

IaC (Infrastructure as Code) & LernMAAS (Metal-As-A-Service)

Mai 2021

Marcel Bernet

Lizenz



Namensnennung - Nicht-kommerziell - Weitergabe unter gleichen Bedingungen 3.0 Schweiz (CC BY-NC-SA 3.0 CH)

This is a human-readable summary of (and not a substitute for) the license, which is available in the following languages: [Deutsch](#) [Französisch](#) [Haftungsbeschränkung](#).



Sie dürfen:

Teilen — das Material in jedwedem Format oder Medium vervielfältigen und weiterverbreiten

Bearbeiten — das Material remixen, verändern und darauf aufbauen

Der Lizenzgeber kann diese Freiheiten nicht widerrufen solange Sie sich an die Lizenzbedingungen halten.

Unter folgenden Bedingungen:

 **Namensnennung** — Sie müssen [angemessene Urheber- und Rechteangaben machen](#), einen Link zur Lizenz beifügen und angeben, ob [Änderungen vorgenommen](#) wurden. Diese Angaben dürfen in jeder angemessenen Art und Weise gemacht werden, allerdings nicht so, dass der Eindruck entsteht, der Lizenzgeber unterstützte gerade Sie oder Ihre Nutzung besonders.

 **Nicht kommerziell** — Sie dürfen das Material nicht für [kommerzielle Zwecke](#) nutzen.

 **Weitergabe unter gleichen Bedingungen** — Wenn Sie das Material remixen, verändern oder anderweitig direkt darauf aufbauen, dürfen Sie Ihre Beiträge nur unter [derselben Lizenz](#) wie das Original verbreiten.

 **Keine weiteren Einschränkungen** — Sie dürfen keine zusätzlichen Klauseln oder [technische Verfahren](#) einsetzen, die anderen rechtlich irgendetwas untersagen, was die Lizenz erlaubt.

Agenda

- Infrastructure as Code (wie kommt die Software auf die VM?)
 - Übung 1 + 2
- Projekt LernMAAS (Eine Cloud für die Ausbildung)
 - Übung 3 + 4
- Abschluss



Infrastructure as Code

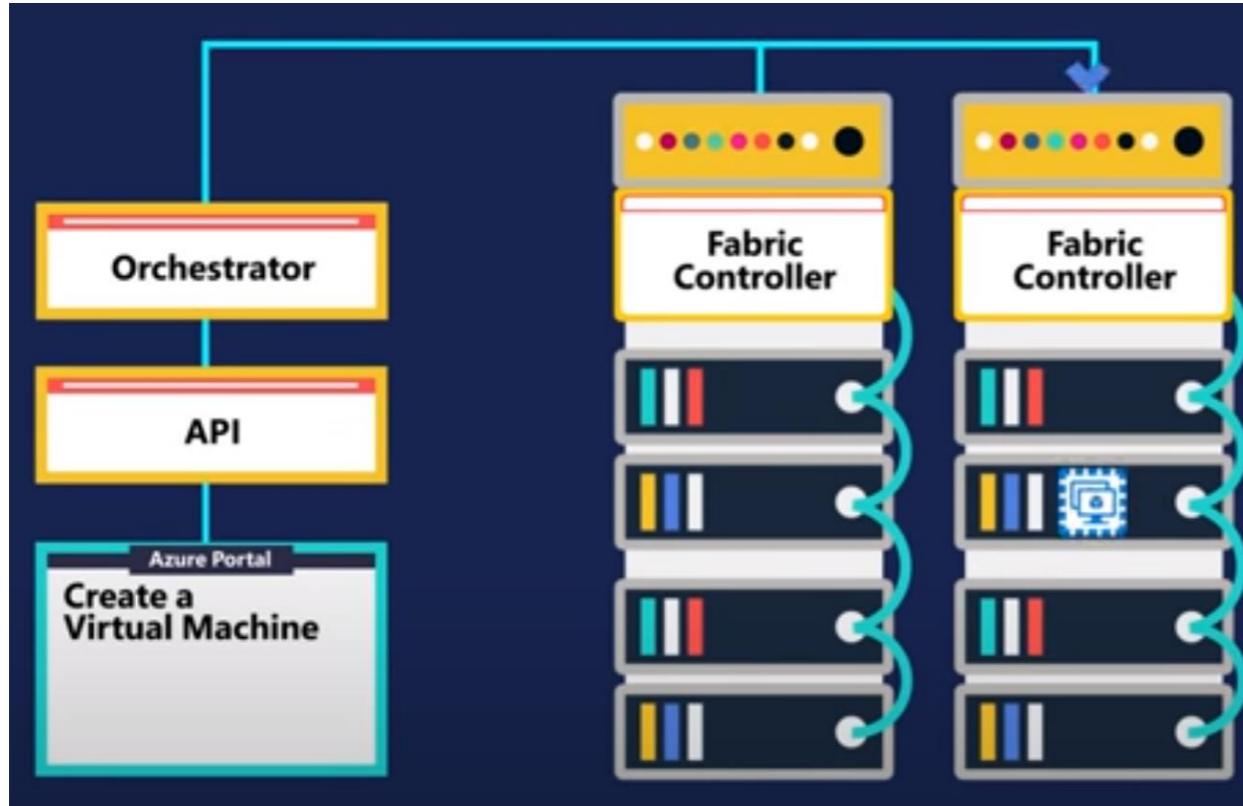
Wie kommt die Software auf die VM?

Die Eisenzeit (Imperativ) vs. die Cloud



- In der „Eisenzeit“ der IT waren Systeme direkt an physische Hardware gebunden.
- Das Bereitstellen und Warten der Infrastruktur war manuelle Arbeit, die die Menschen dazu zwang, zu klicken und zu tippen, um die Systeme am laufen zu halten.
- Da Änderungen viel Zeit und Geld erforderten, wurde in Änderungsmanagement und sorgfältige Prüfung investiert
- Dies machte Sinn, weil es teuer war, etwas falsch zu machen.

Die Eisenzeit vs. die Cloud (Deklarativ)



- Im „Cloud-Zeitalter“ der IT werden Systeme von der physischen Hardware entkoppelt.
- Die routinemässige Bereitstellung und Wartung kann an Softwaresysteme delegiert werden, um den Menschen von Routinearbeiten zu befreien.
- Änderungen können in Minuten, wenn nicht Sekunden vorgenommen werden.
- Wir können diese Geschwindigkeit nutzen für eine höhere Zuverlässigkeit sowie schnellere Einführung von neuen Produkten.

Quelle: <https://www.youtube.com/watch?v=KXkBZCe699A>

Was ist Infrastruktur als Code?

- Infrastruktur als Code ist ein Ansatz zur Automatisierung der Infrastruktur, der auf Praktiken aus der Softwareentwicklung basiert.
- Dabei werden konsistente, wiederholbare Prozesse für die Bereitstellung und Änderung von Systemen und deren Konfiguration verwendet.
- Änderungen werden an Deklarationen (Code) vorgenommen und dann durch automatisierte Prozesse, auf Systeme übertragen.
- Infrastruktur als Code hat sich in den anspruchsvollsten Umgebungen bewährt. Für Unternehmen wie Amazon, Netflix, Google und Facebook sind IT-Systeme nicht nur geschäftskritisch. Sie sind das Geschäft!

Ziele von Infrastruktur als Code

- Die IT-Infrastruktur unterstützt und ermöglicht Veränderungen, anstatt ein Hindernis oder eine Einschränkung zu sein.
- Änderungen am System sind Routine, ohne Drama oder Stress für Benutzer oder IT-Mitarbeiter.
- IT-Mitarbeiter verbringen ihre Zeit mit wertvollen Dingen, die ihre Fähigkeiten einbeziehen, und nicht mit sich wiederholenden Routineaufgaben.
- Benutzer können die benötigten Ressourcen definieren, bereitstellen und verwalten, ohne dass IT-Mitarbeiter dies für sie tun müssen.
- Teams können sich einfach und schnell von Fehlern erholen, anstatt davon auszugehen, dass Fehler vollständig verhindert werden können.
- Verbesserungen werden kontinuierlich vorgenommen und nicht durch teure und riskante „Urknall“ - Projekte.
- Lösungen für Probleme werden durch Implementierung, Test und Messung bewiesen, anstatt sie in Besprechungen und Dokumenten zu diskutieren.

