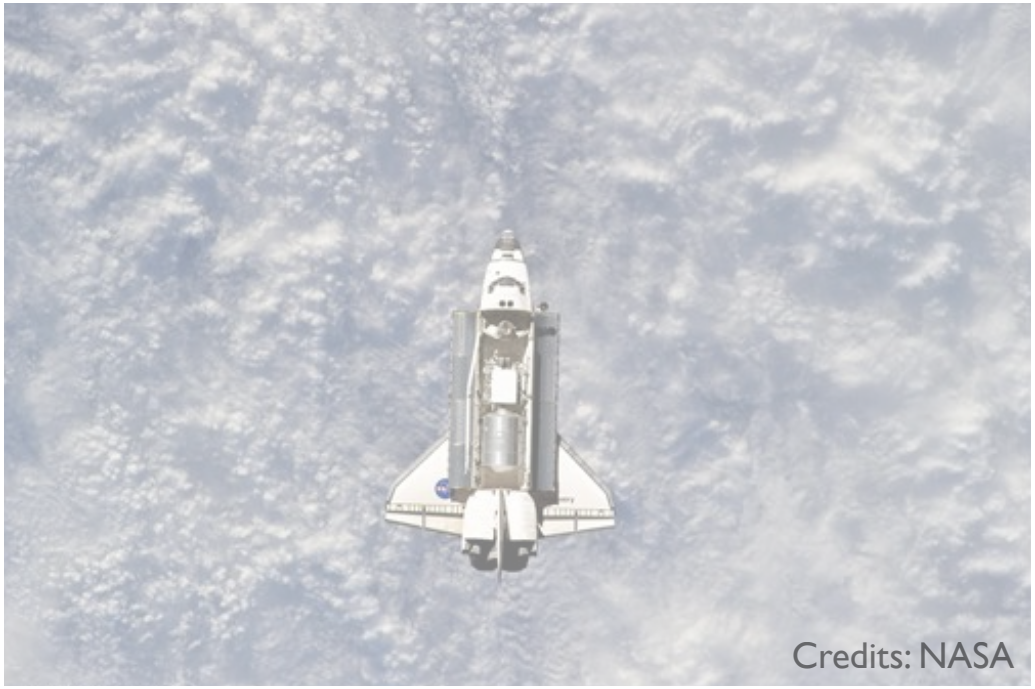
# Beyond the Clouds, the DISCOVERY Initiative




Localization is a key element to deliver
***efficient*** as well as sustainable Utility Computing Solutions

Adrien Lèbre / Ascola Project Team
June, 2013

# Context

# xxx Computing

- Meta / Cluster / Grid / Desktop / "Hive" / Cloud / Sky ...
- A common objective: provide computing resources (both hardware and software) in a flexible, transparent, secure, reliable, ... way

⇒ xxx as Utility Computing

- Challenges

  Software/Hardware heterogeneity

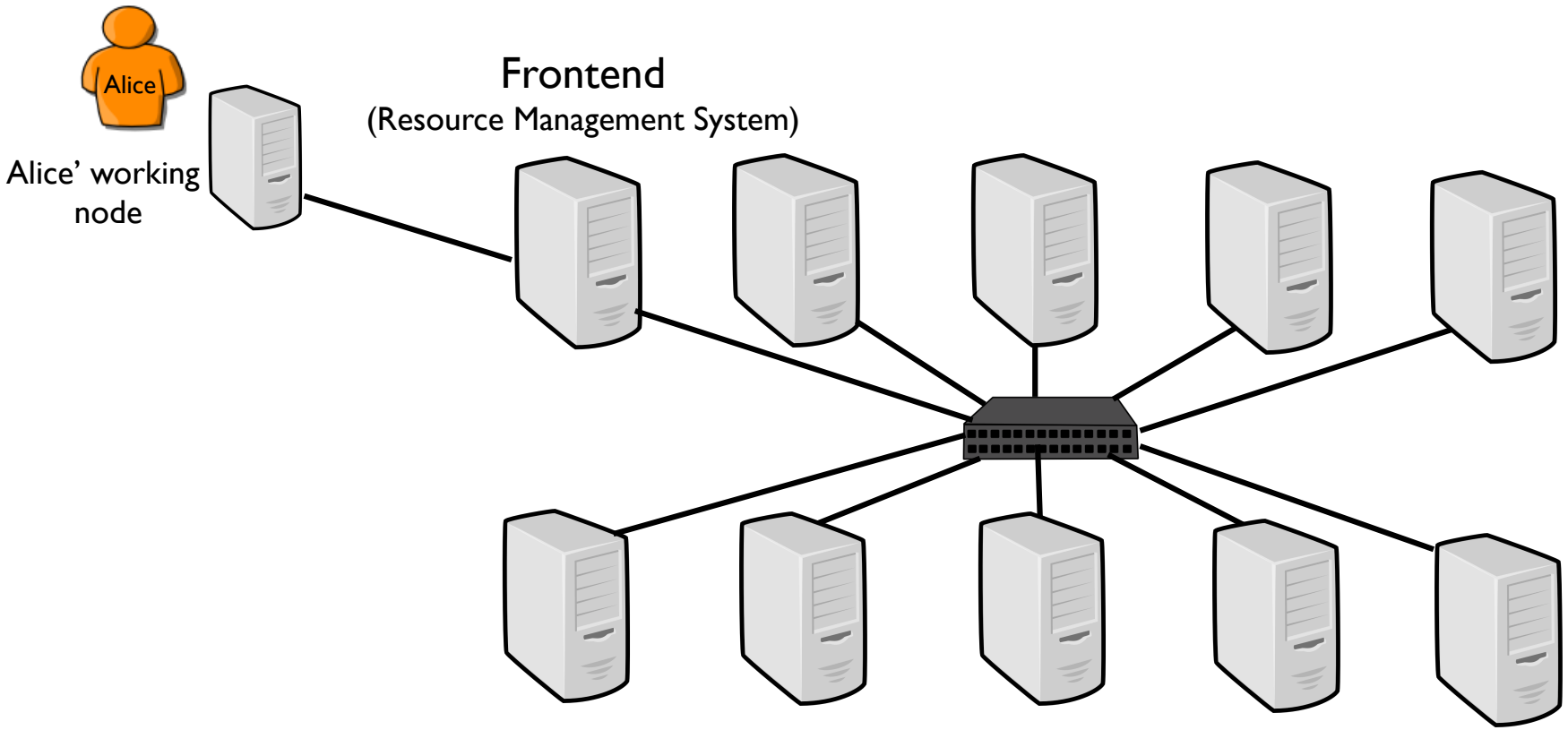
  Security (Isolation between applications, ...)

  Reliability / Resiliency

  Data Sharing ...

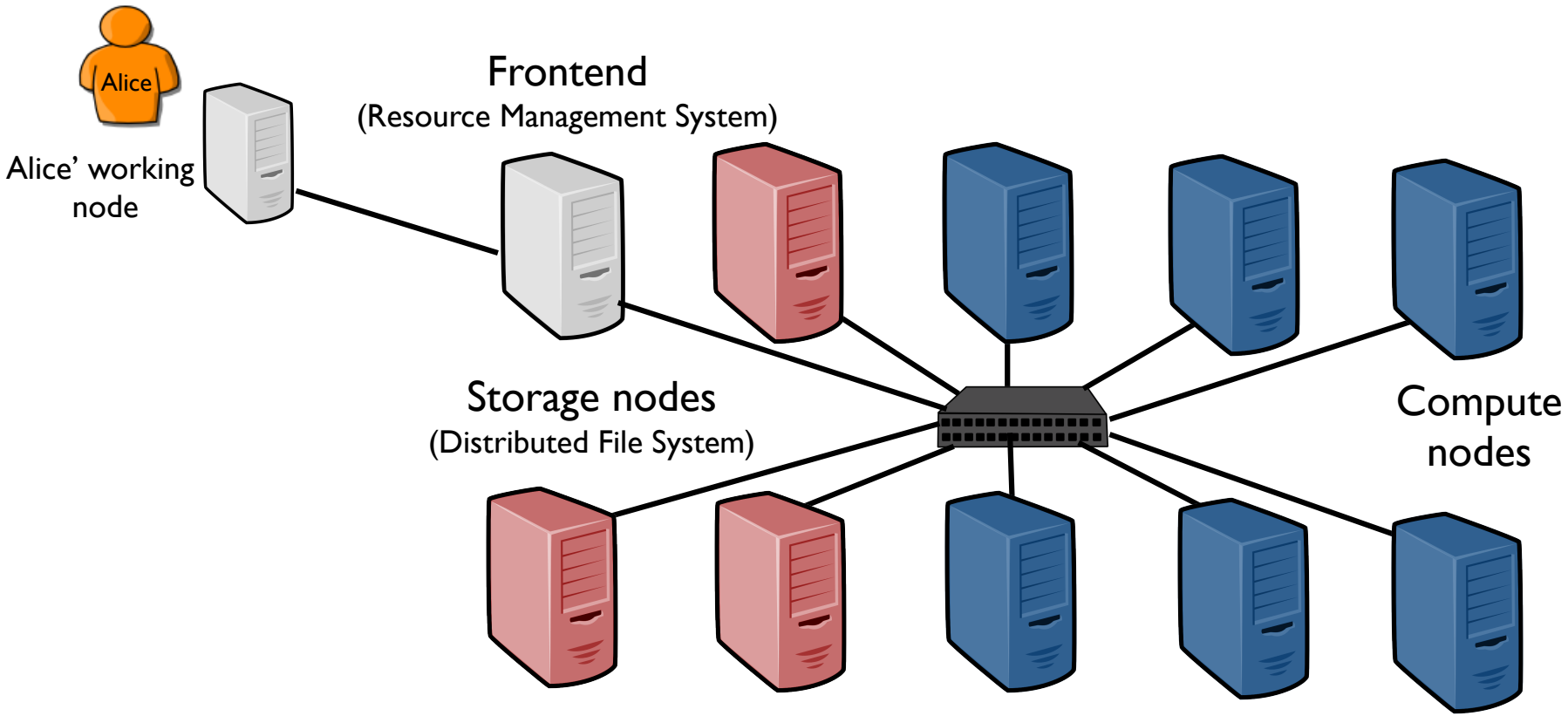
# Utility Computing and Data Management

- Network of Workstations 1990 / 20**xx**



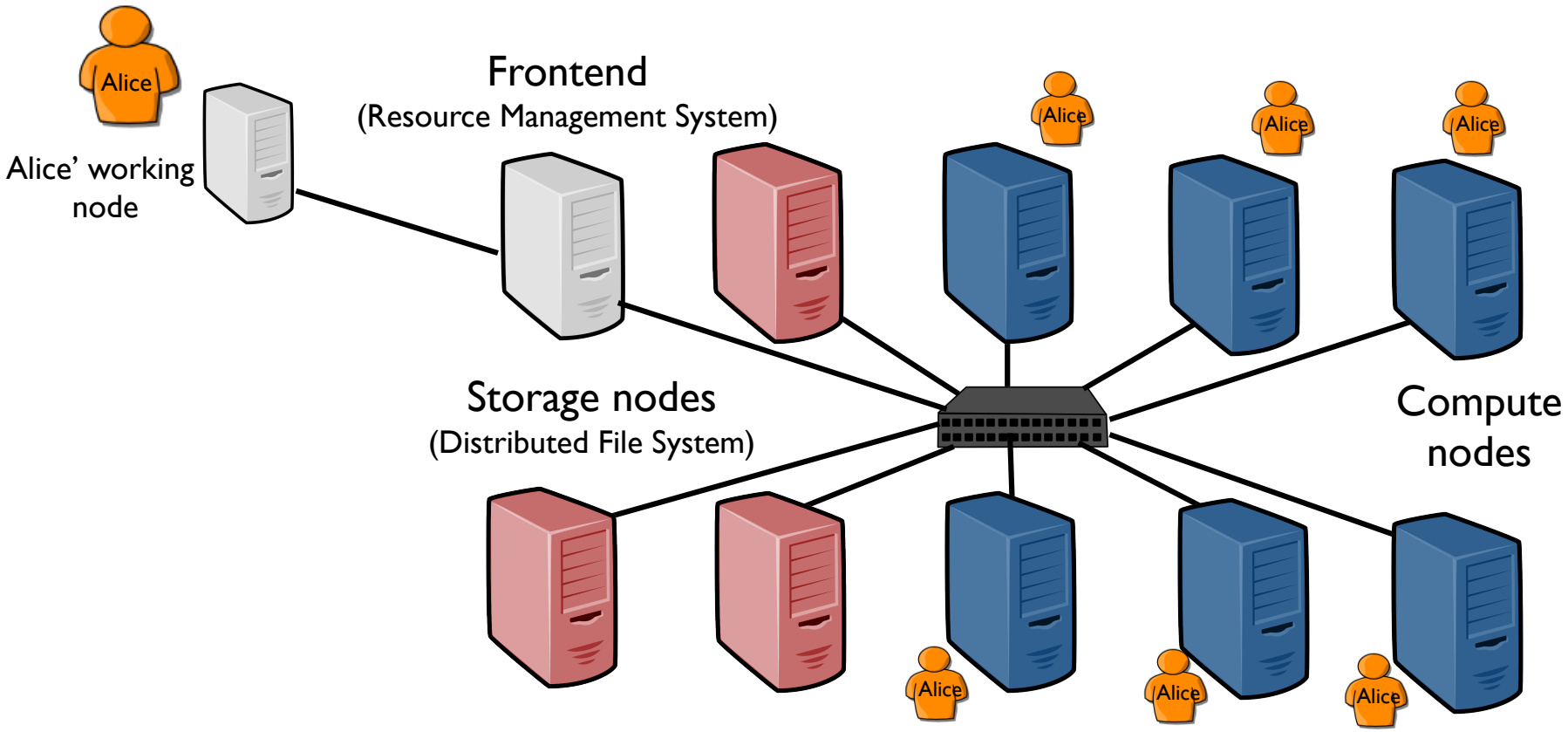
# Utility Computing and Data Management

- Network of Workstations 1990 / 20**xx**



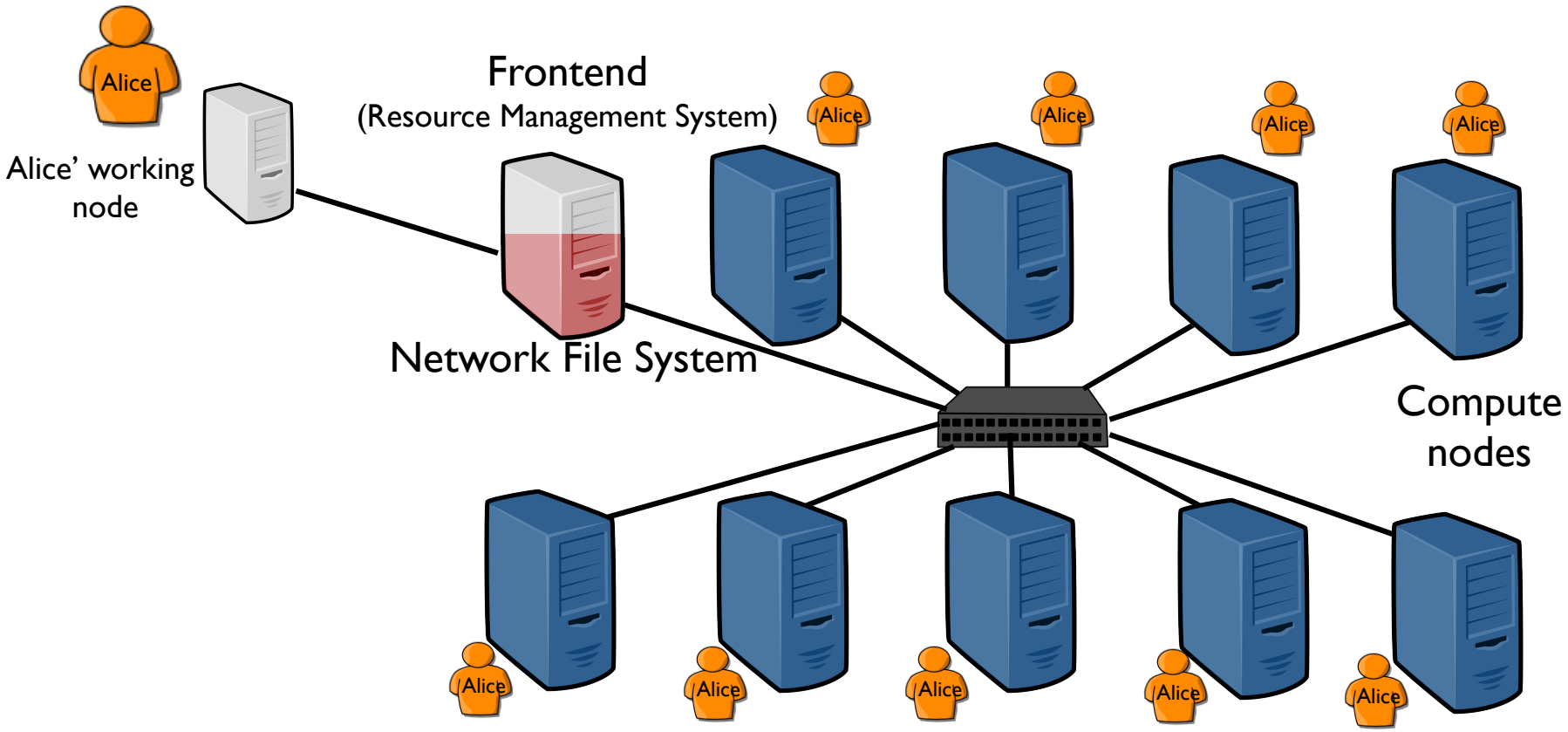
# Utility Computing and Data Management

- Network of Workstations 1990 / 20**xx**



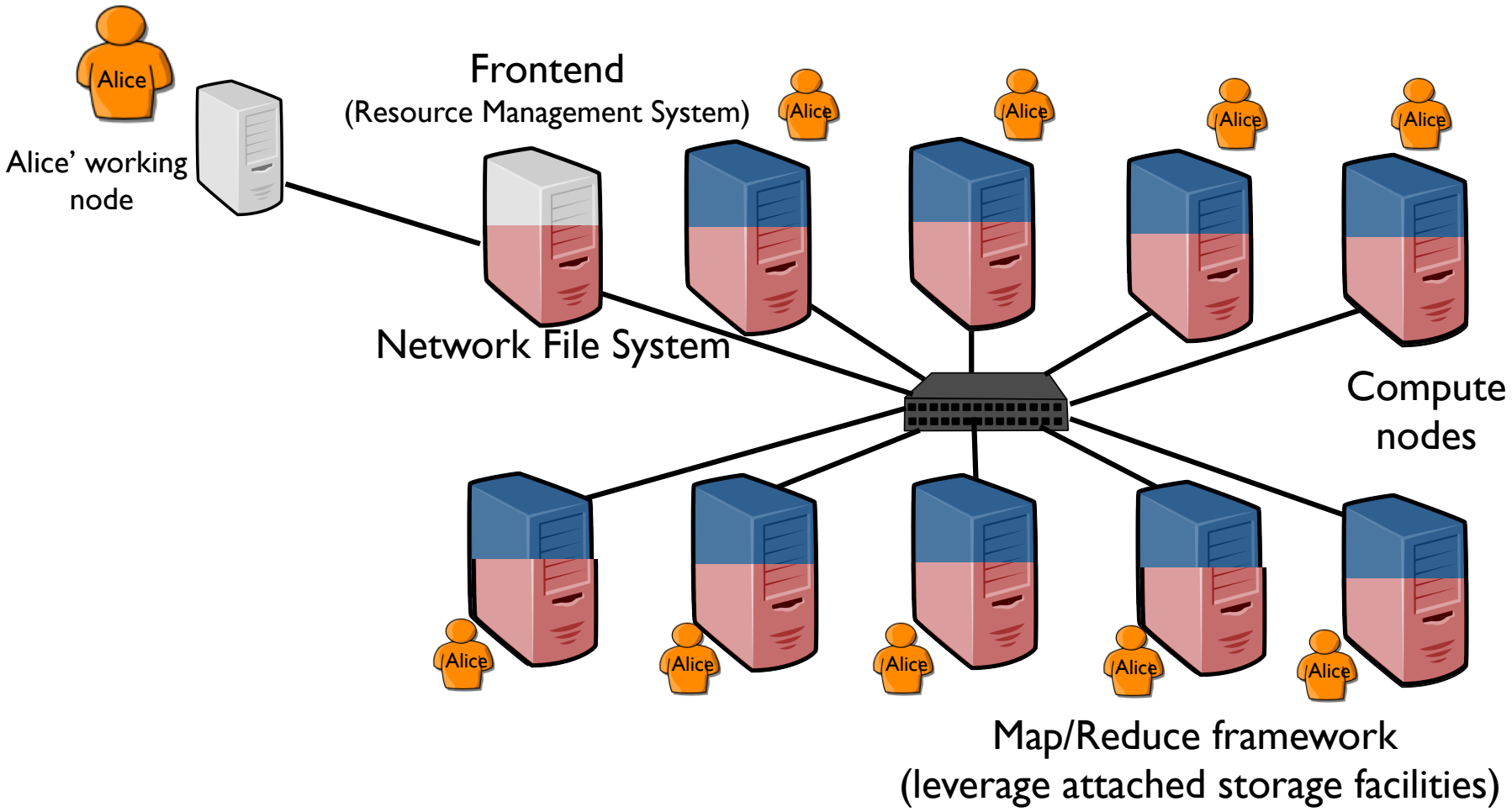
# Utility Computing and Data Management

- Network of Workstations 1990 / 20**xx**



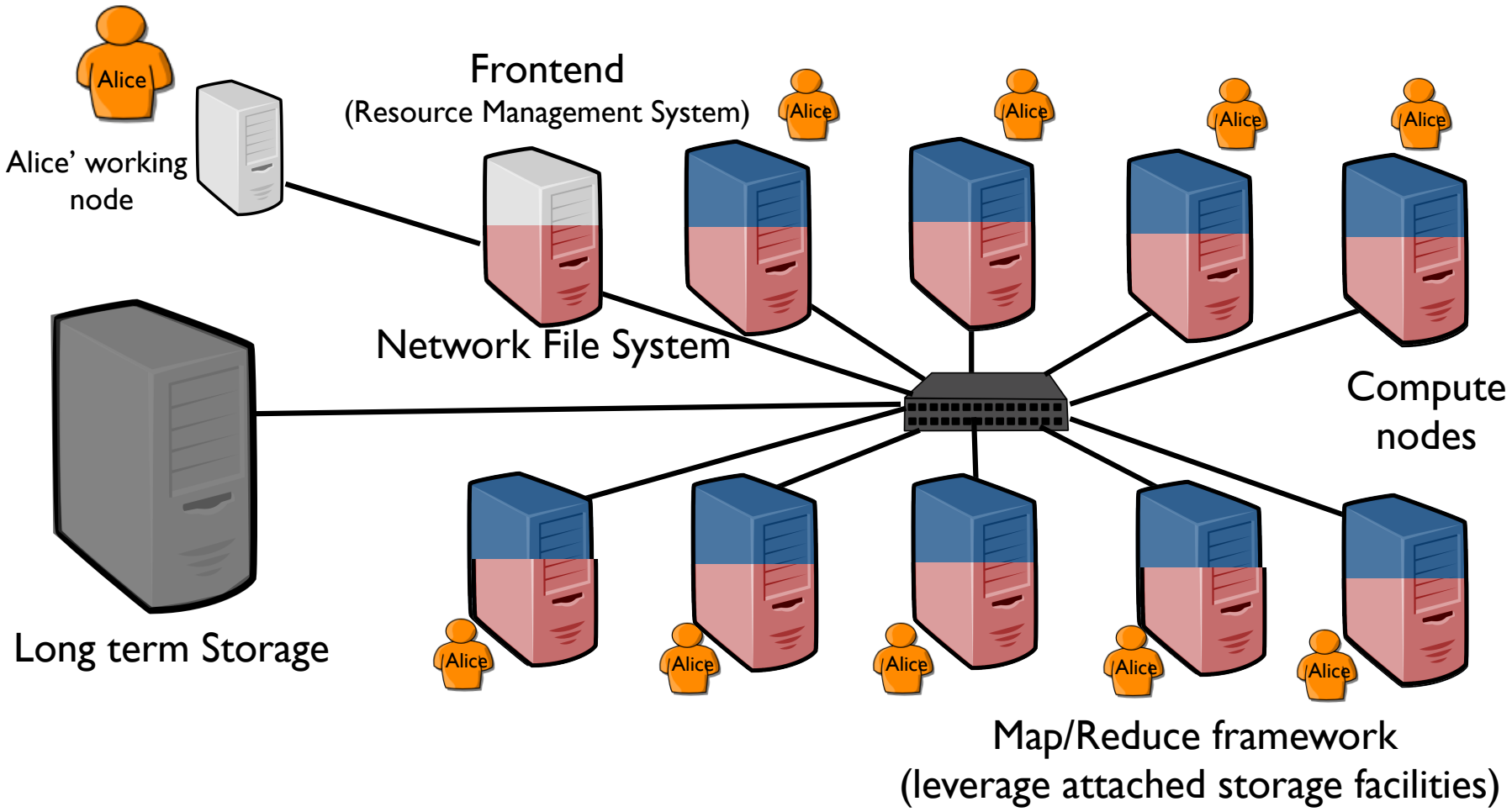
# Utility Computing and Data Management

- Network of Workstations 1990 / 20**xx**



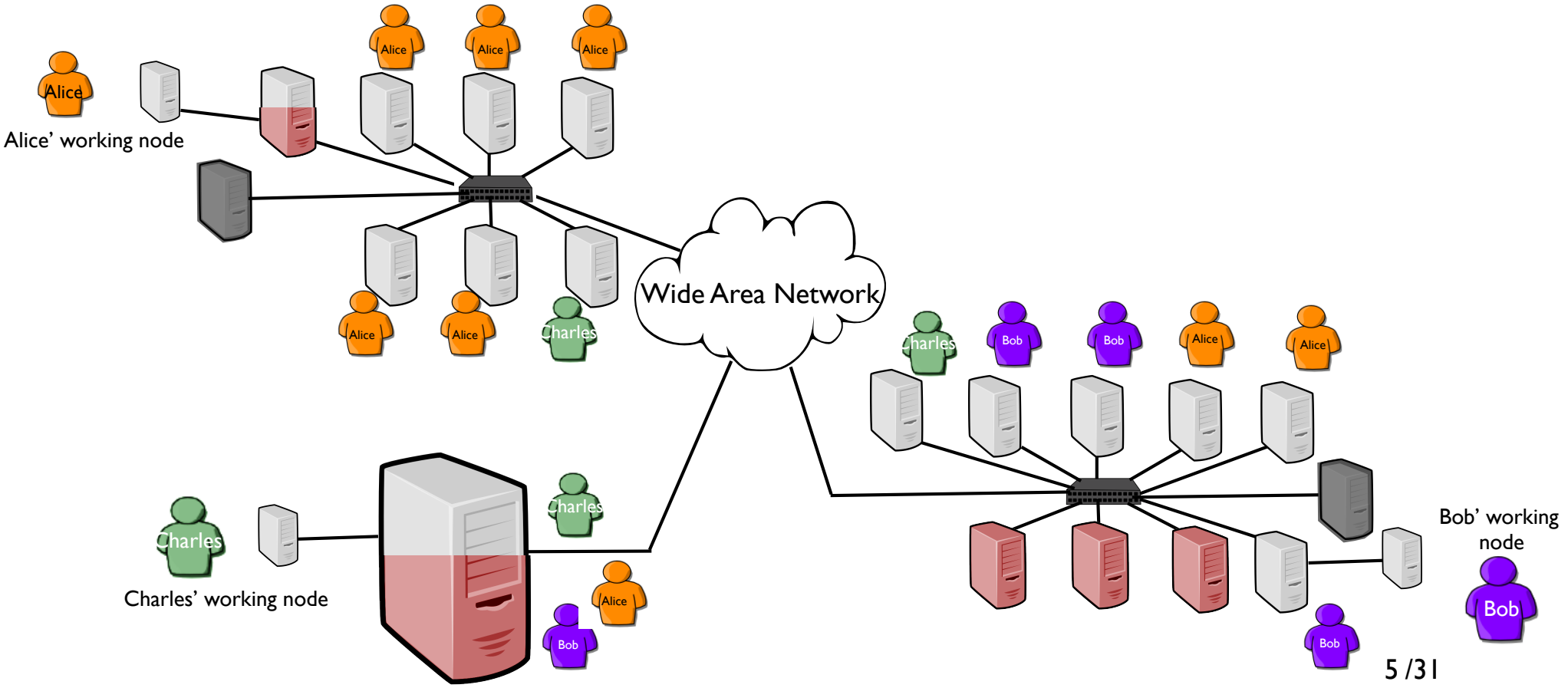
# Utility Computing and Data Management

- Network of Workstations 1990 / 20**xx**



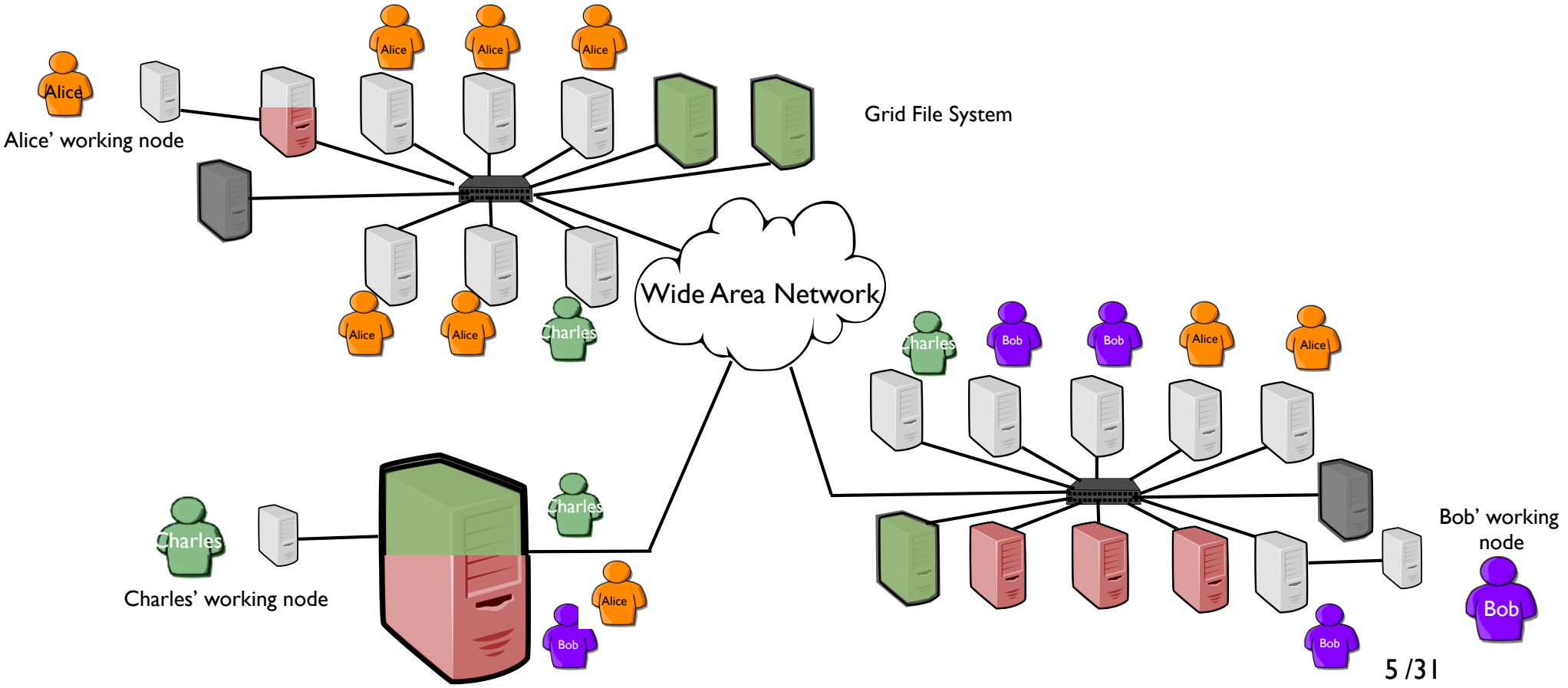
# Utility Computing and Data Management

- Network of Workstations 1990 / 20**xx**
- The Grid 1997 / 201**x**



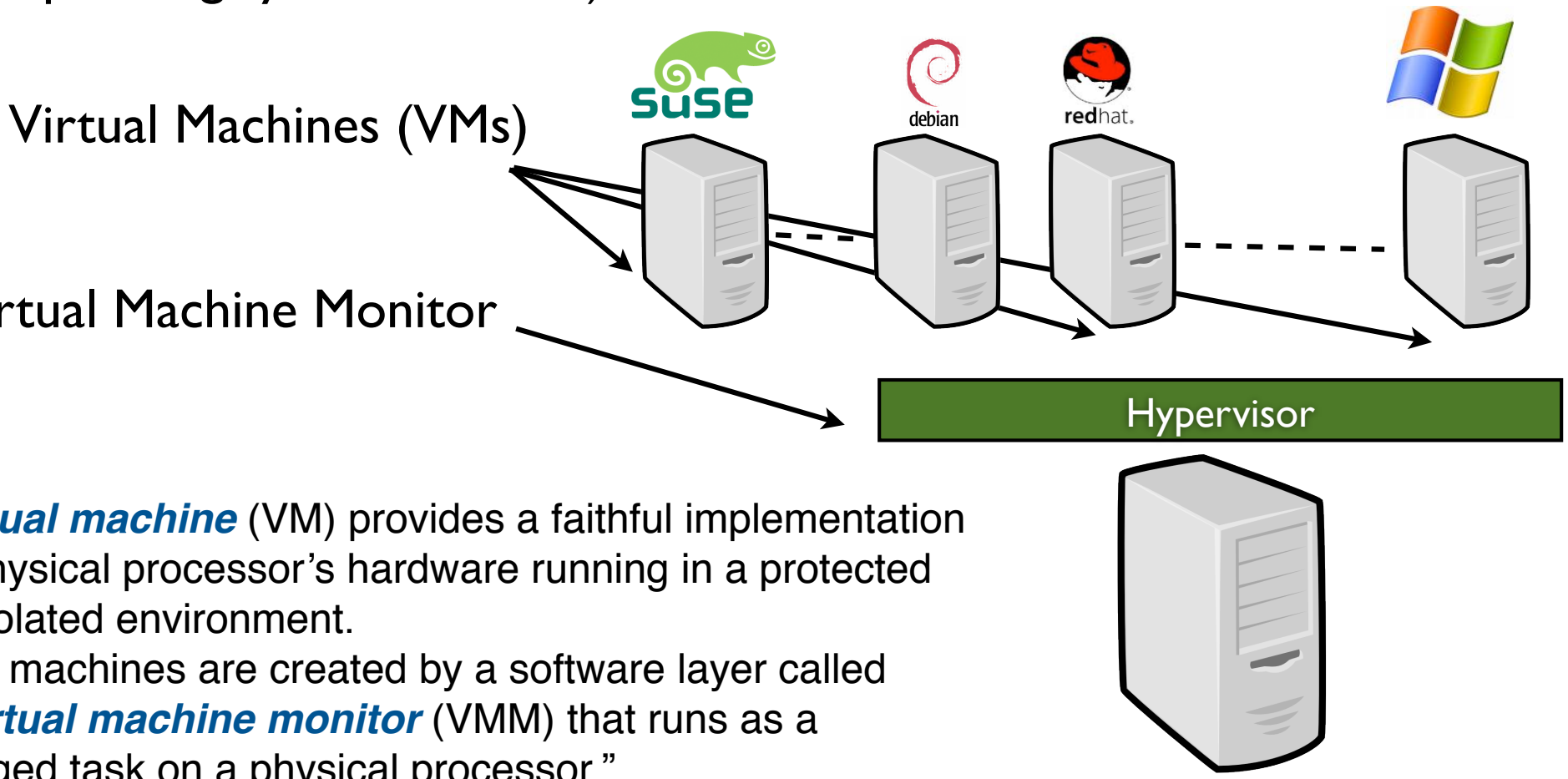
# Utility Computing and Data Management

- Network of Workstations 1990 / 20**xx**
- The Grid 1997 / 201**x**



# Here Comes *System Virtualization*

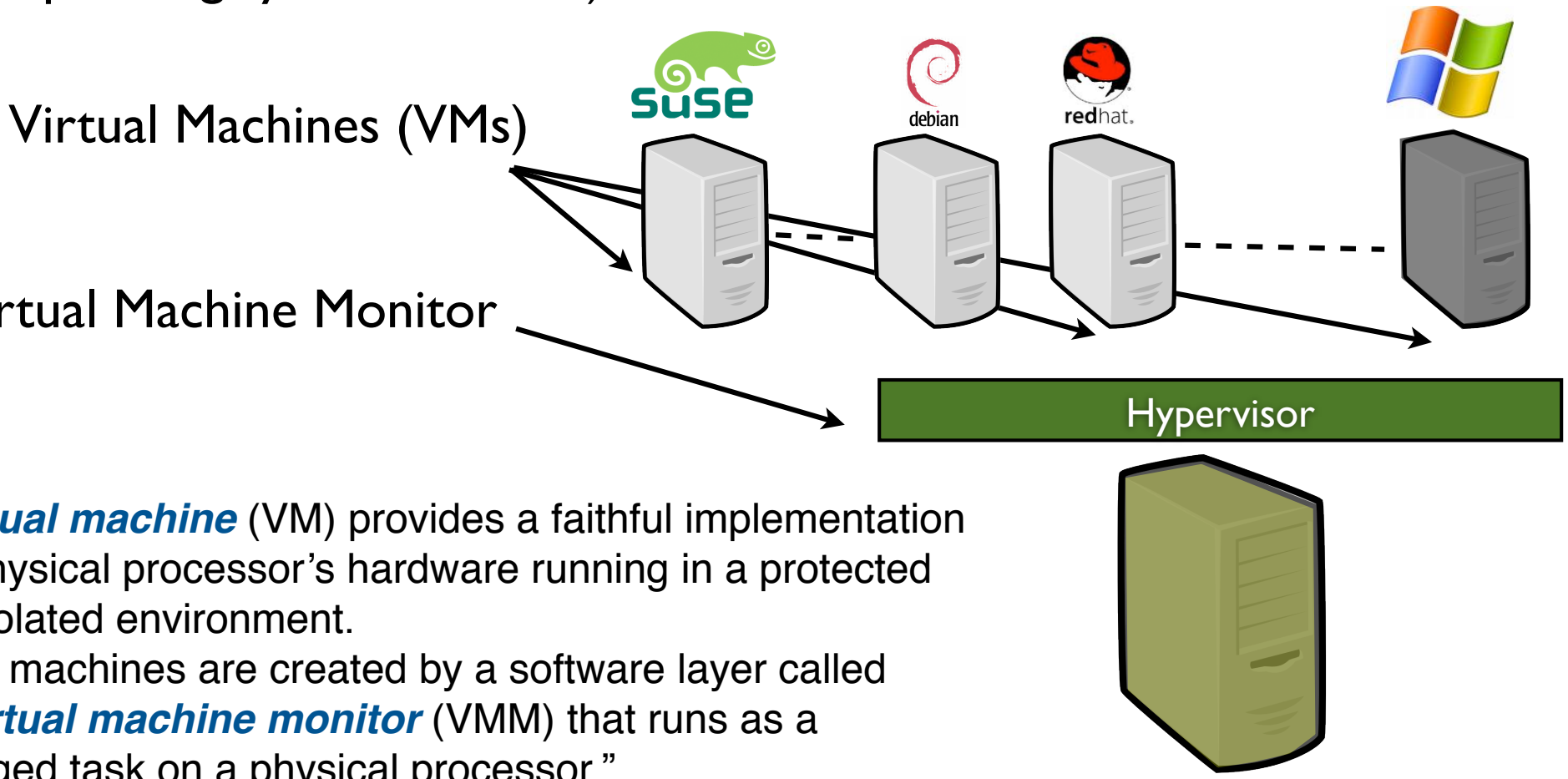
- One to multiple OSes on a physical node thanks to a hypervisor (an operating system of OSes)




"A ***virtual machine*** (VM) provides a faithful implementation of a physical processor's hardware running in a protected and isolated environment.
Virtual machines are created by a software layer called the ***virtual machine monitor*** (VMM) that runs as a privileged task on a physical processor."

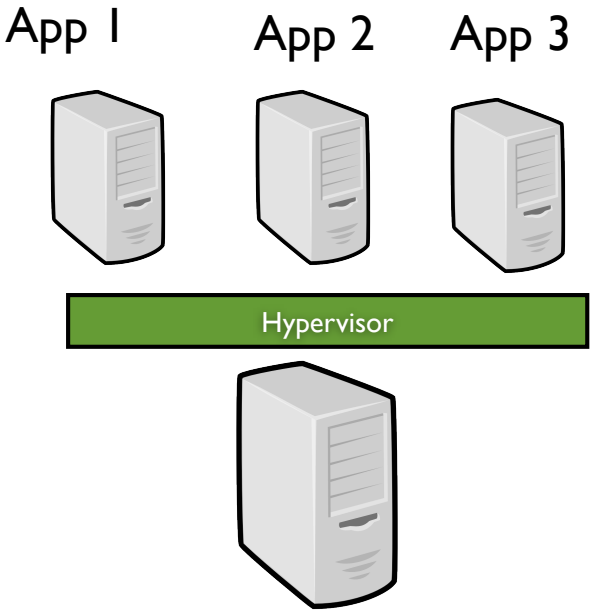
# Here Comes *System Virtualization*

- One to multiple OSes on a physical node thanks to a hypervisor (an operating system of OSes)



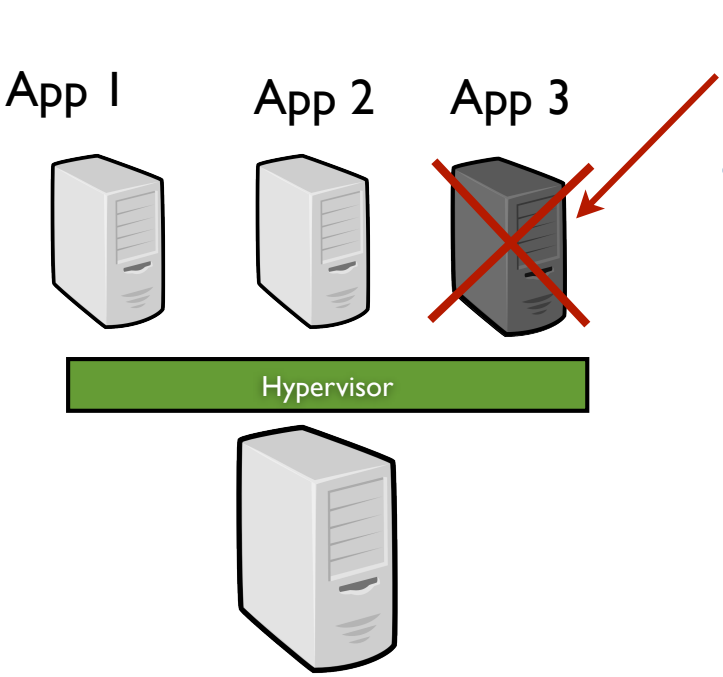

"A ***virtual machine*** (VM) provides a faithful implementation of a physical processor's hardware running in a protected and isolated environment.
Virtual machines are created by a software layer called the ***virtual machine monitor*** (VMM) that runs as a privileged task on a physical processor."

