

## The MNM-CloudLab -- Ideas, Concepts & Implementation

**Nils gentschen Felde**  
**MNM-Team, Ludwig-Maximilians-Universität München**

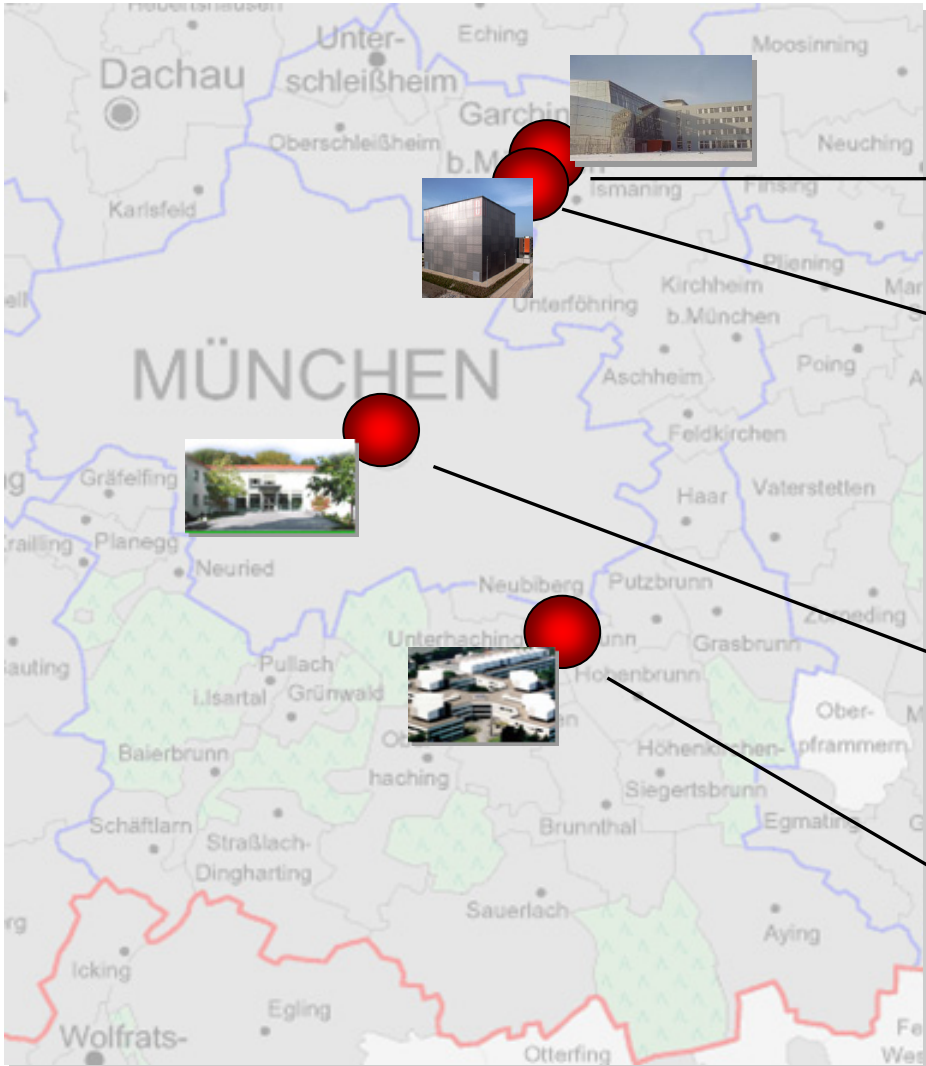


CENTRE DE COOPÉRATION  
UNIVERSITAIRE FRANCO-BAVAROIS



Université  
franco-allemande  
Deutsch-Französische  
Hochschule

# The MNM Team



Technische Universität München



Leibniz-Rechenzentrum  
der Bayerischen Akademie der Wissenschaften

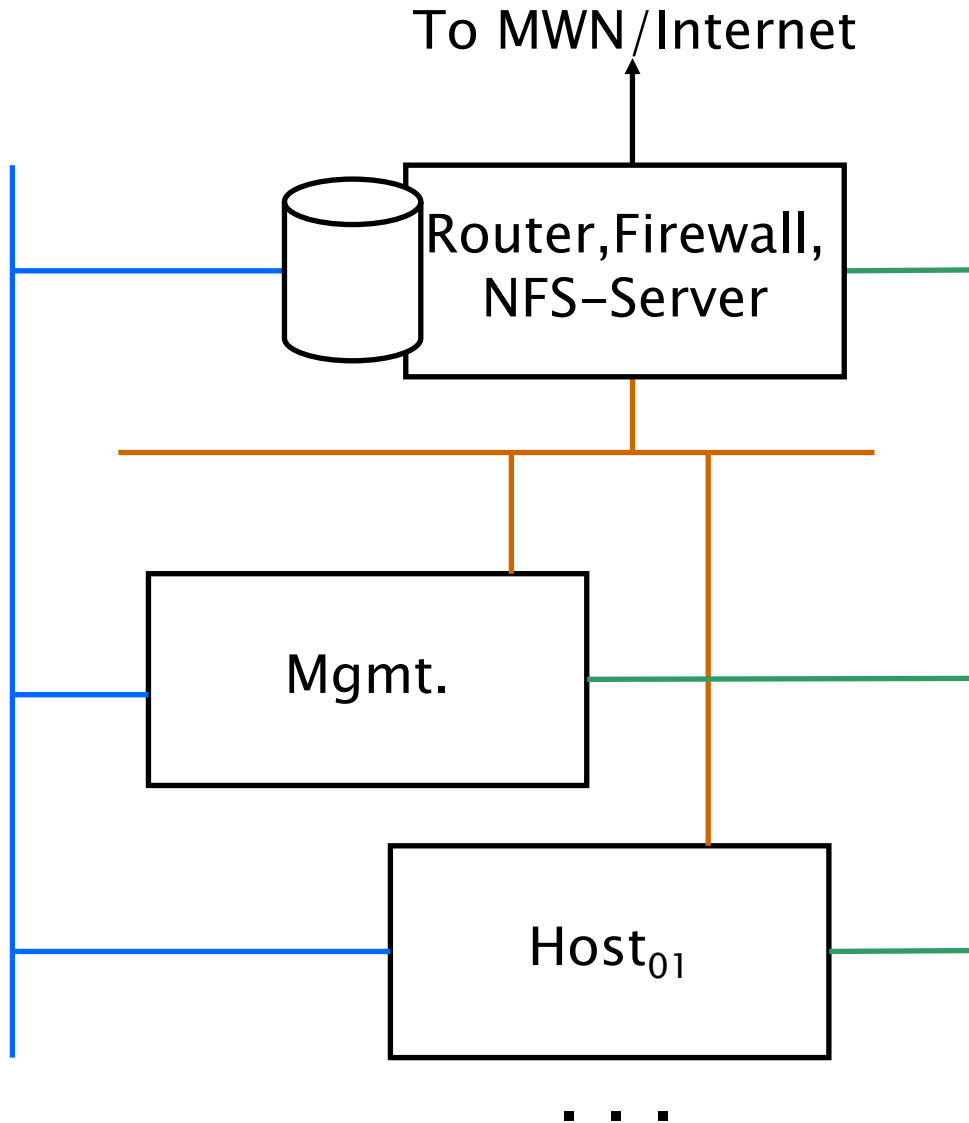


LUDWIG-  
MAXIMILIANS-  
UNIVERSITÄT  
MÜNCHEN

der Bundeswehr  
Universität  München

- The idea & concept/setup
- The implementation: Eucalyptus
  - Deploying VMs
  - Inter-VM communication
  - Elastic Block Storage (EBS)
  - Network security: Concepts & their implementation
- (High Performance?) Computing in the Cloud
  - Effects of concurrency
- Outlook & further work

- Goal: Infrastructure-as-a-Service (IaaS) Cloud
- Idea: Separation of functional areas  
→ Mainly performance & security reasons
- Separation of networks
  - Storage
    - Global file store (NAS/SAN)
    - Elastic block storage (EBS) as a service
  - Management
  - VM-initiated traffic
    - Inter-VM
    - Internet
- Security aspects
  - Separation of networks & traffic
  - “Hiding” hosts
  - Sandboxing of VMs