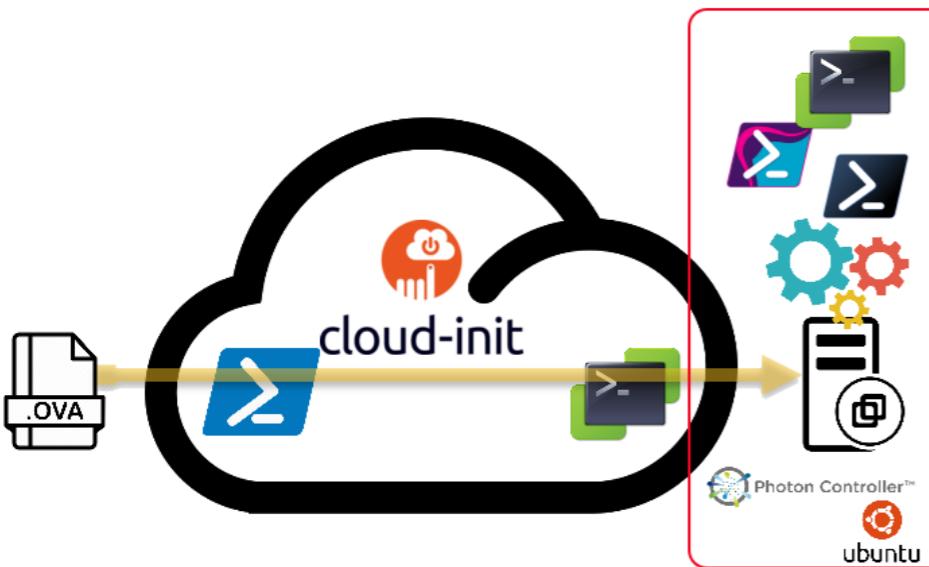
Infrastruktur als Code (die zweite)

Install arbitrary packages

```
1 #cloud-config
2
3 # Install additional packages on first boot
4 #
5 # Default: none
6 #
7 # if packages are specified, this apt_update will be set to true
8 #
9 # packages may be supplied as a single package name or as a list
10 # with the format [<package>, <version>] wherein the specific
11 # package version will be installed.
12 packages:
13   - pwgen
14   - pastebinit
15   - [libpython2.7, 2.7.3-0ubuntu3.1]
```

- Ein [Paradigma](#) (grundsätzliche Denkweise) zur Infrastrukturautomation.
- Basiert auf konsistenten und wiederholbaren Definitionen (Code) für die Bereitstellung von Systemen und deren Konfiguration.
- **Produkte:** Puppet, Chef, Cloud-init, Vagrant ...

Produkt: Cloud-init (Provisioning)



Install arbitrary packages

Microsoft Azure

Home > Virtuelle Computer > Virtuellen Computer erstellen

Virtuellen Computer erstellen

Grundeinstellungen Datenträger Netzwerk Verwaltung Erweitert Tags Überprüfen + erstellen

Fügen Sie zusätzliche Konfigurationen, Agents, Skripts oder Anwendungen über VM-Erweiterungen oder per cloud-init hinzu.

Erweiterungen

Erweiterungen ermöglichen eine Konfiguration und Automatisierung nach der Bereitstellung.

Erweiterungen ⓘ Zu installierende Erweiterung auswählen

cloud-init

cloud-init ist ein weit verbreiteter Ansatz zum Anpassen einer Linux-VM beim ersten Start. Mit cloud-init können Sie Pakete installieren und Dateien schreiben oder Benutzer und Sicherheit konfigurieren. [Weitere Informationen](#)

cloud-init

```

1  #cloud-config
2
3  # Install additional packages on first boot
4  #
5  # Default: none
6  #
7  # if packages are specified, this apt_update
8  #
9  # packages may be supplied as a single package
10 # with the format [<package>, <version>] or
11 # package version will be installed.
12 packages:
13   - pwgen
14   - pastebinit
15   - [libpython2.7, 2.7.3-0ubuntu3.1]

```

- Cloud-init ist die Multi-Distributionsmethode für die plattformübergreifende Initialisierung von Cloud-Instanzen.
- Es wird von allen grossen öffentlichen Cloud-Anbietern, Bereitstellungssystemen für private Cloud-Infrastrukturen und Bare-Metal-Installationen unterstützt.

Infrastruktur als Code in der Cloud mittels Cloud-init

MAAS Machines Devices Controllers KVM Images DNS AZs Subnets Settings admin Log out

Machines 1 of 6 machines selected

OS Ubuntu Release Ubuntu 18.04 LTS "Bionic Beaver" Kernel No minimum kernel

Add hardware Take action

Customise options Register as MAAS kvm Cloud-init user-data...

Microsoft Azure Dashboard > Virtuelle Computer > Virtuellen Computer erstellen

```
#cloud-config
#
# ACHTUNG: es w1
#
# Installation w
packages:
- nginx
```

Choose File No file chosen

Grundeinstellungen Datenträger Netzwerk Verw.

Fügen Sie zusätzliche Konfigurationen, Agents, Skripts oder Anv

Erweiterungen

We've released MAAS 2.8. This version supports LXD VM hosts, faster UI and more.

Filters in:(selected)

FQDN | MAC IP POWER STATUS

Ready 1 machine selected

legal-finch.maas Off Ready

Erweiterungen (1) Zu installierende Erweiterungen

Benutzerdefinierte Daten und cloud-init

Übergeben Sie ein cloud-init-Skript, eine Konfigurationsdatei oder eine Metadatendatei, die bereitgestellt wird. Die Daten werden auf dem virtuellen Computer übertragen.

Weitere Informationen zu benutzerdefinierten Daten für VMs

Benutzerdefinierte Daten

```
#cloud-config
hostname: m122-02
fqdn: m122-20.ma
manage_etc_hosts
users:
- name: ubuntu
```

aws Services Ressourcengruppen Marcel Bernet Frankfurt

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add

Name m122-02

Labels (Optional) + Label hinzufügen

Region us-central1 (Iowa) Zone us-central1-a

Bootlaufwerk Neuer nichtflüchtiger Standardspeicher mit 10 GB Image Ubuntu 20.04 LTS Minimal Ändern

Advanced Details

Metadata accessible Enabled

Metadata version V1 and V2 (token optional)

Metadata token response hop limit 1

User data As text As file I

```
#cloud-config
hostname: dukdemo-03
fqdn: dukdemo-03.north
manage_etc_hosts: true
users:
- name: ubuntu
```

Identität und API-Zugriff

Dienstkonto Compute Engine default service account

Zugriffsbereiche Standardzugriff zulassen

Firewall Sie können Tags und Firewallregeln hinzufügen, um bestimmten Netzwerktraffic aus dem Internet zuzulassen.

HTTP-Traffic zulassen HTTPS-Traffic zulassen

Verwaltung Sicherheit Laufwerke Netzwerke Einzelne Mandanten

Beschreibung (Optional)

Löschschutz Löschschutz aktivieren When deletion protection is enabled, instance cannot be deleted. Learn more

Reservierungen Beim Erstellen dieser VM-Instanz eine vorhandene Reservierung verwenden Erstellte Reservierung automatisch verwenden

Automatisierung Startskript (Optional) You can choose to specify a startup script that will run when your instance boots up or restarts. Startup scripts can be used to install software and updates, and to ensure that services are running within the virtual machine. Learn more

Metadaten (Optional) You can set custom metadata for an instance or project outside of the server-defined metadata. This is useful for passing in arbitrary values to your project or instance that can be queried by your code on the instance. Learn more

user-data

```
#cloud-config
hostname: m122-02
fqdn: m122-02.southcentralus.cloudapp.azure.com
manage_etc_hosts: true
users:
- name: ubuntu
  sudo: ALL=(ALL) NOPASSWD:ALL
  groups: users, admin
  home: /home/ubuntu
  shell: /bin/bash
  lock_passwd: false
  sshAuthorizedKeys:
    - ssh-rsa
      AAAAB3NzaC1v2EAAAQABAAQACQDA
```

<https://github.com/mc-b/lernmaas/tree/master/doc/Cloud>

Cloud-init: Dokumentation und Beispiele

 **cloud-init**
latest

[Docs](#) » cloud-init Documentation [Edit on GitHub](#)

cloud-init Documentation

Cloud-init is the *industry standard* multi-distribution method for cross-platform cloud instance initialization. It is supported across all major public cloud providers, provisioning systems for private cloud infrastructure, and bare-metal installations.

