



Flatcar Linux: what is new in this Container OS?

```
hcloud server create --name flatcar
```



Your speakers

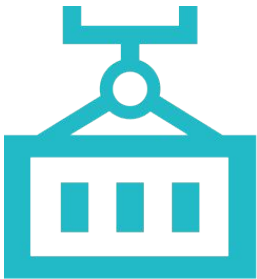


- Julian (@apricote)
- Open Source Integrations at Hetzner



- Mathieu (@tormath1)
- Works on Flatcar and related topics at Microsoft
- *SRE France*

But what is Flatcar, again?



Containers ready



Immutable



Automated updates

Containers ready

- Docker
- Containerd
- (opt-in: Podman, cri-o)

```
[Unit]
Description=NGINX example
After=docker.service
Requires=docker.service
[Service]
TimeoutStartSec=0
ExecStartPre=/usr/bin/docker rm --force nginx1
ExecStart=/usr/bin/docker run --name nginx1 --p
ExecStop=/usr/bin/docker stop
Restart=always
RestartSec=5s
[Install]
WantedBy=multi-user.target
```

```
7 | k8s-app: calico-kube-cont
6 spec:
5 # The controllers can only
4 replicas: 1
3 selector:
2 | matchLabels:
1 | | k8s-app: calico-kube-co
11 strategy:
1 | type: Recreate
2 template:
3 | metadata:
4 | | name: calico-kube-contr
5 | | namespace: kube-system
6 | | labels:
7 | | | k8s-app: calico-kube-controllers
8 | spec:
9 | | nodeSelector:
10 | | | kubernetes.io/os: linux
```

```
[Unit]
Description=A simple Nginx pod
Before=local-fs.target

[Kube]
Yaml=/etc/kubernetes/nginx.yaml
PublishPort=80:80

[Install]
# Start by default on boot
WantedBy=multi-user.target default.target
```



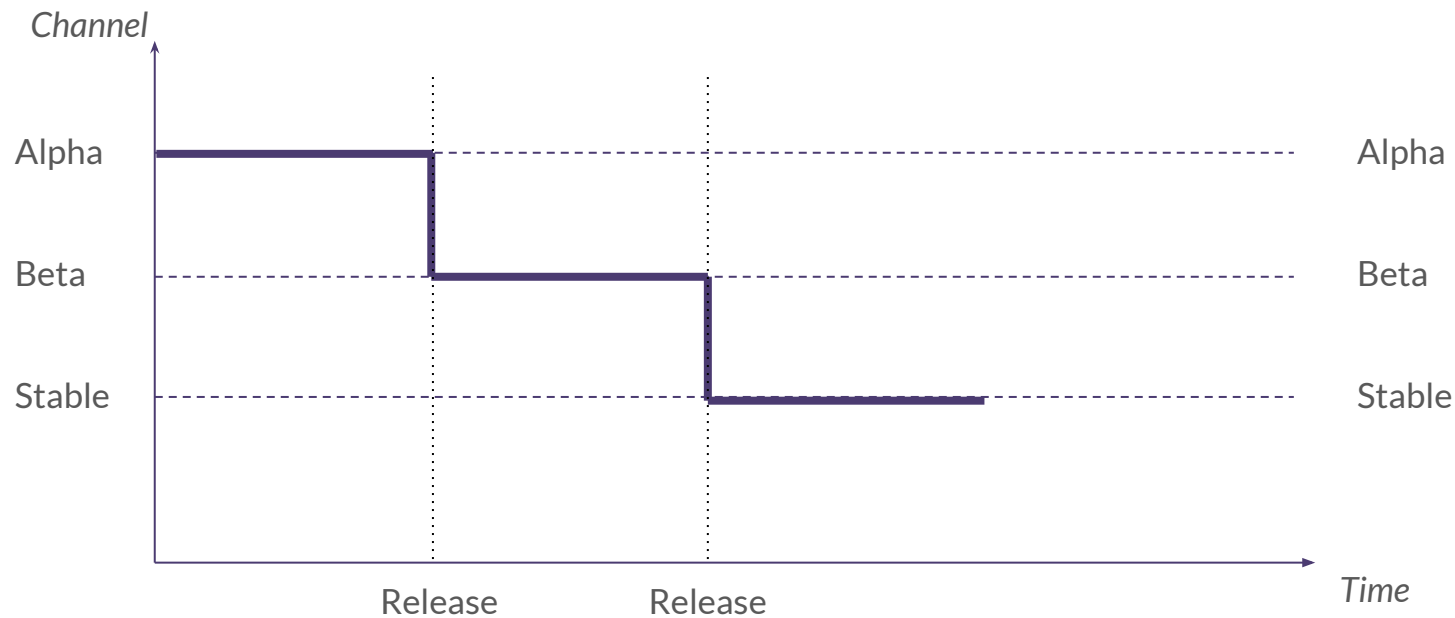
Immutable



```
core@localhost ~ $ apt
-bash: apt: command not found
core@localhost ~ $ emerge
-bash: emerge: command not found
core@localhost ~ $ yum
-bash: yum: command not found
core@localhost ~ $ dnf
-bash: dnf: command not found
core@localhost ~ $ touch /usr/foo
touch: cannot touch '/usr/foo': Read-only file system
core@localhost ~ $ sudo touch /usr/foo
touch: cannot touch '/usr/foo': Read-only file system
core@localhost ~ $ touch /opt/foo
touch: cannot touch '/opt/foo': Permission denied
core@localhost ~ $ sudo touch /opt/foo
core@localhost ~ $
```



