PUPPENSPIELE IN DER WOLKE

HOW TO BRING A NON-CLOUD APP INTO THE SKY

WER?

NORMAN MEßTORFF

Operations Engineer @
Deutsche Post E-Post Development

Infrastructure & Deployment

Puppet, OpenNebula, Python

VIM enthusiast

STELLE DIR VOR...

Das gesamte Budget wird in ein
Virtualisierungs Projekt geworfen

VM's werden nun innerhalb weniger Sekunden erstellt!

Keine Sorge über Ressourcenmangel!
(zumindest jetzt, es ist noch neu 😏)

EINEN MONAT SPÄTER

Jede IT Abteilung kennt dich &
möchte „mal eben“ eine neue VM

Du hast keine Zeit für andere Dinge

Jede VM wird eine Schneeflocke,
Nacharbeiten fallen an

Alle Fragen sich, warum das so lange dauert.

Das war doch so teuer,

der „Cloud Cluster“.

EINE SCHNEEFLOCKE ENTSTEHT

- Neue VM anlegen (die teure, neue Technik)
- DNS Einträge setzen
- Firewallregeln einpflegen
- Puppet Zertifikat signieren
  (oh, wait. Implement a configuration Management…)
- Monitoring konfigurieren
- Neuen Knoten in Loadbalancer einhängen

„Als Cloud Cluster bezeichnet man große Gewitter- und Schauergebiete…“

— Wikipedia: „Cloud Cluster“

VIRTUALISIERUNG VS. CLOUD

- Vollautomatisierung
- Skaliert dynamisch ohne Probleme
- VMs mit kurzer Lebenserwartung

Du möchtest deine Applikation mit mit einem Klick einsatzbereit haben.

Virtualisierung allein stellt dir ein Betriebssystem mit einem Klick zur Verfügung.

OPENNEBULA

- Cloud Management
- Herstellerunabhängig
- Hybrid, aber kein Toyota.

OPENNEBULA

OpenNebula.org

- Develop & innovate
- Support the community
- Collaborate

Large-scale production deployment: 16,000 VMs

OpenNebula Conf
24. - 26. Sept 2013
Berlin

2005 2008 2009 2010 2011 2012 2013 2014

Research Project

SEVENTH FRAMEWORK PROGRAMME

C12G LABS

5,000 downloads/ month

OPENNEBULA

- Provisionierung / Virtualisierung
- Storage
- Netzwerk / IP-Management
- Ressourcen Monitoring

Du möchtest dir keine Gedanken machen,
auf welchem Wirt noch Ressourcen frei sind.

WAS IST EINE VM?

Definition:

- Image
- Ausstattung (CPU & RAM)
- Netzwerke
- Zustand (running, suspended, shutdown, unknown)

IMAGES

- Image sollte bereits existieren
- cloudinit, lokale VirtualBox, VM aus dem Marketplace
- Import erfolgt in ein Datastore

DATASTORES

- Jeder Wirt hat mindestens einen Datastore
- Anbindungsarten: SSH, NFS, CEPH…
- Shared Storage vs. local Storage

NETZWERK

- VLAN & Bridge Konfiguration auf dem Wirt
- IP-Adressvergabe
- Kodierung in MAC Adresse

NETZWERK

MAC Adresse:

02:00:C0:A8:00:07

Prefix 192.168.0.7

Dadurch keine doppelten MAC-Adressen.

