ÚJ SZÉCHENYI TERV

Óbudai Egyetem
Pro Scientia et Futuro

Írta: Schubert Tamás, Windisch Gergely

Lektorálta: Schubert Tamás

INFORMATIKAI SZOLGÁLTATÁSOK SZÁMÍTÁSI FELHŐBEN (CLOUD COMPUTING)

INFORMATIKAI SZOLGÁLTATÁSMENEDZSMENT MODUL

PROAKTÍV INFORMATIKAI MODULFEJLESZTÉS

TYPOTEX

LEKTORÁLTA: Dr. Schubert Tamás

TÁMOGATÁS:

Készült a TÁMOP-4.1.2-08/2/A/KMR-2009-0053 számú, "Proaktív informatikai modulfejlesztés (PRIM1): IT Szolgáltatásmenedzsment modul és Többszálas processzorok és programozásuk modul" című pályázat keretében

Nemzeti Fejlesztési Ügynökség
www.ujszechenyiterv.gov.hu
06 40 638 638

MAGYARORSZÁG MEGÚJUL

A projekt az Európai Unió támogatásával, az Európai Szociális Alap társfinanszírozásával valósul meg.

ÚJ SZÉCHENYI TERV

KÉSZÜLT: a Typotex Kiadó gondozásában

FELELŐS VEZETŐ: Votisky Zsuzsa

ISBN 978-963-279-557-7

KULCSSZAVAK:

Számítási felhő, Infrastructure as a Service, Platform as a Service, Software as a Service, Database as a Service, felhő szolgáltató, megbízhatóság, biztonság, megfelelőség, méretezhetőség, rugalmasság, önjavító képesség, önkiszolgálás, on-line fizetés, sok felhasználó, tároló hálózat, nyilvános felhő, magán felhő, hibrid felhő

ÖSSZEFOGLALÓ:

Új egyéni, kis- és nagyvállalati igényeket egyaránt kielégítő informatikai szolgáltatások jelentek meg az interneten. Standard és testre szabható szolgáltatások, tetszőleges számú és teljesítményű számítógép és tárterület bérelhető előre megkötött szerződések szerint, vagy az igény felmerülésekor. Mindezt a világszerte kiépített hatalmas adatközpontok, a hálózati sávszélesség növekedése, a virtualizáció, az infrastruktúrát kezelő szoftverháttér, és új alkalmazásfejlesztő eszközök teszik lehetővé. A számítási felhő vagy Cloud Computing az informatikai szolgáltatások bérleti rendszerű igénybevételével szükségtelenné teszi az infrastruktúra helyi kiépítését. Az informatikai szolgáltatások olcsóbbá válnak, mivel az adatközpontok kihasználása többszöröse is lehet a helyi infrastruktúra kihasználásánál.

A tárgy keretében a hallgatók megismerik a számítási felhőben nyújtott szolgáltatások gazdasági kérdéseit, technológiáit, hardver infrastruktúráit, szoftver fejlesztő platformjait, üzemeltetését, biztonsági kérdéseit és a rendelkezésre állását növelő lehetőségeket. Ezen kívül megismerkednek még egy privát infrastruktúra felhő kialakításának szempontrendszerével, lépéseivel, és a létrehozott infrastruktúra menedzselésével.

Cloud Computing

Introduction

Tamás Schubert

1. Cloud Computing definition

Cloud Computing Defined [3]

- The National Institute for Standards and Technology (NIST), Information Technology Laboratory offers this definition of Cloud Computing:

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. The cloud model of computing promotes availability.

Cloud Computing Defined (Cont.)

- „Cloud" is a metaphor of the internet
- Customers use real-time and scalable information resources and services from the internet using mainly an internet browser:
  - o computers (virtual machines)
  - o storages (SAN, NAS storage)
  - o databases
  - o operating systems and standard/customized applications running on them
  - o networks of virtual machines
  - o standard applications made for many people
  - o all the services above can be used in a specific time and on-demand way
- These services are called … as a services. E.g.: Software as a Service – SaaS)
- The resources reside in data centers of the service providers (mainly distributed
- Customers only have thin client devices or internet browsers

Cloud Computing Defined (Cont.)

- Customers can be users, small and middle companies or enterprises alike
- CC is the result of the convergence of 3 main trends:
  - Service orientation
  - Virtualization
  - Standardization of the operations available on the internet
- The build-up of CC is irrelevant, several technologies are used
- In the back-end of the services are data centers, grids, traditional technologies, management applications and new application development languages and tools
- Services are scalable – The service needs to be available all the time (7 days a week, 24 hours a day) and it has to be designed to scale upward for high periods of demand and downward for lighter ones. *Scalability also means that an application* can scale when additional users are added and when the application requirements change

Cloud Computing Defined (Cont.)

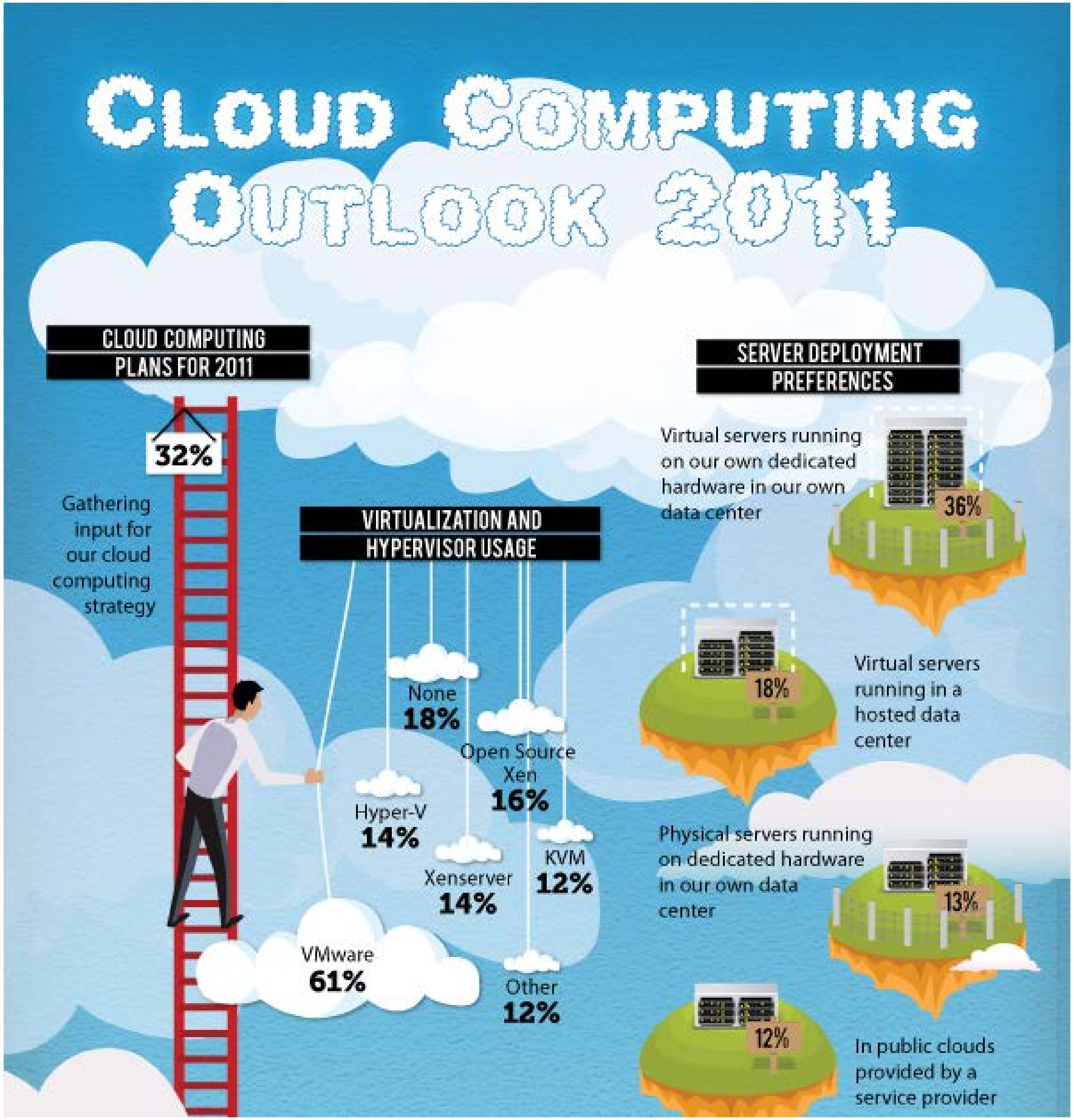
- Services are elastic – Elasticity is a trait of shared pools of resources. Elasticity is associated with not only scale but also an economic model that enables scaling in both directions in an automated fashion. This means that services scale on demand to add or remove resources as needed
- The system is self-healing
- Service-level agreements (SLA) can be contracted - A cloud service provider must include a service management environment. A service management environment is an integrated approach for managing your physical environments and IT systems. This environment must be able to maintain the required service level for that organization
- Services are available by on demand (self-service provisioning)
- Services can be used by several customers at the same time (Multi-tenancy)
- Security – Providers must ensure the security of the stored data and the security of the communication. (Compliance)
- Customers pay for the use of resources and services (time based fee, storage fee (GB/month), bandwidth use, etc.)
- Pay on-line

2. Cloud Computing Outlook 2011

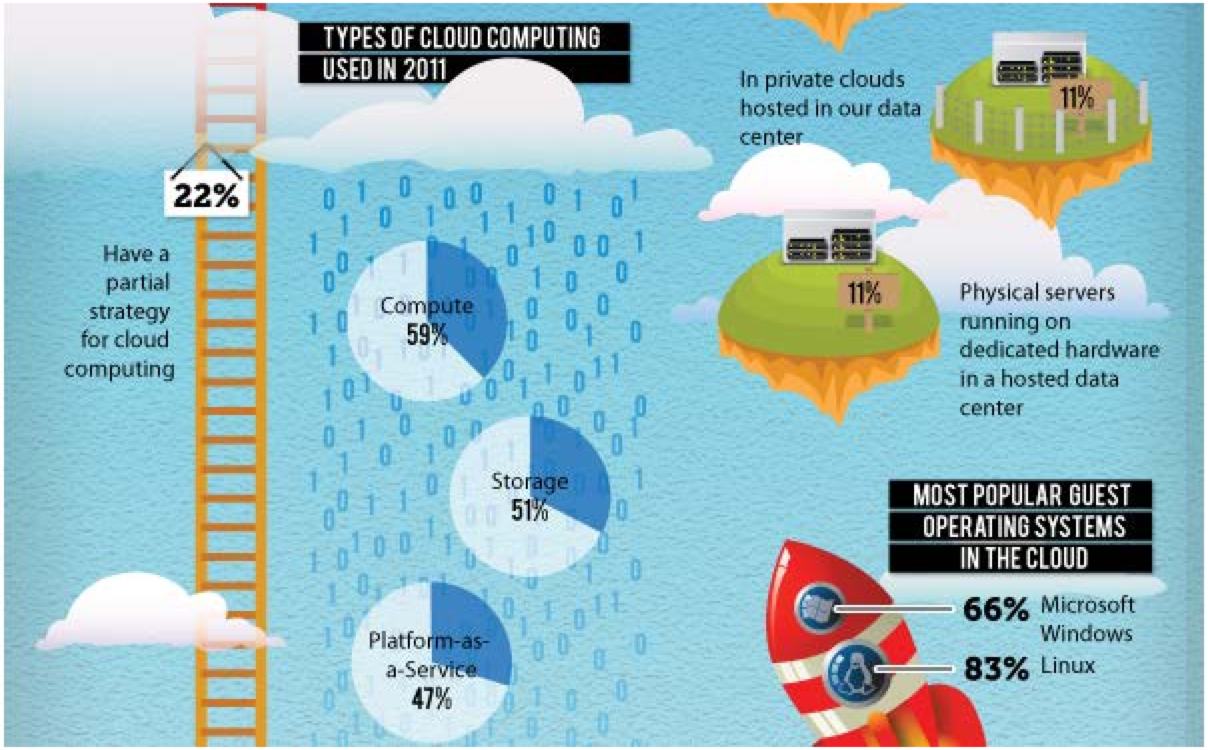
Cloud Computing Outlook 2011 [1]

- Cloud computing plans for 2011
- Virtualization and hypervisor usage
- Server deployment preferences
- Types of cloud computing used in 2011
- Most popular guest operating systems in the cloud
- Stance on using open source software
- Perceived benefits from cloud computing
- Factors driving the adoption of cloud computing
- Cloud computing use cases

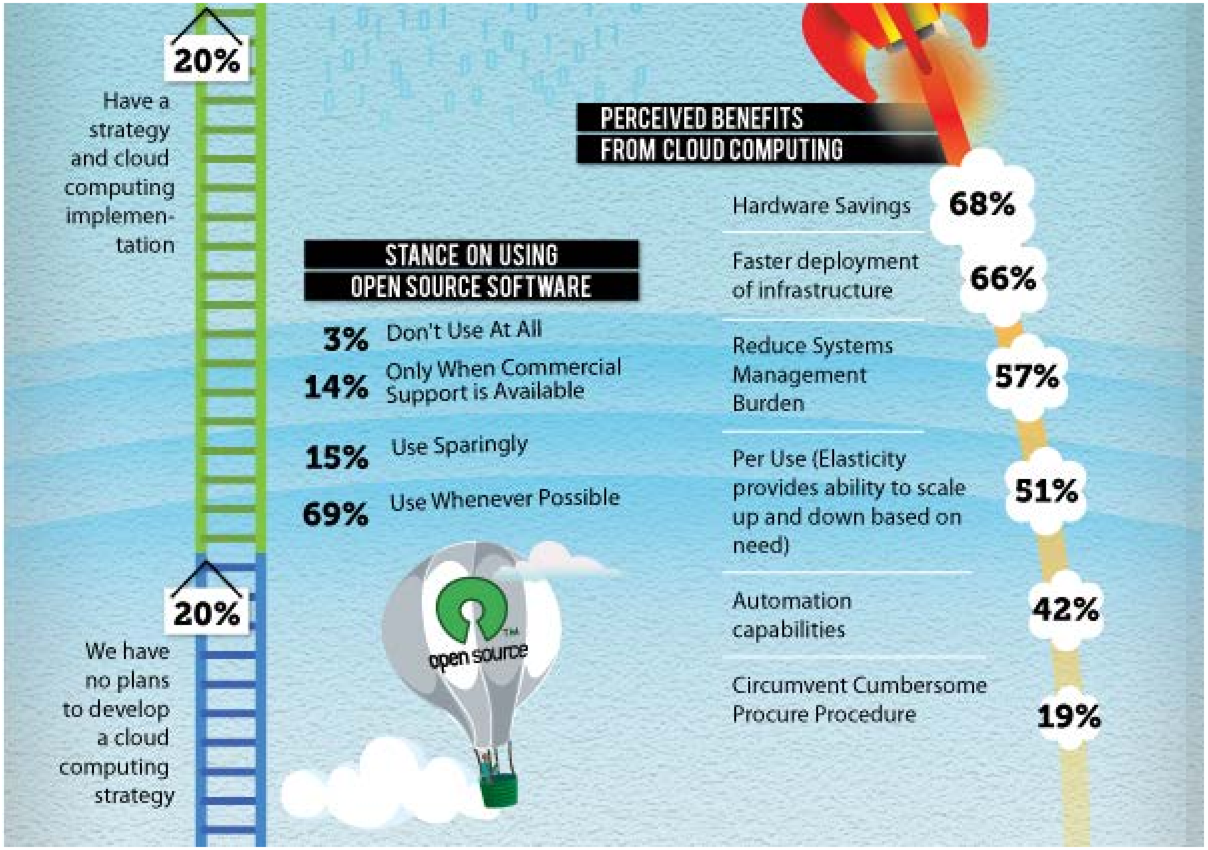
Cloud Computing Outlook 2011 (Cont.)



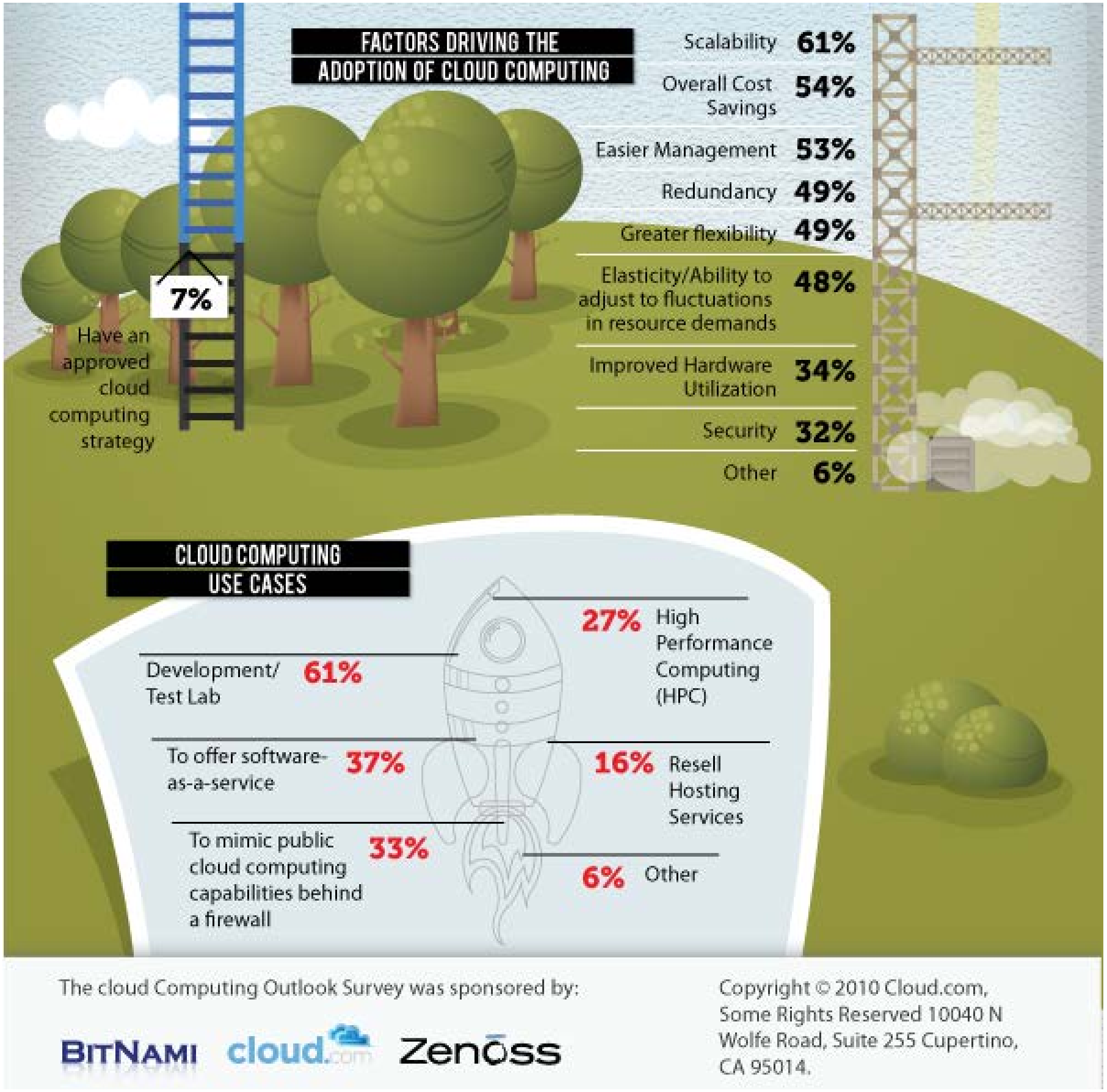
Cloud Computing Outlook 2011 (Cont.)



Cloud Computing Outlook 2011 (Cont.)



Cloud Computing Outlook 2011 (Cont.)



3. Traditional vs. Cloud services

Disadvantages of the traditional information infrastructure

- The utilization of servers in the data centers of a company: ~18%
- The utilization of other components of the information infrastructure is also small
- CAPEX (Capital expenditure) is high (data center, network, environment, software licenses, etc.)
- Continuous expansion and upgrade of the infrastructure (HW, SW)
- OPEX (Operational expenditure) is also high
- Skilled staff is needed
  - Hardware
  - Software
  - Network
  - Security
  - Management
  - Etc.

Advantages of the traditional information infrastructure

- The operation of the information infrastructure (supporting the business goals) depends on only the company itself
- The company itself ensures the expected level of availability and service level
- The level of security depends on only factors of the company
- The business continuity can be ensured easily

Advantages of hiring information services from the Cloud

- Data centers offering the services concentrate (consolidate) the resources
- The specific expenditure is lower
- The utilization of the resources in the cloud is higher. The utilization of the servers reaches 50-70%
- The utilization of the software licenses is also higher, because it doesn't need to buy so many licenses than the number of people who use the cloud services
- The highly skilled IT experts work for the cloud provider, not for the company who consume the services
- The utilization of the resources is increased that
  - the resources (processor, memory, storage, software) are dynamically assigned to the applications and customers, and
  - the services even span continents, so due to the time-lag the load of the data centers may be smoother

Advantages of hiring information services from the Cloud (Cont.)