# VM Capabilities



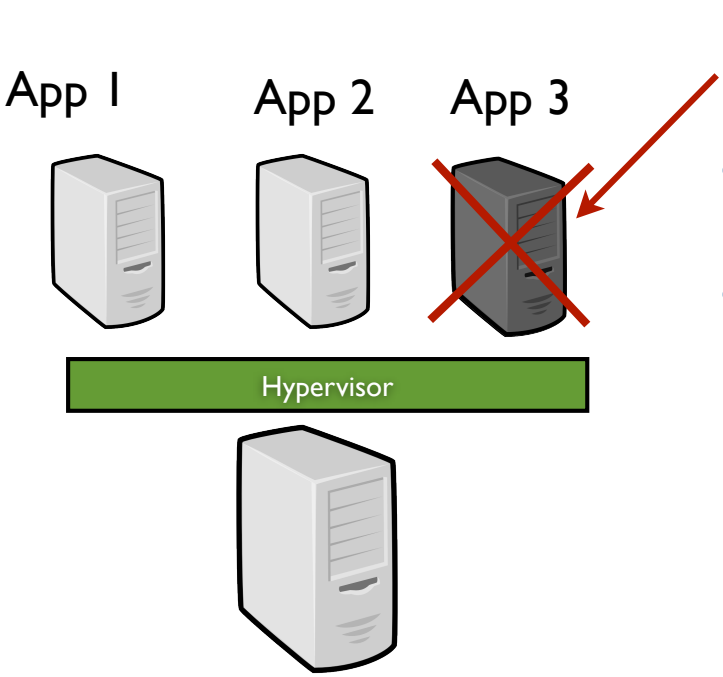

- Isolation ("security" between each VM)

# VM Capabilities



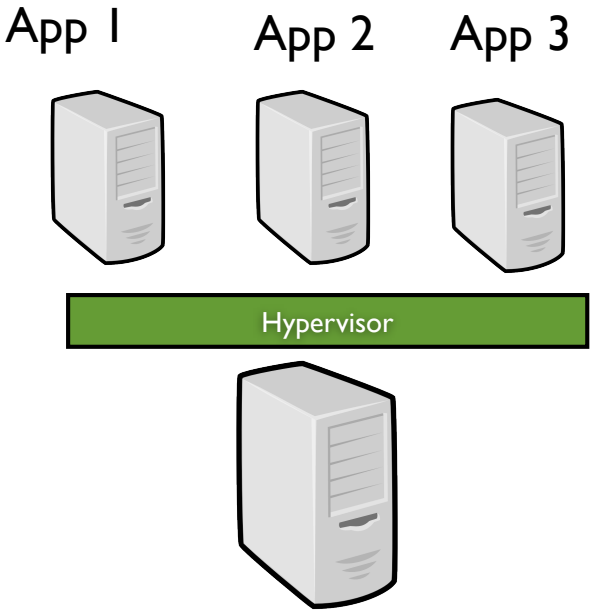

- Isolation ("security" between each VM)

# VM Capabilities



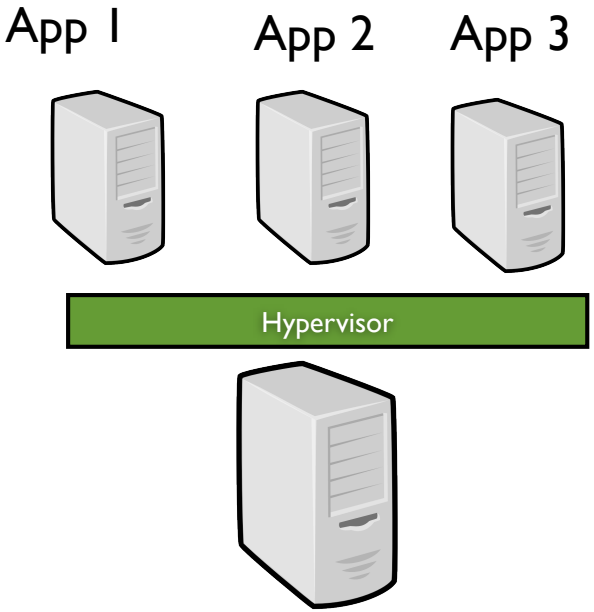

- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

# VM Capabilities



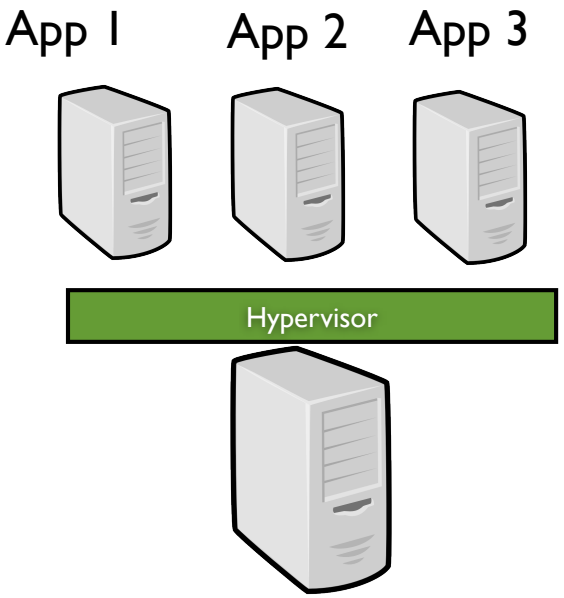

- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

# VM Capabilities

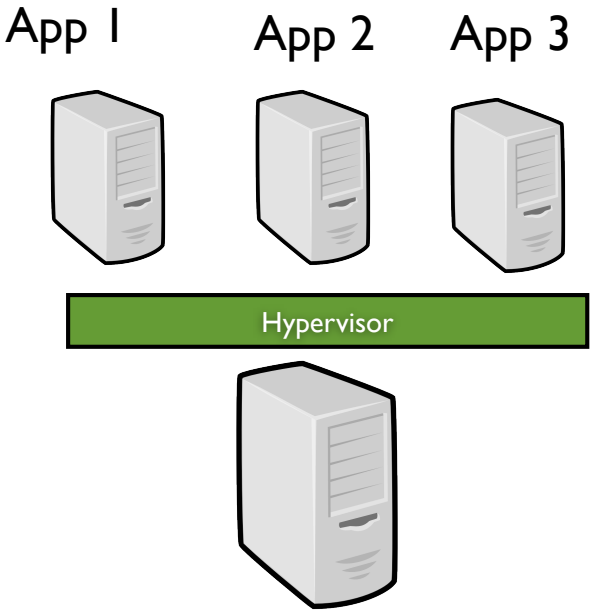



- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

- Suspend/Resume

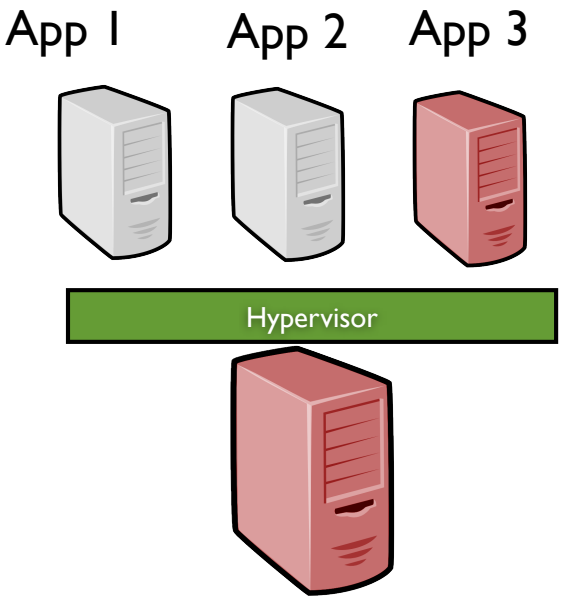


# VM Capabilities

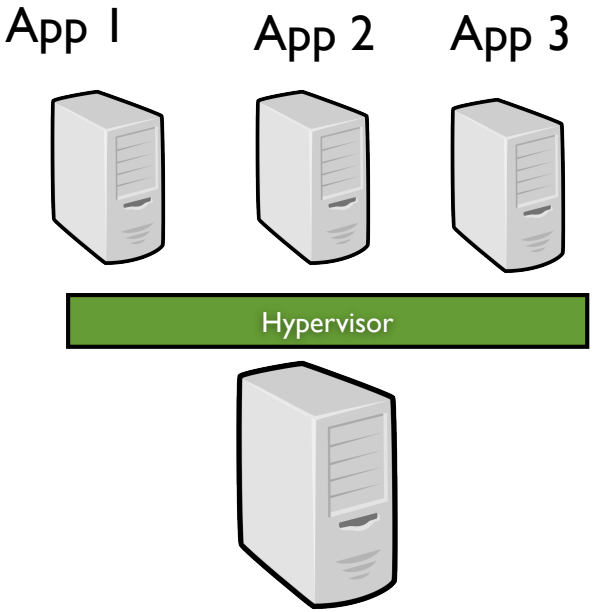



- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

- Suspend/Resume

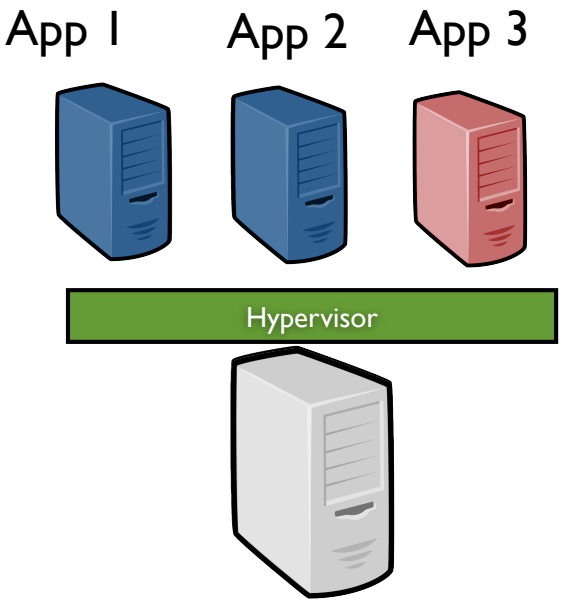


# VM Capabilities

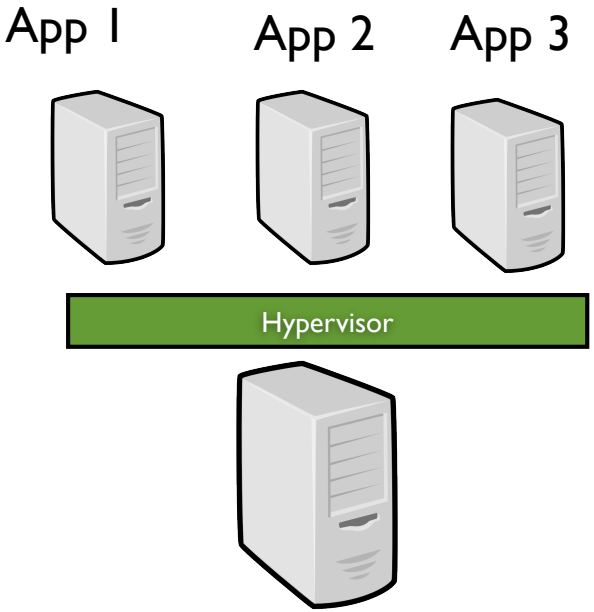



- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

- Suspend/Resume

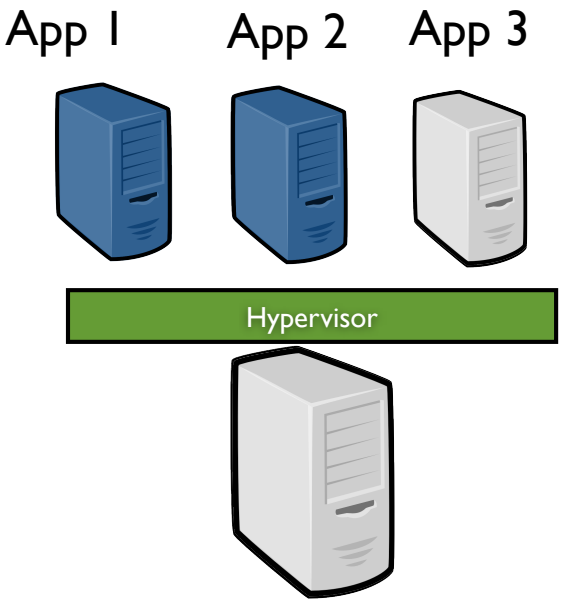


# VM Capabilities

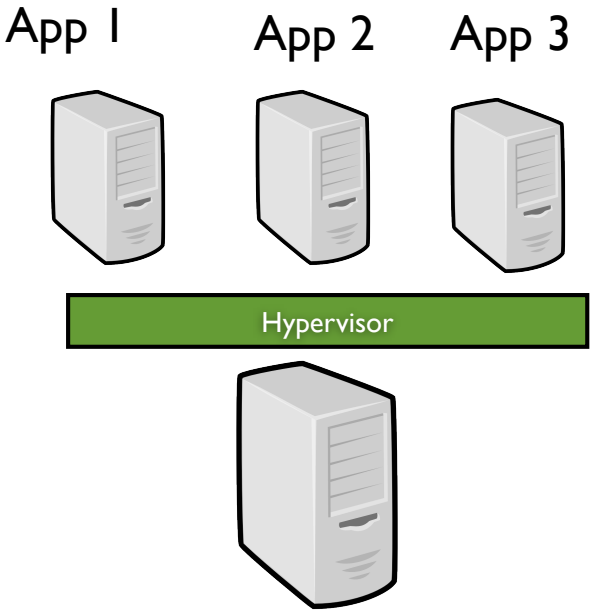



- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

- Suspend/Resume

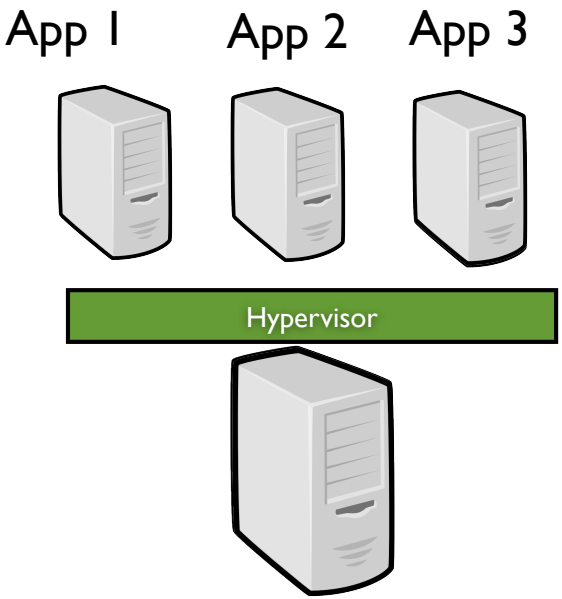


# VM Capabilities

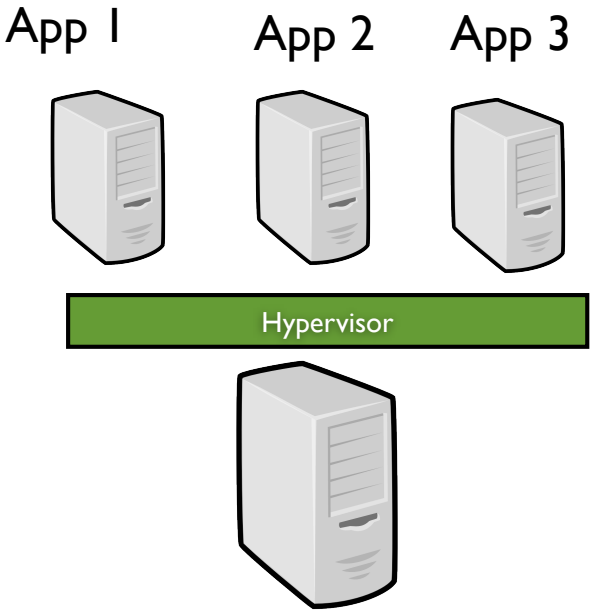



- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

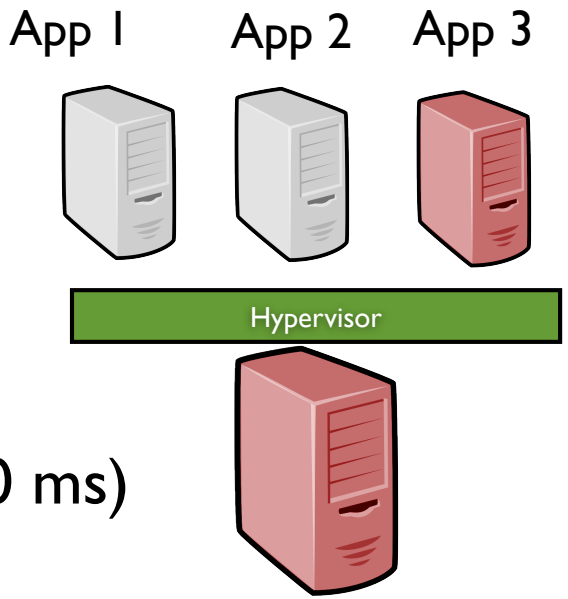
- Suspend/Resume



# VM Capabilities

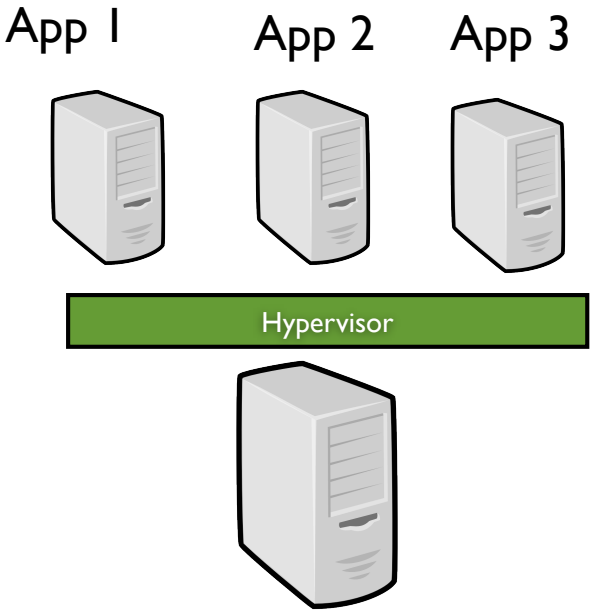



- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)



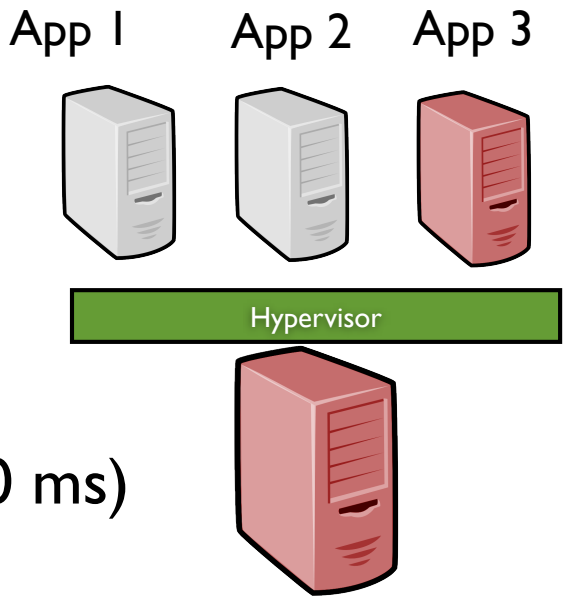

- Suspend/Resume
- Live migration
  (negligible downtime ~ 60 ms)
  Post/Pre Copy

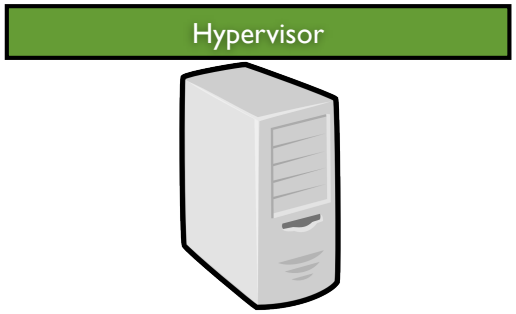
# VM Capabilities



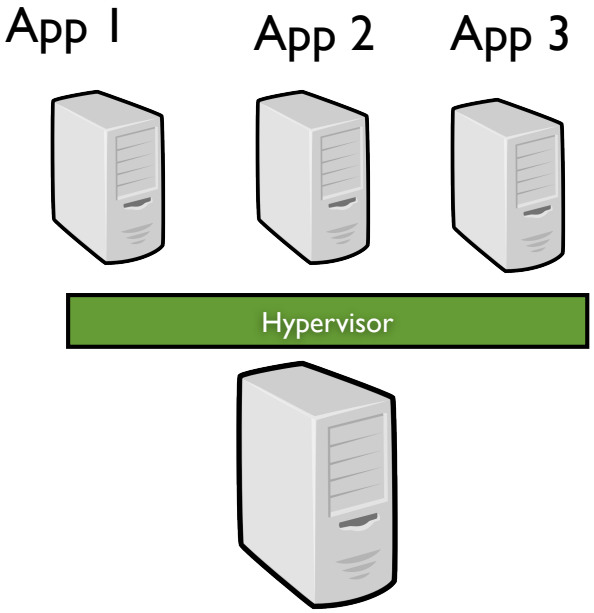

- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

- Suspend/Resume
- Live migration
  (negligible downtime ~ 60 ms)
  Post/Pre Copy



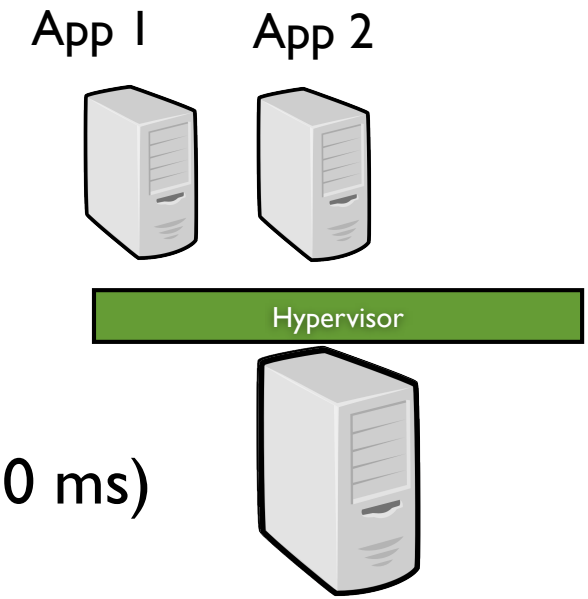



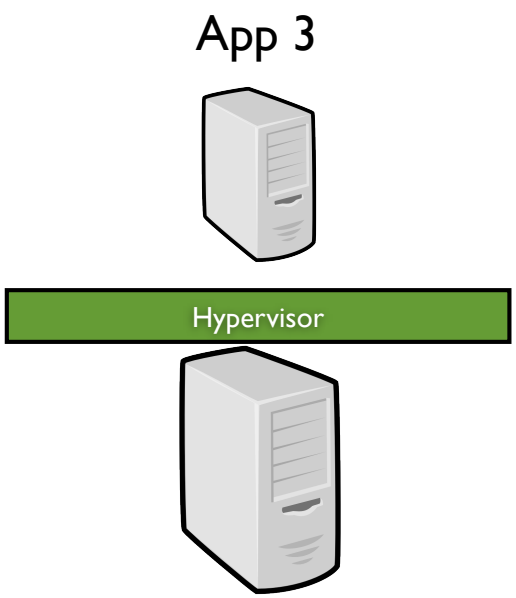
# VM Capabilities



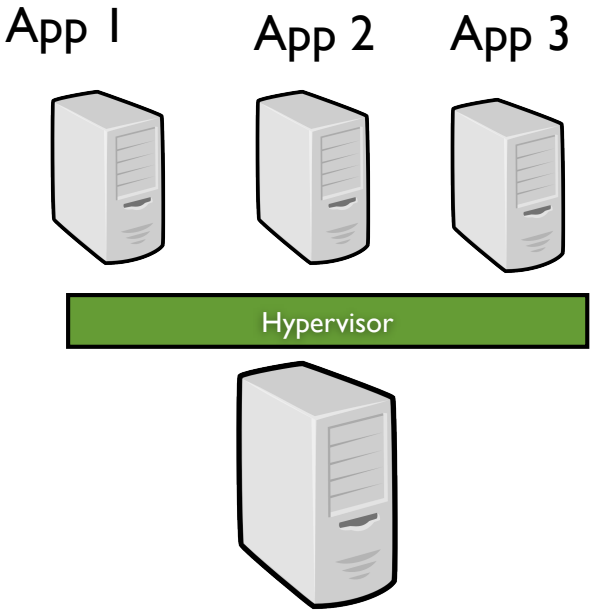

- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

- Suspend/Resume
- Live migration
  (negligible downtime ~ 60 ms)
  Post/Pre Copy



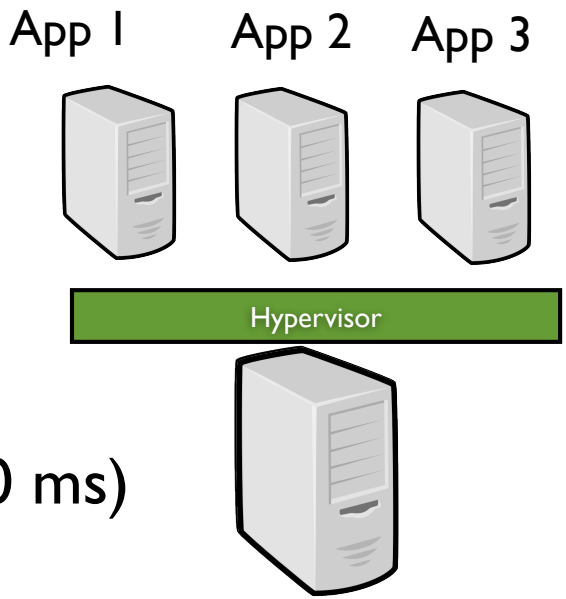



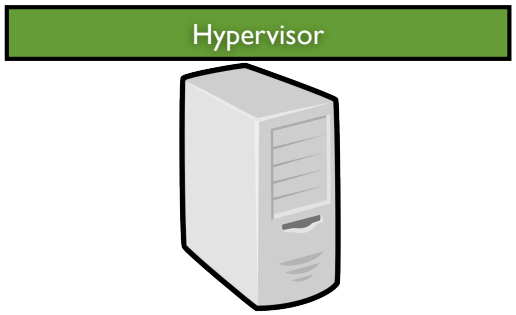
# VM Capabilities



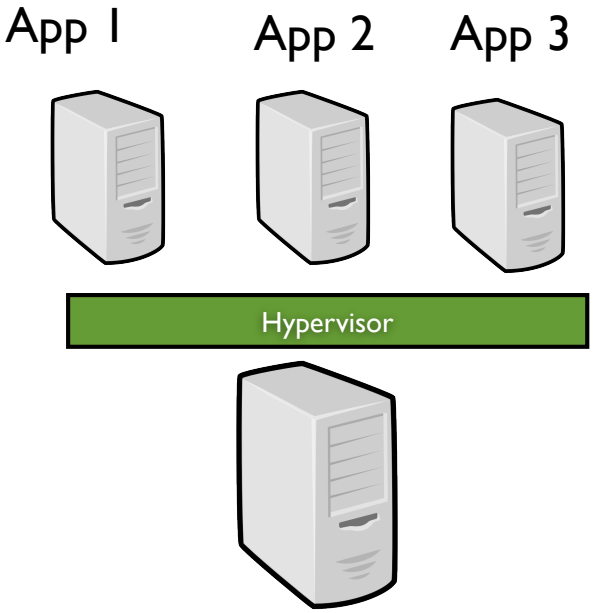

- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)

- Suspend/Resume
- Live migration
  (negligible downtime ~ 60 ms)
  Post/Pre Copy

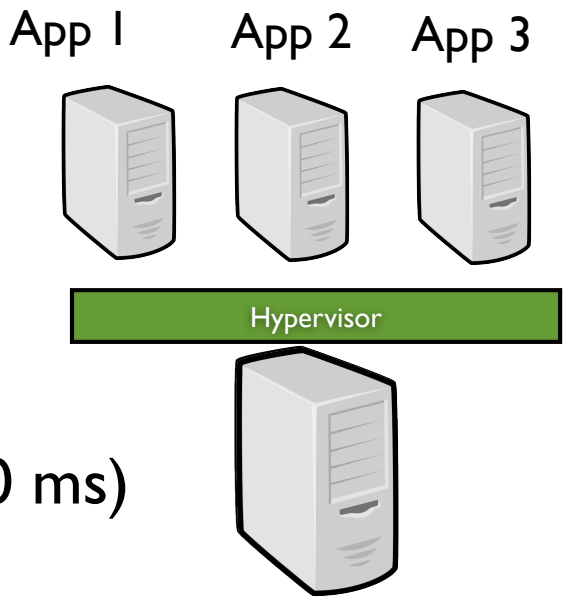


# VM Capabilities



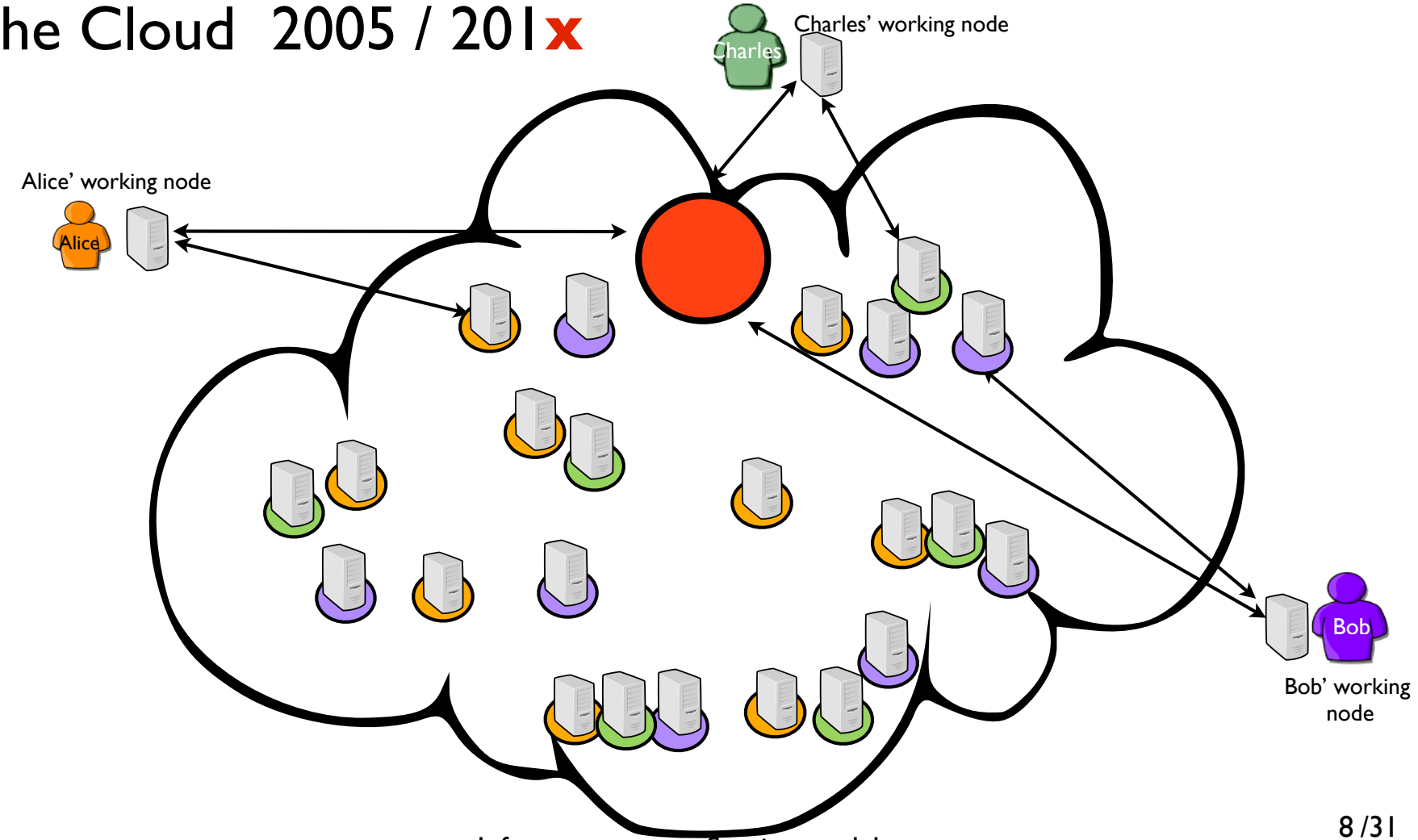

- Isolation ("security" between each VM)
- Snapshotting (a VM can be easily resumed from its latest consistent state)




- Suspend/Resume
- Live migration
  (negligible downtime ~ 60 ms)
  Post/Pre Copy

# Utility Computing and Data Management

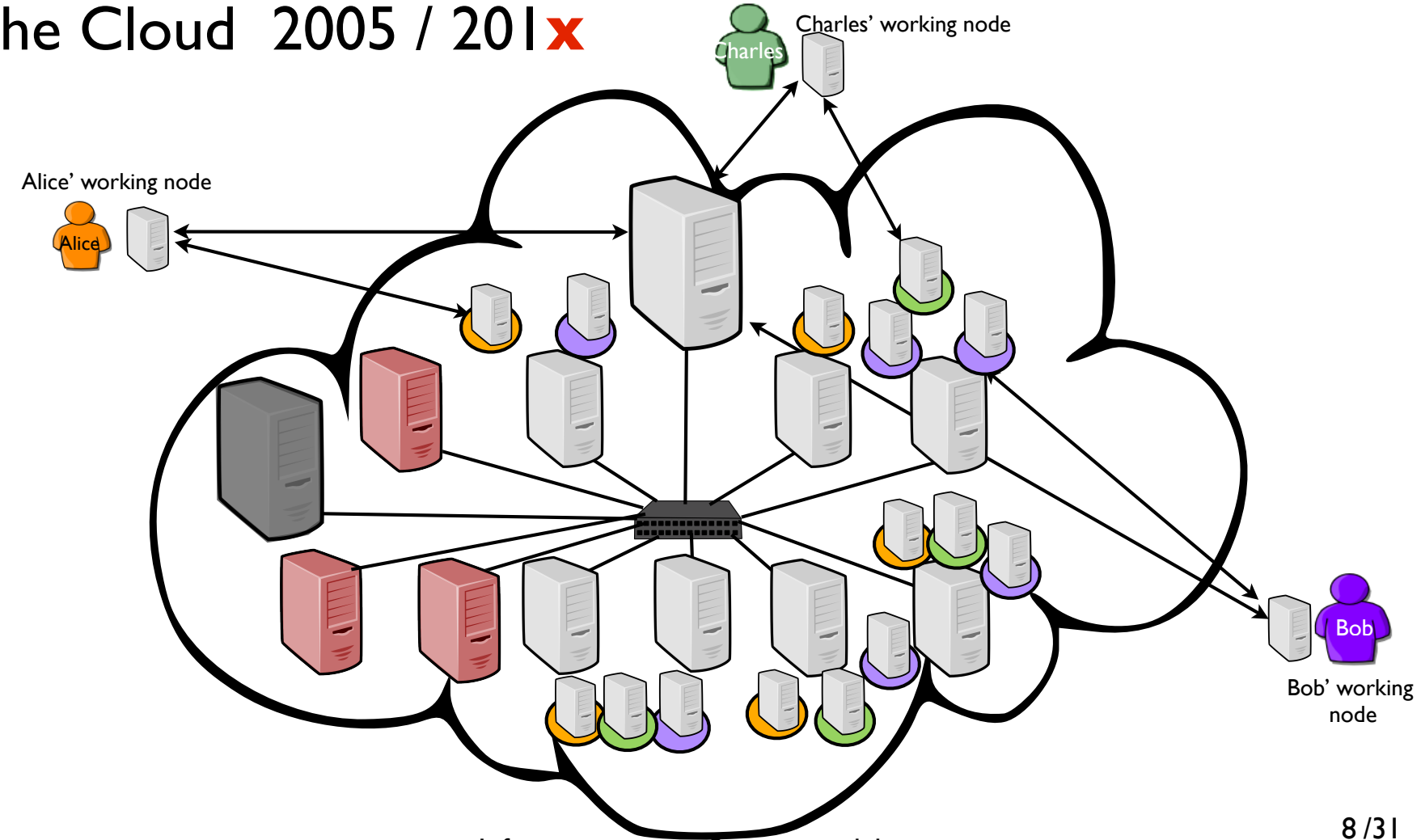
- Network of Workstations 1990 / 20**xx**
- The Grid 1997 / 201**x**
- The Cloud 2005 / 201**x**




Infrastructure as a Service model

# Utility Computing and Data Management

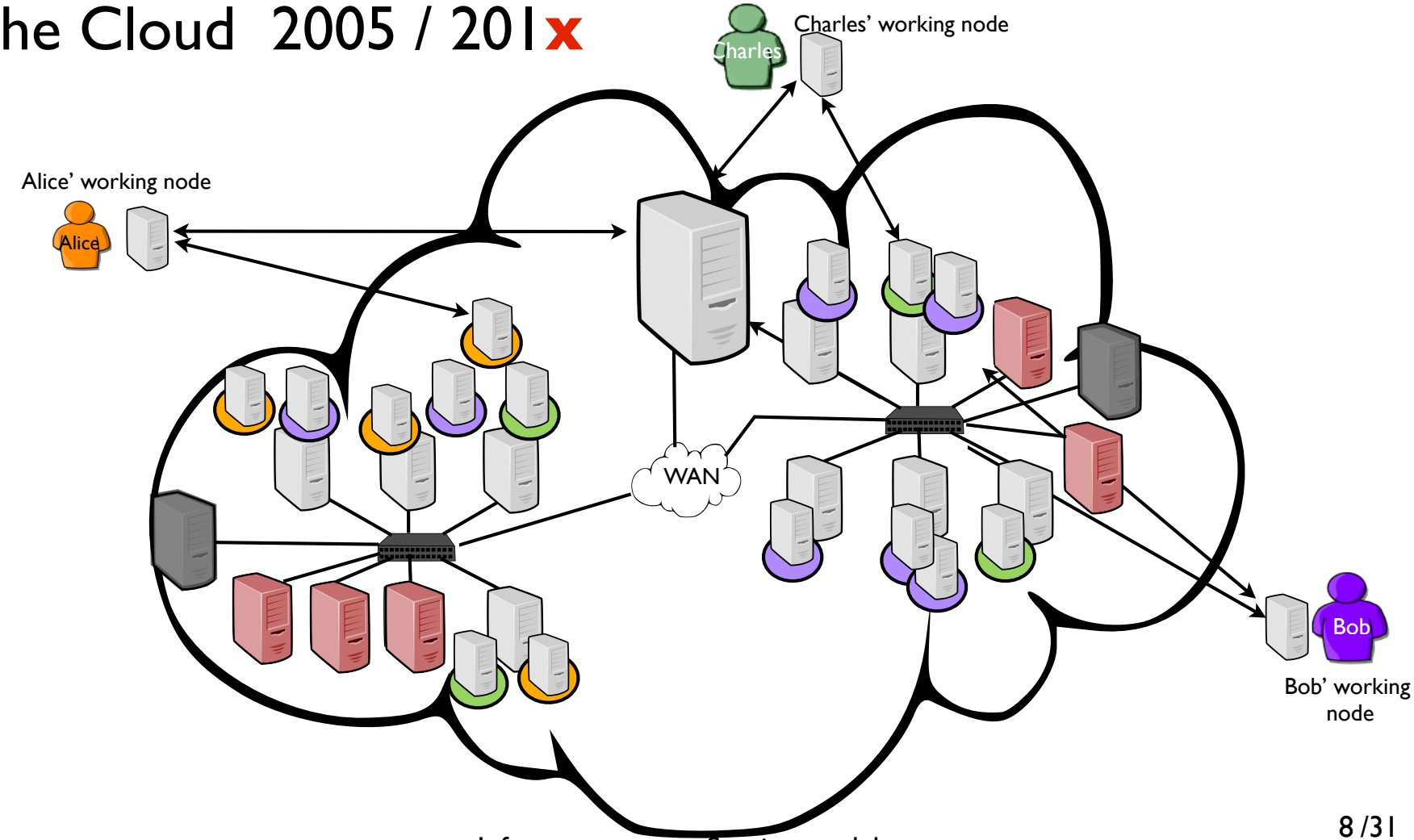
- Network of Workstations 1990 / 20**xx**
- The Grid 1997 / 201**x**
- The Cloud 2005 / 201**x**