KONTEXTUALISIERUNG

Setzen von Hostnamen

MAC-Adresse 2 IP

Lokale DNS Konfiguration

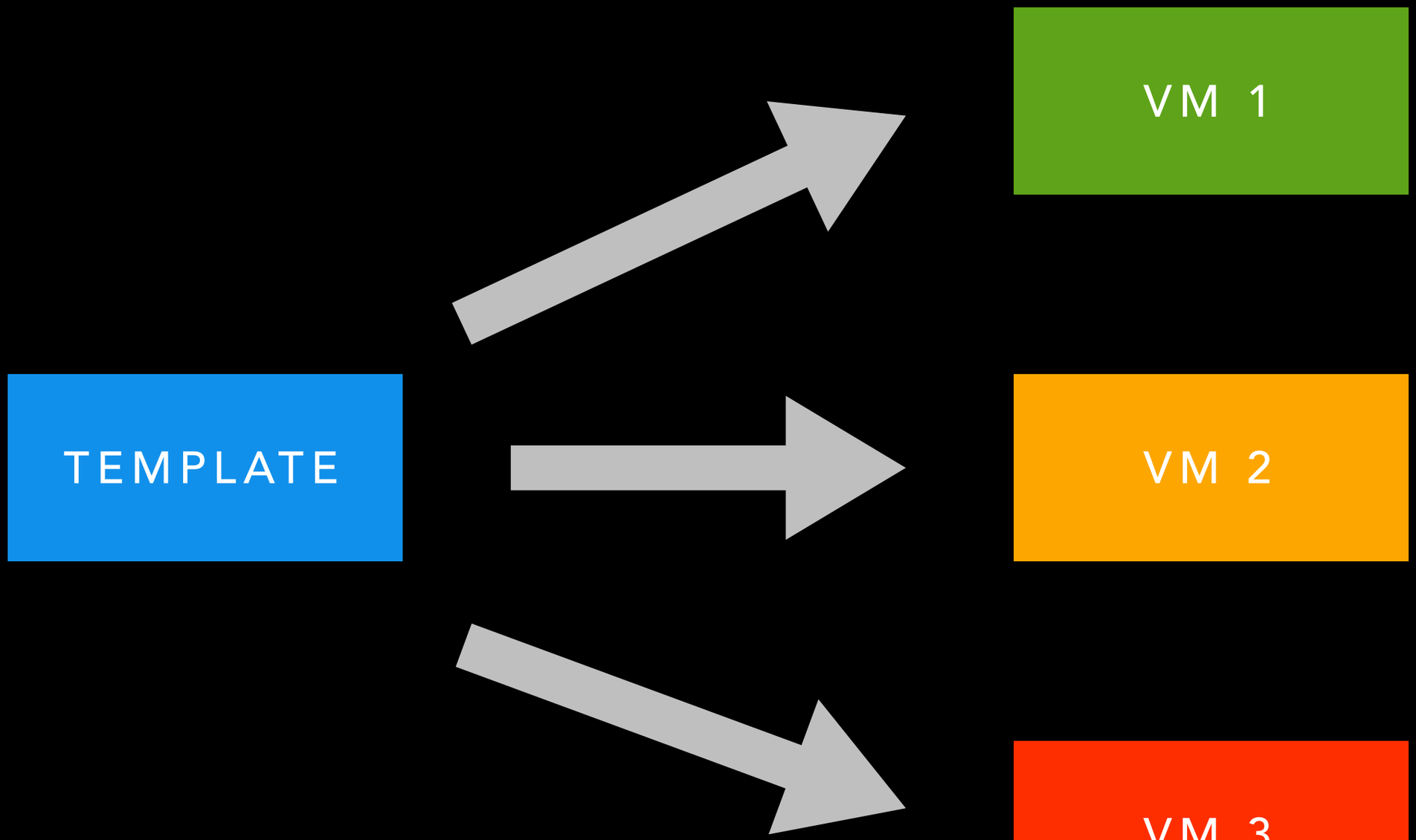
SSH Key für root User

Benutzerdefinierte Scripte

TEMPLATES

- Image
- Ausstattung (CPU & RAM)
- Netzwerke
- Kontextualisierung

TEMPLATES



VM BOOTVORGANG

VM wird aus Template instanziiert. Bedeutet:

- Wirt auswählen, Image & Netzwerk bereitstellen
- VM starten (Kontextualisierung)
- Ressourcen überwachen

PUPPET

- Ein definierter Systemzustand wird beliebig oft herbeigeführt

- Definition gilt Lokal, nicht übergreifend

- Puppet Master erstellt einen Katalog,
  der vom Puppet Agent lokal geprüft wird

PUPPET

```
package{ 'httpd':
  ensure  => latest,
}

service{ 'httpd':
  ensure  => running,
  enable  => true,
  require => Package['httpd'],
}
```

PUPPET

- Basiskonfiguration (SSH, Routing, Standardsoftware)
- Application Deployment
- Tomcat
- Applikation

PUPPET

1. VM Provisionieren
2. Kontextualisierung
3. Puppet Rollout
4. Service steht zur Verfügung

PUPPET & OPENNEBULA

Auch OpenNebula installiert sich (fast) von alleine:

https://github.com/epost-dev/opennebula-puppet-module

APPLICATION DEPLOYMENT

Alles ist in einem RPM:

- SSH, Apache, Tomcat…
- Applikation
- Monitoring NRPE Checks
- Puppet Manifeste!