Cloud instances are initialized from a disk image and instance data:

- Cloud metadata
- User data (optional)
- Vendor data (optional)

Cloud-init will identify the cloud it is running on during boot, read any provided metadata from the cloud and initialize the system accordingly. This may involve setting up the network and storage devices to configuring SSH access key and many other aspects of a system. Later on the cloud-init will also parse and process any optional user or vendor data that was passed to the instance.

Getting help

Having trouble? We would like to help!

- Try the [FAQ](#) – its got answers to some common questions
- Ask a question in the [#cloud-init](#) IRC channel on Freenode
- Join and ask questions on the [cloud-init mailing list](#)
- Find a bug? Report bugs on [Launchpad](#)

<https://cloudinit.readthedocs.io/en/latest/>

 **cloud-init**
latest

[Docs](#) » Cloud config examples [Edit on GitHub](#)

Cloud config examples

Including users and groups

```

1  #cloud-config
2  # Add groups to the system
3  # The following example adds the ubuntu group with members 'root' and 'sys'
4  # and the empty group cloud-users.
5  groups:
6      - ubuntu: [root,sys]
7      - cloud-users
8
9  # Add users to the system. Users are added after groups are added.
10 # Note: Most of these configuration options will not be honored if the user
11 # already exists. Following options are the exceptions and they are
12 # applicable on already-existing users:
13 #   - 'plain_text_passwd', 'hashed_passwd', 'lock_passwd', 'sudo',
14 #   'ssh_authorized_keys', 'ssh_redirect_user'.
15 users:
16     - default
17     - name: foobar
18       gecos: Foo B. Bar
19       primary_group: foobar
20     groups: users
21     selinux_user: staff_u
22     expiredate: 2012-09-01
23     ssh_import_id: foobar
24     lock_passwd: false
25     passwd: $6$j212wezy$7H/1LT4f9/N3wpgNunhsIqtMj62OKiS3nyNwuiouQc3u7MbYCarYeAHwYPYb2FT.1bioDm2
26     - name: barfoo
27       gecos: Bar B. Foo
28       sudo: ALL=(ALL) NOPASSWD:ALL
29       groups: users, admin
30       ssh_import_id: None

```

<https://cloudinit.readthedocs.io/en/latest/topics/examples.html>

Cloud-init: YAML (Dateien)

Install arbitrary packages

```
1 #cloud-config
2
3 # Install additional packages on first boot
4 #
5 # Default: none
6 #
7 # if packages are specified, this apt_update will be set to true
8 #
9 # packages may be supplied as a single package name or as a list
10 # with the format [<package>, <version>] wherein the specific
11 # package version will be installed.
12 packages:
13   - pwgen
14   - pastebinit
15   - [libpython2.7, 2.7.3-0ubuntu3.1]
```

Achtung

Wichtig bei allen folgenden Yaml/JSON-Files sind natürlich insbesondere die korrekten Einrückungen, da über sie die Hierarchien der Direktiven definiert werden.

Zudem ist darauf zu achten, dass die Verwendung von Tabs bzw. die Vermischung von Tabs und Spaces problematisch sein kann. Infofern sollten für alle Einrückungen am besten durchgängig Spaces verwendet werden.

- YAML ist eine vereinfachte Auszeichnungssprache (englisch markup language) zur Datenserialisierung, angelehnt an XML (ursprünglich) und an die Datenstrukturen in den Sprachen Perl, Python und C sowie dem in [RFC2822](#) vorgestellten E-Mail-Format.
- Die grundsätzliche Annahme von YAML ist, dass sich jede beliebige Datenstruktur nur mit assoziativen Listen, Listen (Arrays) und Einzelwerten (Skalaren) darstellen lässt. Durch dieses einfache Konzept ist YAML wesentlich leichter von Menschen zu lesen und zu schreiben als beispielsweise XML, ausserdem vereinfacht es die Weiterverarbeitung der Daten, da die meisten Sprachen solche Konstrukte bereits integriert haben.
- **Cloud-init verwendet YAML als Beschreibungssprache.**

Cloud-init: wichtige Einträge

- **user:** Dient zum Erstellen und konfigurieren von Usern.
- **packages:** Beschreibt, in einem Array, welche Linux-Packages installiert werden sollen.
- **runcmd:** Beschreibt, in einem Array, welche Shell (sh) Befehle ausgeführt werden sollen.
- **write_files:** dient zum Erstellen von Dateien.
- Cloud-init: [Examples](#)

Cloud-init: (Ubuntu) Linux - Packages

ubuntu[®] packages

» Ubuntu » Pakete » focal » Index

[xenial][xenial-updates][xenial-backports][bionic][bionic-updates][bionic-backports][focal][focal-updates][focal-backports][groovy][groovy-updates][groovy-backports][hirsute]

Liste der Bereiche in »focal«

Administrationswerkzeuge

Hilfsprogramme, um Systemressourcen zu verwalten, Benutzerkonten zu verwalten, usw.

Mono/CLI

Alles rund um Mono und die »Common Language Infrastructure«.

Kommunikationsprogramme

Software, um Ihr Modem auf altmodische Art zu verwenden.

Datenbanken

Datenbankserver und -clients

udeb-Pakete des Debian-Installers

Spezielle Pakete zum Erzeugen von angepassten Debian-Installer-Varianten. Installieren Sie sie nicht auf einem n

Debug-Pakete

Pakete, die Debug-Informationen für Programme und Laufzeitbibliotheken bereit stellen.

Entwicklung

Entwicklungswerzeuge, Compiler, Entwicklungsumgebungen, Bibliotheken, usw.

Dokumentation

FAQs, HOWTOs und andere Dokumente, die versuchen, alles in Bezug zu Debian zu erklären, sowie Software, die nutzen (man, info, usw.)

Editoren

Software, um Dateien zu editieren. Programmierumgebungen.

Elektronik

Elektronikprogramme.

Wissenschaft

Programme zum wissenschaftlichen Arbeiten

Shells

Kommandoshells. Benutzerfreundliche Schnittstellen für Anfänger.

Klang

Programme, um mit Klängen zu arbeiten: Abmischen, Abspielen, Aufzeichnen, CD-Abspielen, usw.

TeX

Die berühmte Schriftsatz-Software und verwandte Programme.

Textverarbeitung

Programme, um Textdokumente zu formatieren und zu drucken.

Übersetzungen

Übersetzungspakete und Sprachunterstützungs-Metapakete

Hilfsprogramme

Hilfsprogramme zur Datei/Platten-Manipulation, Backup- und Archivierungswerzeuge, System-Beobachtung, Eingabesysteme, usw.

Versionskontrollsysteme

Versionskontrollsysteme und zugehörige Hilfswerzeuge.

Video

Videobetrachter, -editoren, -rekorder, -sender.

Virtuelle Pakete

Virtuelle Pakete.

Web-Software

Web-Server, Browser, Proxys, Download-Tools, usw.

»X Window System«-Software

X-Server, Bibliotheken, Zeichensätze, Windowmanager, Terminal-Emulatoren und viele verwandte Anwendungen.

Xfce

Xfce, eine schnelle und leichtgewichtige Desktop-Umgebung.

Zope/Plone-Rahmenwerk

Zope-Anwendungs-Server und Plone-Inhaltsverwaltungssystem

Quelle: <https://packages.ubuntu.com/focal/>

Warum Linux?

>50% Azure VMs running Linux

60% of solutions in Azure Marketplace Linux based

Strategic partnerships with OSS providers

AWS Cloud

m5.xlarge		On-demand		Reserved Instance		Savings Plan	
OS		Linux	Windows	Linux	Windows	Linux	Windows
Hourly Rate		\$0.192	\$0.376	\$0.116	\$0.300	\$0.1341	\$0.318
OS Cost Difference			x 1.96		x 2.59		x 2.37
Type	N/A	N/A	Standard	Standard	Compute	Compute	
Terms	N/A	N/A	Partial upfront 1 year				
Region	US East (N. Virginia)		US East (N. Virginia)		US East (N. Virginia)		

Demo: Azure Cloud und Cloud-init

Cloud-init Script

```
#cloud-config - Installiert den nginx Web Server
packages:
- nginx
```

Home > Create a resource >
Create a virtual machine ...