(Glossary of the automatic updates)





(Glossary of the automatic updates)

3815.2.5



(Glossary of the automatic updates)

3815.2.5



Days since the first CoreOS release



(Glossary of the automatic updates)

Promotion level:

- 0 - Alpha
- 1 - Beta
- 2 - Stable
- 3 - LTS

3815.2.5

Days since the first CoreOS release



(Glossary of the automatic updates)

3815.2.5

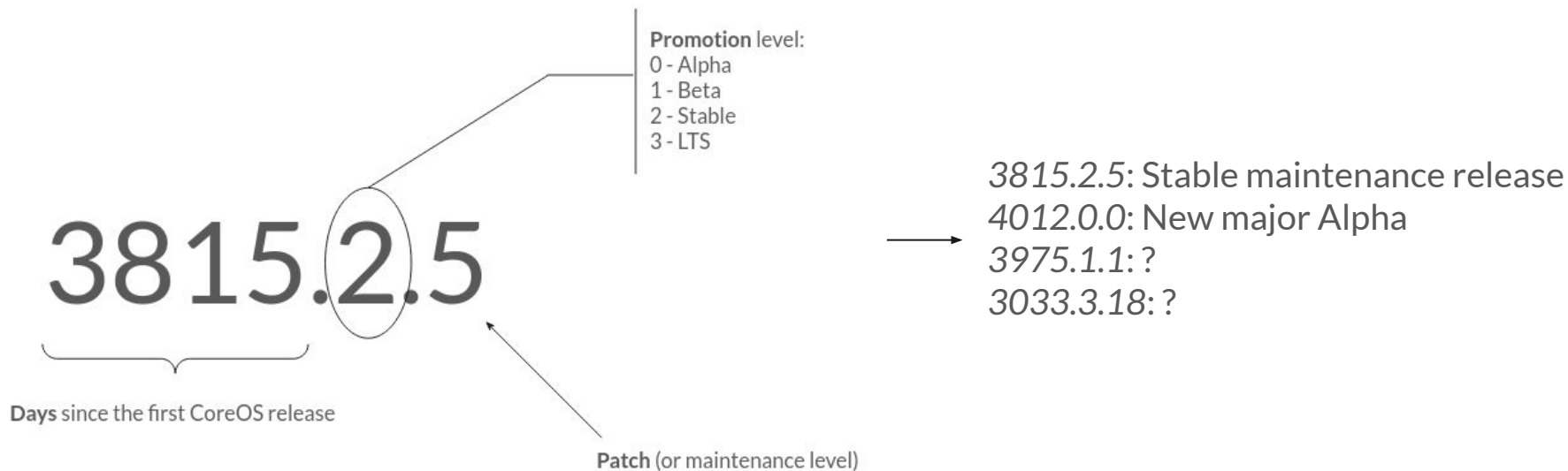
Days since the first CoreOS release

Promotion level:

- 0 - Alpha
- 1 - Beta
- 2 - Stable
- 3 - LTS

Patch (or maintenance level)

(Glossary of the automatic updates)





Automated updates



```
core@localhost ~ $ cat /usr/share/flatcar/update.conf  
SERVER=https://public.update.flatcar-linux.net/v1/update/  
GROUP=alpha
```



Example: RegreSSHion

- CVE-2024-6387 (OpenSSH)
- Monday: public announcement
- Tuesday: Flatcar users safe on all channels

No problem! I fully understand. And, besides compliance BS, I'm actually glad my servers were patched in a few hours! Thanks for all!