LOADBALANCER

Fertige Puppet Module:

- F5 Big IP
- HAProxy

FIREWALL

Lokal:

- puppet-firewall
- Shorewall

Zentral:

- Juniper
- Cisco

MAGISCHER KLEBSTOFF

OpenNebula Hook:

- Puppet Zertifikate & Puppet DB
- DNS

MAGISCHER KLEBSTOFF

Puppet:

- Application Deployment
- Loadbalancer
- Firewall

Cloud oder nicht Cloud.

CLOUDFÄHIGKEIT

Zustandslos:

- keine lokale Datenhaltung
- Zugriffe unabhängig (wo ist die Session?)

CLOUDFÄHIGKEIT

Deployment:

- kompatible Schnittstellen
- „weiche“ DB Änderungen

CLOUDFÄHIGKEIT

Service Discovery:

- Wer sind meine Nachbarn?
- DNS: wann erfolgt die Auflösung wirklich?

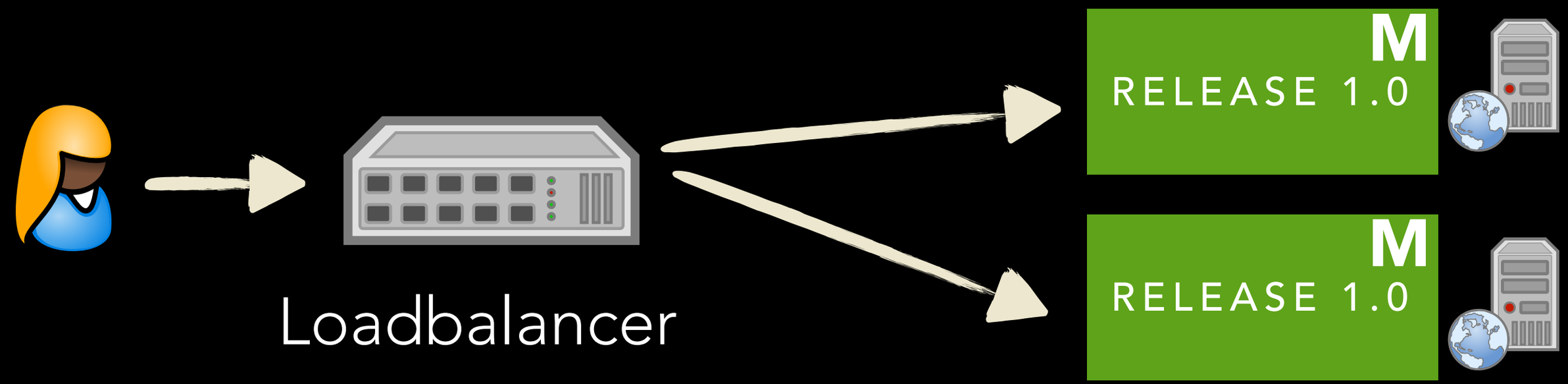
DEPLOYMENT

Deployment Szenario:

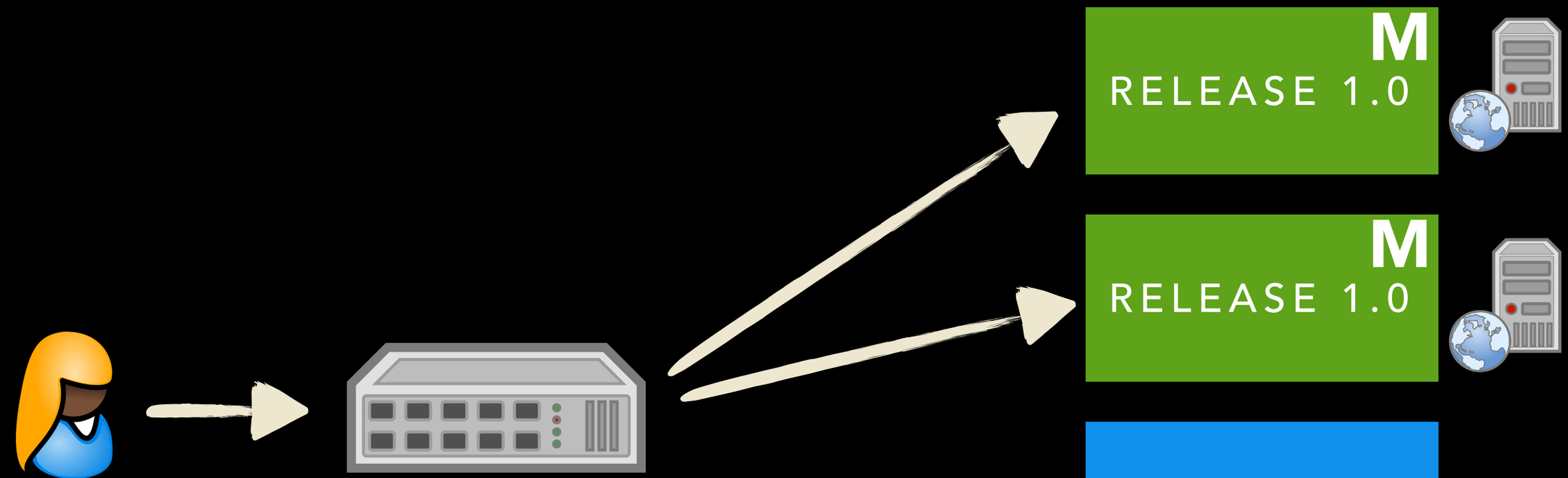
2 bestehende VMs mit Release Stand 1.0

Update auf Release 1.1, ohne Außenwirkung

DEPLOYMENT



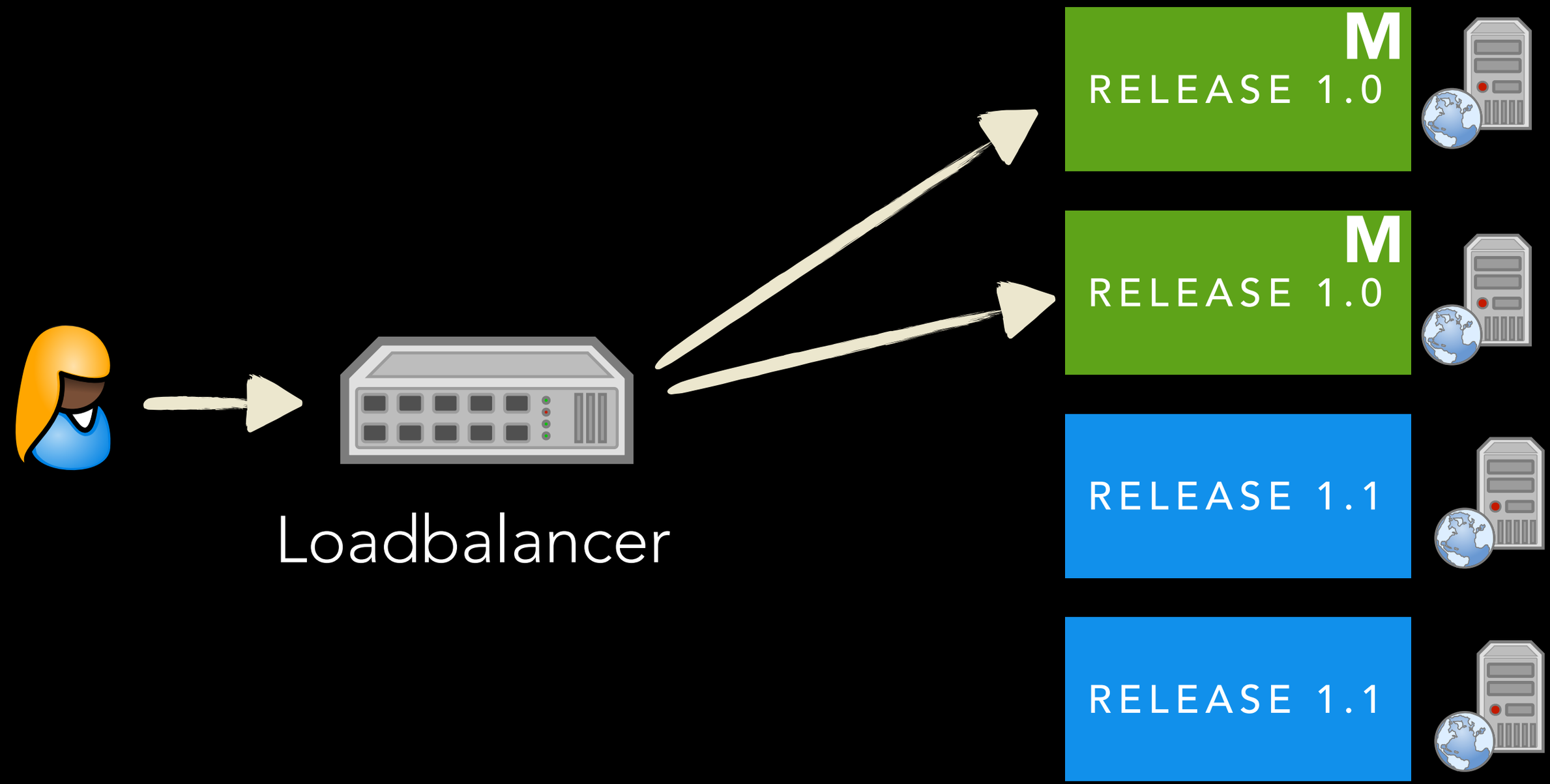
DEPLOYMENT



Loadbalancer

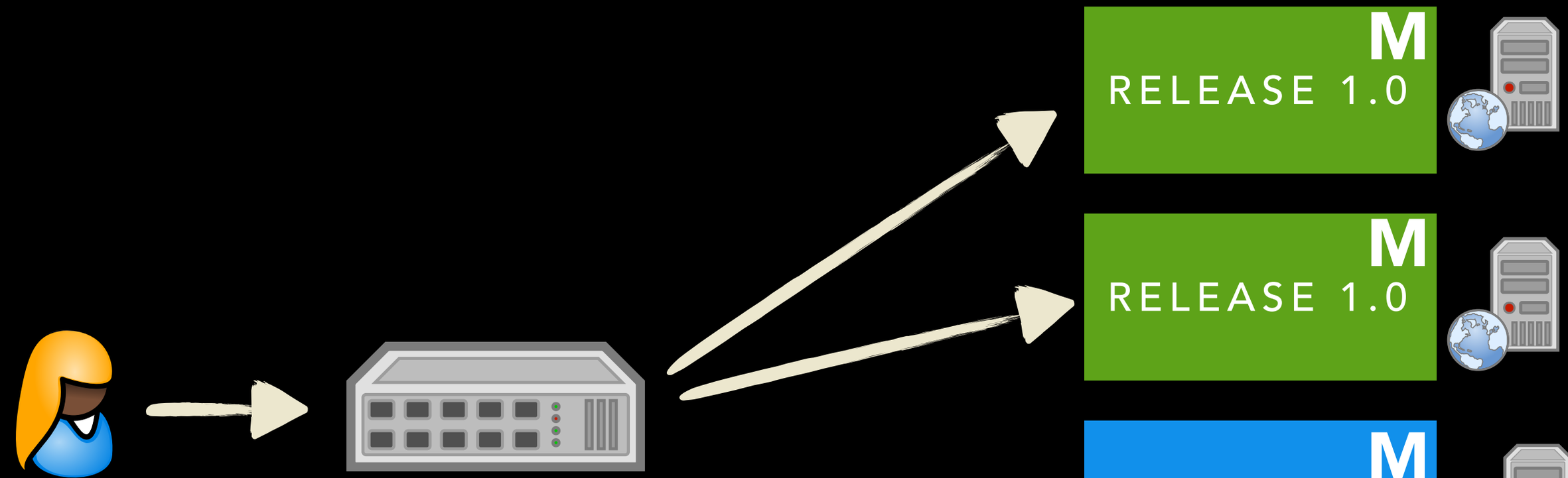
2 neue VMs Provisionieren