Basics Disks Networking Management Advanced Tags Review + create

Add additional configuration, agents, scripts or applications via virtual machine extensions or cloud-init.

Extensions

Extensions provide post-deployment configuration and automation.

Extensions ⓘ Select an extension to install

Custom data and cloud init

Pass a cloud-init script, configuration file, or other data into the virtual machine **while it is being provisioned**. The data will be saved on the VM in a known location. [Learn more about custom data for VMs](#)

Custom data

```
#cloud-config - Installiert den nginx Web Server
packages:
- nginx
```

Erstelle VM

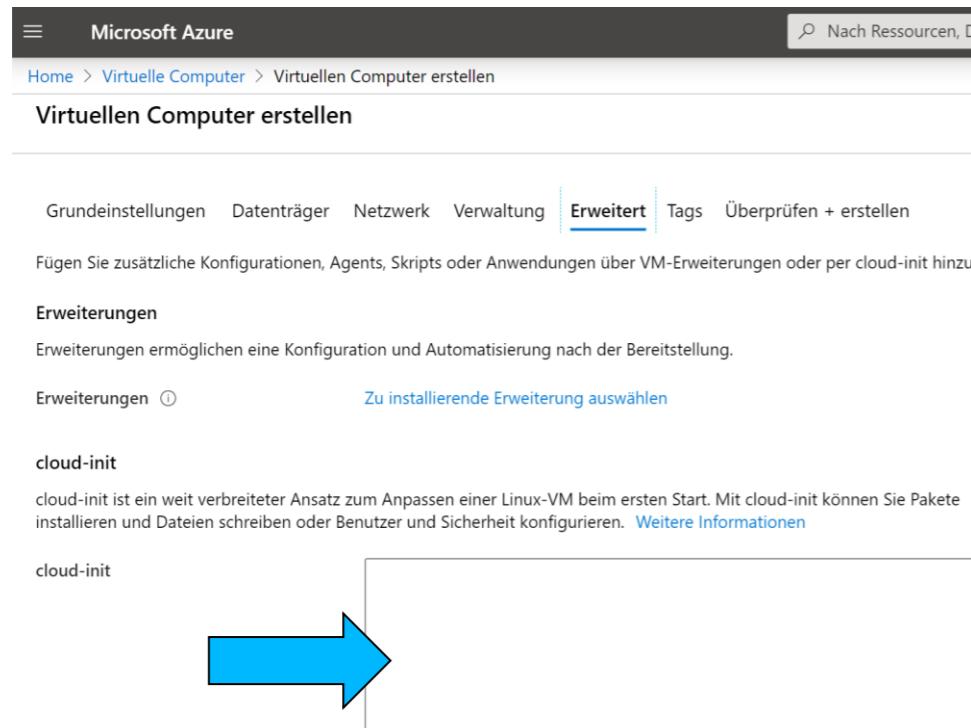
Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.
For online documentation and support please refer to nginx.org. Commercial support is available at nginx.com.
Thank you for using nginx.

<https://github.com/mc-b/virtar#infrastruktur-als-code>

Übung: VM mit Services in der Cloud anlegen

- Erstellt, wie in der Vorbereitung für den Workshop bereits gelernt, eine VM in der Cloud Eurer Wahl und gebt jeweils eines der Cloud-init Scripts ([URL](#)) mit:
 - <https://github.com/mc-b/iac/blob/main/cloud-iac.md>



Microsoft Azure

Nach Ressourcen, Dien:

Home > Virtuelle Computer > Virtuellen Computer erstellen

Virtuellen Computer erstellen

Grundeinstellungen Datenträger Netzwerk Verwaltung **Erweitert** Tags Überprüfen + erstellen

Fügen Sie zusätzliche Konfigurationen, Agents, Skripts oder Anwendungen über VM-Erweiterungen oder per cloud-init hinzu.

Erweiterungen

Erweiterungen ermöglichen eine Konfiguration und Automatisierung nach der Bereitstellung.

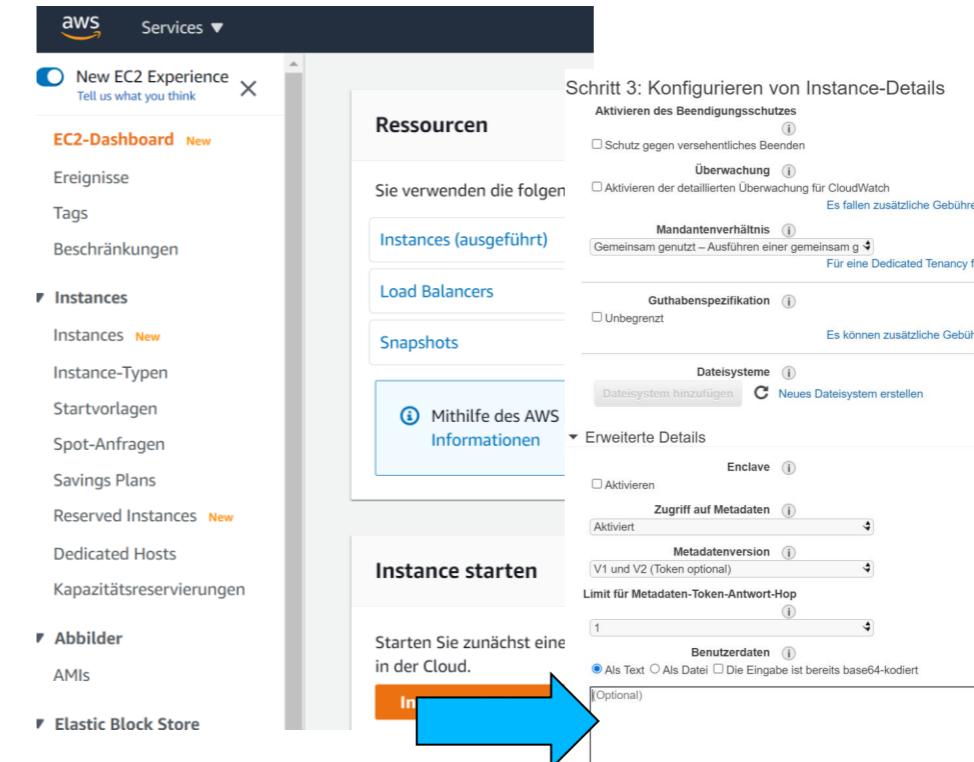
Erweiterungen  Zu installierende Erweiterung auswählen

cloud-init

cloud-init ist ein weit verbreiteter Ansatz zum Anpassen einer Linux-VM beim ersten Start. Mit cloud-init können Sie Pakete installieren und Dateien schreiben oder Benutzer und Sicherheit konfigurieren. [Weitere Informationen](#)

cloud-init





New EC2 Experience Tell us what you think X

EC2-Dashboard New

Ereignisse

Tags

Beschränkungen

Instances Instances New

Instance-Typen

Startvorlagen

Spot-Anfragen

Savings Plans

Reserved Instances New

Dedicated Hosts

Kapazitätsreservierungen

Abbildner AMIs

Elastic Block Store

Ressourcen

Sie verwenden die folgen

Instances (ausgeführt)

Load Balancers

Snapshots

Mithilfe des AWS Informationen

Schritt 3: Konfigurieren von Instance-Details

Aktivieren des Beendigungsschutzes

Schutz gegen versehentliches Beenden

Überwachung

Aktivieren der detaillierten Überwachung für CloudWatch

Es fallen zusätzliche Gebühren an.

Mandantenverhältnis

Gemeinsam genutzt – Ausführen einer gemeinsam g

Für eine Dedicated Tenancy fallen zu

Unbegrenzt

Dateisysteme

Dateisystem hinzufügen Neues Dateisystem erstellen

Instance starten

Enclave

Aktivieren Zugriff auf Metadaten

Aktiviert

Metadatenversion V1 und V2 (Token optional)

Limit für Metadaten-Token-Antwort-Hop

Starten Sie zunächst eine in der Cloud.