- Technologies used in the cloud may ensures the expected availability and SLA
- The IT security can also be ensured according to the contract
- All the factors mentioned above make possible significant reduction of the cost for the customers. It may be more rewarding for the companies and the customers to hire IT services from the cloud than build up and run their own infrastructure

Disadvantages of hiring information services from the Cloud

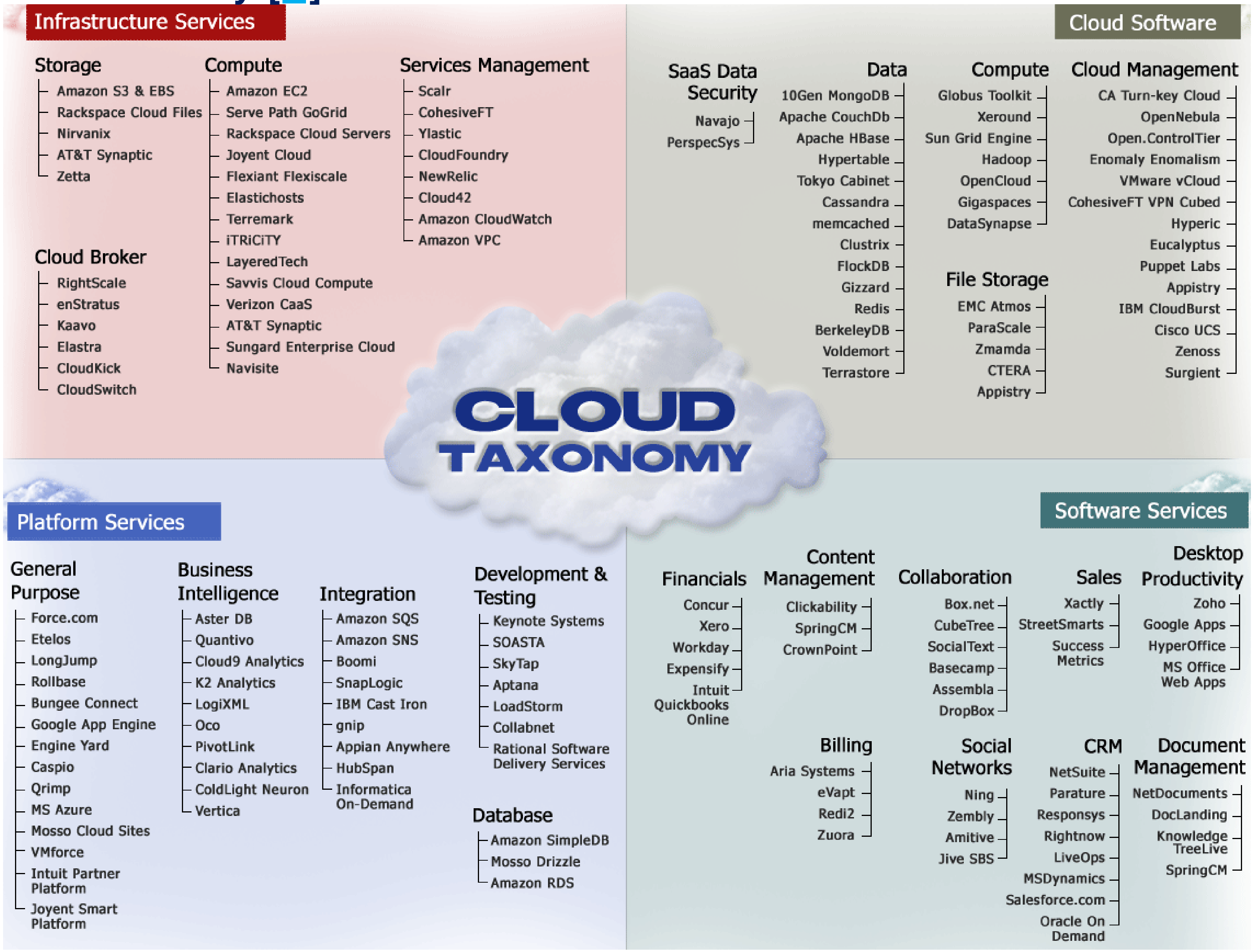
- Data are stored in an unknown place (continent, country), so the management of the information asset is getting out from the control of the data owner
- In the case of an outage of the services or the loss of information asset companies can become to bankrupt
- In the case of a disaster, war, etc., the access of the services is impossible. (Individuals are less affected than companies)
- To reduce that risk and yet the advantages of the cloud technology is leverage, companies can build and run their own Private Cloud

4. Cloud Taxonomy

Cloud Computing now is not a finished technology

- To day, several good services are available
- There are several R&D projects
- Big IT companies are running joined R&D projects to decrease their costs
- Stakeholders: Microsoft, IBM, HP Sun, Intel, Google, Amazon, Yahoo, etc.
- Small IT companies are also participating in the developments
- Several open source products are available for the IT community

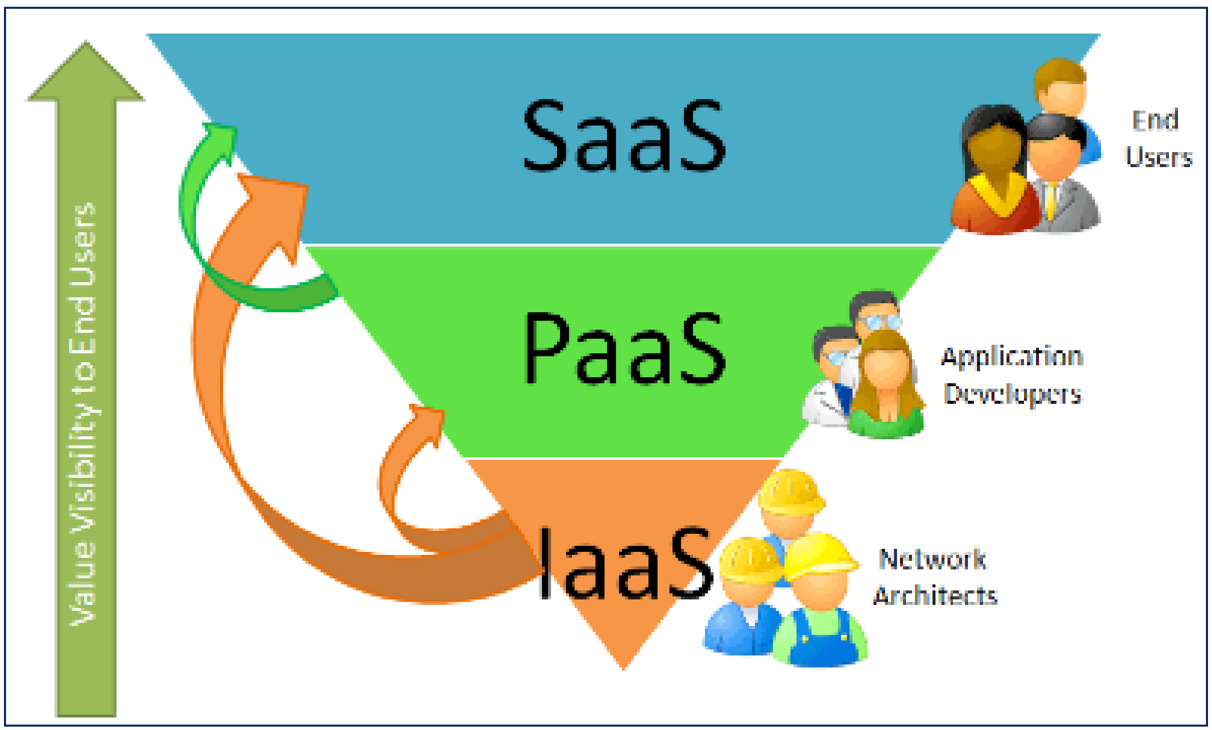
Cloud taxonomy [2]



5. Cloud Computing Architecture

Cloud Computing architecture

- Services build upon other services:
  - Infrastructure as a service (IaaS)
  - Platform as a service (PaaS)
  - Software as a service (SaaS)



Infrastructure as a Service (IaaS)

- Infrastructure as a Service (formerly Hardware as a Service)
- Providers let virtual machines (mainly in platform virtualization environment)
- No need for the customers to purchase and run servers, storages, network devices, software licenses, computer rooms, etc.
- Customers purchase resources as an outsourced service
- Payment is similar that of utility computing, according to the consumed resources (e.g.: specific amount per hour)
- The quality of service can be described in the Service Level Agreement
- IaaS is frequently implemented by grids
- The network can be protected by firewalls, load balancing and redundant solutions can be applied
- IaaS service can be reached via internet
- There are several providers: Amazon EC2, Amazon S3, GoGrid, etc.

Platform as a Service (PaaS)

- *Platform includes the whole lifecycle of the development, test, deployment and operation of CC applications*
- The whole lifecycle is based on Cloud Computing
- Key components of PaaS:
  - The development, test, deployment, run and management of the cloud applications is operated in the same integrated environment (cost is decreasing, quality and availability are increasing)
  - The user comfort, response time and quality must be ensured without any compromise (same quality expectation as in the traditional applications). Software download, plug-in installation and local program run can't influence the use of the cloud application
  - The realization of built-in scalability, reliability and security without extra development, configuration and cost. Automatic multi-tenancy. The storage and the transmission of data, and the financial transactions should be secure during the whole lifecycle of the application

Platform as a Service (PaaS) (Cont.)

- Built-in integration with Web Services and databases. Link services running at distant locations and link data stored at distant locations
- Support cooperation of developers and developer groups. The platform must ensure the cooperation during the whole lifecycle (development, test, documentation, deployment and operation) of the application without any special configuration
- Deep monitoring built into the application, which records the activity of the users, the faults and the performance issues. The recorded information helps the developers in the enhancement of the applications, and in the exploration of new user expectations

Software as a Service (SaaS) [7]

- Applications are available and can be managed via the internet
- Applications can be accessed exclusively by an internet browser, local installation isn't necessary
- The data structure of the application (distributed model) and the program architecture permit, that the application be used by several people at the same time (multi-tenancy)
- Uniform applications can be easily migrated to the cloud. The SaaS application needs to be generalized enough so that lots of customers will be interested in the service
- Customization can be achieved (without code change) by parameterization
- The security of the communication can be achieved by using SSL
- Customers needn't buy software licenses (on demand licensing), customers only pay for the service (e.g. per-month, per-user fee)
- An SaaS application needs to include measuring and monitoring so customers can be charged actual usage

Software as a Service (SaaS) (Cont.)

- An SaaS application must have a built-in billing service
- SaaS applications need published interfaces and an ecosystem of partners who can expand the company's customer base and market reach
- SaaS applications have to ensure that each customer's data and specialized configurations are separate and secure from other customers' data and configurations
- SaaS applications need to provide sophisticated business process configurators for customers
- SaaS applications need to constantly provide fast releases of new features and new capabilities
- SaaS applications have to protect the integrity of customer data

Software as a Service (SaaS) (Cont.)

- Software licenses are managed by the cloud provider
- Costs are shared by several customers
- Software maintenance is managed by the cloud provider
- Version tracking are made by the provider
- Hardware costs decrease at the customer
- Hardware scaling can be more easily managed at the provider in the case of mass utilization
- Possible disadvantages:
  - Network problems (bandwidth shortage)
  - Security deficiency
  - Provider dependency
  - Limited customization

SaaS application types [6]

- Packaged software

  This is the biggest area of the SaaS market

  Examples:

  - Customer relationship management (CRM)
  - Supply chain management
  - Financial management
  - Human resources
  - Etc.

SaaS application types (Cont.)

- Some companies in the packaged software market:
  - Salesforce.com is a leader in cloud computing customer relationship management (CRM) applications
  - Netsuite, like Salesforce.com, offers a CRM foundation. Netsuite has added a number of modules for *enterprise resource planning (ERP)* application including financial capabilities, e-commerce, and business intelligence
  - Intuit provides a Financial Services Suite of products that support accounting services for small- and medium-sized businesses. The company provides a rich set of interfaces that enables partners to connect their services and applications into its environment
  - RightNow provides a CRM suite of products that includes marketing, sales, and various industry solutions
  - Concur focuses on employees spend management. It automates costs control via automated processes
  - Taleo focuses on talent management tasks
  - SugarCRM is a CRM platform built on an open-source platform. The company offers support for a fee

SaaS application types (Cont.)

- Some companies in the packaged software market:
  - Constant Contact is a marketing automation platform that partners directly with Salesforce.com and other CRM platforms. They automate the process of sending emails and other marketing efforts
  - Microsoft with its Dynamics package
  - SAP with its By Design offering for the small- to medium-sized business market
  - Oracle with its On Demand offering based on its acquisition of Siebel Software

SaaS application types (Cont.)

- Collaborative software
  - Web conferencing
  - Document collaboration
  - Project planning
  - Instant messaging
  - E-mail
  - Etc.

SaaS application types (Cont.)

- Some companies in the collaborative software market:
  - MicrosoftLive has made its first foray into collaboration as a service with its Meeting Live offering. Today Microsoft offers Meeting Live and live messaging services. In addition, Microsoft offers the ability to run its email server (Exchange as a Service). In the future, the company will have online versions of many of its collaborative applications
  - LotusLive is IBM's collaborative environment that includes a set of tools including social networking, instant messaging, and the ability to share files and conduct online meetings. IBM is publishing interfaces to allow other collaborative tools to be integrated into the platform
  - GoogleApps from Google, which has as many as 1.5 million businesses that use its various collaborative applications including e-mail, document management, and instant messaging. It publishes APIs so third-party software developers can integrate with the platform

SaaS application types (Cont.)

- Some companies in the collaborative software market:
  - Cisco Webex Collaboration platform comes from Cisco and it has become the centerpiece of its collaboration SaaS platform. It will probably use this platform to add unified communications as a service
  - Zoho, an open-source collaboration platform, includes email, document management, project management, and invoice management. It offers APIs to its environment and has begun to integrate its collaboration tools with other companies, such as Microsoft. Zoho offers support for a fee
  - Citrix GotoMeeting offers an online meeting service as part of its larger suite of virtualization products

SaaS application types (Cont.)

- Enabling and management tools

  They support the development and the deployment of SaaS

  - Testing as a service
  - Monitoring and management as a service
  - Development tooling as a service
  - Security as a service
  - Compliance and governance as a service

SaaS application types (Cont.)

- Enabling and management tools: Testing as a service
    - When a company moves to using a public or private cloud, it still needs to conduct the same testing it would need in an on-premise data center, including functional, unit, stress, compatibility, performance, requirements management and integration testing
    - Developers need to accurately simulate the conditions when software is deployed
    - More companies are looking at testing as a service and development as a service as a way to keep track of development teams that are often distributed across the globe
    - Having developers rely on SaaS-based services for testing can save tremendous amounts of time and money
    - Many vendors produce testing as a service platforms, including HP, IBM, Sogeti, Compuware, as well as smaller companies

SaaS application types (Cont.)

- Enabling and management tools: Monitoring and management as a service
  - Companies using SaaS need to do some of their own monitoring to determine if their service levels have been met by their SaaS providers. More complicated is when companies are using more than one SaaS application, and companies must monitor not just a single application but also the combination of applications
  - Companies in the systems management space come at this market from two different perspectives:
    - Large telecommunications are packaging their capabilities so they can help provide cloud management and monitoring
    - Traditional Web services monitoring companies offering services that will tell customer if its Web site has added new services to support the cloud

SaaS application types (Cont.)

- Enabling and management tools: Development tooling as a service
  - Development is done in a cloud based environment instead of implementing development within a single internal-development environment
  - This model of development infrastructure can be done through one of the Platform as a Service vendors such as Google, Intuit, Microsoft, Force.com, and Bungee Labs
  - Infrastructure as a Service vendors such as Amazon.com offer support services for developers

SaaS application types (Cont.)

- Enabling and management tools: Security as a service
  - Almost without exception, vendors providing antivirus software are offering their products as a service. These vendors include Symantec, McAfee, CA, and Kapersky Labs
  - Companies such as Hewlett-Packard and IBM have tools that scan environments for vulnerability scanning and testing
  - Identity management is an important aspect of on premise as well as cloud services. Lots of companies in this market will begin offering identity management as a service

SaaS application types (Cont.)

- Enabling and management tools: Compliance and governance as a service
  - Compliance and governance tasks are time consuming and complicated tasks that large companies are required to do. Therefore, offering these capabilities as a service is critical
  - Services that are becoming SaaS include the following:
    - Patch management
    - Business continuity planning
    - Discovery of records and messages
    - Various governance requirements such as SOX (Sarbanes-Oxley)

SaaS applications - Google Apps - Google Docs

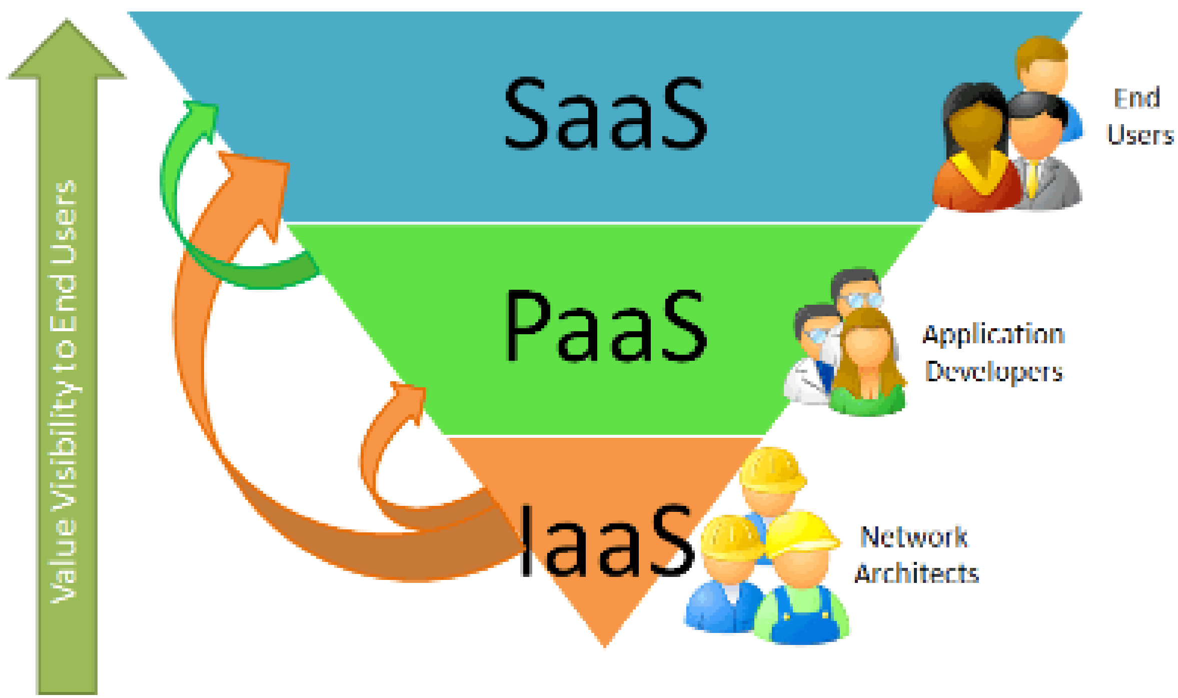
- Free Web-based Google services:
  - Word processor
  - Spreadsheet
  - Slide show
  - Data storage service
- Document
  - create
  - edit
  - import/export
  - send in e-mail
  - store on a Google server
- Real-time cooperation of users. Concurrent
  - open
  - edit
  - e-mail notification of users in the case of modification
- Support Microsoft .doc, .xls, .ppt forms
- Manage .pdf documents

Everything as a Service (EaaS, XaaS, *aaS)

- Naming of ... as a services:
  - Communication as a service
  - Infrastructure as a service
  - Monitoring as a service
  - Software as a service
  - Platform as a service
  - Database as a service
  - ...