Infrastructure as a Service model

# Utility Computing and Data Management

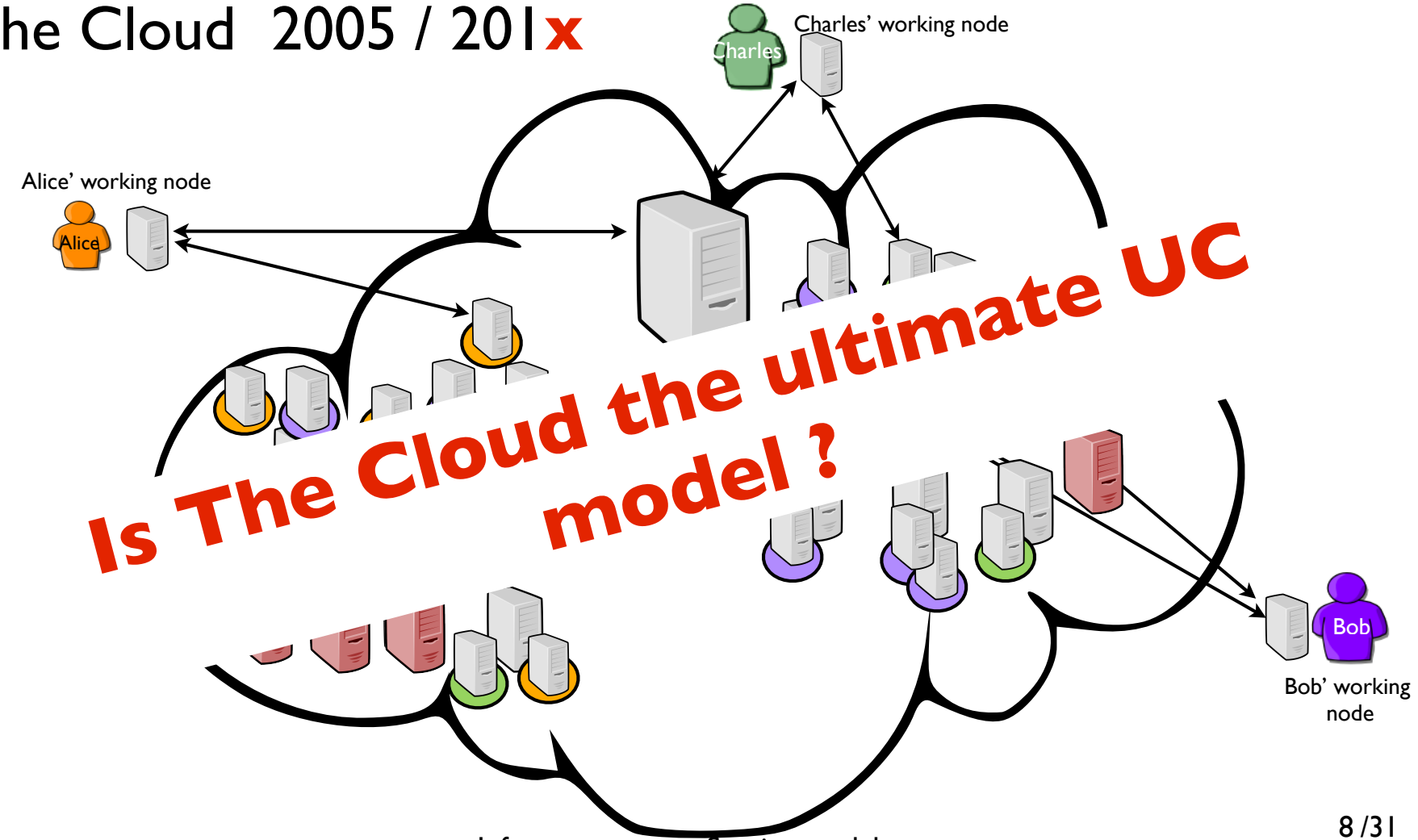
- Network of Workstations 1990 / 20**xx**
- The Grid 1997 / 201**x**
- The Cloud 2005 / 201**x**




Infrastructure as a Service model

# Utility Computing and Data Management

- Network of Workstations 1990 / 20**xx**
- The Grid 1997 / 201**x**
- The Cloud 2005 / 201**x**



# Operating Large Offshore Centralized Infrastructures

# Managing IaaS - OpenSource solutions

- Academic proposals

## Nimbus (Freeman and Keahey, University of Chicago)

Based on GT4 and the Globus Virtual Workspace Service
Target: cloud for science
Tutorials and documentation in "grid space"


## Open Nebula (Montero & Llorente, DSA-Research at UCM)

Support for the Xen, KVM and VMware
Access to Amazon EC2 (cloud bursting)
Probably, the most deployed


## Eucalyptus (Wolsky, University of Santa Barbara)

Web services based implementation of elastic/utility/cloud computing infrastructure

# Managing IaaS - OpenSource solutions

- Proprietary proposals
- vCloud/vSphere (vmware) 60%

  ESXi

- XenServer/Xen Cloud platform (20%)

  Xen

- Microsoft System Center VM (20%)

  Hypervisors agnostic




Credits: http://www.v-index.com
Virtualization Industry Quaterly Survey

# Managing IaaS - OpenSource solutions

- Community proposals

## OpenStack


Supported by several industrials
Recently selected by Ubuntu for the core of their cloud proposal

## CloudStack


Supported by CITRIX
Full JAVA implementation
Apache project

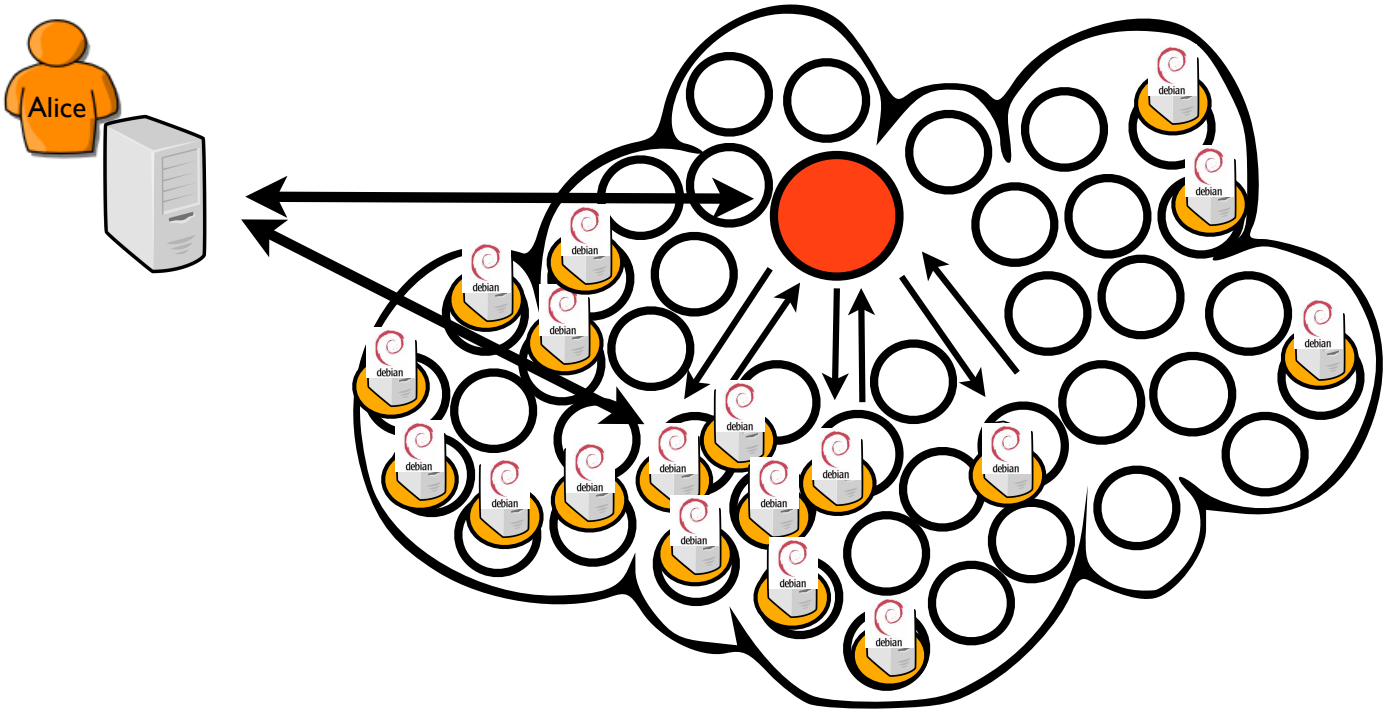
# Putting everything into the cloud

- Mature for one site !

  More flexibility ! ?
  Infinite resources ! ?

- New concerns !?

  Scalability



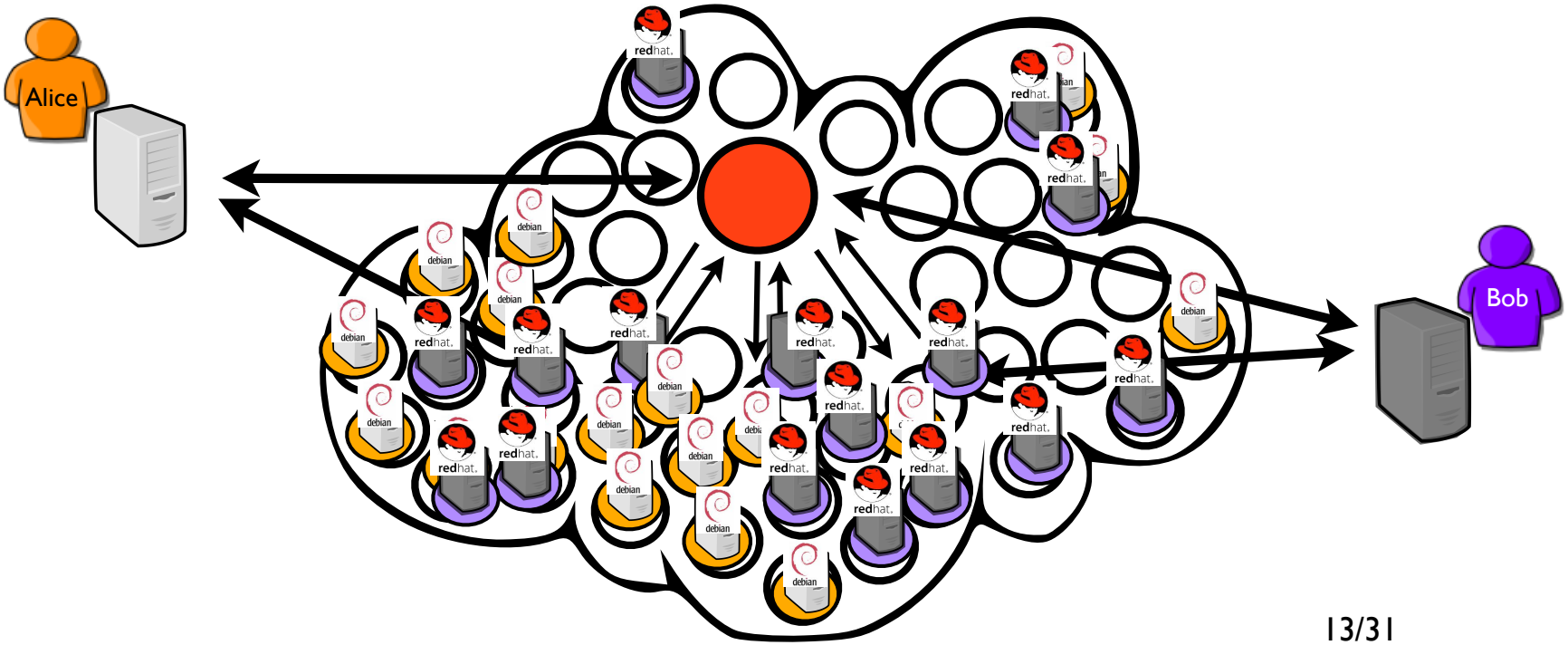
# Putting everything into the cloud

- Mature for one site !

  More flexibility ! ?
  Infinite resources ! ?

- New concerns !?

  Scalability



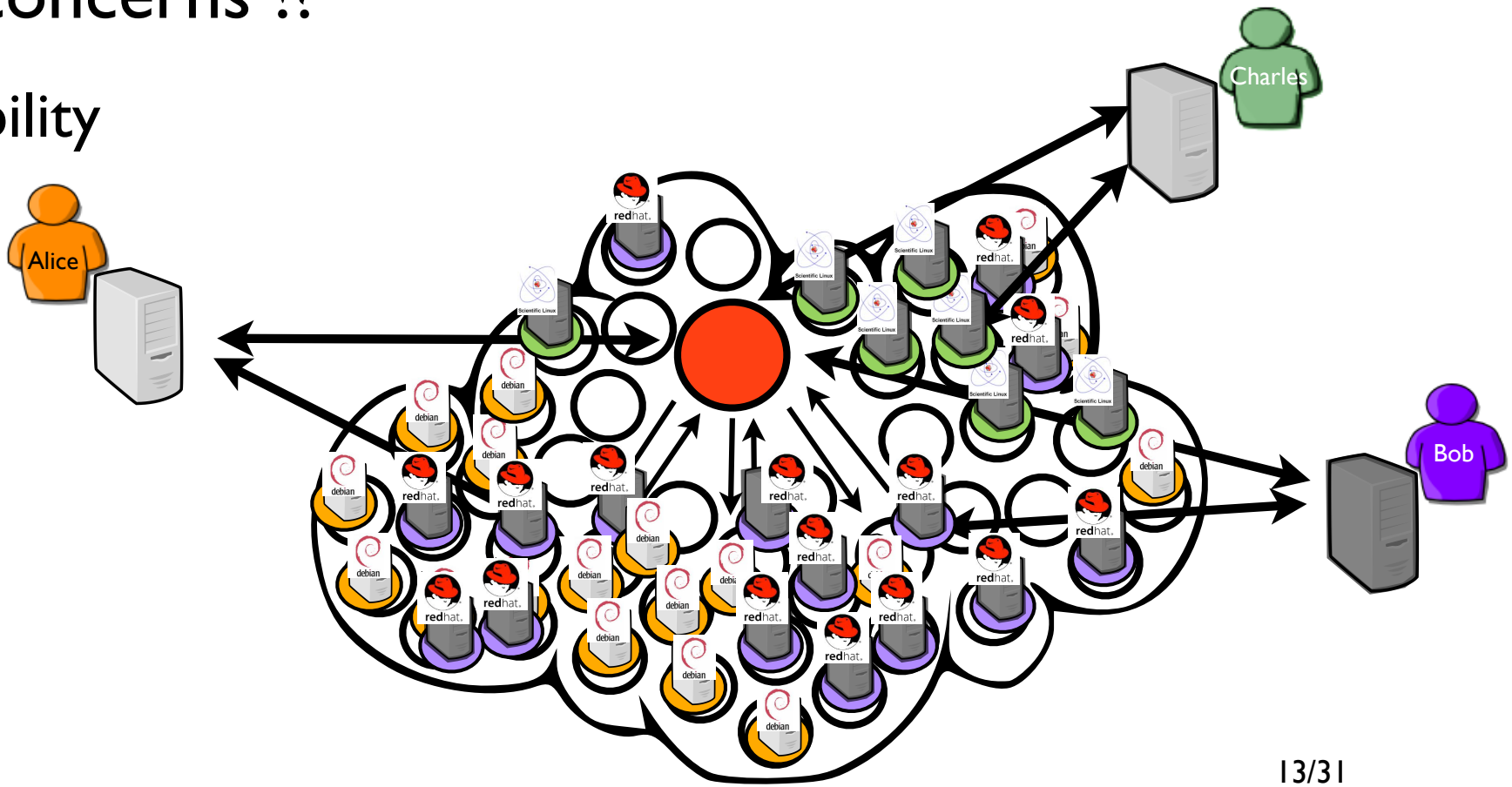
# Putting everything into the cloud

- Mature for one site !

  More flexibility ! ?
  Infinite resources ! ?

- New concerns !?

  Scalability



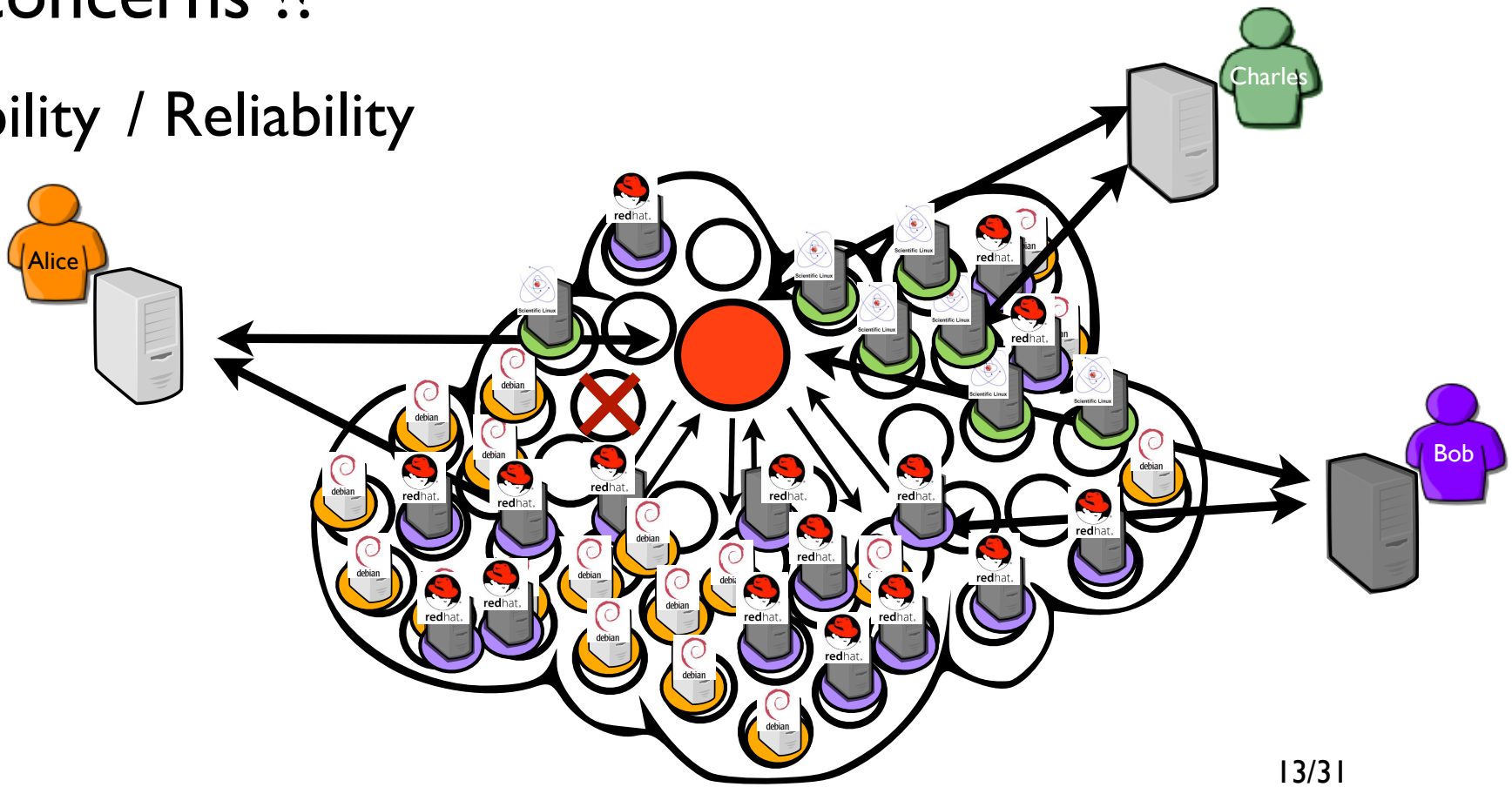
# Putting everything into the cloud

- Mature for one site !

  More flexibility ! ?
  Infinite resources ! ?

- New concerns !?

  Scalability / Reliability



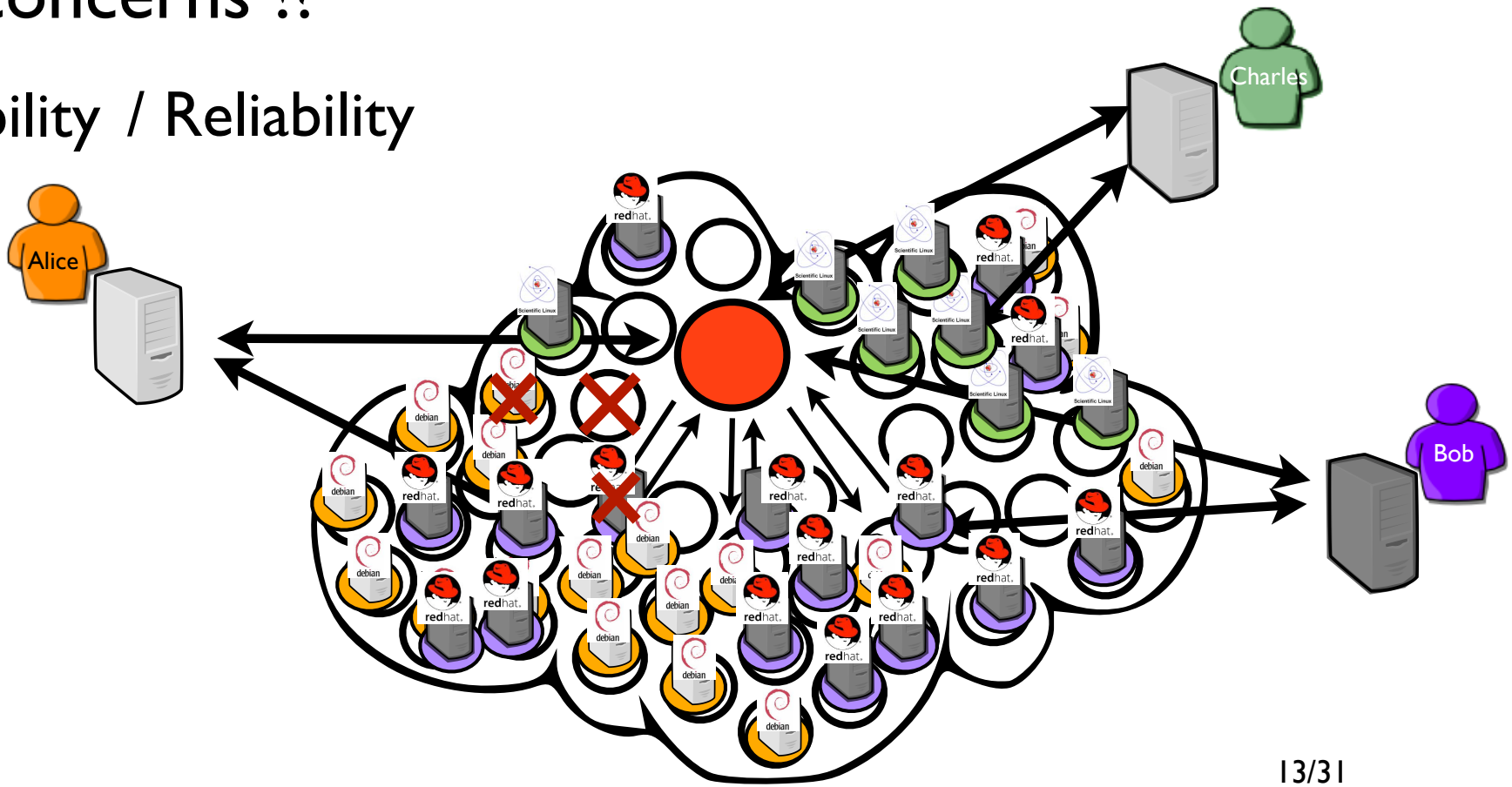
# Putting everything into the cloud

- Mature for one site !

  More flexibility ! ?
  Infinite resources ! ?

- New concerns !?

  Scalability / Reliability



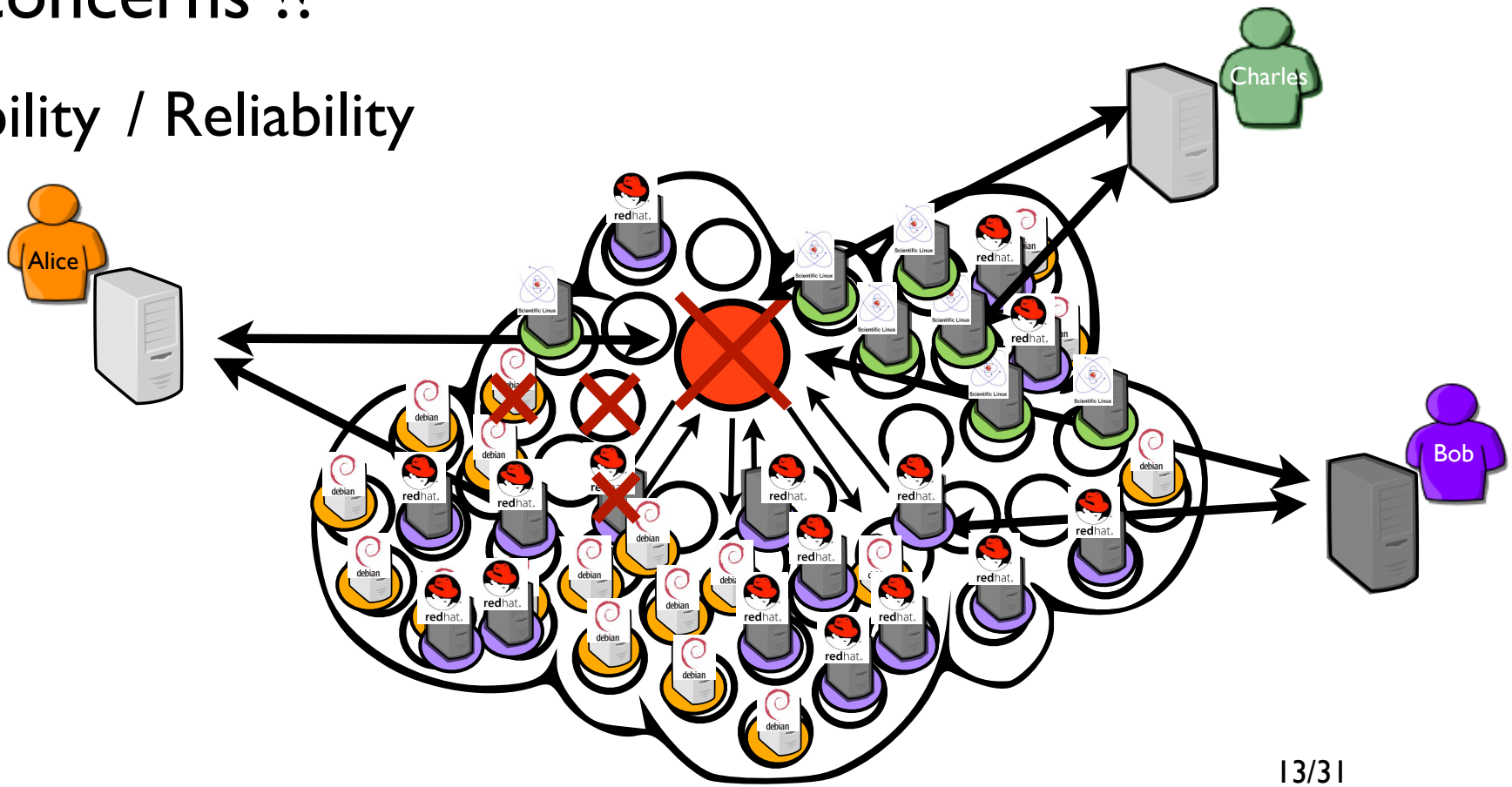
# Putting everything into the cloud

- Mature for one site !

  More flexibility ! ?
  Infinite resources ! ?

- New concerns !?

  Scalability / Reliability



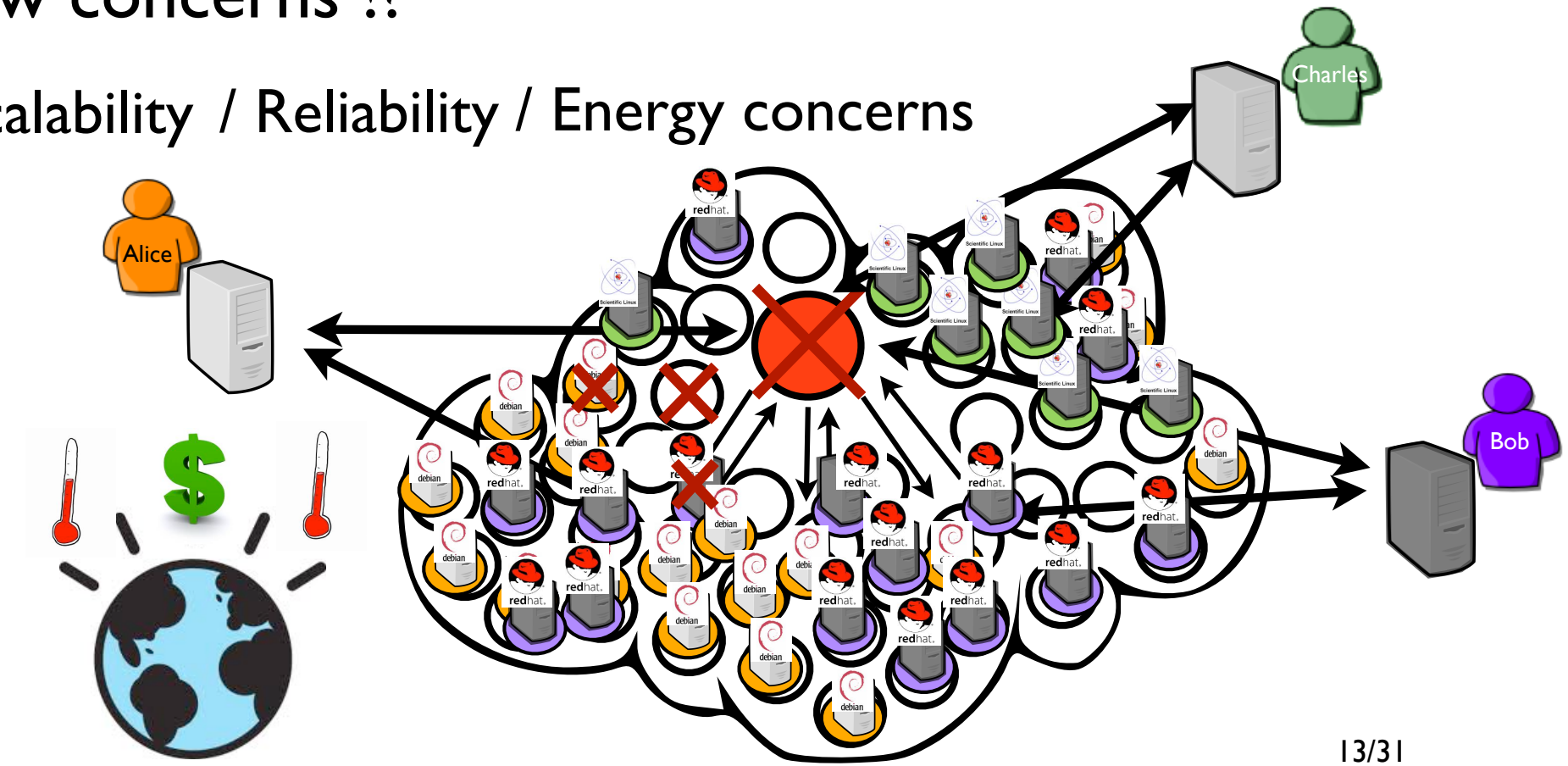
# Putting everything into the cloud

- Mature for one site !

  More flexibility ! ?
  Infinite resources ! ?

- New concerns !?

  Scalability / Reliability / Energy concerns



# Putting everything into the cloud

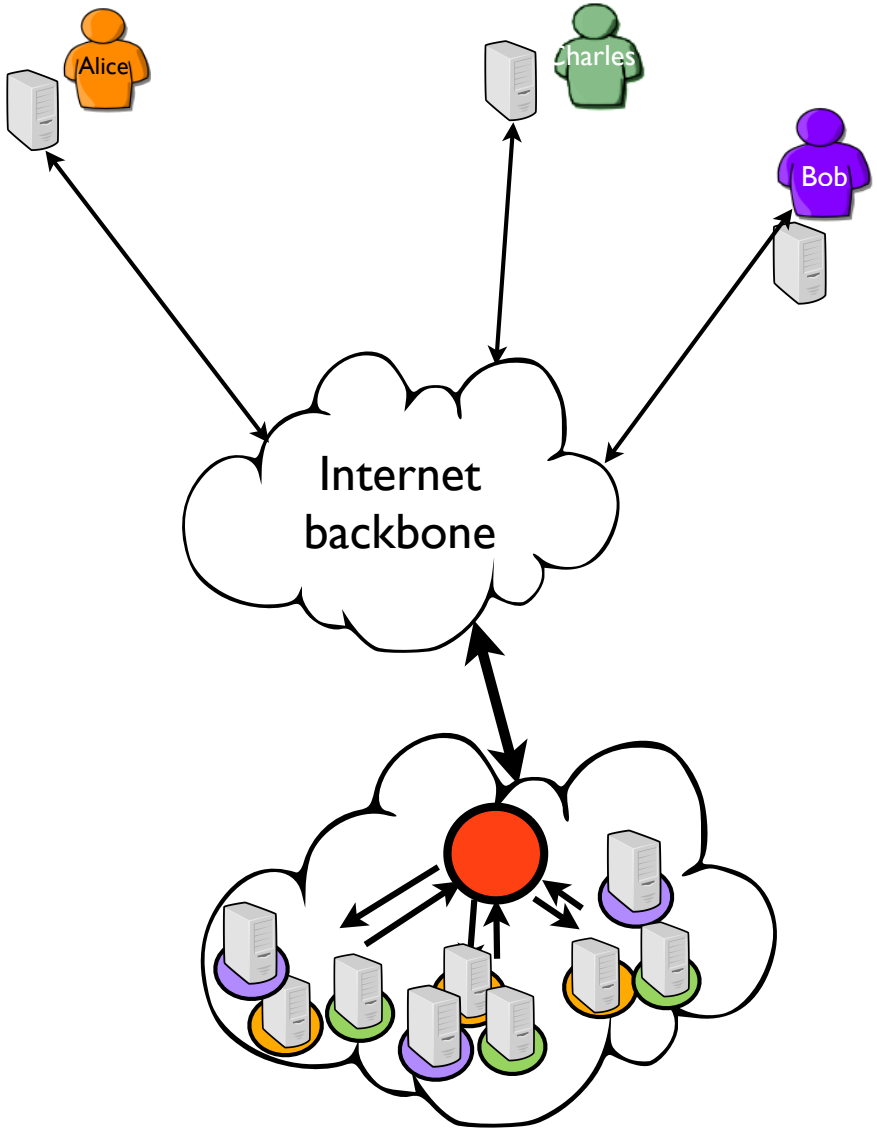
- Inherent limitations of the cloud computing model w.r.t data concerns and public offers

1. Externalization of private applications/ data (jurisdiction concerns)

2. Overhead implied by the unavoidable use of the Internet to reach distant platforms

3. The connectivity to the application/data cannot be insured by centralized dedicated centers (disaster recovery)

4. Energy concerns (footprint but also physical limitations)



# Putting everything into the cloud

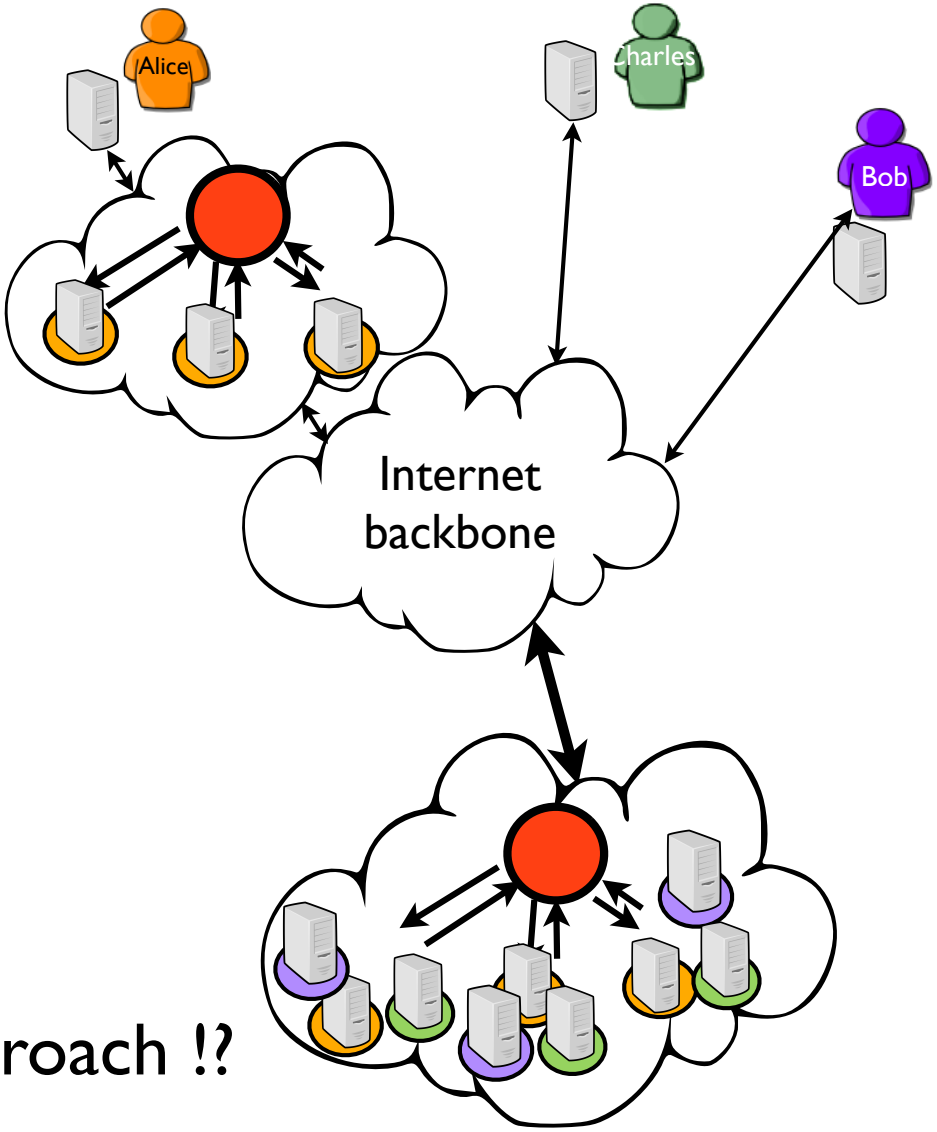
- Inherent limitations of the cloud computing model w.r.t data concerns and public offers

  1. Externalization of private applications/ data (jurisdiction concerns)

  2. Overhead implied by the unavoidable use of the Internet to reach distant platforms

  3. The connectivity to the application/data cannot be insured by centralized dedicated centers (disaster recovery)

  4. Energy concerns (footprint but also physical limitations)




- Hybrid platforms: a promising approach !?

# Putting everything into the cloud

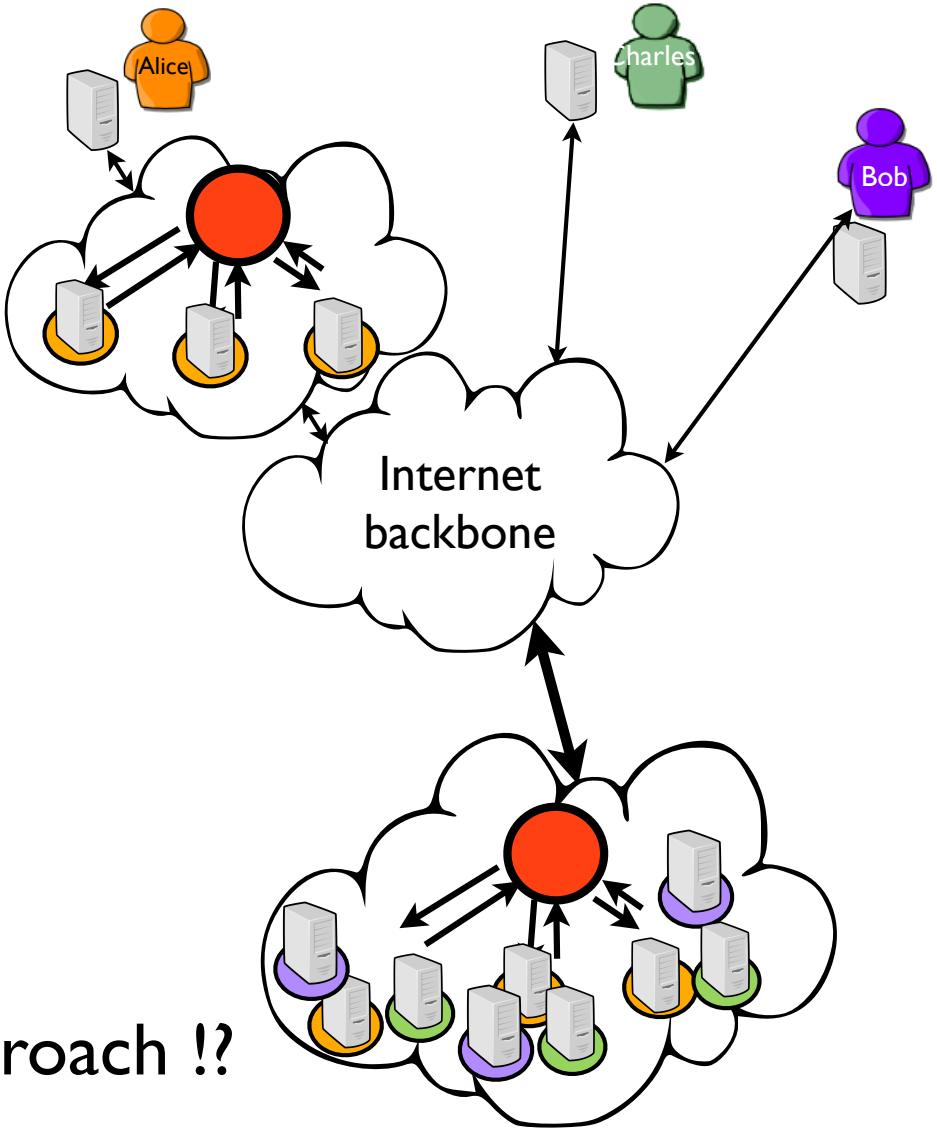
- Inherent limitations of the cloud computing model w.r.t data concerns and public offers

  1. Externalization of private applications/ data (jurisdiction concerns)

  2. Overhead implied by the unavoidable use of the Internet to reach distant platforms

  3. The connectivity to the application/data cannot be insured by centralized dedicated centers (disaster recovery)

  4. Energy concerns (footprint but also physical limitations)




- Hybrid platforms: a promising approach !?