Benutzerdaten

Als Text Als Datei Die Eingabe ist bereits base64-kodiert

In

Optional

Kann ich noch mehr Automatisieren? Ja: mit dem jeweiligen Cloud CLI Tools

Azure Cloud

Azure CLI

Nach der [Installation des Azure CLI](#) auf der lokalen Maschine, kann die VM mittels diesem angelegt werden.

Für das Modul M122 sieht das z.B. wie folgt aus.

Erstellen der `cloud-init.cfg` Datei und SSH-Key mit `export VMNAME=m122-02`, wie oben beschrieben.

Anmelden an der Azure Cloud und erstellen einer Ressource Gruppe, wo unsere VMs abgelegt werden:

```
az login

export GROUP=m122
az group create --name ${GROUP} --location southcentralus
```

Erstellen der VM mit Size Standard_D2_v4 ([mit 8 GB RAM, 2 CPUs, 30 GB HD](#)).

```
az vm create --resource-group ${GROUP} --name ${VMNAME} --image UbuntuLTS --size Standard_D2_v4 --lo
```

AWS Cloud

Security Group erstellen und Ports öffnen

```
aws ec2 create-security-group --group-name lernmaas --description "Standard Ports"
aws ec2 authorize-security-group-ingress --group-name lernmaas --protocol tcp --port 22 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-name lernmaas --protocol tcp --port 80 --cidr 0.0.0.0/0
```

Block Device Grösse festlegen und als JSON speichern

```
cat <<%EOF% >device.json
[ {
  "DeviceName": "/dev/sda1",
  "Ebs": {
    "VolumeSize": 30,
    "DeleteOnTermination": true
  }
}
%EOF%
```

Anschliessend die Instance starten. Die Einträge `tag-specifications` sind optional. Diese sind aber hilfreich, zum Wiederfinden oder einfach zur Gruppierung von VMs, weil AWS keine Gruppen wie MAAS oder Azure kennt.

```
aws ec2 run-instances --image-id ami-0767046d1677be5a0 \
--security-group-ids lernmaas \
--instance-type t2.micro \
--count 1 \
--user-data file://cloud-init.cfg \
--block-device-mappings file://device.json \
--tag-specifications 'ResourceType=instance,Tags=[{Key=class,Value=st17a},{Key=modul,Value=m122}]'
```

Für die meisten Module sollte der Instanz Typ `t2.micro` (1 CPU, 1 GB RAM) ausreichend sein. Für Umgebungen, wie z.B. M300, sollte ein Instanz Typ nicht kleiner als `t3.large` (2 CPU, 8 GB RAM) verwendet werden.

```
export VMNAME=m300-03
export GROUP=m300
# cloud-init Datei erzeugen wie oben

aws ec2 run-instances --image-id ami-0767046d1677be5a0 \
--security-group-ids lernmaas \
--instance-type t3.xlarge \
--count 1 \
--block-device-mappings file://device.json \
--user-data file://cloud-init.cfg
```

<https://github.com/mc-b/lernmaas/tree/master/doc/Cloud>

Übung: VM mit Services mittels CLI anlegen

- Erstellt mittels dem jeweiligen Kommandozeilentool (CLI) eine VM inkl. Services.
 - <https://github.com/mc-b/iac/blob/main/cloud-iac-cli.md>

Azure Cloud

Anmelden an der Azure Cloud

```
az login
```

Anschliessend müssen folgende Aktionen ausgeführt werden:

- Erstellen einer Ressource Gruppe, wo unsere VMs abgelegt werden:
- Erstellen der VM
- Freigeben des Ports 80, damit wir via Browser auf die VM bzw. die installierten Services zugreifen können.

```
az group create --name mygroup --location southcentralus
az vm create --resource-group mygroup --name myvm --image UbuntuLTS --size Standard_D2_v4 --location southcentralus --custom-data
az vm open-port --port 80 --resource-group mygroup --name myvm
```

Überprüft das Ergebnis, durch Anwählen der IP-Adresse Eurer VM im Browser.

Um die VM zu löschen, genügt es, die Resource Gruppe zu löschen.

```
az group delete --name mygroup --yes
```

AWS Cloud

Einloggen in AWS Cloud

```
aws configure
```

```
AWS Access Key ID [*****WBM7*]:
AWS Secret Access Key [*****eKJA*]:
Default region name [eu-central-1]:
Default output format [None]:
```

Anschliessend müssen folgende Aktionen ausgeführt werden:

- Security Group erstellen und Ports öffnen
- Erstellen der VM

```
aws ec2 create-security-group --group-name mygroup --description "Standard Ports"
aws ec2 authorize-security-group-ingress --group-name mygroup --protocol tcp --port 22 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-name mygroup --protocol tcp --port 80 --cidr 0.0.0.0/0
```

```
aws ec2 run-instances --image-id ami-0767046d1677be5a0 --security-group-ids mygroup --instance-type t2.micro --count 1 --user-data
```

Anschliessend können wir uns die laufenden VMs anzeigen

```
aws ec2 describe-instances --output table
```

Überprüft das Ergebnis, durch Anwählen der IP-Adresse Eurer VM im Browser.

Projekt LernMAAS

Eine Cloud für die Ausbildung

Projekt LernMAAS - Ziele

Machines 88 machines available

88 Machines 8 Resource pools

Filters FQDN | MAC IP

pool:=(m242-st17a)

	POWER	STATUS	OWNER	ZONE	CORES	ARCH
<input checked="" type="checkbox"/> m242-01-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.87 (PXE)	Virsh		pod-console-lo...			
<input checked="" type="checkbox"/> m242-02-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.105 (PXE)	Virsh		pod-console-lo...			
<input checked="" type="checkbox"/> m242-03-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.106 (PXE)	Virsh		pod-console-lo...			
<input checked="" type="checkbox"/> m242-04-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.104 (PXE)	Virsh		pod-console-lo...			
<input checked="" type="checkbox"/> m242-05-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.88 (PXE)	Virsh		pod-console-lo...			
<input checked="" type="checkbox"/> m242-06-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.89 (PXE)	Virsh		pod-console-lo...			
<input checked="" type="checkbox"/> m242-07-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.90 (PXE)	Virsh		pod-console-lo...			
<input checked="" type="checkbox"/> m242-08-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.91 (PXE)	Virsh		pod-console-lo...			
<input checked="" type="checkbox"/> m242-09-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.92 (PXE)	Virsh		pod-console-lo...			
<input checked="" type="checkbox"/> m242-10-st17a.maas	On	18.04 LTS	ubuntu	default	2	amd64
192.168.8.93 (PXE)	Virsh		pod-console-lo...			

Modul 242: Mikroprozessoranwendung realisieren

Zugriff auf den Server

User / Password

Der User ist **ubuntu**, das Passwort steht in der Datei **/data/.ssh/passwd**.

Einloggen mittels

ssh ubuntu@[IP Adresse]

SSH

Um den Server kann mittels ssh zugegriffen werden.

Der private SSH Key ist auf dem installierten Server unter **/data/.ssh/id_rsa** zu finden. Downloaden und dann wie folgt auf den Server übertragen:

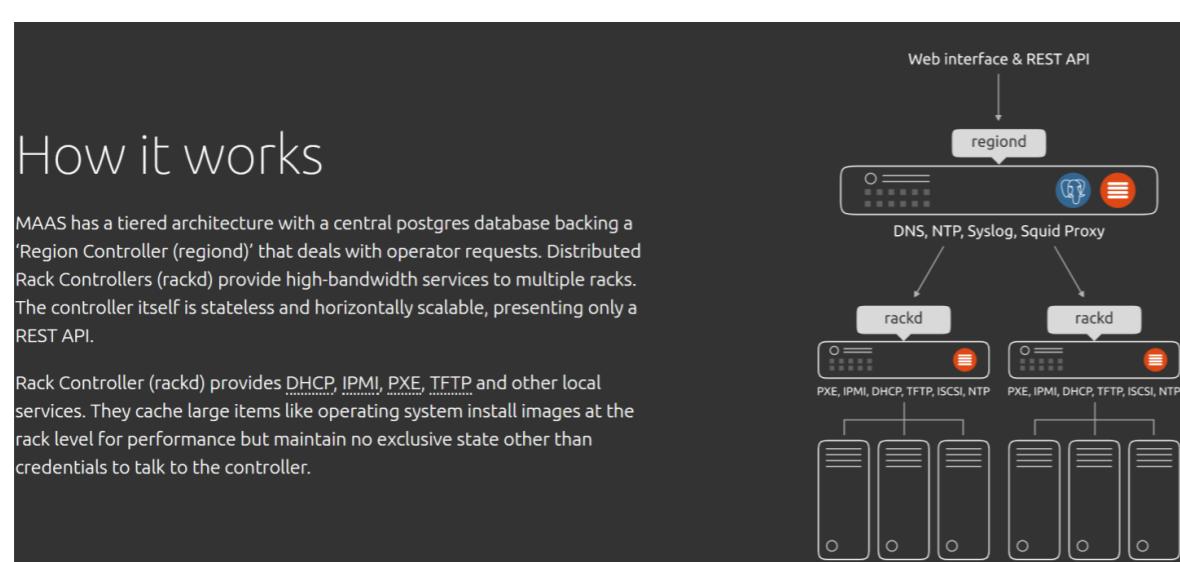
```
ssh -i id_rsa ubuntu@[IP Adresse]
```

Hinweis: Windows User verwenden Putty und den Putty Key **/data/.ssh/id_rsa.ppk**.