+2



2





Ignition

- Used by *Flatcar*, *Fedora CoreOS*, and *OpenSUSE MicroOS*
- **On first boot:** configure based on config file
- Produce the machine specified, or no machine at all
- User-friendly config transpiler: **Butane**



```
variant: fcos
version: 1.5.0
passwd:
  users:
    - name: core
      ssh_authorized_keys:
        - ssh-ed25519 ...
systemd:
  units:
    - name: foo.service
      enabled: false
      contents: |
        ...
storage:
  files:
    - path: /etc/chrony.conf
      overwrite: true
      mode: 0644
      contents:
        inline: |
          foo
```

butane < config.yaml > config.json



```
{
  "ignition": {
    "version": "3.4.0"
  },
  "passwd": {
    "users": [
      {
        "name": "core",
        "sshAuthorizedKeys": [
          "ssh-ed25519 ..."
        ]
      }
    ]
  },
  "storage": {
    "files": [
      {
        "overwrite": true,
        "path": "/etc/chrony.conf",
        "contents": {
          "compression": "",
          "source": "data:foo%0A"
        },
        "mode": 420
      }
    ]
  },
  "systemd": {
    "units": [
      {
        "contents": "...\\n",
        "enabled": false,
        "name": "foo.service"
      }
    ]
  }
}
```



Now, what is new in Flatcar?




Cluster API

TL; DR: Deploy Kubernetes from Kubernetes



Cluster API - Ignition support



```
apiVersion: controlplane.cluster.x-k8s.io/v1beta1
kind: KubeadmControlPlane
metadata:
  name: ${CLUSTER_NAME}-control-plane
spec:
  format: ignition
  ignition:
    containerLinuxConfig:
      additionalConfig: |
        systemd:
          units:
            - name: coreos-metadata-sshkeys@.service
              enabled: true
            - name: kubeadm.service
              enabled: true
            ...
```



Cluster API - Flatcar templates

- AWS
- vSphere
- (Hetzner)
- (Known to work with Tinkerbell)
- Azure
- OpenStack
- Akamai/Linode

systemd-sysext

```
$ sudo systemd-dissect --with flatcar-zfs.raw tree -d
├─ usr
│   ├── bin
│   ├── lib
│   │   ├── extension-release.d
│   │   ├── modules
│   │   │   └─ 6.6.36-flatcar
│   │   │       └─ extra
│   │   ├── modules-load.d
│   │   └─ systemd
│   │       ├── system
│   │       │   ├── multi-user.target.d
│   │       │   ├── multi-user.target.wants
│   │       │   ├── systemd-udev.service.d
│   │       │   ├── zfs-import.target.wants
│   │       │   ├── zfs.target.wants
│   │       │   └─ zfs-volumes.target.wants
│   │       ├── system-generators
│   │       ├── system-preset
│   │       └─ tmpfiles.d
│   └─ ...
```

```
core@localhost ~ $ zfs version
-bash: /usr/sbin/zfs: No such file or directory
core@localhost ~ $ sudo systemd-sysext merge
Using extensions 'containerd-flatcar', 'docker-flatcar', 'flatcar-zfs', 'oem-qemu'.
Merged extensions into '/usr'.
core@localhost ~ $ zfs version
zfs-2.2.2-r1-gentoo
zfs-kmod-2.2.2-r0-gentoo
core@localhost ~ $ mount | grep sysext
sysext on /usr type overlay (ro,nodev,relatime,seclabel,lowerdir=/run/systemd/sysext/meta/usr:/run/systemd/sysext/extensions/oem-qemu/usr:/run/systemd/sysext/extensions/flatcar-zfs/usr:/run/systemd/sysext/extensions/docker-flatcar/usr:/run/systemd/sysext/extensions/containerd-flatcar/usr:usr)
```