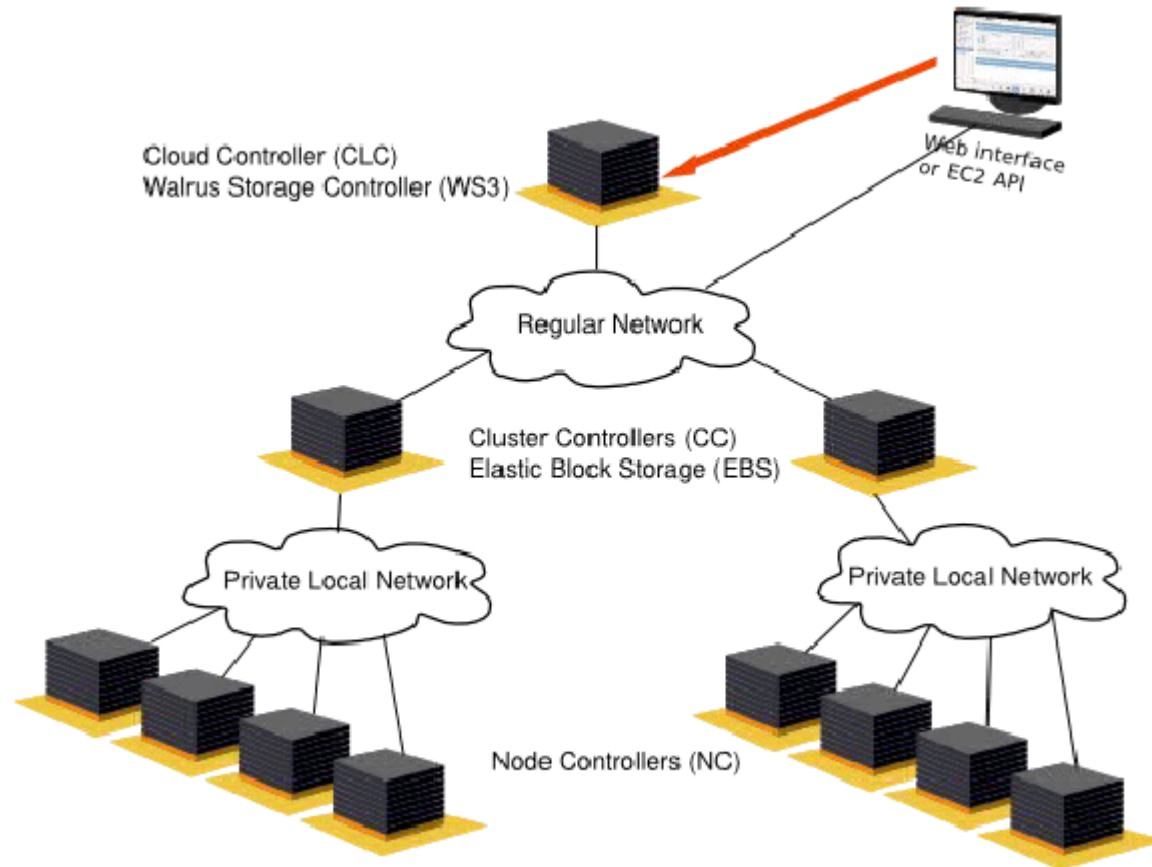


- **Storage/NFS:**
  - Export NFS-shares to mgmt.
- **Mgmt. Network:**
  - VM-image transfer (for initial deployment)
  - Accessing network-based storage from within VMs
  - Further monitoring & mgmt.
- **VM-based communication:**
  - Multinet (!)
  - “Public” network
    - /24 subnet
    - Shared & routed
  - “Private” network
    - /27-subnets
    - One subnet per user here: Layer-3 separation

- The idea & concept/setup
- The implementation: Eucalyptus
  - Deploying VMs
  - Inter-VM communication
  - Elastic Block Storage (EBS)
  - Network security: Concepts & their implementation
- (High Performance?) Computing in the Cloud
  - Effects of concurrency
- Outlook & further work

- Ubuntu Enterprise Cloud (UEC)
- Based on Eucalyptus
- Components
  - Cloud Controller (CLC)
  - Walrus Storage Controller (WS3)
  - Elastic Block Storage Controller (EBS)
  - Cluster Controller (CC)
  - Node Controller (NC)
- All components...
  - ... are implemented as Web Services
  - ... expose Web Service Description Language (WSDL) documents defining their API
- Further information are taken from the...
  - Technical White Paper “Ubuntu Enterprise Cloud Architecture”
  - By Simon Wardley, Etienne Goyer & Nick Barcet

# The UEC architecture



Source: *Technical White Paper "Ubuntu Enterprise Cloud Architecture"*  
by Simon Wardley, Etienne Goyer & Nick Barcet

- Components

- Cloud Controller (CLC)

- Provides interface for users to interact with
- SOAP-based API
- Fully compatible to Amazon Elastic Compute Cloud (EC2)
- CLC talks to the Cluster Controllers (CC)
- Makes top level choices for allocating new VMs
- Holds most information
  - linking users to running instances
  - collection of available machines to be run
  - view of the load of the entire system

- Walrus Storage Controller (WS3)

- Elastic Block Storage Controller (EBS)

- Cluster Controller (CC)

- Node Controller (NC)

- Components
  - Cloud Controller (CLC)
  - Walrus Storage Controller (WS3)
    - Implements Representational State Transfer (REST) and SOAP API
    - Fully compatible with Amazon Simple Storage Protocol (S3)
    - It is used for:
      - Storing machine images
      - Accessing and storing data
    - File level storage system
  - Elastic Block Storage Controller (EBS)
  - Cluster Controller (CC)
  - Node Controller (NC)

- Components
  - Cloud Controller (CLC)
  - Walrus Storage Controller (WS3)
  - Elastic Block Storage Controller (EBS)
    - Runs on the same machine as the Cluster Controller
    - Allows for creating persistent block devices
      - Block devices can be mounted on running machines
    - Ability to create point-in-time snapshots of volumes stored on WS3
      - Starting point for new EBS volumes
      - Protect data for long-term durability
    - At the network level
      - ATA over Ethernet (AoE)
        - > no routing possible!
      - iSCSI (SCSI over TCP or (unlikely) UDP)
        - > routing possible
    - Cluster Controller (CC)
    - Node Controller (NC)

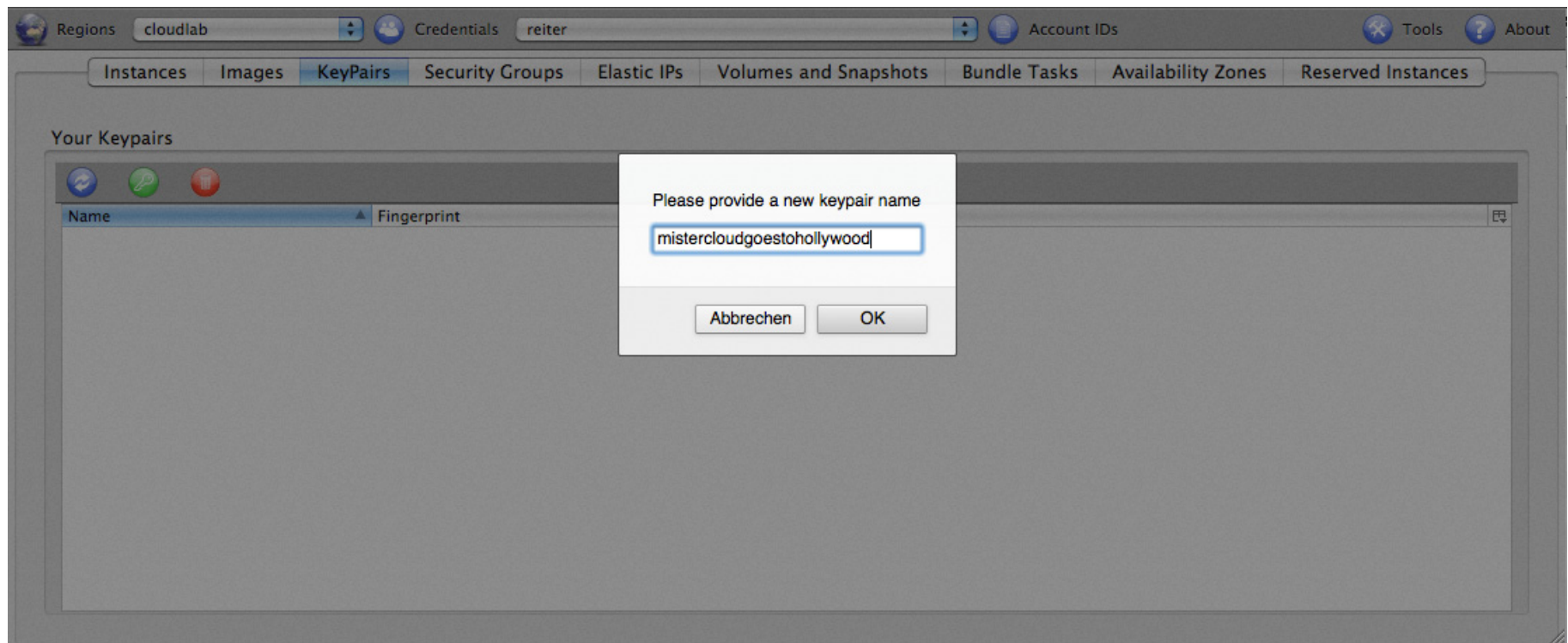
- Components
  - Cloud Controller (CLC)
  - Walrus Storage Controller (WS3)
  - Elastic Block Storage Controller (EBS)
  - Cluster Controller (CC)
    - “Sits” between the NC and the CLC
    - Receives requests from the CLC to allocate VMs
    - Decides which NC will run the VM
      - Decision based upon status reports from NCs
      - Different strategies possible
    - In charge of managing any virtual network
    - Routing traffic to and from VMs
  - Node Controller (NC)

- Components

- Cloud Controller (CLC)
- Walrus Storage Controller (WS3)
- Elastic Block Storage Controller (EBS)
- Cluster Controller (CC)
- Node Controller (NC)
  - Runs on the physical machines on which VMs will be operated
  - Interacts with the OS and hypervisor
  - Instructed by the Cluster Controller
    - Start/stop VMs
    - Reply to availability queries
    - etc.