Dokumentation

- Für den Kursleiter/Lehrenden
 - Mit 2 Klicks zu einem voll funktionsfähigen Kurs/Modul
- Für die Kursteilnehmer/Lernenden
 - Selbsterklärend
 - Interaktiv
- Allgemein
 - Eine Umgebung von der einfachen Virtuellen Maschine bis zu einem Kubernetes Cluster.
 - Jederzeit "*ab Code*" wiederholbare Umgebungen automatisch aufgesetzt.
 - Erweiterbar ohne Spezialisten
 - Ortsneutral (geeignet für Distance Learning)
 - Verwendung von Cloud Technologien (IaaS, Container, Kubernetes)

Projekt LernMAAS

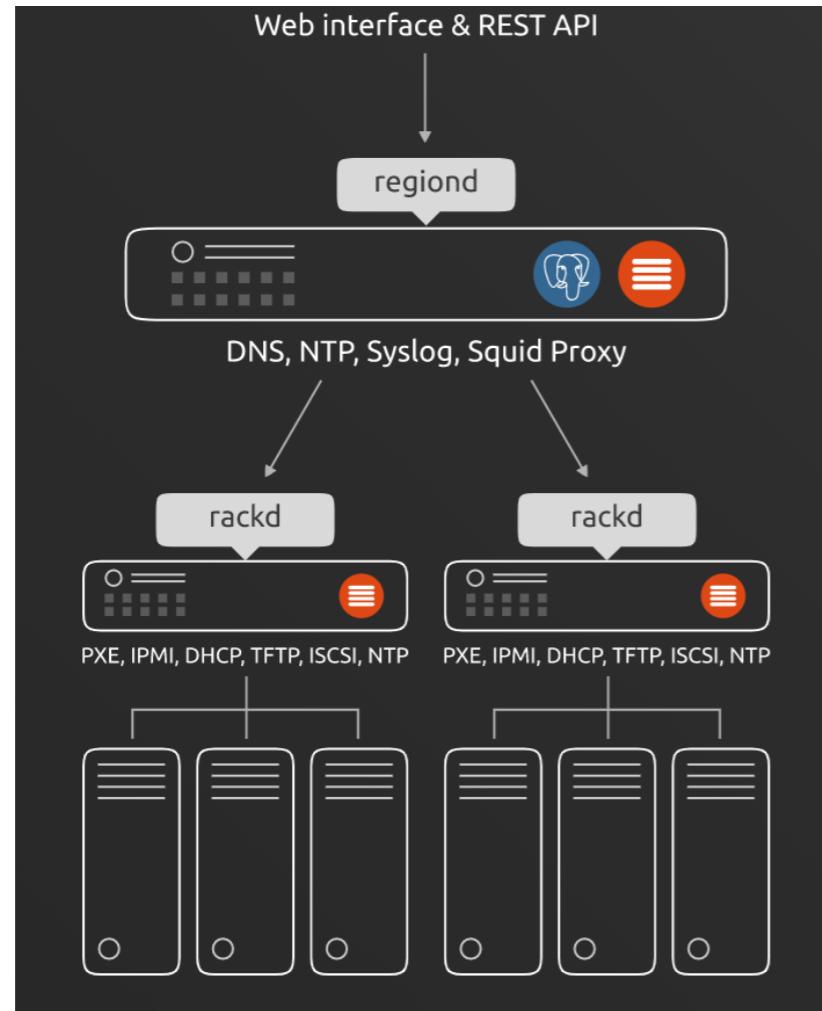


- Bündelt die Technologien/Produkte:
 - Ubuntu [MAAS](#) oder irgendeine [Cloud!](#)
 - [Cloud-init](#) (Initialisierung von Cloud Instanzen)
 - [WireGuard](#) (VPN)
- Und erweitert diese um weitere Funktionen wie:
 - Deklaration von Ausbildungsmodulen und Kursen
 - Aufsetzen von VMs pro Modul und Klasse

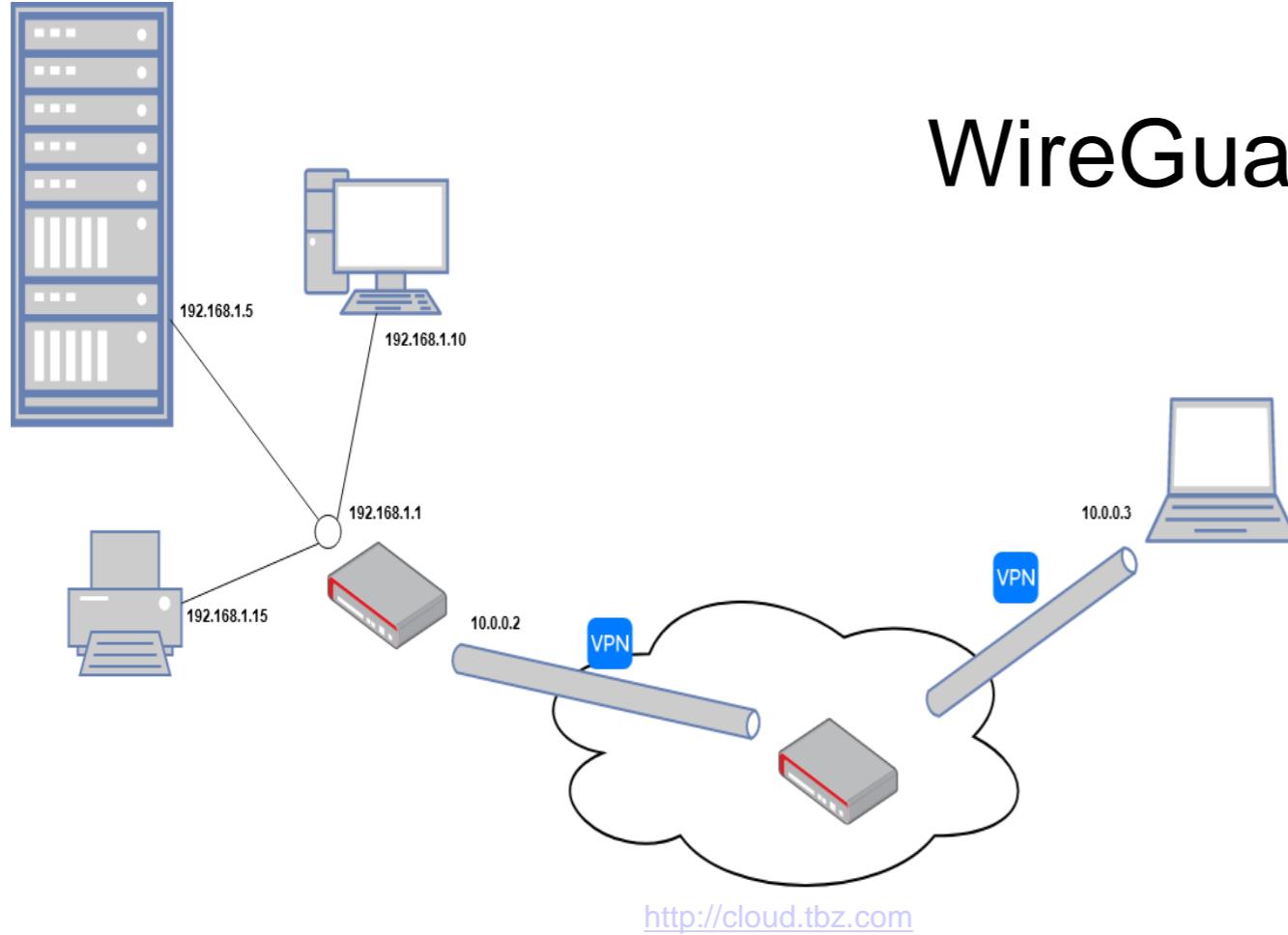
Ist frei verfügbar auf Github:

<https://github.com/mc-b/lernmaas>

Produkt Canonical (Ubuntu) MAAS



- Canonical (Ubuntu) MAAS bittet schnelle Serverbereitstellung für Ihr Rechenzentrum.
- Dabei MAAS verfügt über eine abgestufte Architektur
 - Ein 'Region Controller (regiond)' nimmt die Bereitstellungsanforderungen via UI oder REST-API entgegen.
 - Rack Controller (Rackd) verwalten die Bare-Metal Server wo die virtuellen Maschinen betrieben werden.
- **Zusammengefasst: MAAS verwandelt Ihr Rechenzentrum in eine Bare-Metal-Cloud**



WireGuard

[Interface]

Address = <Replace IP>/24

PrivateKey = <Replace Key>

[Peer]

PublicKey = xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Endpoint = cloud.tbz.ch:51931

AllowedIPs = 10.1.41.0/24

```
# This is for if you're behind a NAT and
# want the connection to be kept alive.
# PersistentKeepalive = 25
```

- WireGuard® ist ein extrem einfaches, aber schnelles und modernes VPN, das **modernste Kryptografie verwendet**.
- Ermöglicht, in Zeiten von Distance Learning, den Fernzugriff auf die Cloud-Instanzen (VMs).
- Vergibt eine statische IPv4-Adresse, für dynamisches Cloud-Computing. Vergleichbar mit "[Elastic IP addresses](#)" von AWS.

Projekt LernMAAS - Konfiguration

```
m242:
  vm:
    storage: 32
    memory: 7680
    cores: 4
  services:
    nfs: true
    docker: true
    k8s: master
    wireguard: use
    ssh: use
    samba: false
    firewall: fa
  scripts: modte
  repositories:
```

VM
Definiert Standardgrößen für die VM. Wird durch [helper Scripts](#) ausgewertet.

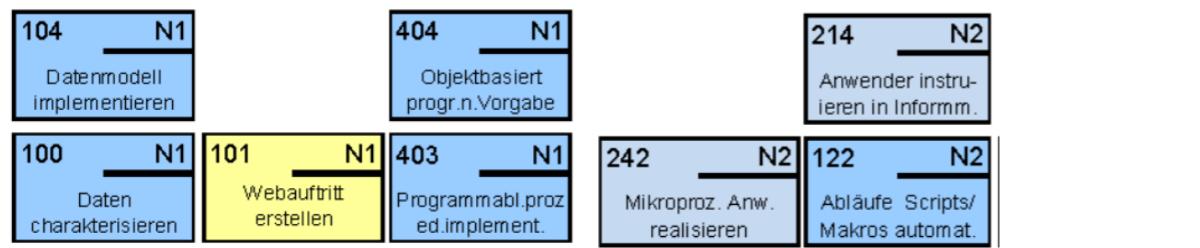
- storage - Disk in GB
- memory - RAM in MB
- cores - Anzahl der CPU Cores

Services
Neben der Möglichkeit eigene Scripts oder Repositories, analog [lernkube](#) einzubinden, stehen eine Anzahl von Services zur Verfügung.
Die Services können pro VM in der Datei [config.yaml](#) aktiviert werden.
Es stehen folgende Services zur Verfügung:

- nfs - `true` = es werden \$HOME/data, \$HOME/template \$HOME/config auf den MAAS Server weitergeleitet, bzw. gemountet.
- wireguard - `use` = ist eine Datei \$HOME/config/wireguard/\$HOSTNAME vorhanden wird diese als Konfigurationsdatei für WireGuard verwendet. Bei Kubernetes wird zusätzlich als 1. IP nicht die interne IP sondern die von WireGuard verwendet.
- ssh - `generate` = erstellt eine SSH-Key pro VM, fügt den Public Key `.ssh/authorized_keys` an und kopiert den Private Key nach data/.ssh/. für den Zugriff von aussen.
- samba - `true` = installiert die CIFS Freigabe Samba und gibt \$HOME/data allgemein frei.
- firewall - `true` = installiert ufw als Firewall mit Standardeinstellungen
- docker - `true` = installiert Docker in der VM
- k8s - `master` = installiert einen Kubernetes Master, `worker` = installiert einen Kubernetes Worker und joint diesen mit dem Kubernetes Master.

- Jedes Modul muss in der config.yaml beschrieben werden.
- Eine detaillierte Beschreibung findet man auf: <https://github.com/mc-b/lernmaas#konfigurationsdatei-configyaml>
- Mittels Repositories, bzw. dem Installationsscript script/install.sh können eigene Installationsroutinen zu den Modulen hinzugefügt werden.
- Beispiele: <https://github.com/tbz-it>

Projekt LernMAAS - ist Modular



tbz-it Beispiele: <https://github.com/tbz-it>

The screenshot shows a GitHub repository page for the 'tbz-it' organization. The main navigation bar includes 'Repositories', 'Packages', 'People', 'Teams', 'Projects', and 'Settings'. Below the navigation, there's a search bar and filters for 'Type: All' and 'Language: All'. A 'New' button is also present.

The repository list displays several projects:

- M426**: Software mit agilen Methoden entwickeln. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 20 hours ago.
- M254**: Geschäftsprozesse beschreiben. Description: Daten charakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 2 days ago.
- M242**: Mikroprozessoranwendung realisieren. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 2 days ago.
- M239**: Internet Server in Betrieb nehmen. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 2 days ago.
- M437**: Arbeiten im Support. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 2 days ago.
- M183**: Applikationssicherheit implementieren. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 2 days ago.
- M100**: Daten charakterisieren, aufbereiten und auswerten. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 3 days ago.
- M141**: Datenbanksystem in Betrieb nehmen. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 3 days ago.
- M152**: Multimedia-Inhalte in Webauftritt integrieren. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 4 days ago.
- M104**: Datenmodell implementieren. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 5 days ago.
- my-cloud-init**: Beispiel für eigene Cloud-Init Implementierung für lemaas. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 12 days ago.
- M226**: Objektorientiert implementieren. Description: Datencharakterisieren, aufbereiten und auswerten. Languages: Shell, MIT. Updated 15 days ago.

- Module erweitern die Funktion von LernMAAS mittels Shell-Scripten.
- ## Ergebnis

 - Vorbereitete Konfigurationen für IT Module.
 - Helper Scripts zum Erstellen von VMs für ganze Klassen.
 - Ein Erweiterungsmechanismus mittels GitHub Repositories.
- <https://github.com/tbz-it>

Demo: VMs für eine Klasse anlegen

- Um Routineaufgaben wie das Anlegen von mehreren Machines (VMs) pro Klasse zu automatisieren, stehen einen Reihe von [Helper Scripts](#) zur Verfügung.
- So können für eine ganze Klasse mit ein paar Befehlen Machines (VMs) erzeugt werden.
- Das Vorgehen ist dabei wie folgt:
 - Wechseln in den Rack Server cloud-hf-01
 - ssh ubuntu@10.9.39.8
 - Erstellen von 20 VMs für die Klasse st17a
 - cd lernmaas
 - createvms config.yaml m122 20 st17a
 - Öffnen des MAAS UI <http://10.9.39.8:5240/MAAS> im Browser, wechseln in den Resource Pool m122-st17a und alle anklicken, AZ zuweisen und alle Machines (VMs) deployen.
 - Nach dem deployen stehen alle Machines (VMs) im entsprechend VPN zur Verfügung.

Übung: Ausbildungsmodul in der Cloud anlegen

- Siehe: <https://github.com/mc-b/iac/blob/main/lernmaas-iac.md>

Verwendet das nachfolgende Cloud-init Script um eine Umgebung für das Modul 122 anzulegen.