systemd-sysext - opt-out extensions



```
core@localhost ~ $ docker ps
-bash: docker: command not found
core@localhost ~ $ sudo crictl --runtime-endpoint unix:///run/containerd/containerd.sock pull nginx
Image is up to date for sha256:a72860cb95fd59e9c696c66441c64f18e66915fa26b249911e83c3854477ed9a
core@localhost ~ $ sudo crictl --runtime-endpoint unix:///run/containerd/containerd.sock images
```

IMAGE	TAG	IMAGE ID	SIZE
docker.io/library/nginx	latest	a72860cb95fd5	71MB




systemd-sysext - community extensions



```
core@localhost ~ $ docker
-bash: docker: command not found
core@localhost ~ $ ctr
-bash: ctr: command not found
core@localhost ~ $ sudo crictl --runtime-endpoint unix:///var/run/crio/crio.sock version
Version: 0.1.0
RuntimeName: cri-o
RuntimeVersion: 1.28.4
RuntimeApiVersion: v1
```



systemd-sysext - provisioning



```
variant: flatcar
version: 1.0.0
storage:
  links:
    - path: /etc/extensions/docker-flatcar.raw
      target: /dev/null
      overwrite: true
    - path: /etc/extensions/containerd-flatcar.raw
      target: /dev/null
      overwrite: true
  files:
    - path: /etc/extensions/crio.raw
      contents:
        source: https://github.com/flatcar/sysext-bakery/releases/download/latest/crio-1.28.4-
x86-64.raw
```

Flatcar Linux on Hetzner Cloud



Flatcar Container Linux 🚚

@flatcar@hachyderm.io

New Flatcar Alpha, Beta, Stable releases now available!



Many package updates: Linux, ca-certificates, glibc



CVE fixes & security patches: Linux, glibc, curl



Podman sysexit for Alpha



Hetzner images now available



Release notes at the usual spot: flatcar.org/releases/

May 30, 2024, 12:35 · 🌐 · Web · ↻ 8 · ★ 9



Hetzner Background

- German hosting provider
- 3 Datacenter parks in Europe + 3 Cloud-only locations in the US+Singapore
- Hetzner Cloud is an **Infrastructure-as-a-Service** provider
 - VMs, Block Storage, Load Balancers, Private Networks



Running Flatcar on Hetzner Cloud

- No API to directly upload images
 - **Workaround:** Server -> Rescue -> Overwrite Root Disk -> Snapshot
 - **Solutions:** [Packer](#) or [hcloud-upload-image](#)



Flatcar on Hetzner Cloud Demo

Adapted from [Flatcar Docs](#)

```
+ flatcar-demo export HCLOUD_TOKEN=$(hcloud config get token --allow-sensitive)
+ flatcar-demo export CHANNEL=stable
+ flatcar-demo IMAGE_URL=https://${CHANNEL}.release.flatcar-linux.net/arm64-usr/current/flatcar_production_hetzner_image.bin.bz2
+ flatcar-demo hcloud-upload-image upload \
  --architecture=arm \
  --compression=bz2 \
  --image-url=${IMAGE_URL} \
  --description flatcar-${CHANNEL}-arm
# Step 1: Generating SSH Key run-id="29353b62"
# Step 2: Creating Server run-id="29353b62"
█
```



Adding a new Platform to Flatcar Linux

Using Hetzner Cloud as an example



Platform Integration Steps

- ☒ Ignition support (>3913.0.0)
- ☒ Afterburn support
- ☒ OEM support on Flatcar (i.e build images with Hetzner OEM ID + additional configurations like starting afterburn service for example): [🔗 OEM: Provide Hetzner Images](#) scripts#1880
- ☒ Documentation: [🔗 docs\(hetzner\): update for new OEM images](#) flatcar-website#336
- ☐ Mantle support (test suite): [🔗 \[wip\] platform: add Hetzner](#) mantle#533
- ☐ Test automation on Flatcar: [🔗 ci-automation: add hetzner testing](#) scripts#2142