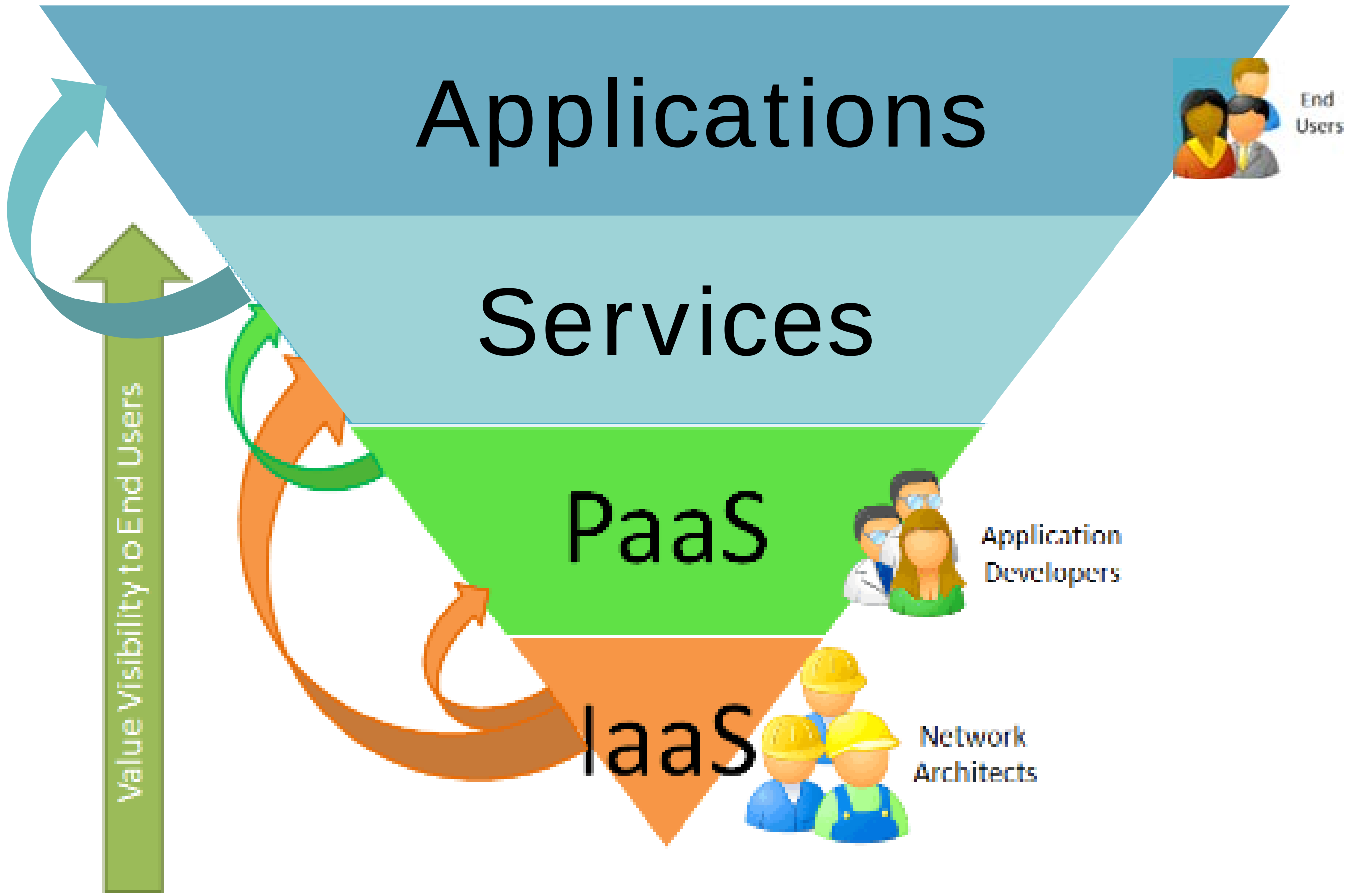
Modified architecture of Cloud Computing

- Services build upon other services (layers)
- The upper layer splits into two sub layers
  - Services
  - Applications



Modified architecture of Cloud Computing

- Infrastructure
- Platform
- Services
- Applications



Modified architecture of Cloud Computing

- Infrastructure: Computing, storage and network resources backing CC
- Platform: Software infrastructure that helps the development, test, deployment and operation of CC applications
- Services: Services in close symbiosis with the applications, like invoicing, storage, system integration
- Applications: The end-applications that directly serve users

6. Public, private and hybrid clouds

Public cloud

- Cloud provider offers services to companies and persons
- Some examples, when a public cloud is the obvious choice:
  - The standardized workload for applications is used by lots of people. Email is an excellent example
  - It needs to test and develop application code
  - Company has SaaS applications from a vendor who has a well-implemented security strategy
  - Company needs incremental capacity (to add compute capacity for peak times)
  - Company are doing collaboration projects
  - Company are doing an ad-hoc software development project using a Platform as a Service (PaaS) offering

Private cloud

- The private cloud is a highly virtualized cloud data center located inside your company's firewall
- It may also be a private space dedicated to the company within a cloud vendor data center designed to handle the company's workloads
- The main reasons why private clouds are used
  - Privacy and security of data is mandate
  - Companies need to keep their data center running in accordance with rules of governance and compliance
  - Companies have already invested in a lot of hardware, software, and space and would like to be able to leverage their investments, but in a more efficient manner
  - Companies have critical performance requirements (e.g. 99.9999 percent availability). Therefore, a private cloud may be their only option. This higher level of service is more expensive, but is a business requirement
- Some early adopters of private cloud technology have experienced server use rates of up to 90 percent. This is a real breakthrough

Hybrid Cloud

- Some public cloud companies are now offering private versions of their public clouds
- Some companies that only offered private cloud technologies are now offering public versions of those same capabilities
- Hybrid Cloud: A computing environment combining both private (internal) and public (external) cloud computing environments. May either be on a continuous basis or in the form of a ‘cloudburst’
- In most situations, hybrid clouds satisfy business needs:
    - A company likes a SaaS application and wants to use it as a standard throughout the company; company is concerned about security. To solve this problem, the SaaS vendor creates a private cloud just for the company inside their firewall. It provides a virtual private network (VPN) for additional security. Now the cloud have both public and private cloud ingredients
    - A company may want to use a public cloud to create an online environment so each customers can send requests and review their account status. However, the company might want to keep the data for these customers within its own private cloud

7. Cloud infrastructure

Cloud infrastructure components

- Servers, clusters, grids, supercomputers
- Storage networks (SAN, NAS)
- Data centers (resource consolidation)
- Virtualization
- Powerful data networks
- Management solutions
- High availability
- Quality of Service (QoS) according to the Service level agreement – SLA
- On-line payment
- Security
- Development tools

Data Centers (Concentration of devices)

Mainframe is on the left and on the right is the Customer Technology Center (CTC) for product functionality and interoperability testing

Data Centers (Concentration of devices)

Datacenter with a Sun Blade 6048

Blade Centers

- Concentration of powerful servers
- Blade servers: 2-4 processors, 4 - 192 GB RAM
- No or low capacity HDD; Use SAN or NAS
- Servers are connected to a common, high speed, redundant backplane
- Common and redundant power supply and cooling
- Common network interfaces (LAN and SAN)

IBM BladeCenter H

HP BladeSystem c7000

Storage Area Network (SAN)

- Computers use a dedicated storage network to attach to the storage devices
- The storage access mechanism is block based. Servers directly access data blocks via storage area network
- File system is provided by servers, workstations or NAS devices
- Computers provide volume management
- RAID is ensured by the storage device
- Block aggregation are shared among computers, storages and storage network elements

Fibre Channel servers

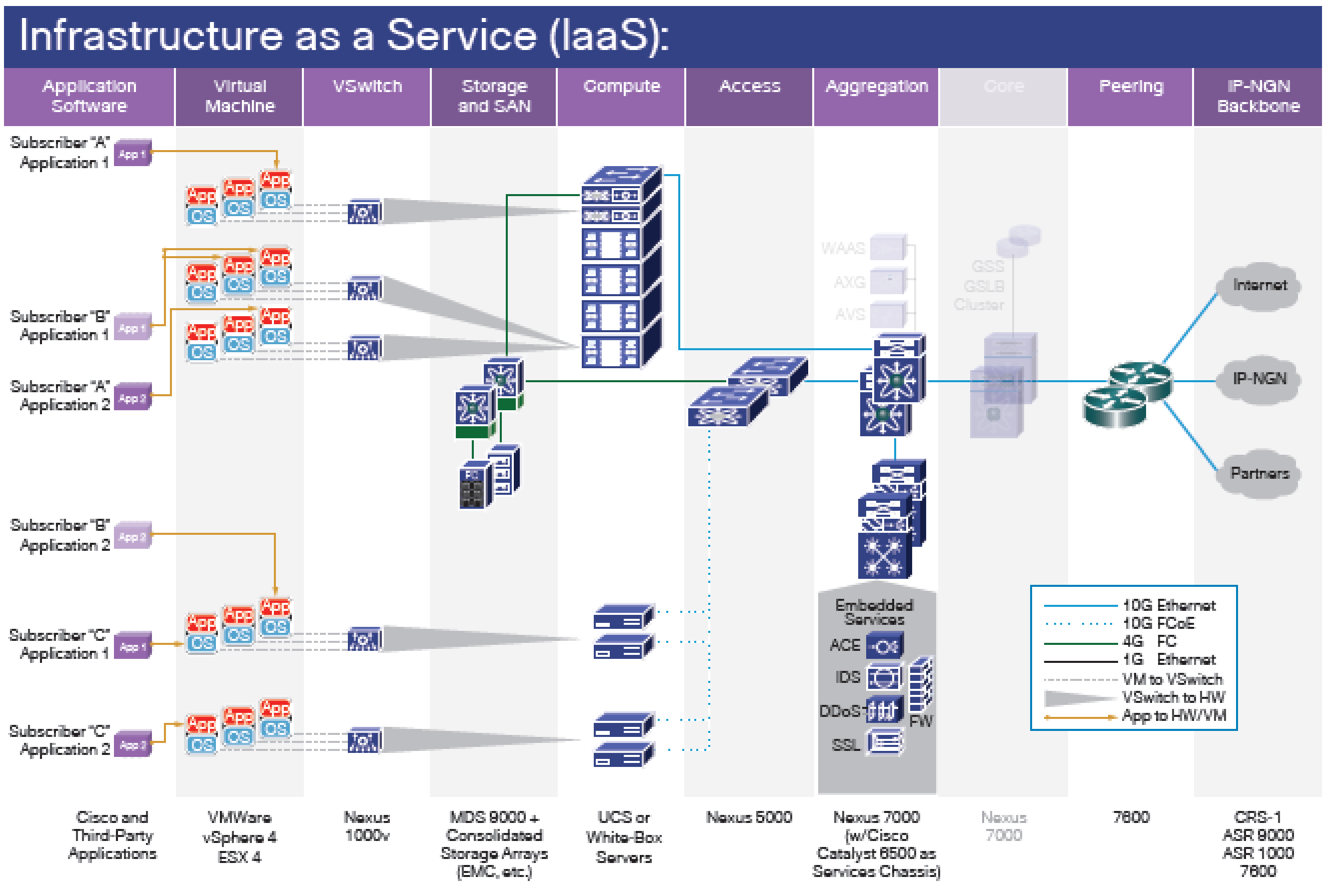
Storage Area Network

SAN network technologies:

- Fibre Channel
- InfiniBand
- Ethernet

Storage devices

Cisco Unified Service Delivery (USD) - Infrastructure as a Service



Cisco Unified Service Delivery (USD) - Infrastructure as a Service (Cont.)

- IaaS enabled by Cisco technology comprises the following product set:
  - Computing: Cisco Unified Computing System, third-party servers, VMware ESX Vi4 Hypervisor
  - Virtual Access: Cisco Nexus™ 1000V
  - Access and Aggregation: Redundant Cisco Nexus 5020 Series Switches, 10G network supporting Fibre Channel over Ethernet (FCoE) to servers
  - Storage Array: Third-party storage
  - Core Switching: Redundant Cisco Nexus 7010 Series Switches, Layer 2 multipathing
  - Services Core: Redundant Cisco Catalyst® 6500-VSS, Cisco Application Control Engine, and Cisco Firewall Service Modules
  - Peering Router: Redundant Cisco 7600 Series Routers; Carrier Ethernet with L2VPN and L3VPN

8. The Vocabulary of Cloud Computing

The Vocabulary of Cloud Computing

- Cloudburst (negative): The failure of a cloud computing environment
- Cloudburst (positive): The dynamic deployment of a software application that runs on internal organizational compute resources to a public cloud to address a spike in demand
- Cloudstorming: The act of connecting multiple cloud computing environments
- Vertical Cloud: A cloud computing environment optimized for use in a particular vertical -- i.e., industry -- or application use case
- Private Cloud: A cloud computing-like environment within the boundaries of an organization and typically for its exclusive usage
- Internal Cloud: A cloud computing-like environment within the boundaries of an organization and typically available for exclusive use by said organization

The Vocabulary of Cloud Computing (Cont.)

- Hybrid Cloud: A computing environment combining both private (internal) and public (external) cloud computing environments. May either be on a continuous basis or in the form of a 'cloudburst‘
- Cloudware: A general term referring to a variety of software, typically at the infrastructure level, that enables building, deploying, running or managing applications in a cloud computing environment
- External Cloud: A cloud computing environment that is external to the boundaries of the organization. Although it often is, an external cloud is not necessarily a public cloud. Some external clouds make their cloud infrastructure available to specific other organizations and not to the public at-large
- Public Cloud: A cloud computing environment that is open for use to the general public, whether individuals, corporations or other types of organizations. Amazon Web Services are an example of a public cloud
- Cloud Provider: An organization that makes a cloud computing environment available to others, such as an external or public cloud

The Vocabulary of Cloud Computing (Cont.)

- Cloud-Oriented Architecture (COA): An architecture for IT infrastructure and software applications that is optimized for use in cloud computing environments. The term is not yet in wide use, and as is the case for the term "cloud computing" itself, there is no common or generally accepted definition or specific description of a cloud-oriented architecture
- Cloud Service Architecture (CSA): A term coined by Jeff Barr, chief evangelist at Amazon Web Services. The term describes an architecture in which applications and application components act as services on the cloud, which serve other applications within the same cloud environment

The Vocabulary of Cloud Computing (Cont.)

- Virtual Private Cloud (VPC): A term coined by Reuven Cohen, CEO and founder of Enomaly. The term describes a concept that is similar to, and derived from, the familiar concept of a Virtual Private Network (VPN), but applied to cloud computing. It is the notion of turning a public cloud into a virtual private cloud, particularly in terms of security and the ability to create a VPC across components that are both within the cloud and external to it
- Cloud Portability: The ability to move applications (and often their associated data) across cloud computing environments from different cloud providers, as well as across private or internal cloud and public or external clouds

[1]: http://www.cloud.com, Cloud Computing Outlook, 2011

[2]: http://cloudtaxonomy.opencrowd.com/taxonomy, 2005

[3]: Sarna, D.E.Y.: Implementing and Developing Cloud Computing Applications, Auerbach Publications, 2011

[4]: Cârstoiu, B.: Cloud SaaS infrastructure, Control and Computers Faculty, University Politehnica of Bucharest, Romania

[5]: Dean J., Ghemawat S.: MapReduce: Simplified Data Processing on Large Clusters, Google, Inc., 2004

[6]: Hurwitz J,. Bloor R., Kaufman M., Halper F.: Cloud Computing For Dummies, Wiley Publishing, Inc., 2010

[7]: Velte A. T., Velte, T. J., Ph.D., Elsenpeter R., Cloud Computing: A Practical Approach, McGraw-Hill, 2010

[8]: http://docs.google.com/: Google Docs

[9]: Amazon Simple Storage Service, Quick Reference Card (Version 2006-03-01), http://awsdocs.s3.amazonaws.com/S3/latest/s3-qrc.pdf

[10]: Running Databases on AWS: http://aws.amazon.com/running_databases/

[11]: http://www.cloudsecurityalliance.org/guidance/csaguide.v2.1.pdf: Security Guidance for Critical Areas of Focus in Cloud Computing V2.1, Cloud Security Alliance, 2009

[12]: http://hadoop.apache.org/ : Apache™ Hadoop™

[13]: http://code.google.com/intl/hu-HU/appengine/docs/whatisgoogleappengine.html: Google App Engine, 2011

Cloud Computing

Managing Cloud Data

Tamás Schubert

1. Data types

Data Types [6]

- The amount of data available for company use is exploding. The nature of data is changing:
  - Data diversity is increasing. Data in the cloud is becoming more diverse. In addition to traditional structured data (revenue, name, and so on), it includes emails, contracts, images, blogs, and so on
  - The amount of data is increasing
    - Videos in YouTube
    - Images in Facebook
    - In traditional data centers, organizations are starting to aggregate huge amounts of data
    - These require massive amounts of computing resources under very controlled circumstances

Data Types (Cont.)

- o Latency requirements are becoming more demanding. Companies are increasingly demanding lower latency for many applications. For example real-time data for Radio Frequency ID tags (RFID). This requires a powerful management environment

- Even in the traditional data centers, organizations aggregate huge amounts of data to solve problems. The cloud can
  - o provide resources to access data on demand and at a much lower price point than the company can
  - o help businesses looking to support the use of data collaboratively across their employees, customers, and business partners
- The cost associated with managing data on demand is a controversial topic in cloud circles
  - o Using data across applications that are in two different clouds can get expensive
  - o It involves real-time synchronization or permanent cloud-hosted data, regardless of the current application demand

2. Securing data in the cloud

Securing Data in the Cloud [6]

- Key areas related to security and privacy of data:
  - Location of data
  - Control of data
  - Secure transfer of data
- More information about security in the cloud: Cloud Security Alliance (www.cloudsecurityalliance.org) [11]
- In the cloud, company data that was previously secured inside of the firewall may now move outside to feed any number of business applications and processes

Securing Data in the Cloud (Cont.)

- Cloud providers must ensure the security and privacy of data, but companies are ultimately responsible for their data. Industry and government regulations created to protect personal and business information still apply even if the data is managed or stored by an outside vendor
- For example, the European Union has implemented a complex set of data protection laws for its member states. In addition, industry regulations (such as the Health Insurance Portability and Accountability Act [HIPAA]) must be followed whether or not your data is in the cloud

2.1. Data location

Securing Data in the Cloud – Data location

After data goes into the cloud, you may not have control over where it's stored geographically. Issues:

- Specific country laws:
  - Laws governing data differ across geographic boundaries
  - The country's legal protections may not apply if data is located outside of the country
  - A foreign government may be able to access the owner's data or keep the owner from fully controlling their data
- Data transfer across country borders:
  - A global company with partners in other countries may be concerned about cross-border transfer of data due to local laws
  - Virtualization makes this an especially tough problem because the cloud provider might not know where the data is at any particular moment

Securing Data in the Cloud – Data location (Cont.)

- Co-mingling of data:
  - The customer's data may be physically stored in a database along with data from other companies
  - This raises concerns about virus attacks or hackers trying to get at another company's data
- Secondary data use:
  - In public cloud situations, the customer's data or metadata may be vulnerable to alternative or secondary uses by the cloud service provider. Without proper controls or service level agreements, data may be used for marketing purposes (and merged with data from other organizations for these alternative uses)

2.2. Data control

Securing Data in the Cloud – Data control

- Controls include the governance policies set in place to make sure that customer's data can be trusted
- The integrity, reliability, and confidentiality of data must be beyond reproach. And this holds for cloud providers too
- Customers must understand what level of controls will be maintained by the cloud provider and consider how these controls can be audited

Securing Data in the Cloud – Data control (Cont.)

- Some different types of controls designed to ensure the completeness and accuracy of data input, output, and processing:
  - Input validation controls to ensure that all data input to any system or application are complete, accurate, and reasonable
  - Processing controls to ensure that data are processed completely and accurately in an application
  - File controls to make sure that data are manipulated accurately in any type of file (structured and unstructured)
  - Output reconciliation controls to ensure that data can be reconciled from input to output
  - Access controls to ensure that only those who are authorized to access the data can do so. Sensitive data must also be protected in storage and transfer

Securing Data in the Cloud – Data control (Cont.)

- Change management controls to ensure that data can't be changed without proper authorization
- Backup and recovery controls. Many security breaches come from problems in data backup. It is important to maintain physical and logical controls over data backup
- Data destruction controls to ensure that when data is permanently deleted it is deleted from everywhere – including all backup and redundant storage sites

2.3. Securing data for transport

Securing Data in the Cloud – Securing data for transport

- At data transport:
  - make sure that no one can intercept your data as it moves from point A to point B in the cloud
  - make sure that no data leaks (malicious or otherwise) from any storage in the cloud
- In the cloud, the journey from point A to point B might take on three different forms:
  - Within a cloud environment
  - Over the public Internet between an enterprise and a cloud provider
  - Between clouds
- The security process includes segregating a company's data from other companies' data and then encrypting it by using an approved method
- A virtual private network (VPN) is one way to manage the security of data during its transport in a cloud environment

Securing Data in the Cloud – Securing data for transport (Cont.)

- The expected level of security may vary, depending on the governance requirements for data
- Customers need to evaluate how the cloud vendor treats the security issues
- Customers need to determine how they can audit the ongoing security processes to make sure that their data remains secure
- Concerns about privacy and security of data have contributed to many companies' interest in developing private cloud environments – where company data remains inside the firewall – and to consider hybrid cloud environments – which incorporate some elements of a private cloud and some elements of a public cloud

3. Large-scale data processing

Large-scale data processing [6]

- The lure of cloud computing is its elasticity
  - Customers can add as much capacity as they need to process and analyze their data
  - The data might be processed on clusters of computers. This means that the processing is occurring across machines
- This model is large-scale, distributed computing and a number of frameworks are emerging to support this model, including
  - MapReduce [5]
  - Apache Hadoop [12]

Large-scale data processing (Cont.)

- MapReduce
  - A software framework introduced by Google to support distributed computing on large sets of data
  - It is designed to take advantage of cloud resources
  - This computing is done across large numbers of computers, called clusters
  - Each cluster is referred to as a node
  - MapReduce can deal with both structured and unstructured data
  - Users specify a map function that processes a key/value pair to generate a set of intermediate pairs and a reduction function that merges these pairs

Large-scale data processing (Cont.)