  Not really (points 1 and 2 still persist)

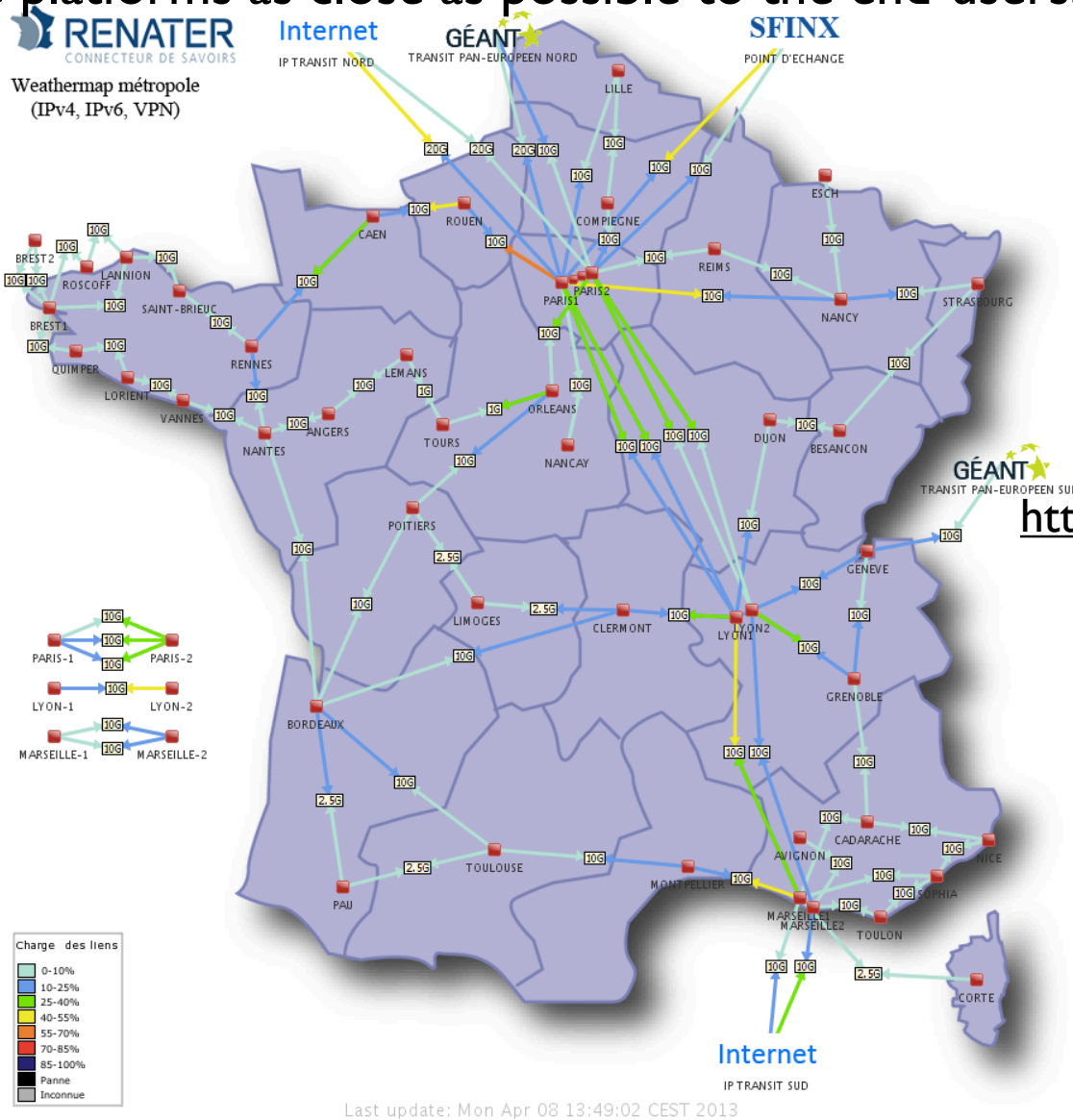
**Can we address these concerns "all in one" ? ?**

# Locality Based Utilty Computing Toward LUC Infrastructures

# Beyond the Cloud, the DISCOVERY Initiative

- Locality-based UC infrastructures

The only way to deliver highly efficient and sustainable UC services is to provide UC platforms as close as possible to the end-users.

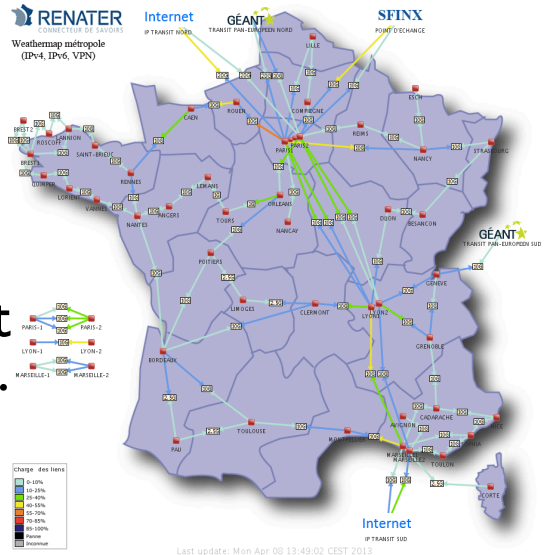



http://www.renater.fr/raccourci?lang=fr

# Beyond the Cloud, the DISCOVERY Initiative

- Locality-based UC infrastructures

  The only way to deliver highly efficient and sustainable UC services is to provide UC platforms as close as possible to the end-users.




- Leveraging network backbones

  Extend any point of presence of a network backbone with UC servers (from major network hubs up to DSLAMs that are operated by telecom companies/network institutions).

- Leveraging the data furnaces concept

  Deploy UC servers in medium and large institutions and use them as sources of heat inside public buildings such as hospitals or universities

- Combining both approaches ! ?

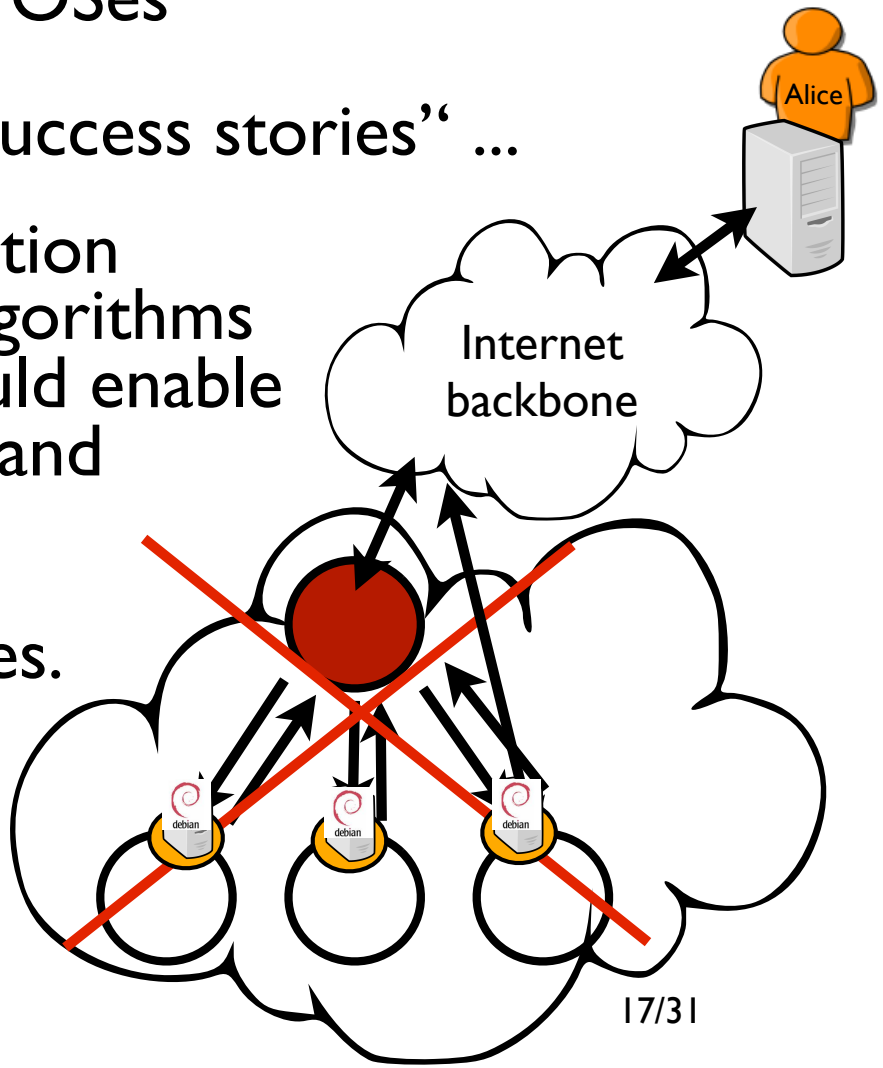
**⇒ Operating such widely distributed resources requires the definition of a fully distributed system**

# The DISCOVERY Proposal

- DIStributed and COoperative framework to manage Virtual EnviRonments autonomicallY
- Designing/implementing Distributed OSes

Deeply investigated with no "real success stories" ...

... But maturity of system virtualization capabilities as well as large scale algorithms and autonomous mechanisms should enable to design and implement a unified and autonomic system manipulating virtual environments (VEs) like traditional OS manipulate processes.

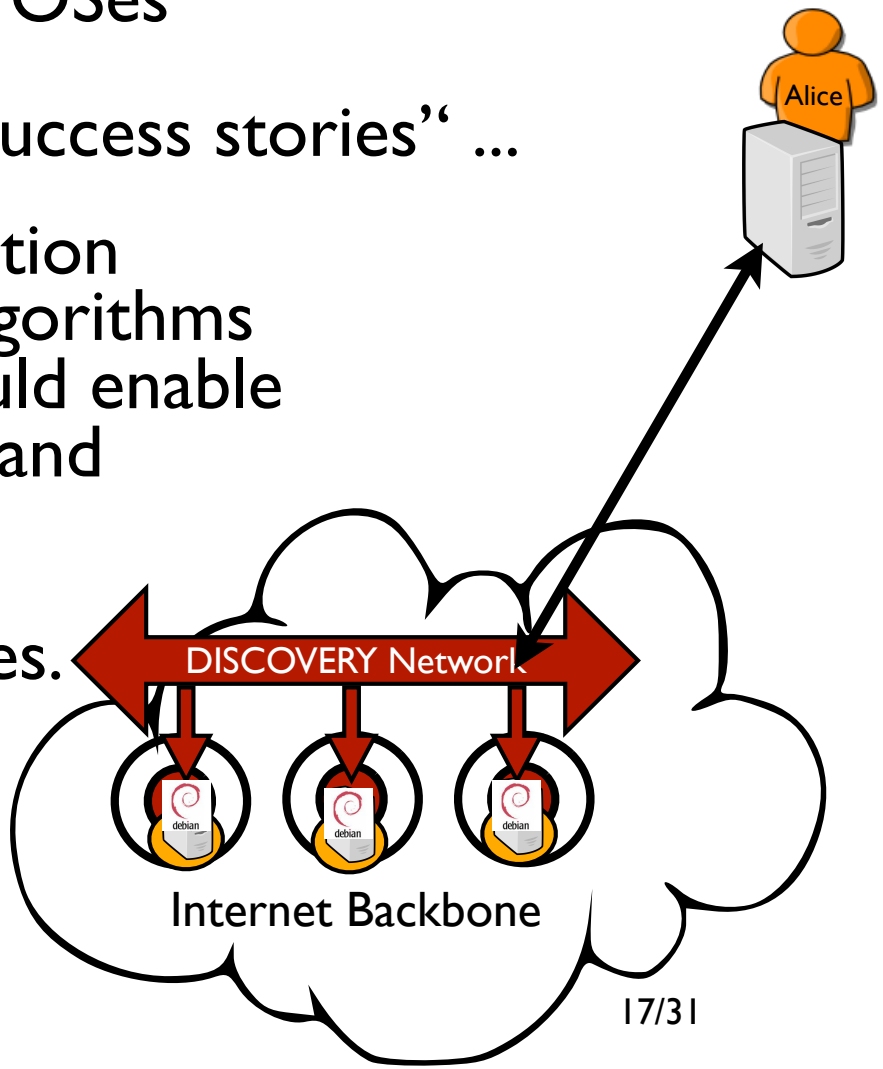


# The DISCOVERY Proposal

- DIStributed and COoperative framework to manage Virtual EnviRonments autonomicallY
- Designing/implementing Distributed OSes

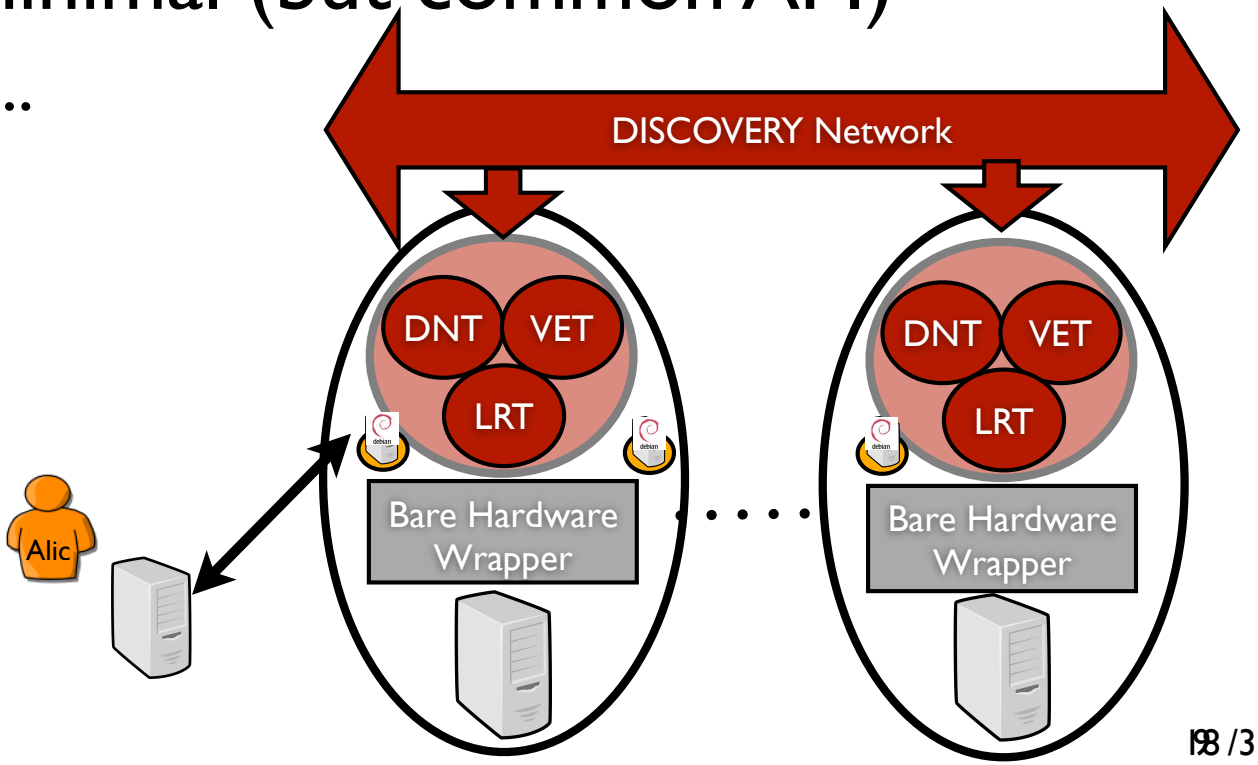
Deeply investigated with no "real success stories" ...

... But maturity of system virtualization capabilities as well as large scale algorithms and autonomous mechanisms should enable to design and implement a unified and autonomic system manipulating virtual environments (VEs) like traditional OS manipulate processes.



# The LUC OS Agent - Overview

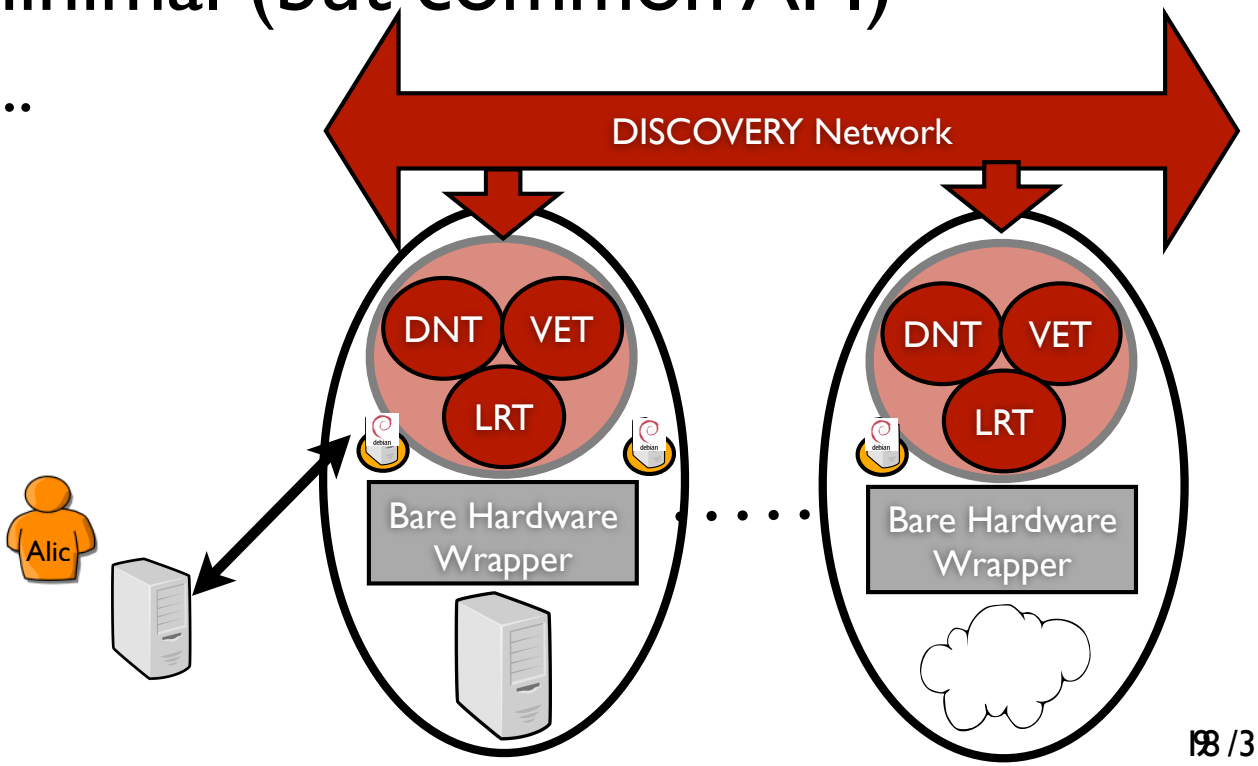
- 3 services
  Discovery Network Tracker (DNT)
  Virtual Environments Tracker (VET)
  Local Resources Tracker (LRT)
- Relying on a minimal (but common API)
  libvirt / OCCI / ...



# The LUC OS Agent - Overview

- 3 services
  Discovery Network Tracker (DNT)
  Virtual Environments Tracker (VET)
  Local Resources Tracker (LRT)

- Relying on a minimal (but common API)
  libvirt / OCCI / ...



# The DISCOVERY Initiative

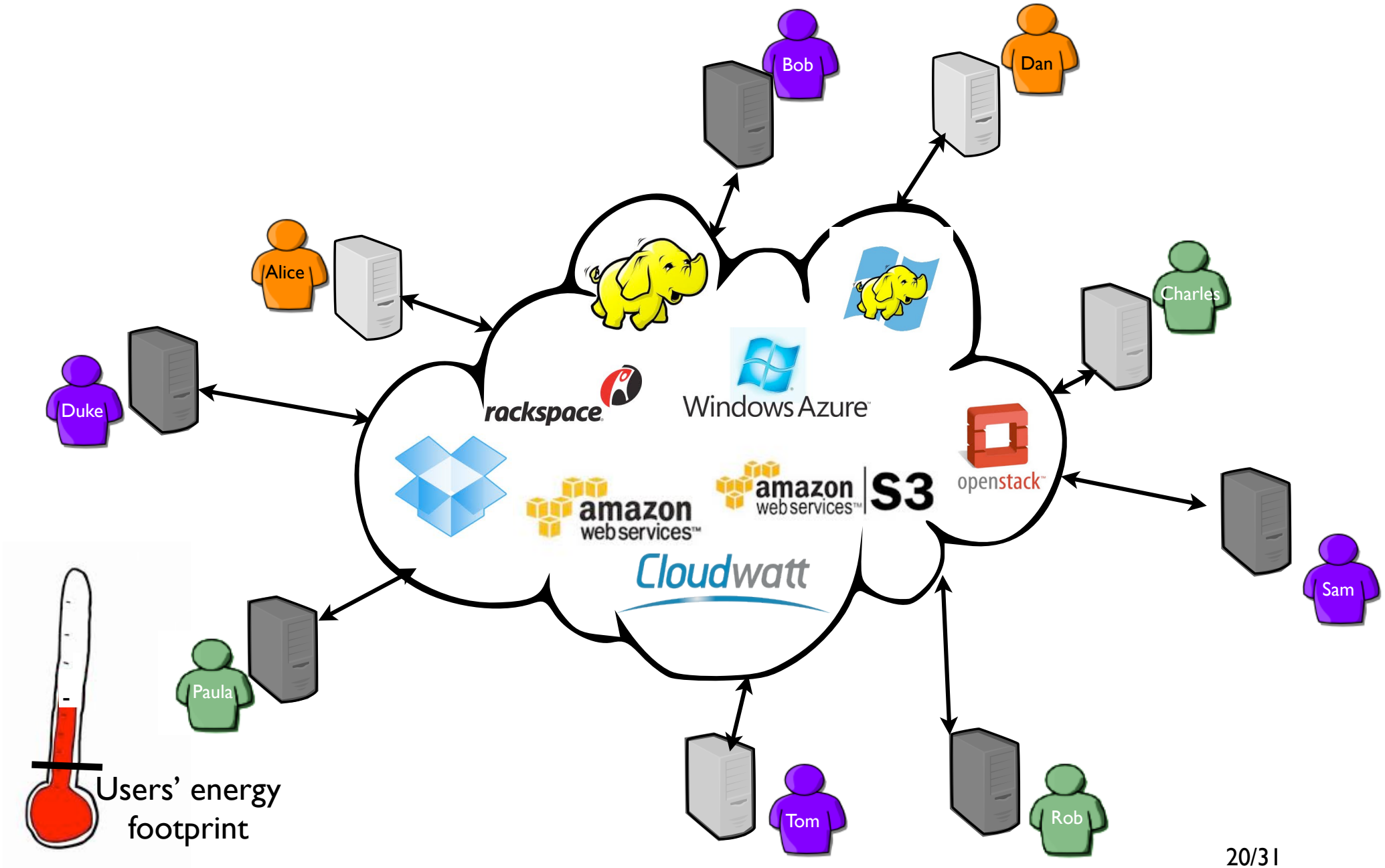
- Focusing on the design and the implementation of a complete OS for IaaS platforms based on VMs and VEs (group of VMs) as the fundamental granularity

  **Scalability**, targeting the management of hundred thousands of VMs upon thousands of physical machines (PMs)
  **Reliability**, considering "hardware failures as the norm rather the exception"
  **Reactivity**, handling each reconfiguration event as swiftly as possible to maintain VEs' QoS.

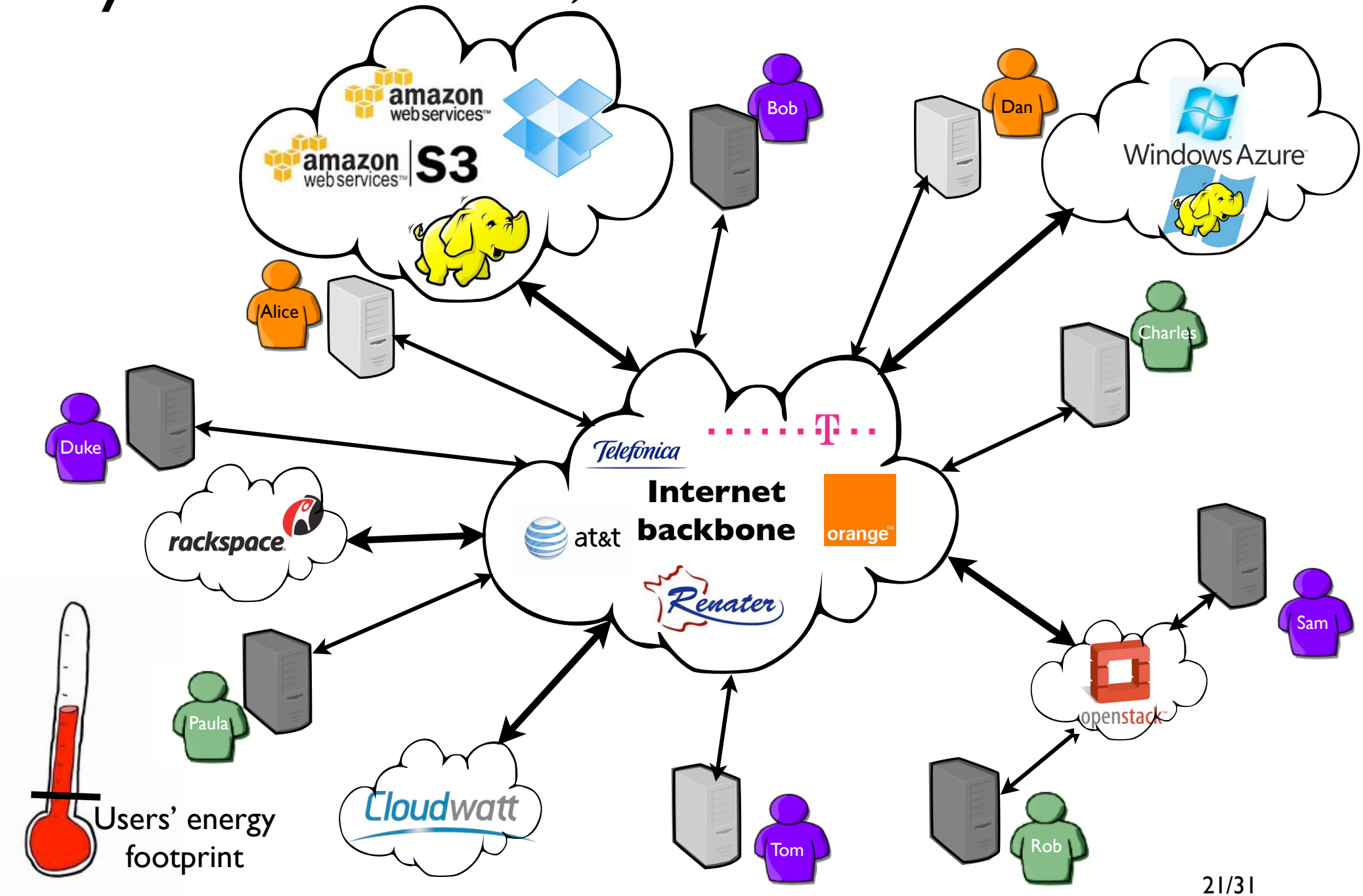
- May look simple but lots of scientific/technical challenges

  Cost of the DISCOVERY network !? partial view of the system !?
  Impact on the others VMs !?, management of VM images !?
  Which software abstractions to make the development easier and more reliable (distributed event programming) ?

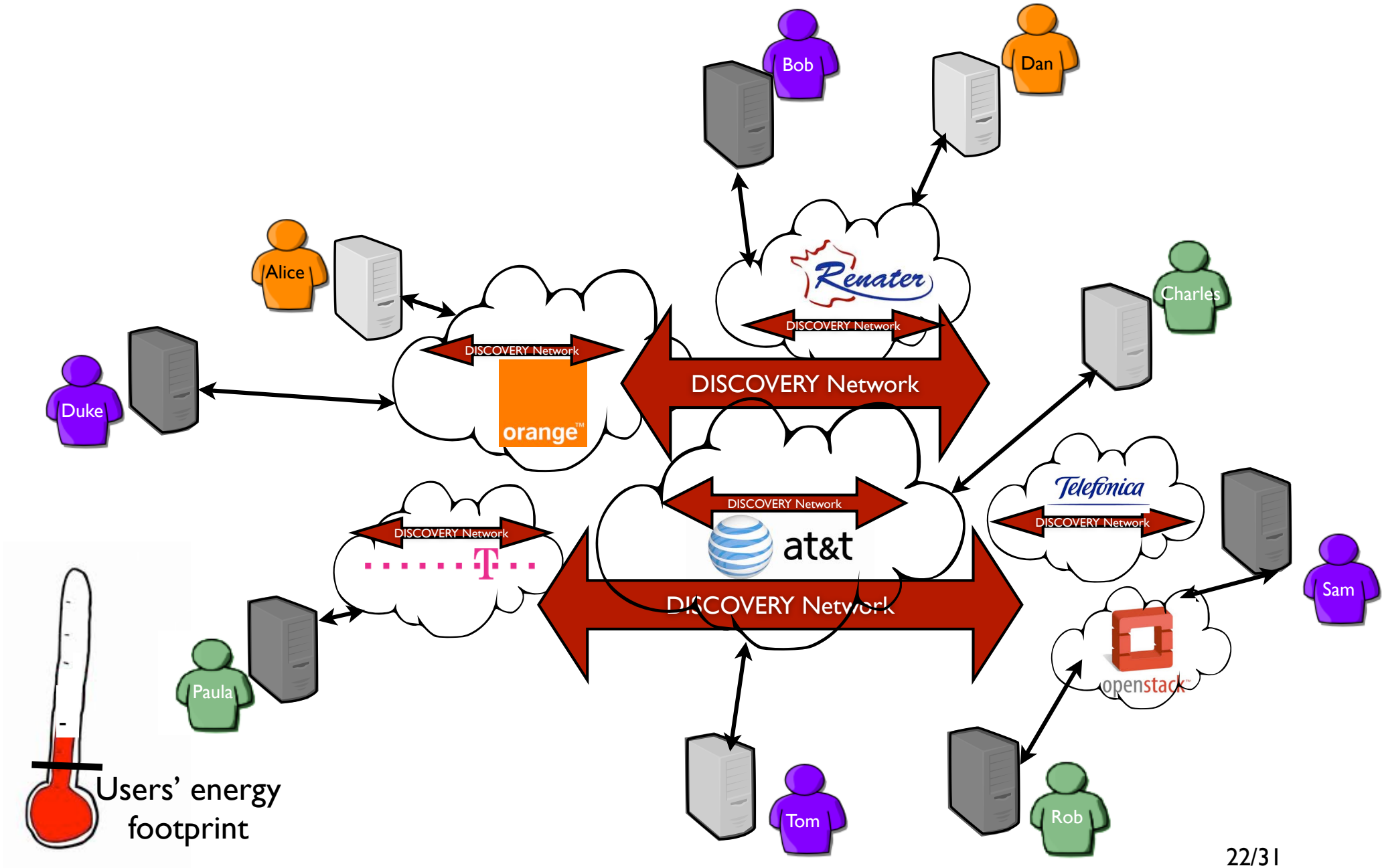
# Beyond the Cloud, the DISCOVERY Initiative



# Beyond the Cloud, the DISCOVERY Initiative



# Beyond the Cloud, the DISCOVERY Initiative



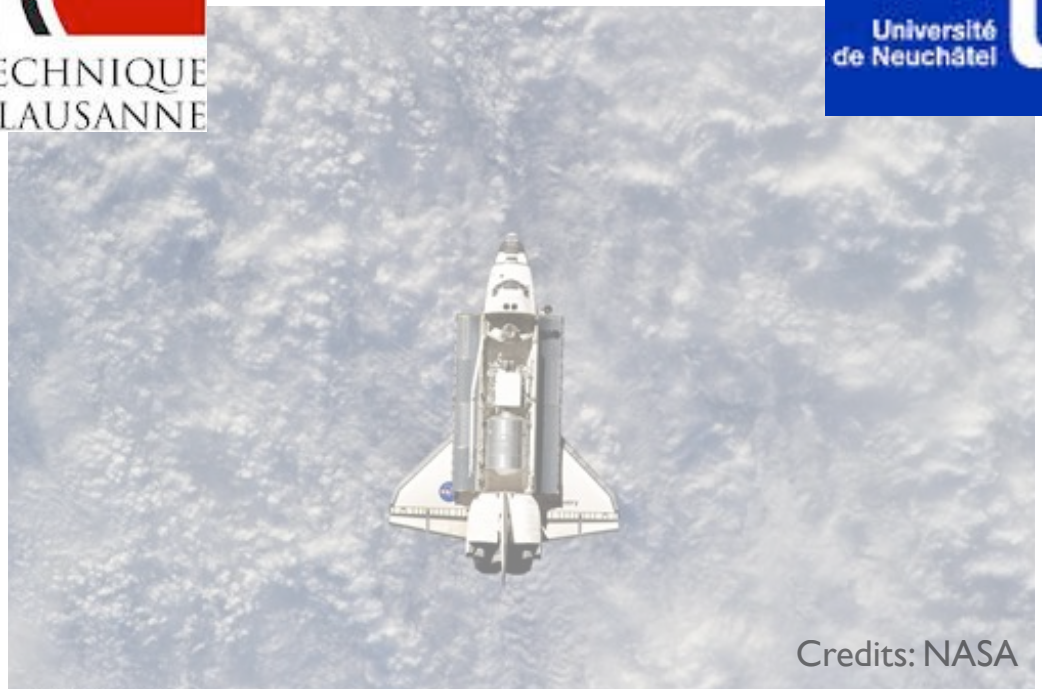
# The DISCOVERY Initiative

- Leveraging former projects but still on the starting blocks!
- Strong interests of large companies
  (SAP, Orange Lab, Citrix, ...)
- RENATER
- An important actor to follow: Akamai
- Preliminary works with promising results
  (especially on the LRT: a first POC)
- Long term objective: impact on the design of distributed applications in order to take advantage of the locality
  (building S3 like system)

# The DISCOVERY Initiative

- Thank you / Questions ?
- Want more ?
  - Focus on LRT
    (Flavien Quesnel's Phd, ended in Feb 2013)
  - See Discovery internals in a nutshell

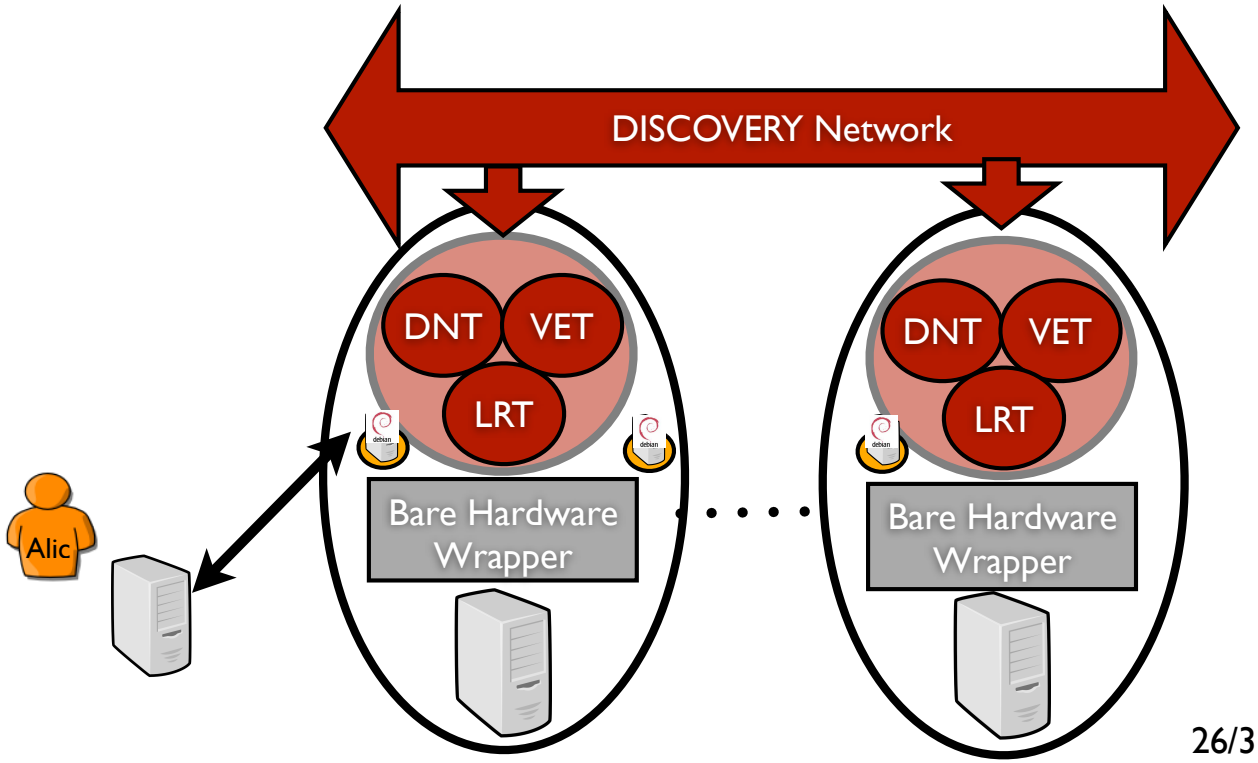
# Beyond the Clouds, the DISCOVERY Initiative




Adrien Lèbre / Ascola Research Group

# The LUC OS Agent - Overview

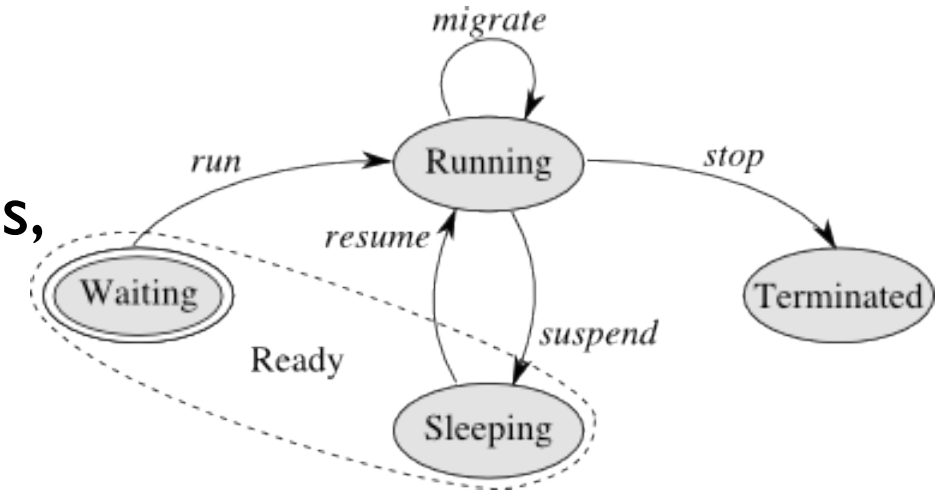
- FOCUS on the LRT and a part of the VET
  Dynamic scheduling of VMs



# Background - a VE-based OS

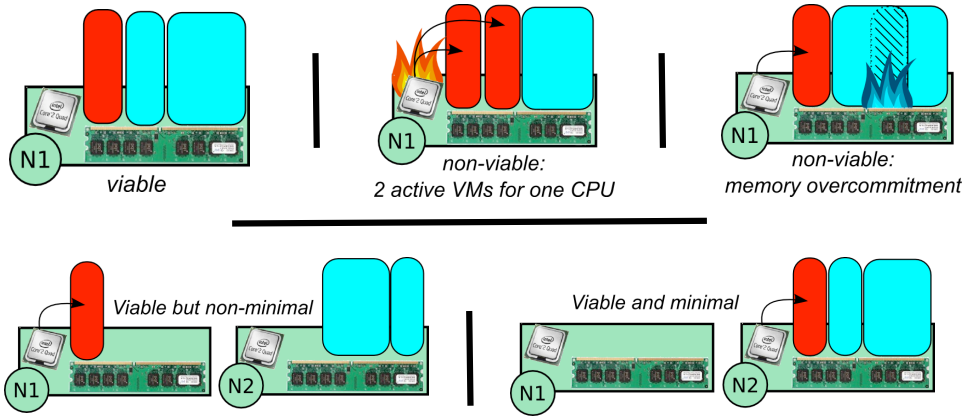
- General idea: manipulate VEs instead of processes
  (a VE is a users' working environment, possibly composed of several interconnected VMs)

- In a similar way of usual processes, each VE is in a particular state:

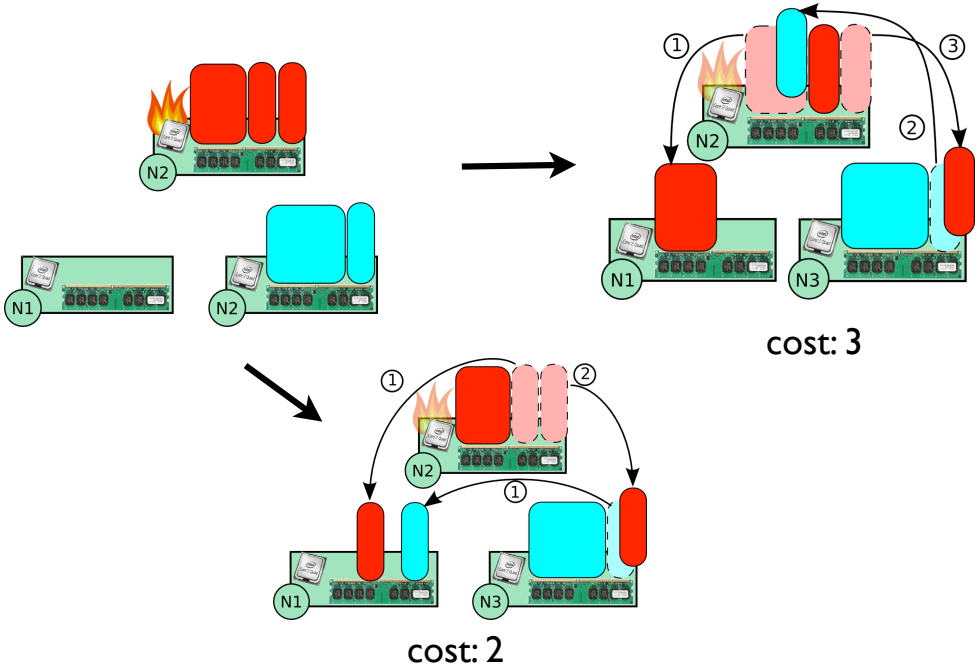


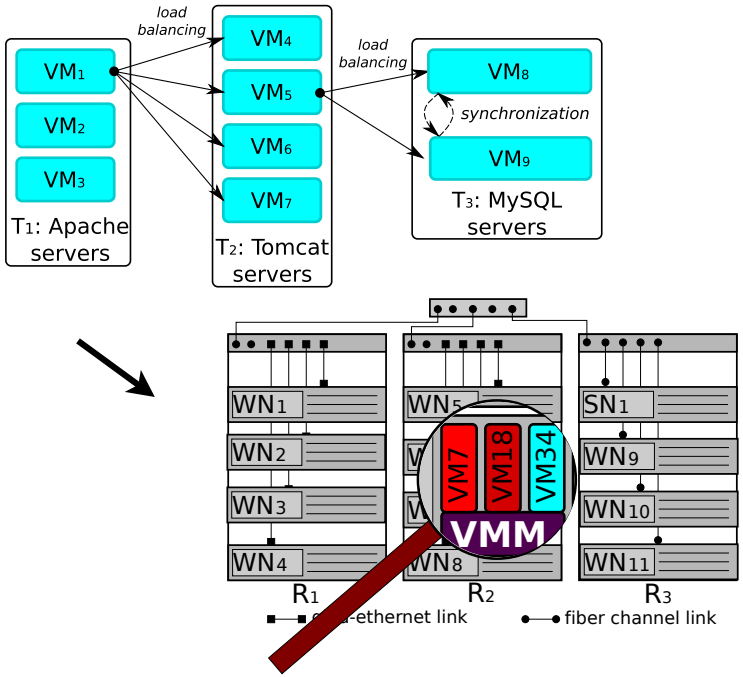

- Perform VE context switches (a set of VM context switches) to rebalance the LUC infrastructure according to the:
  scheduler objectives / available resources / waiting queue / ...

