/-1" : { "basic": { "name":"cp23", "latitude":-44, "longitude":136 } },"00:00:00:01:03:04/-1" : {
"basic": { "name":"cp31", "latitude":-44, "longitude":140 } },"00:00:00:02:03:04/-1" : { "basic": { "name":"cp32", "latitude":-44, "longitude":142 } },"00:00:00:03:03:04/-1" : {
"basic": { "name":"cp33", "latitude":-44, "longitude":144 } },"00:00:00:01:01:05/-1" : { "basic": { "name":"ecpe11", "latitude":-36, "longitude":148 } },"00:00:00:02:01:05/-1" :
{ "basic": { "name":"ecpe12", "latitude":-36, "longitude":152 } },"00:00:00:03:01:05/-1" : { "basic": { "name":"ecpe13", "latitude":-36, "longitude":156 } },"00:00:00:01:01:06/-
1" : { "basic": { "name":"sbng11", "latitude":-36, "longitude":160 } },"00:00:00:02:01:06/-1" : { "basic": { "name":"sbng21", "latitude":-36, "longitude":162 }
},"00:00:00:01:02:06/-1" : { "basic": { "name":"sdhcp11", "latitude":-44, "longitude":164 } },"00:00:00:02:02:06/-1" : { "basic": { "name":"sdhcp21", "latitude":-44,
"longitude":166 } },"00:00:00:01:03:06/-1" : { "basic": { "name":"sdns11", "latitude":-52, "longitude":168 } },"00:00:00:02:03:06/-1" : { "basic": { "name":"sdns21",
"latitude":-52, "longitude":172 } },"00:00:00:01:04:06/-1" : { "basic": { "name":"saaa11", "latitude":-44, "longitude":174 } },"00:00:00:02:04:06/-1" : { "basic": {
"name":"saaa21", "latitude":-44, "longitude":176 } },"00:00:00:01:05:06/-1" : { "basic": { "name":"scache11", "latitude":-36, "longitude":178 } },"00:00:00:02:05:06/-1" : {
"basic": { "name":"scache21", "latitude":-36, "longitude":180 } },
"00:00:00:00:00:00/-1" : { "basic": { "name":"NOT_EXISTENT" } }
}
}
root@sdn-onos-prod-01:~# systemctl start onos
```

Achtung, der Text ist eventuell zu lang und wird irgendwo abgeschnitten. Die Topologie wird dann nicht dargestellt. Im Logfile beschwert sich ONOS dann über eine defekte network-cfg.json. Wenn er das nicht tut, kommt ONOS mit der richtigen Topologie hoch.



Nach einem Mininet `pingall`



Perfekt !