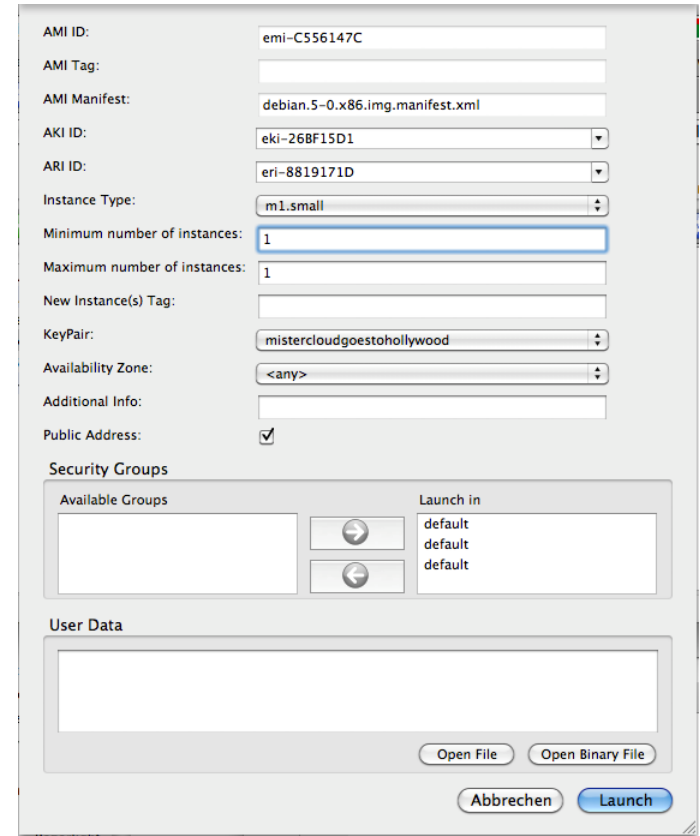
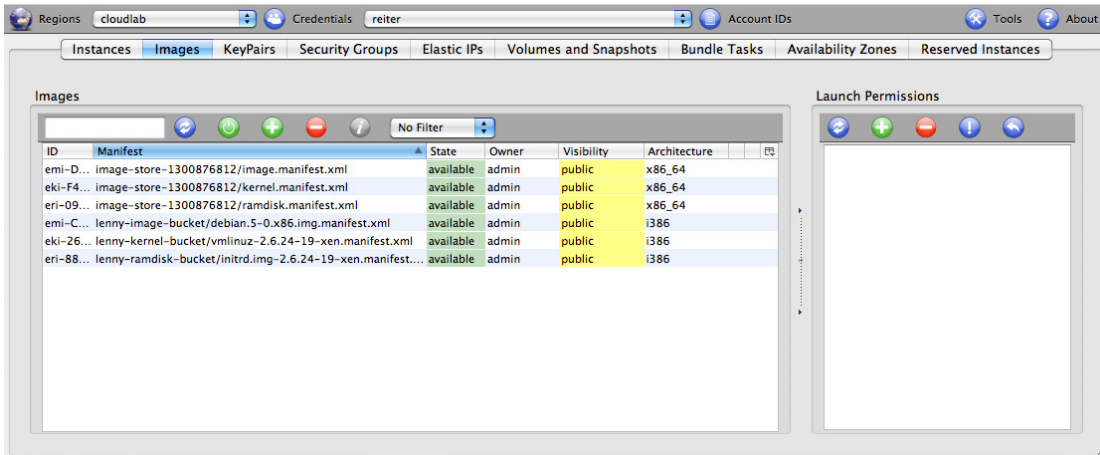
- The idea & concept/setup
- The implementation: Eucalyptus
  - Deploying VMs
  - Inter-VM communication
  - Elastic Block Storage (EBS)
  - Network security: Concepts & their implementation
- (High Performance?) Computing in the Cloud
  - Effects of concurrency
- Outlook & further work

# Deploying a VM (1/2)



- Generate an ssh key-pair (this step is only required once!)
- VMs do not grant access using username/password combinations
- Only strong ssh-key-based authentication!

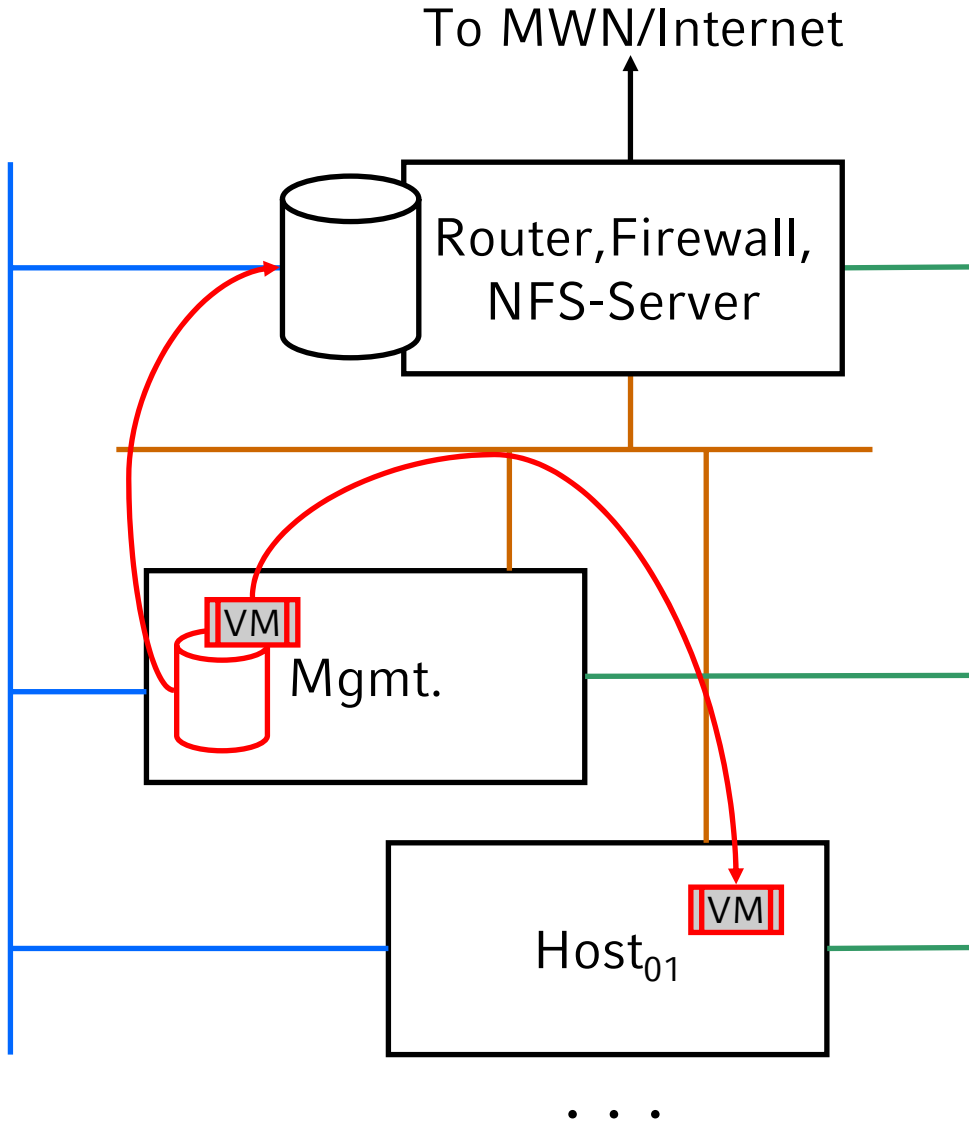
# Deploying a VM (2/2)



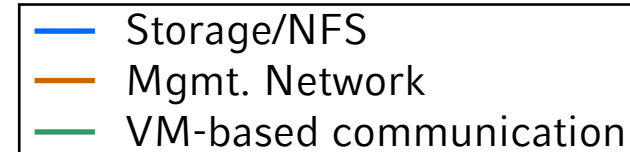
As easy as it is:

- Choose VM-image (Eucalyptus Machine Image, EMI)
- Deploy desired number of machines
- Wait... Done.

# Deploying a VM: ...and technically?

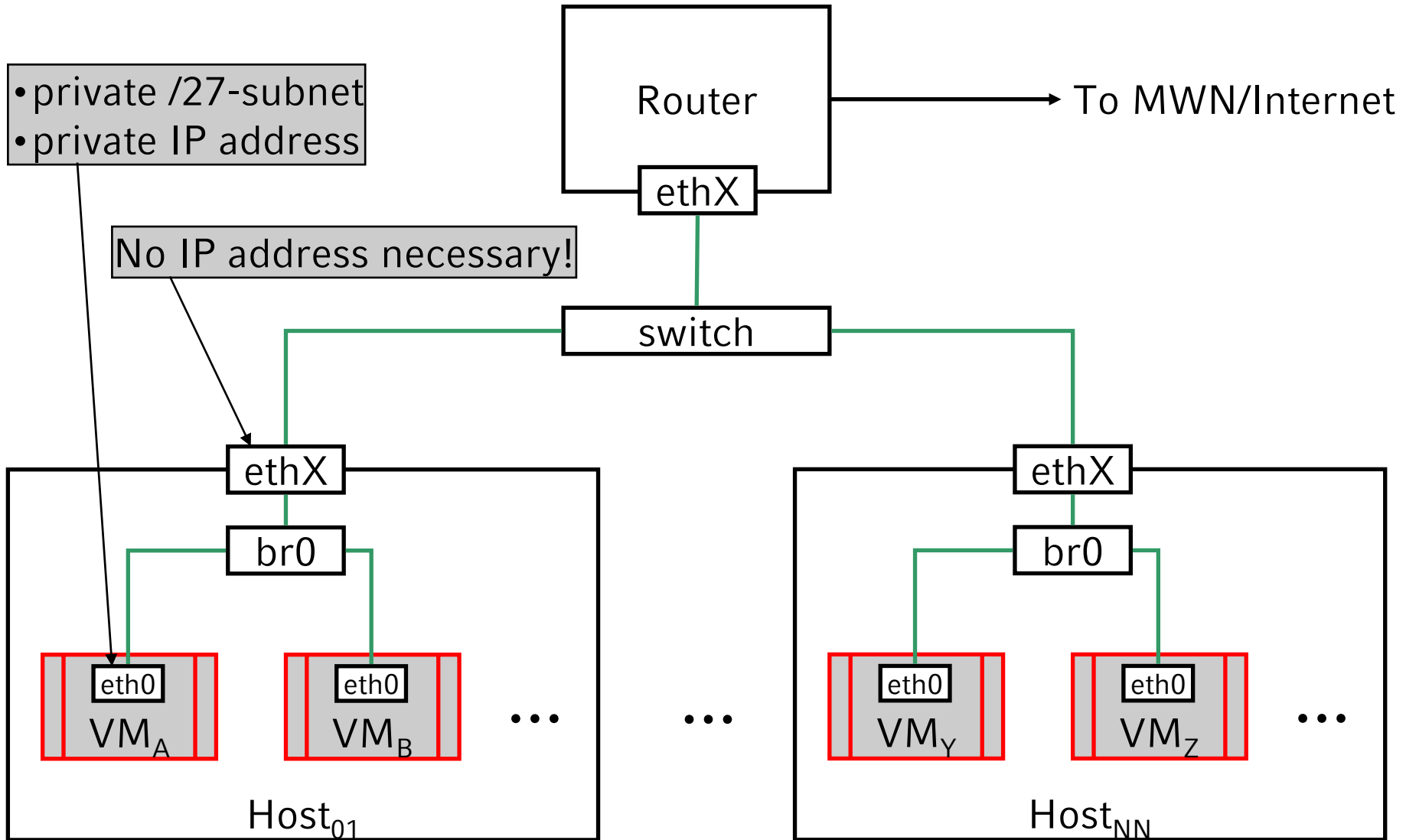


- Mgmt. holds database
  - stored on NFS-share
  - DB holds VM-images (EMIs)
- Choose host
  - Different strategies (Random, Round-Robin etc.)
  - Constraint: Hosts' resources
- Copy VM-image to host
  - Caching occurs
  - Deploy image locally
- Adjust security settings on mgmt.
- Inject ssh-key into VM
- Connect VM to bridge on host
- Launch VM
  - Network config via DHCP (DHCP-server on mgmt. host)
  - User chooses public or private IP in advance