# Background - the Entropy Proposal



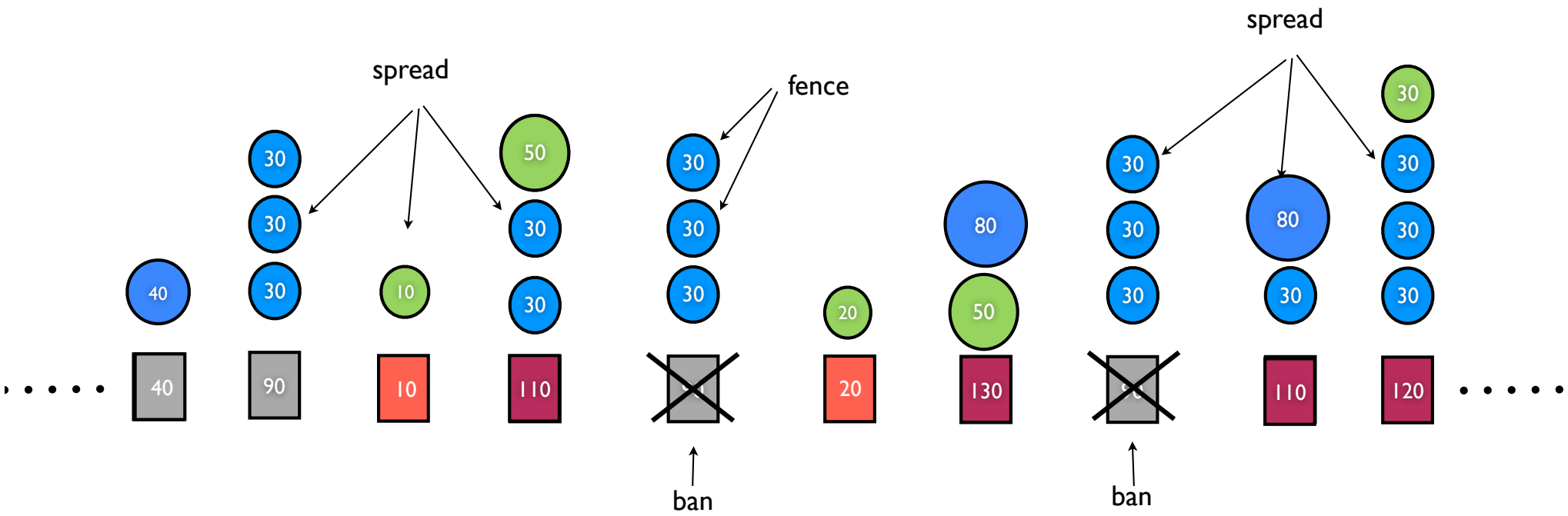

- Find the "right" mapping between needs of VMs, their constraints and resources provided by PMs



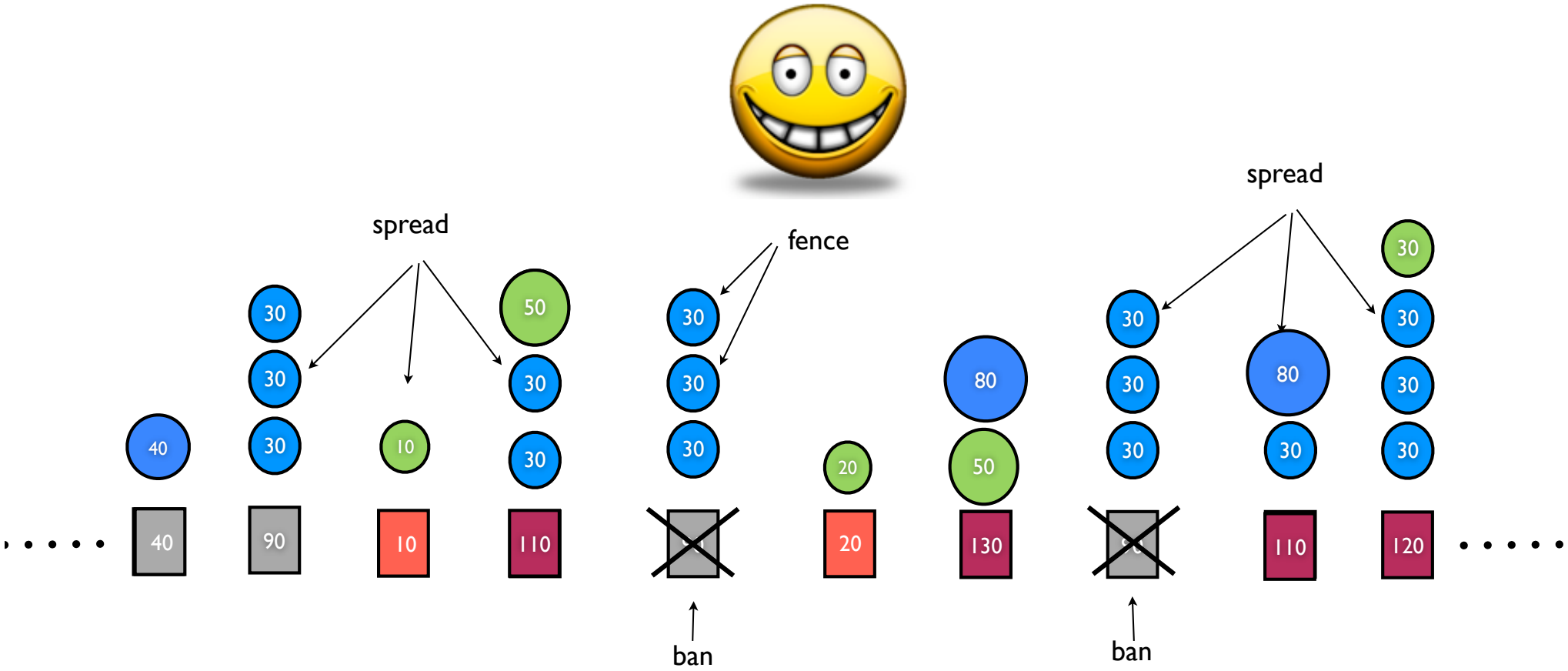

credits: F. Hermenier

# Background - a Small Example



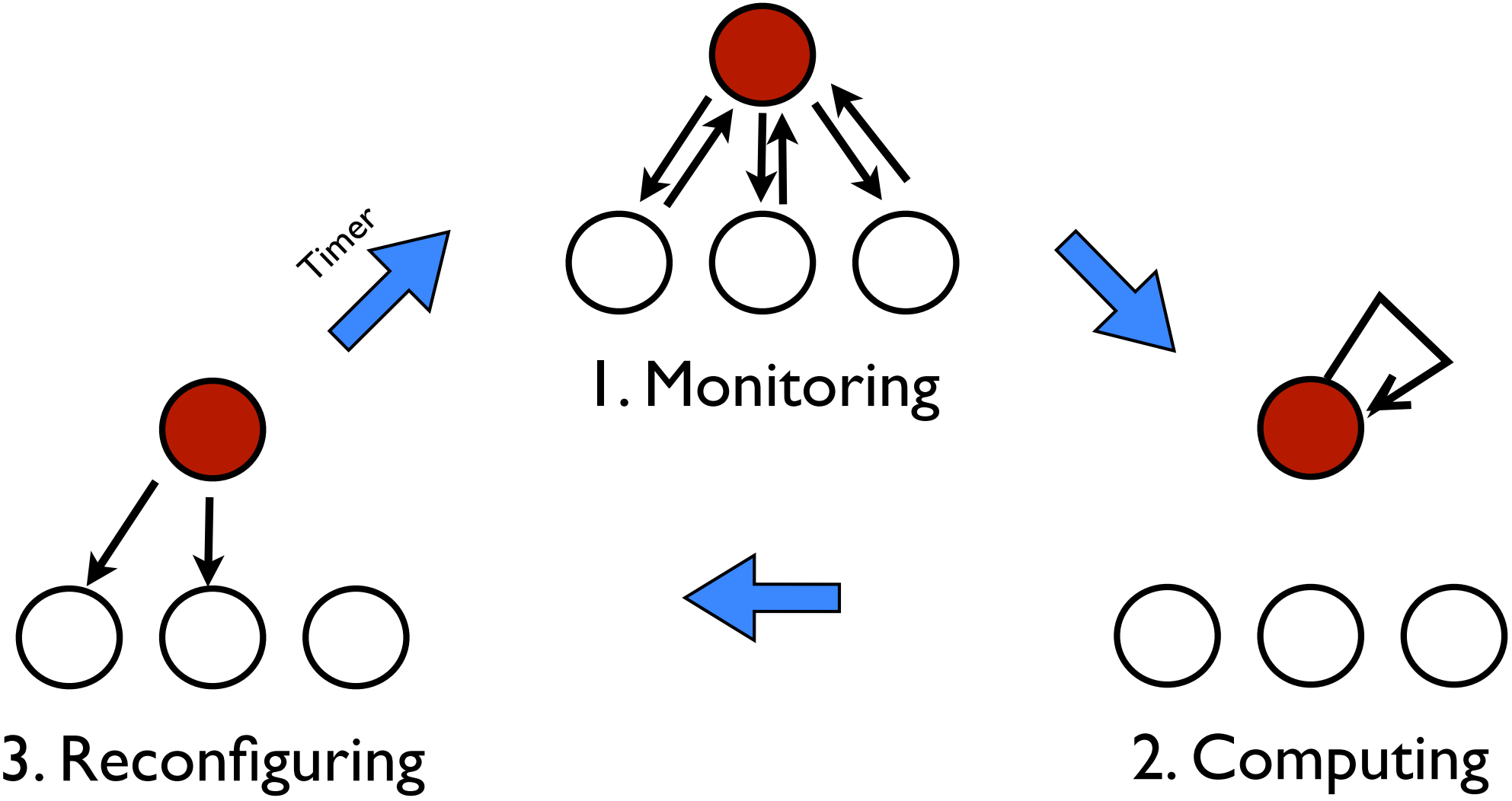
# Background - a Small Example

## Only CPU is considered in this simple example



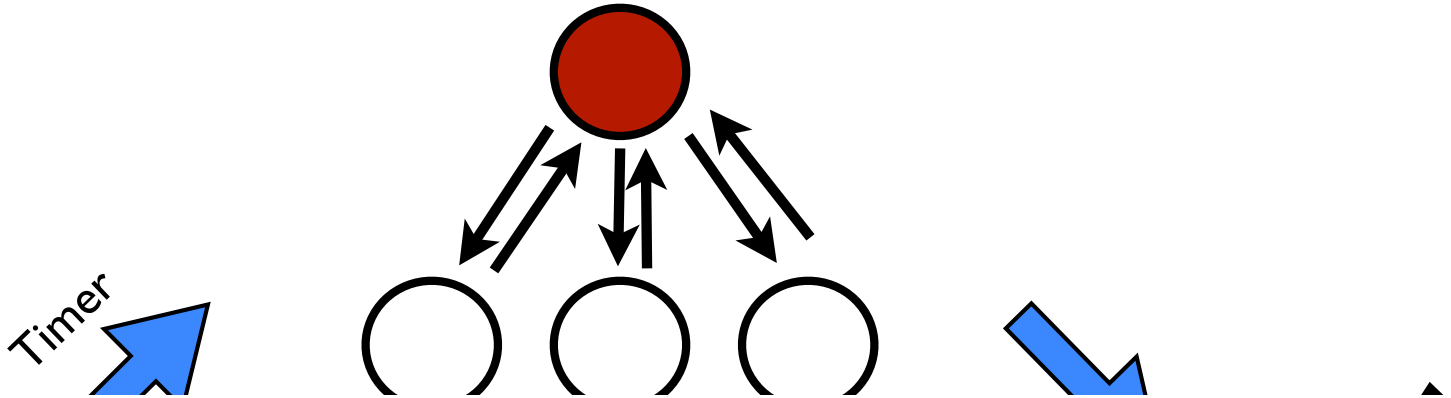
# Dynamic Scheduling of VMs

## Centralized approach

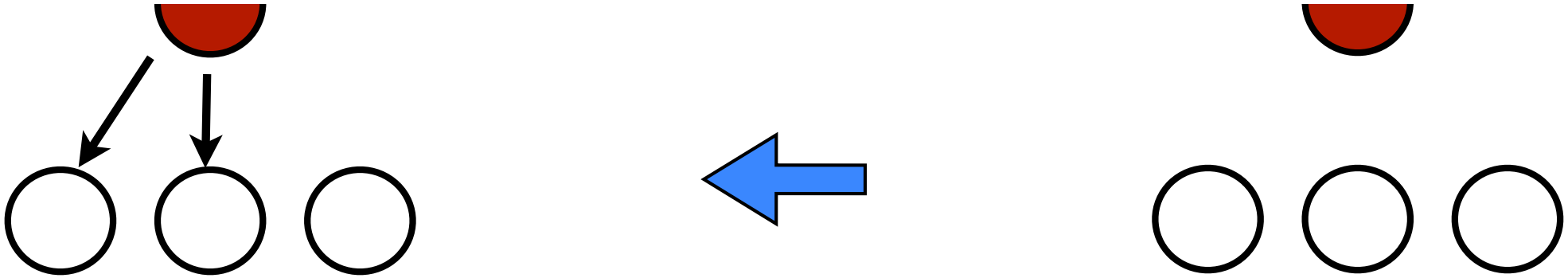



# Dynamic Scheduling of VMs

## Centralized approach


⇒ Not suited for a LUC platform



3. Reconfiguring

2. Computing

# The LUC OS - VEs Scheduling

- Make cooperation between hypervisors
- Main characteristics

  Event driven

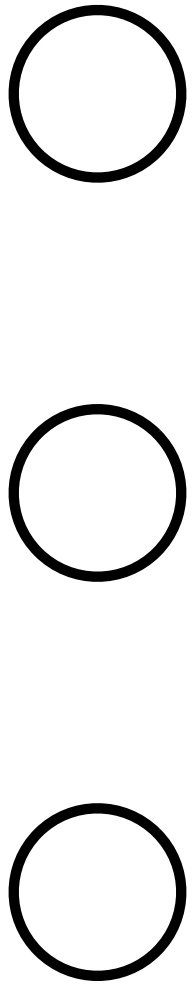
  Peer to peer, no service node

  Local interactions between nodes

  Scheduling performed on partitions of the system, created dynamically (nodes are reserved for exclusive use by a scheduler, to prevent several schedulers from migrating the same VMs)

# The LUC OS - VEs Scheduling

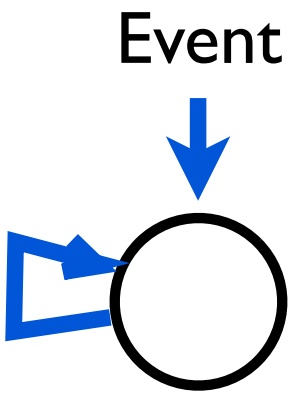
Event occurs on $node_i$

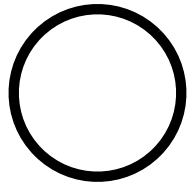


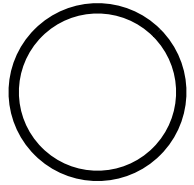
# The LUC OS - VEs Scheduling

Event occurs on $node_i$

Can current node scheduler calculate valid schedule?





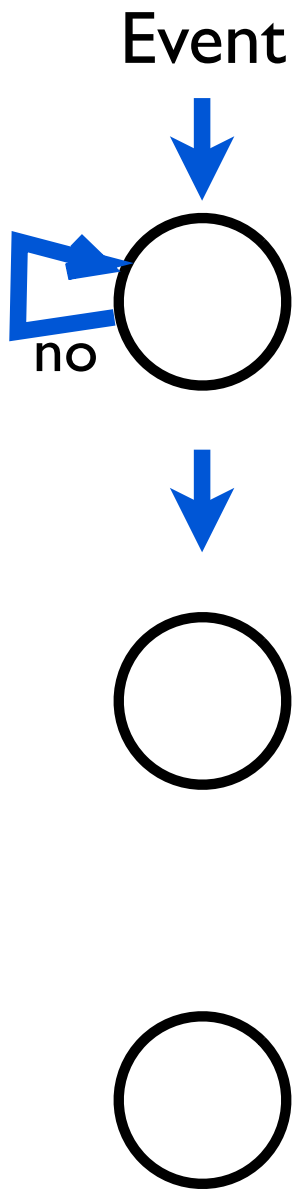
# The LUC OS - VEs Scheduling

Event occurs on node$_i$

Can current node scheduler calculate valid schedule?

no

Contact neighbor and ask it to solve the problem



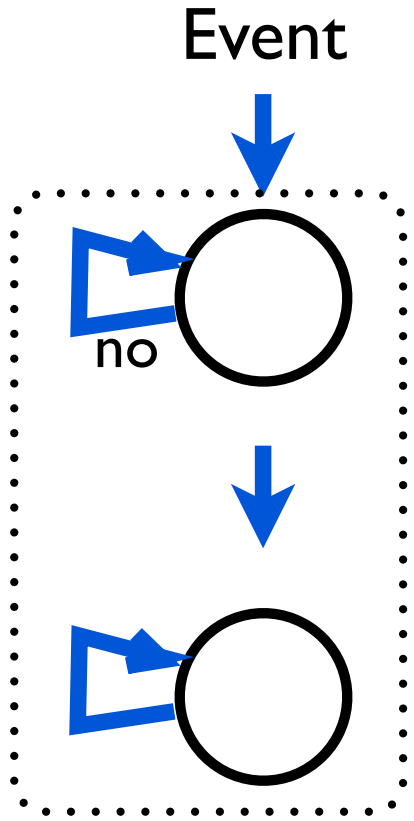
# The LUC OS - VEs Scheduling

Event occurs on node$_i$

Can current node scheduler calculate valid schedule?

no

Contact neighbor and ask it to solve the problem



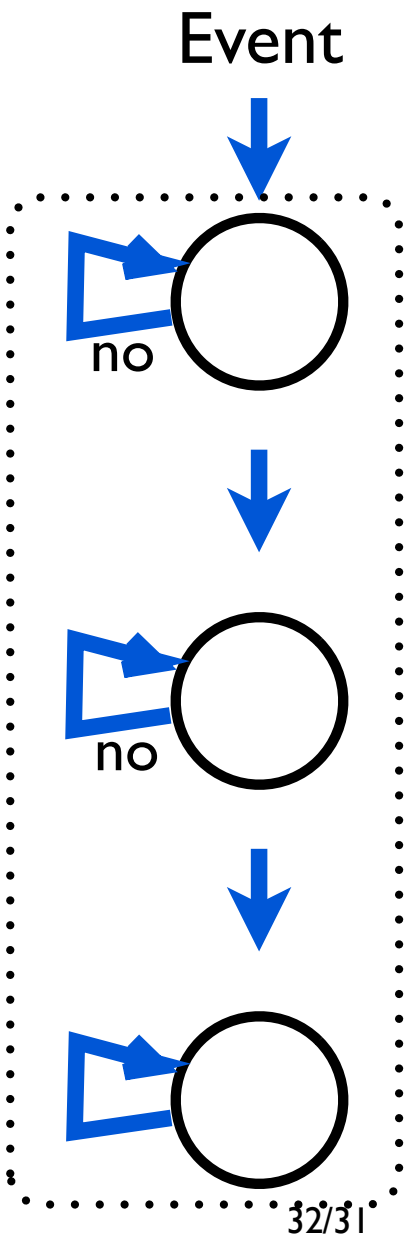
# The LUC OS - VEs Scheduling

Event occurs on node$_i$

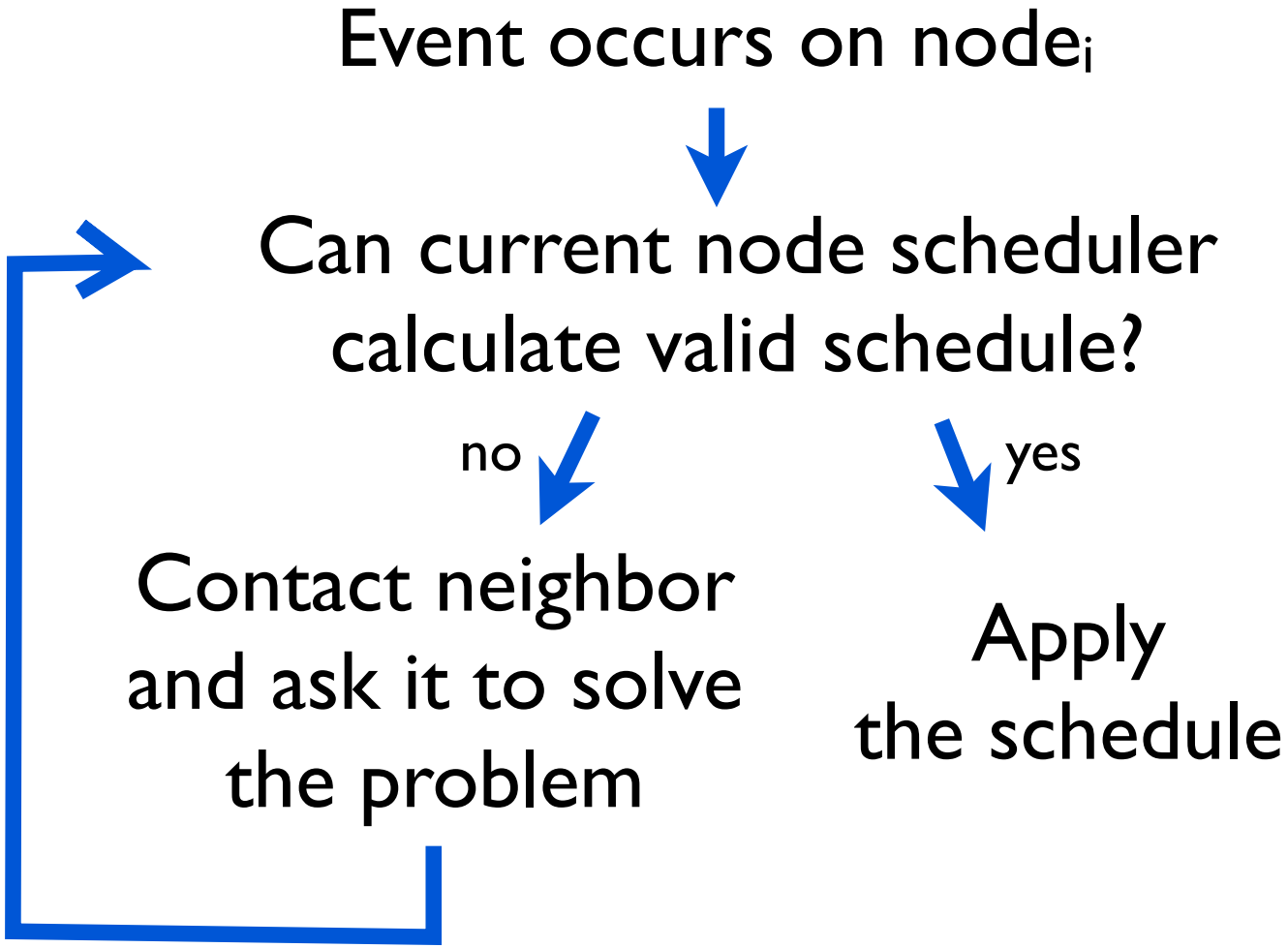
Can current node scheduler calculate valid schedule?

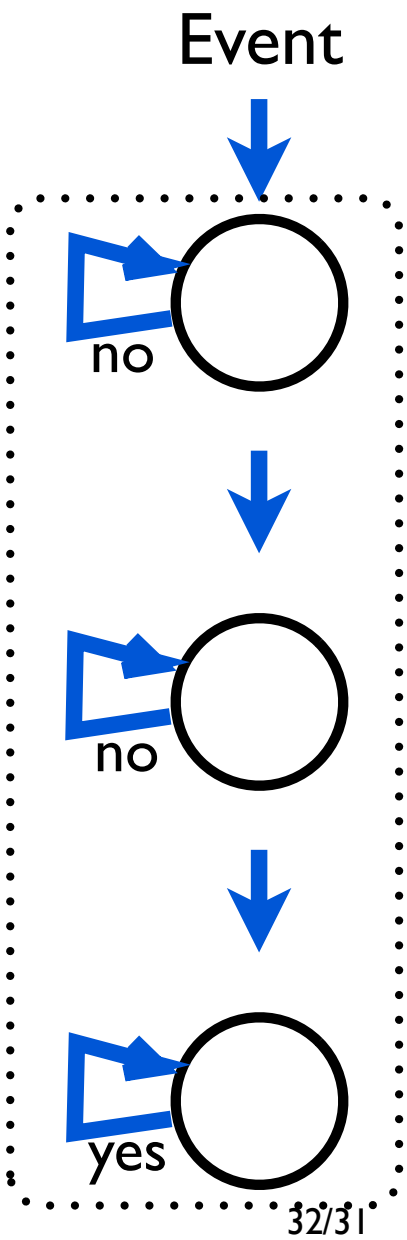
no

Contact neighbor and ask it to solve the problem

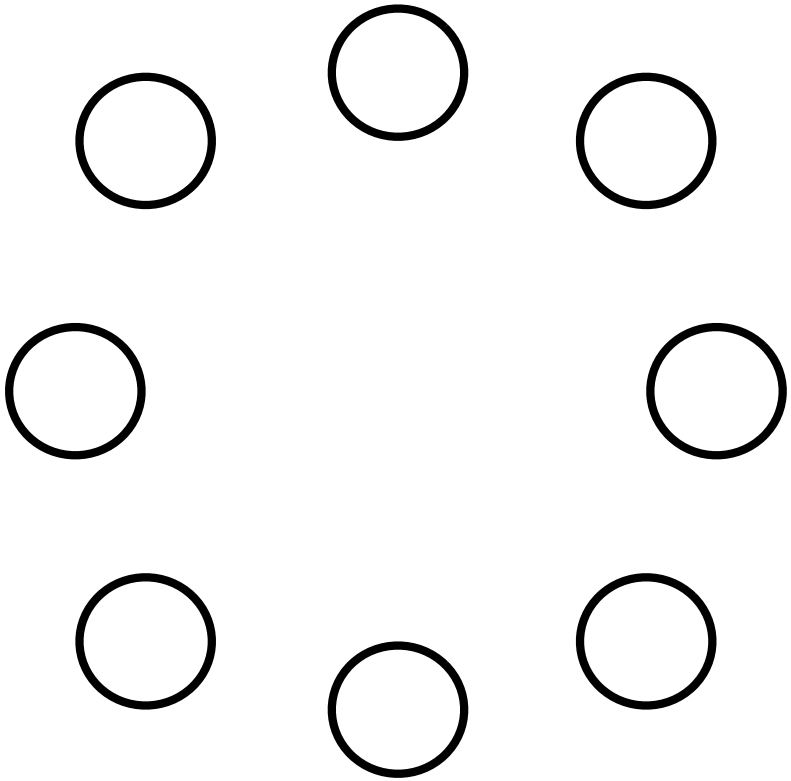


# The LUC OS - VEs Scheduling

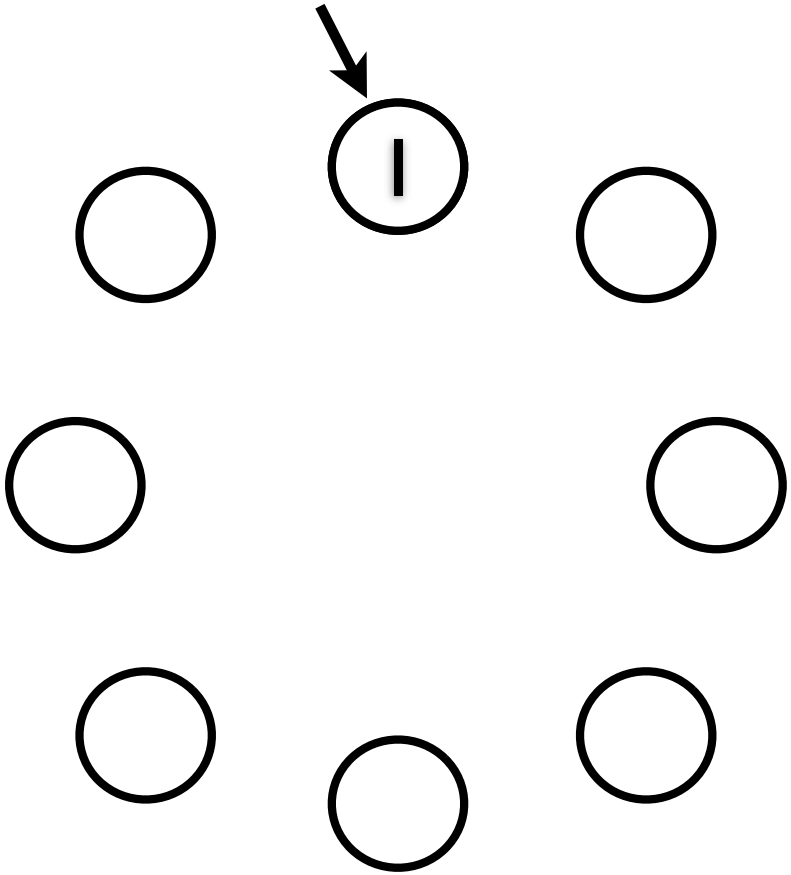


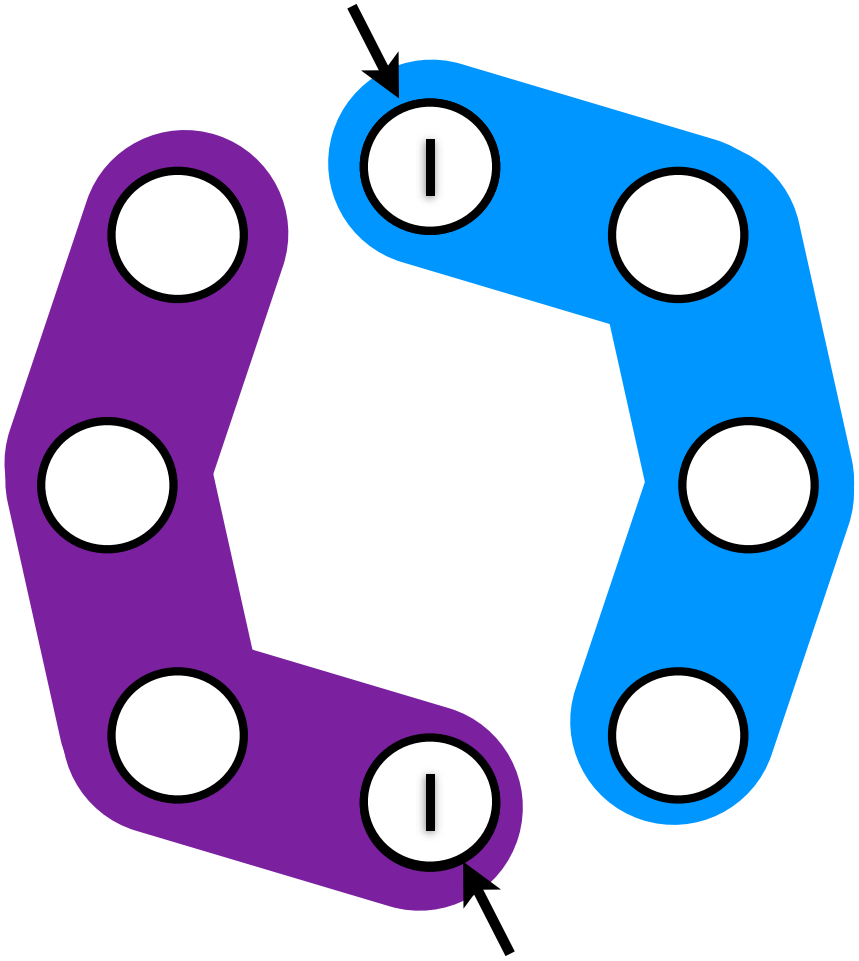
# The LUC OS - VEs Scheduling



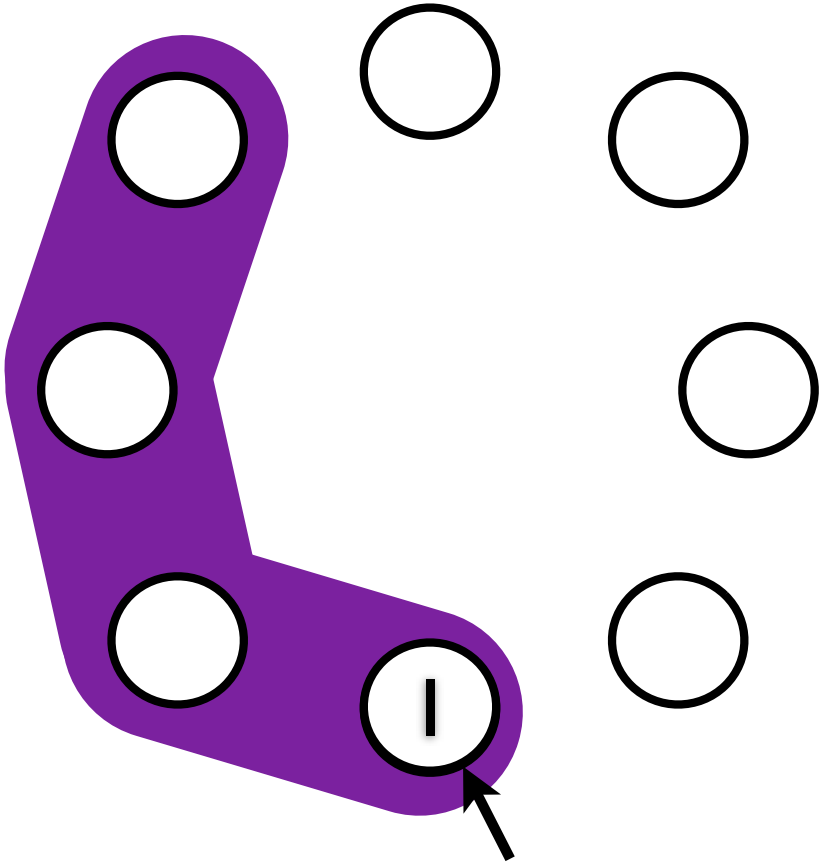
# The LUC OS - VEs Scheduling



# The LUC OS - VEs Scheduling



# The LUC OS - VEs Scheduling



# The LUC OS - VEs Scheduling

- Published in [CCPE'12]
- Reactivity/scalability

  Scheduling started when an event is generated

  Few nodes considered for scheduling

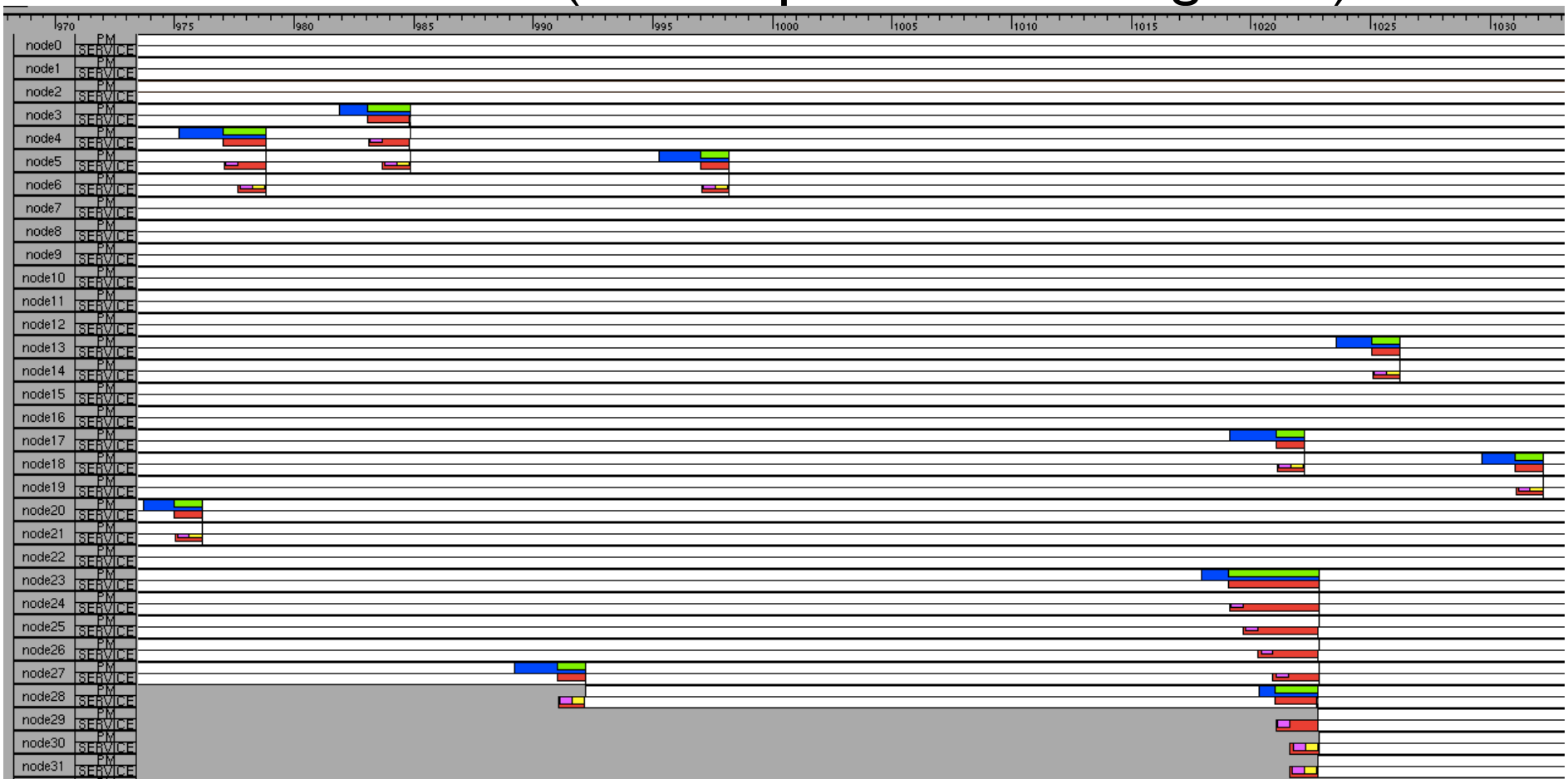
  ⇒ much faster computation

- Parallelism

  Several events can be processed simultaneously and independently

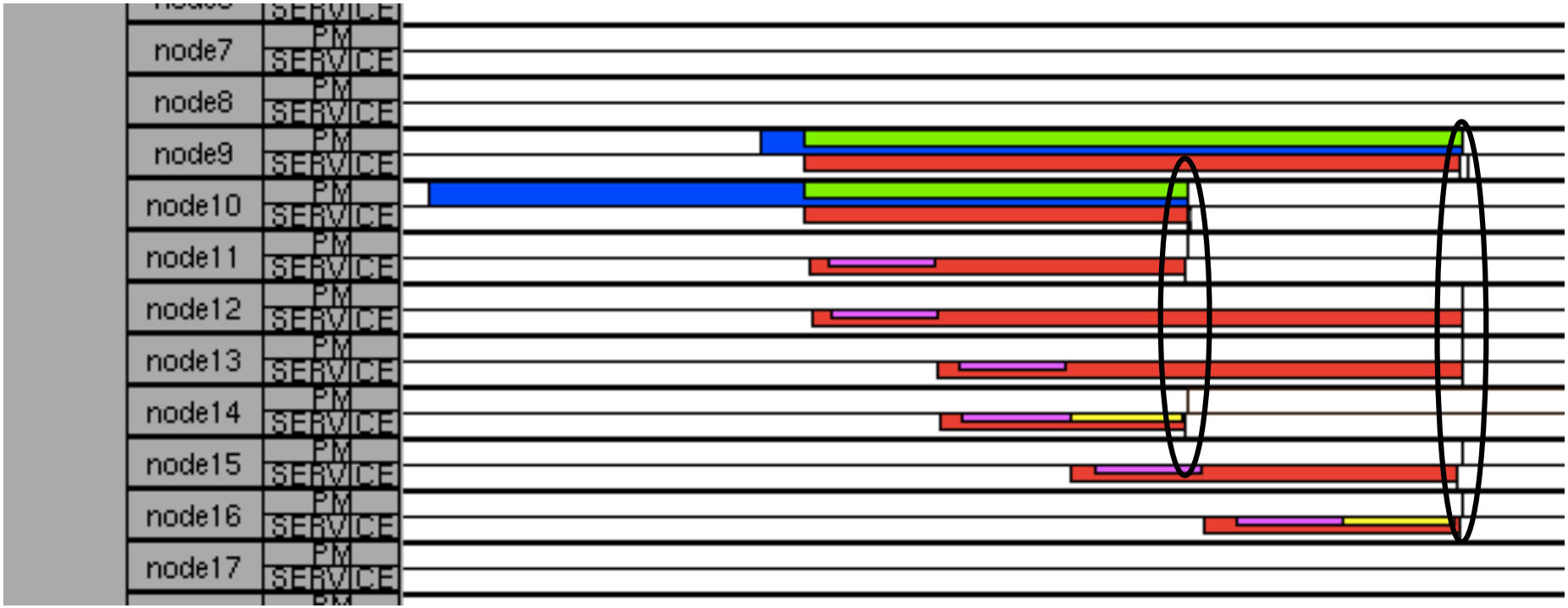
# The LUC OS - VEs Scheduling

- POC (leveraging Entropy to solve non viable configuration)
- 100K VMs / 10K PMs (simulation using the SimGrid framework)
- 10K VMs / 512 PMs ("real experiments" using G5K)



