CLOUD COMPUTING SECURITY

HP Labs G-Cloud
A Secure Cloud Infrastructure

Frederic Gittler

Cloud and Security Laboratory, HP Labs

Covering…

- A few words about HP Labs
- An outline of Cloud Computing
  - Business drivers, Goals, etc.
- Cloud Security
  - Stakeholders and their issues
  - Routes of attack
  - Properties required
- Secure services
  - Control over security properties
- Infrastructure design
  - Virtual machines, networks and storage
  - Sensors and monitoring

HP LABS AROUND THE WORLD

– Global talent, local innovation

AMERICAS
- 46 university collaborations in the Americas
- Guadalajara Advanced Prototyping Center $1M
- 29 projects with HP Brazil R&D
- DARPA, DOE, US Army, MPO external funding
- NSF Post-Docs support $1M
- UC Discovery awards

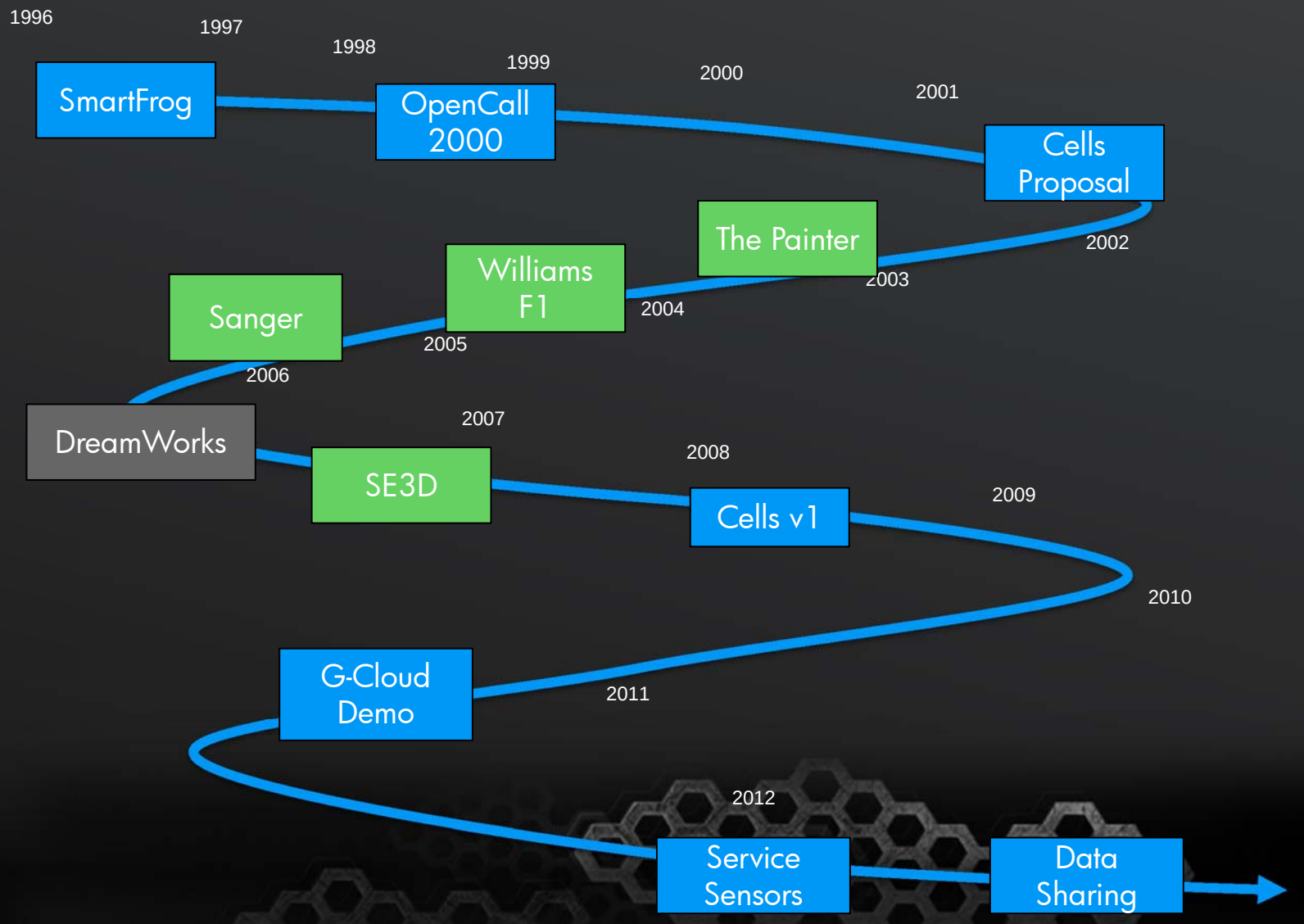
EMEA
- 10 university collaborations in EMEA
- 4 EU FP7 consortia, UK Tech Strategy Board awards
- UK CASE PhD support

APJ
- 7 university collaborations in the Asia-Pacific Region
- A*STAR and EDB support, Singapore

HP LABS RESEARCH AREAS

– Innovation at every touchpoint of information

A long history



Cloud Services

IT delivered as a logical service, available on demand, charged by usage

Logical Service: details of delivery hidden

On Demand: scales up and down immediately and seamlessly

Charged by Usage: metering and billing of services, pay for what you use

Cloud computing is **computation** offered as a service

Cloud computing Concepts

- Multi-tenancy
  - Shared service infrastructure, running simultaneous collocated workloads, for multiple customers
  - Dynamic sharing for flexibility and utilisation efficiency
- Cloud-scale infrastructure
  - Commodity scale-out hardware
  - Targeting extreme economies of scale
- Ubiquitous access
  - Using portable client devices
  - Any where, any time

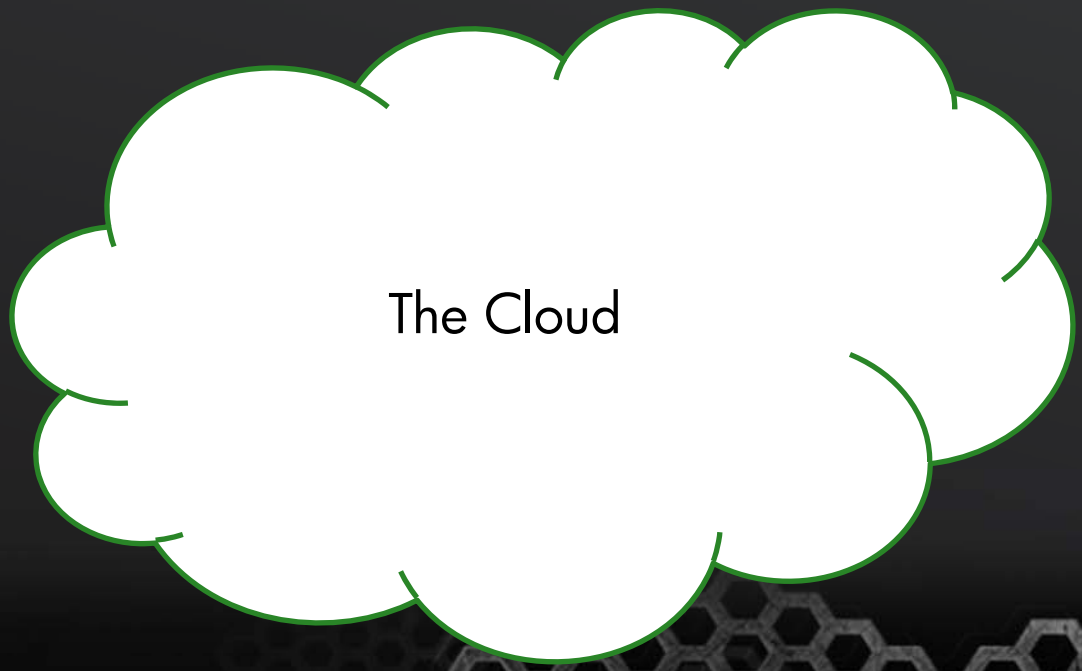
Cloud Computing benefits

- **Cost management**
  - Benefit from economies of scale
  - Avoids cost of over-provisioning
  - Reduction in up-front capital investment, switch to expense more in line with business needs
- **Risk reduction**
  - Someone else worries about running the data-centre, protecting your data, and providing disaster recovery
  - Reduces risk of under-provisioning
- **Flexibility**
  - Add/remove services on demand
  - Scale up and down as needed – rapidly
- **Ubiquity**
  - Access from any place, any device, any time

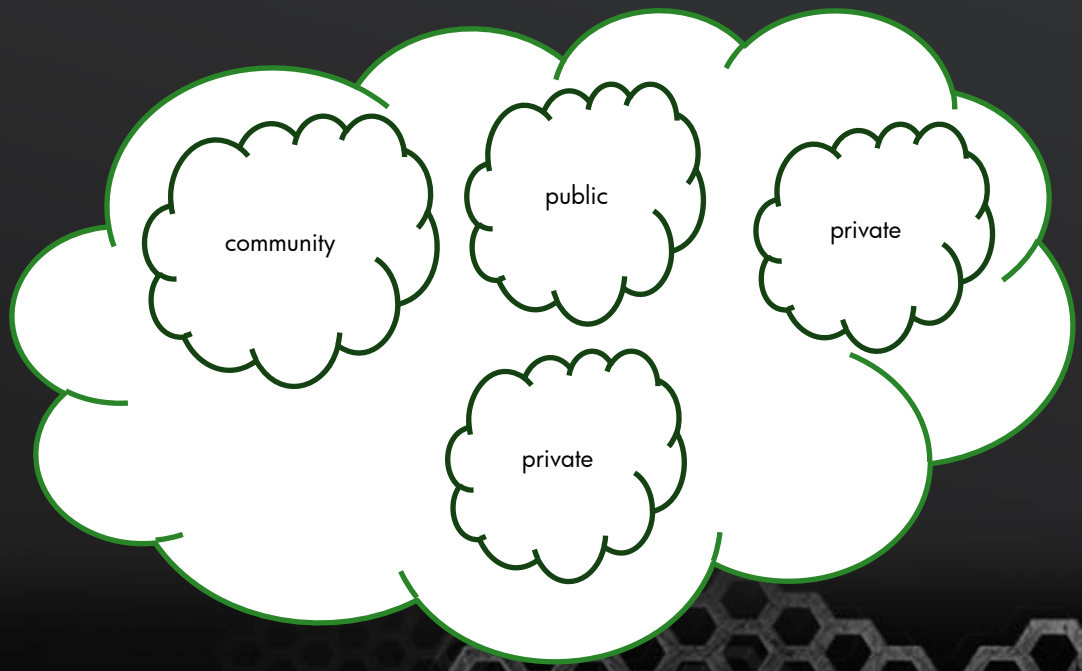
Barriers to adoption

- **Security, Regulatory, Data locality concerns**
- Concerns about lock-in, lack of multi-vendor options
- Challenge of migrating from in-house (or outsourced) apps
- Trust in the service vendors
  - Service levels
  - Stability
  - Geographic presence
- Vested Interests