- The idea & concept/setup
- The implementation: Eucalyptus
  - Deploying VMs
  - Inter-VM communication
  - Elastic Block Storage (EBS)
  - Network security: Concepts & their implementation
- (High Performance?) Computing in the Cloud
  - Effects of concurrency
- Outlook & further work

# VM-based network traffic – an overview

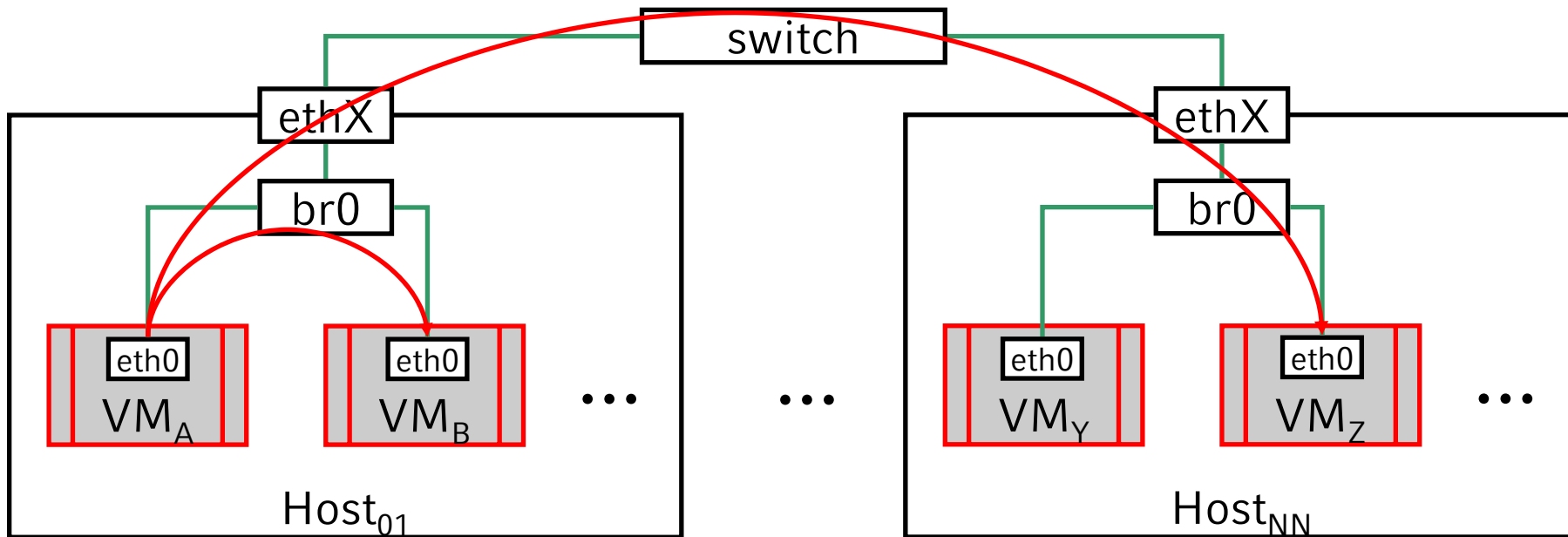


- Performance aspects:

- Dedicated Inter-VM network
- Communication via switch (here: 1Gbit/sec. Ethernet)
- Communication via bridge device (Kernel-based (mem copy) possible, depends on Hypervisor)
- Drawbacks:
  - Shared network for all customers
  - Traffic demands CPU resources

- Security aspects:

- Hiding hosts from VMs (no layer-3 config for hosts)
- BUT:
  - VM-isolation up to Hypervisor
  - Shared network for all customers
  - Network isolation
    - Here: layer-3 basis only
    - Others possible (!)

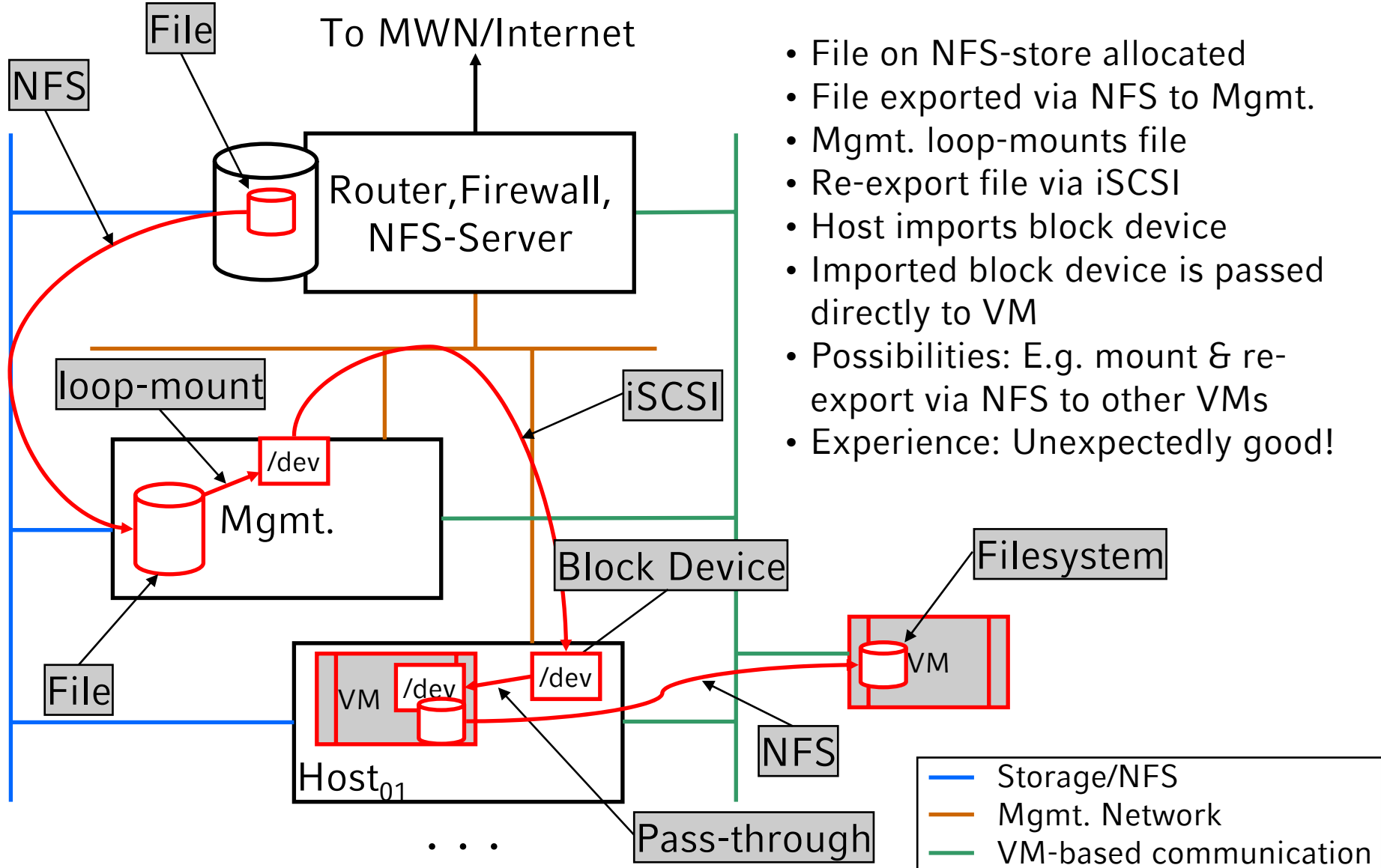


- The idea & concept/setup
- The implementation: Eucalyptus
  - Deploying VMs
  - Inter-VM communication
  - Elastic Block Storage (EBS)
  - Network security: Concepts & their implementation
- (High Performance?) Computing in the Cloud
  - Effects of concurrency
- Outlook & further work

## Repetition:

- Walrus Storage Controller
  - Implements Representational State Transfer (REST) and SOAP API
  - Fully compatible with Amazon Simple Storage Protocol (S3)
  - It is used for:
    - Storing machine images
    - Accessing and storing data
  - File level storage system
- Elastic Block Storage Controller
  - Runs on the same machine as the Cluster Controller
  - Allows for creating persistent block devices
    - Block devices can be mounted on running machines
  - Ability to create point-in-time snapshots of volumes stored on WS3
    - Starting point for new EBS volumes
    - Protect data for long-term durability
  - At the network level
    - ATA over Ethernet (AoE)
      - > no routing possible!
    - iSCSI ("SCSI oder TCP or unlikely UDP")
      - > routing possible

# Implementation: Elastic Block Storage



- File on NFS-store allocated
- File exported via NFS to Mgmt.
- Mgmt. loop-mounts file
- Re-export file via iSCSI
- Host imports block device
- Imported block device is passed directly to VM
- Possibilities: E.g. mount & re-export via NFS to other VMs
- Experience: Unexpectedly good!