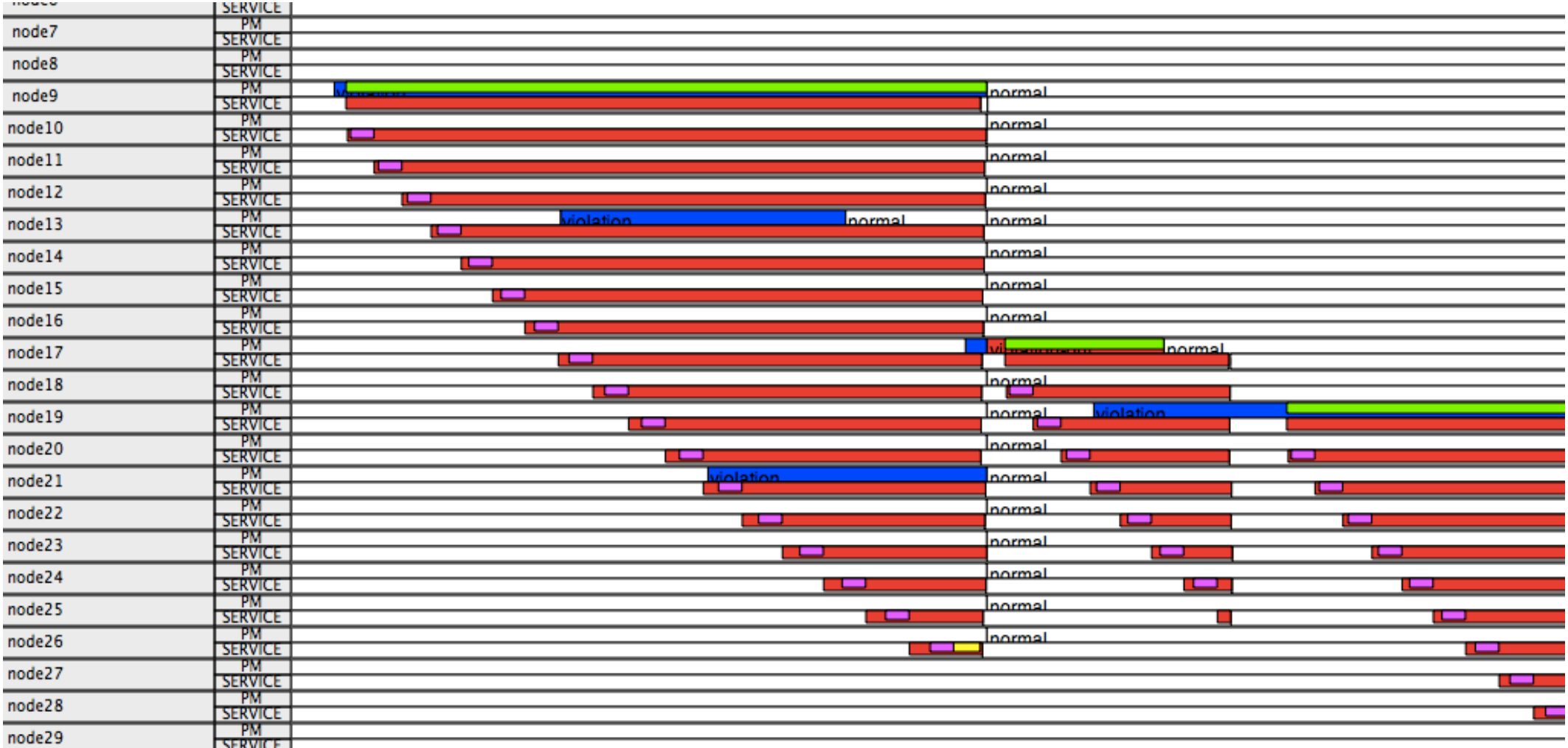
# The LUC OS - VEs Scheduling

- POC (leveraging Entropy to solve non viable configuration)
- 100K VMs / 10K PMs (simulation using the SimGrid framework)
- 10K VMs / 512 PMs ("real experiments" using G5K)



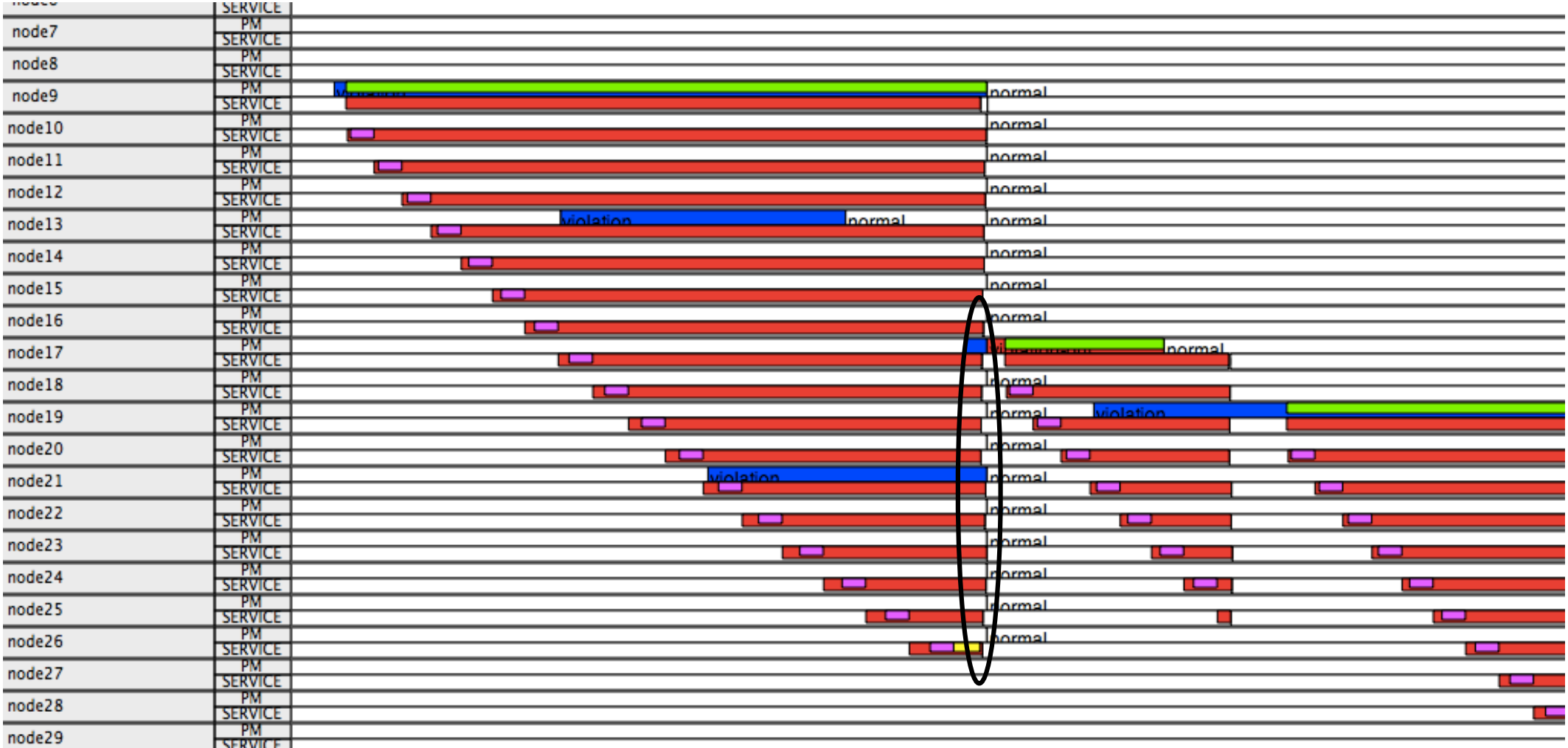
# The LUC OS - VEs Scheduling

- POC (leveraging Entropy to solve non viable configuration)
- 100K VMs / 10K PMs (simulation using the SimGrid framework)



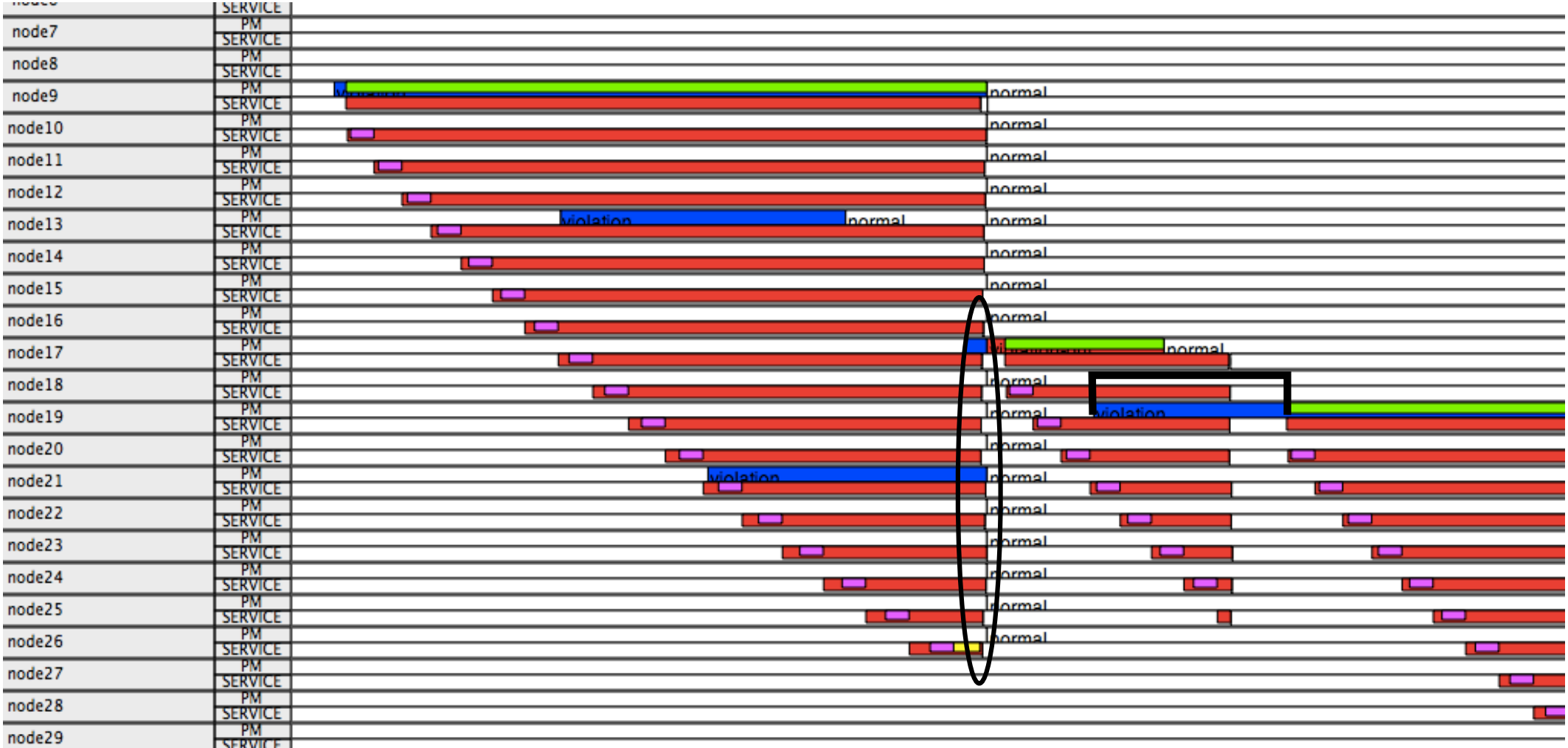
# The LUC OS - VEs Scheduling

- POC (leveraging Entropy to solve non viable configuration)
- 100K VMs / 10K PMs (simulation using the SimGrid framework)

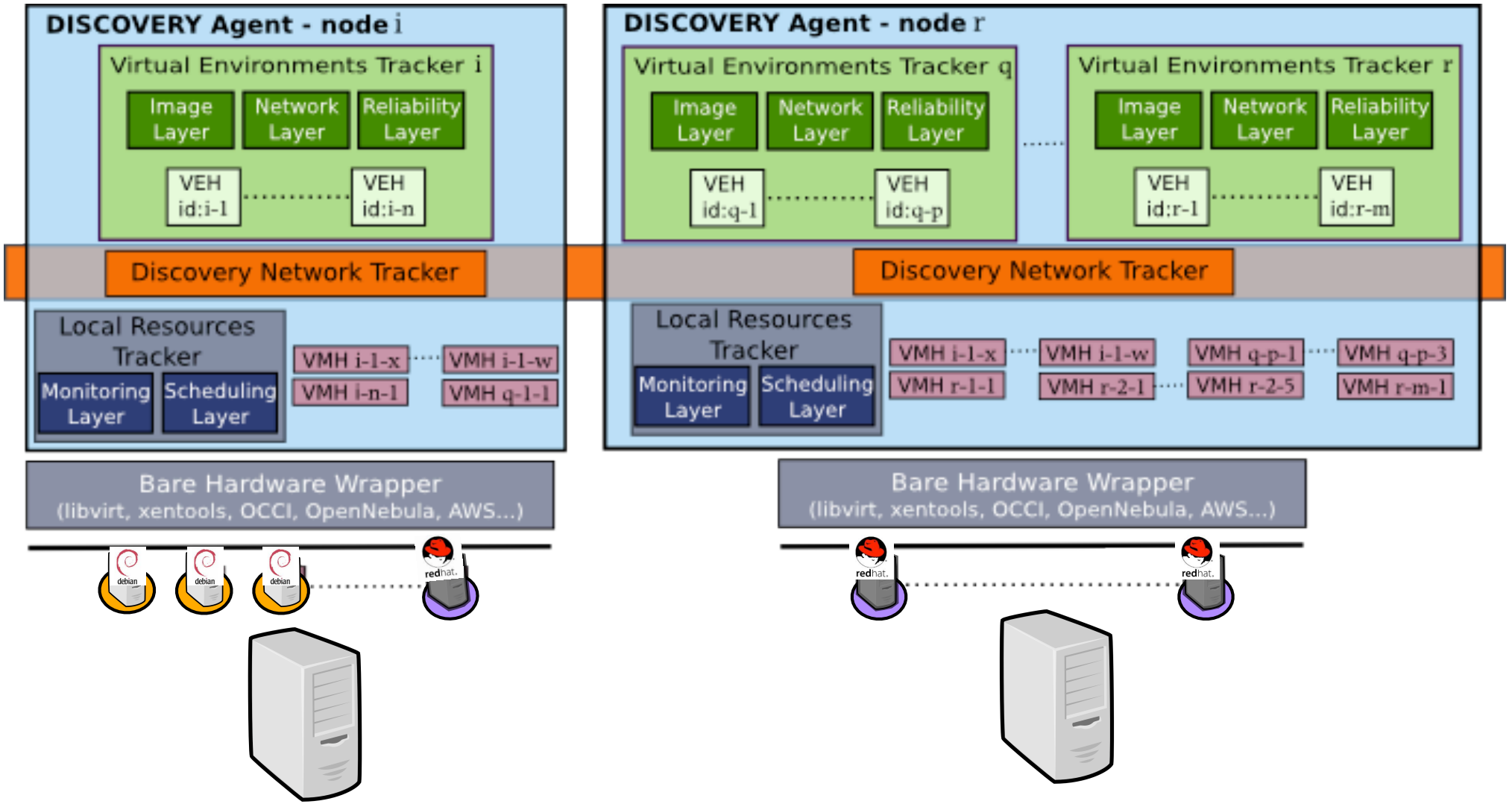


# The LUC OS - VEs Scheduling

- POC (leveraging Entropy to solve non viable configuration)
- 100K VMs / 10K PMs (simulation using the SimGrid framework)