Platform/OEM Identifier

- Behaviour of many components depends on the platform
- Choose a single platform identifier that is used in all places
- For us: **hetzner**



Metadata Service

- When creating a VM, the user can choose to install an **authorized SSH Key**, and set some arbitrary “**user-data**”
- Made available through the *Metadata Service*
- On the VM through a link-local IPv4 address

Example: Hostname

```
1 $ curl http://169.254.169.254/hetzner/v1/metadata/hostname
2 my-server
```



Ignition

- Platform Integration
 - Read config file from platform-specific file / URL
- Hetzner
 - Read from the Instance Metadata user-data endpoint
 - GET `http://169.254.169.254/hetzner/v1/userdata`
=> Parse => Pass to ignition



Afterburn

- “one-shot agent for cloud-like platforms”
- **Hostname, network configuration**, authorized **SSH Keys**, “attributes” from instance metadata, **boot check-in**
- Used in *Flatcar* and *Fedora CoreOS*



Afterburn Attributes

- AFTERBURN_HETZNER_AVAILABILITY_ZONE
- AFTERBURN_HETZNER_HOSTNAME
- AFTERBURN_HETZNER_INSTANCE_ID
- AFTERBURN_HETZNER_PUBLIC_IPV4
- AFTERBURN_HETZNER_PRIVATE_IPV4_0
- AFTERBURN_HETZNER_REGION



Flatcar OEM Images > flatcar/scripts

- **common-oem-files**
 - Sets Kernel Parameters with Platform ID through grub
- **oem-\$platform**
 - Pre-installed sysexr for any additional files or agents for the Platform
 - amazon-ssm-agent, WALinuxAgent, ...
- **vm_image_util.sh**
 - Platform ID, Disk Layout, Disk Format (raw, qcow2, vmdk), which sysexr and package to install



Building Images

- Flatcar provides an SDK (Container Image) for this (flatcar.org/docs)
- `./run_sdk_container -t`
`./build_packages`
`./build_image`
`./image_to_vm.sh --from=$image_path --format=$platform`
- Upload & test
- Open pull requests with the proposed changes
- Ask a maintainer to build the images in CI for you



Documentation

- Features are worthless if no one knows how to use them
- Topics
 - Tutorial for deploying a Flatcar server on your provider
 - Which platform features work, and which do not
- For Hetzner Cloud
 - ❌ SSH Key optional => initial root password
 - ❌ Volume automount



Tests

- Flatcar test suite for OS functionality: **mantle**
- Runs on different platforms
- Needs glue code for your platforms APIs
- Donate resources for tests before each release
- Tested platforms can be marked as “Official” vs “Community Supported”

Cloud Providers

- [Amazon EC2](#)
- [Microsoft Azure](#)
- [Google Compute Engine](#)
- [Equinix Metal](#)
- [VMware](#)
- [DigitalOcean](#)
- [Hetzner](#)
- [OpenStack](#)
- [Brightbox](#)
- [Scaleway](#) (community support)
- [OVHcloud](#) (community support)
- [Akamai](#) (community support)



Platform Integration Tips & Tricks

- Join the Matrix channel and ask questions
- Find an experienced maintainer to help guide you through the process
- Invest time into your dev setup, low iteration time is very valuable
- Experiment: Try to make changes and see what happens



Links and resources

Thanks a lot - see you on
the Flatcar booth!



Generated with Libre QR from Framasoft

Appendix



Afterburn

```
pub trait MetadataProvider {  
    fn attributes(&self) -> Result<HashMap<String, String>>  
    fn hostname(&self) -> Result<Option<String>>  
    fn ssh_keys(&self) -> Result<Vec<PublicKey>>  
    fn networks(&self) -> Result<Vec<network::Interface>>  
    fn boot_checkin(&self) -> Result<()>  
}
```



Flatcar OEM Images > Afterburn Services

- **Hostname** - flatcar/bootengine
 - Enable `flatcar-metadata-hostname.service` for \$platform
- **Authorized SSH Keys** - flatcar/init
 - Enable `sshkeys.service` for \$platform
- **Attributes** - flatcar/scripts
 - Enable `coreos-metadata.service` for \$platform