DEPLOYMENT



Application Deployment via Puppet

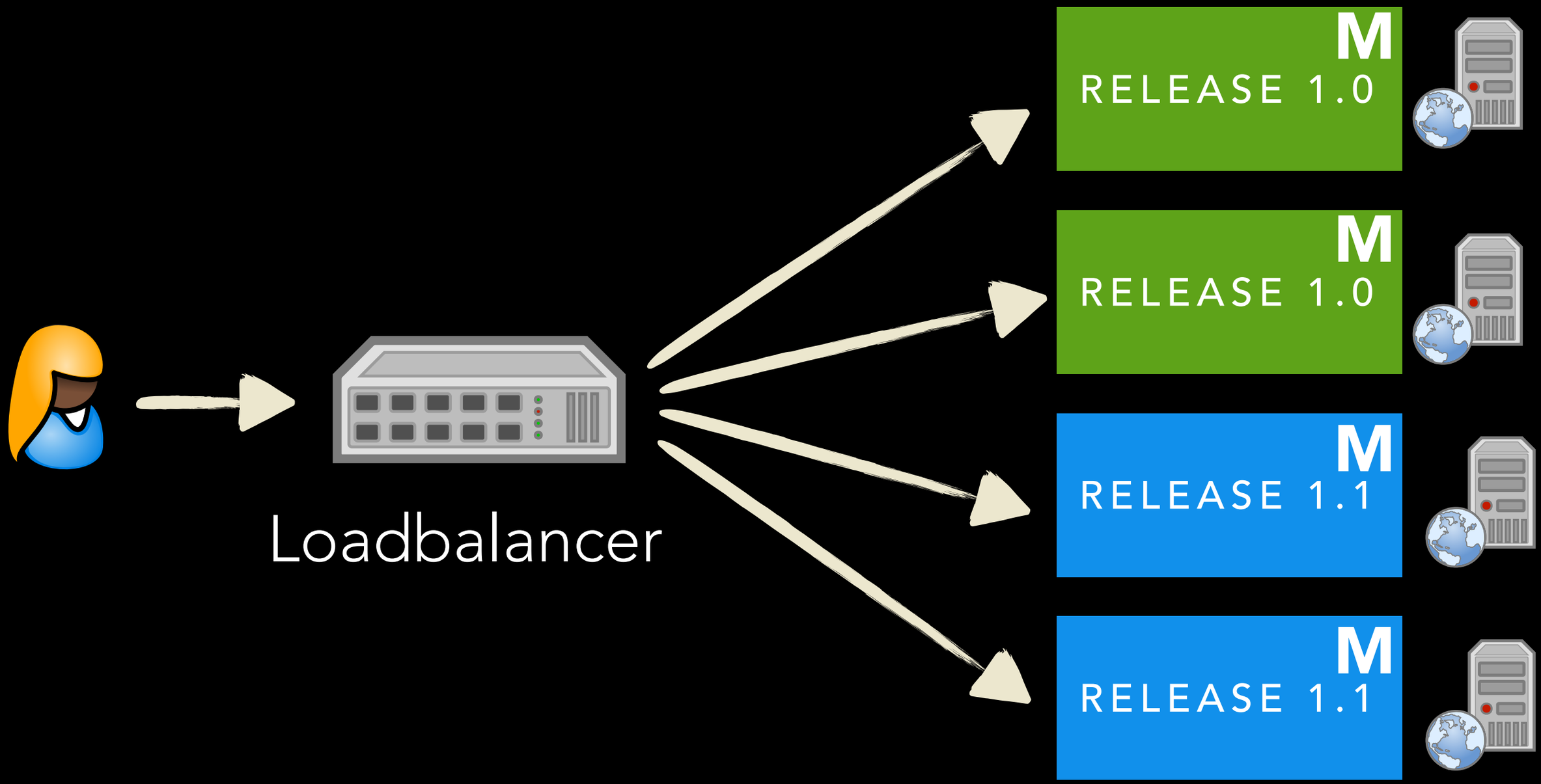
DEPLOYMENT



Loadbalancer

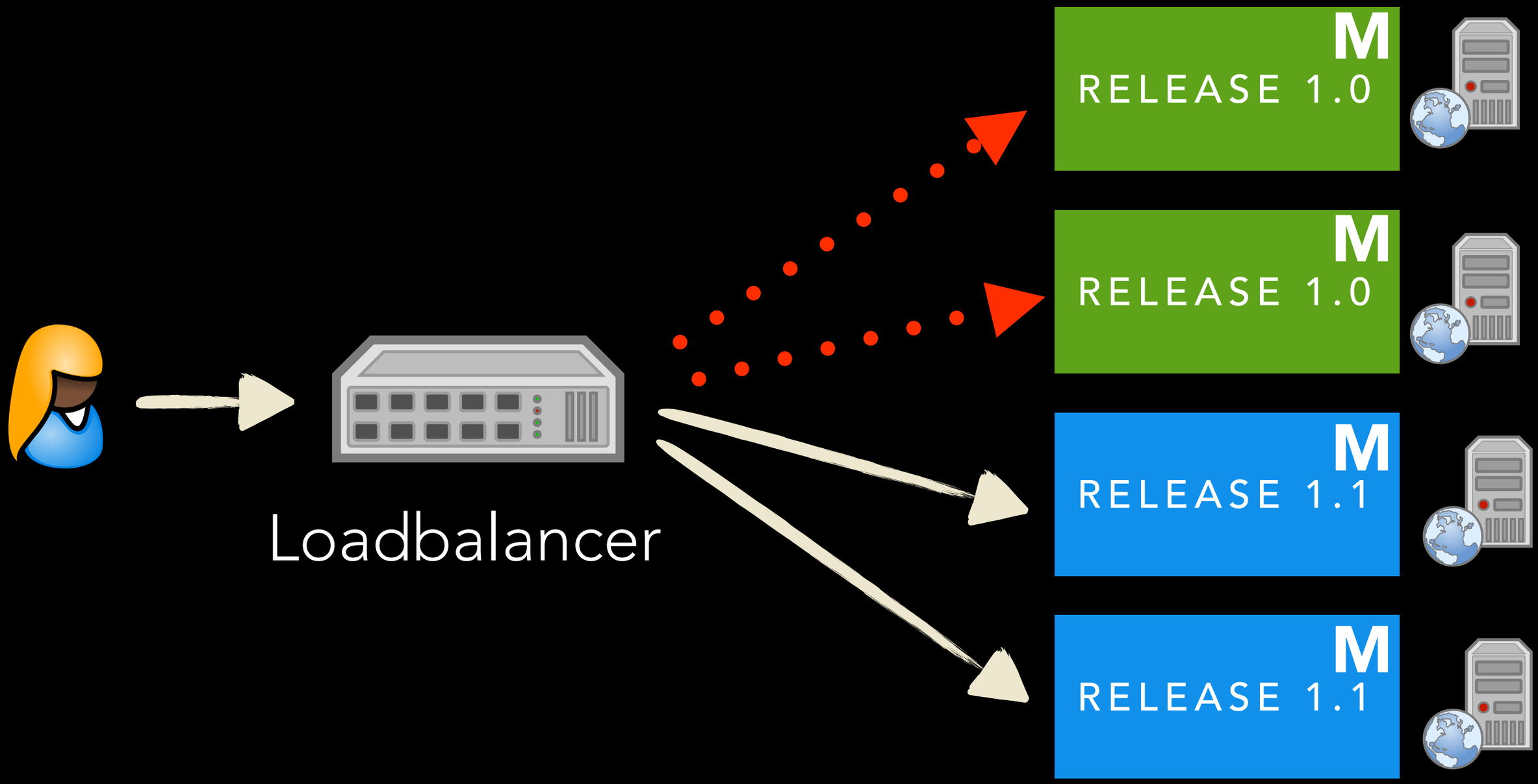
VMs in Monitoring aufnehmen

DEPLOYMENT



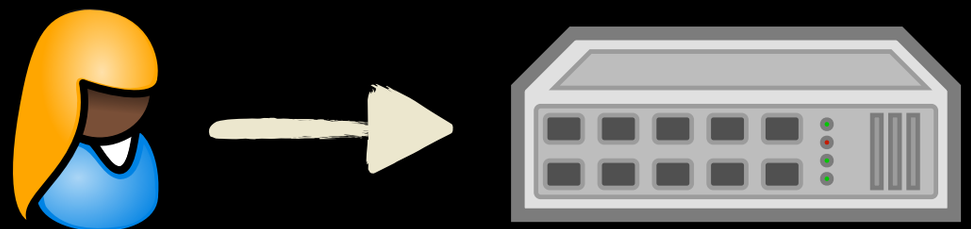