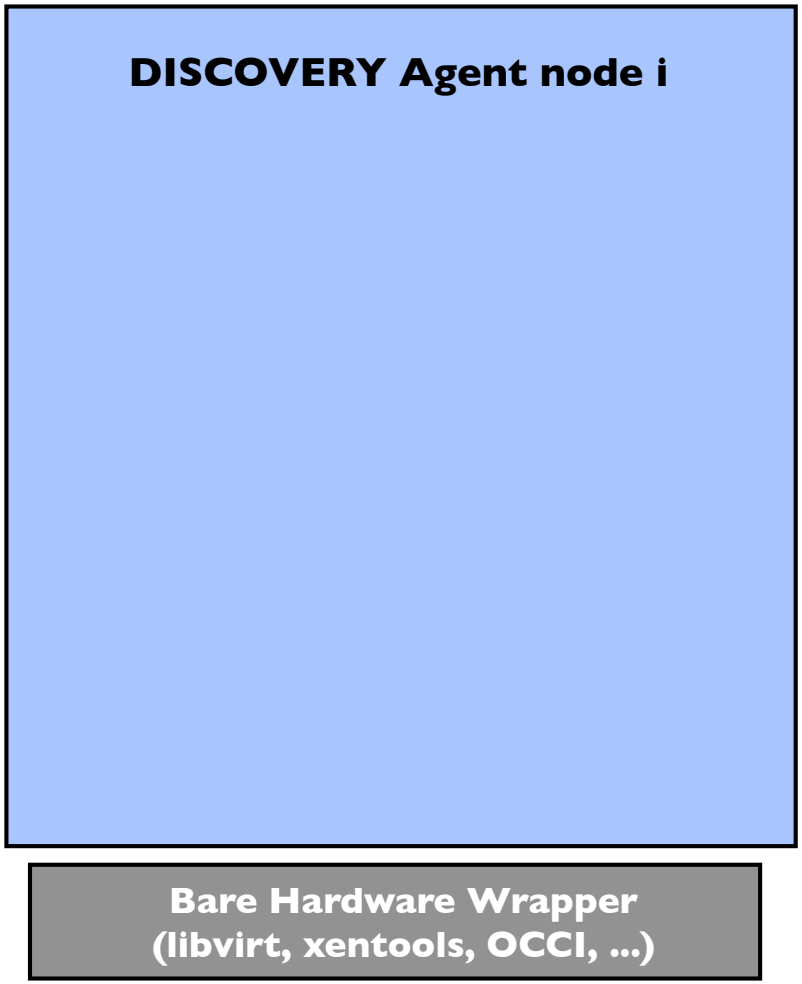


# Understanding the DISCOVERY Agent

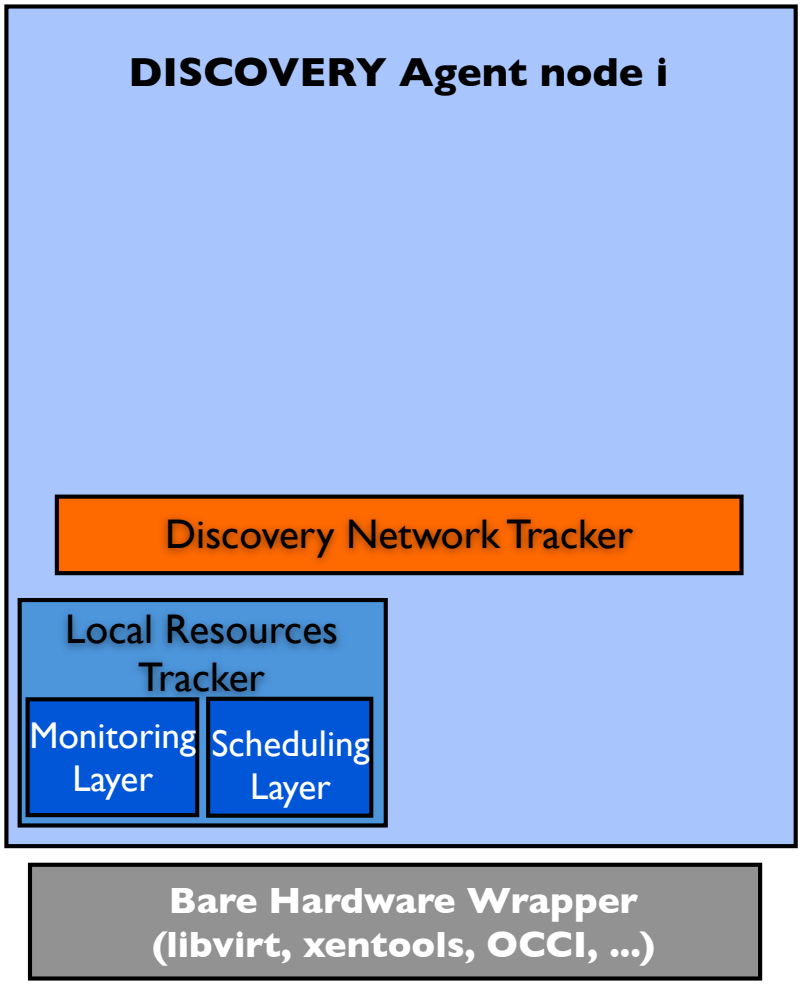


# DISCOVERY - Basic Usage

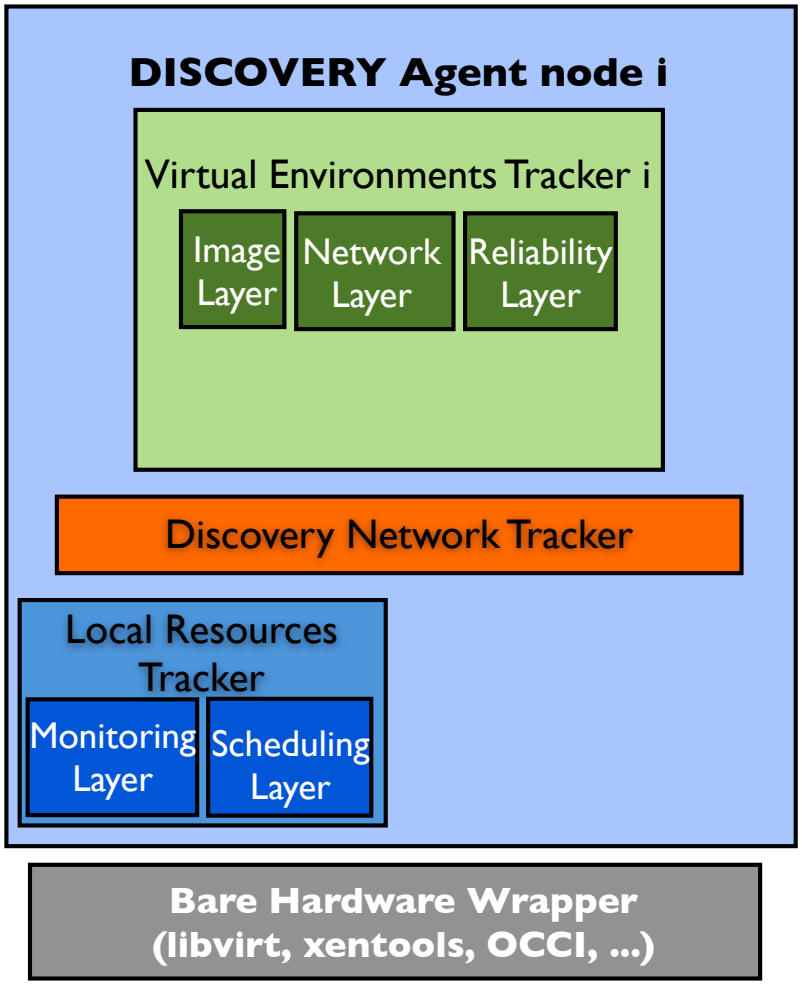
# DISCOVERY - Basic Usage



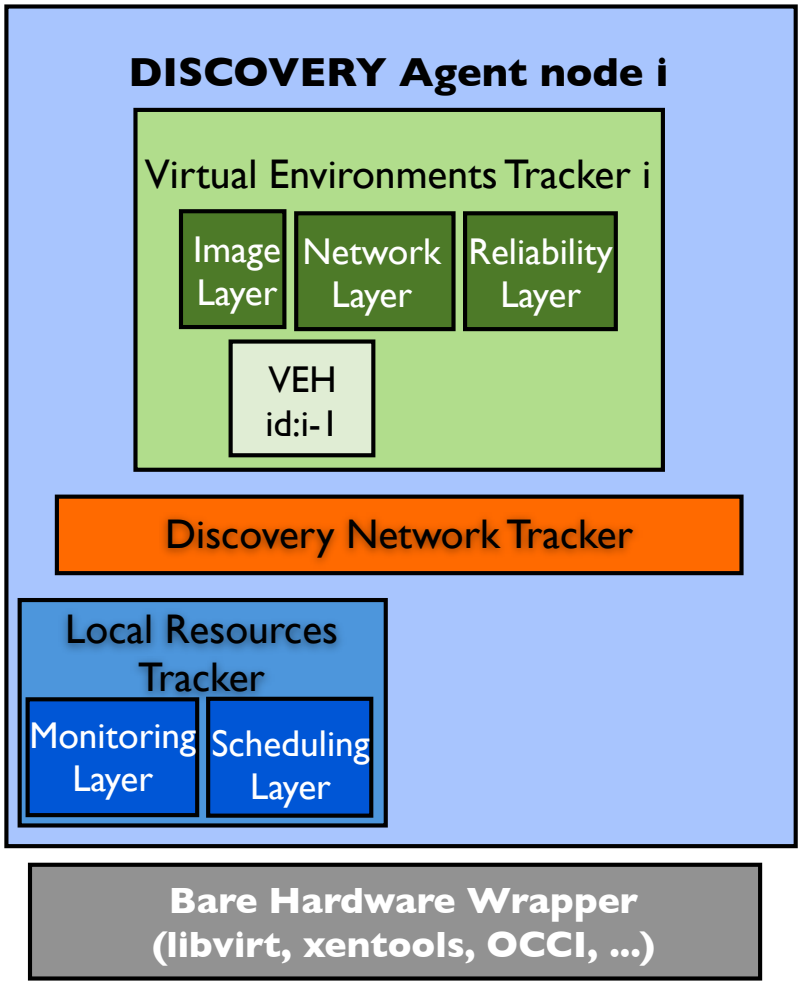
# DISCOVERY - Basic Usage



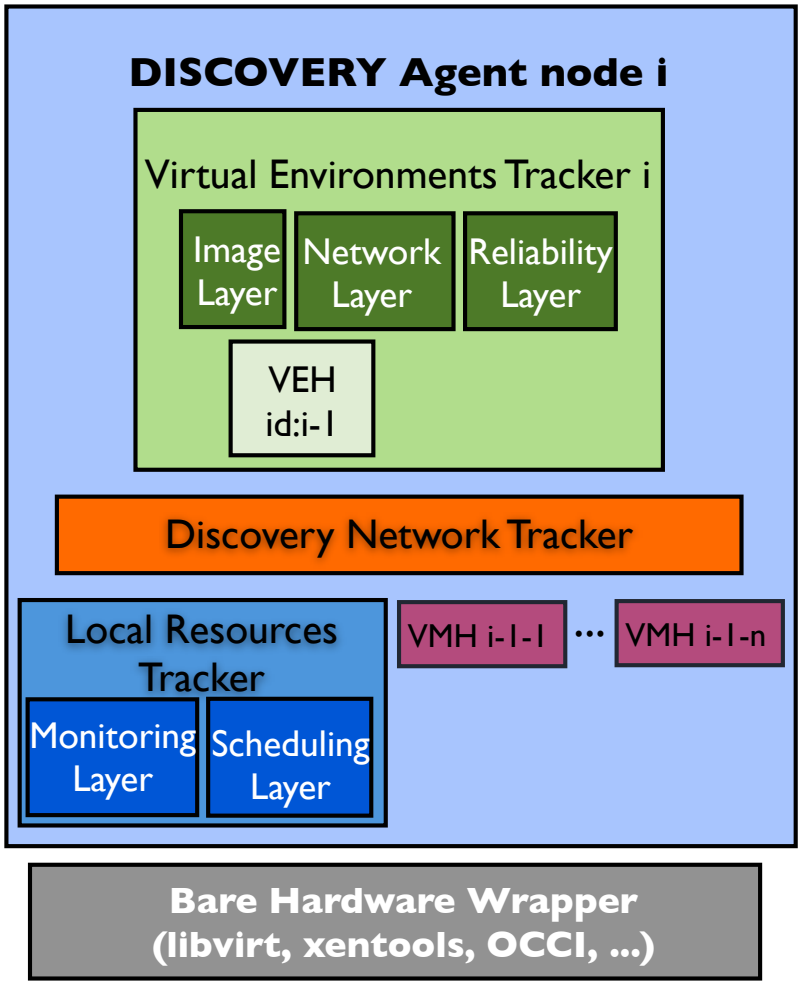

# DISCOVERY - Basic Usage



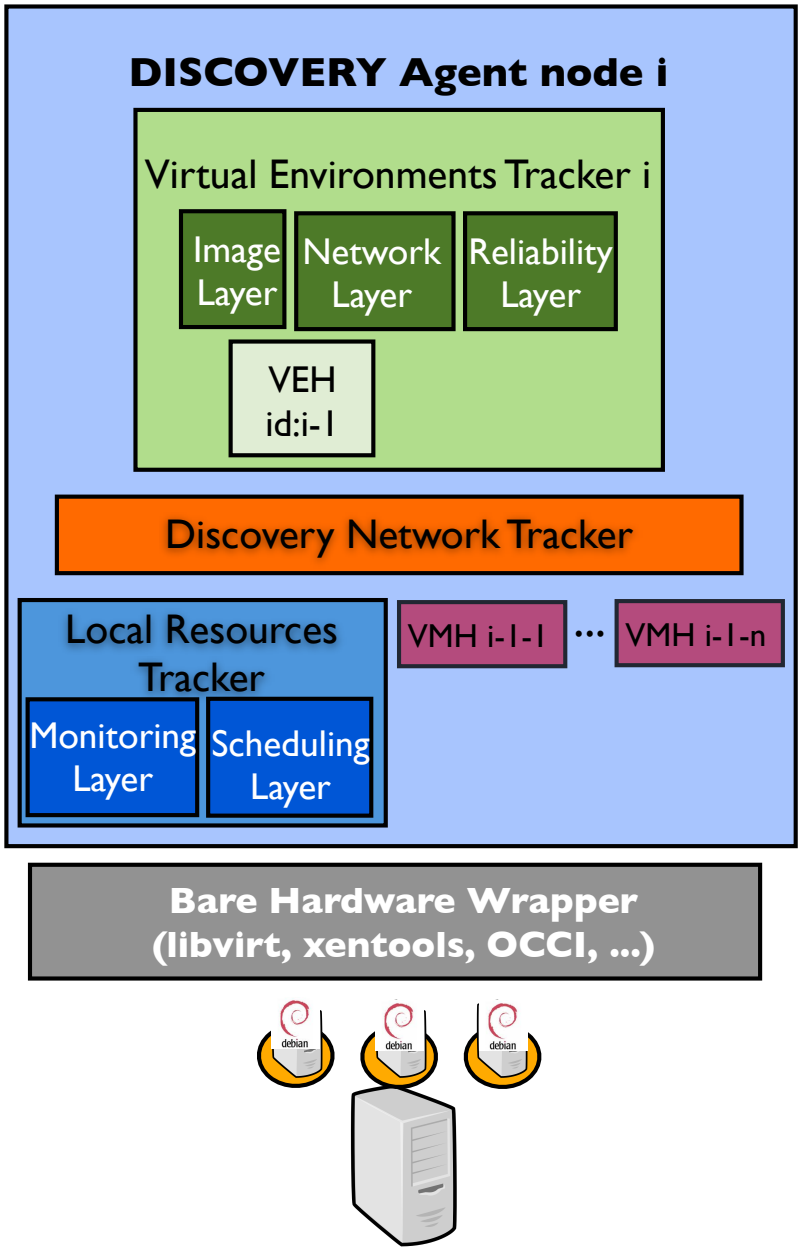

# DISCOVERY - Basic Usage



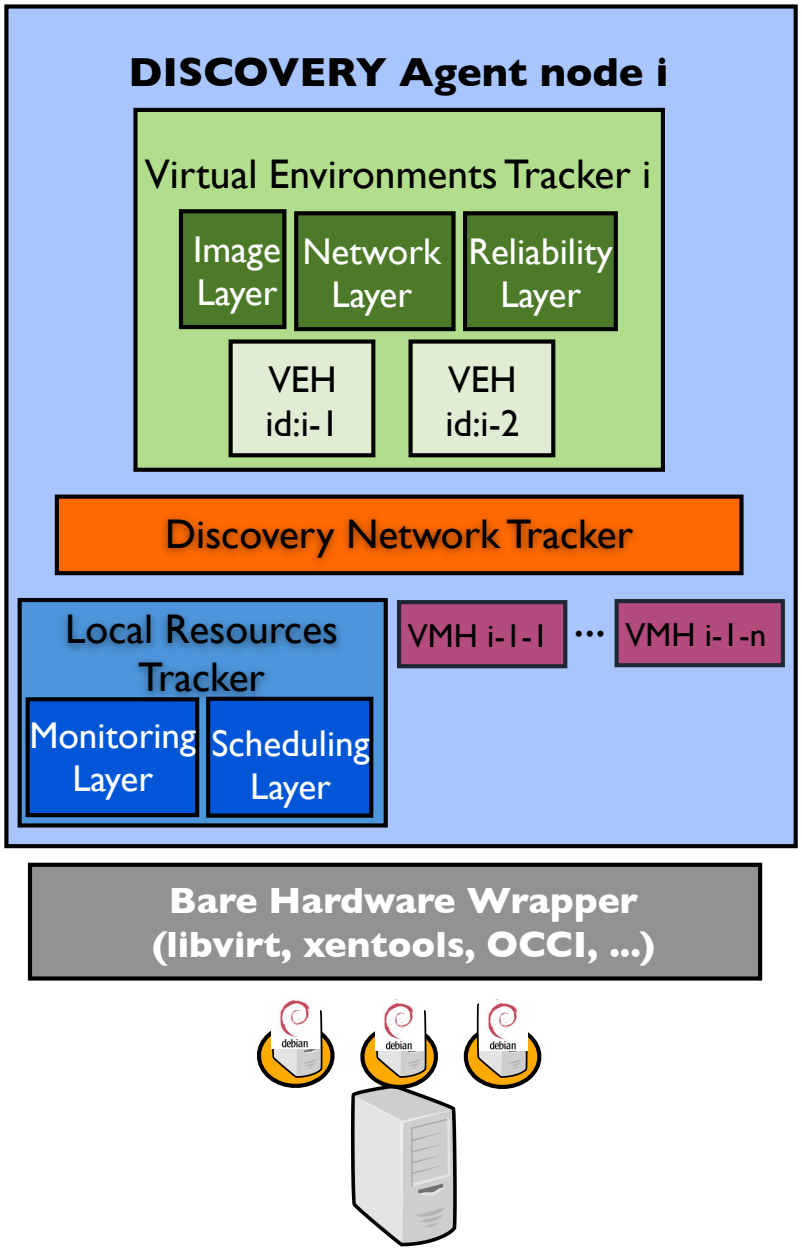

# DISCOVERY - Basic Usage



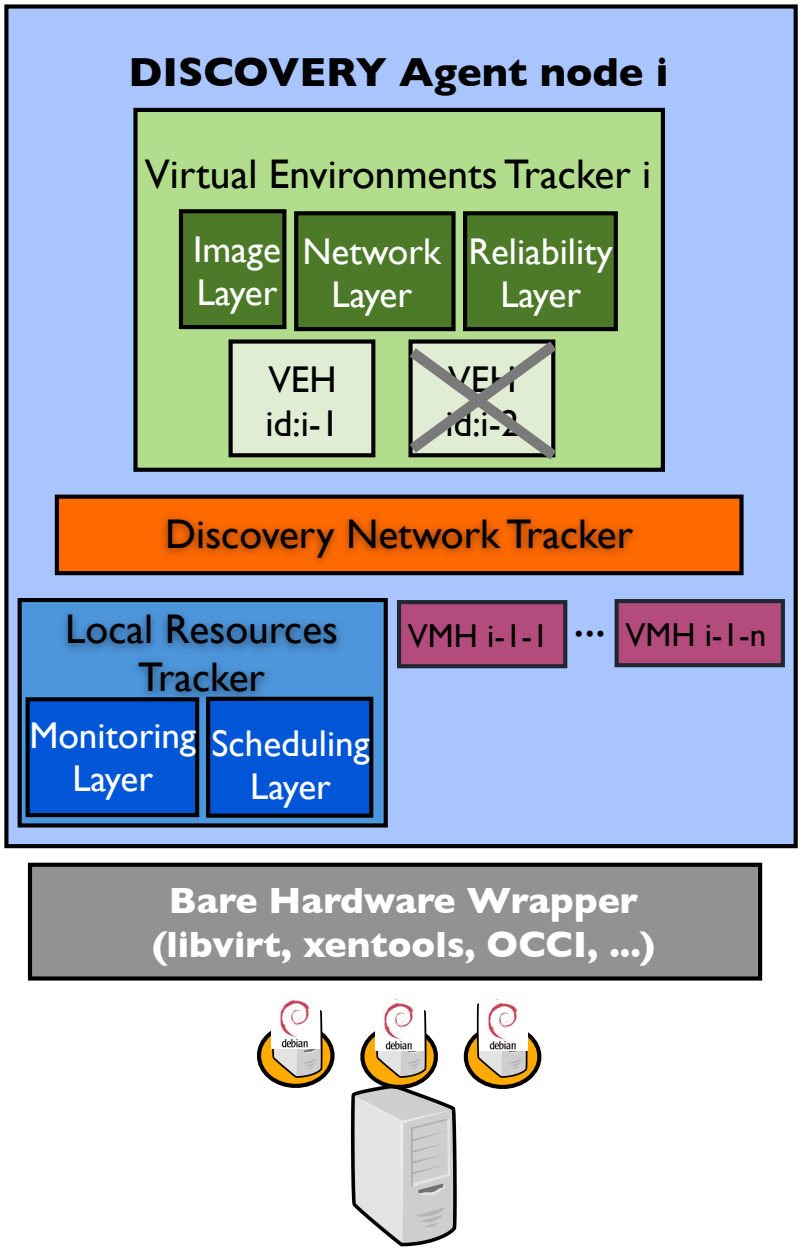

# DISCOVERY - Basic Usage



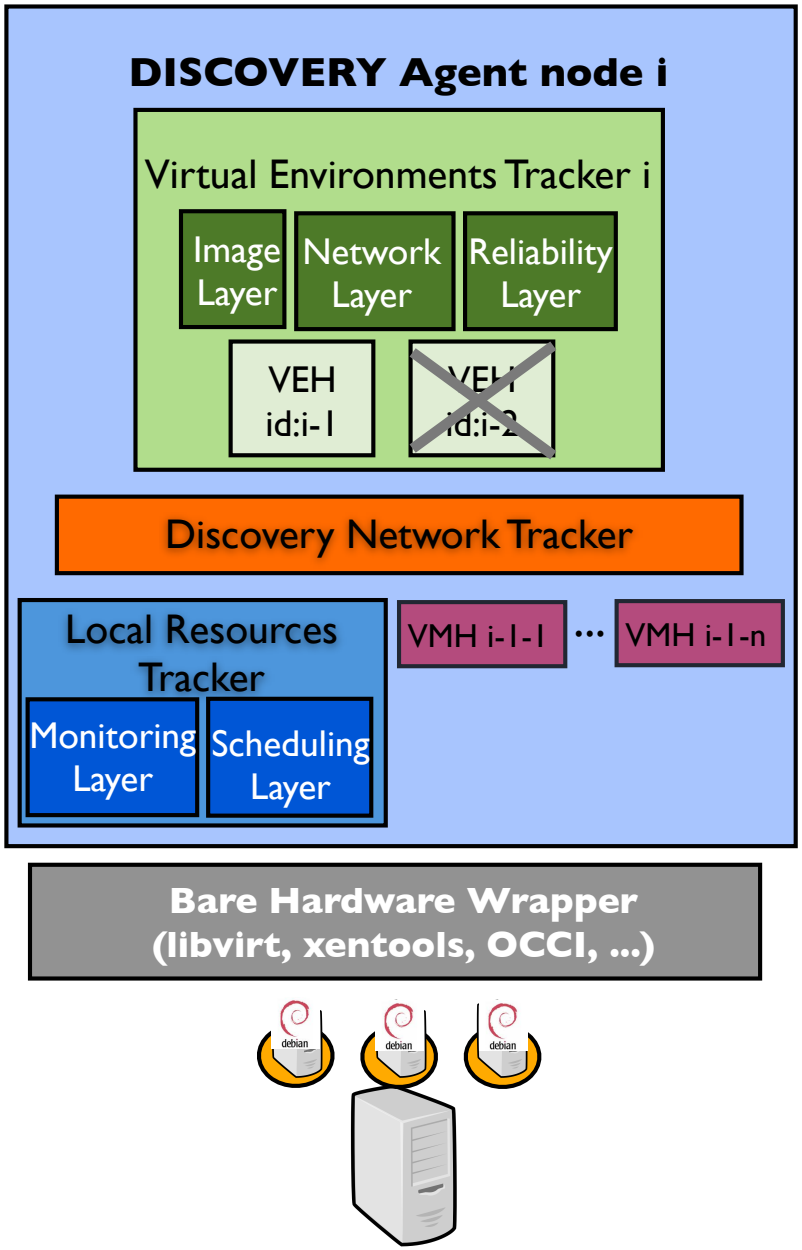
# DISCOVERY - Basic Usage



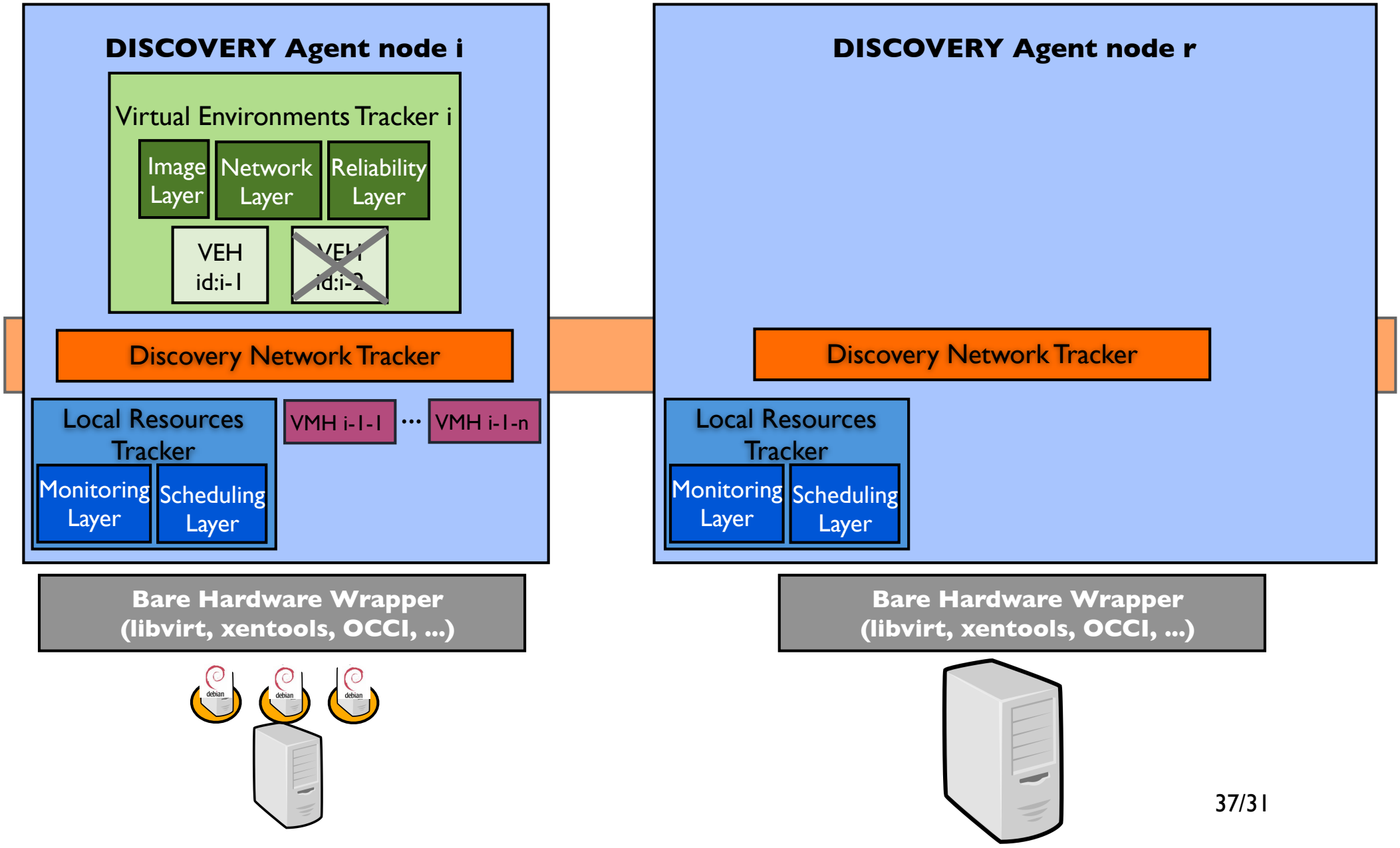
# DISCOVERY - Basic Usage



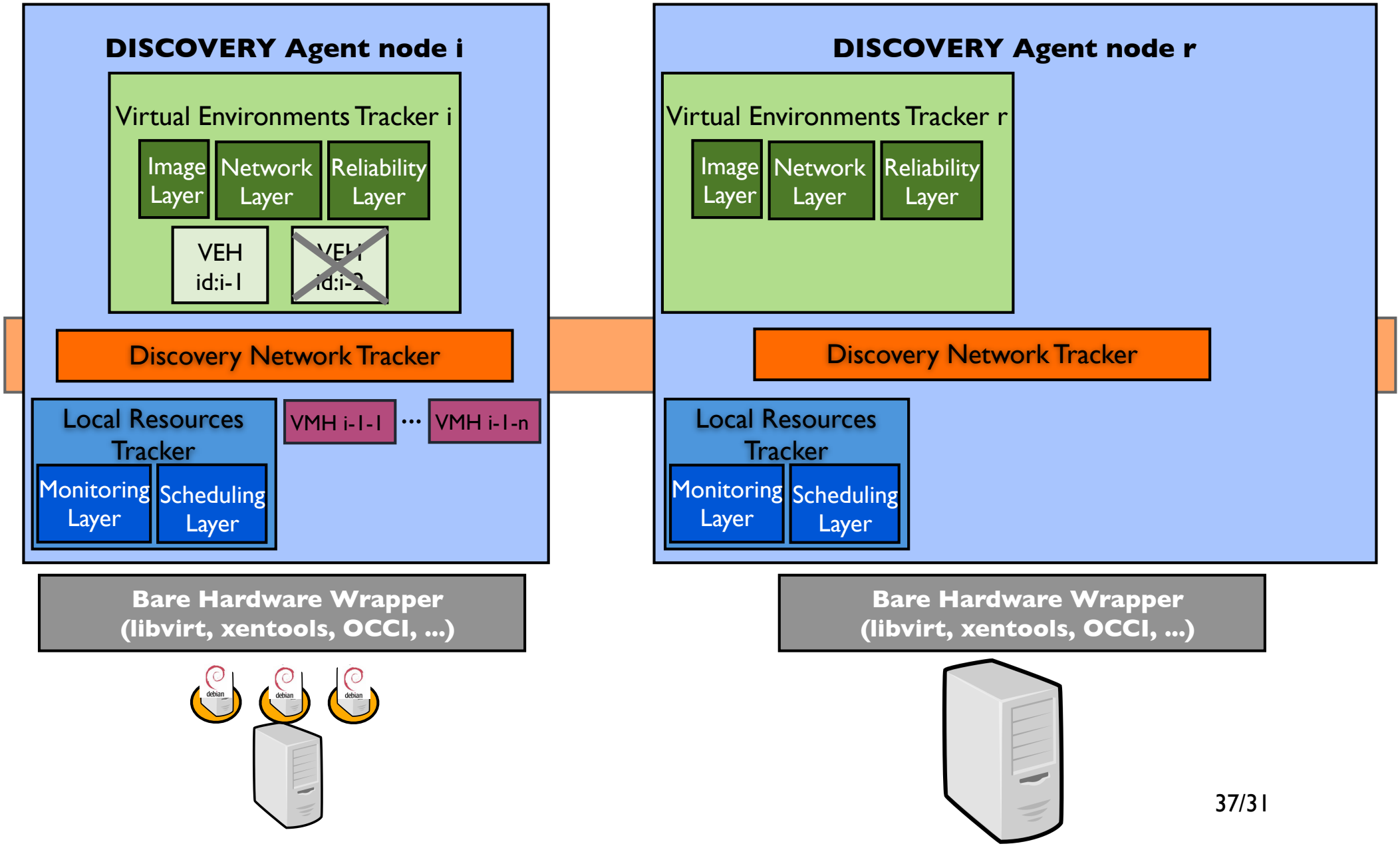
# DISCOVERY - Basic Usage



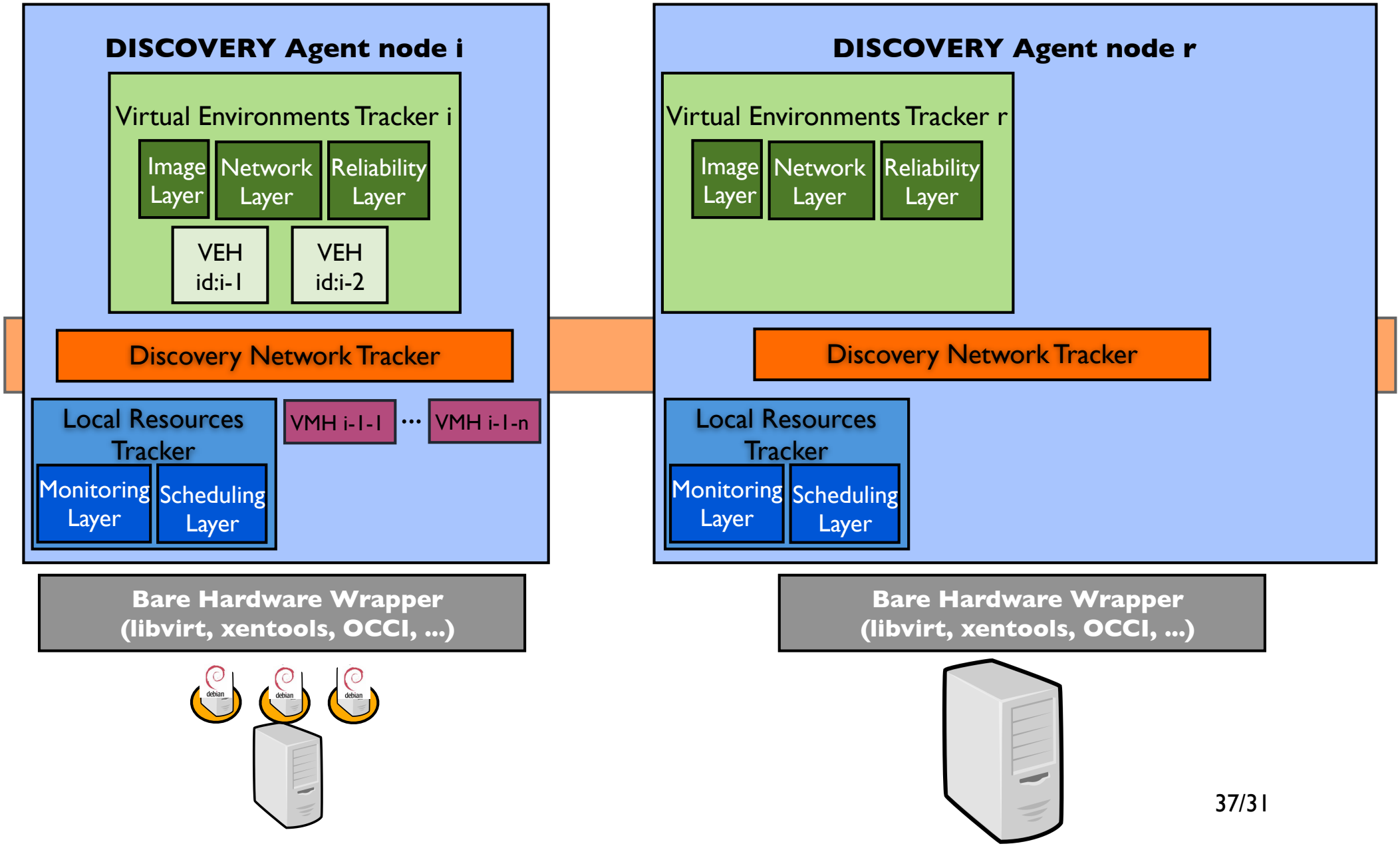

# DISCOVERY - Basic Usage



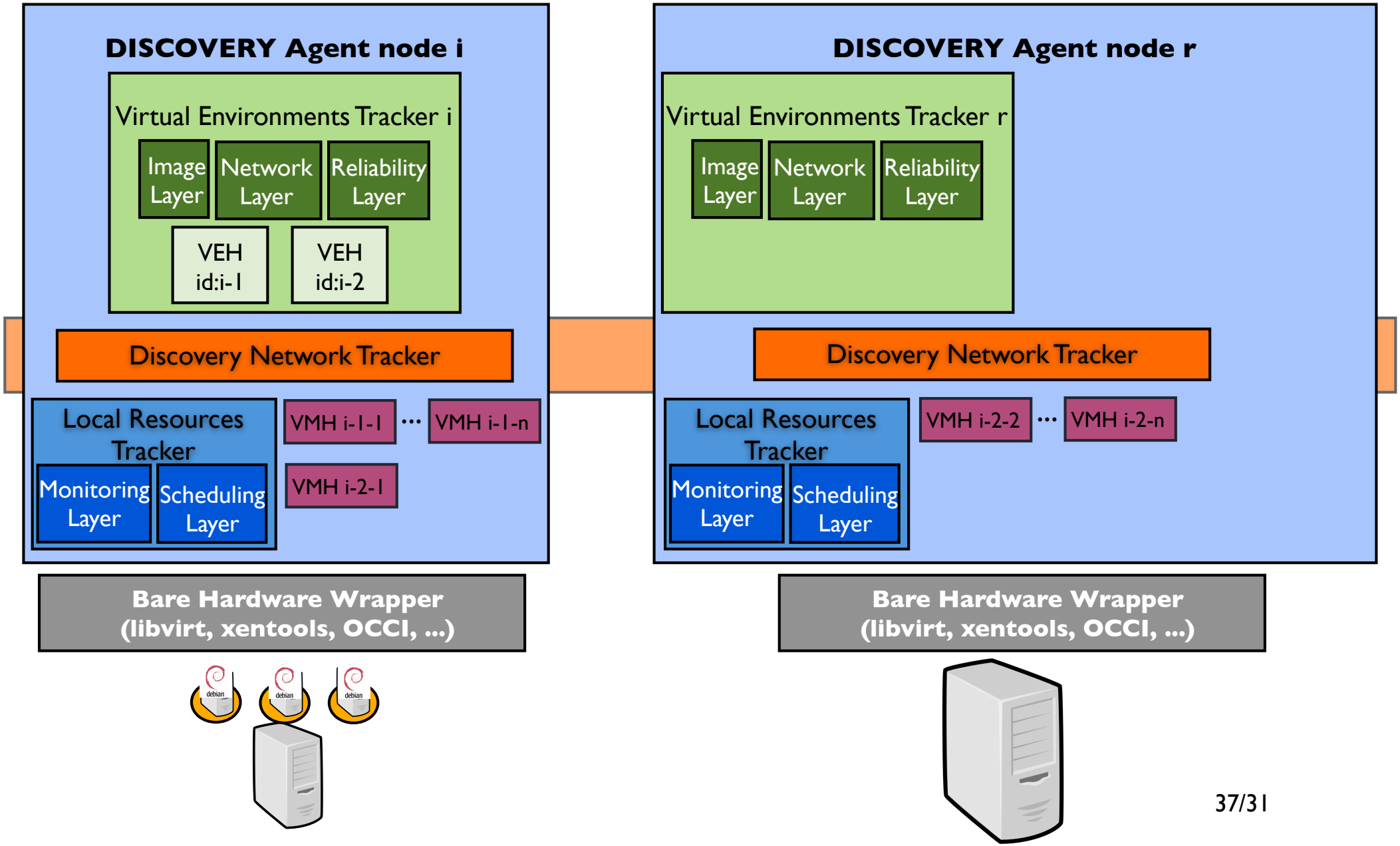
# DISCOVERY - Basic Usage



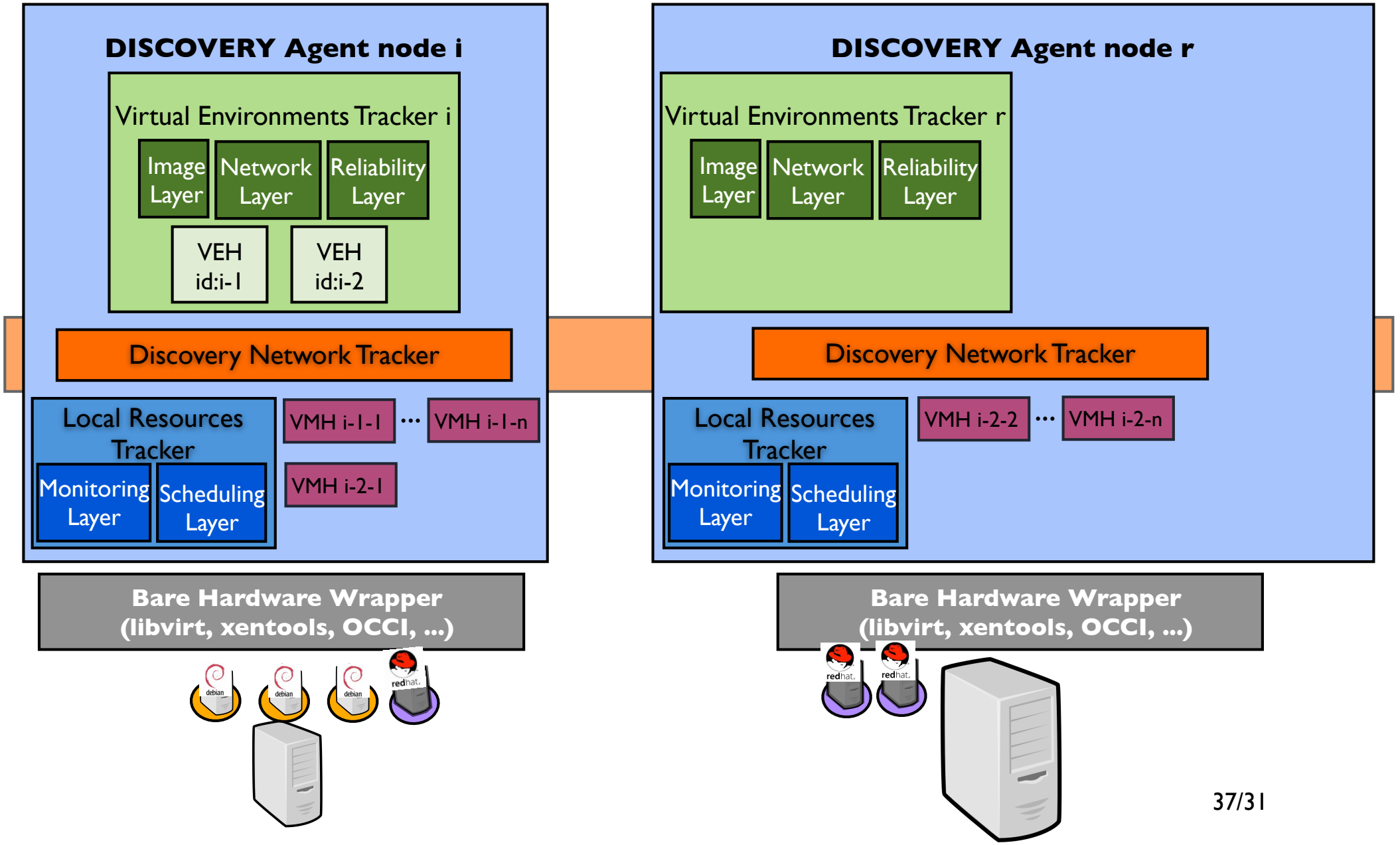
# DISCOVERY - Basic Usage



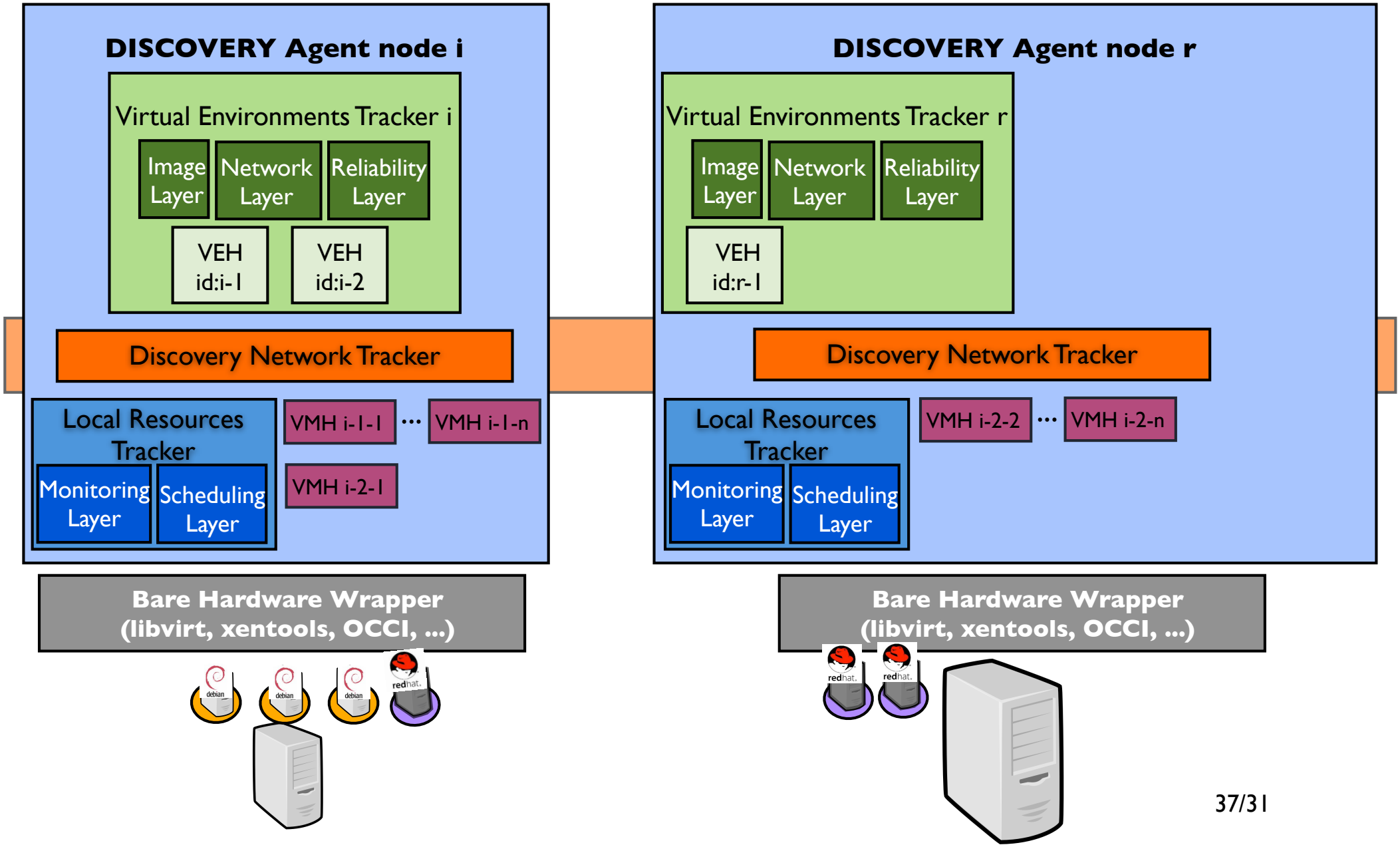
# DISCOVERY - Basic Usage



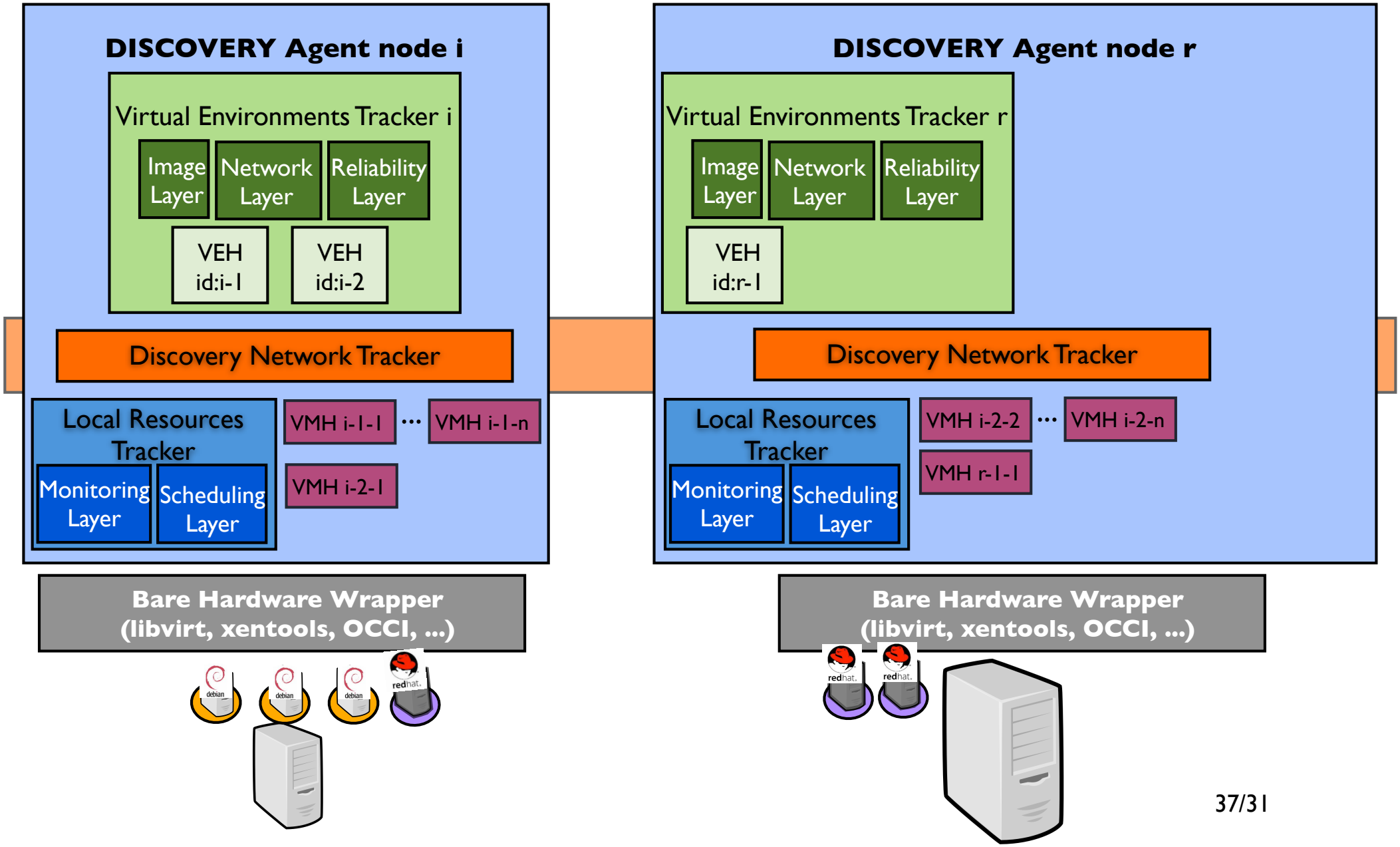
# DISCOVERY - Basic Usage



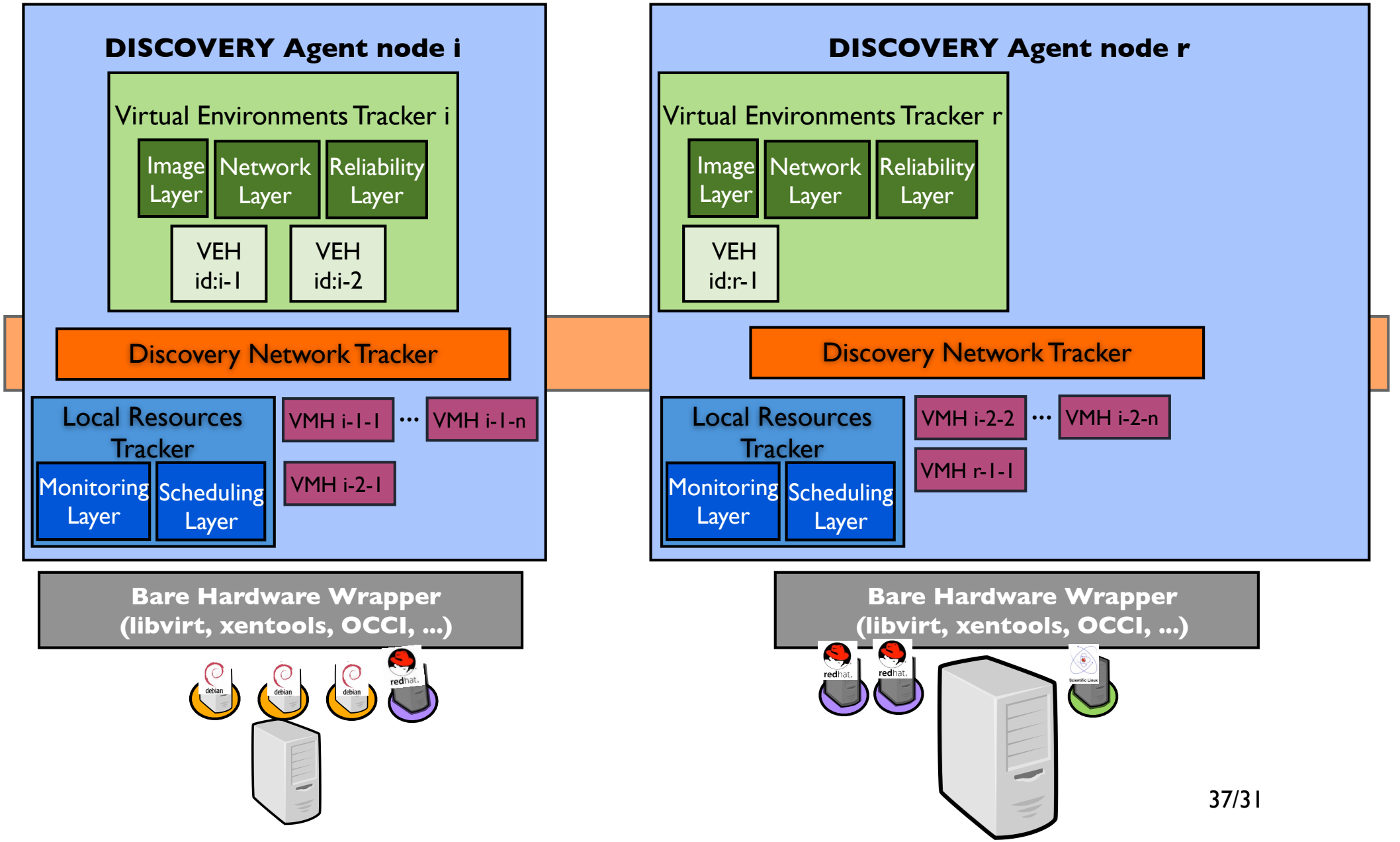