Cluster API





Cluster API - Management Cluster



```
$ kind get nodes  
kind-control-plane
```



```
$ kubectl get pods -A -o name --field-selector "metadata.namespace!=kube-system"  
pod/caaph-controller-manager-648bb96cbd-9m579  
pod/capi-kubeadm-bootstrap-controller-manager-6d667d9766-27nz5  
pod/capi-kubeadm-control-plane-controller-manager-7b9b59c8f7-z8php  
pod/capi-controller-manager-6459498466-4zxr8  
pod/capl-controller-manager-5d4cbf977b-5mjdc  
pod/cert-manager-79d7c5cb6c-nfh5c  
pod/cert-manager-cainjector-5cf7fdcd9c-qpc1s  
pod/cert-manager-webhook-5df5556dd7-jc4rl  
pod/local-path-provisioner-7577fdbbfb-zlvbc
```



Cluster API - Provider implementation

NodeBalancers

Label ^	Backend Status
test-cluster	3 up - 0 down

Linodes

Label ^	Status ⚙
test-cluster-control-plane-rq56w	● Provisioning (0%)
test-cluster-control-plane-xjrvs	● Running
test-cluster-md-0-7vmpw-5wsnn	● Provisioning (0%)
test-cluster-md-0-7vmpw-vq98t	● Running



Cluster API - Workload cluster



```
$ kubectl --kubeconfig test-cluster.kubeconfig get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
test-cluster-control-plane-bn9v4	Ready	control-plane	8m28s	v1.30.0
test-cluster-control-plane-rq56w	Ready	control-plane	10m	v1.30.0
test-cluster-control-plane-xjrvs	Ready	control-plane	13m	v1.30.0
test-cluster-md-0-7vmpw-5wsnn	Ready	<none>	10m	v1.30.0

```
$ kubectl --kubeconfig=test-cluster.kubeconfig get nodes -o yaml | yq ".items[0].status.nodeInfo"
```

architecture: amd64
bootID: 3ab04310-2b8f-4011-97a8-c8d396decbeb
containerRuntimeVersion: containerd://1.7.18
kernelVersion: 6.6.36-flatcar
kubeProxyVersion: v1.30.0
kubeletVersion: v1.30.0
machineID: be484f288e1041c2aac0ccb44b9163a8
operatingSystem: linux
osImage: Flatcar Container Linux by Kinvolk 4012.0.1 (Oklo)
systemUUID: be484f288e1041c2aac0ccb44b9163a8



Flatcar on Hetzner Cloud Demo

```
export HCLOUD_TOKEN=<your token>
export CHANNEL=stable
IMAGE_URL=https://${CHANNEL}.release.flatcar-linux.net/amd64-usr/current/flatcar_production_hetzner_image.bin.bz2
```

```
hcloud-upload-image upload \
  --architecture=x86 \
  --compression=bz2 \
  --image-url=${IMAGE_URL} \
  --description flatcar-${CHANNEL}-x86
```




Flatcar on Hetzner Cloud Demo

(Minimal example, does not work directly)

```
version: 1.0.0
variant: flatcar
systemd:
  units:
    - name: nginx.service
      enabled: true
      contents: |
        [Service]
        EnvironmentFile=/run/metadata/flatcar
        ExecStartPre=-docker rm --force nginx1
        ExecStartPre=-bash -c "echo \"Hello from $COREOS_HETZNER_HOSTNAME\" > /var/www/index.html"
        ExecStart=docker run --name nginx1 --volume /var/www:/usr/share/nginx/html --net host nginx:1
        ExecStop=/usr/bin/docker stop nginx1
```

```
butane < nginx-example.yaml > nginx-example.json
```



Flatcar on Hetzner Cloud Demo

SNAPSHOT_ID=<ID from hcloud-upload-image>

SSH_KEY_ID=<your ssh key id>

```
hcloud server create \  
  --name flatcar-test \  
  --type cpx11 \  
  --location fsn1 \  
  --image ${SNAPSHOT_ID} \  
  --ssh-key ${SSH_KEY_ID} \  
  --user-data-from-file nginx-example.json
```



Flatcar on Hetzner Cloud Demo

```
local$ curl $(hcloud server ip flatcar-test)
Hello from flatcar-test
local$ hcloud server ssh -u core flatcar-test
server$ systemctl status nginx.service
```