- Apache Hadoop
  - An open-source distributed computing platform written in Java and inspired by MapReduce
  - It creates a computer pool, each with a Hadoop file system
  - It then uses a hash algorithm to cluster data elements that are similar
  - Hadoop can create a map function of organized key/value pairs that can be output to a table, to memory, or to a temporary file to be analyzed
  - Three copies of the data exist so that nothing gets lost

4. Characteristics of cloud data services

Characteristics of cloud data services [6]

- Data integrity:
  - What controls do customers have to ensure the integrity of their data? For example, are there controls to make sure that all data input to any system or application is complete, accurate, and reasonable? What about any processing controls to make sure that data processing is accurate?
- Compliance
  - Customers are probably aware of any compliance issues particular to their industry
  - Obviously, they need to make sure that their provider can comply with these regulations
- Loss of data
  - What provisions are in the contract if the provider does something to customers' data (loses it because of improper backup and recovery procedures, for instance)?

Characteristics of cloud data services (Cont.)

- Business continuity plans
  - What happens if the cloud vendor’s data center goes down? What business continuity plans does the provider have in place: How long will it take the provider to get data back up and running?
- Uptime
  - The provider might tell customers that they will be able to access their data 99.999 percent of the time. Does this uptime include scheduled maintenance?
- Data storage costs
  - Pay-as-you-go and no-capital-purchase. But how much will it cost to move data into the cloud? What about other hidden integration costs? How much will it cost to store data?

Characteristics of cloud data services (Cont.)

- Contract termination
  - How will data be returned if the contract is terminated? How the provider would destroy data to make sure that it isn't floating around in the cloud
- Data ownership
  - Who owns data after it goes into the cloud? Some service providers might want to take customers' data, merge it with other data, and do some analysis
- Switching vendors
  - If customers create applications with one cloud vendor and then decide to move to another vendor, how difficult will it be to move their data? How interoperable are the services? Some vendors may have proprietary APIs and it might be costly to switch

5. Cloud Storage Providers

Cloud Storage Providers [7]

- Terms used with databases in the cloud
  - Database as a service

    describes vendors that offer clients a hosted database solution. The database is in the cloud, but customers know that the cloud provider is managing it and customers know where the data center is physically located. Customers don't pay for the hardware and pay on a pay-per-use basis
  - Cloud database

    term is used when the database is in the cloud, meaning that customers may not know where the data physically resides

Cloud Storage Providers (Cont.)

- Some examples of several hundreds of storage providers:
  - Amazon and Nirvanix are the biggest industry players
  - Google Bigtable and Gdrive
  - Microsoft Cloud-based SQL
  - EMC Greenplum
  - IBM Blue Cloud

5.1. Databases on Amazon Web Services

Databases on Amazon Web Services (AWS) [10]

- Amazon Web Services provides a number of storage and database alternatives for developers
  - Amazon Simple Storage Service (S3) – is storage for the Internet. Amazon S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the web. It gives any developer access to the same highly scalable, reliable, secure, fast, inexpensive infrastructure that Amazon uses to run its own global network of web sites
  - Amazon SimpleDB – provides simple index and query capabilities with seamless scalability
  - Amazon Relational Database Service (Amazon RDS) – enables users to run a fully featured relational database while offloading database administration
  - Amazon EC2 Relational Database AMIs – are using one of Amazon's many relational database AMIs on Amazon EC2 and Amazon EBS that allow users to operate their own relational database in the cloud
- There are important differences between these alternatives that may make one more appropriate for the customer's use case

Amazon Simple Storage Service (S3) [7]

- The best-known cloud storage service is Amazon's Simple Storage Service (S3)
- Launched in 2006
- Amazon S3 is designed to make web-scale computing easier for developers
- Amazon S3 provides a simple web services interface that can be used to store and retrieve any amount of data, at any time, from anywhere on the Web
- Highly scalable data storage infrastructure

Amazon Simple Storage Service (S3) (Cont.)

- Amazon S3 is intentionally built with a minimal feature set that includes the following functionality:
  - Write, read, and delete objects containing from 1 byte to 5 gigabytes of data each. The number of objects that can be stored is unlimited
  - Each object is stored and retrieved via a unique developer-assigned key
  - Objects can be made private or public, and rights can be assigned to specific users
  - Uses standards-based REST and SOAP interfaces designed to work with any Internet-development toolkit

Amazon Simple Storage Service (S3) – Design Requirements

- Scalable – Amazon S3 can scale in terms of storage, request rate, and users to support an unlimited number of web-scale applications
- Reliable – Store data durably, with 99.99 percent availability. Amazon says it does not allow any downtime
- Fast – Amazon S3 was designed to be fast enough to support high-performance applications. Server-side latency must be insignificant relative to Internet latency. Any performance bottlenecks can be fixed by simply adding nodes to the system
- Inexpensive – Amazon S3 is built from inexpensive commodity hardware components. As a result, frequent node failure is the norm and must not affect the overall system. It must be hardware-agnostic, so that savings can be captured as Amazon continues to drive down infrastructure costs
- Simple – Building highly scalable, reliable, fast, and inexpensive storage is difficult. Doing so in a way that makes it easy to use for any application anywhere is more difficult. Amazon S3 must do both

Amazon Simple Storage Service (S3) – Design Principles

- Decentralization – It uses fully decentralized techniques to remove scaling bottlenecks and single points of failure
- Autonomy – The system is designed such that individual components can make decisions based on local information
- Local responsibility – Each individual component is responsible for achieving its consistency; this is never the burden of its peers
- Controlled concurrency – Operations are designed such that no or limited concurrency control is required

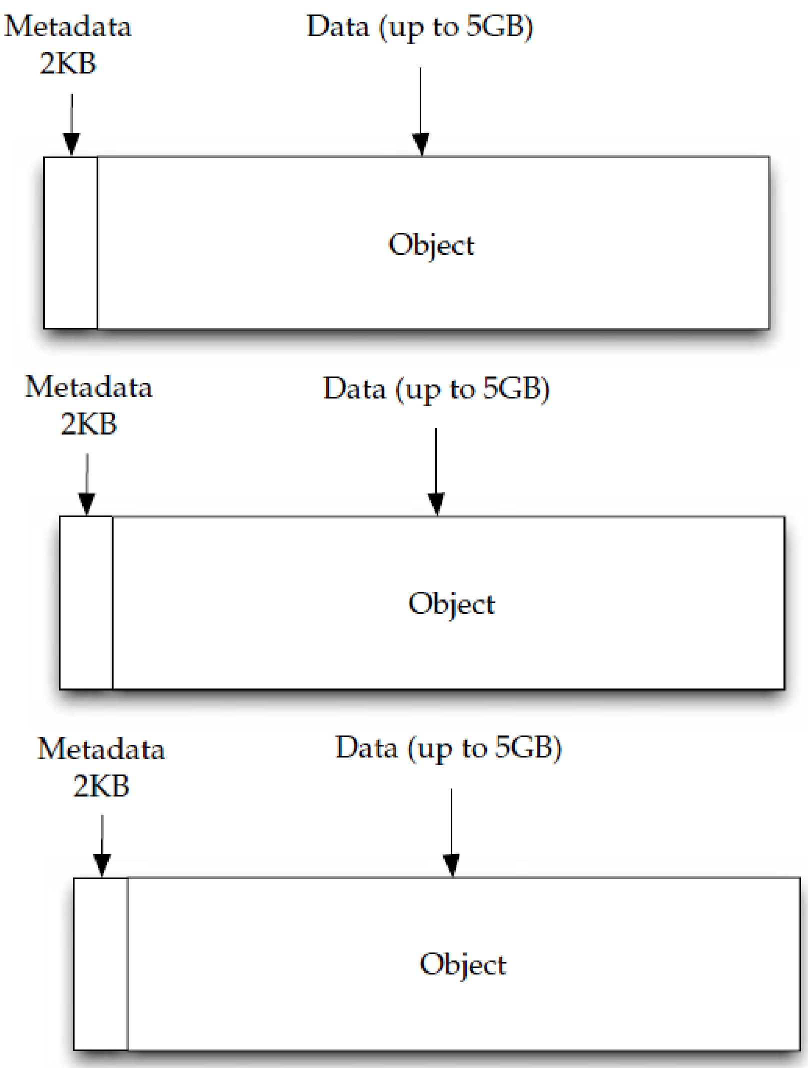
Amazon Simple Storage Service (S3) – Design Principles (Cont.)

- Failure toleration – The system considers the failure of components to be a normal mode of operation and continues operation with no or minimal interruption
- Controlled parallelism – Abstractions used in the system are of such granularity that parallelism can be used to improve performance and robustness of recovery or the introduction of new nodes
- Small, well-understood building blocks – Do not try to provide a single service that does everything for everyone, but instead build small components that can be used as building blocks for other services
- Symmetry – Nodes in the system are identical in terms of functionality, and require no or minimal node-specific configuration to function
- Simplicity – The system should be made as simple as possible, but no simpler

Amazon Simple Storage Service (S3) – How S3 Works

- S3's design aims to provide scalability, high availability, and low latency at commodity costs
- S3 stores arbitrary objects at up to 5GB in size, and each is accompanied by up to 2KB of metadata
- Objects are organized by buckets. Each bucket is owned by an AWS (Amazon Web Services) account and the buckets are identified by a unique, user-assigned key
- Buckets and objects are created, listed, and retrieved using either a REST-style or SOAP interface
- Objects can also be retrieved using the HTTP GET interface or via BitTorrent
- An access control list restricts who can access the data in each bucket
- Bucket names and keys are formulated so that they can be accessed using HTTP
- Requests are authorized using an access control list associated with each bucket and object, for instance:
    - http://s3.amazonaws.com/examplebucket/examplekey
    - http://examplebucket.s3.amazonaws.com/examplekey

Amazon Simple Storage Service (S3) – How S3 Works (Cont.)



Multiple objects are stored in buckets in Amazon S3

Amazon Simple Storage Service (S3) – How S3 Works (Cont.)

- The Amazon AWS Authentication tools allow the bucket owner to create an authenticated URL with a set amount of time that the URL will be valid
- For instance, the owner could create a link to his data on the cloud, give that link to someone else who could access the owner's data for an amount of time the owner predetermine, be it 10 minutes or 10 hours
- Bucket items can also be accessed via a BitTorrent feed, enabling S3 to act as a seed for the client. Buckets can also be set up to save HTTP log information to another bucket. This information can be used for later data mining

Amazon Simple Storage Service (S3) – Quick Reference Card [9]

Service Operations

GET Service
Returns a list of all buckets owned by the authenticated request sender.

```
GET / HTTP/1.1
Host: s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
```

Bucket Operations

PUT Bucket
Creates a new bucket belonging to the account of the authenticated request sender. Optionally, you can specify a EU (Ireland) or US-West (N. California) location constraint.

```
PUT / HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
Content-Length: {0 | length}
[<CreateBucketConfiguration>
  <LocationConstraint>EU</LocationConstraint>
</CreateBucketConfiguration>]
```

GET Bucket
Lists information about the objects in a bucket for a user that has read access to the bucket.

```
GET ?prefix=prefix&marker=marker&max-keys=max-keys&delimiter=delimiter HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
```

GET Bucket location
Lists the location constraint of the bucket for the bucket owner.

```
GET /?location HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
```

DELETE Bucket
Deletes the specified bucket. All objects in the bucket must be deleted before the bucket itself can be deleted.

```
DELETE / HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
```

POST Object
For more information about POST, refer to the *Amazon Simple Storage Service Developer Guide*.

Object Operations

GET Object
Gets an object for a user that has read access to the object.

```
GET /destinationObject HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
[Range:bytes=byte_range]
[x-amz-metadata-directive: metadata_directive]
[x-amz-if-match: etag]
[x-amz-if-none-match: etag]
[x-amz-if-unmodified-since: time_stamp]
[x-amz-if-modified-since: time_stamp]
```

PUT Object
Adds an object to a bucket for a user that has write access to the bucket. A success response indicates the object was successfully stored; if the object already exists, it will be overwritten.

```
PUT /destinationObject HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
Content-Length: length
Content-MD5: md5_digest
Content-Type: type
Content-Disposition: object_information
Content-Encoding: encoding
Cache-Control: caching
Expires: expiration
<request metadata>
```

COPY Object
Copies an object for a user that has write access to the bucket and read access to the object. All headers prefixed with x-amz- must be signed, including x-amz-copy-source.

```
PUT /destinationObject HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
x-amz-copy-source: /source_bucket/sourceObject
[x-amz-metadata-directive: metadata_directive]
[x-amz-copy-source-if-match: etag]
[x-amz-copy-source-if-none-match: etag]
[x-amz-copy-source-if-unmodified-since:
time_stamp]
[x-amz-copy-source-if-modified-since: time_stamp]
<request metadata>
```

HEAD Object
Retrieves information about an object for a user with read access without fetching the object.

```
HEAD /destinationObject HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
```

DELETE Object
Deletes the specified object. Once deleted, there is no method to restore or undelete an object.

```
DELETE / HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
Date: date
Authorization: AWS AWSAccessKeyId:signature
```

POST Object
Adds an object to a bucket using forms.

```
POST / HTTP/1.1
Host: destinationBucket.s3.amazonaws.com
User-Agent: browser_data
Accept: file_types
Accept-Language: locales
Accept-Encoding: encoding
Accept-Charset: character_set
Keep-Alive: 300
Connection: keep-alive
Content-Type: multipart/form-data; boundary=----------ID
Content-Length: length
<multiform_data>
```

Note: For more information about the multiform data, refer to the *Amazon Simple Storage Service Developer Guide*.

Amazon Simple Storage Service (S3) – Quick Reference Card (Cont.)

Miscellaneous

Bucket Name Restrictions

Amazon S3 bucket names must:

- Only contain lowercase letters, numbers, periods (.), and dashes (-)
- Start with a number or letter
- Be between 3 and 63 characters long
- Not be in an IP address style (e.g., "192.168.5.4")
- Not end with a dash
- Not contain dashes next to periods (e.g., "my-.bucket.com" and "my.-bucket" are invalid)

Note: Although legacy bucket names can be up to 255 characters, include underscores, and end with a dash, they are not recommended.

REST Request Signature

Construct a signature by making an RFC2104 HMAC-SHA1 of the following and converting it to Base64.

Item	Example
HTTP-Verb + "\n" +	GET\n
[Content-MD5] + "\n" +	4gJE4saaMU4BqNR0kLY+lw==\n or \n
[Content-Type] + "\n" +	image/jpeg\n or \n
Date + "\n" +	Tue, 6 Mar 2007 19:42:41 +0000\n
[CanonicalisedAmsHeaders +/n]	x-ams-acl:public-read\n x-ams-meta-checksum:crc32\n
[CanonicalisedResource]	/mybucket/photos/lolcats.jpg

The content cannot contain whitespaces and \n represents Unicode code point U+000A.

Versioning

GET Bucket Object versions

Lists metadata about all of the versions of objects in a bucket.

GET /?versions HTTP/1.1
Host: BucketName.s3.amazonaws.com

GET Bucket versioning

Returns the versioning state of a bucket.

GET /?versioning HTTP/1.1
Host: myBucket.s3.amazonaws.com

PUT Bucket versioning

Sets the versioning state of an existing bucket.

PUT /?versioning HTTP/1.1
Host: bucket.s3.amazonaws.com
<VersioningConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
<Status>Enabled</Status>
</VersioningConfiguration>

DELETE Object versionId

Removes a specific object version.

DELETE /my-third-image.jpg?
Host: bucket.s3.amazonaws.com
versionId=<versionID>

GET Object versionId

Returns a specific object version.

GET /myObject?versionId=<versionID>
Host: bucket.s3.amazonaws.com

HEAD Object versionId

Returns metadata of a specific object version.

HEAD /myObject?versionId=<versionID>
Host: bucket.s3.amazonaws.com

PUT Object copy versionId

Copies a specific object version.

PUT /my-second-image.jpg HTTP/1.1
Host: bucket.s3.amazonaws.com
Date: Wed, 28 Oct 2009 22:32:00 GMT
x-amz-copy-source: /bucket/my-image.jpg?versionId=<versionID>

Amazon SimpleDB [10]

- For database implementations that do not require a relational model, and that principally demand index and query capabilities
- Amazon SimpleDB eliminates the administrative overhead of running a highly-available production database, and is unbound by the strict requirements of a RDBMS
- Data items are stored and queried via simple web services requests, and Amazon SimpleDB does the rest
- Amazon SimpleDB is handling infrastructure provisioning, software installation and maintenance
- Amazon SimpleDB automatically indexes data, creates geo-redundant replicas of the data to ensure high availability, and performs database tuning on customers' behalf

Amazon SimpleDB (Cont.)

- For workloads with large data sets or throughput requirements, data set and requests can be spread across additional machine resources by creating additional Domains
- Amazon SimpleDB will charge customers only for the resources actually consumed in storing data and serving requests
- Amazon SimpleDB doesn’t enforce a rigid schema for data. This gives customers flexibility – if their business changes, they can easily reflect these changes in Amazon SimpleDB without any schema updates or changes to the database code

Amazon SimpleDB (Cont.)

- Amazon SimpleDB is not a relational database, and does not offer some features needed in certain applications, e.g. complex transactions or joins
- The use of Amazon SimpleDB is recommend for customers who:
  - Principally utilize index and query functions rather than more complex relational database functions
  - Don't want any administrative burden at all in managing their structured data
  - Want a service that scales automatically up or down in response to demand, without user intervention
  - Require the highest availability and can't tolerate downtime for data backup or software maintenance

Amazon Relational Database Service (Amazon RDS) [10]

- For database implementations requiring relational storage and built on MySQL or Oracle
- Amazon RDS automates common administrative tasks
- Offers feature rich functionality that enhances database availability and scalability, significantly reducing the complexity of managing and the cost of owning database assets

Amazon Relational Database Service (Amazon RDS) (Cont.)

- Amazon RDS automatically backs up your database and maintains database software
- Using the Multi-AZ (Availability Zone) deployment option (currently available for MySQL only), you can have Amazon RDS provision and maintain a synchronous „standby” replica of the database in a different Availability Zone, enhancing the database availability
- Additionally, the Read Replica feature available for MySQL, enables users to exploit MySQL native replication and setup replicas in minutes for read scaling
- Amazon RDS for MySQL manages the replication and replicas for users
- Users are able to scale the compute resources or storage capacity associated with the relational database instance of the user via few clicks or a single API call

Amazon Relational Database Service (Amazon RDS) (Cont.)

- Amazon RDS is recommended for customers who:
  - Have existing or new applications, code, or tools that require a relational database
  - Want native access to a MySQL or Oracle database, but prefer to offload the infrastructure management and database administration to AWS
  - Want to exploit the Multi-AZ and Read Replica features (currently available for MySQL only) to achieve enhanced database availability and read scalability
  - Like the flexibility of being able to scale their database compute and storage resources with an API call, and only pay for the infrastructure resources they actually consume

Amazon EC2 Relational Databases AMIs [10]

- Developers may use a number of leading relational databases on Amazon EC2
- An Amazon EC2 instance can be used to run a database, and the data can be stored within an Amazon EBS volume
- Amazon EBS is a fast and reliable persistent storage feature of Amazon EC2
- With Amazon EC2 Relational Database AMIs, developers avoid the friction of infrastructure provisioning while gaining access to a variety of standard database engines
- Amazon EC2 Relational Database AMIs enable developers to skip the infrastructure and hardware provisioning typically associated with installing a new database server
- Customers retain complete control over the administrative and tuning tasks associated with running a database server

Amazon EC2 Relational Databases AMIs (Cont.)

- Amazon EC2 Relational Database AMIs recommend for customers who:
  - Wish to select from a wide variety of database engines
  - Want to exert complete administrative control over their database server
- Installing Relational Databases via AMIs
  - An Amazon Machine Image (AMI) is an encrypted machine image stored in Amazon S3
  - It contains all the information necessary to boot instances of your software
  - Many existing AMIs already come packaged with relational databases

Amazon EC2 Relational Databases AMIs (Cont.)