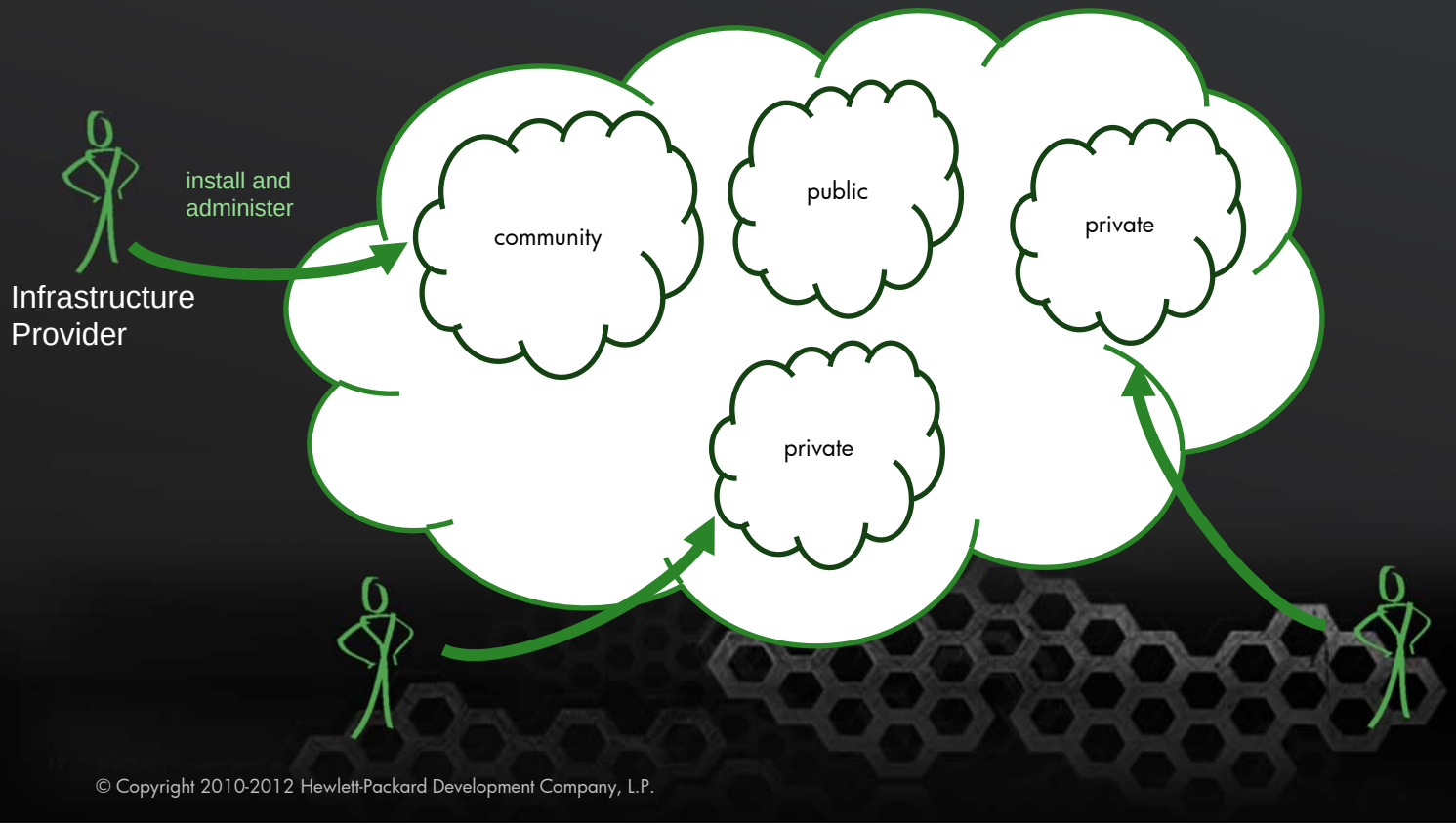
Roles in the Cloud



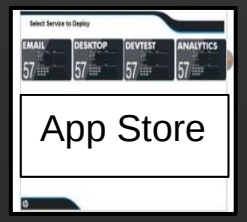
Roles in the Cloud



Roles in the Cloud

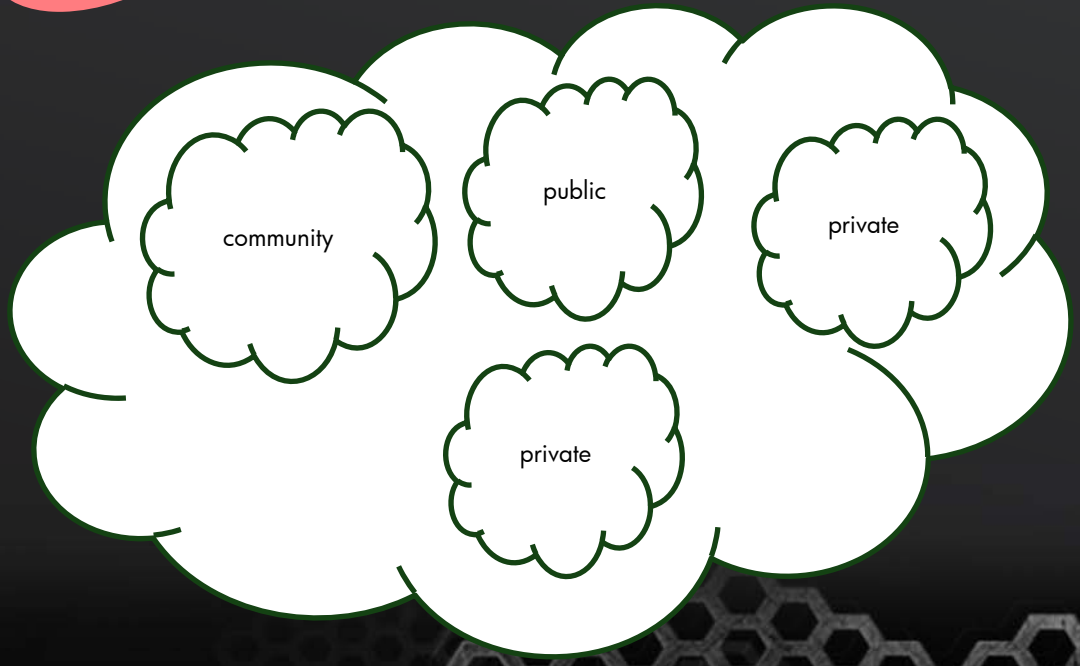


Roles in the Cloud



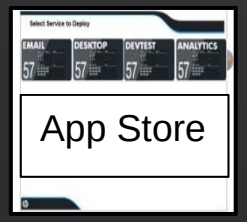
Service Developer

Infrastructure Provider



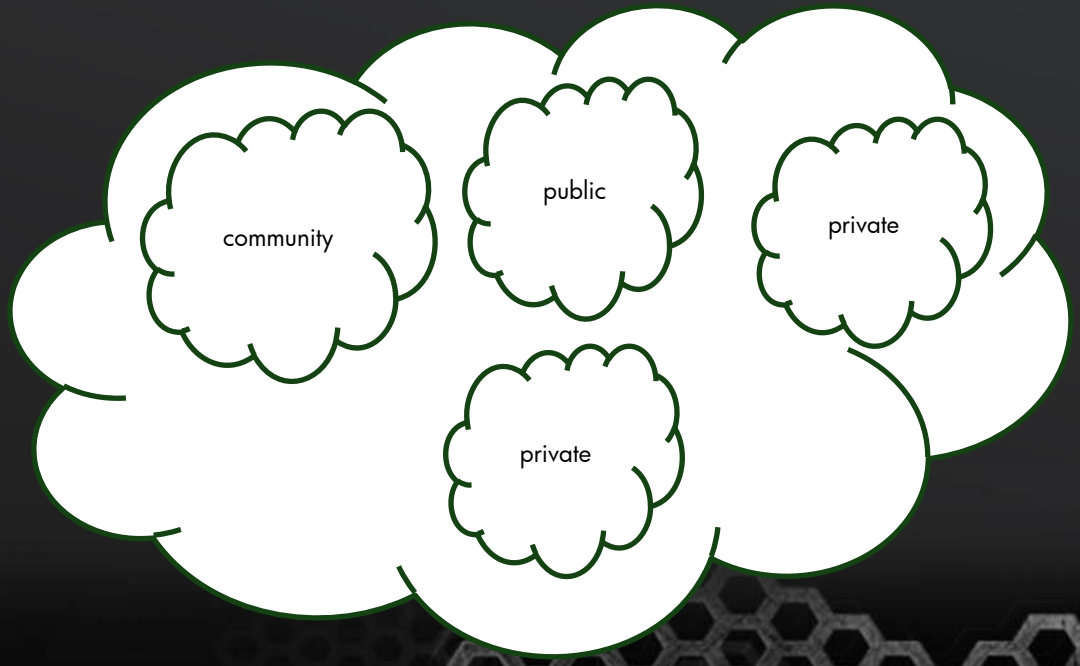
Roles in the Cloud

Service Provider

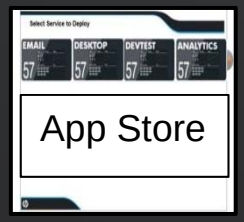


Service Developer

Infrastructure Provider



Roles in the Cloud

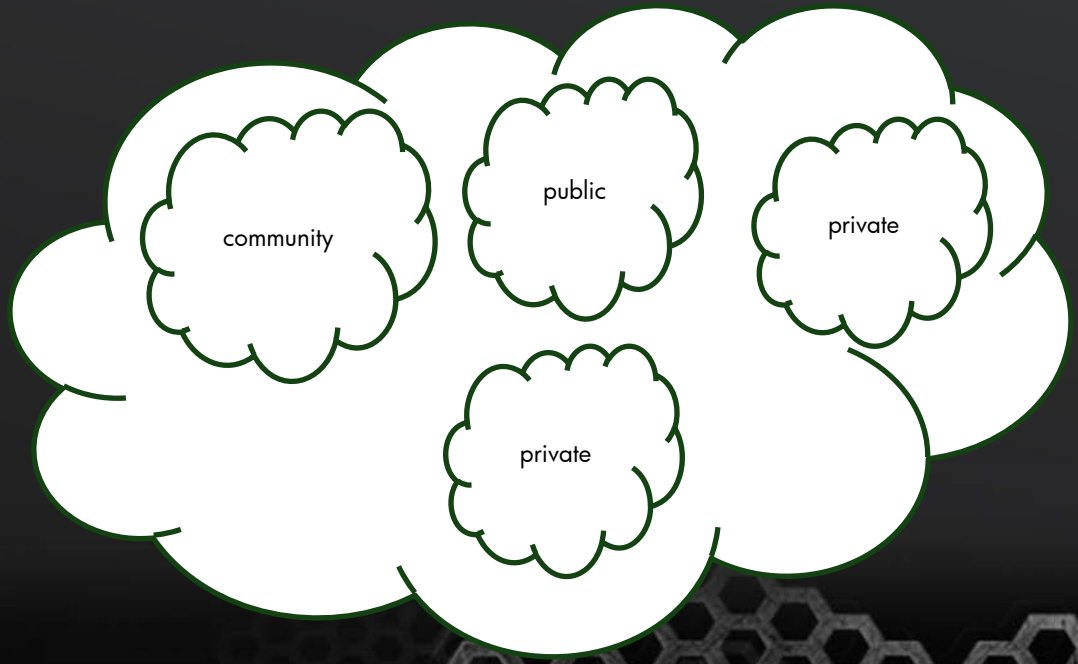


procure

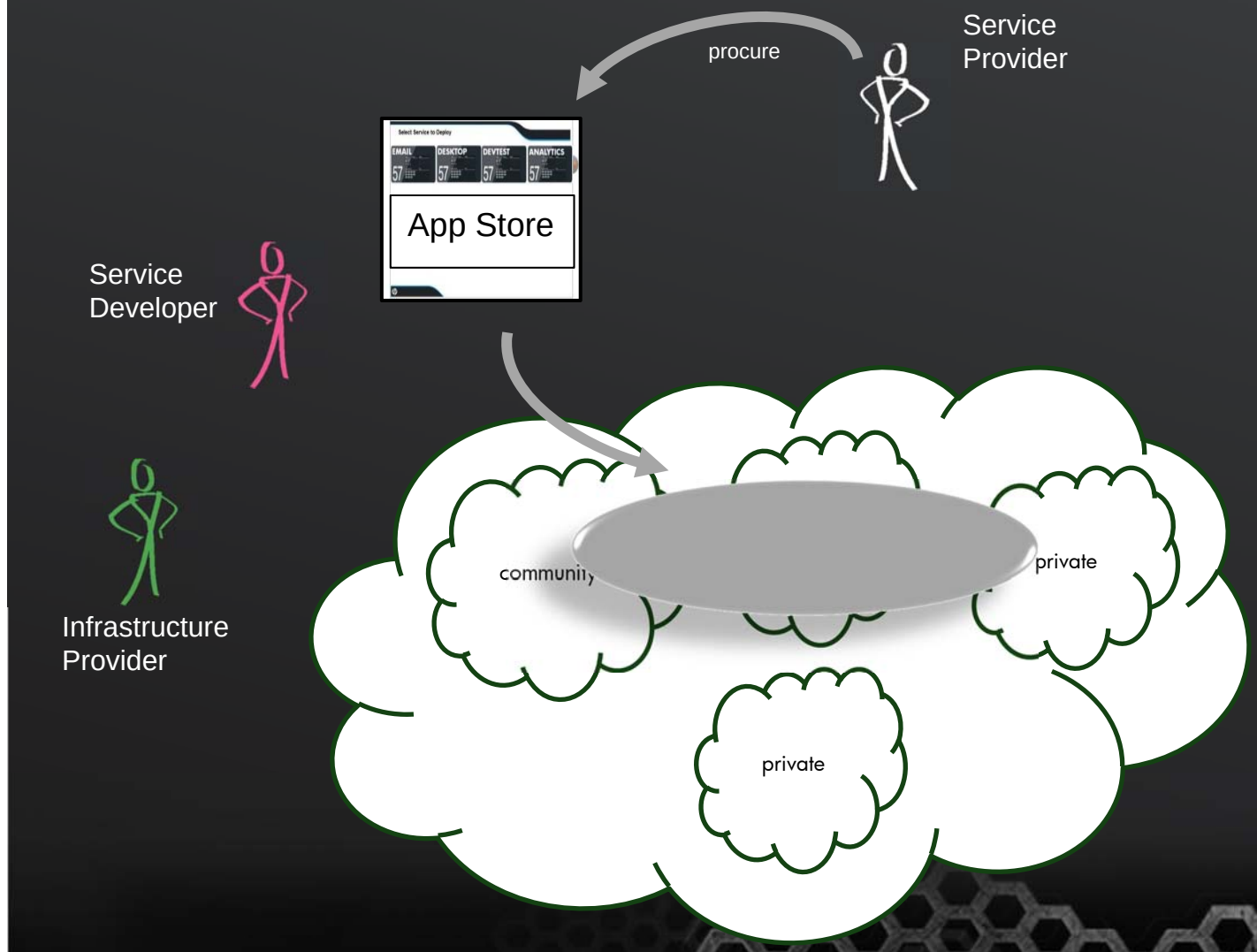
Service Provider

Service Developer

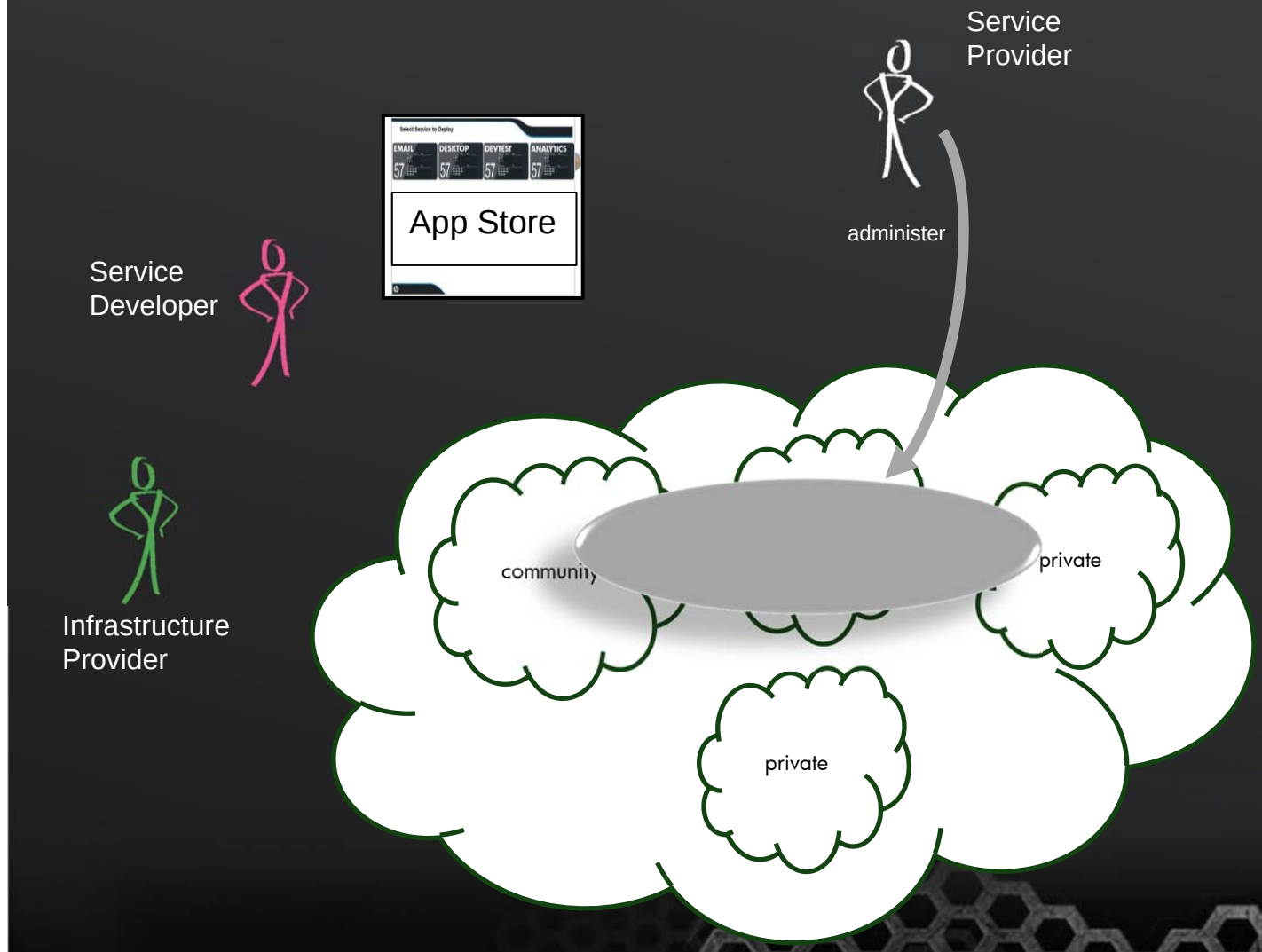
Infrastructure Provider