# DISCOVERY - Basic Usage



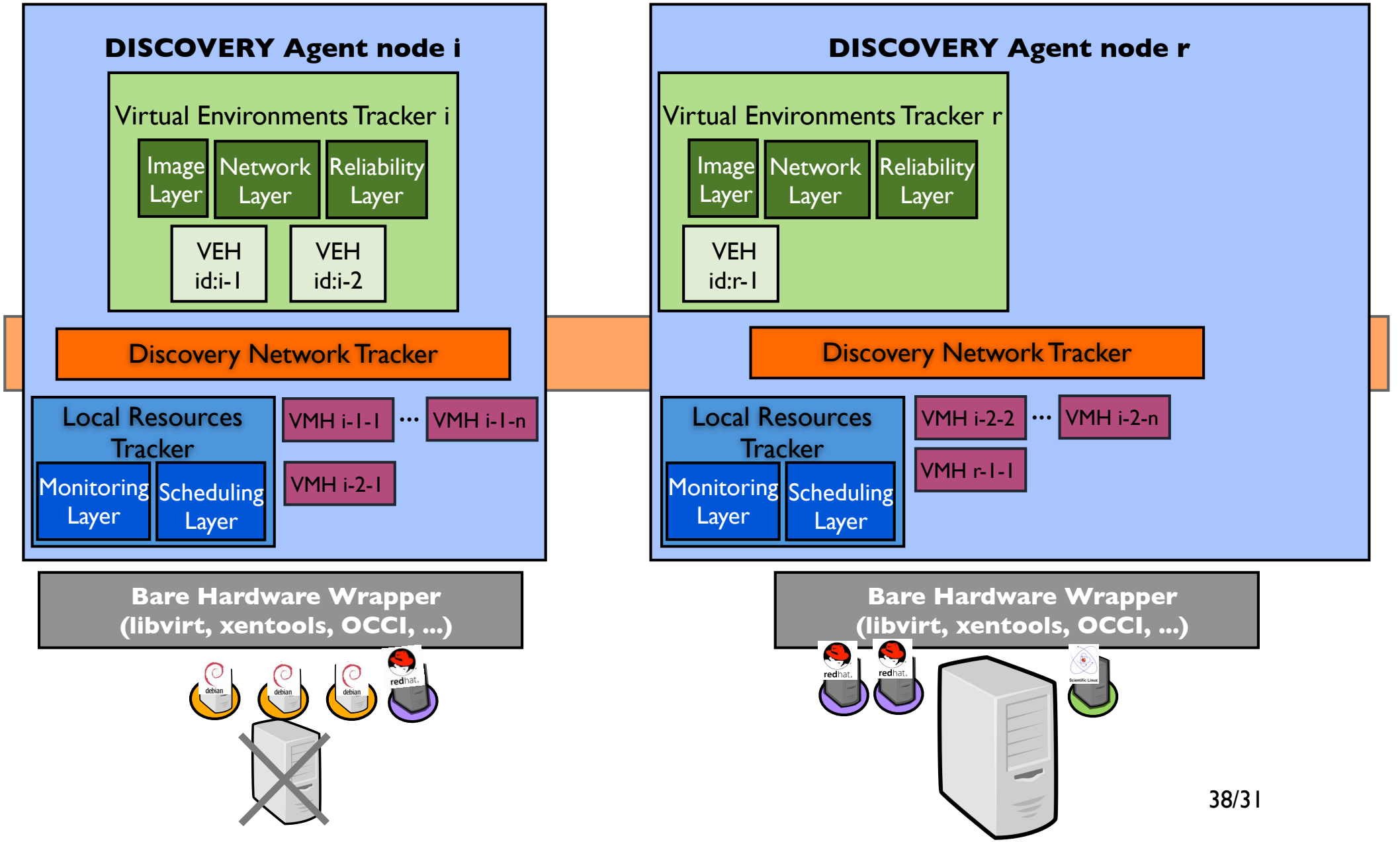
# DISCOVERY - Basic Usage



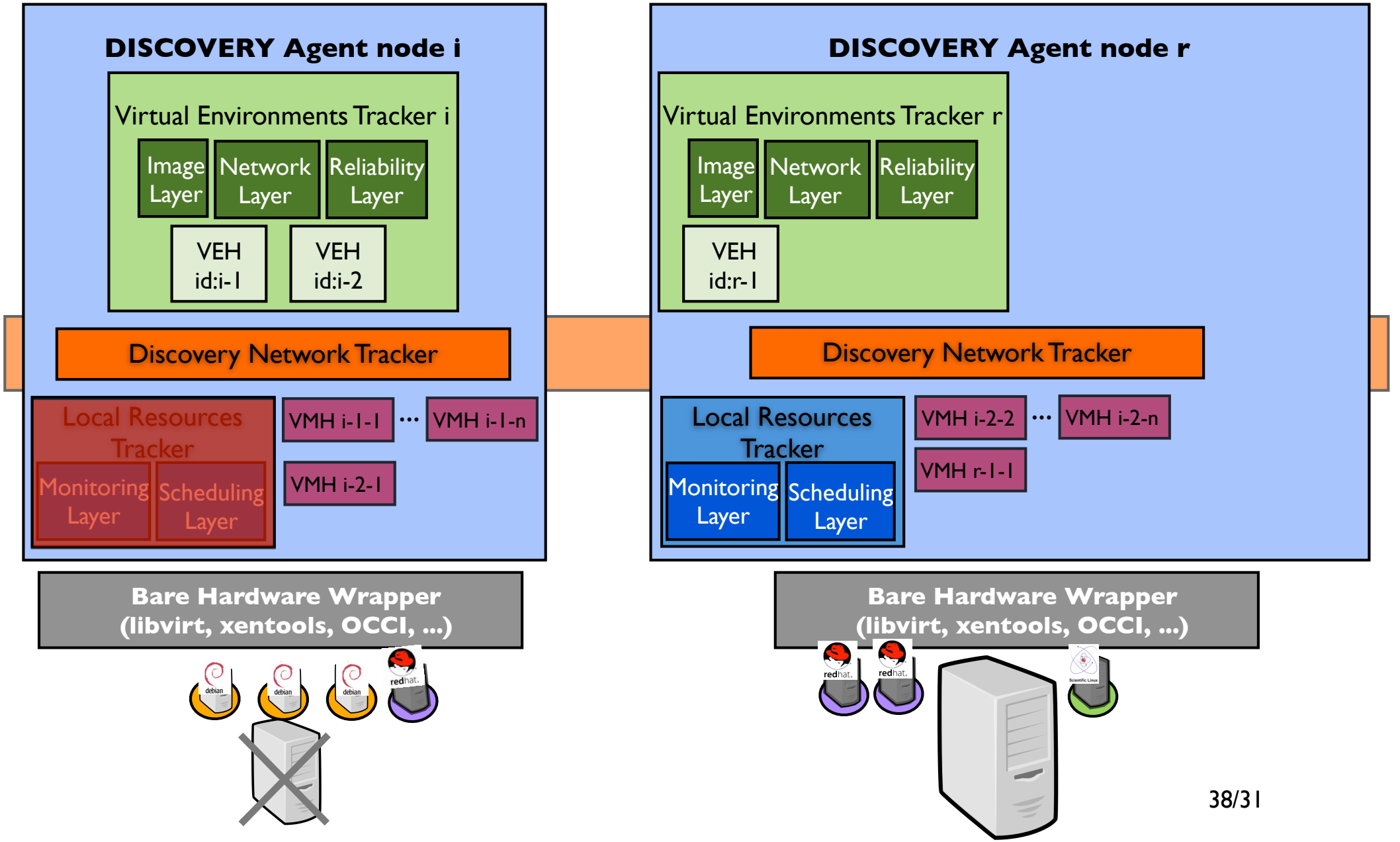
# DISCOVERY - Basic Usage



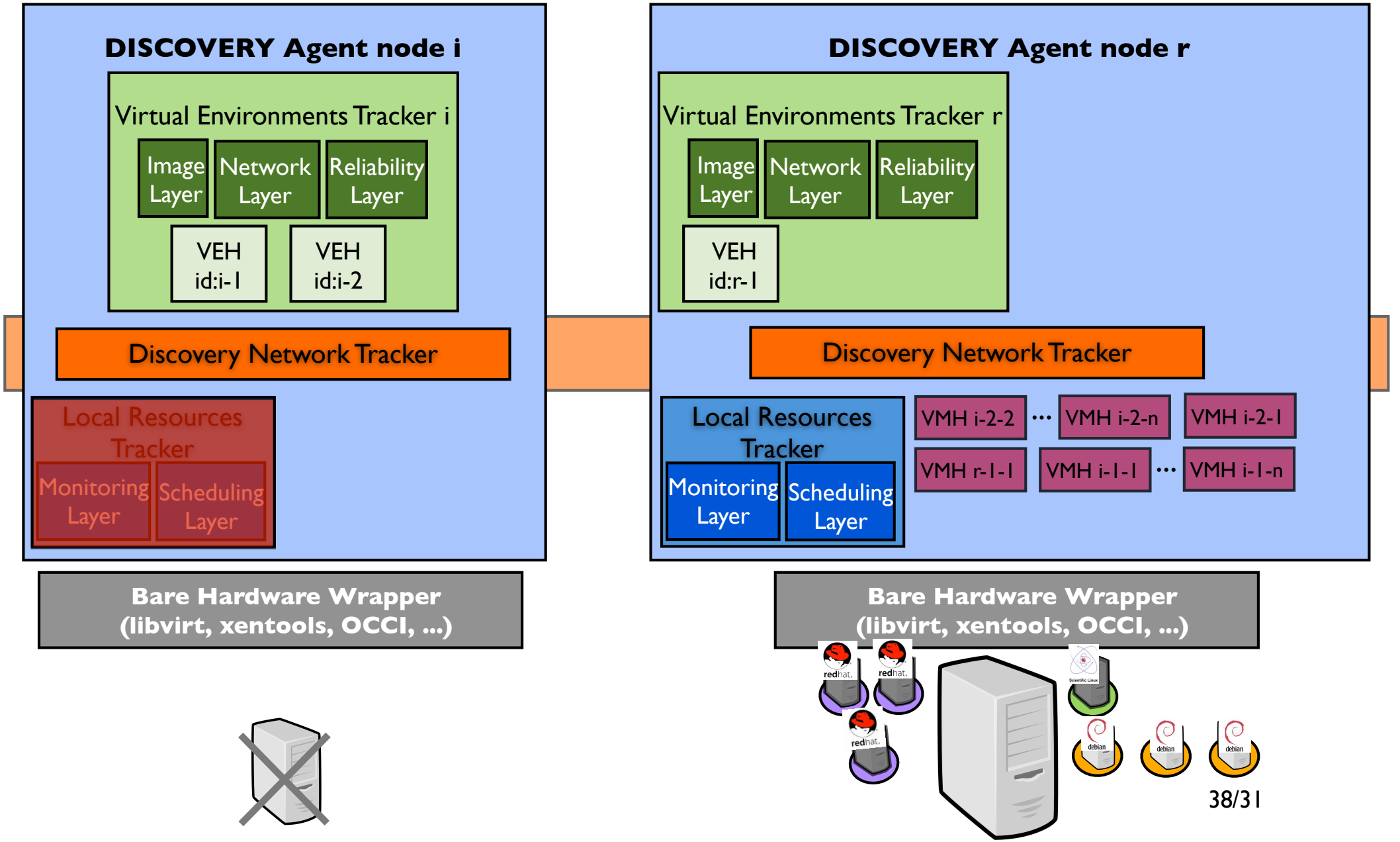
# DISCOVERY - Human removals



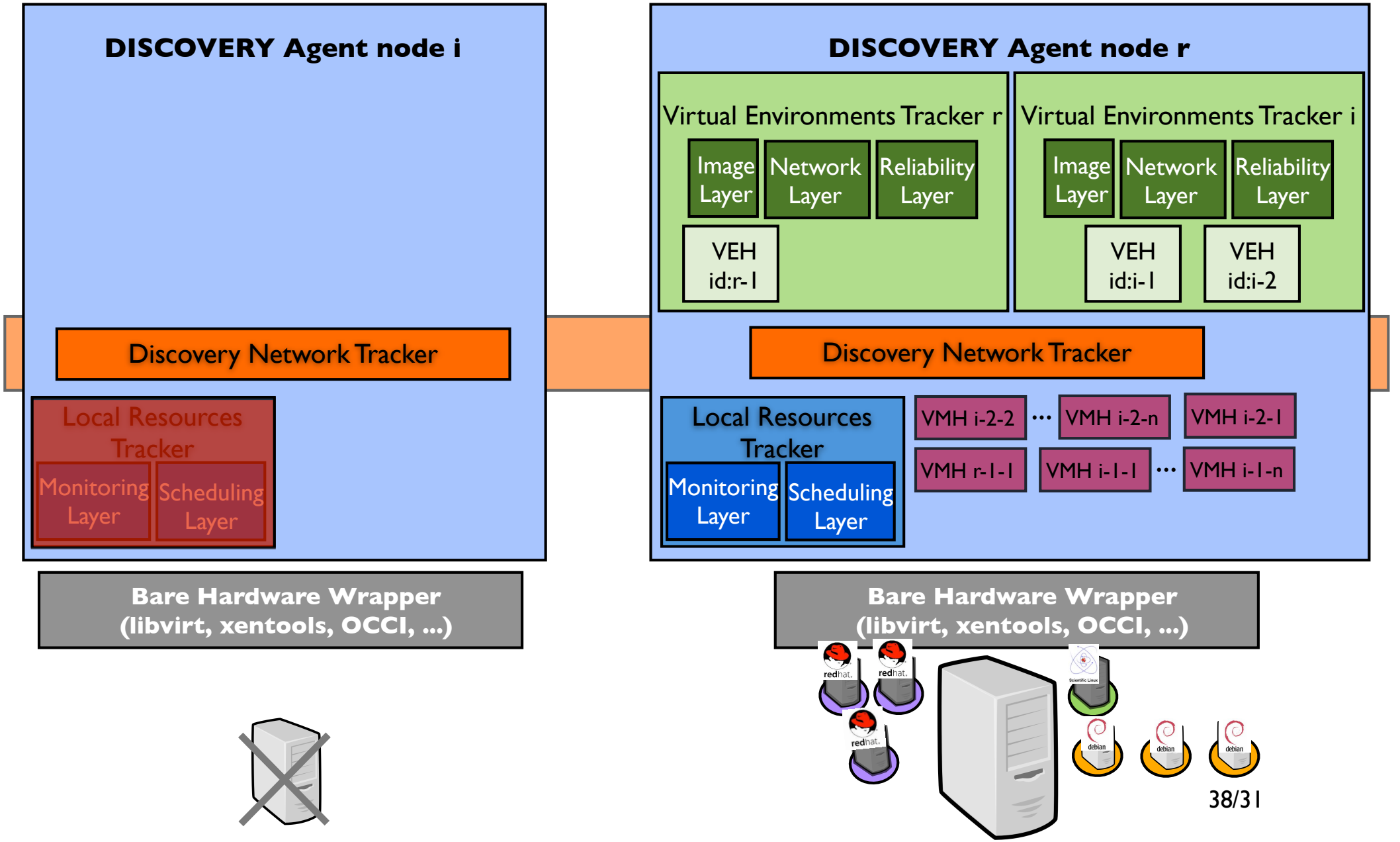
# DISCOVERY - Human removals



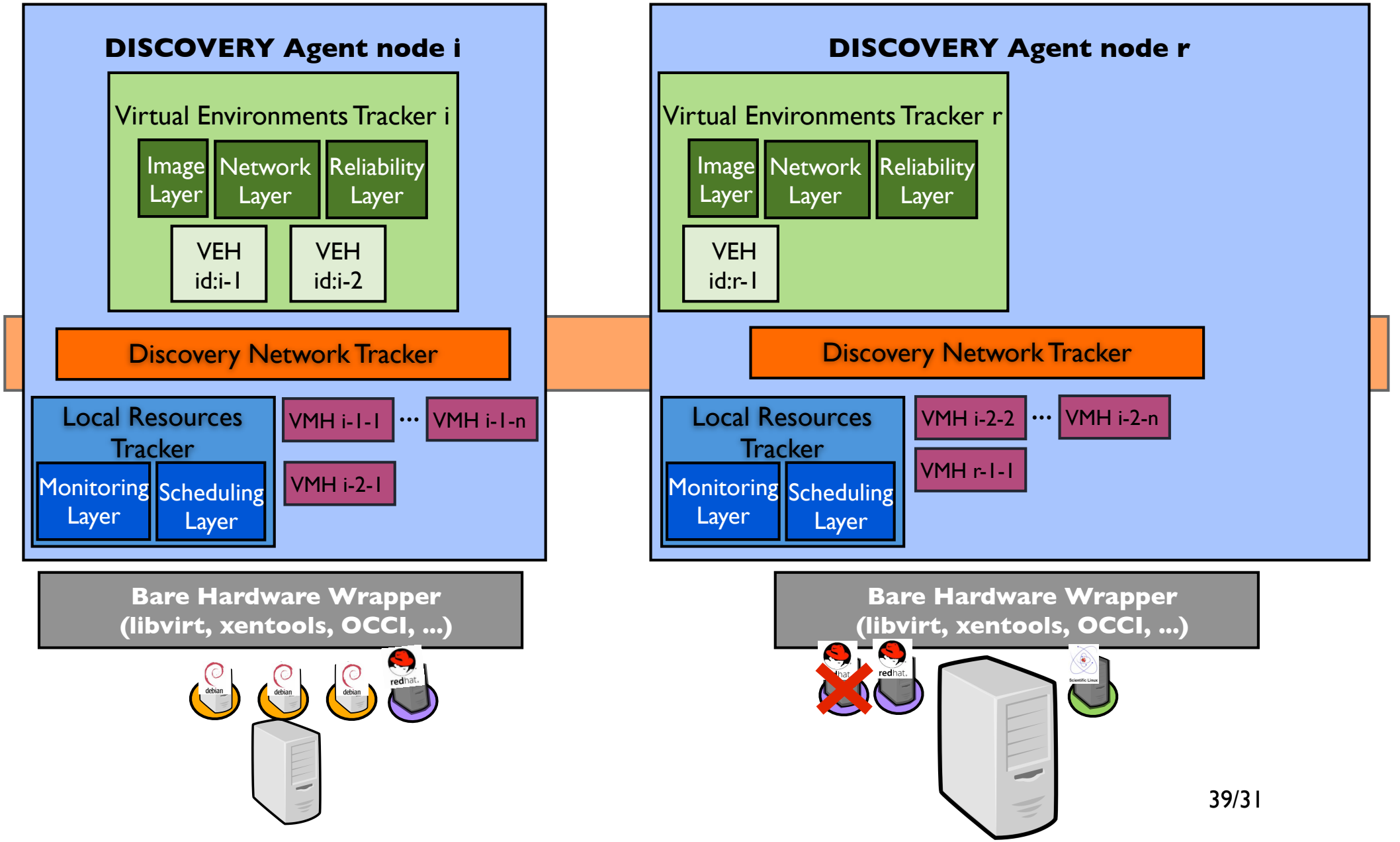
# DISCOVERY - Human removals



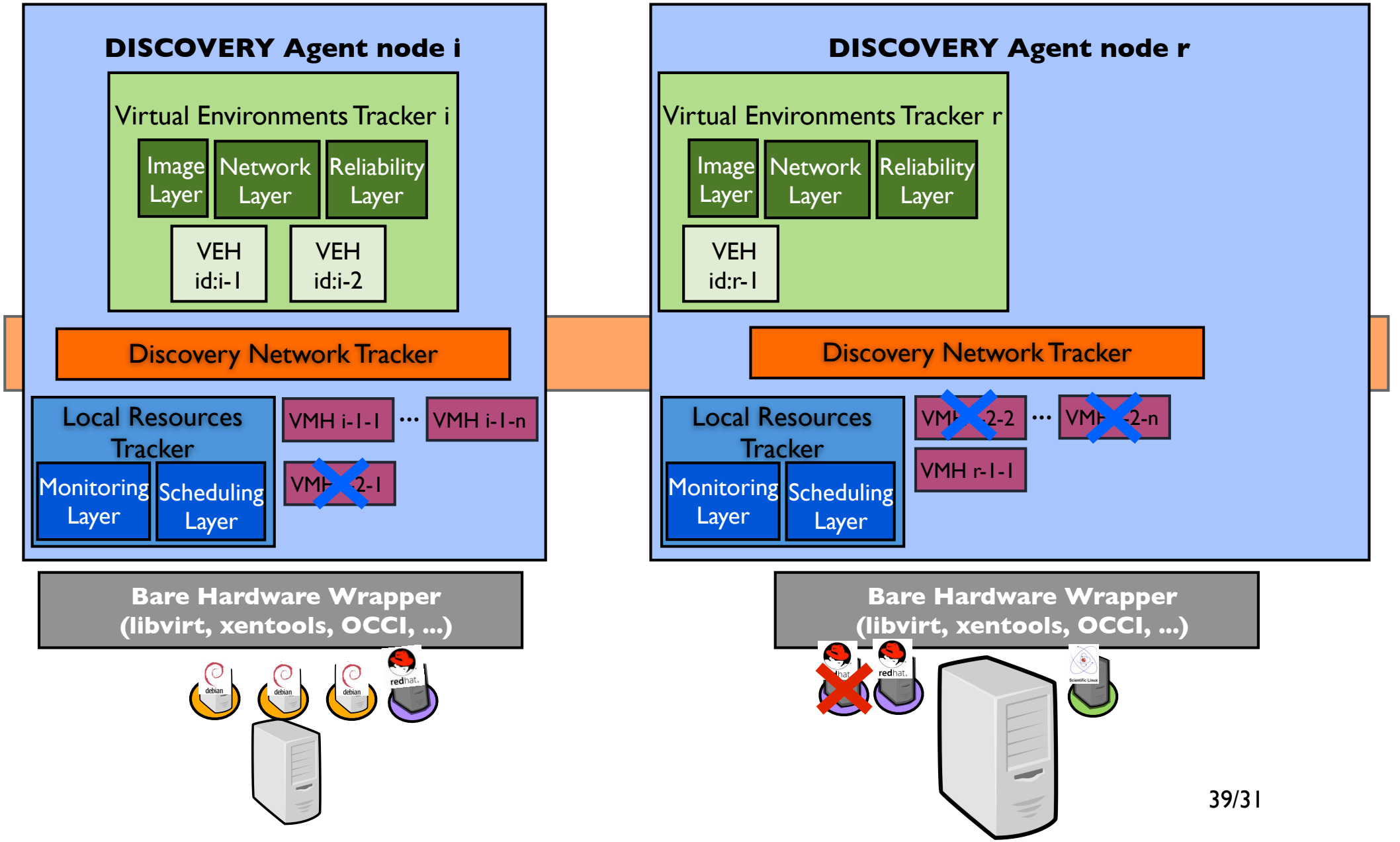
# DISCOVERY - Human removals



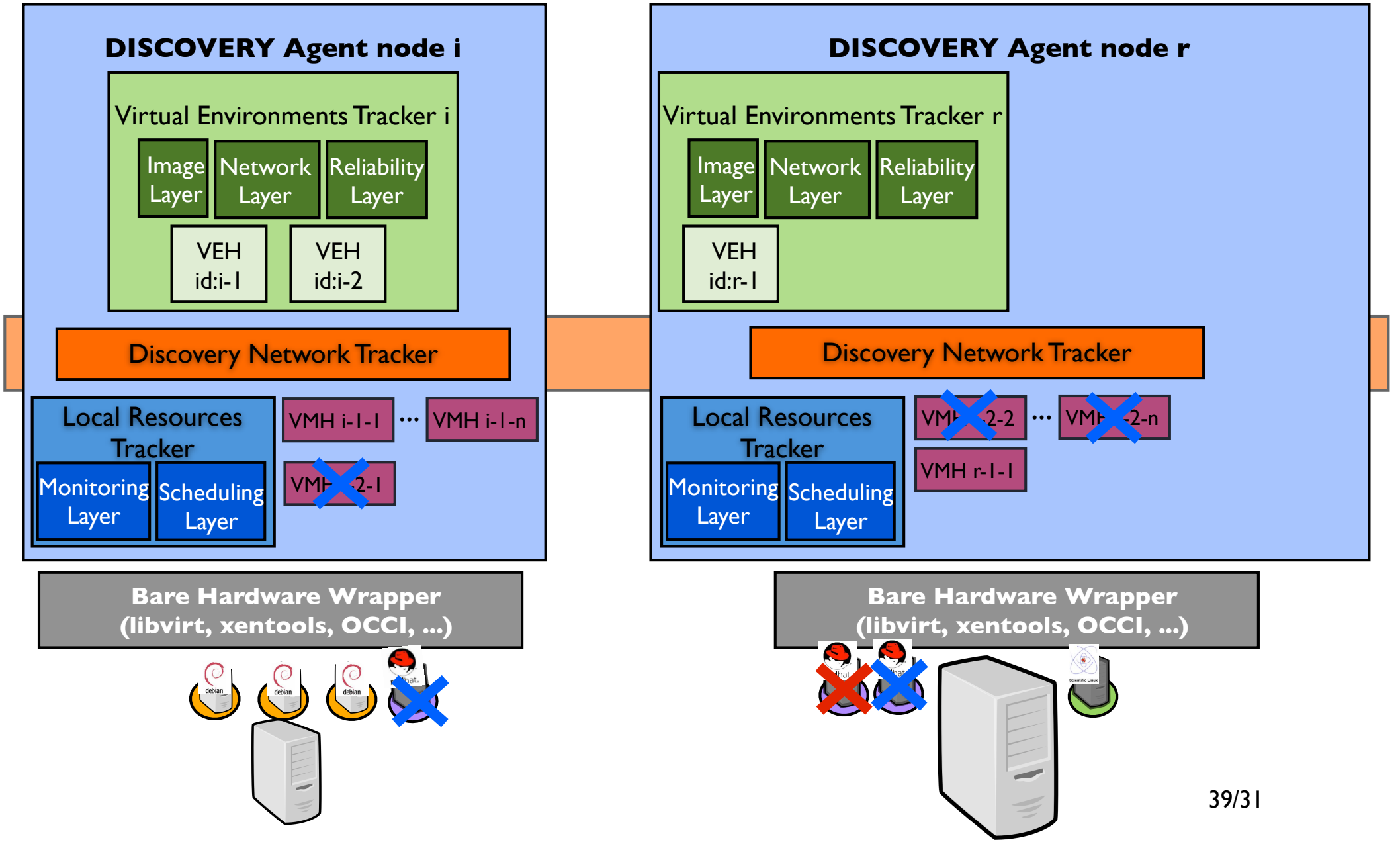
# DISCOVERY - VM Crashes



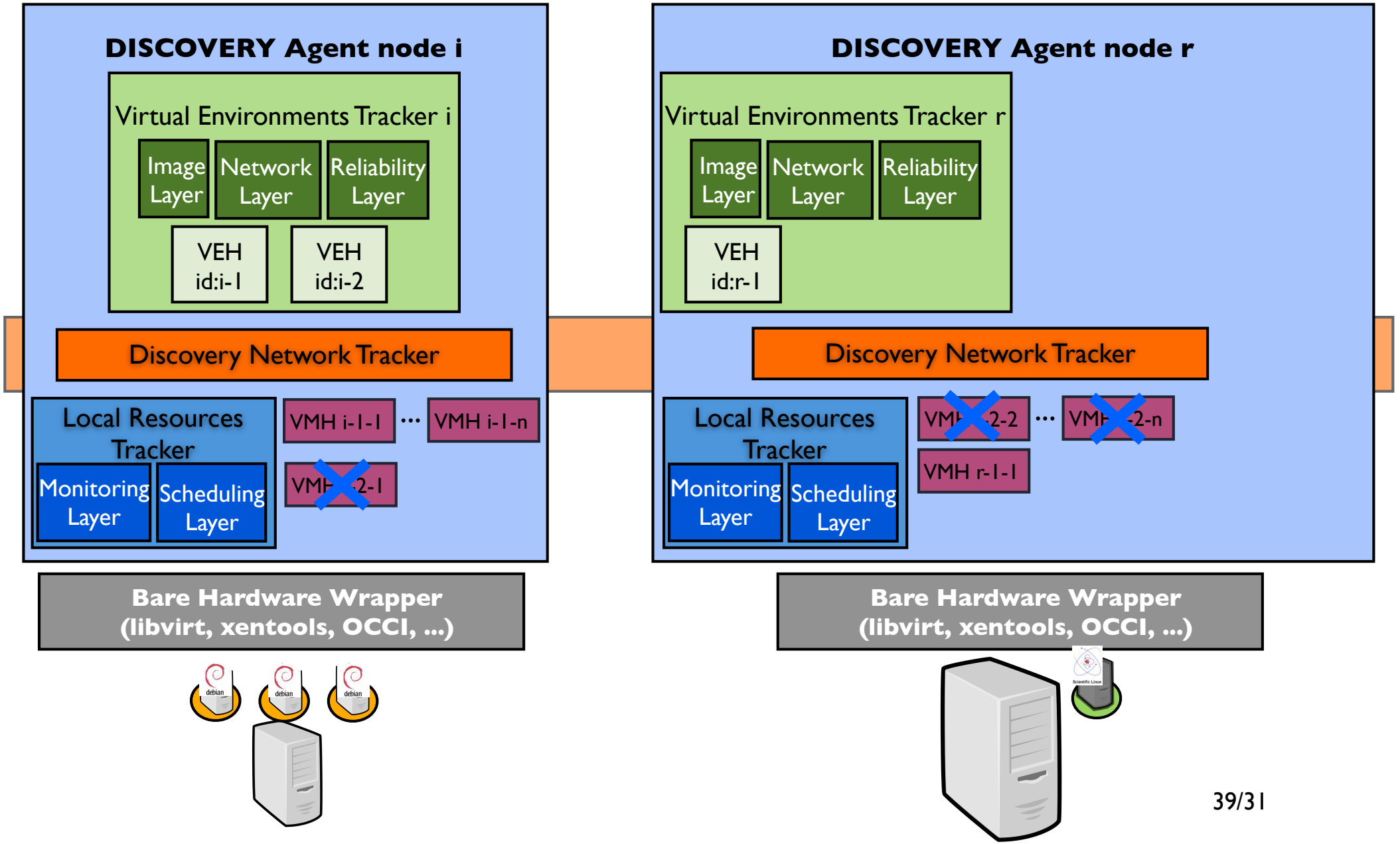
# DISCOVERY - VM Crashes



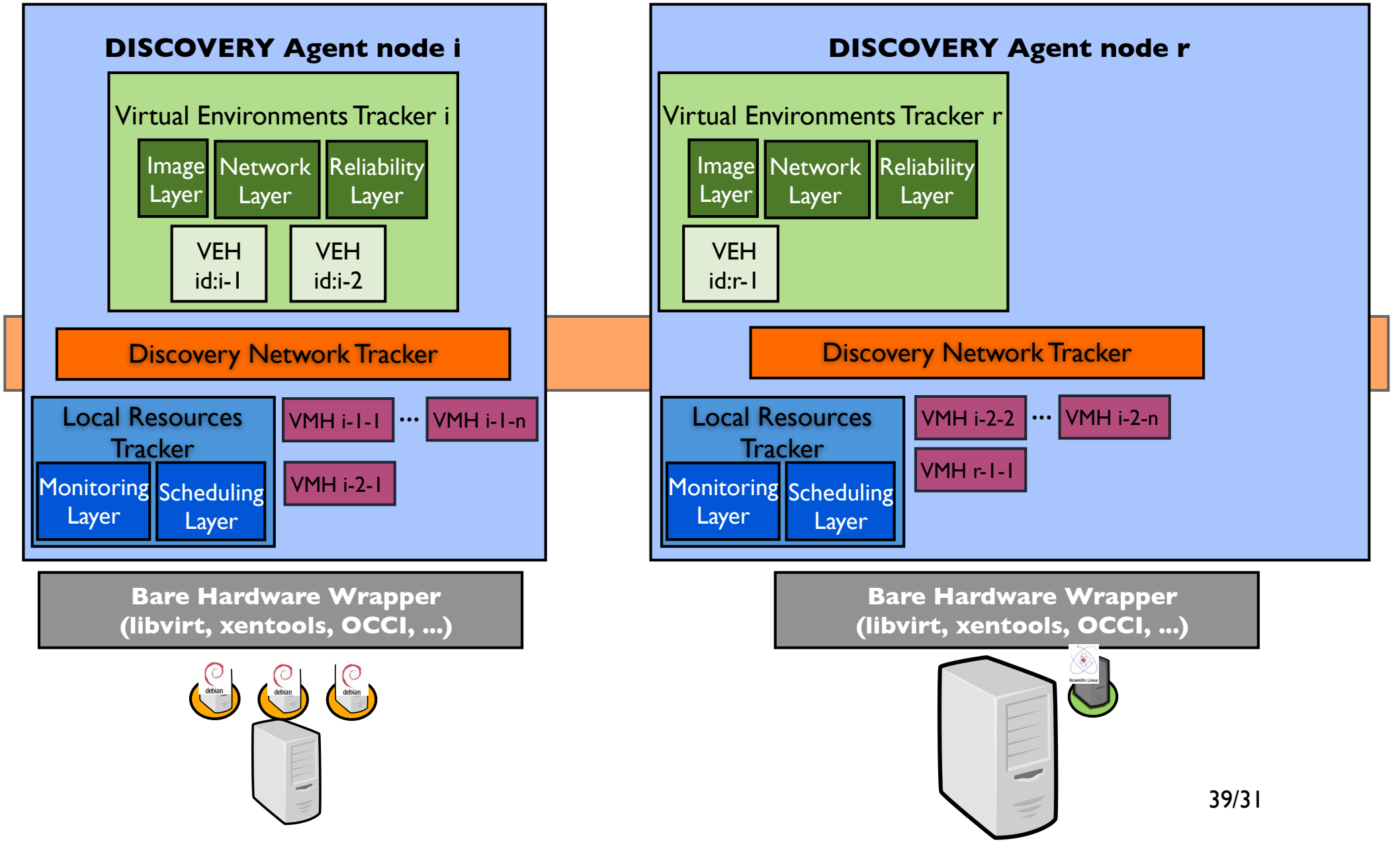
# DISCOVERY - VM Crashes



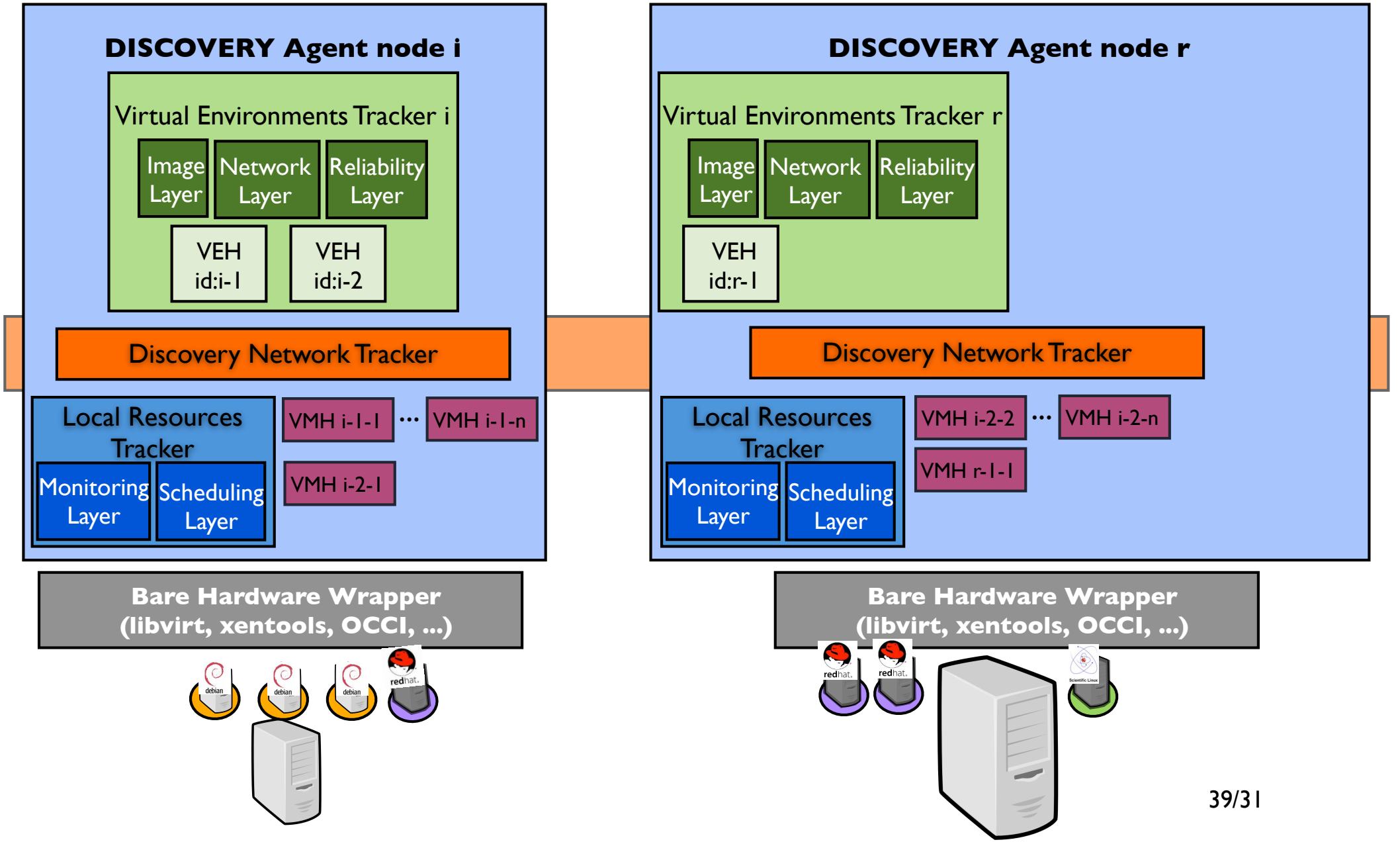
# DISCOVERY - VM Crashes



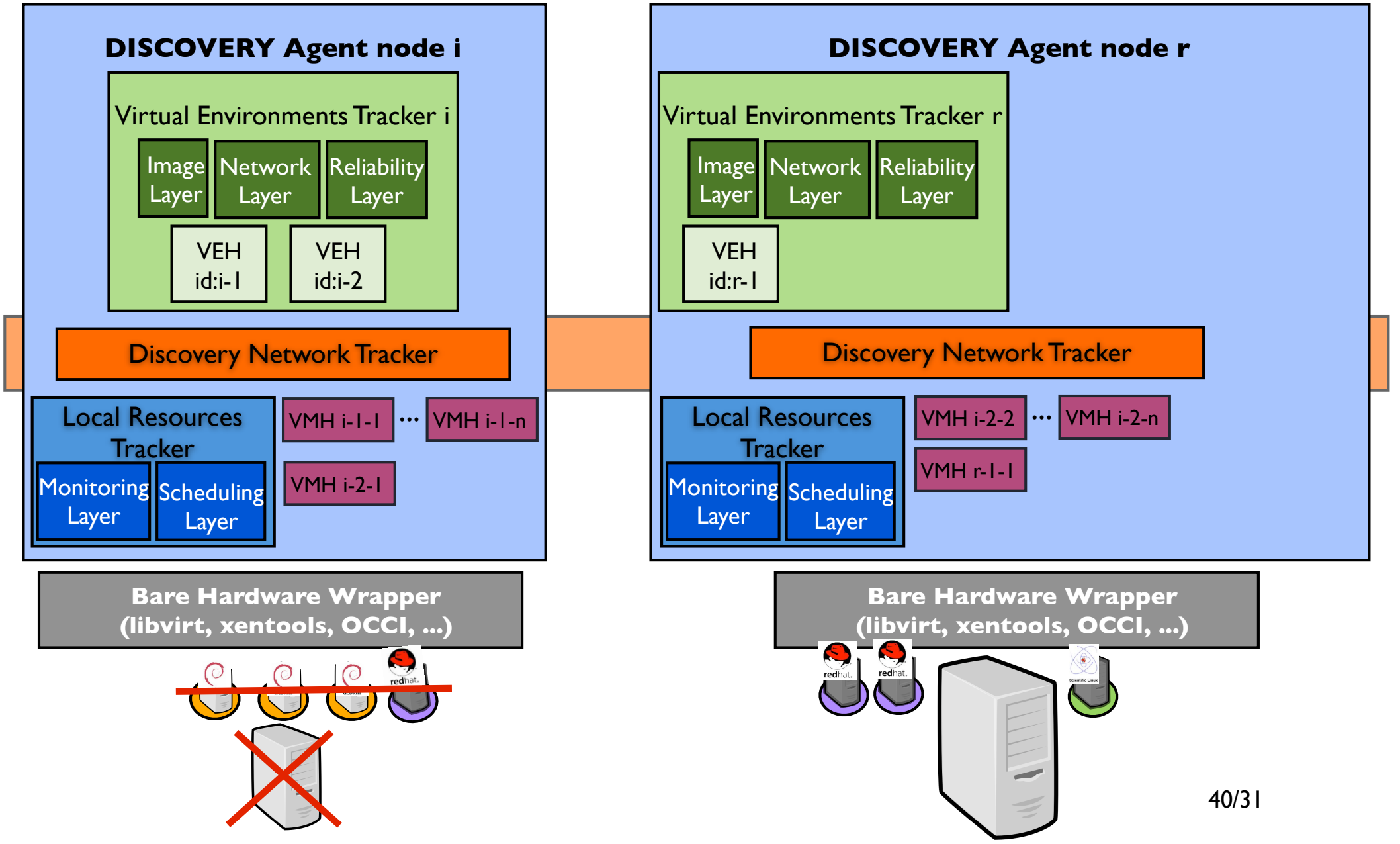
# DISCOVERY - VM Crashes



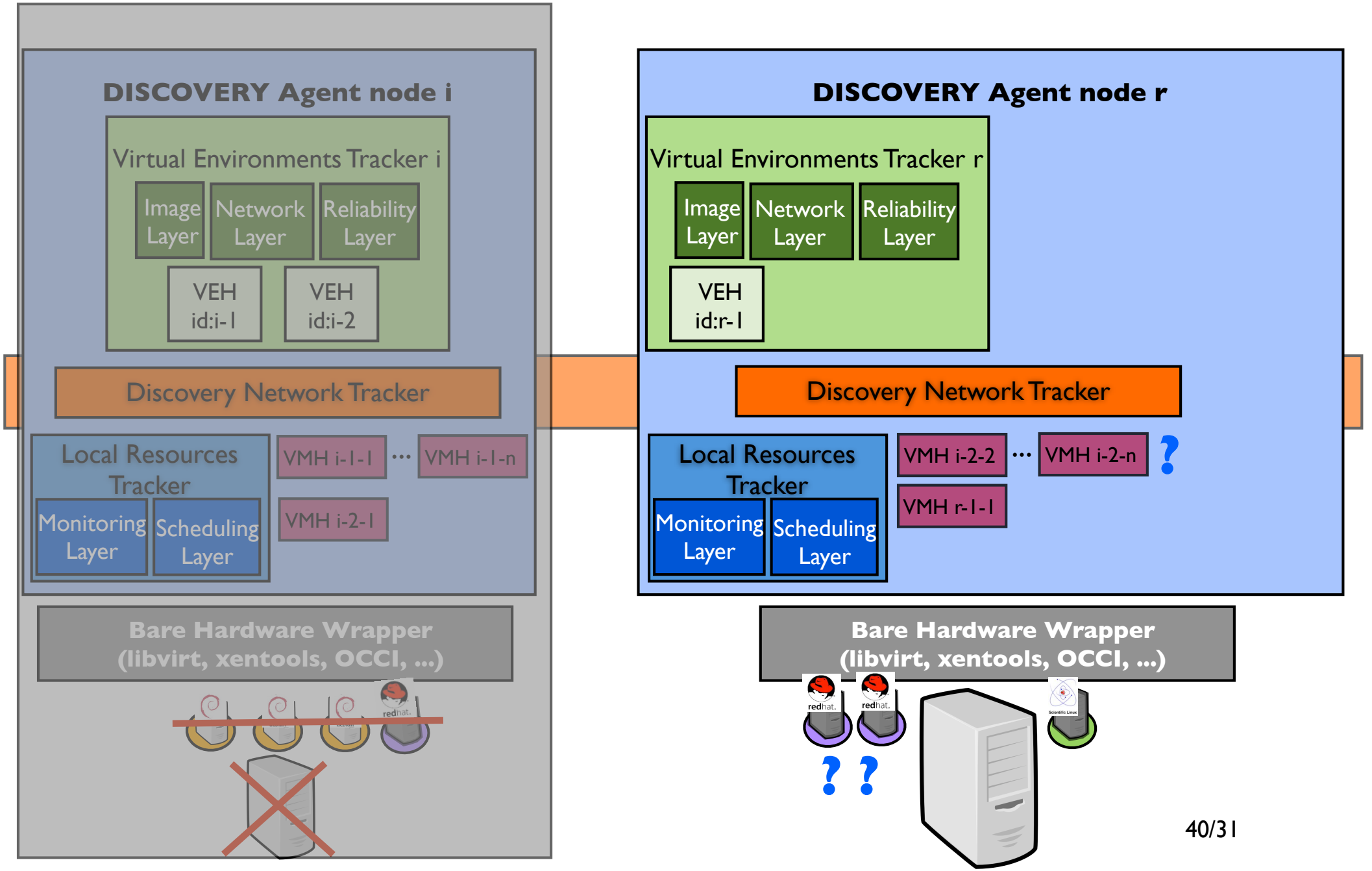
# DISCOVERY - VM Crashes



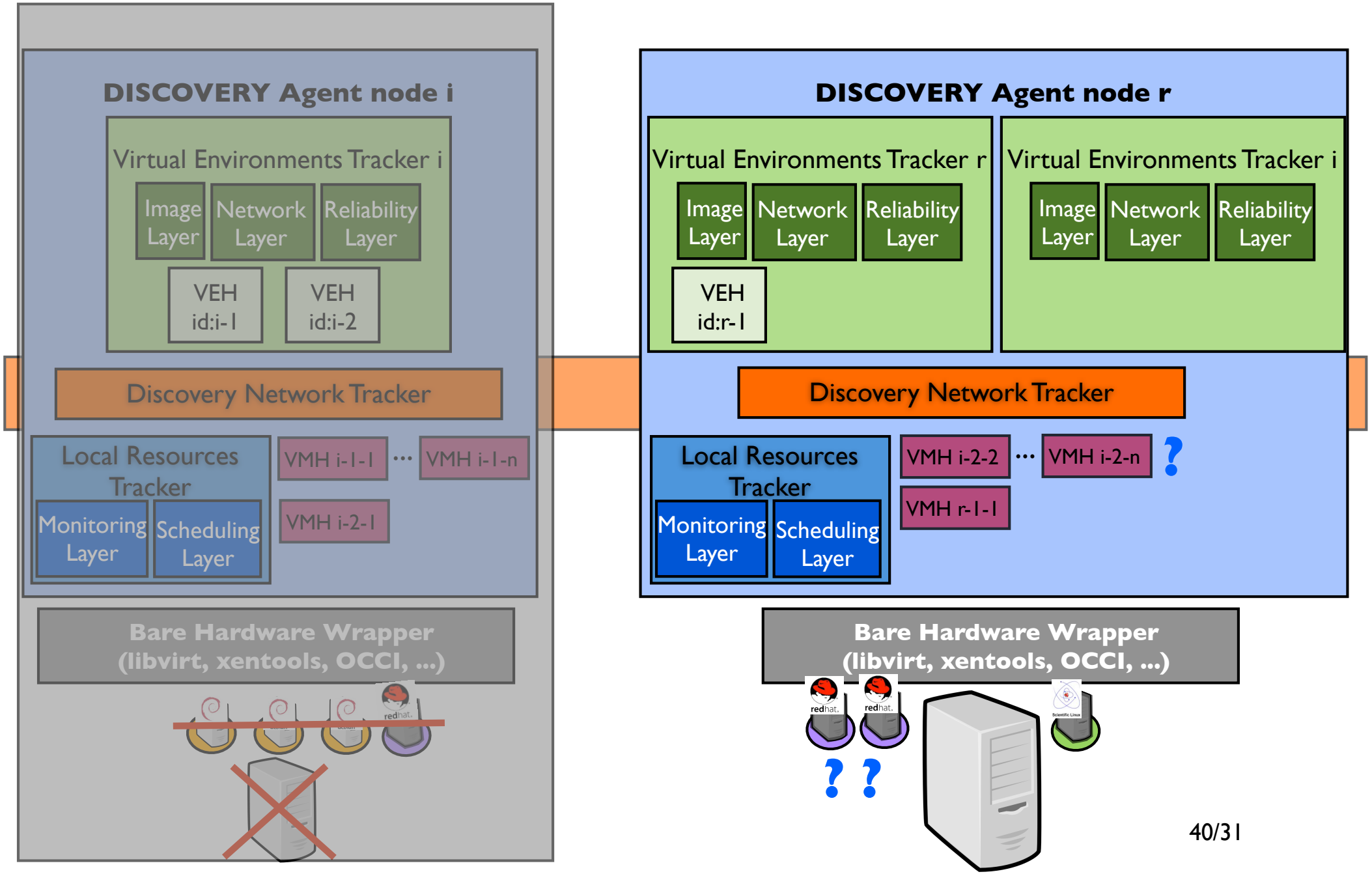
# DISCOVERY - Nodes Crashes



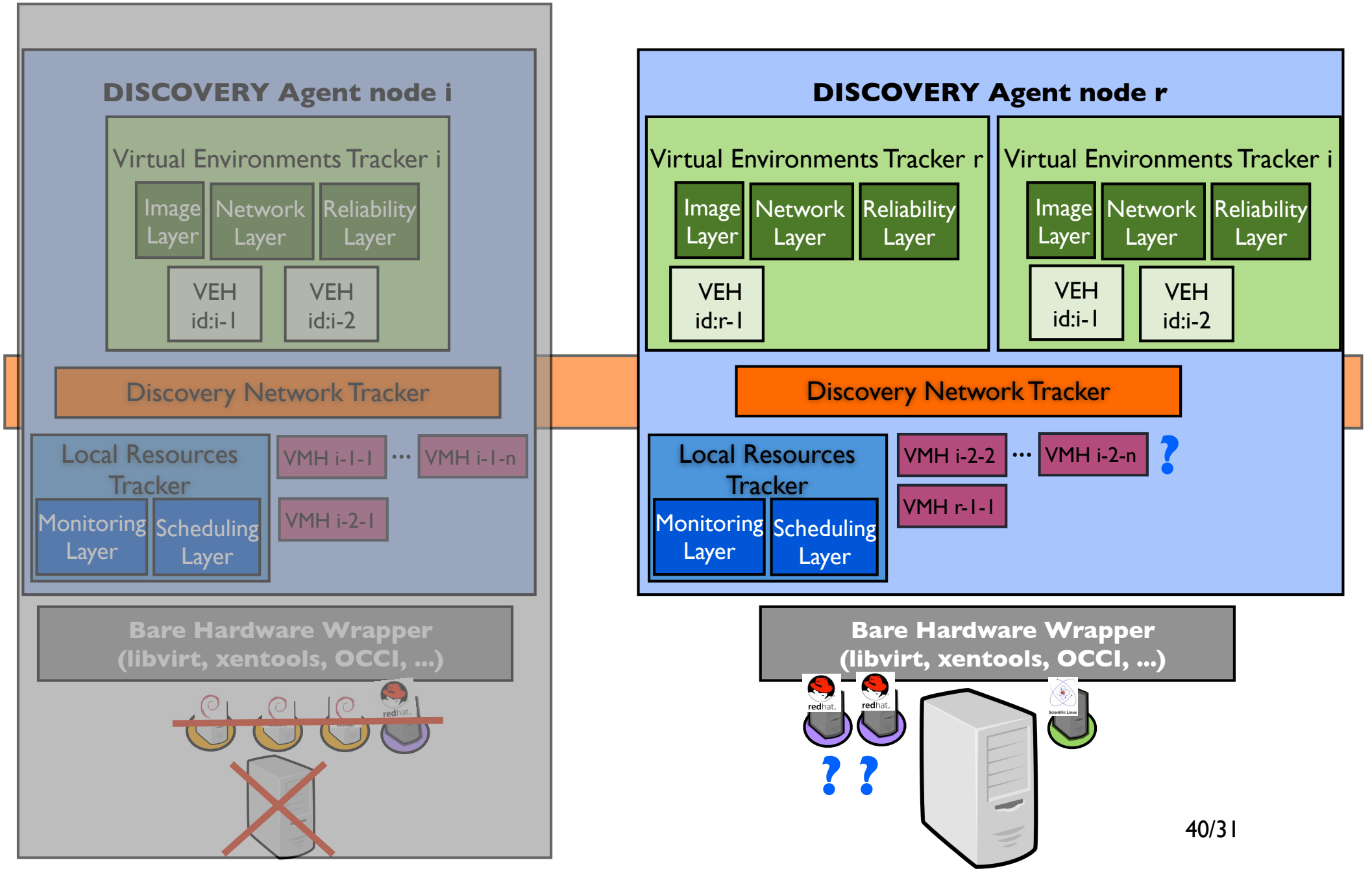
# DISCOVERY - Nodes Crashes



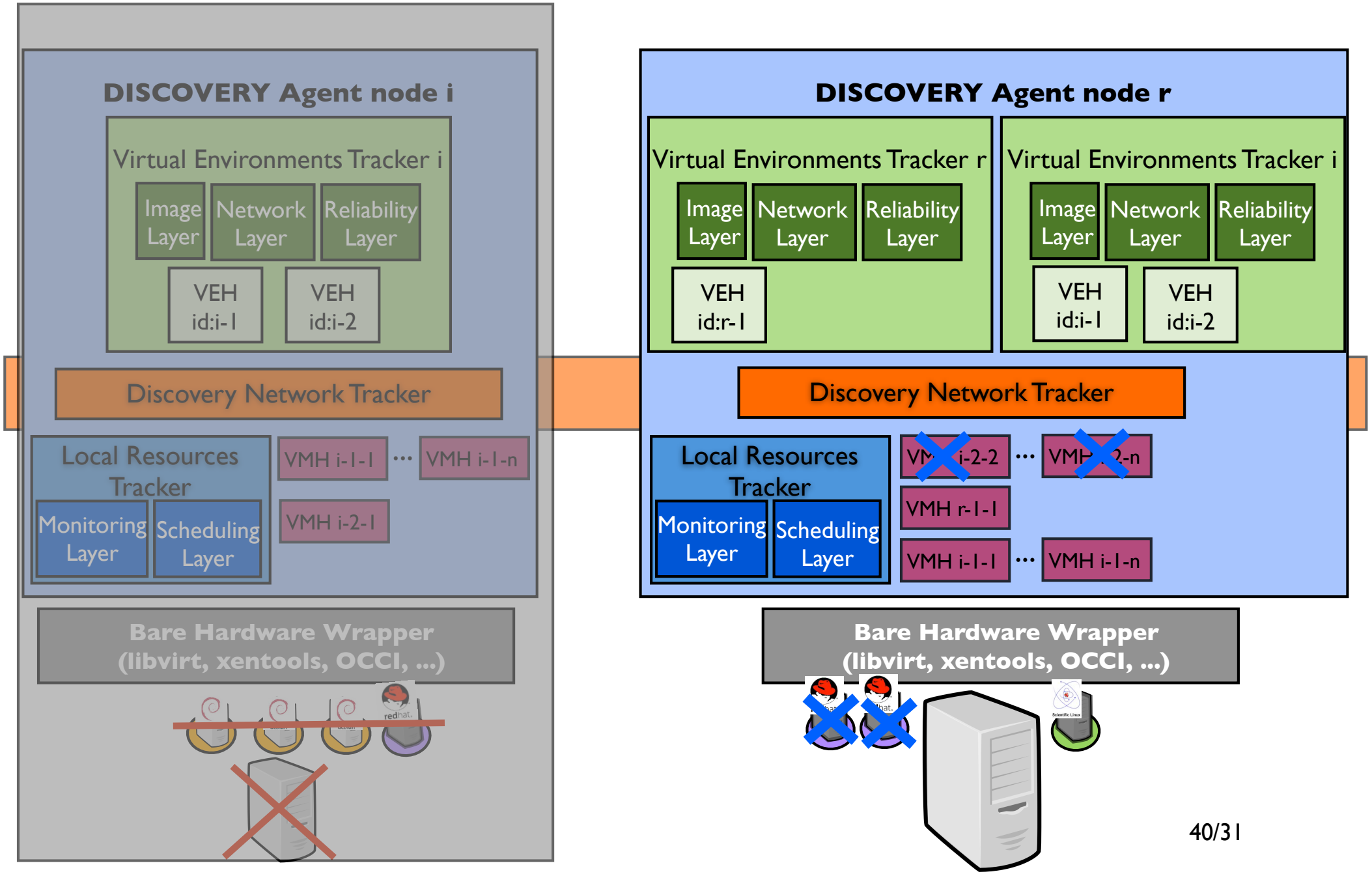
# DISCOVERY - Nodes Crashes



# DISCOVERY - Nodes Crashes



# DISCOVERY - Nodes Crashes



# DISCOVERY - Nodes Crashes