- Relational Database AMIs:
  - IBM DB2
  - IBM Informix
  - Oracle
  - MySQL
  - Microsoft SQL Server
  - PostgreSQL
  - Sybase
  - EnterpriseDB

5.2. Google Bigtable Datastore

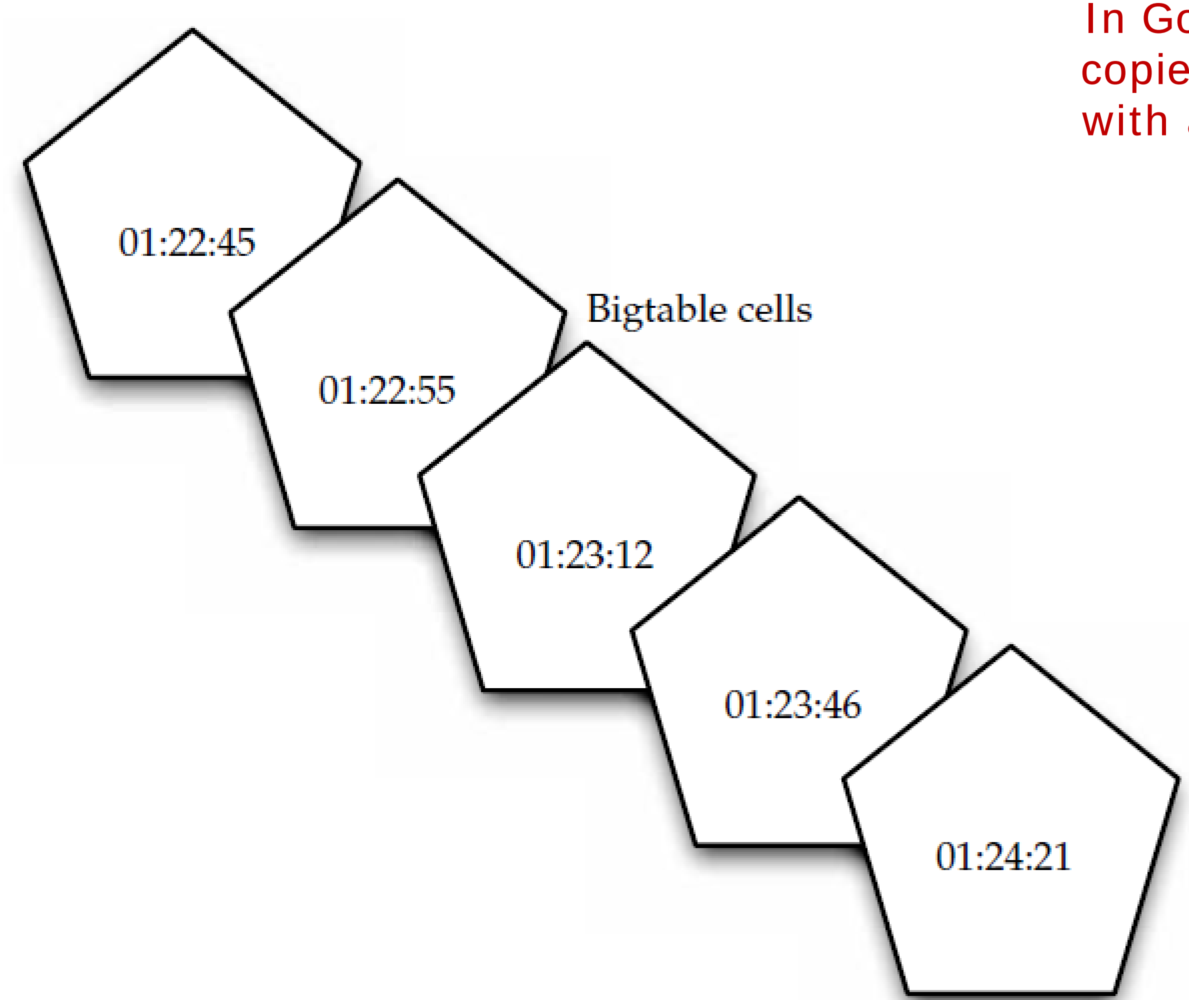
Google Bigtable Datastore [7]

- Bigtable is a database that is capable of handling numerous users on an on-demand basis
- Bigtable went public in April 2008
- Bigtable was developed with very high speed, flexibility, and extremely high scalability in mind
- Bigtable is a distributed system for storing structured data at Google
- A Bigtable database can be petabytes in size and span thousands of distributed servers
- Bigtable is available to developers as part of the Google App Engine, their cloud computing platform

Google Bigtable Datastore – How Bigtable Works (Cont.)

- Google describes Bigtable as a fast and extremely scalable DBMS
- This allows Bigtable to scale across thousands of commodity servers that can collectively store petabytes of data
- Each table in Bigtable is a multidimensional sparse map. That is, the table is made up of rows and columns, and each cell has a timestamp
- Multiple versions of a cell can exist, each with a different timestamp. With this stamping, users can select certain versions of a web page, or delete cells that are older than a given date and time

Google Bigtable Datastore – How Bigtable Works (Cont.)



In Google Bigtable, multiple copies of a cell exist, each with a different timestamp

Google Bigtable Datastore – How Bigtable Works (Cont.)

- Because the tables are so large, Bigtable splits them at row boundaries and saves them as tablets
- Each tablet is about 200MB, and each server houses 100 tablets. Given this, data from a database is likely to be stored in many different servers—maybe not even in the same geographic location
- This architecture also allows for load balancing. If one table is getting a lot of queries, it can remove other tablets or move the busy table to another machine that is not as busy. Also, if a machine fails, since the tablet is spread to different machines, users may not even notice the outage
- When a machine fills up, it compresses some tablets using a Google-proprietary technique. On a minor scale, only a few tablets are compressed. On a large scale, entire tablets are compressed, freeing more drive space

Google Bigtable Datastore – How Bigtable Works (Cont.)

- Bigtable tablet locations are stored in cells, and looking them up is a three-tiered system
- Clients point to the META0 table. META0 then keeps track of many tables on META1 that contain the locations of the tablets. Both META0 and META1 make use of prefetching and caching to minimize system bottlenecks
- Issues
  - While Bigtable is a robust tool, developers have been cautious about using it. Because it is a proprietary system, they get locked into Google
  - On the other hand, Google App Engine and Bigtable are affordable, costing about the same as Amazon's S3

[1]: http://www.cloud.com, Cloud Computing Outlook, 2011

[2]: http://cloudtaxonomy.opencrowd.com/taxonomy , 2005

[3]: Sarna, D.E.Y.: Implementing and Developing Cloud Computing Applications, Auerbach Publications, 2011

[4]: Cârstoiu, B.: Cloud SaaS infrastructure, Control and Computers Faculty, University Politehnica of Bucharest, Romania

[5]: Dean J., Ghemawat S.: MapReduce: Simplified Data Processing on Large Clusters, Google, Inc., 2004

[6]: Hurwitz J,. Bloor R., Kaufman M., Halper F.: Cloud Computing For Dummies, Wiley Publishing, Inc., 2010

[7]: Velte A. T., Velte, T. J., Ph.D., Elsenpeter R., Cloud Computing: A Practical Approach, McGraw-Hill, 2010

[8]: http://docs.google.com/: Google Docs

[9]: Amazon Simple Storage Service, Quick Reference Card (Version 2006-03-01), http://awsdocs.s3.amazonaws.com/S3/latest/s3-qrc.pdf

[10]: Running Databases on AWS: http://aws.amazon.com/running_databases/

[11]: http://www.cloudsecurityalliance.org/guidance/csaguide.v2.1.pdf: Security Guidance for Critical Areas of Focus in Cloud Computing V2.1, Cloud Security Alliance, 2009

[12]: http://hadoop.apache.org/ : Apache™ Hadoop™

[13]: http://code.google.com/intl/hu-HU/appengine/docs/whatisgoogleappengine.html: Google App Engine, 2011

Cloud Computing

Platform as a Service – PaaS

Tamás Schubert

1. Platform as a Service

Platform as a Service (PaaS)

- *Platform includes the whole lifecycle of the development, test, deployment and operation of CC applications*
- The whole lifecycle is based on Cloud Computing
- Key components of PaaS:
  - The development, test, deployment, run and management of the cloud applications is operated in the same integrated environment (cost is decreasing, quality and availability are increasing)
  - The user comfort, response time and quality must be ensured without any compromise (same quality expectation as in the traditional applications). Software download, plug-in installation and local program run can't influence the use of the cloud application
  - The realization of built-in scalability, reliability and security without extra development, configuration and cost. Automatic multi-tenancy. The storage and the transmission of data, and the financial transactions should be secure during the whole lifecycle of the application

Platform as a Service (PaaS) (Cont.)

- Built-in integration with Web Services and databases. Link services running at distant locations and link data stored at distant locations
- Support cooperation of developers and developer groups. The platform must ensure the cooperation during the whole lifecycle (development, test, documentation, deployment and operation) of the application without any special configuration
- Deep monitoring built into the application, which records the activity of the users, the faults and the performance issues. The recorded information helps the developers in the enhancement of the applications, and in the exploration of new user expectations

2. Google App Engine – Cloud Computing Platform [13]

Characteristics of Google App Engine

- A Cloud Computing Platform
- Running Web application on the infrastructure of Google
- Development and management of simple applications
- Good scalability (bandwidth and storage)
- Developers don't need to manage the physical infrastructure
- Customers can use the free appspot.com Google or their own domain names
- The use of the application can be limited (e.g. only for employees)
- App Engine costs nothing to get started
- All applications can use up to 500 MB of storage and enough CPU and bandwidth to support an efficient app serving around 5 million page views a month, absolutely free
- In a case of billing, the free limits are raised, and customers only pay for resources they use above the free levels

Application environment

- Google App Engine makes it easy to build an application that runs reliably, even under heavy load and with large amounts of data
- App Engine includes features:
  - Dynamic web serving, with full support for common web technologies
  - Persistent storage with queries, sorting and transactions
  - Automatic scaling and load balancing
  - APIs for authenticating users and sending email using Google Accounts
  - A fully featured local development environment that simulates Google App Engine on the developer's computer
  - Task queues for performing work outside of the scope of a web request
  - Scheduled tasks for triggering events at specified times and regular intervals
- Application development using Python, Java and Go programming languages
- Each environment provides standard protocols and common technologies for web application development

Sandbox

- Applications run in a secure environment that provides limited access to the underlying operating system
- These limitations allow App Engine to distribute web requests for the application across multiple servers, and start and stop servers to meet traffic demands
- The sandbox isolates applicatiosn in their own secure, reliable environment that is independent of the hardware, operating system and physical location of the web server

Sandbox (Cont.)

- Examples of the limitations of the secure sandbox environment:
  - An application can only access other computers on the Internet through the provided URL fetch and email services. Other computers can only connect to the application by making HTTP (or HTTPS) requests on the standard ports
  - An application cannot write to the file system. An app can read files, but only files uploaded with the application code.
  - The app must use the App Engine datastore, memcache or other services for all data that persists between requests
  - Application code only runs in response to a web request, a queued task, or a scheduled task, and must return response data within 30 seconds in any case. A request handler cannot spawn a sub-process or execute code after the response has been sent

Python Runtime Environment

- Implement applications using the Python programming language, and run it on an optimized Python interpreter
- App Engine includes rich APIs and tools for Python web application development (feature rich data modeling API, an easy-to-use web application framework, and tools for managing and accessing app's data)
- Wide variety of mature libraries and frameworks for Python web application development, such as Django
- Python runtime environment uses Python version 2.5.2
- Additional support for Python 3 is being considered for a future release
- Python environment includes the Python standard library
- Not all of the library's features can run in the sandbox environment (e.g. creating socket, writing to file, etc. raise an exception)

Python Runtime Environment (Cont.)

- Application code written for the Python environment must be written exclusively in Python. Extensions written in the C language are not supported
- Rich Python APIs for the datastore, Google Accounts, URL fetch, and email services
- App Engine also provides a simple Python web application framework called webapp to make it easy to start building applications
- Developers can upload other third-party libraries with their application implemented in pure Python

Java Runtime Environment

- Develop application for the Java runtime environment using common Java web development tools and API standards
- Applications interacts with the environment using the Java Servlet standard, and can use common web application technologies such as JavaServer Pages (JSPs)
- Java runtime environment uses Java 6
- App Engine Java SDK supports developing apps using either Java 5 or 6
- The platform and libraries environment includes the Java SE Runtime Environment (JRE) 6
- Restrictions of the sandbox environment are implemented in the JVM
- An app can use any JVM bytecode or library feature that does not exceed the sandbox restrictions (open a socket or write to a file will throw a runtime exception)

Java Runtime Environment (Cont.)

- Applications access most App Engine services using Java standard APIs
- Java SDK includes implementations of the Java Data Objects (JDO) and Java Persistence API (JPA) interfaces for the use App Engine datastore
- Applications can use the JavaMail API to send email messages with the App Engine Mail service
- Java.net HTTP APIs access the App Engine URL fetch service
- App Engine includes low-level APIs for its services to implement additional adapters, or to use directly from the application (datastore, memcache, URL fetch, mail, images and Google Accounts APIs)
- Typically, Java developers use the Java programming language and APIs to implement web applications for the JVM
- With the use of JVM-compatible compilers or interpreters, developers can also use other languages to develop web applications (e.g.: JavaScript, Ruby, or Scala)

Datastore

- Distributed data store: query language, transaction management
- Data stores can grow dynamically
- Developers have the choice between two different data storage options differentiated by their availability and consistency guarantees:
  - High Replication Datastore
  - Master/Slave Datastore
- Not a traditional relational database management system
- Data objects (entities) have kind (type) and a set of properties
- The objects can form a complex structure
- The data objects can be queried by their type, can be filtered and sorted by their value of the properties
- Datastore entities are „chemaless”
- The structure of data entities is provided by and enforced by the application code

Datastore (Cont.)

- The Java JDO/JPA interfaces and the Python datastore interface include features for applying and enforcing structure within the application
- Application can also access the datastore directly to apply as much or as little structure as it needs
- The datastore is strongly consistent and uses optimistic concurrency control
- An update of a entity occurs in a transaction that is retried a fixed number of times if other processes are trying to update the same entity simultaneously
- Application can execute multiple datastore operations in a single transaction which either all succeed or all fail, ensuring the integrity of data
- The datastore implements transactions across its distributed network using „entity groups”
- A transaction manipulates entities within a single group. Entities of the same group are stored together for efficient execution of transactions
- Application can assign entities to groups when the entities are created

Datastore – High Replication Datastore (HDR)

- HRD is the default for new applications
- HRD is a highly available, highly reliable storage solution
- It remains available for reads and writes during planned downtime and is extremely resilient in the face of catastrophic failure – but it costs more than the master/slave option
- Data in the HRD is replicated across data centers using the Paxos algorithm
- It provides the highest level of availability for reads and writes and uses approximately three times the storage and CPU cost of the master/slave option

Datastore – Master/Slave Datastore

- Master-slave replication system asynchronously replicates data to other data centers on write operation
- Since only one datacenter is the master for writing at any given time, this option offers strong consistency for all reads and queries
- Data may be temporarily unavailable during data center issues or planned downtime
- The master/slave datastore is suitable only for a limited class of applications that:
  - Do not require high availability of data
  - Can tolerate spikes in datastore latency
  - Need to incur the lowest possible serving cost

Datastore – Differences between the HRD and master/slave datastores

	High Replication	Master/Slave
Cost		
Storage	1x	1/3x
Put/Delete CPU	1x	5/8x
Get CPU	1x	1x
Query CPU	1x	1x
Performance		
Put/Delete Latency	1/2x–1x	1x
Get Latency	1x	1x
Query Latency	1x	1x
Consistency		
Get/Put/Delete	Strong	Strong
Most Queries	Eventual	Strong
Ancestor Queries	Strong	Strong
Occasional Planned Read-Only Period	No	Yes
Unplanned Downtime	Extremely rare. No data loss.	Rare. Possible to lose a small % of writes that occurred near the downtime (recoverable after event).

Google Accounts

- App Engine integrates applications with Google Accounts for user authentication
- Users can login to the application with their own account ID, and they can access their e-mail, etc.
- No need to build an independent identification system for the applications

Services

App Engine services enables developers to perform common operations when managing their application. The following APIs are provided to access these services:

- URL Fetch
  - Applications can access resources on the Internet, such as web services or other data, using App Engine's URL fetch service
  - The URL fetch service retrieves web resources using the same high-speed Google infrastructure that retrieves web pages for many other Google products
- Mail
  - Applications can send email messages using App Engine's mail service
  - The mail service uses Google infrastructure to send email messages

Services (Cont.)

- Memcache
  - The Memcache service provides application with a high performance in-memory key-value cache that is accessible by multiple instances of the application
  - Memcache is useful for data that does not need the persistence and transactional features of the datastore (temporary data or data copied from the datastore to the cache for high speed access)
- Image Manipulation
  - The Image service lets application manipulate images
  - With this API, users can resize, crop, rotate and flip images in JPEG and PNG formats

Development Workflow

- App Engine software development kits (SDKs) for Java, Python, and Go each include a web server application that emulates all of the App Engine services on your local computer
- Each SDK includes all of the APIs and libraries available on App Engine
- Web server also simulates the secure sandbox environment, including checks for attempts to access system resources disallowed in the App Engine runtime environment
- Each SDK includes a tool to upload your application to App Engine
- Once developers have created their application's code, static files and configuration files, they run the tool to upload the data
- The tool prompts developers for their Google account email address and password
- When developers build a new major release of an application, they can upload the new release as a new version
- The old version will continue to serve users until developers switch to the new version
- Developers can test the new version on App Engine while the old version is still running

Development Workflow (Cont.)

- Java SDK runs on any platform with Java 5 or Java 6
- SDK is available as a Zip file
- In the Eclipse development environment, developers can use the Google Plugin for Eclipse to create, test and upload App Engine applications
- SDK also includes command-line tools for running the development server and uploading the application
- Python SDK is implemented in pure Python, and runs on any platform with Python 2.5, including Windows, Mac OS X and Linux
- SDK is available as a Zip file, and installers are available for Windows and Mac OS X

Development Workflow (Cont.)

- Administration Console is the web-based interface for managing applications running on App Engine:
  - Create new applications
  - Configure domain names
  - Change which version of the application is live
  - Examine access and error logs
  - Browse an application's datastore

3. Application development in Python [13]

Application development in Python

- Setting up the development environment:
  - Download and install the Python SDK (Python 2.5)
  - Download and install the App Engine Software Development Kit (SDK) (ensures the runtime environment for the development)
- Develop and test the application using Python language
- Upload the application into the Google App Engine
- The applications can be managed by the Web-based Administrator Console in the App Engine:
  - Create new application
  - Manage versions
  - Configure domain names
  - Monitor the application usage
  - Investigate the error log
  - Browse the datastore

3.1. Application 1

Application 1 – Hello, world!

- Python App Engine applications communicate with the web server using the CGI standard
- When the server receives a http request, it runs the application with the request data in environment variables and on the standard input stream (POST data)
- To respond, the application writes the response to the standard output stream, including HTTP headers and content
- The first application displays a short message: Hello, World!

Application 1 – Creating Request the Handler

- Create a directory: C:\Google\schappx001
- All files for this application reside in this directory
- Inside the schappx001 directory, create a file named schappx001.py with the following contents:

```
print 'Content-Type: text/plain'
print ''
print ‘Hello, World!'
```

- This Python script responds to a request with an HTTP header that describes the content, a blank line, and the message: Hello, world!

Application 1 – Creating the Configuration File

- An App Engine application has a configuration file called app.yaml
- Among other things, this file describes which handler scripts should be used for which URLs
- Inside the schappx001 directory, create a file named app.yaml with the following contents:

```
application: schappx001        -- application name (lower case letters)
version: 1                     -- application version
runtime: python
api_version: 1

handlers:
- url: /.*
  script: schappx001.py        -- reference to request handler
```

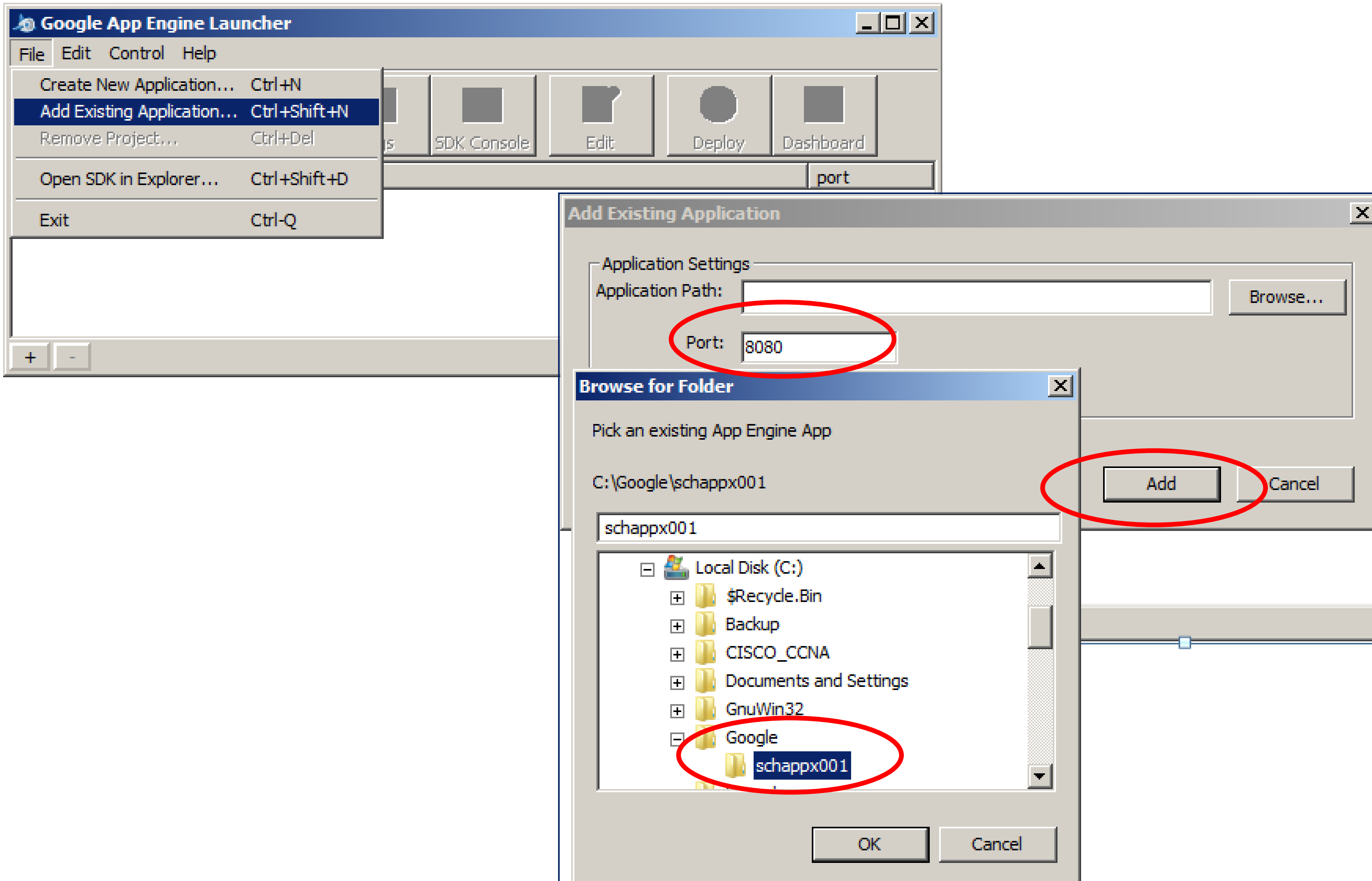
Application 1 – Creating the Configuration File (Cont.)