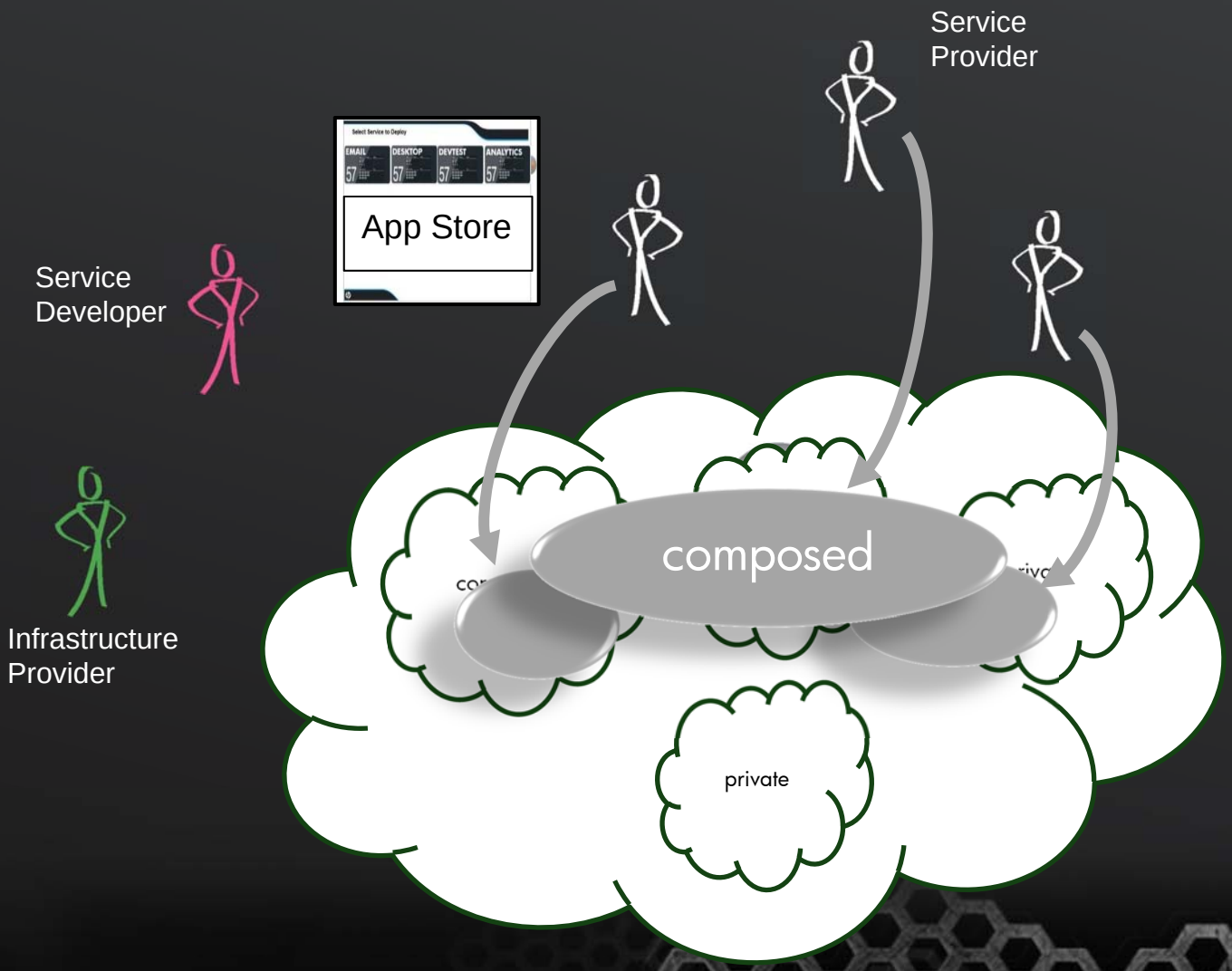
Roles in the Cloud



Roles in the Cloud

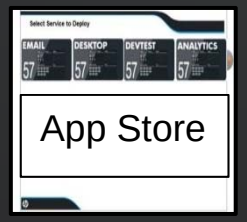


Roles in the Cloud



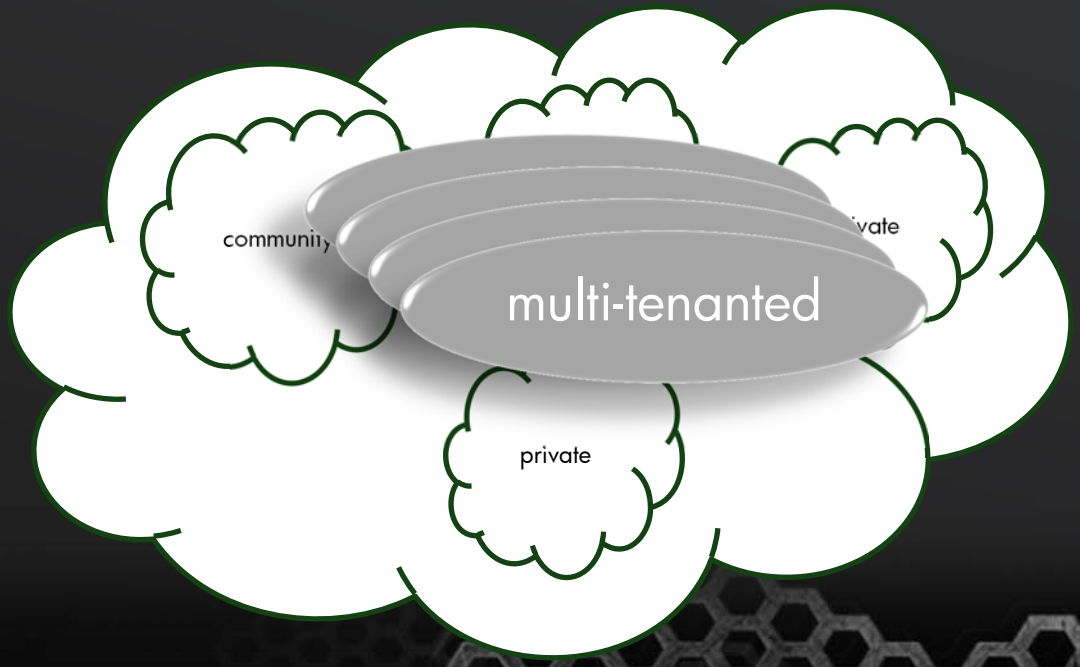
Roles in the Cloud

Service Provider



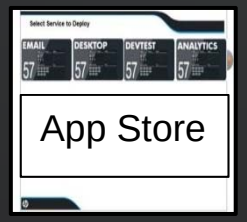
Service Developer

Infrastructure Provider



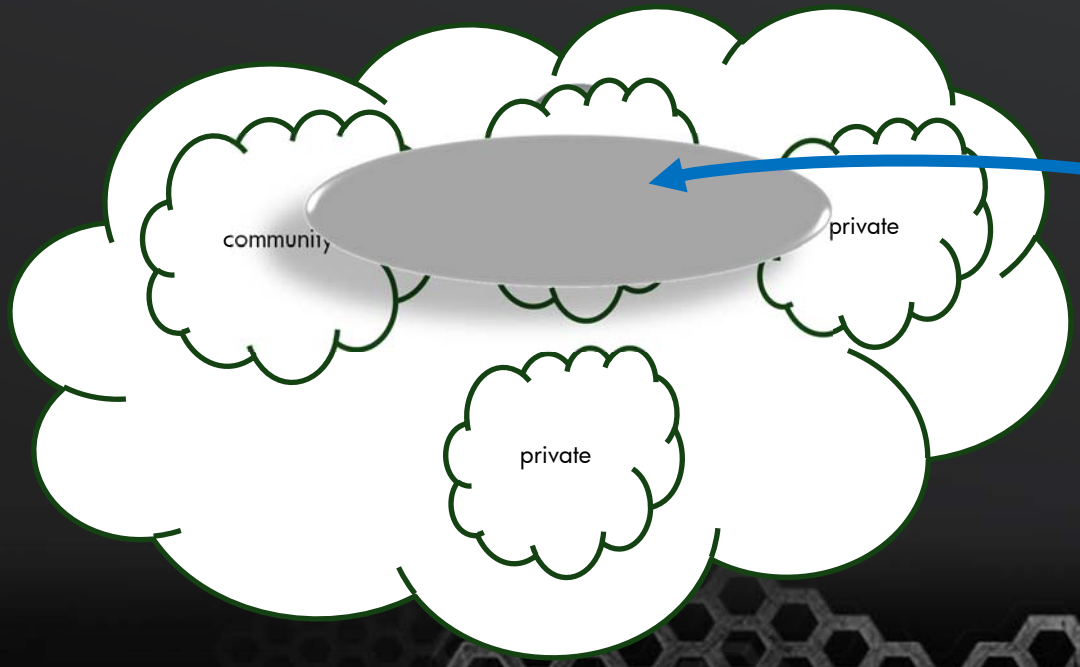
Roles in the Cloud

Service Provider



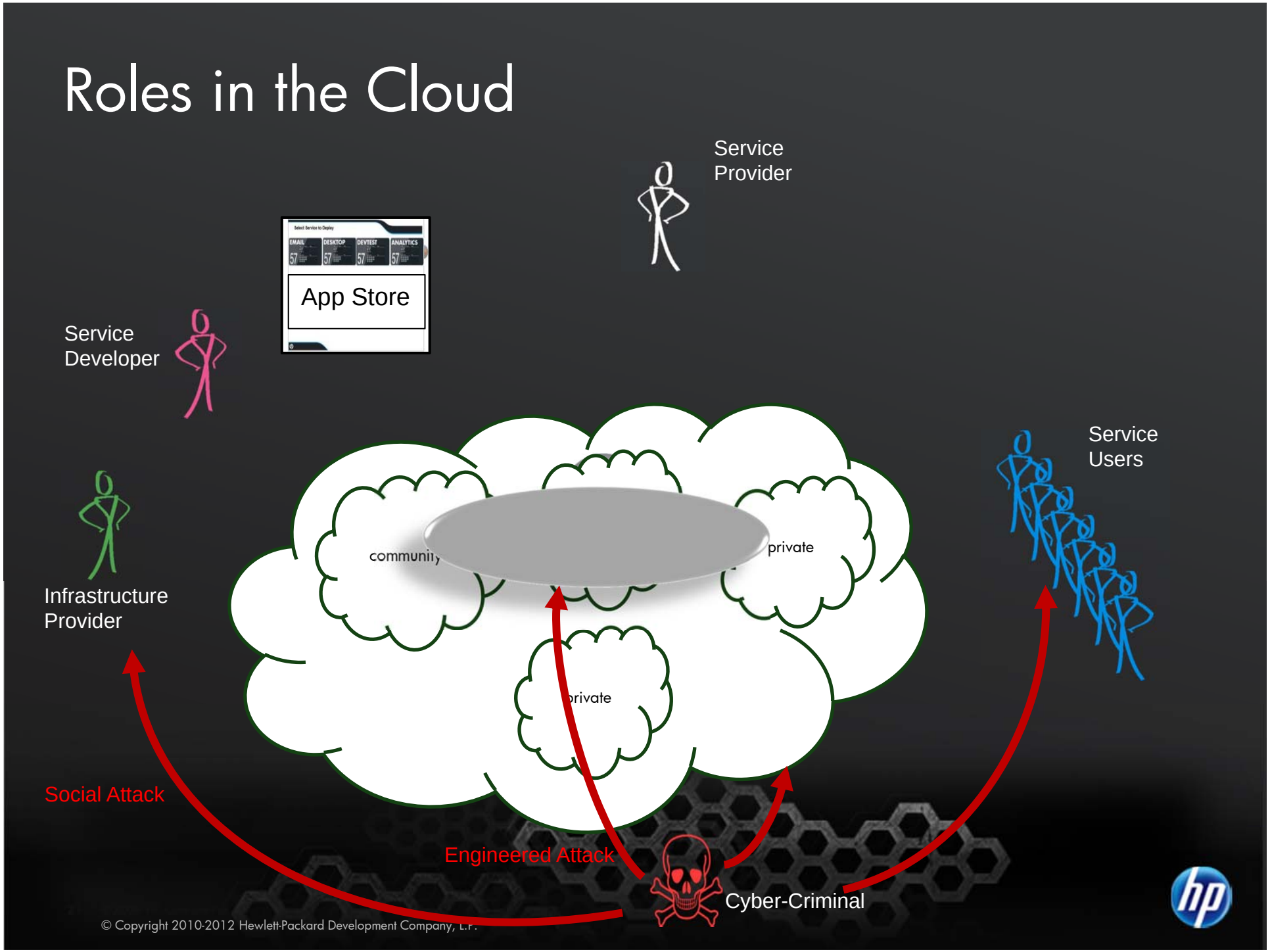
Service Developer

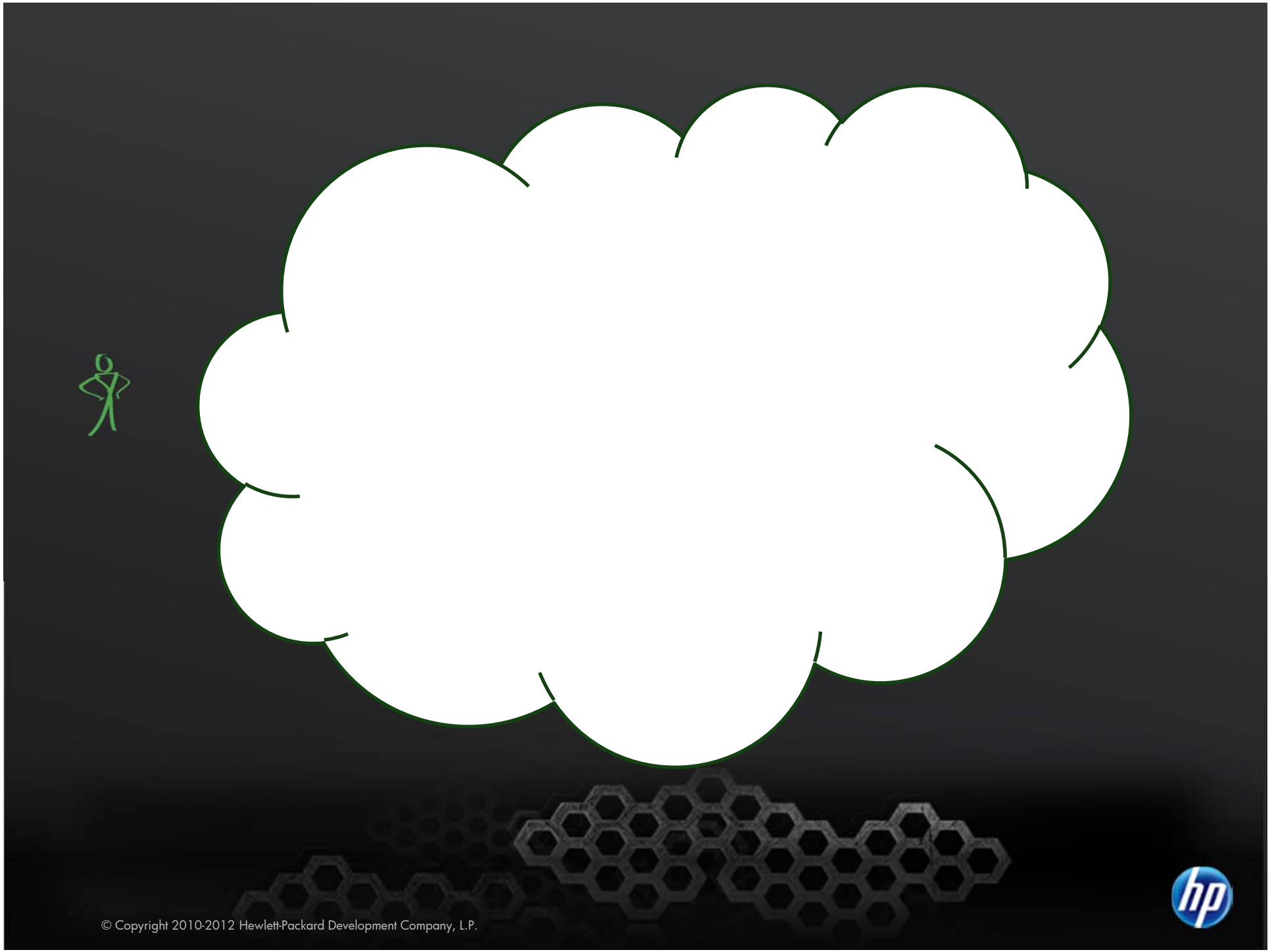
Infrastructure Provider

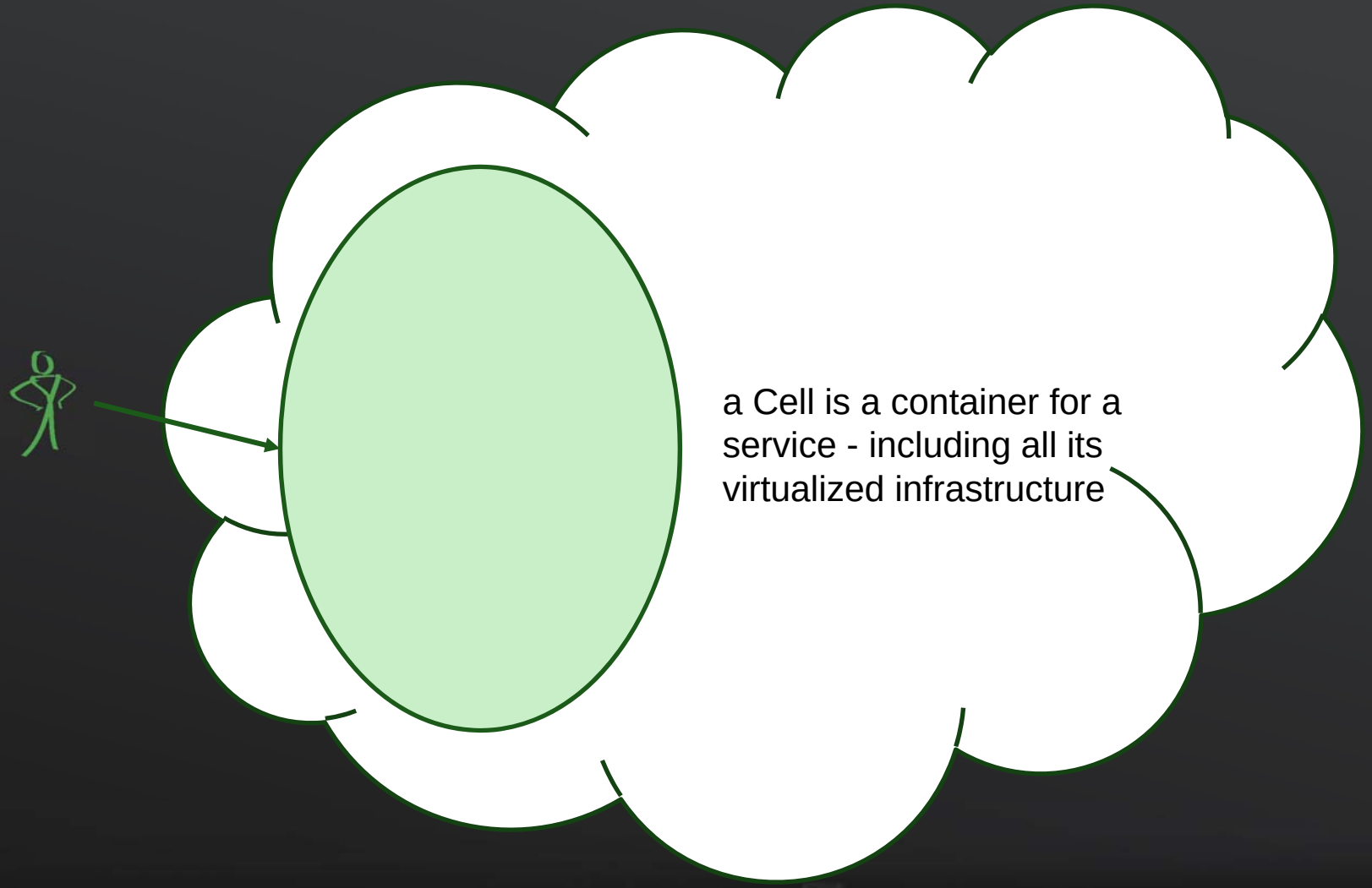


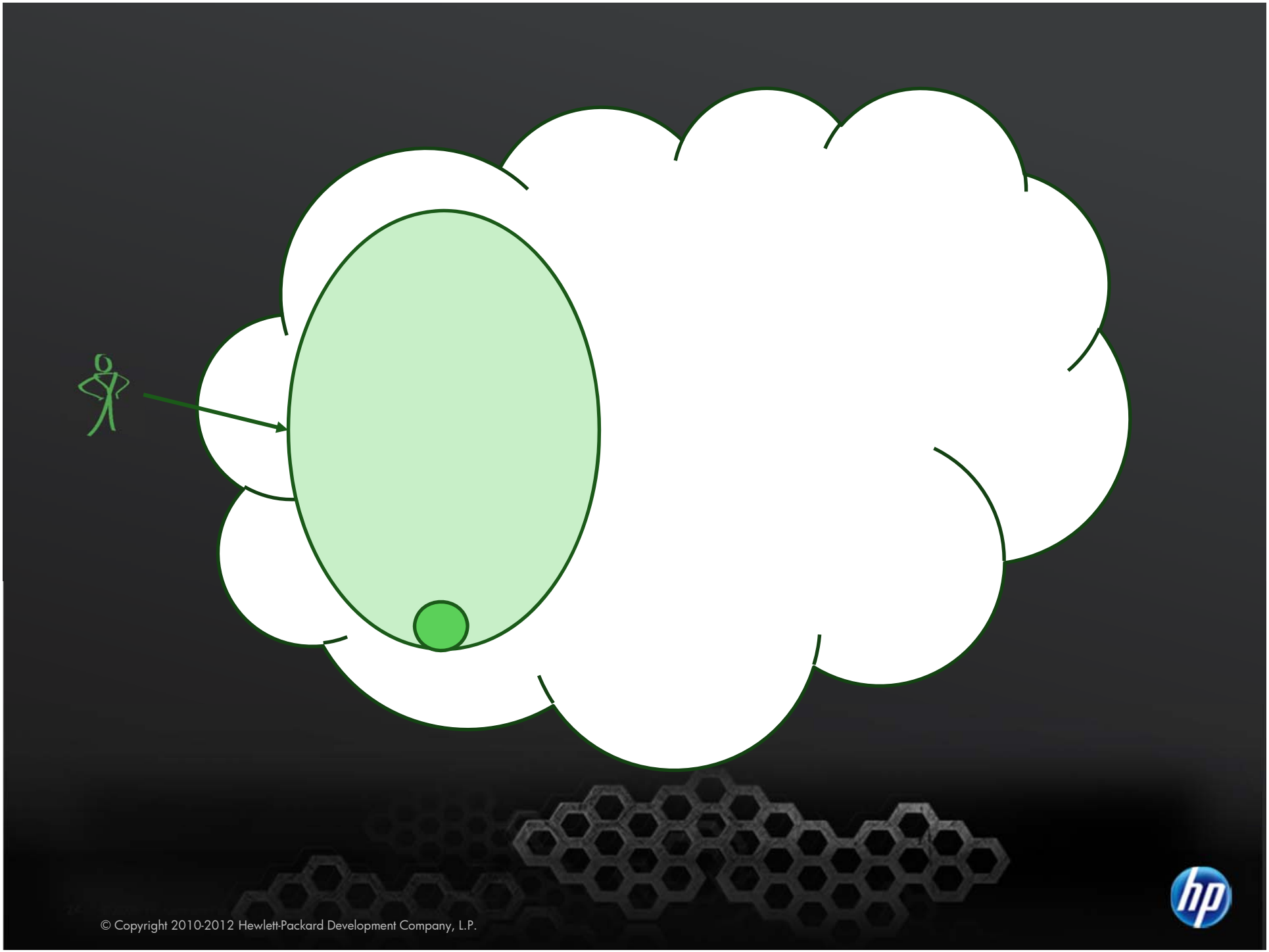
Service Users

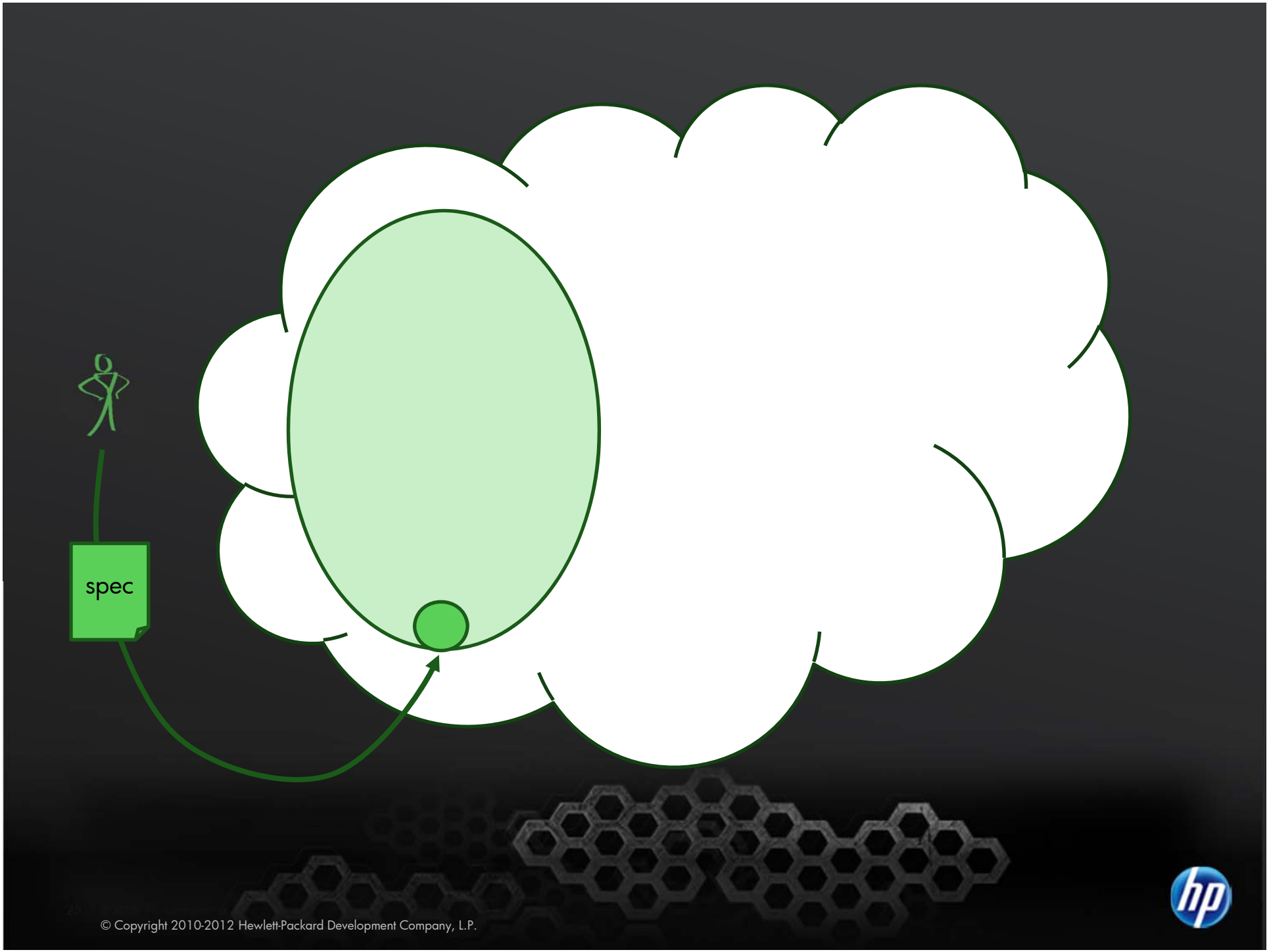
Roles in the Cloud