- The idea & concept/setup
- The implementation: Eucalyptus
  - Deploying VMs
  - Inter-VM communication
  - Elastic Block Storage (EBS)
  - Network security: Concepts & their implementation
- (High Performance?) Computing in the Cloud
  - Effects of concurrency
- Outlook & further work

- 4 networking modes supported by Eucalyptus:
  - System Mode
  - Static Mode
  - Managed Mode
  - Managed-NoVLAN Mode
- Different levels of security granted
- Choice influenced by aspects of corporate network setup/configuration

- Considered the simplest networking mode
- Assigns random MAC address to the VM before booting
- Attaches the VM's Ethernet device to the physical Ethernet through node's local bridge
- VM instances typically obtain IP address using DHCP (same way any non-VM machine using DHCP would obtain address)
- Obvious disadvantages:
  - Typically, VMs not separated from nodes

- Mapping of MAC address      IP address needed
- Static entry in DHCP server is set up
- At VM's startup, next free MAC/IP pair is assigned to VM
- VM's Ethernet device connected to physical Ethernet through node's bridge (similar to SYSTEM mode).
- Obvious advantages:
  - More control over VM IP address assignment
- Disadvantages:

System and Static mode do not offer...

  - Security groups (see later)
  - Elastic IPs  
(user-defined IP address assignments / IP reservations)
  - Isolation to network traffic between VMs
  - Availability of the metadata service  
(to obtain instance specific information)

- Admin defines large network
  - usually private and un-routable
- Eucalyptus maintains DHCP server
  - static mappings for each VM
- One VLAN per customer implemented!
  - VM's Ethernet device connected to physical Ethernet through node's bridge
  - VLAN-tagging enabled
  - Prerequisite: reserved VLAN-range for Eucalyptus
- Static/public IPs possible  
("Elastic IPs")
- Users can define "named networks"  
(aka Amazon's "security groups")
  - Subnet out of previously defined IP range
  - Application of ingress rules for whole network possible
- Most feature-full mode

# Pitfall: Initially no rule defined!

Add New Permission for Security Group: default

External Group

Select a protocol and specify a host/range

Protocol Details

Other

Protocol ICMP

ICMP Type -1 ICMP Code -1

Host/Network Details

Host

Network 0.0.0.0/0

Get My Host Address

Get My Network Range

Abbrechen Add

Add New Permission for Security Group: default

External Group

Select a protocol and specify a host/range

Protocol Details

SSH

Protocol TCP/IP

Port 22

Host/Network Details

Host

Network 0.0.0.0/0

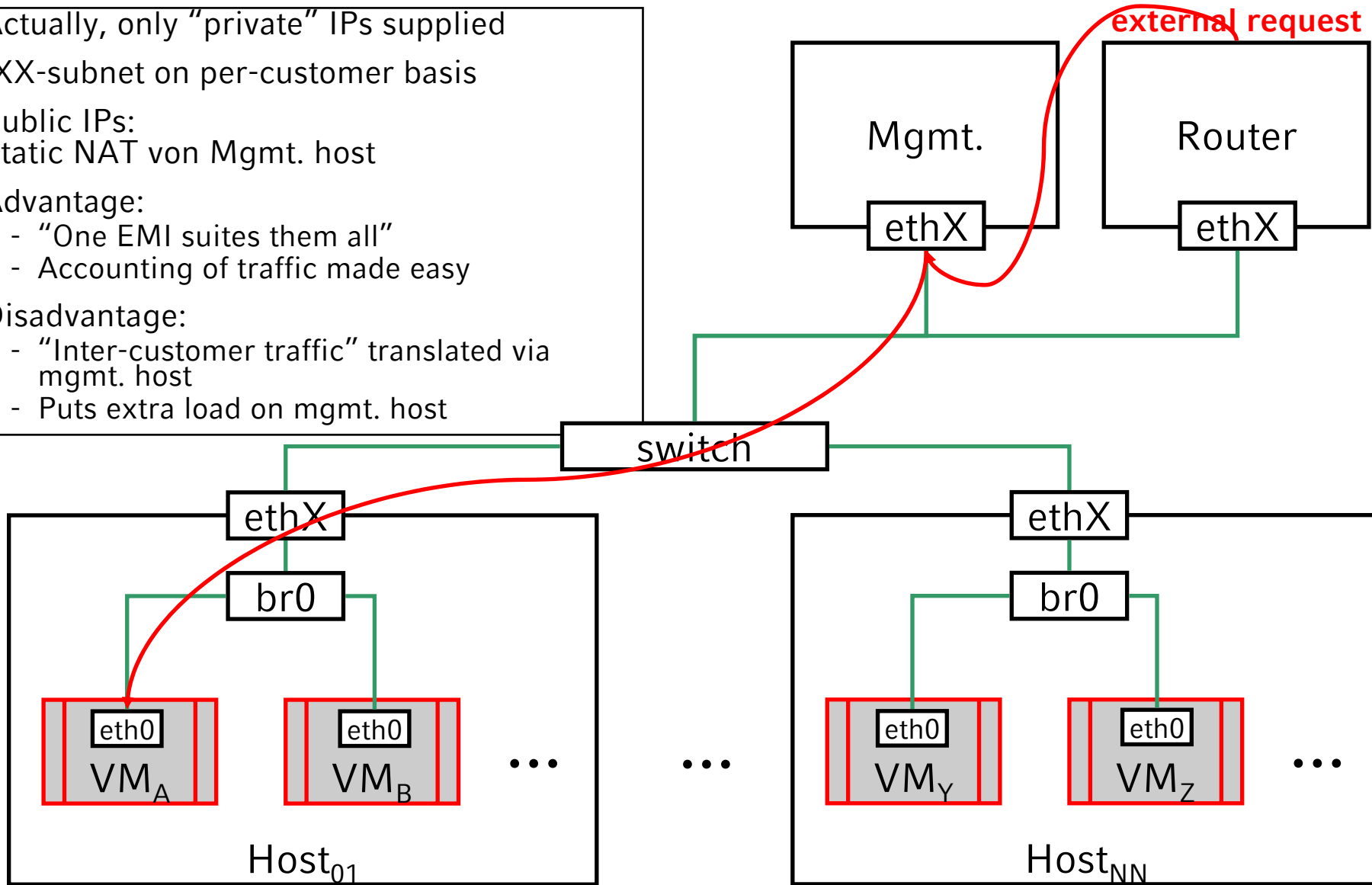
Get My Host Address

Get My Network Range

Abbrechen Add

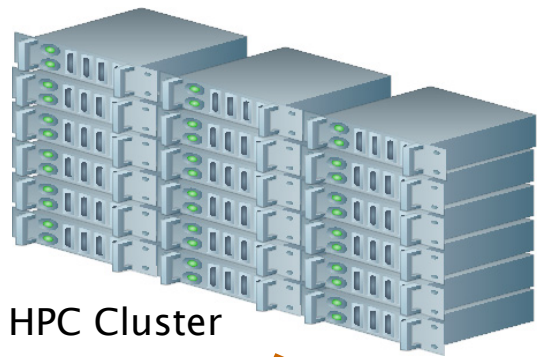
# Public vs. private IP addresses

- Actually, only "private" IPs supplied
- /XX-subnet on per-customer basis
- Public IPs:  
Static NAT von Mgmt. host
- Advantage:
  - "One EMI suites them all"
  - Accounting of traffic made easy
- Disadvantage:
  - "Inter-customer traffic" translated via mgmt. host
  - Puts extra load on mgmt. host



- Identical to MANAGED mode, but...
  - does not provide VM network isolation
- Layer-3 isolation granted
- Advantage:
  - Alternative for not “VLAN-clean” setups

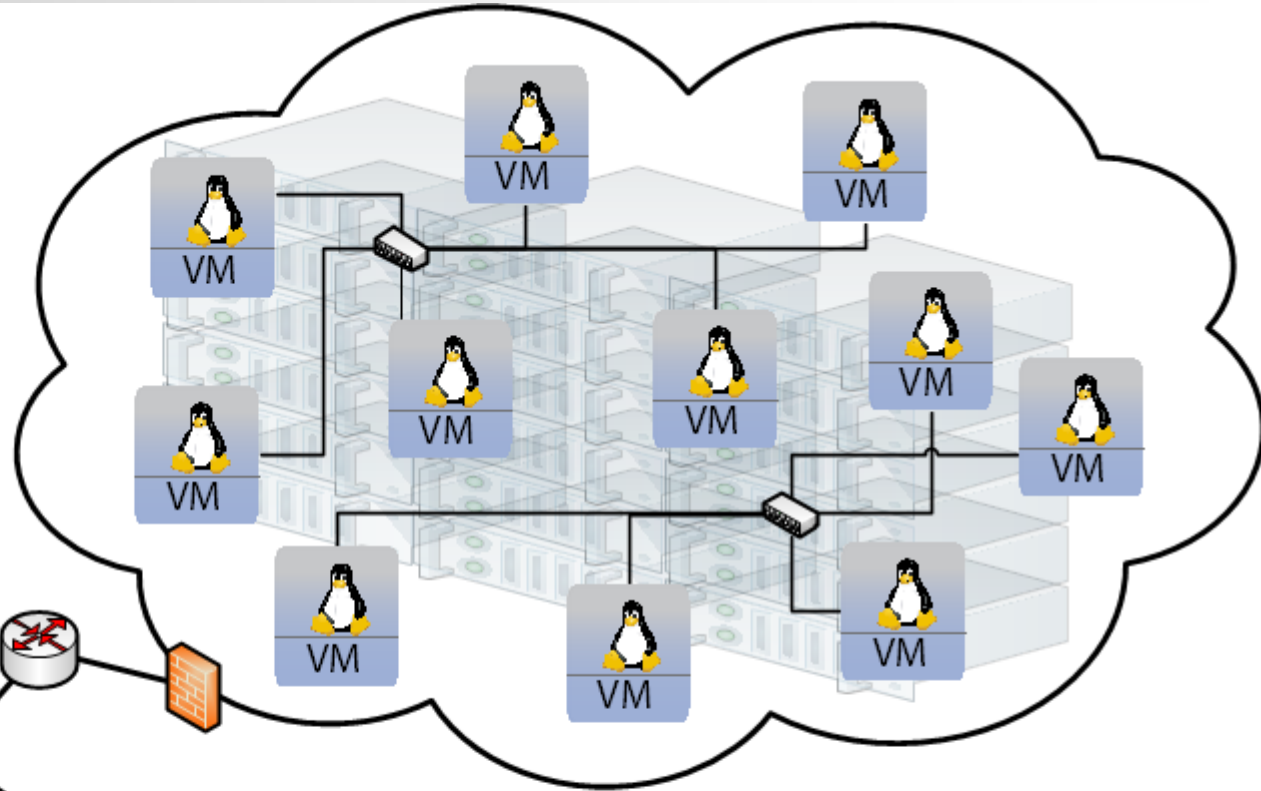
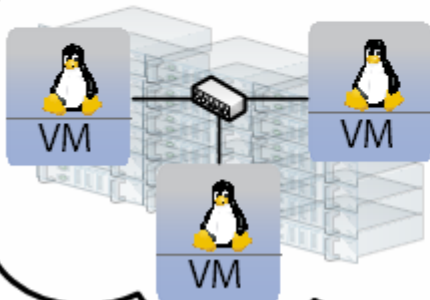
- The idea & concept/setup
- The implementation: Eucalyptus
  - Deploying VMs
  - Inter-VM communication
  - Elastic Block Storage (EBS)
  - Network security: Concepts & their implementation
- (High Performance?) Computing in the Cloud
  - Effects of concurrency
- Outlook & further work