- Application identifier: schappx001
- When you register the application with App Engine in the final step, a unique identifier should be selected
- version: 1 – Changing version number before uploading, a new versions of the application software will be created
- App Engine retain previous versions, and developers may roll back to a previous version using the Administrative Console
- api_version: 1 – This code runs in the python runtime environment, version "1"
- - url: /.* - Every request to a URL whose path matches the regular expression /.* (all URLs) should be handled by the schappx001.py script
- script: schappx001.py – Reference to the request handler
- The syntax of this file is YAML

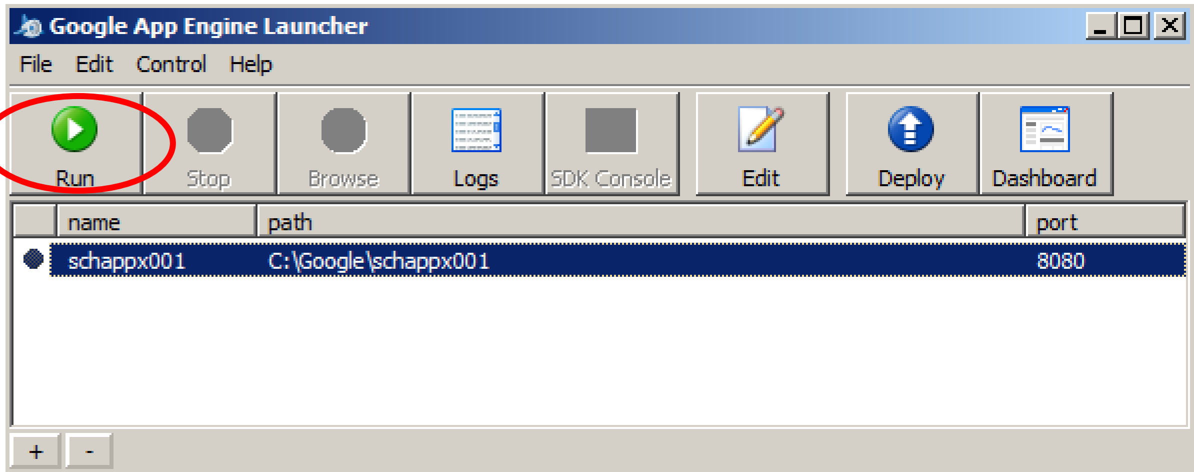
Application 1 – Testing the Application

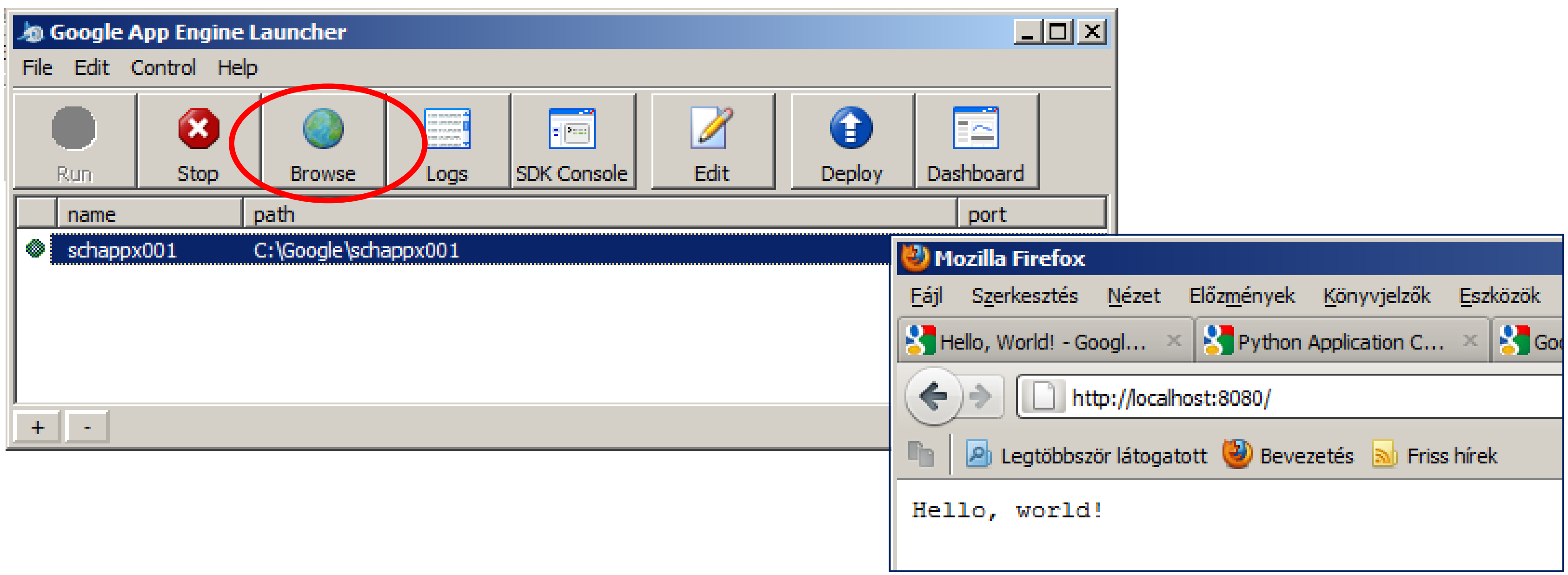
- Test application with the web server included in the App Engine SDK
  1. Launching the local App Engine web server:
     - dev_appserver.py C:\Google\schappx001/
     - The web server is listening for requests on port 8080
     - URL in the web browser: http://localhost:8080/
     - Changing the port of the web server: dev_appserver.py --port=8086 C:\Google\schappx001/
  2. Using the Google App Engine Launcher
     - Set up the application by selecting the File menu, Add Existing Application...
     - Select the C:\Google\schappx001 directory
     - Select the application in the app list, click the Run button
     - Click the Browse button to view it. Clicking Browse loads (or reloads) http://localhost:8080/ in your default web browser
- The web server can be running during application development. The web server knows to watch for changes in the source files and reload them if necessary

Application 1 – Testing the Application (Cont.)



Application 1 – Testing the Application (Cont.)



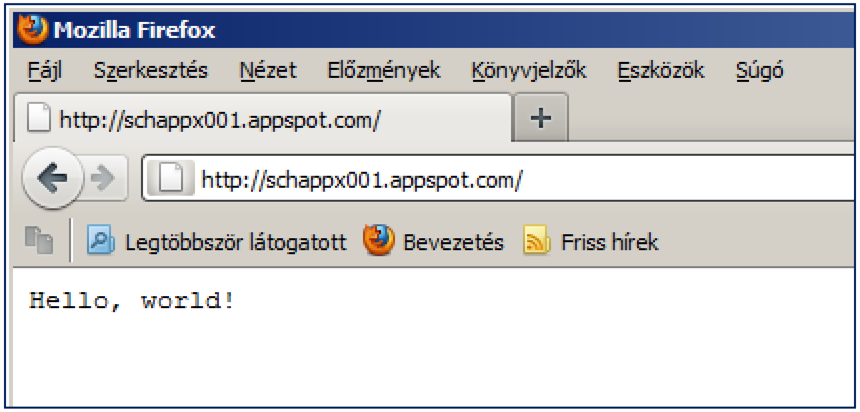


Application 1 – Registering and Uploading the Application (Cont.)

- Register application ID for the application in the Google App Engine
  - Create and manage App Engine web applications from the App Engine Administration Console, at the following URL:
    - https://appengine.google.com/
    - Google App Engine Launcher users can reach this URL by clicking the Dashboard button
  - Sign in to App Engine using the Google account (Google account can be created with an email address and password)
  - Create a new application
    - Click the "Create an Application" button
    - Follow the instructions to register an application ID (unique name)
    - Elect to use the free appspot.com domain name
    - The full URL for the application will be http://application-id.appspot.com (http://schappx001.appspot.com)
    - Users can purchase a top-level domain name for the app, or use one that they have already registered
    - Edit the Application line in the app.yaml file if necessary

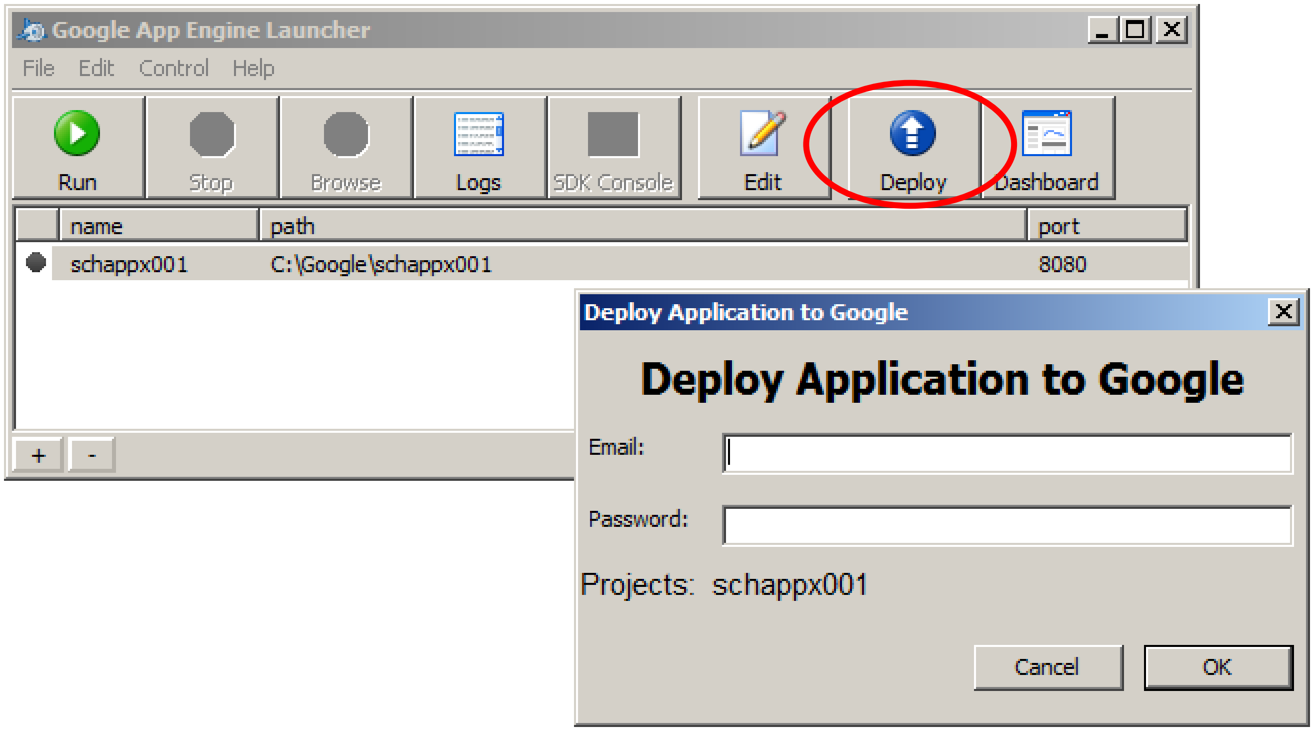
Application 1 – Registering and Uploading the Application (Cont.)

- Upload the application to the App Engine:
  1. From command line run:
     - appcfg.py update c:\Google\schappx001/
     - Enter the Google username and password at the prompts
  2. From Google App Engine Launcher
     - Click Deploy button
     - Enter the Google username and password in GUI
- Run the application on App Engine:
  - o http://application-id.appspot.com (http://schappx001.appspot.com)



Application 1 – Registering and Uploading the Application (Cont.)

Upload application from Google App Engine Launcher



Managing Application with the Administration Console

- Applications in App Engine can be managed by the Administration Console
- The Google App Engine Administration Console gives users complete access to the public version of their application
- From a web browser:

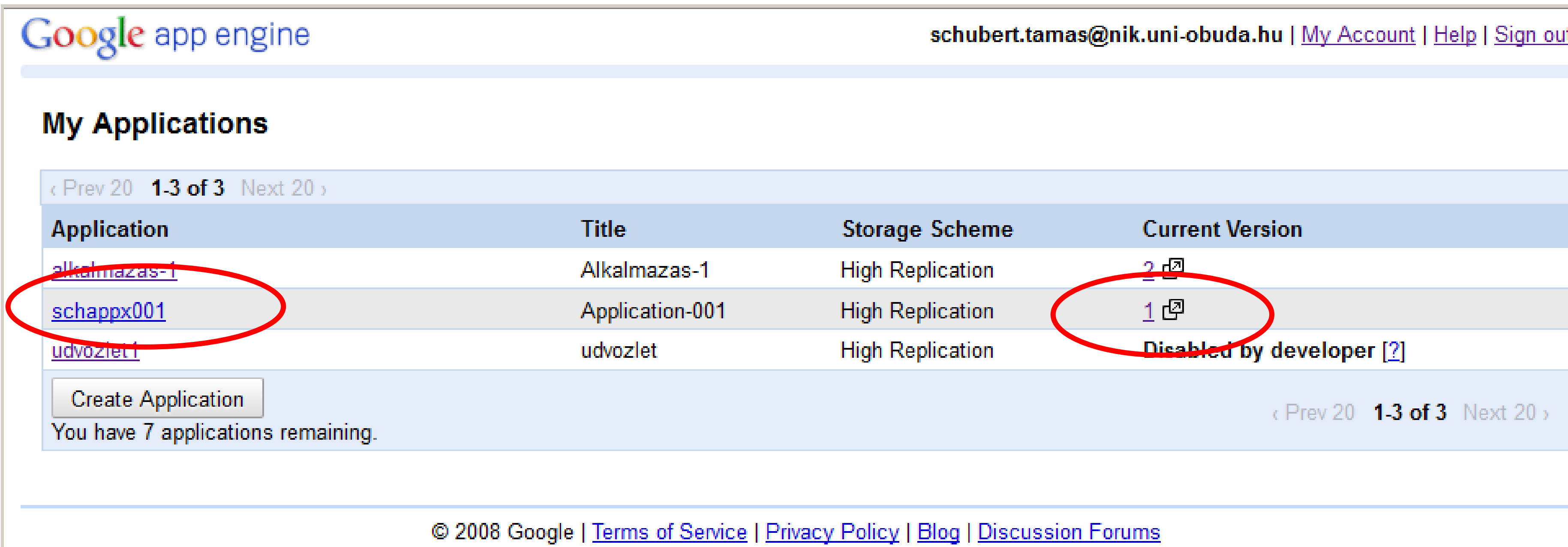
  https://appengine.google.com/
- Sign in with the Google account
- Administration Console can be used to:
  - Create a new application with a free appspot.com sub-domain, or a top-level domain name
  - Invite other people to be developers for an application, so they can access the Console and upload new versions of the code
  - View request and error logs, and analyze traffic
  - Browse application's datastore and manage indexes
  - Administer datastore

Managing Application with the Administration Console (Cont.)

- View application's instances
- Manage task queues, allowing for pausing, purging, and deleting queues
- Manage individual tasks in a task queue, allowing for viewing, deleting, or running individual tasks immediately
- Test new versions of applications, and switch the version that users can see

Managing Application with the Administration Console (Cont.)

My Applications



Managing Application with the Administration Console (Cont.)

Versions

app.yaml

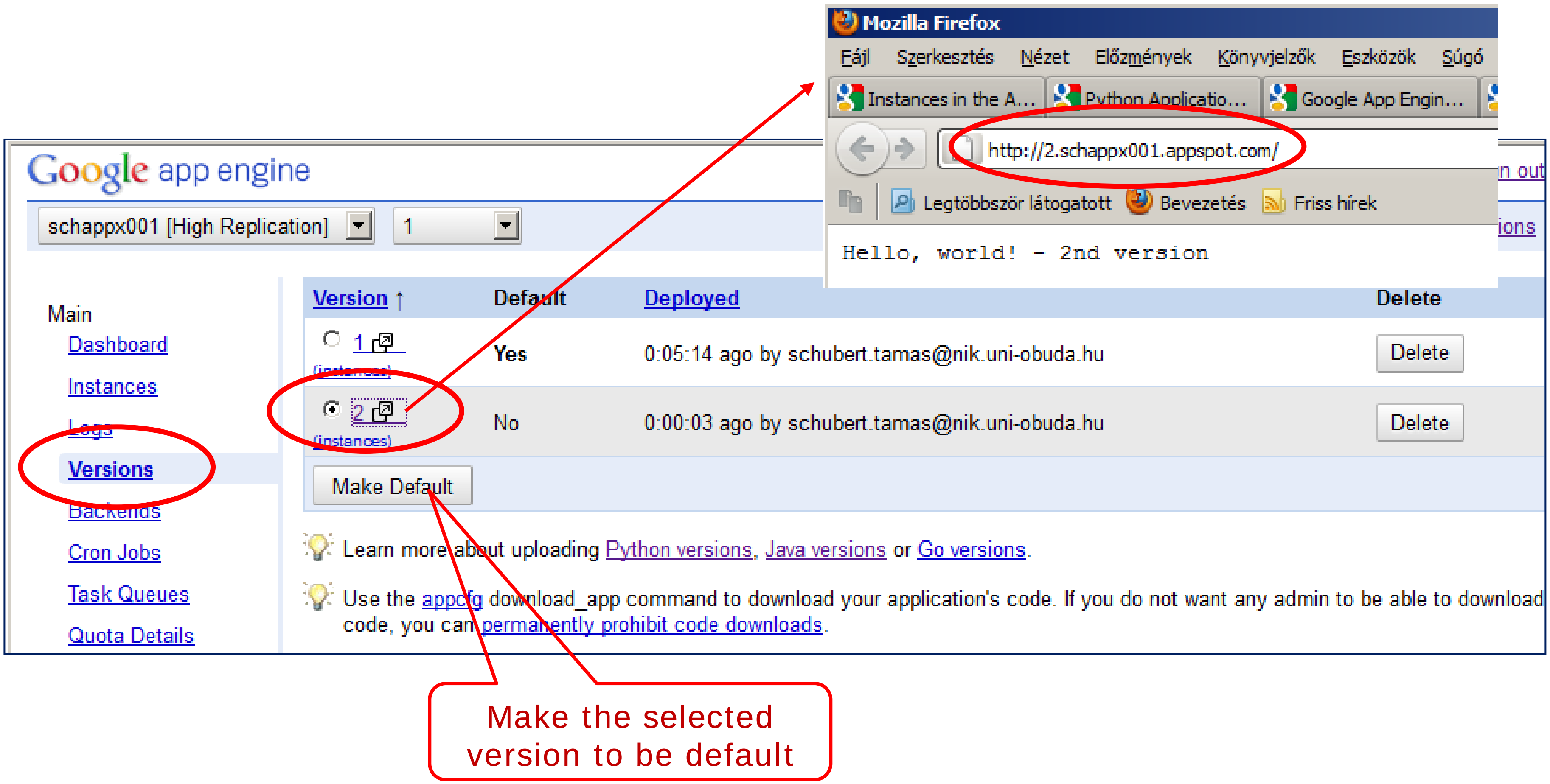
```
application: schappx001
version: 2
runtime: python
api_version: 1

handlers:
- url: /.*
  script: schappx001.py
```

schappx001.py

```
print 'Content-Type: text/plain'
print ''
print 'Hello, world! - 2nd version'
```

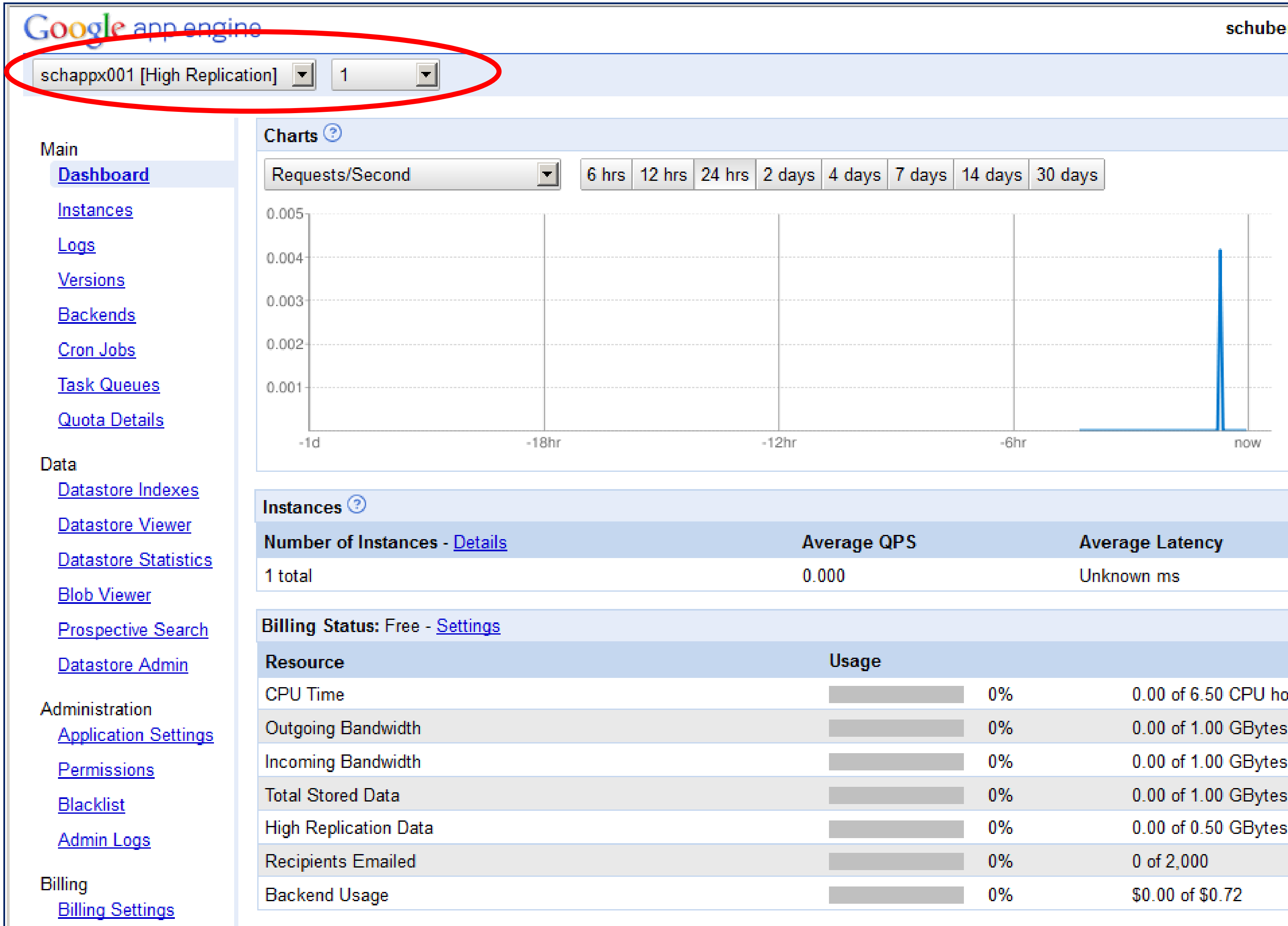
Managing Application with the Administration Console (Cont.)