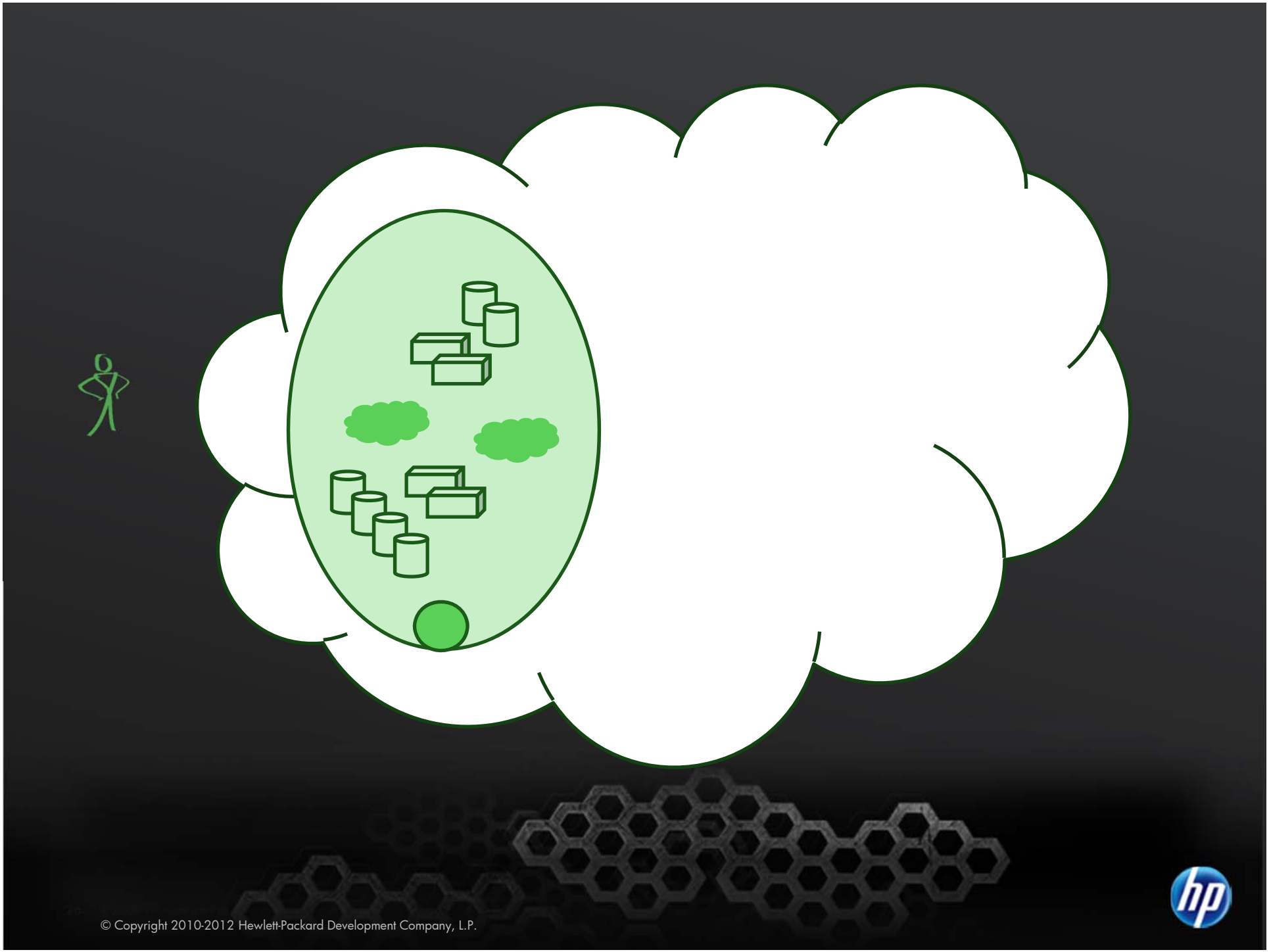


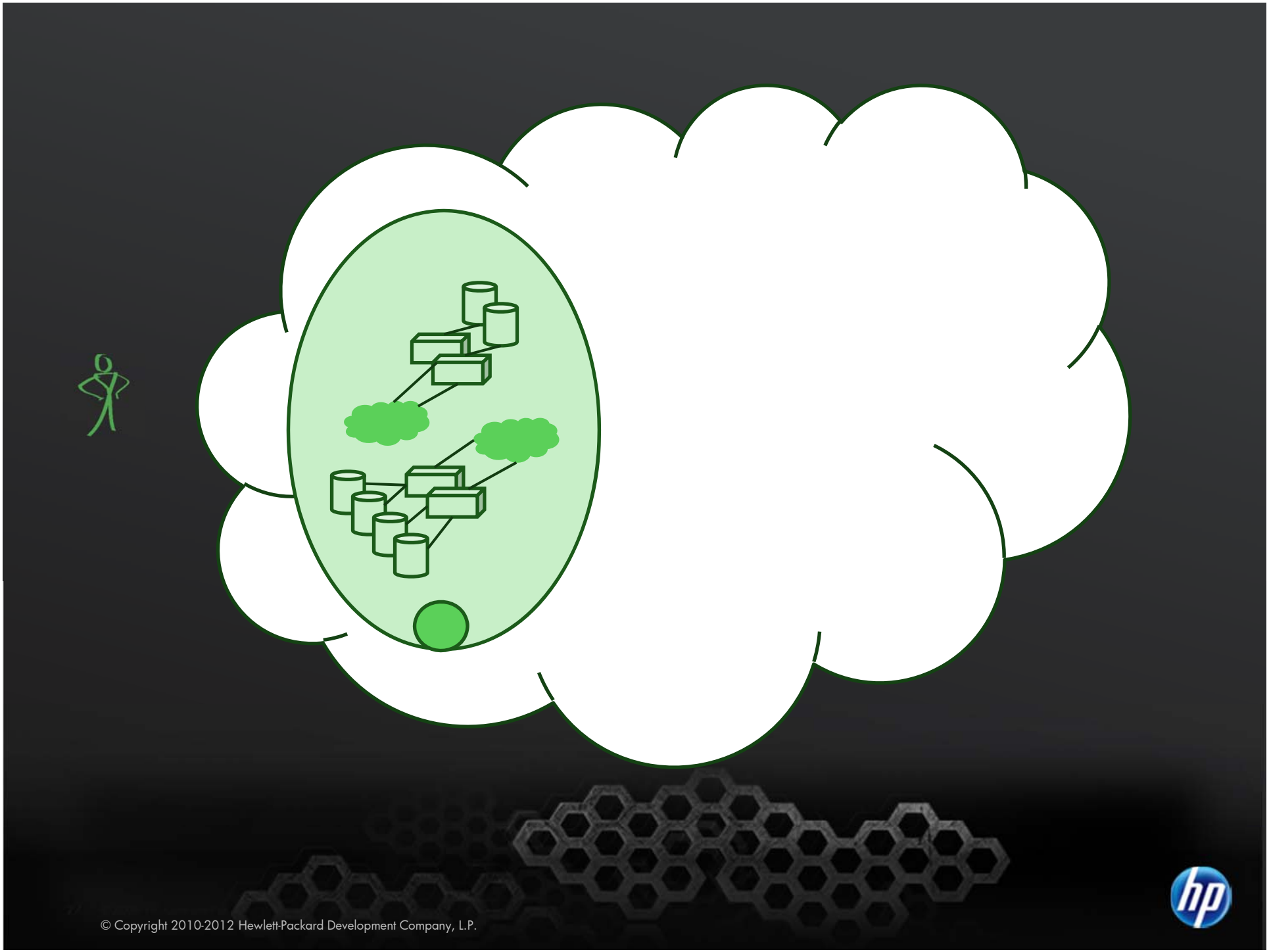


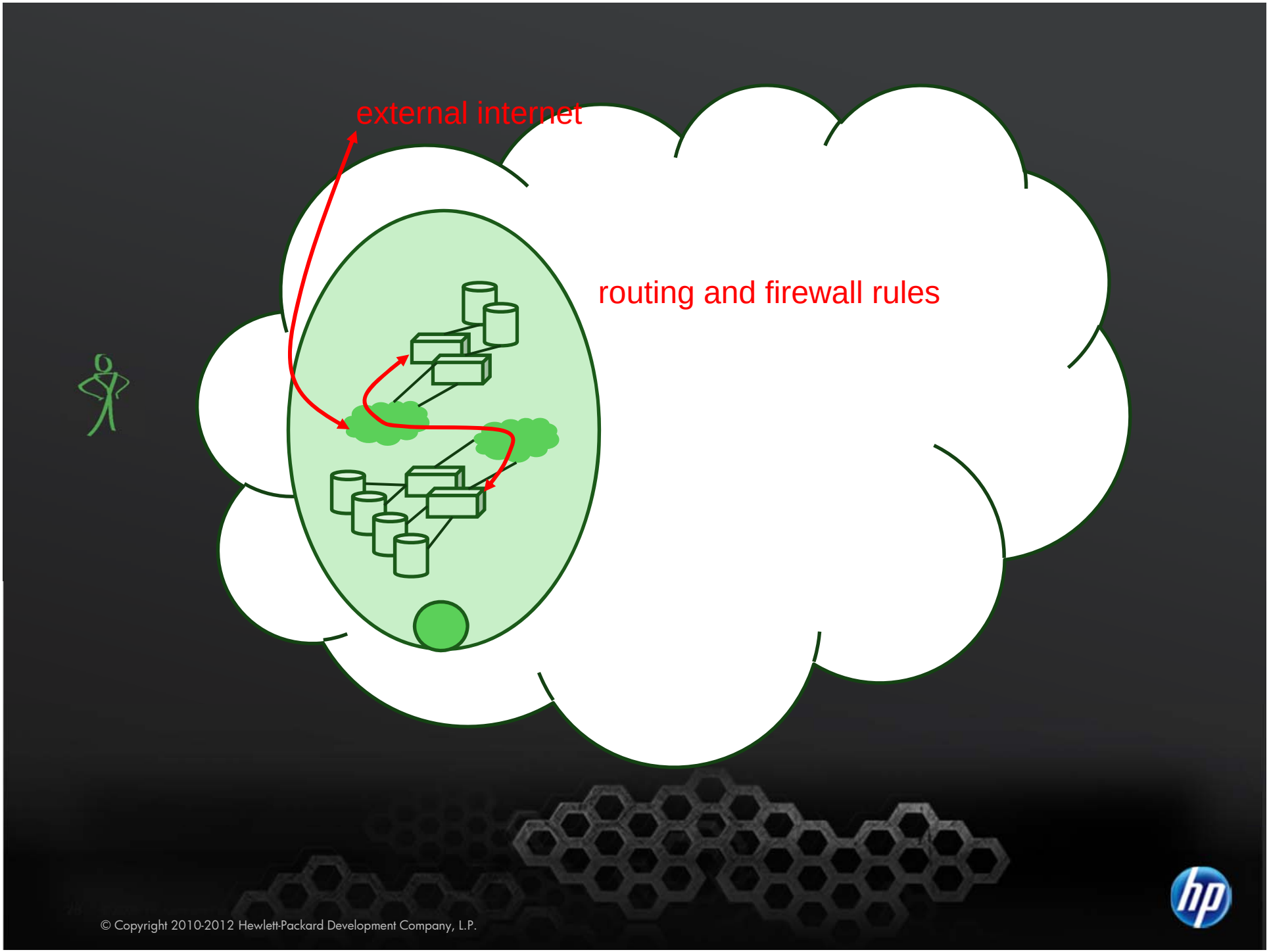


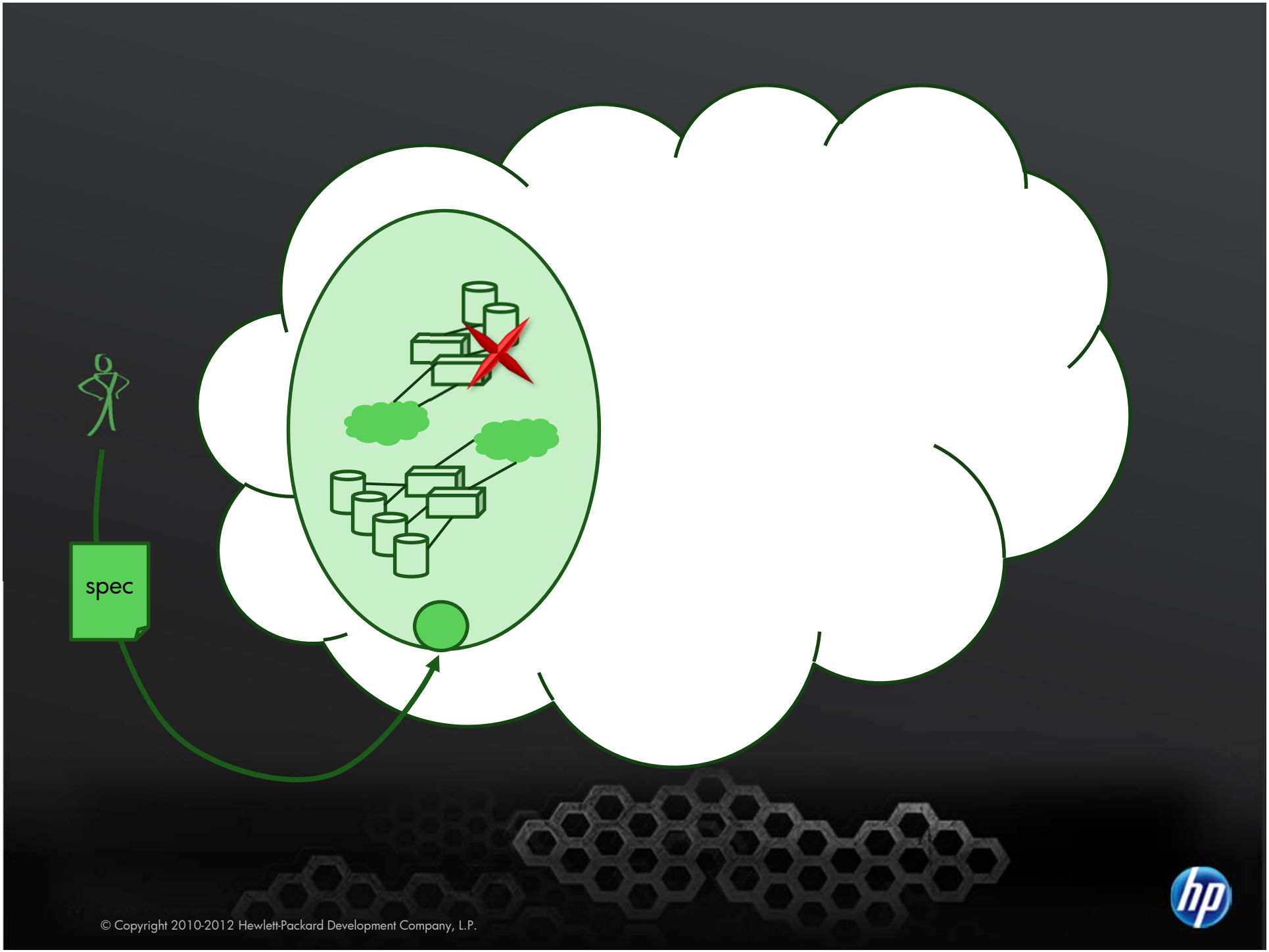


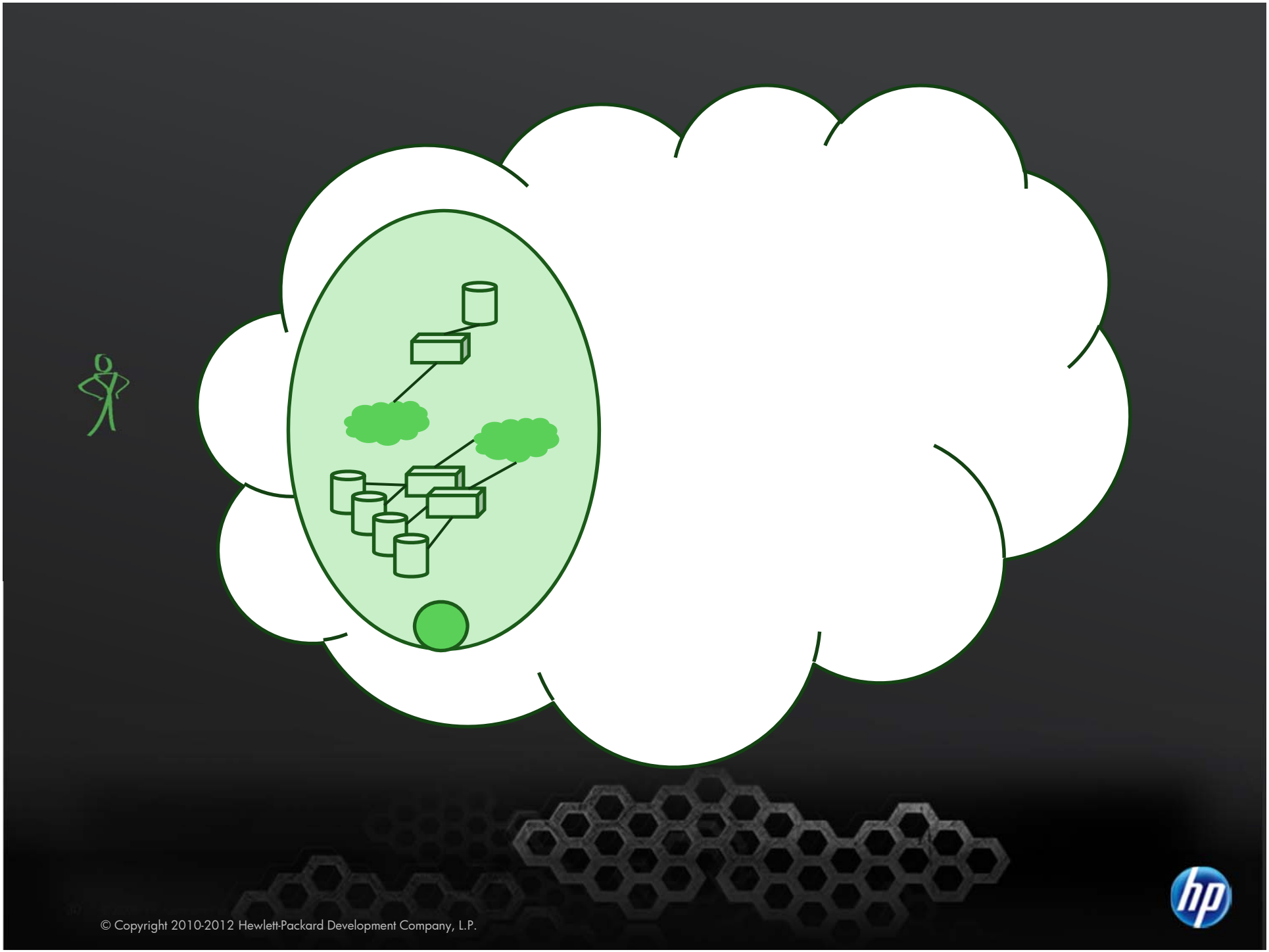


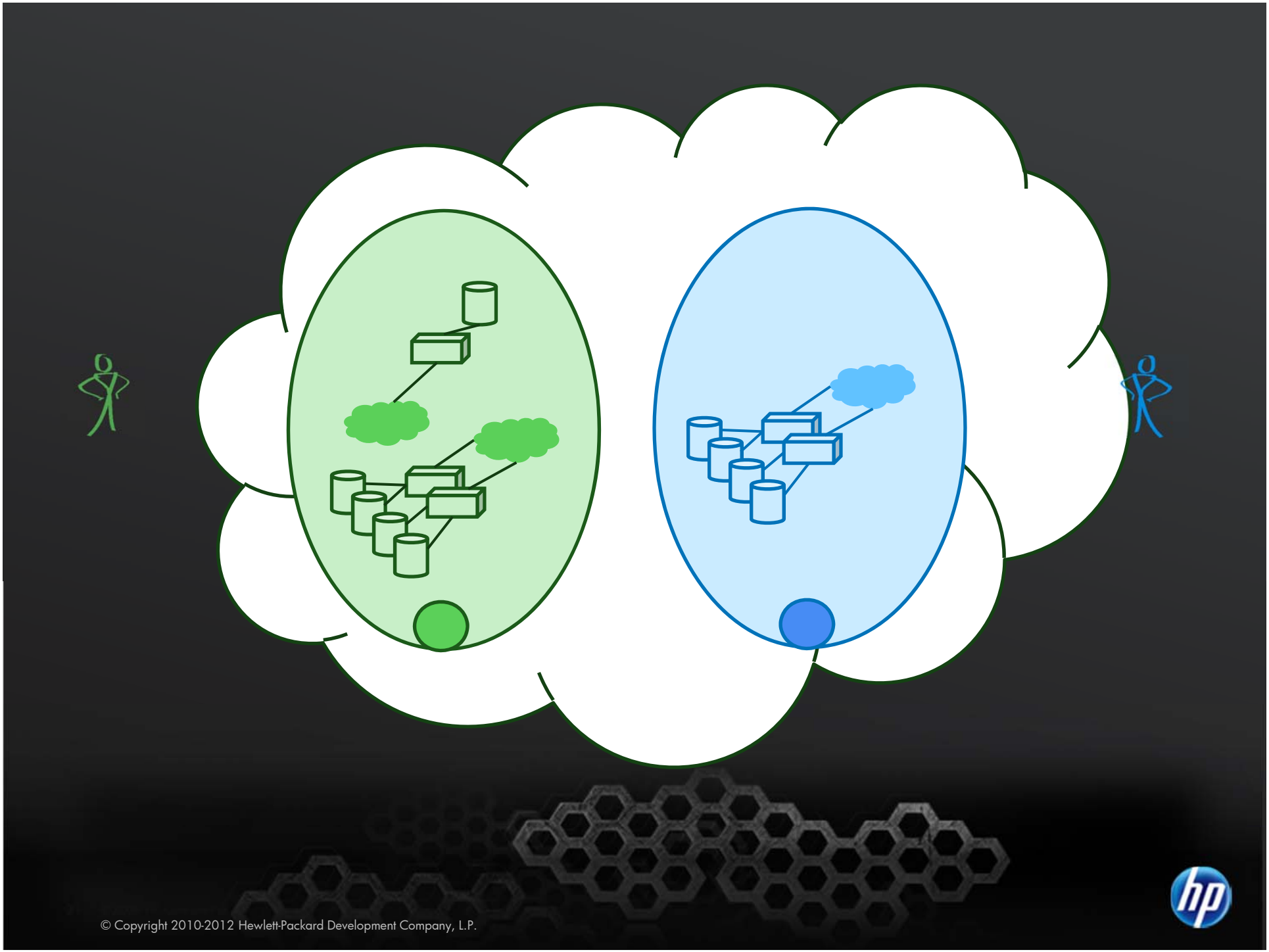


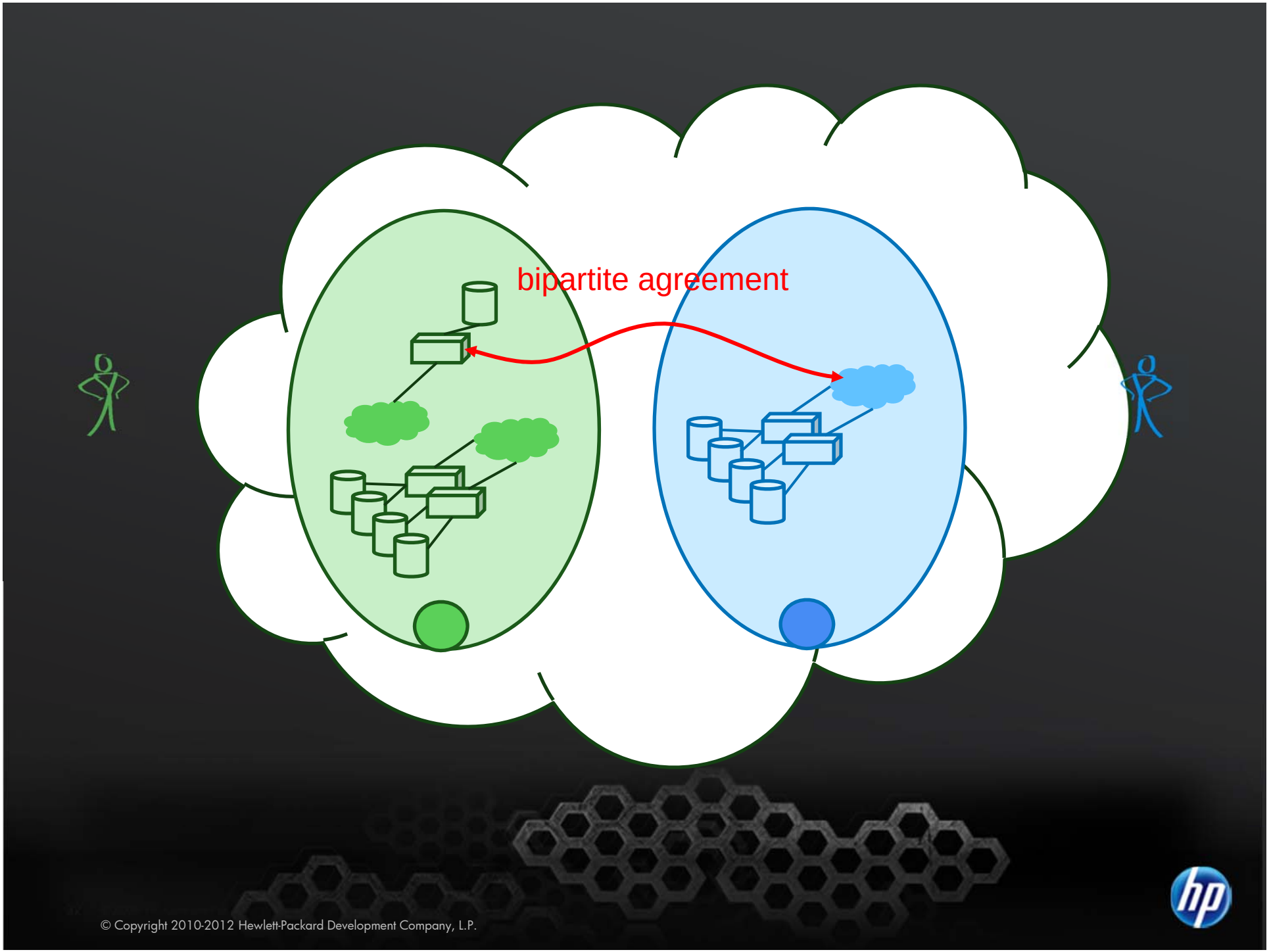






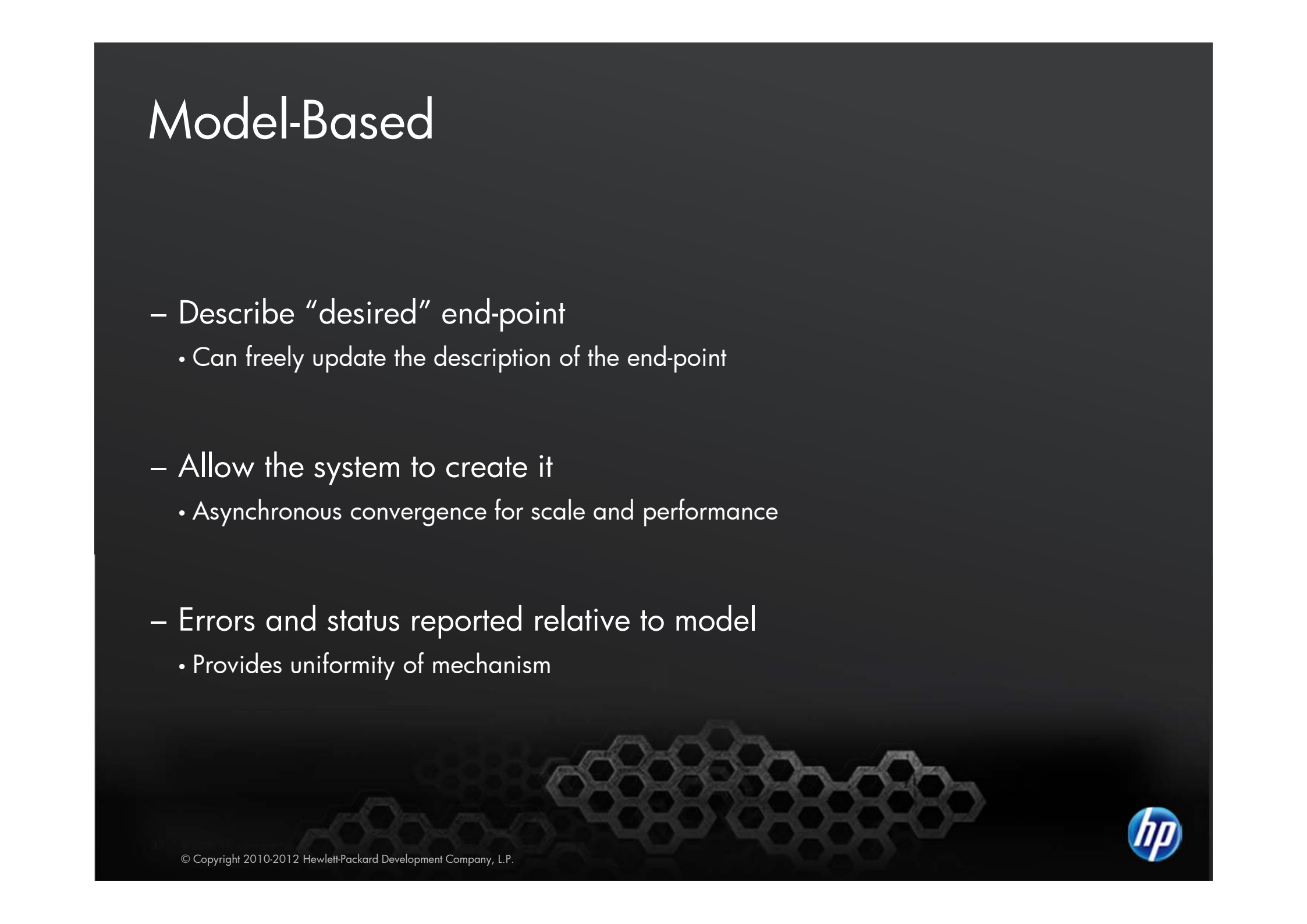






Model-Based

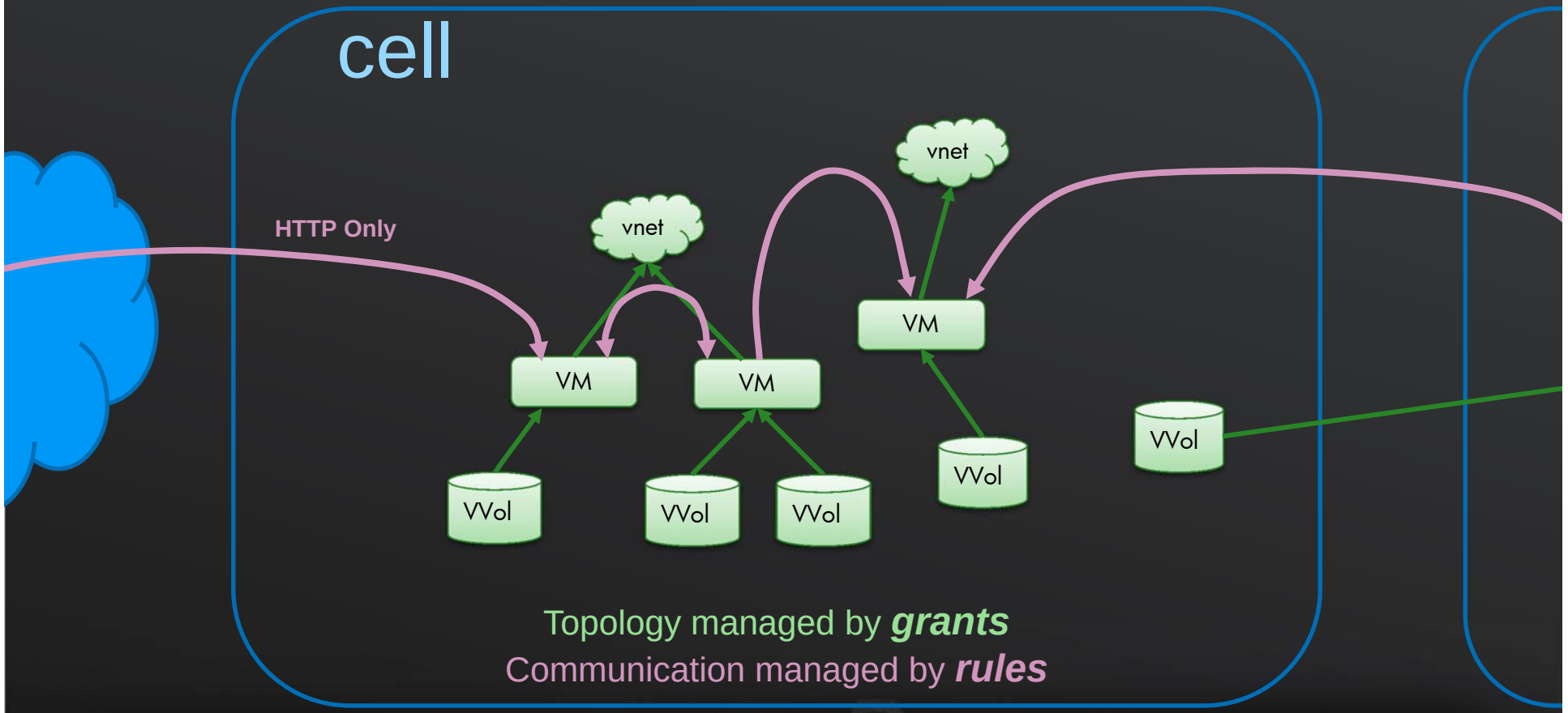
- Describe "desired" end-point
  - Can freely update the description of the end-point
- Allow the system to create it
  - Asynchronous convergence for scale and performance
- Errors and status reported relative to model
  - Provides uniformity of mechanism



Model-Based: Why

- Declarative
  - Enables analysis and tool support
  - Cross-model properties and policies
  - Basis for compliance and transparent management
  - Enables different principals to sign different parts of the model, independent of right of model submission, and mapping into enterprise IT roles
  - Enables IT best practice
- Inherently idempotent
  - Hugely simplifies interaction model, improves security
- Enables back-end asynchrony and parallelism for scale and performance
- Template descriptions of services
  - Simplifies service packaging
  - Ease of integration with a Cloud Marketplace
- Easy to map transactional interfaces to model-based; hard to do the other way around and maintain the advantages