HPC Cluster



HPC Cloud Cluster



## Management benefits of virtual HPC

- Dynamical sizing / partitioning
- Loadbalancing
- Automation / scripting of management tasks
- Fault tolerance without „Checkpoint and restart“
- Better security due to sandboxing of processes
- ...

## But: What about performance?







- Effects expected:
  - Virtualization layer will induce overhead
  - Concurrent effects appear when running several VMs in parallel; e.g. effects on storage, network, memory access
  - Effects will depend on virtualization architecture, implementations, OS, ...
- What scale will that effects be?
- How to measure?

- The idea & concept/setup
- The implementation: Eucalyptus
  - Deploying VMs
  - Inter-VM communication
  - Elastic Block Storage (EBS)
  - Network security: Concepts & their implementation
- (High Performance?) Computing in the Cloud
  - Effects of concurrency
- Outlook & further work

# Benchmark Suite – Design Goals

- Measure core components of virtualized systems

- CPU 
- Main Memory 
- Network 
- Disk 

- Compare results of different hypervisors

- Architectures (full-, native-, ... , para-virtualization)
- Implementations (VMware ESXi, Xen, MS Hyper-V, ...)



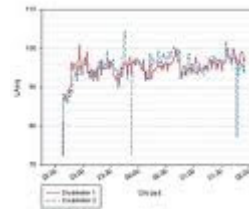
- Analyze impact of virtualization on different...

- Operating Systems (Windows, Linux, ...)
- System architectures (32bit, 64bit, ...)



- Measure

- Overhead of virtualization
- Effects of concurrency

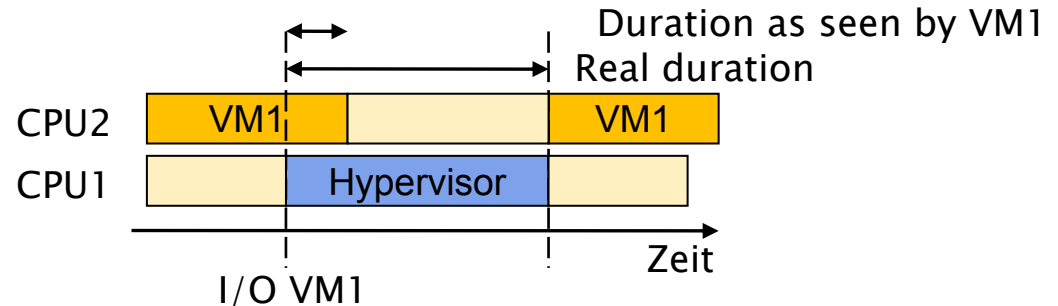


- Take care about special time measurement in VMs

Measurement of time difficult in virtualized environments:

-Definition of time

-Time synchronization



**Example:** Linpack-Benchmark

```
static REAL second(void){
    return ((REAL) ((REAL) clock() / (REAL) CLOCKS_PER_SEC));
}
# clock()          : Wall-Clock-Time in time ticks since process start
# CLOCKS_PER_SEC: Frequency of the hardware clock
```

➔ External clocks may be necessary, depending on counters used by the benchmark tool

- Benchmarks used:

Component	Benchmark
CPU	Linpack
Main Memory	Ramspeed
Disk	Iometer
Network	Iometer

- Run combinations of single benchmarks to test for concurrent effects, e.g.
  - CPU + Network
  - Parallel network access
  - Different network setups (VM2VM, VM2PM, ...)
  - ...

- Benchmarks used:

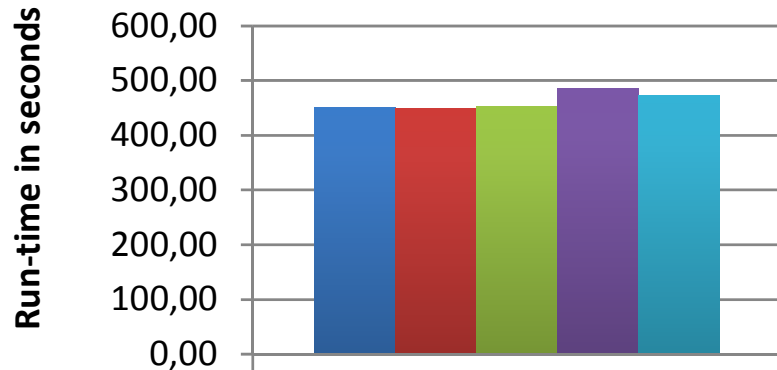
Component	Benchmark
CPU	Linpack
Main Memory	Ramspeed
Disk	Iometer
Network	Iometer

- Run combinations of single benchmarks to test for concurrent effects, e.g.

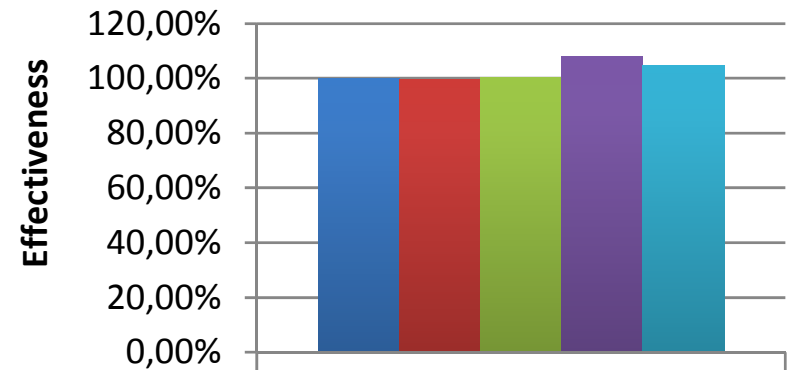
- CPU + Network
- Parallel network access
- Different network setups (VM2VM, VM2PM, ...)
- ...

# Applying the Test Suite

Linpack - Absolute/Relative runtime



Physical Linux	450,58
Xen Para	448,79
OpenVZ	452,53
MS Hyper-V	485,76
VMware ESXi	472,00



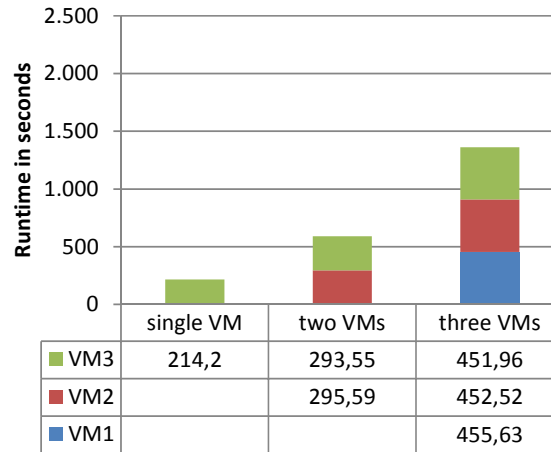
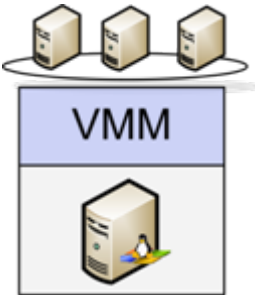
Physical Linux	100,00%
Xen Para	99,60%
OpenVZ	100,43%
MS Hyper-V	107,81%
VMware ESXi	104,75%