Geht dabei gleich vor wie bei [Übung 1](#).

```
#cloud-config
hostname: m122-02
manage_etc_hosts: true
users:
  - name: ubuntu
    sudo: ALL=(ALL) NOPASSWD:ALL
    groups: users, admin
    home: /home/ubuntu
    shell: /bin/bash
    lock_passwd: false
    plain_text_passwd: 'password'
# login ssh and console with password
ssh_pwauth: true
disable_root: false
packages:
  - git
  - curl
  - wget
  - jq
  - markdown
  - nmap
  - traceroute
runcmd:
  - git clone https://github.com/mc-b/lernmaas /opt/lernmaas
  - sudo su - ubuntu -c "cd /opt/lernmaas && bash -x services/cloud-init.sh"
```

Übung: Ausbildungsmodul in der Cloud via CLI anlegen

- Siehe: <https://github.com/mc-b/iac/blob/main/lernmaas-iac-cli.md>

Azure Cloud

Anmelden an der Azure Cloud

```
az login
```

Anschliessend müssen folgende Aktionen ausgeführt werden:

- Erstellen einer Ressource Gruppe, wo unsere VMs abgelegt werden:
- Erstellen der VM
- Freigeben des Ports 80, damit wir via Browser auf die VM bzw. die installierten Services zugreifen können.

```
az group create --name mygroup --location southcentralus
az vm create --resource-group mygroup --name myvm --image UbuntuLTS --size Standard_D2_v4 --location southcentralus --custom-data
az vm open-port --port 80 --resource-group mygroup --name myvm
```

Überprüft das Ergebnis, durch Anwählen der IP-Adresse Eurer VM im Browser.

Um die VM zu löschen, genügt es, die Resource Gruppe zu löschen.

```
az group delete --name mygroup --yes
```

AWS Cloud

Einloggen in AWS Cloud

```
aws configure
```

```
AWS Access Key ID [*****WBM7*]:
AWS Secret Access Key [*****eKJA*]:
Default region name [eu-central-1]:
Default output format [None]:
```

Anschliessend müssen folgende Aktionen ausgeführt werden:

- Security Group erstellen und Ports öffnen
- Erstellen der VM

```
aws ec2 create-security-group --group-name mygroup --description "Standard Ports"
aws ec2 authorize-security-group-ingress --group-name mygroup --protocol tcp --port 22 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-name mygroup --protocol tcp --port 80 --cidr 0.0.0.0/0

aws ec2 run-instances --image-id ami-0767046d1677be5a0 --security-group-ids mygroup --instance-type t2.micro --count 1 --user-data
```

Anschliessend können wir uns die laufenden VMs anzeigen

```
aws ec2 describe-instances --output table
```

Überprüft das Ergebnis, durch Anwählen der IP-Adresse Eurer VM im Browser.

Zusammenfassung

- Infrastruktur als Code
 - Ist ein Paradigma zur Infrastruktur-Automation.
 - Basiert auf konsistenten und wiederholbaren Definitionen (Code) für die Bereitstellung von Systemen und deren Konfiguration.
- LernMAAS
 - Vereint "Infrastruktur als Code" Projekte zu einem grossen Ganzen
 - Erlaubt es mittels 2 Klicks zu einem voll funktionsfähigen Kurs/Modul zu kommen, sei es in der Privaten (MAAS.io) oder einer Public Cloud.

Abschluss

- Viel Erfolg mit "Infrastruktur als Code"!
- Marcel Bernet
 - <https://github.com/mc-b>

Azure Cloud: VM anlegen

Microsoft Azure

Dashboard > Ubuntu Server 18.04 LTS >

Create a virtual machine

Basics Disks Networking Management Advanced Tags Review

⚠️ Changing Basic options may reset selections you have made. Review all options prior to creating the virtual machine.

Create a virtual machine that runs Linux or Windows. Select an image from Azure marketplace. Complete the Basics tab then Review + create to provision a virtual machine with full customization. [Learn more](#)

Project details

Select the subscription to manage deployed resources and costs. Use resource groups to group your resources.

Subscription * ⓘ

Resource group * ⓘ [Create new](#)

Instance details

Virtual machine name * ⓘ

Region * ⓘ

Availability options ⓘ

Availability zone * ⓘ

Image * ⓘ [See all images](#)

Azure Spot instance

Size * ⓘ

Networking Disks Basics Management Advanced Tags Review

Define network connectivity for your virtual machine by configuring network interface ports, inbound and outbound connectivity with security group rules, or place behind a load balancer. [Learn more](#)

Network interface

When creating a virtual machine, a network interface will be created for you.

Virtual network * ⓘ [Create new](#)

Subnet * ⓘ [Manage subnet configuration](#)

Public IP ⓘ [Create new](#)

NIC network security group ⓘ None Advanced

Public inbound ports * ⓘ None Allow selected ports

Select inbound ports *

Advanced Disks Networking Basics Management Tags Review + create

Add additional configuration, agents, scripts or applications via virtual machine extensions or cloud-init.

Extensions

Extensions provide post-deployment configuration and automation. [Select an extension to install](#)

Custom data and cloud init

Pass a cloud-init script, configuration file, or other data into the virtual machine **while it is being provisioned**. The data will be saved on the VM in a known location. [Learn more about custom data for VMs](#)

Custom data

Custom data on the selected image will be processed by cloud-init. [Learn more about custom data and cloud init](#)

AWS Cloud: VM anlegen

Schritt 1: Auswählen eines Amazon Machine Images (AMI)

Ubuntu Server 20.04 LTS (HVM), SSD Volume
ami-05f7491af5eef733a (64-Bit x86) / ami-08f11f4114f566d1a (64-Bit Arm)

Zur kostenlosen Nutzung berechtigt

Schritt 2: Wählen eines Instance-Typs

Amazon EC2 bietet eine große Auswahl von Instance-Typen, die für unterschiedliche Anwendungsfälle geeignet sind. Instance-Typen umfassen verschiedene Kombinationen von Prozessoren, Arbeitsspeicher, Speicher und Netzwerkkapazität und sorgen so für Flexibilität bei der Auswahl der geeigneten Anwendungen. Weitere Informationen über Instance-Typen und wie sie Ihren Computeranforderungen entsprechen.

Familie	Typ	vCPUs	Arbeitsspeicher (GiB)	Instance-Speicher (GB)
t2	t2.nano	1	0.5	Nur EBS
<input checked="" type="checkbox"/>	t2.micro	1	1	Nur EBS

Schritt 3: Konfigurieren von Instance-Details

Aktivieren des Beendigungsschutzes

Schutz gegen versehentliches Beenden

Überwachung

Aktivieren der detaillierten Überwachung für CloudWatch

Mandantenverhältnis

Gemeinsam genutzt – Ausführen einer gemeinsam g

Guthabenspezifikation

Unbegrenzt

Dateisysteme

Dateisystem hinzufügen

Erweiterte Details

Aktivieren

Zugriff auf Metadaten

Aktiviert

Metadatenversion

V1 und V2 (Token optional)

Limit für Metadaten-Token-Antwort-Hop

1

Benutzerdaten

Als Text Als Datei Die Eingabe ist bereits base64-kodiert

(Optional)

Schritt 7: Überprüfen des Instance-Starts

Bitte überprüfen Sie Ihre Instance-Details. Sie können zurückgehen, um Änderungen für jeden Abschnitt zu ändern. Um Ihre Instance zu starten, klicken Sie auf den Button **Launch (Start)**.

AM-Details

Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-05f7491af5eef733a

Zur kostenlose Nutzung berechtigt

Ubuntu Server 20.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Root-Gerätetyp: ebs Virtualisierungstyp: hvm ENA-aktiv

Instance-Typ

Instance-Typ	ECUs	vCPUs	Arbeitsspeicher (GiB)	Instance-Speicher (GB)	EBS-Optimized
t2.micro	-	1	1	Nur EBS	-

Sicherheitsgruppen

Name der Sicherheitsgruppe: launch-wizard-2

Beschreibung: launch-wizard-2 created 2021-05-08T15:02:09.618+02:00

Typ	Protokoll	Port-Bereich	Quelle
SSH	TCP	22	0.0.0.0/0
HTTP	TCP	80	0.0.0.0/0
HTTP	TCP	80	::/0
HTTPS	TCP	443	0.0.0.0/0
HTTPS	TCP	443	::/0