Infrastructure Models

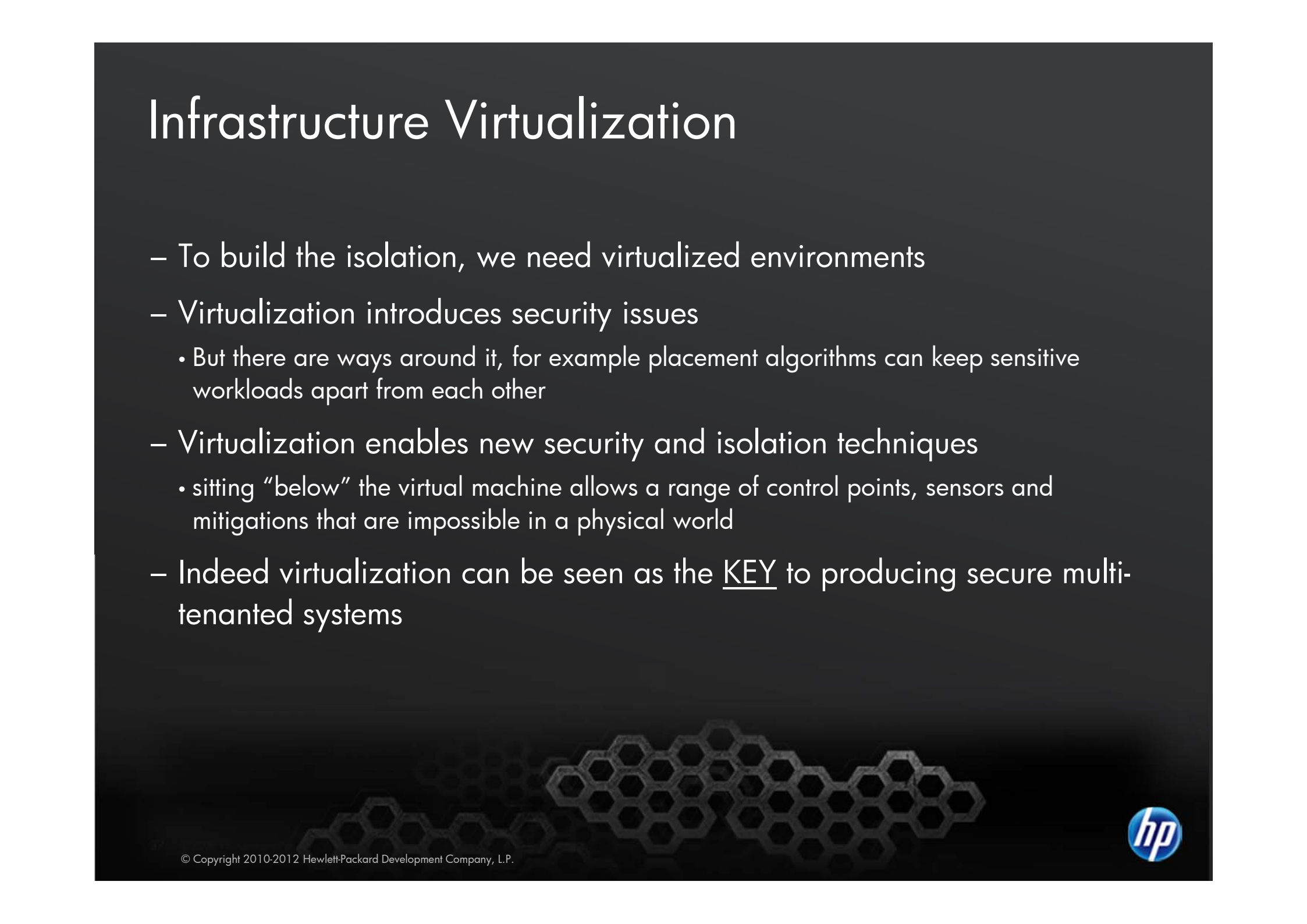


Future Modelling

- Currently we provide explicit description of topology and some security properties
- However, this is but the start….
  - Support for loose models and constraints
    - Models that are configured according to user or business-level concepts
  - Order-dependencies
    - Declarative description of state transitions and dependencies on the state
  - Specification of QoS policies
    - linked through sensor framework, constraints and dependencies, to create auto-flexing models
  - Specification of service high-availability policies
    - Automatically mapped into placement and recovery decisions
    - Federation properties
  - Specification of additional security policies
    - Guiding placement and other aspects such as providing “security probes”
    - Semantic controls over data sharing

Infrastructure Virtualization

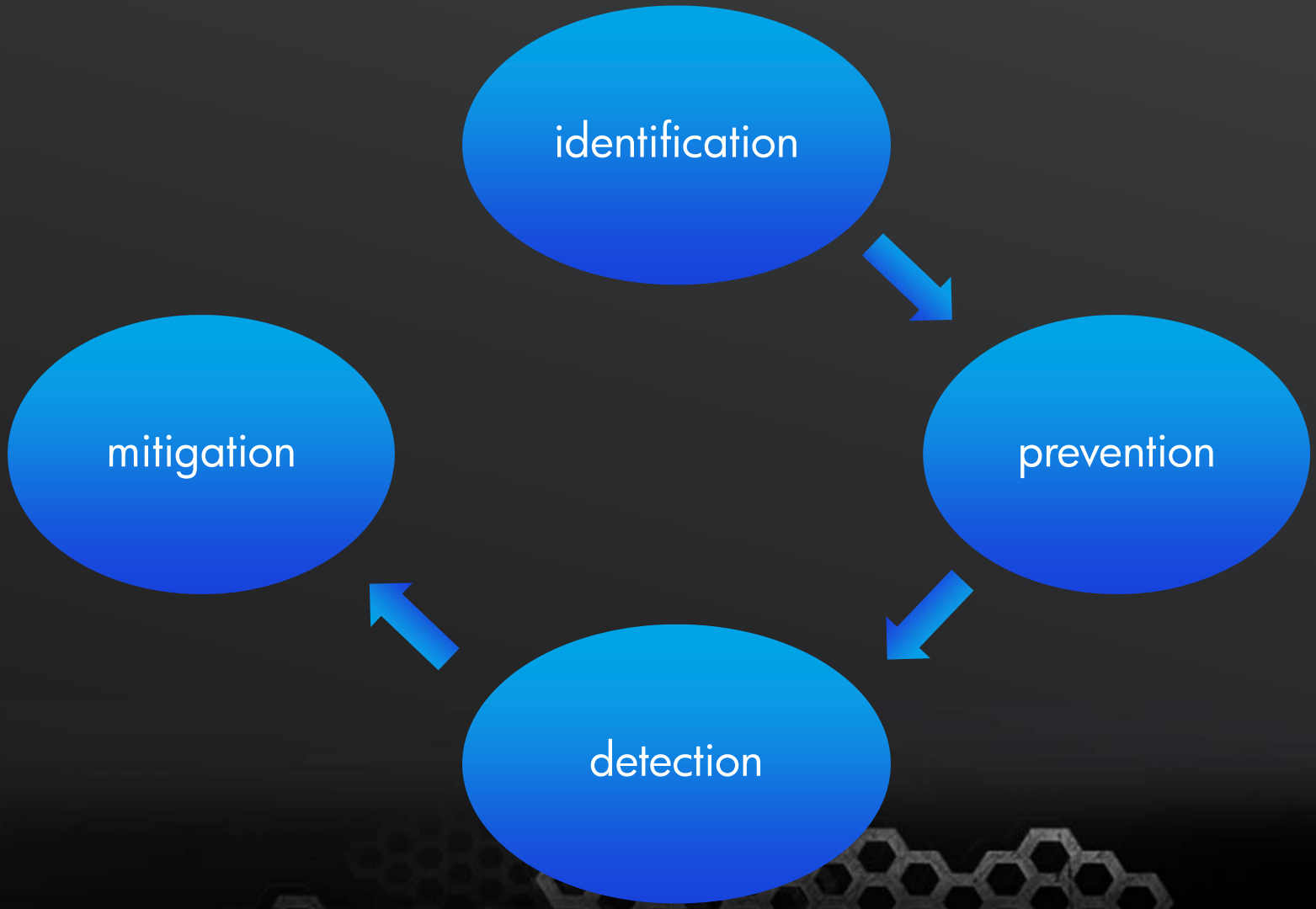
- To build the isolation, we need virtualized environments
- Virtualization introduces security issues
  - But there are ways around it, for example placement algorithms can keep sensitive workloads apart from each other
- Virtualization enables new security and isolation techniques
  - sitting "below" the virtual machine allows a range of control points, sensors and mitigations that are impossible in a physical world
- Indeed virtualization can be seen as the KEY to producing secure multi-tenanted systems

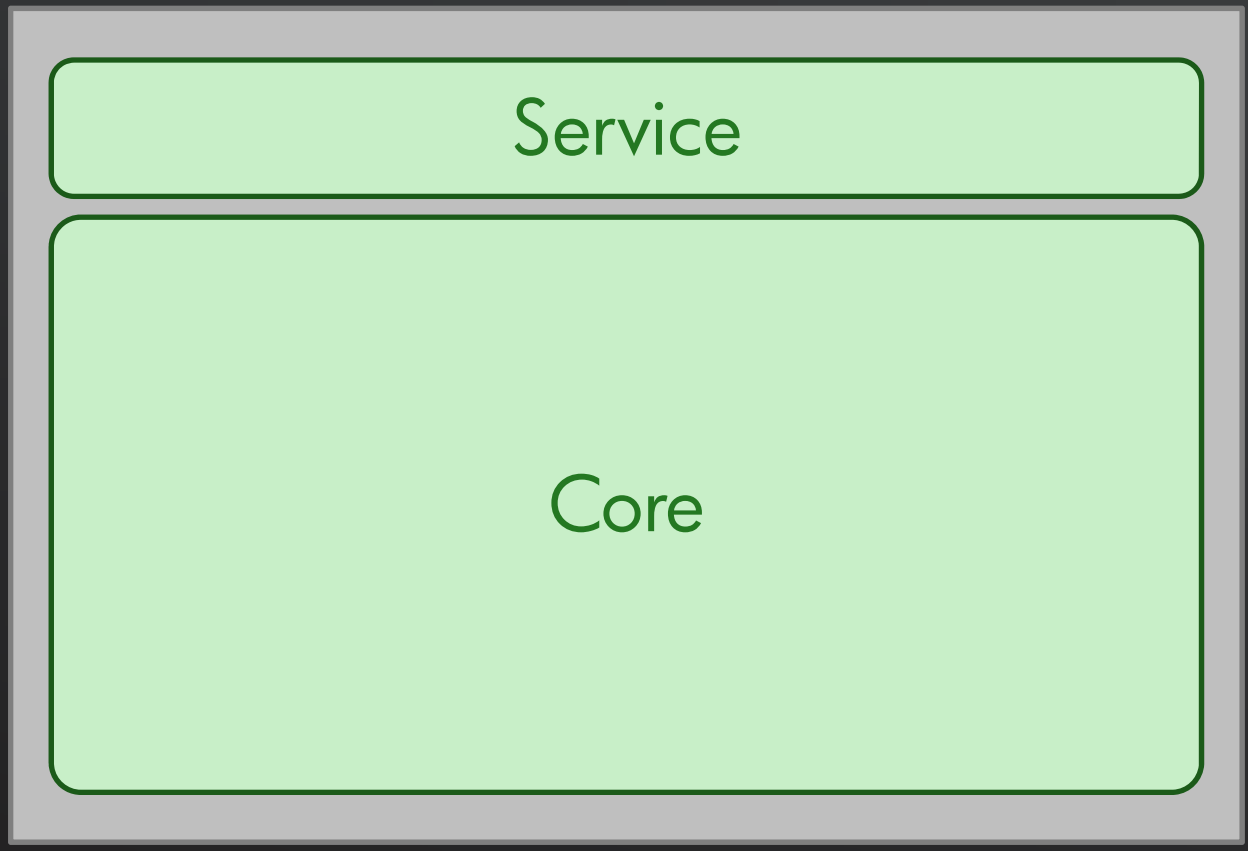


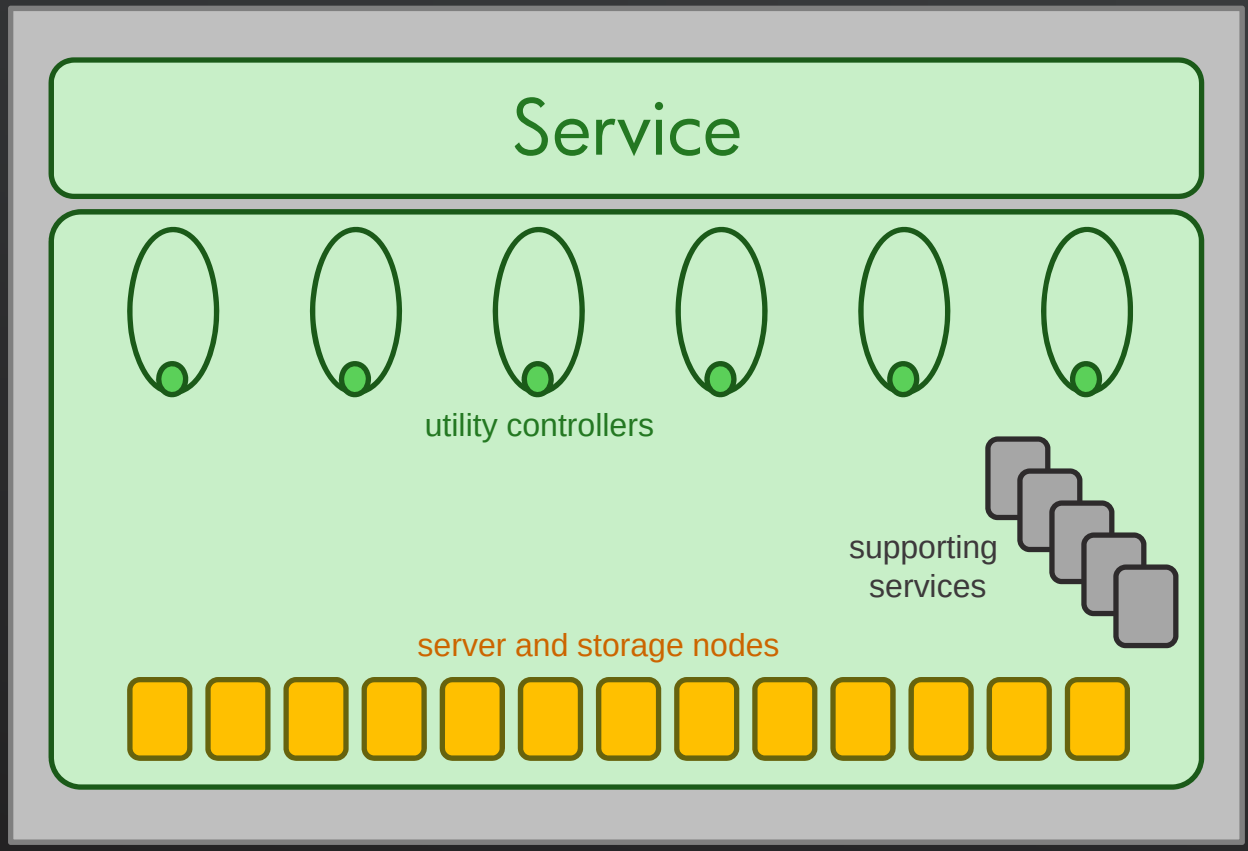
Infrastructure Integrity

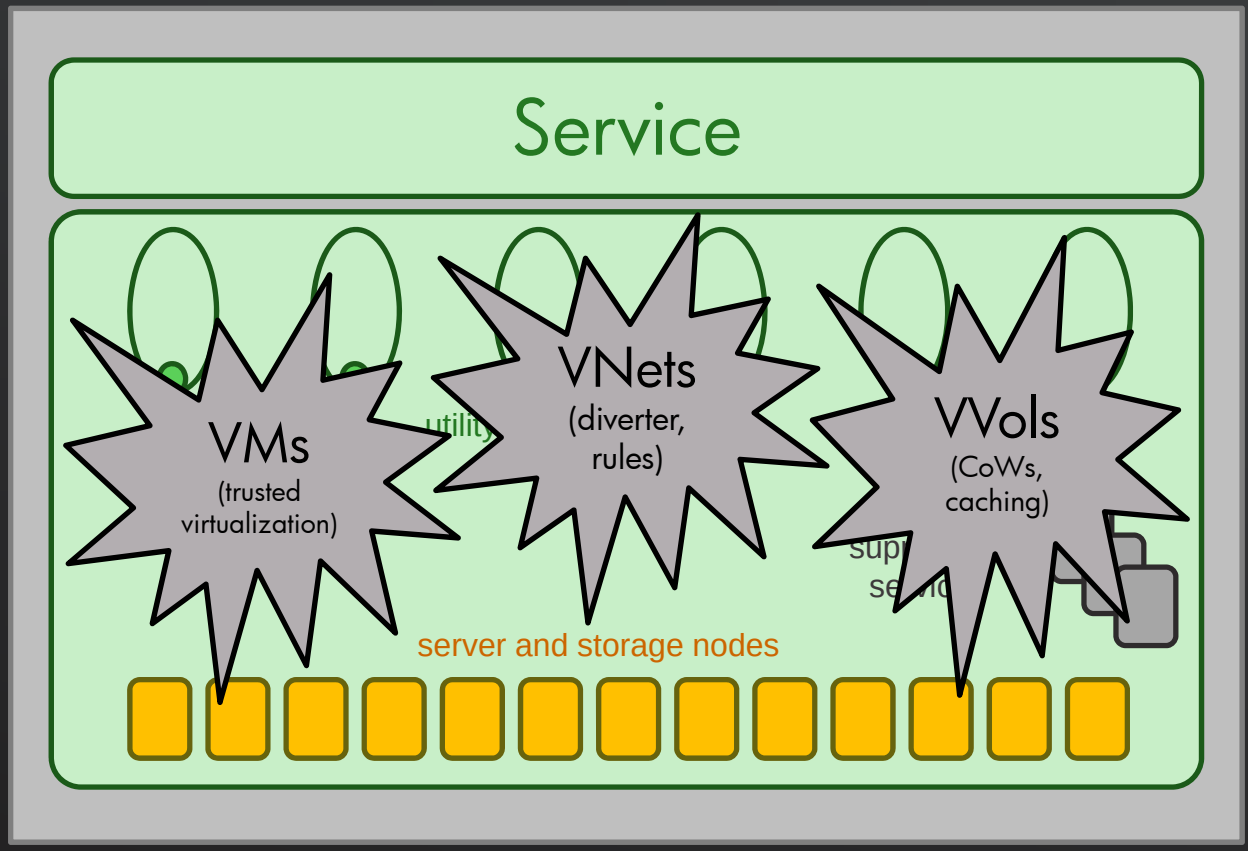
- We must protect the cloud at all costs
- Threats come from
  - Services run upon it
  - Direct attacks from outside
  - Internal administrators
- We must understand these threats, and the paths that they may use to undermine the cloud
- We must provide a number of engineering solutions to deal with the threats
  - Minimize attack surface, defence in depth
  - Provide a framework for countermeasures
  - Sensors to detect attacks, both attempts and successes
  - Mitigations to remove attempted and successful attacks
  - Diagnosing attacks by turning sensor data into diagnoses