VMs in Loadbalancer einbinden

DEPLOYMENT



Vorhandene Sessions ausbluten lassen

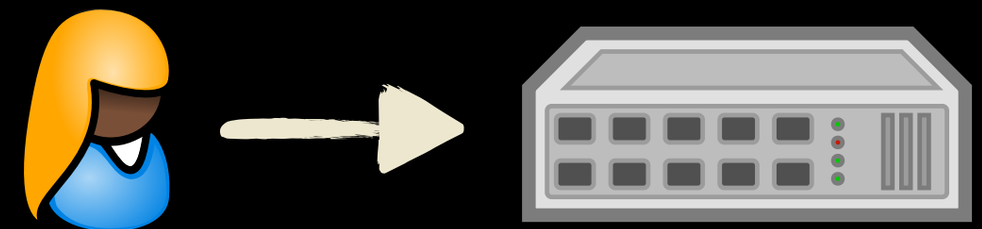
DEPLOYMENT



Loadbalancer

ehemalige VMs Loadbalancing nehmen

DEPLOYMENT



Loadbalancer

RELEASE 1.0

RELEASE 1.0

M

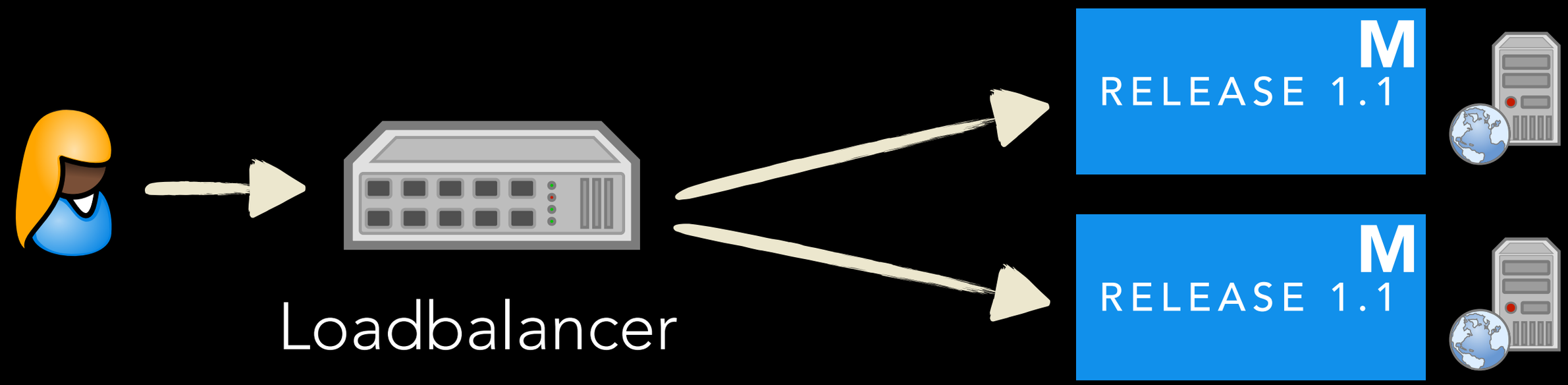
RELEASE 1.1

M

RELEASE 1.1

ehemalige VMs aus Monitoring entfernen

DEPLOYMENT



Loadbalancer

Deprovisionierung ehemaliger VMs

DEPLOYMENT

Täglicher Beweis, dass die Automatisierung funktioniert.

Routine.

Security: es kann sich nichts lange einnisten.

Kein Backup notwendig.

SUMMARY

- Virtualisierung != Cloud
- Konfigurationsmanagement oft nicht übergreifend
- Magischer Klebstoff: Puppet & OpenNebula Hooks
- Orchestrierung von Deployments

THX.

Q & A ?

Norman Meßtorff
@nmesstorff
normes@normes.org