Versions



Managing Application with the Administration Console (Cont.)

Dashboard



Using the webapp Framework

- The CGI standard is simple, but it would be cumbersome to write all of the code that uses it by hand
- Web application frameworks handle these details for developers, so they can focus their development efforts
- Google App Engine supports any framework written in pure Python that speaks CGI (and any WSGI-compliant framework using a CGI adaptor), including Django, CherryPy, Pylons, web.py, and web2py
- Developers can bundle their own framework copying its code into the application directory
- App Engine includes a simple web application framework of its own, called webapp
- The webapp framework is already installed in the App Engine environment and in the SDK
- This version (Hello, webapp!) of application uses the webapp framework

Using the webapp Framework – schappx001-6.py

```
from google.appengine.ext import webapp
from google.appengine.ext.webapp.util import run_wsgi_app

class MainPage(webapp.RequestHandler):
    def get(self):
        self.response.headers['Content-Type'] = 'text/plain'
        self.response.out.write('Hello, webapp World!')

application = webapp.WSGIApplication(
                                     [('/', MainPage)],
                                     debug=True)

def main():
    run_wsgi_app(application)

if __name__ == "__main__":
    main()
```

Using the webapp Framework – schappx001-6.py (Cont.)

- A webapp application has three parts:
  - One or more RequestHandler classes that process requests and build responses
  - A WSGIApplication instance that routes incoming requests to handlers based on the URL
  - A main routine that runs the WSGIApplication using a CGI adaptor

Using the webapp Framework – schappx001-6.py (Cont.)

- The webapp module is in the google.appengine.ext package
- This module is provided in the SDK, as well as in the production runtime environment
- This code defines one request handler, MainPage, mapped to the root URL (/)
- When webapp receives an HTTP GET request to the URL /, it instantiates the MainPage class and calls the instance's get method
- Inside the method, information about the request is available using self.request
- Typically, the method sets properties on self.response to prepare the response, then exits
- webapp sends a response based on the final state of the MainPage instance

Using the webapp Framework – schappx001-6.py (Cont.)

- The application itself is represented by a webapp.WSGIApplication instance
- The parameter debug=true passed to its constructor tells webapp to print stack traces to the browser output if a handler encounters an error or raises an uncaught exception
- The function run_wsgi_app() takes a WSGIApplication instance (or another WSGI-compatible application object) and runs it in App Engine's CGI environment
- run_wsgi_app() is similar to the WSGI-to-CGI adaptor provided by the wsgiref module in the Python standard library, but includes a few additional features
- For example, it can automatically detect whether the application is running in the development server or on App Engine, and display errors in the browser if it is running on the development server

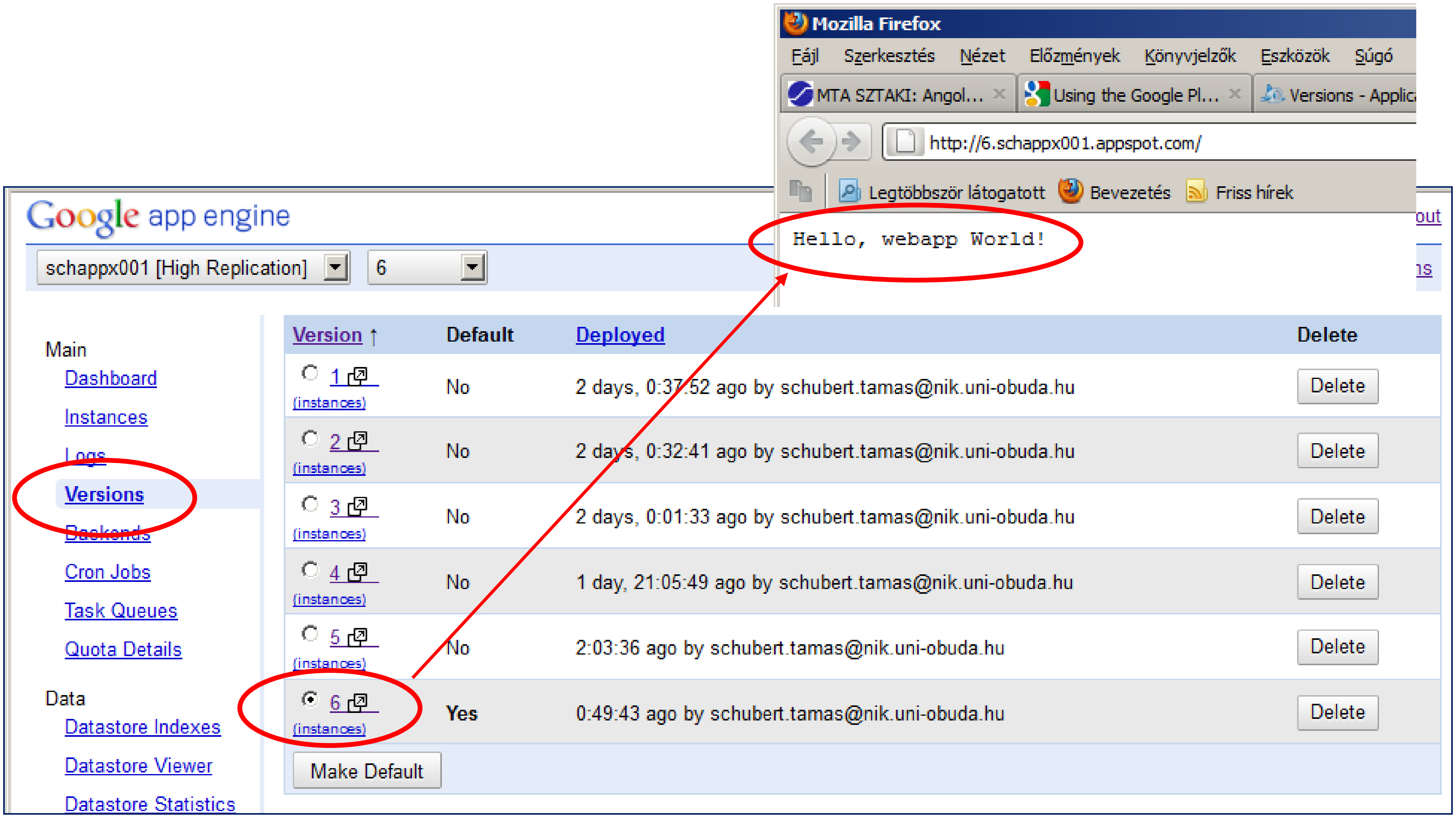
Using the webapp Framework – app.yaml

6th version of the same application

```
application: schappx001
version: 6
runtime: python
api_version: 1

handlers:
- url: /.*
  script: schappx001-6.py
```

Using the webapp Framework – Running application

6th version of the same application



3.2. Application 2

Application 2 – Using the Users Service

- Google App Engine provides several services based on Google infrastructure, accessible by applications using libraries included with the SDK
- The Users service lets application integrate with Google user accounts
- With the Users service, users can use the Google accounts they already have to sign in
- This application uses Users service to personalize this application's greeting

Application 2 – schappx001-3.py

```
from google.appengine.api import users
from google.appengine.ext import webapp
from google.appengine.ext.webapp.util import run_wsgi_app

class MainPage(webapp.RequestHandler):
    def get(self):
        user = users.get_current_user()
        if user:
            self.response.headers['Content-Type'] = 'text/plain'
            self.response.out.write('Hello, ' + user.nickname())
        else:
            self.redirect(users.create_login_url(self.request.uri))
application = webapp.WSGIApplication(
                                     [('/', MainPage)],
                                     debug=True)
def main():
    run_wsgi_app(application)

if __name__ == "__main__":
    main()
```

Application 2 – The Users API

- If the user is already signed in to the application, get_current_user() returns the User object for the user. Otherwise, it returns None

```
user = users.get_current_user()      -- User object
```

- If the user has signed in, display a personalized message, using the nickname associated with the user's account

```
if user:
   self.response.headers['Content-Type'] = 'text/plain'
   self.response.out.write('Hello, ' + user.nickname())
```

- If the user has not signed in, tell webapp to redirect the user's browser to the Google account sign-in screen
- The redirect includes the URL to this page (self.request.uri) so the Google account sign-in mechanism will send the user back here after the user has signed in or registered for a new account

```
else:
   self.redirect(users.create_login_url(self.request.uri))
```

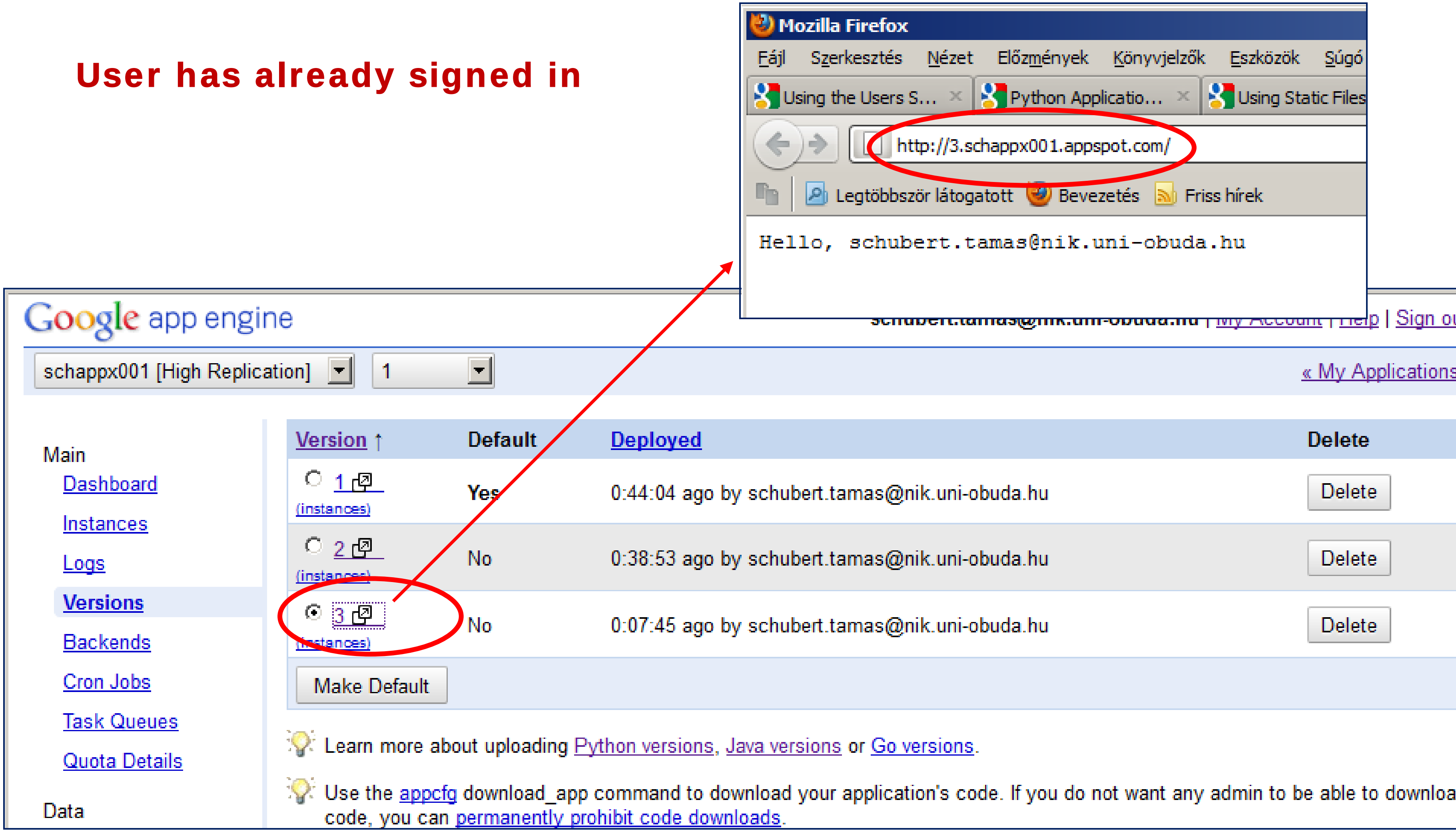
Application 2 – app.yaml

3rd version of the same application

```
application: schappx001
version: 3
runtime: python
api_version: 1

handlers:
- url: /.*
  script: schappx001-3.py
```

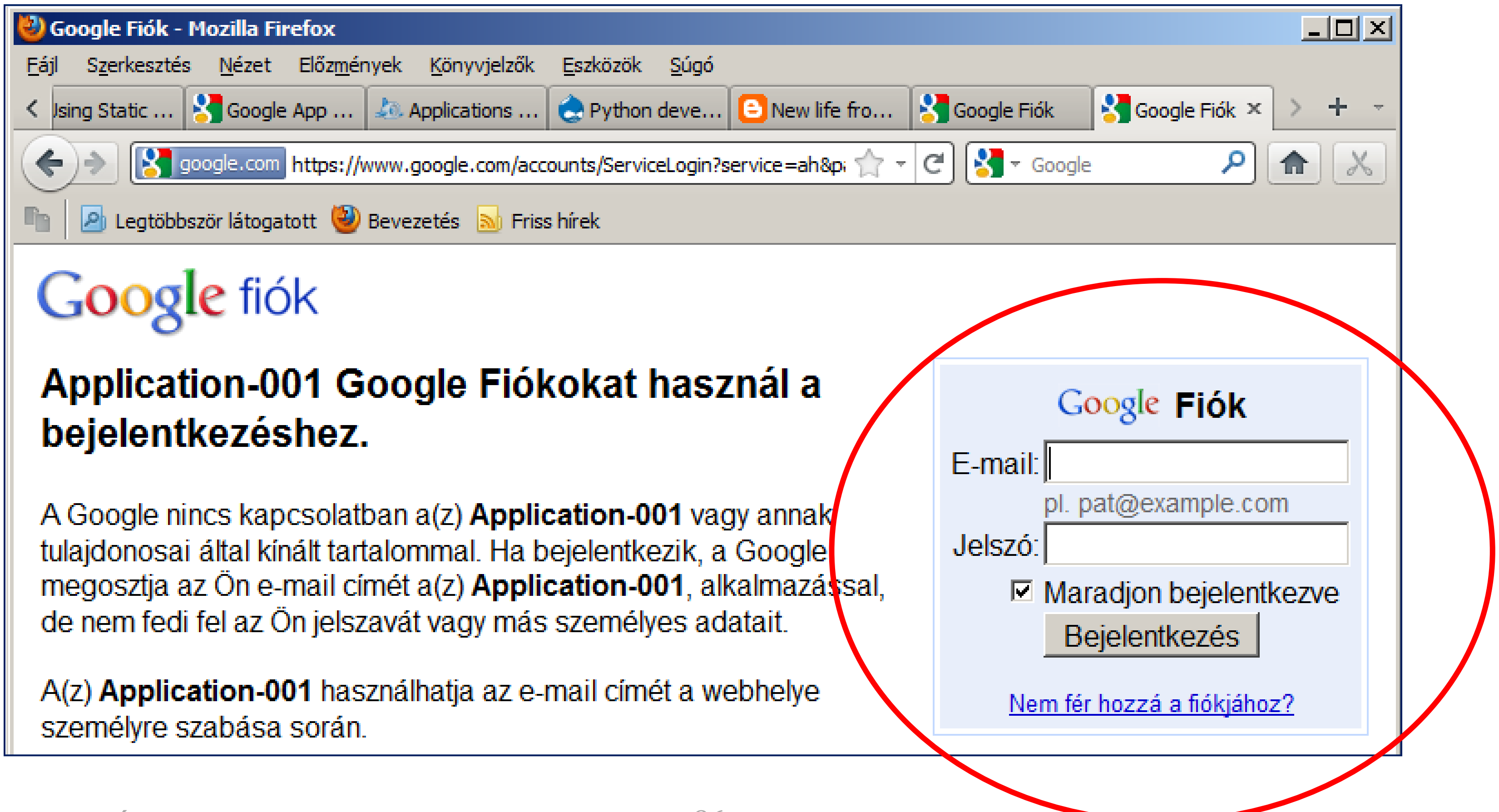
Application 2 – Upload and run the application

User has already signed in



Application 2 – Upload and run the application (Cont.)

User hasn't signed in yet



3.3. Application 3

Application 3 – Using the Datastore

- App Engine's infrastructure takes care of
  - data distribution
  - data replication
  - load balancing
- It has
  - simple API
  - powerful query engine
  - transactions management
- The default datastore for an application is now the High Replication datastore
- This datastore uses the Paxos algorithm to replicate data across datacenters

Application 3 – Using the Datastore (Cont.)

- The datastore writes data in objects known as entities, and each entity has a key that identifies the entity
- Entities can belong to the same entity group
- This allows users to perform a single transaction with multiple entities
- Entity groups have a parent key that identifies the entire entity group
- In the HRD, entity groups are a unit of consistency
- Queries over multiple entity groups may return stale, eventually consistent results
- Queries over a single entity group return up-to-date, strongly consistent, results
- Queries over a single entity group are called ancestor queries
- Ancestor queries use the parent key (instead of a specific entity's key)
- This Guestbook application stores greetings in the datastore

Application 3 – schappx001-4.py – class Greeting

```
import cgi
import datetime
import urllib
import wsgiref.handlers

from google.appengine.ext import db
from google.appengine.api import users
from google.appengine.ext import webapp
from google.appengine.ext.webapp.util import run_wsgi_app

class Greeting(db.Model):
  """Models an individual Guestbook entry with an author, content, and date."""
  author = db.UserProperty()
  content = db.StringProperty(multiline=True)
  date = db.DateTimeProperty(auto_now_add=True)

def guestbook_key(guestbook_name=None):
  """Constructs a datastore key for a Guestbook entity with
     guestbook_name."""
  return db.Key.from_path('Guestbook', guestbook_name or
     'default_guestbook')
```


Application 3 – schappx001-4.py – class MainPage

```
class MainPage(webapp.RequestHandler):
  def get(self):
    self.response.out.write('<html><body>')
    guestbook_name=self.request.get('guestbook_name')

    # Ancestor Queries, as shown here, are strongly consistent with the High
    # Replication datastore. Queries that span entity groups are eventually
    # consistent. If we omitted the ancestor from this query there would be a
    # slight chance that Greeting that had just been written would not show up
    # in a query.
    greetings = db.GqlQuery("SELECT * "
                            "FROM Greeting "
                            "WHERE ANCESTOR IS :1 "
                            "ORDER BY date DESC LIMIT 10",
                            guestbook_key(guestbook_name))
```

Application 3 – schappx001-4.py – class MainPage (Cont.)

```
for greeting in greetings:
     if greeting.author:
       self.response.out.write(
          '<b>%s</b> wrote:' % greeting.author.nickname())
     else:
       self.response.out.write('An anonymous person wrote:')
     self.response.out.write('<blockquote>%s</blockquote>' %
                          cgi.escape(greeting.content))
self.response.out.write("""
         <form action="/sign?%s" method="post">
           <div><textarea name="content" rows="3"
    cols="60"></textarea></div>
           <div><input type="submit" value="Sign Guestbook"></div>
         </form>
         <hr>
         <form>Guestbook name: <input value="%s"
    name="guestbook_name">
         <input type="submit" value="switch"></form>
       </body>
     </html>""" % (urllib.urlencode({'guestbook_name': guestbook_name}),
                         cgi.escape(guestbook_name)))
```

Application 3 – schappx001-4.py – class Guestbook

```
class Guestbook(webapp.RequestHandler):
  def post(self):
    # We set the same parent key on the 'Greeting' to ensure each greeting is
     in
    # the same entity group. Queries across the single entity group will be
    # consistent. However, the write rate to a single entity group should
    # be limited to ~1/second.
    guestbook_name = self.request.get('guestbook_name')
    greeting = Greeting(parent=guestbook_key(guestbook_name))

    if users.get_current_user():
      greeting.author = users.get_current_user()

    greeting.content = self.request.get('content')
    greeting.put()
    self.redirect('/?' + urllib.urlencode({'guestbook_name': guestbook_name}))
```

Application 3 – schappx001-4.py (Cont.)

```
application = webapp.WSGIApplication([
  ('/', MainPage),
  ('/sign', Guestbook)
], debug=True)


def main():
  run_wsgi_app(application)


if __name__ == '__main__':
  main()
```

Application 3 – Storing the Submitted Greetings

- App Engine includes a data modeling API for Python
- The Guestbook application stores greetings posted by users
- Each greeting includes:
    - author's name
    - message content
    - date and time
- To use the data modeling API, import the google.appengine.ext.db module:

```
from google.appengine.ext import db
```

- The following defines a data model for a greeting:

```
class Greeting(db.Model):
    author = db.UserProperty()
    content = db.StringProperty(multiline=True)
    date = db.DateTimeProperty(auto_now_add=True)
```

Application 3 – Storing the Submitted Greetings (Cont.)