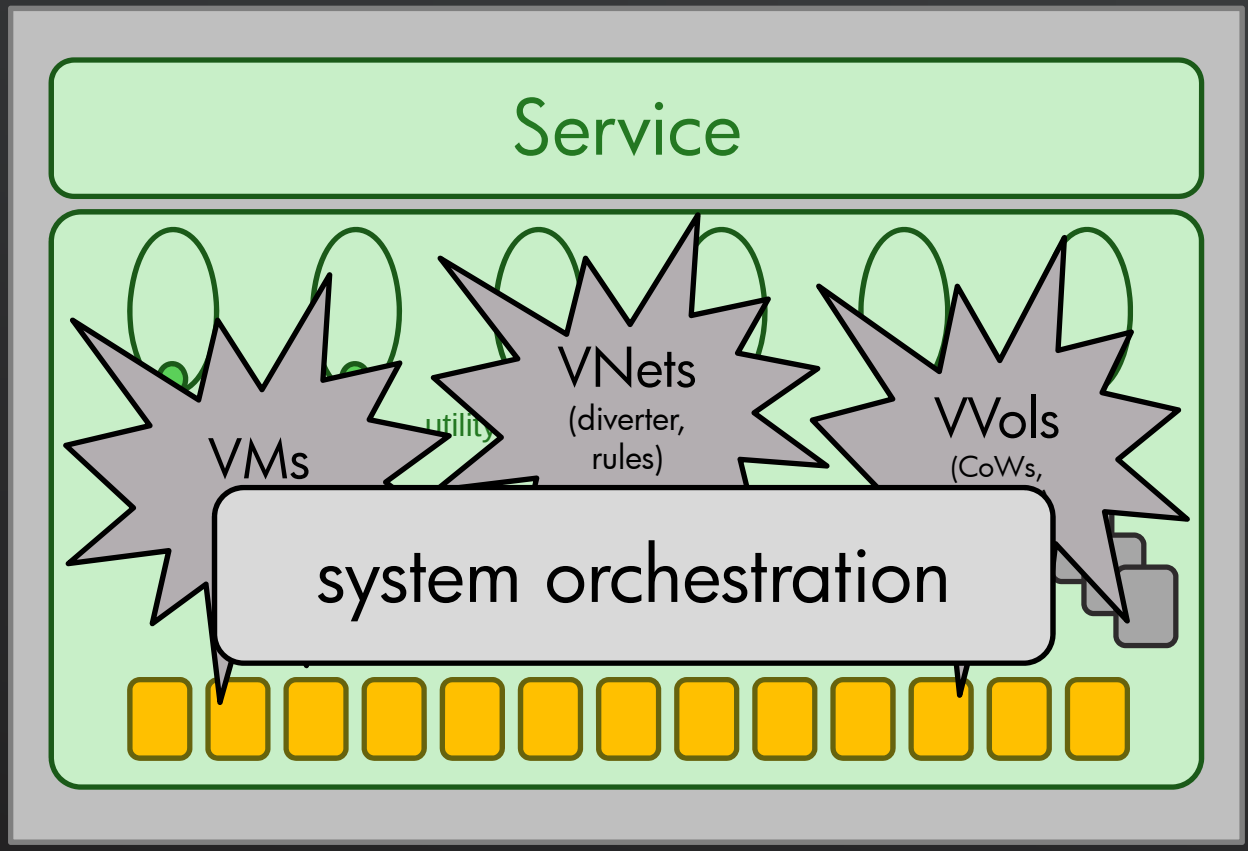
Threats





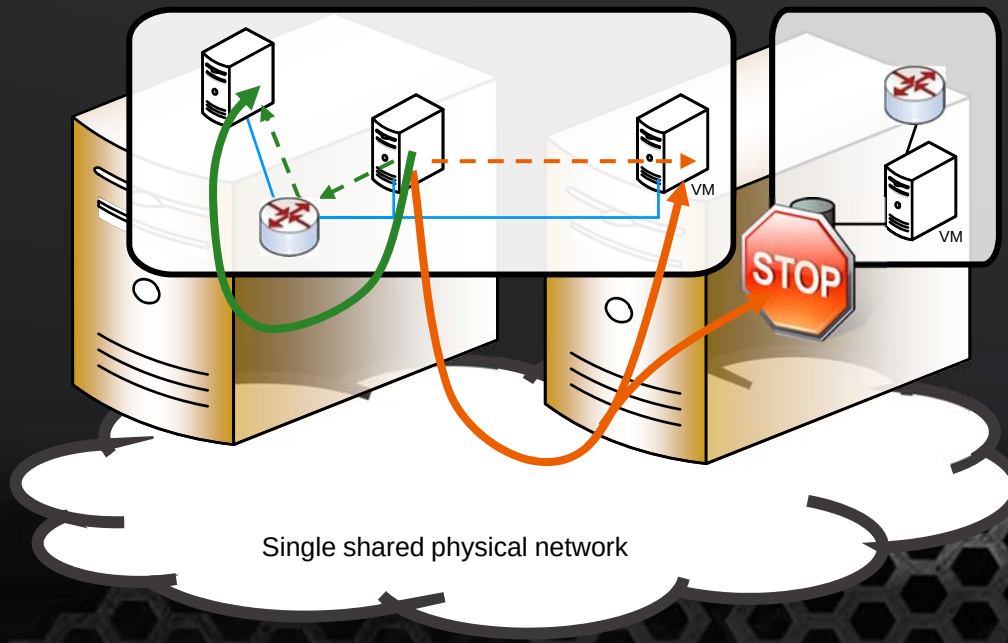




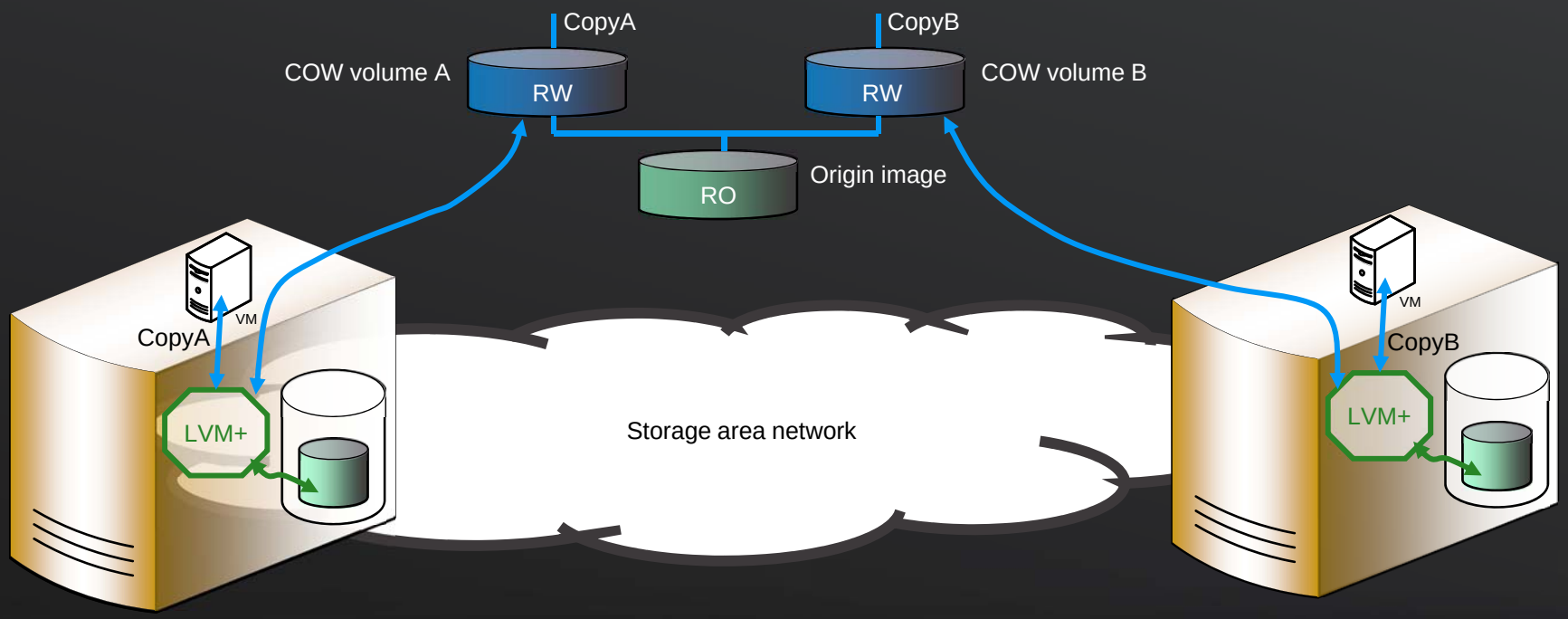


Virtual Networking

- Goals
  - Support full layer-3 unicast, multicast and broadcast packets
  - Create the illusion of subnets and routing
  - Provide all the inter-VM and subnet routing policies
  - Strong and secure separation

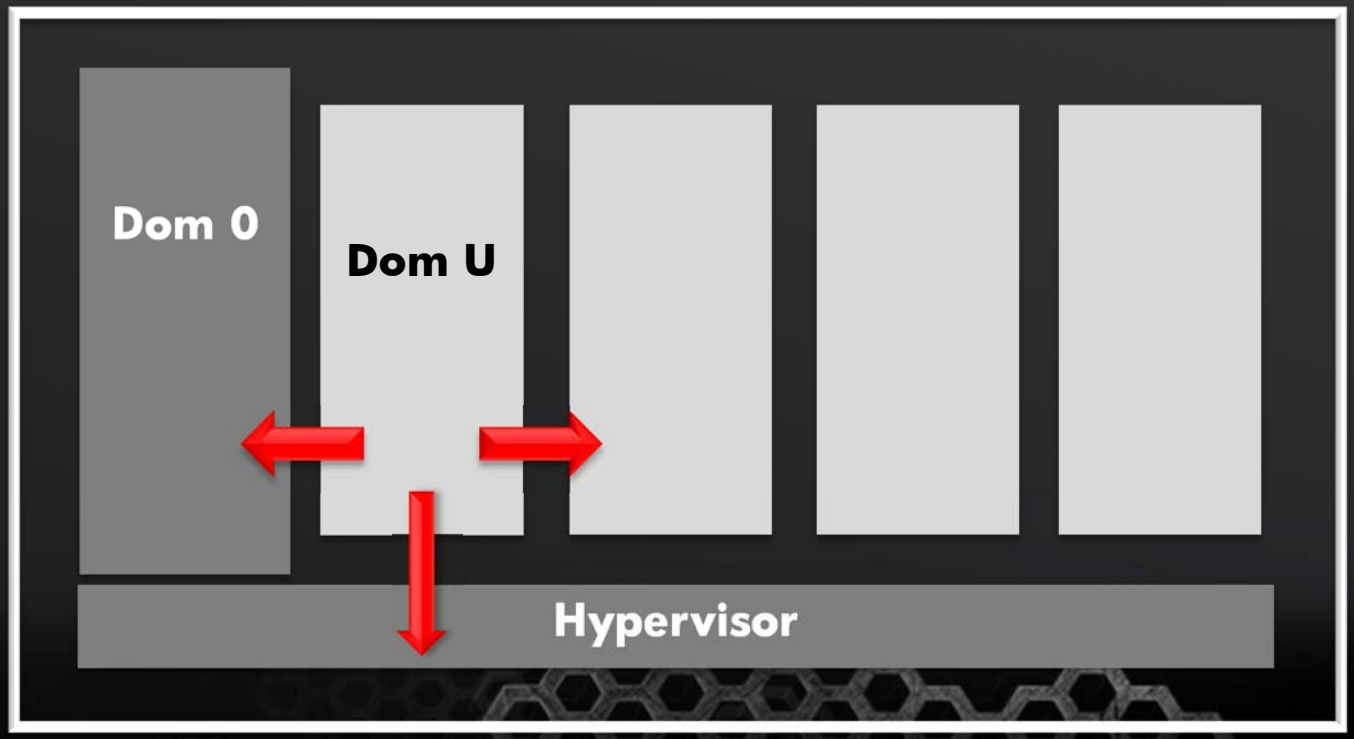


Virtual Storage



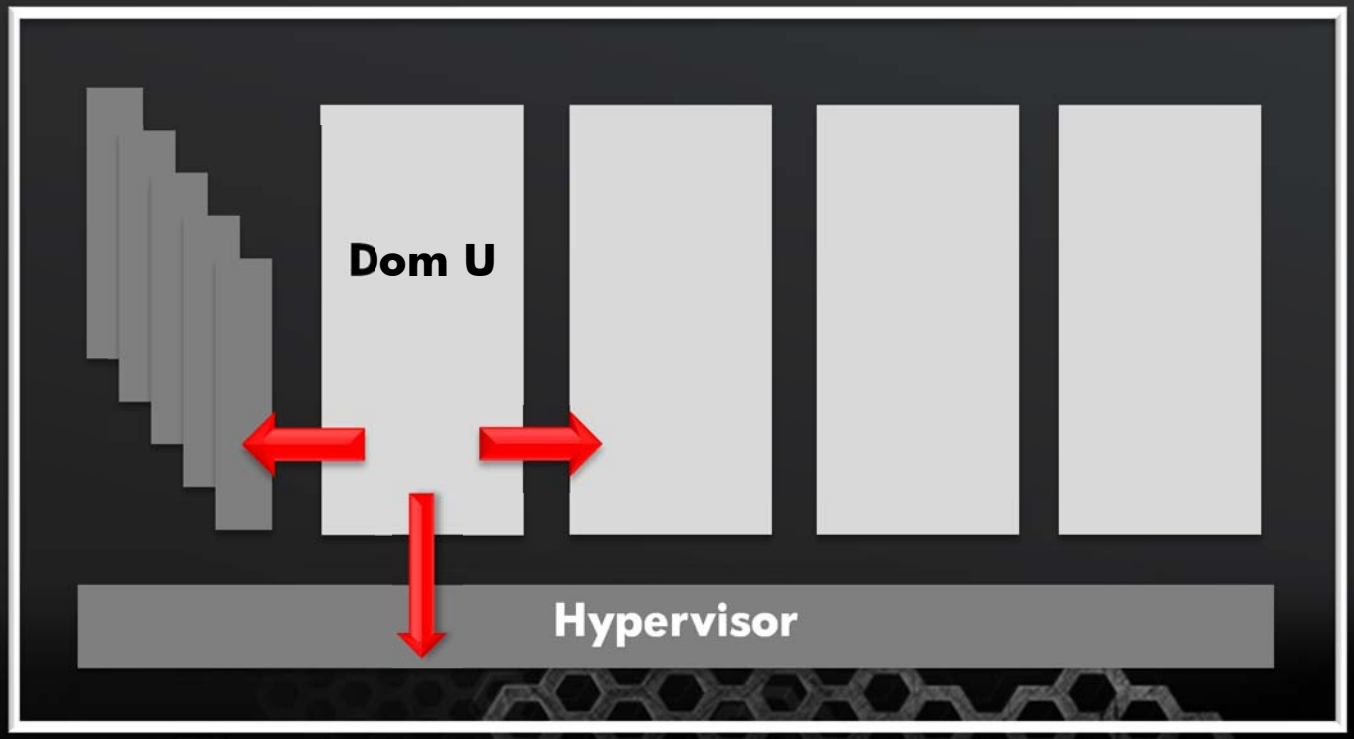
Virtualization Security: Threat

- The main requirement is that the core virtualization technology is secure
- What happens if a VM successfully breaks out of its container and takes over the host?