Absolute runtime

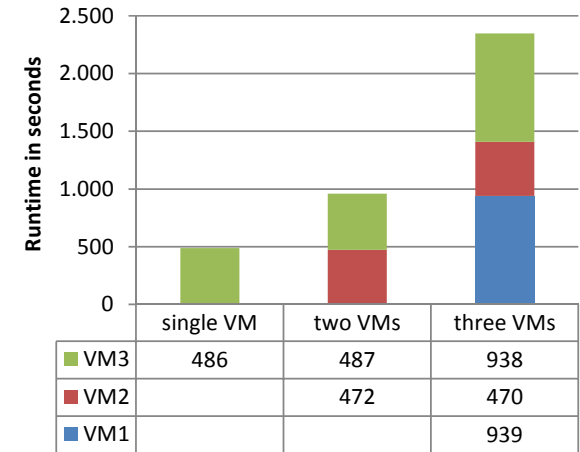
Relative runtime

# Applying the Test Suite

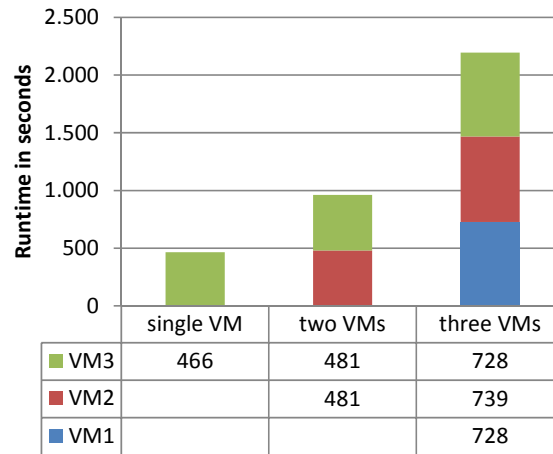
## CPU – effects of concurrency



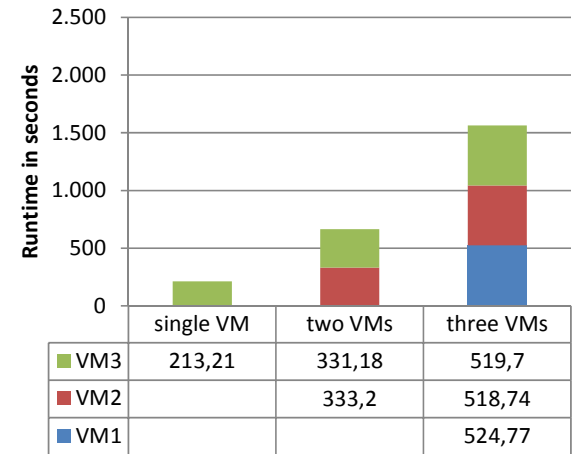
Xen



Virtuozzo



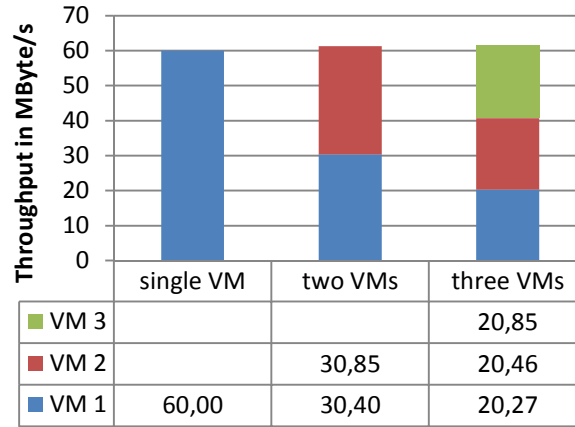
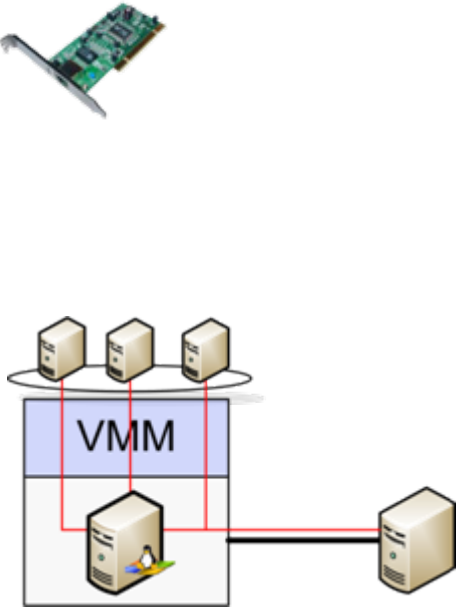
Hyper-V



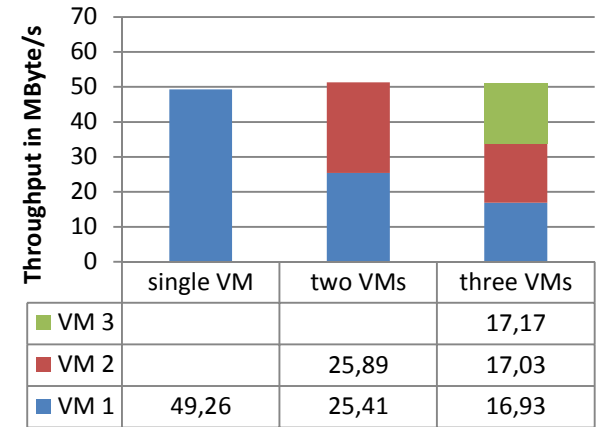
ESXi

# Applying the Test Suite

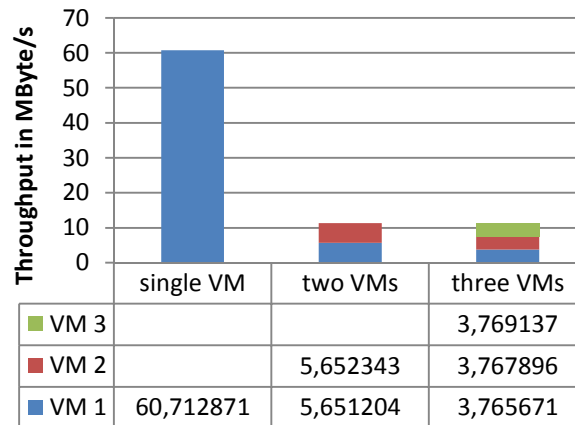
Network – effects of concurrency



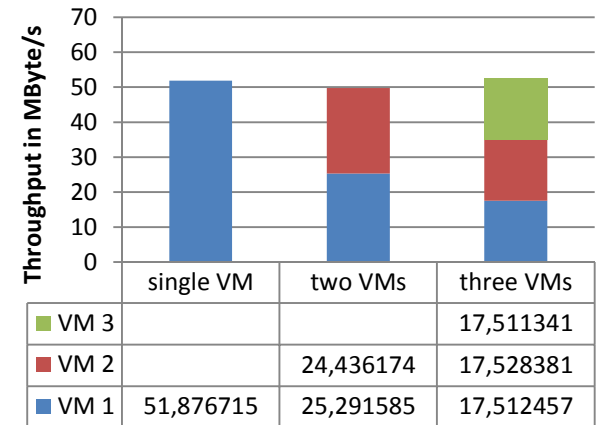
Xen – receive data from network



Xen – send data to network



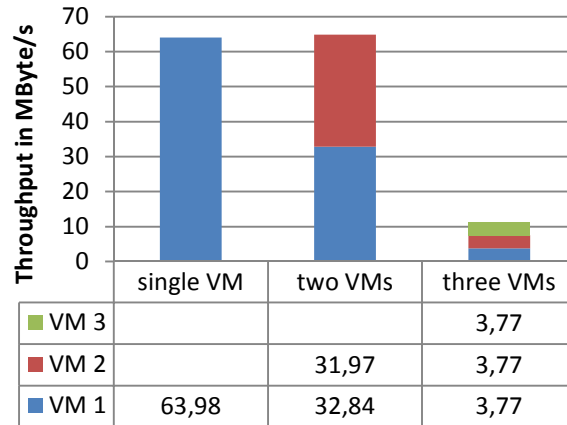
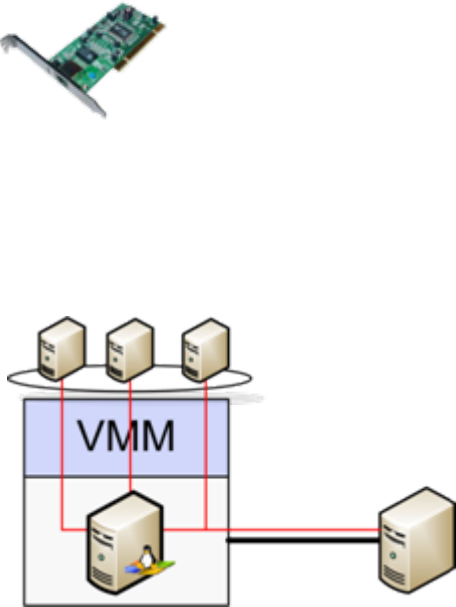
Virtuozzo – receive data from network