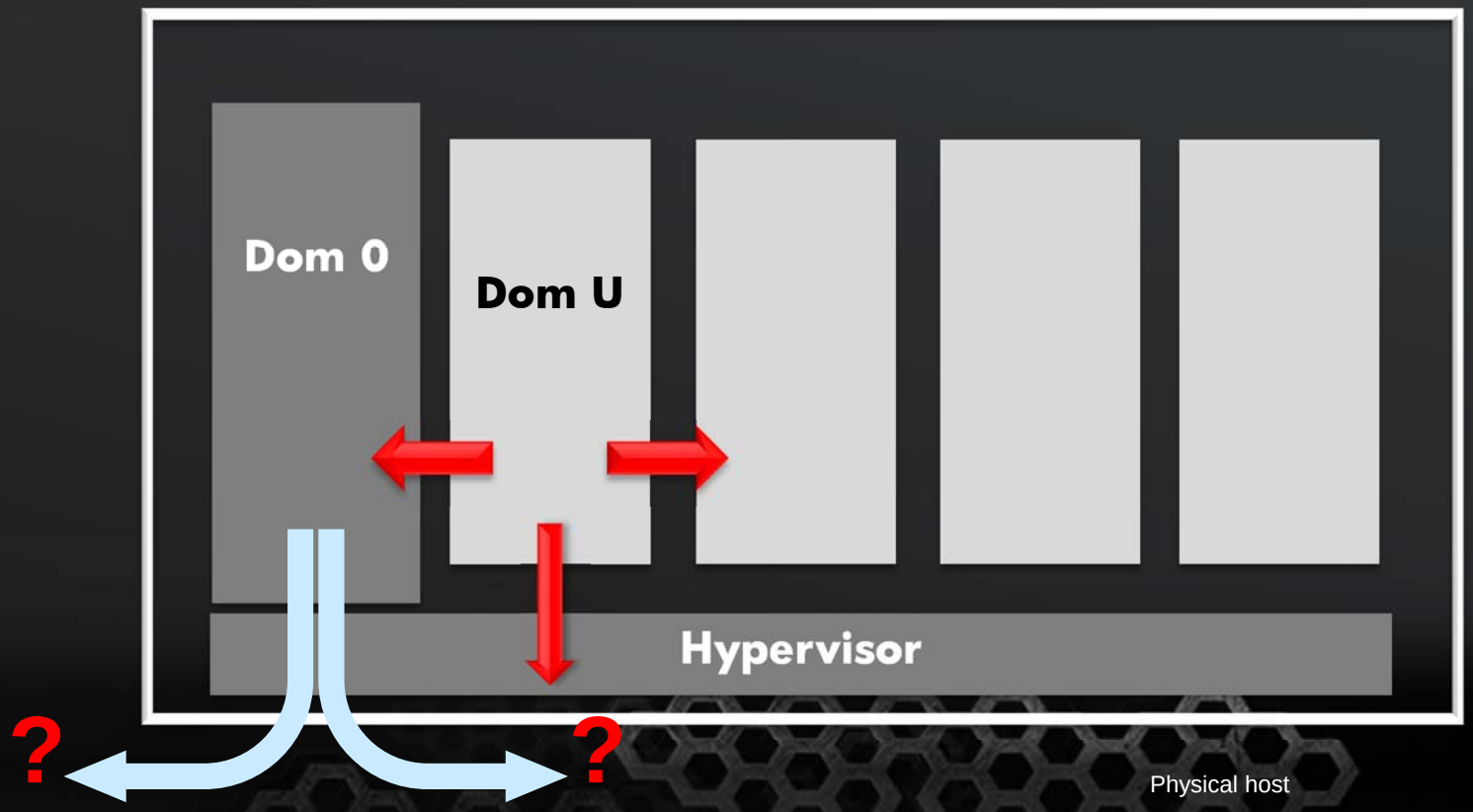
Virtualization Security: Prevention

- Use trusted virtualization technologies to minimize risk of attack and reduce impact
- Subdivide the Hypervisor and DOM0 into smaller, simpler, more secure parts and limiting the impact of success



Virtualization Security: Host Compromise

– Provide host peer-peer detection of dom0 “misbehaviour”



Sensors

Aim to detect:

– Illegal Actions

- System components detecting or initiating abnormal activity with other system components: this is either a bug or a successful attack and needs immediate attention
- Attempts of user VMs to communicate with disallowed targets

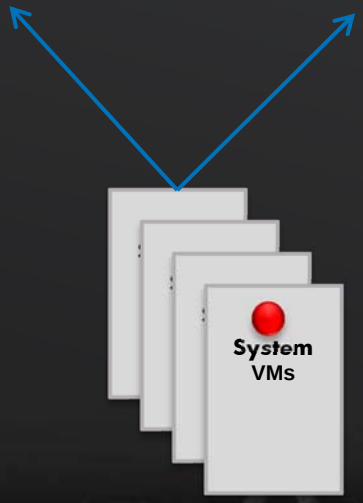
– Abnormal behaviour

- Sudden changes in profiles of IO or CPU usage
- Requests for VMs or other resources beyond reasonable or specified limits
- Excessive churn in topology
- Sudden widening of network rules

Sensors are implemented everywhere

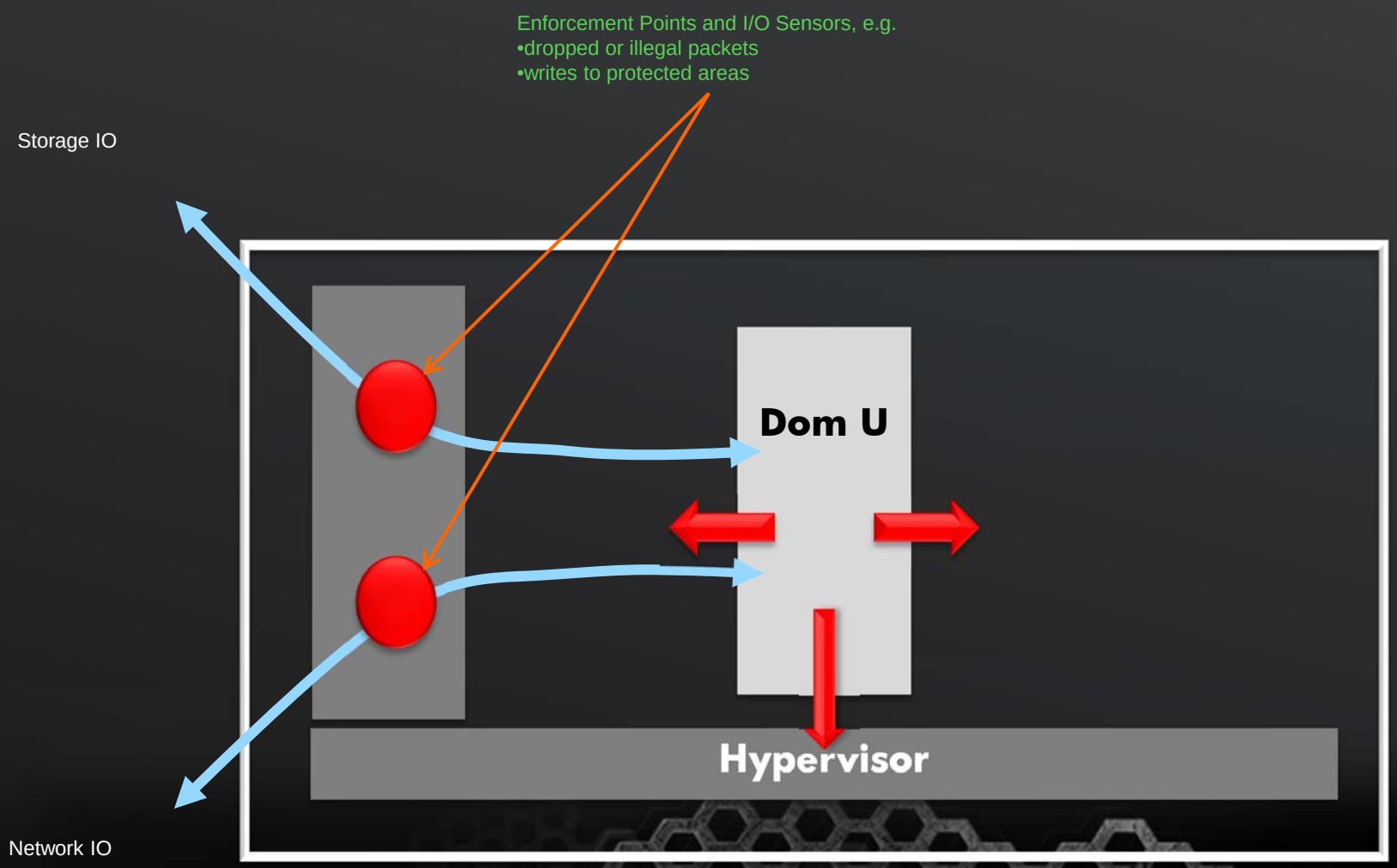
Sensors: System

System components inter-communicate in limited ways



System components can verify requests for "reasonableness"

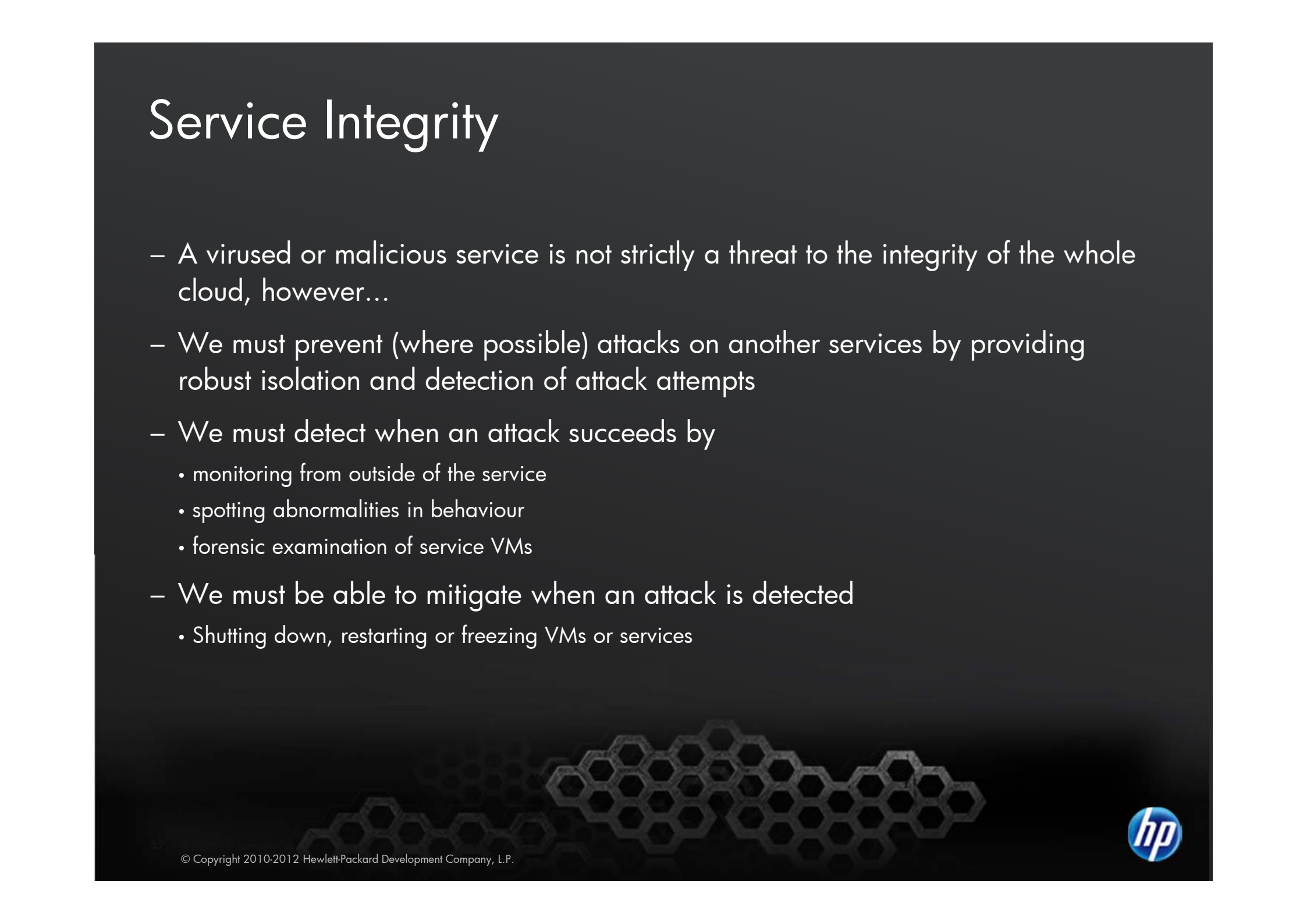
Enforcing and Sensing: IO



Includes checks on System VMs

Service Integrity

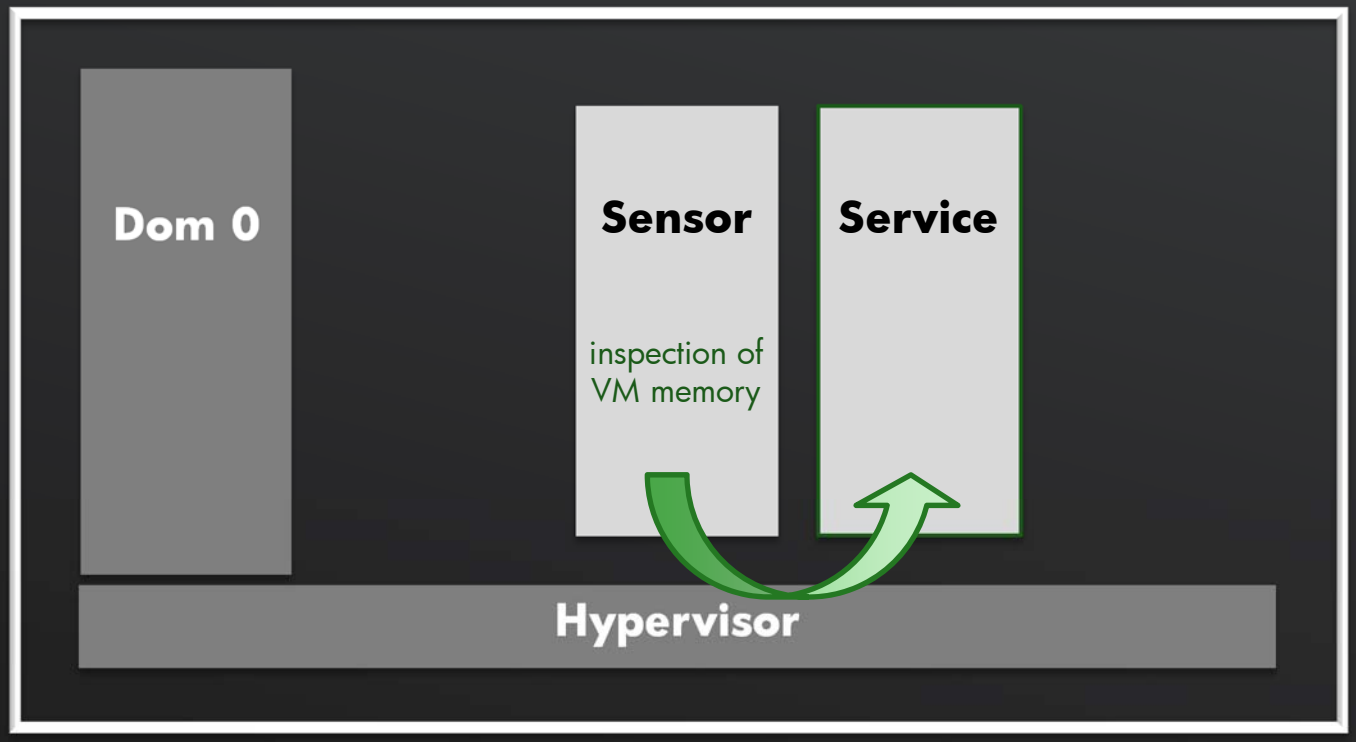
- A virused or malicious service is not strictly a threat to the integrity of the whole cloud, however…
- We must prevent (where possible) attacks on another services by providing robust isolation and detection of attack attempts
- We must detect when an attack succeeds by
  - monitoring from outside of the service
  - spotting abnormalities in behaviour
  - forensic examination of service VMs
- We must be able to mitigate when an attack is detected
  - Shutting down, restarting or freezing VMs or services



Service Sensors

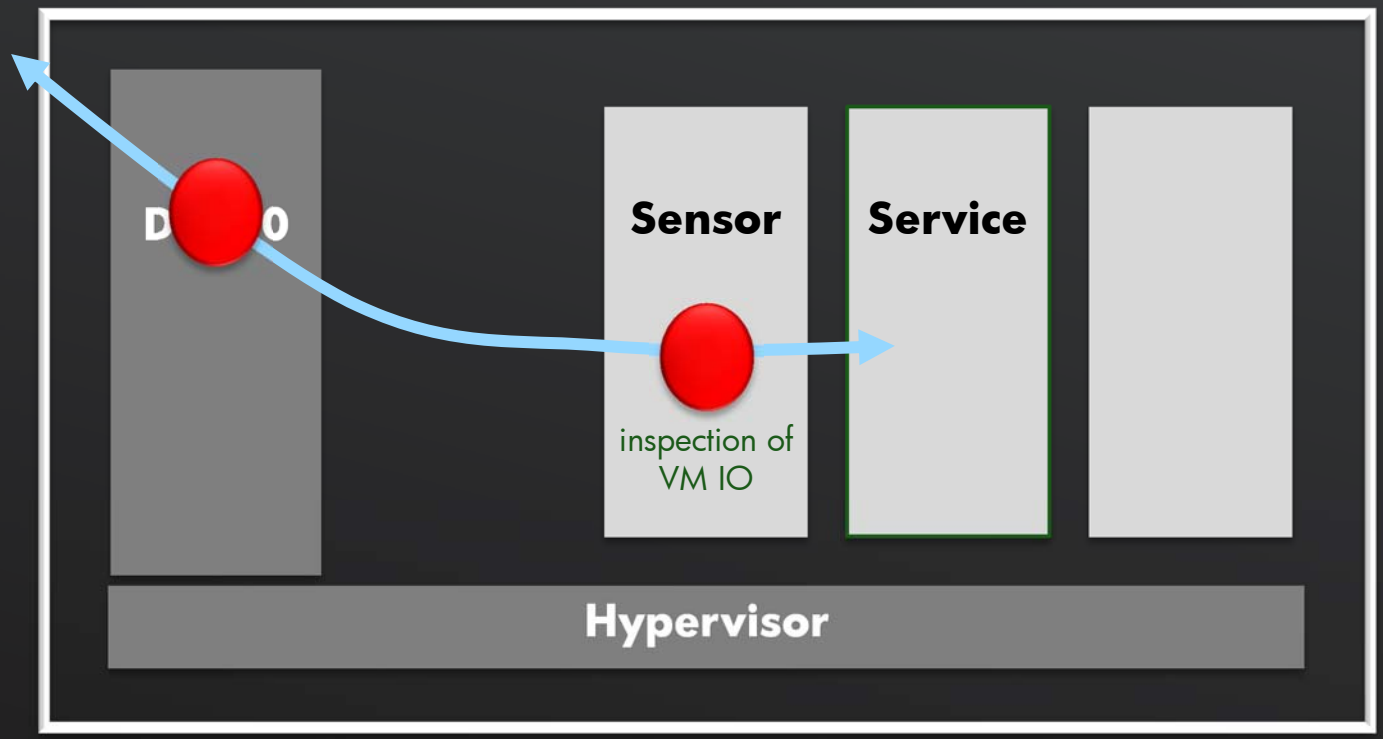
- We want to allow service writers to be able to create sensors that monitor their own services
  - run outside of the service, looking in, and undetectable to it
  - service owners have a better view of the service semantics
  - can be offered by 3rd party monitoring specialists (NSA for USG)
  - that do not have any privileged access to core system capability
- Virtualization gives control over what one compartment can do or see with another compartment.
- We can use the fact that one virtual machine can (given permission)
  - look into the memory space of another.
  - Interpose itself into an IO path of another.

Sensor VMs



- Invisible to the service VM
- Viruses and root-kits cannot hide by altering OS or disabling virus checkers
- Enabled by an API in the hypervisor
- Needs care to ensure that it doesn't become another vector for attack!

Sensor VMs



– Deployable by the service provider, infrastructure provider or trusted 3rd party

Sensor VM properties

- It's hard to detect the presence of the sensors.
- It's impossible to hide the code or IO from the sensors
- We can see if the OS tables have been manipulated
- We can see into disc and network buffers
- We can sit in the IO path and carry out specific deep inspection
- We can look for evidence of the use of different "virus components"
- We do not look only for specific attacks
- We gain evidence to suggest the existence of malware.

Our Demonstrator