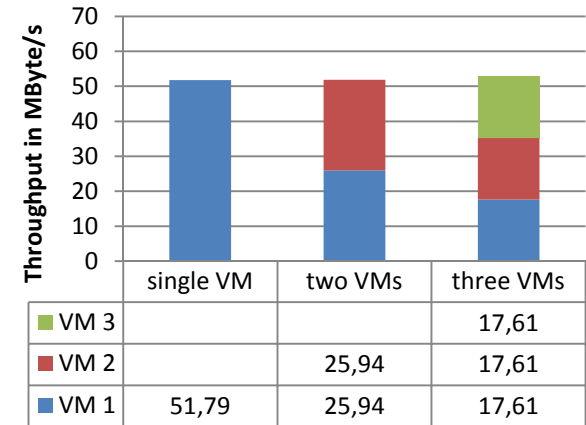
Virtuozzo – send data to network

# Applying the Test Suite

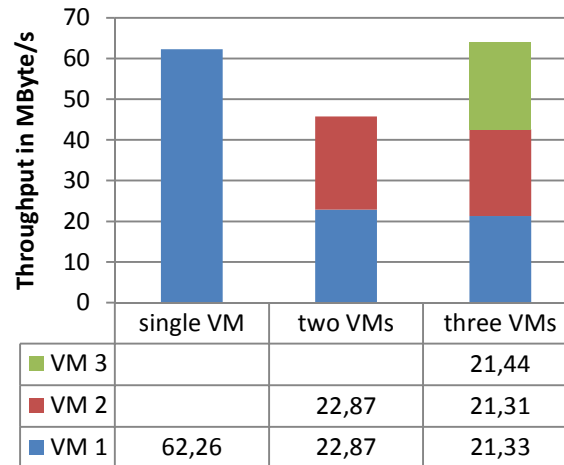
Network – effects of concurrency



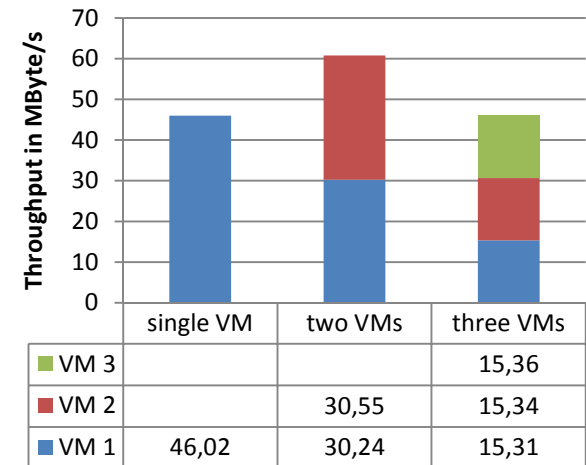
Hyper-V – receive data from network



Hyper-V- send data to network



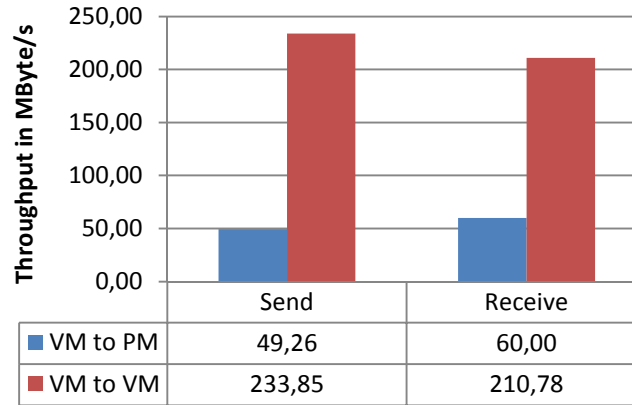
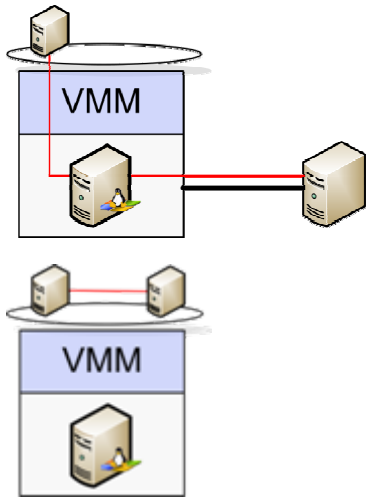
ESXi- receive data from network



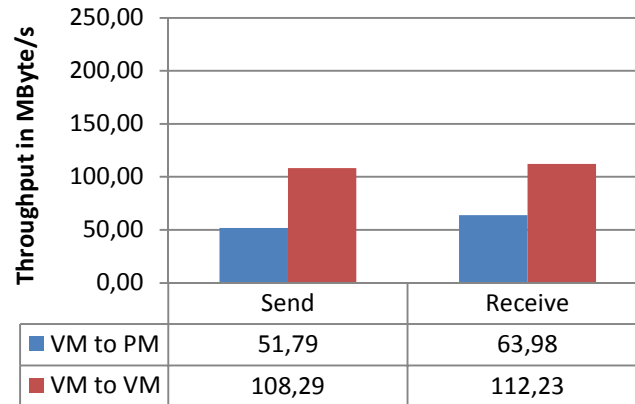
ESXi- send data to network

# Applying the Test Suite

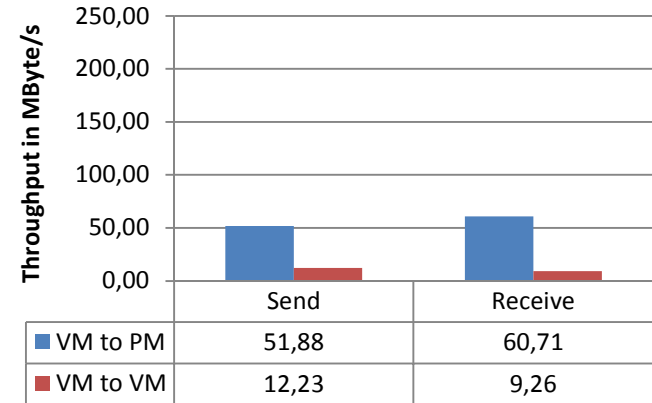
Network – Physical vs. virtual communication peers



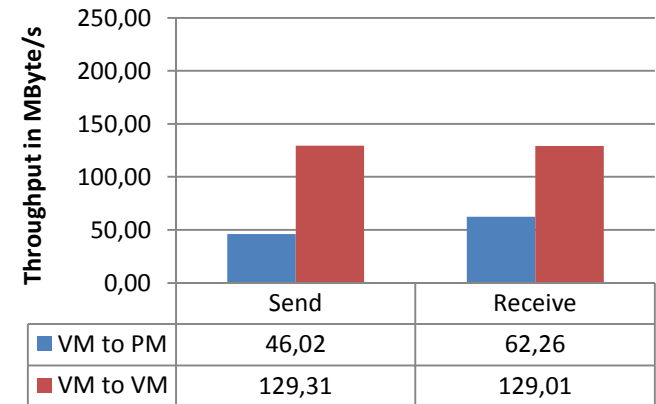
Xen



Hyper-V



Virtuozzo



ESXi

Criterion	High-Performance	High-Throughput
Coupling	tight	loose
CPU impact	co-determinant	critical
Interconnect impact	critical	less
Main memory addressing	sensitive; prefers contiguous, uniform latency addressing	less sensitive
Program structures	inter-communicating program replicas	workflows; pipelining of computing tasks
Examples	fluid dynamics problems, crash codes	high-energy physics (e.g. CERN LHC experiments), general parameter variation studies

- CPU virtualization is efficient (close to 100%)
- RAM access is fast
  - ➔ High Throughput Computing (HTC) Tasks can be virtualized efficiently
- Network Access efficiency depends on
  - flow direction
  - concurrent use
  - Available CPU power
- Topology changes (e.g. live migration) “confuses” MPI
  - ➔ High Performance Computing (HPC) Tasks should not be virtualized yet

- The idea & concept/setup
- The implementation: Eucalyptus
  - Deploying VMs
  - Inter-VM communication
  - Elastic Block Storage (EBS)
  - Network security: Concepts & their implementation
- (High Performance?) Computing in the Cloud
  - Effects of concurrency
- Outlook & further work

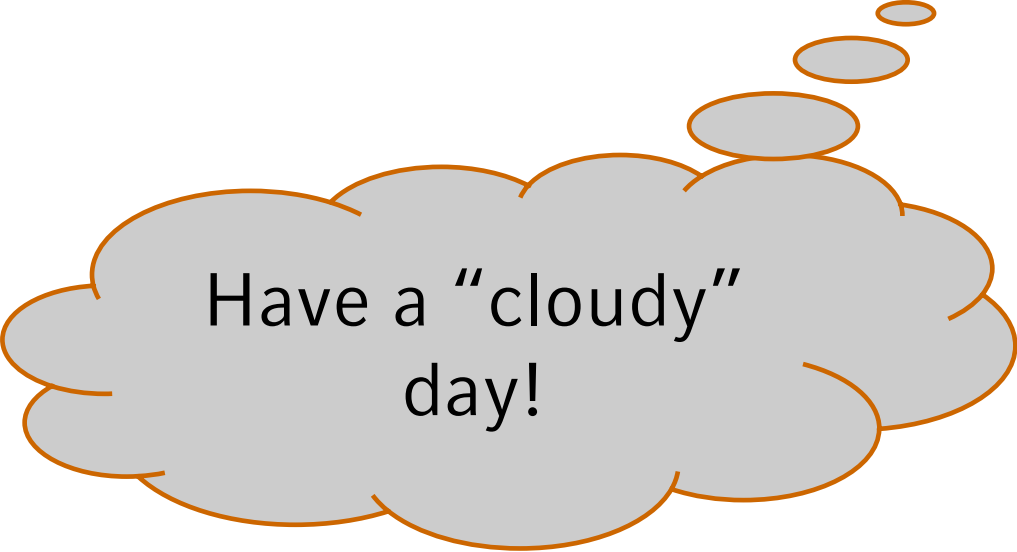
- Eucalyptus 3.0 Roadmap
  - High Availability (HA)
  - Eucalyptus Identity Authorization and Management (EIAM)
  - Active Directory/LDAP integration
  - Windows Hosting Service
  - Boot from EBS (Elastic Block Storage)
- Benchmarking parallel HPC applications in the Cloud
  - Different Hypervisors
    - Uniform Hypervisor per run
    - VMs powered by different Hypervisor during one run
  - Different locations of VMs
    - On the same host
    - On different hosts, but in the same LAN
    - Including host in foreign locations using VPN-tunnels
  - ...
- And much more... ;-)  
Want to collaborate?

**Contact:**

[felde@nm.ifi.lmu.de](mailto:felde@nm.ifi.lmu.de)

**Thank you.**

---



Have a "cloudy"  
day!

- Two type of “actors” need to be authenticated and authorized on UEC:
  - The users or administrators of the system which have specific rights to modify the system and start and stop instances;
  - The components of the system (NC, CLC, CC) which need to trust each other when transmitting requests.
- On a general level, the authentication is performed using locally-generated X509 certificates, as cryptographic keys to authenticate and secure communications between all actors with communication based on the WS-Security policy framework.
- This is true for all internal communication within the cloud. Users authenticate either with an X509 certificate or a Query type key.
- The initial addition of Cluster Controller or Node Controller to the Cloud Controller environment requires using a password to exchange the cryptographic keys from the lower controller to the upper one.
- Once this is done, all operations rely on the trust provided by these keys in the communications between the controllers.
- Authentication and authorization of users:
- Initial user account creation of user is performed in a two-step operation:
  - First, any user with access to the Cloud Controller user interface can fill in a form to request an account;
  - When a request is received, it is up to the administrator to grant access to the user according to its own policy.
- Users which have been granted access retrieve the certificate and query type key that have been generated for them and which are used for all their requests performed through the APIs.
- The type of authentication (query key or certificate) will vary based on the tool used to access the cloud, their password only being used to access the web console that allows them to retrieve their certificate and query key.
- Note: a new certificate and query key is generated each time the user ask to retrieve them.
- The Cloud Controller holds the central right repository for each user.
- Every time a request arrives at a controller (CLC, CC or NC), it is its duty to verify that the user from which the request originated is duly authorized to perform it.
- Users obviously only make requests to the Cloud Controller, so authentication is only performed there, but authorization is verified at each level to prevent abuse.