- This defines a Greeting model with three properties:
  - author – User object
  - content – string
  - date – datetime.datetime
- The multiline=True parameter in the db.StringProperty constructor says that values for this property can contain newline characters
- The auto_now_add=True parameter in the db.DateTimeProperty constructor configures the model to automatically give new objects a date of the time the object is created, if the application doesn't otherwise provide a value

Application 3 – Storing the Submitted Greetings (Cont.)

- The application can use the model to create new Greeting objects and put them into the datastore
- The Guestbook handler creates new greetings and saves them to the datastore:

```
class Guestbook(webapp.RequestHandler):
   def post(self):
     guestbook_name = self.request.get('guestbook_name')
     greeting = Greeting(parent=guestbook_key(guestbook_name))

     if users.get_current_user():
       greeting.author = users.get_current_user()

     greeting.content = self.request.get('content')
     greeting.put()
     self.redirect('/?' + urllib.urlencode({'guestbook_name':
    guestbook_name}))
```

Application 3 – Storing the Submitted Greetings (Cont.)

- The parent has an entity kind „Guestbook”
- There is no need to create the „Guestbook” entity before setting it to be the parent of another entity
- In this example, the parent is used as a placeholder for transaction and consistency purposes
- Objects that share a common ancestor belong to the same entity group
- It does not set the date property, so date is automatically set to „now”
- greeting.put() saves the new object to the datastore
- If this object had been acquired from a query, put() would have updated the existing object
- Since this object was created with the model constructor, put() adds the new object to the datastore

Application 3 – Storing the Submitted Greetings (Cont.)

- Querying in the High Replication datastore is strongly consistent within entity groups
- In this example, all Greetings are assigned to the same entity group
- This means a user will always see a Greeting immediately after it was written. However, the rate at which users can write to the same entity group is limited to 1 write to the entity group per second

Application 3 – Retrieving the Stored Greetings With GQL

- App Engine datastore is not a traditional relational database, queries are not specified using SQL
- Instead, users can prepare queries using a SQL-like query language called GQL
- GQL provides access to the App Engine datastore query engine's features using a familiar syntax
- The following MainPage handler queries the datastore for greetings:

```
greetings = db.GqlQuery(" SELECT * "
                        "FROM Greeting "
                        "WHERE ANCESTOR IS :1 "
                        "ORDER BY date DESC LIMIT 10",
                        guestbook_key(guestbook_name))
```

- The query returns full data objects, it does not make sense to select specific properties from the model

Application 3 – Retrieving the Stored Greetings With GQL (Cont.)

- A GQL query can have a WHERE clause that filters the result set by one or more conditions based on property values
- Unlike SQL, GQL queries may not contain value constants: Instead, GQL uses parameter binding for all values in queries
- For example, to get only the greetings posted by the current user:

```
if users.get_current_user():
        greetings = Greeting.gql(
            "WHERE ANCESTOR IS :1 AND author = :2 ORDER BY date DESC",
            guestbook_key(guestbook_name), users.get_current_user())
```

- Developers can also use named parameters instead of positional parameters:

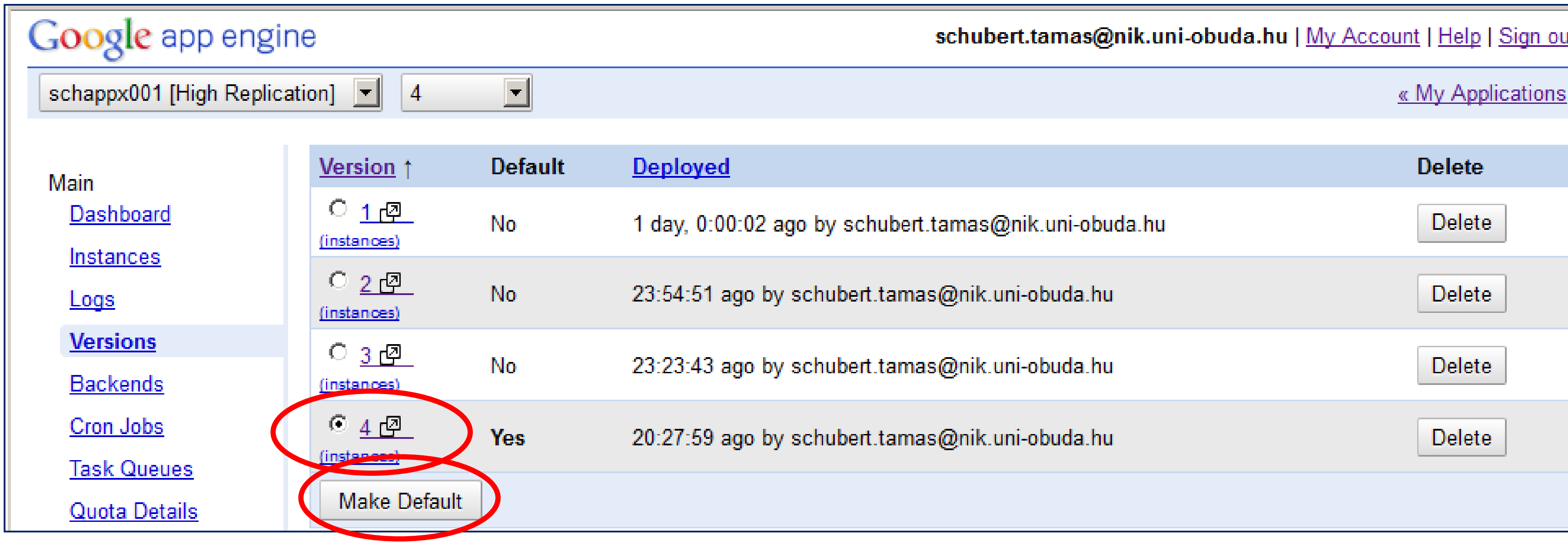
```
greetings = Greeting.gql("WHERE ANCESTOR = :ancestor AND
                          author = :author
                          ORDER BY date DESC",
                          ancestor=guestbook_key(guestbook_name),
                          author=users.get_current_user())
```

Application 3 – app.yaml

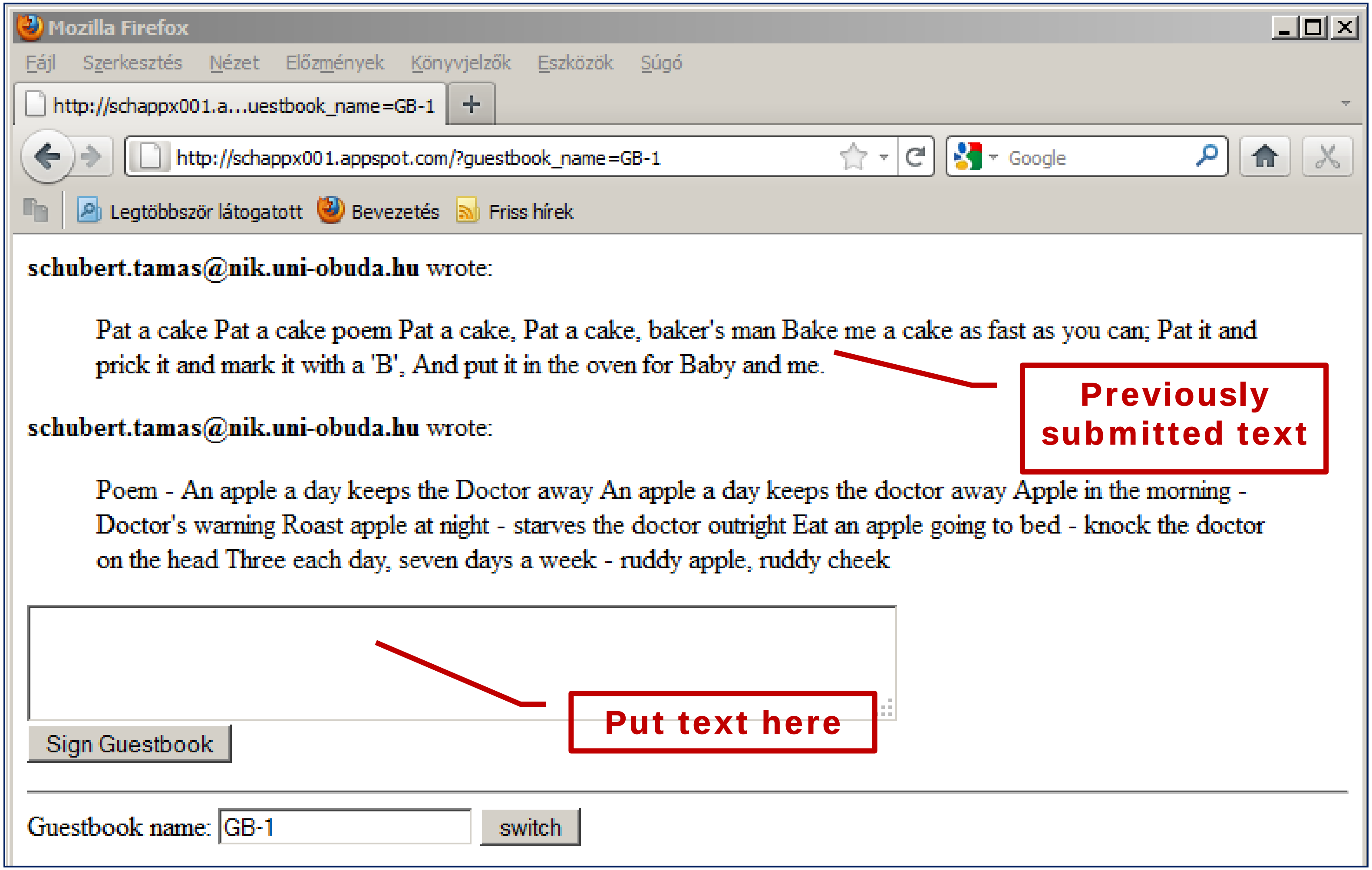
4th version of the same application

```
application: schappx001
version: 4
runtime: python
api_version: 1

handlers:
- url: /.*
  script: schappx001-4.py
```

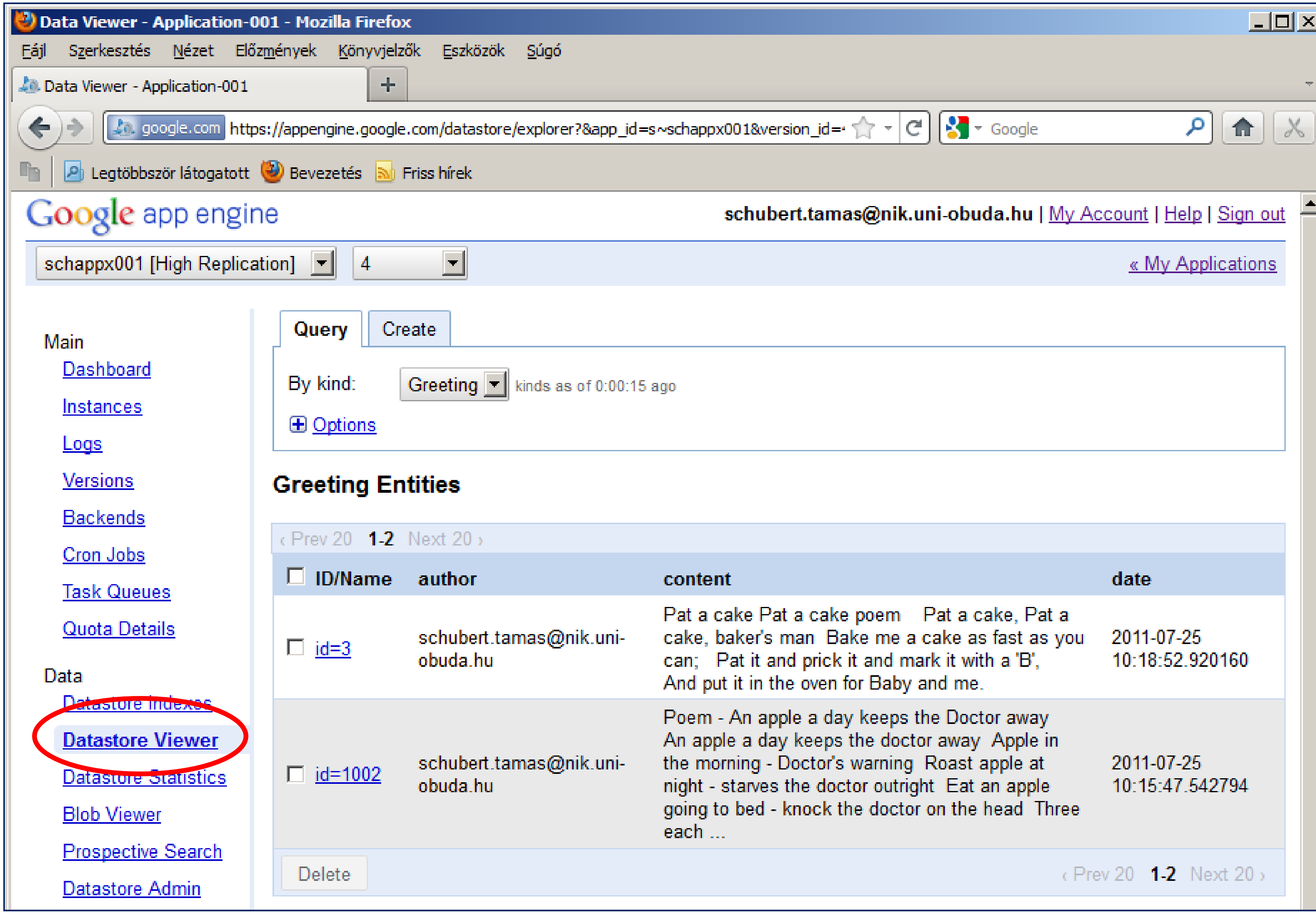
Application 3 – Run application



Application 3 – Run application (Cont.)



Application 3 – Viewing Datastore



[1]: http://www.cloud.com, Cloud Computing Outlook, 2011

[2]: http://cloudtaxonomy.opencrowd.com/taxonomy, 2005

[3]: Sarna, D.E.Y.: Implementing and Developing Cloud Computing Applications, Auerbach Publications, 2011

[4]: Cârstoiu, B.: Cloud SaaS infrastructure, Control and Computers Faculty, University Politehnica of Bucharest, Romania

[5]: Dean J., Ghemawat S.: MapReduce: Simplified Data Processing on Large Clusters, Google, Inc., 2004

[6]: Hurwitz J,. Bloor R., Kaufman M., Halper F.: Cloud Computing For Dummies, Wiley Publishing, Inc., 2010

[7]: Velte A. T., Velte, T. J., Ph.D., Elsenpeter R., Cloud Computing: A Practical Approach, McGraw-Hill, 2010

[8]: http://docs.google.com/: Google Docs

[9]: Amazon Simple Storage Service, Quick Reference Card (Version 2006-03-01), http://awsdocs.s3.amazonaws.com/S3/latest/s3-qrc.pdf

[10]: Running Databases on AWS: http://aws.amazon.com/running_databases/

[11]: http://www.cloudsecurityalliance.org/guidance/csaguide.v2.1.pdf: Security Guidance for Critical Areas of Focus in Cloud Computing V2.1, Cloud Security Alliance, 2009

[12]: http://hadoop.apache.org/ : Apache™ Hadoop™

[13]: http://code.google.com/intl/hu-HU/appengine/docs/whatisgoogleappengine.html: Google App Engine, 2011

Cloud Computing

Software as a Service – SaaS

Tamás Schubert

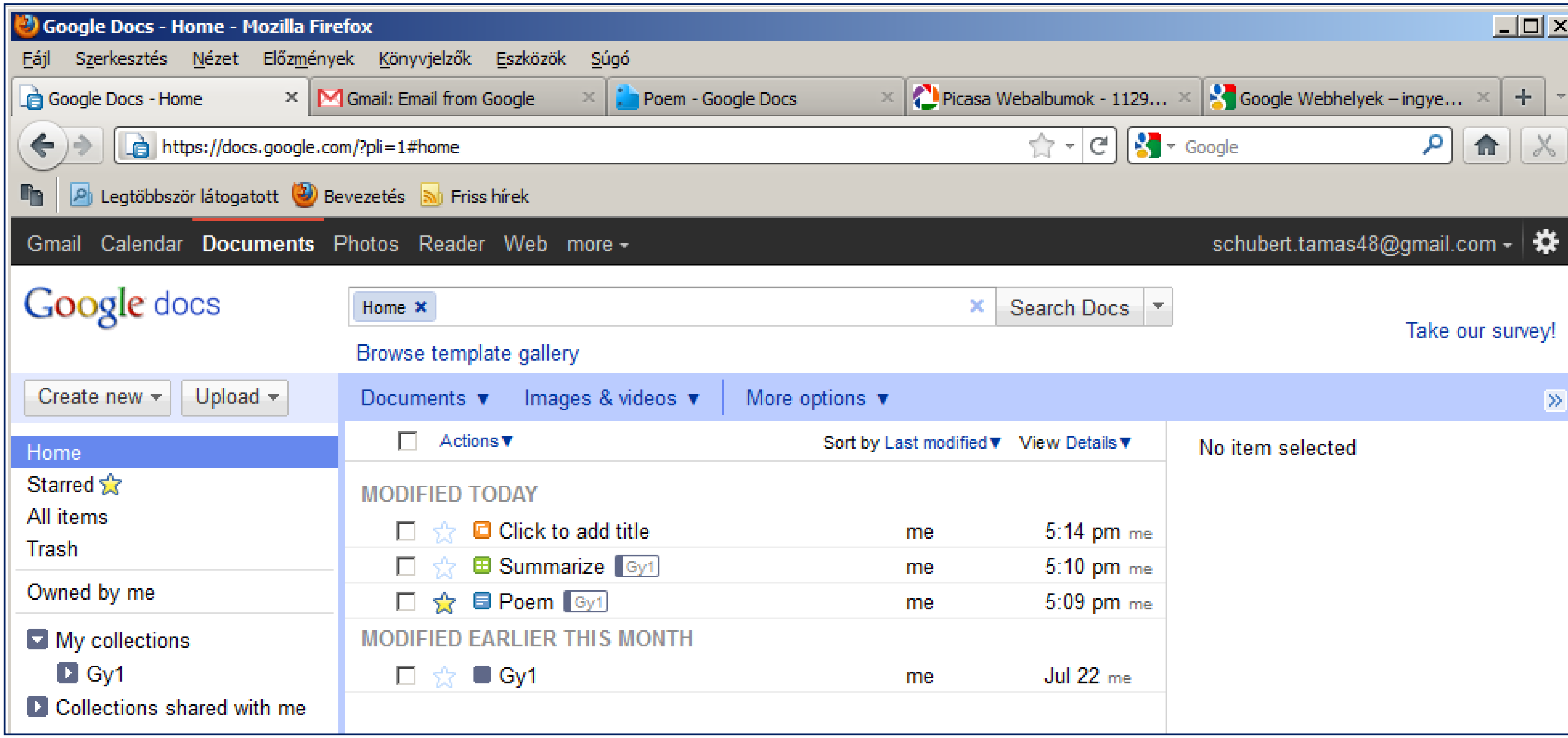
Content

1. Google Apps

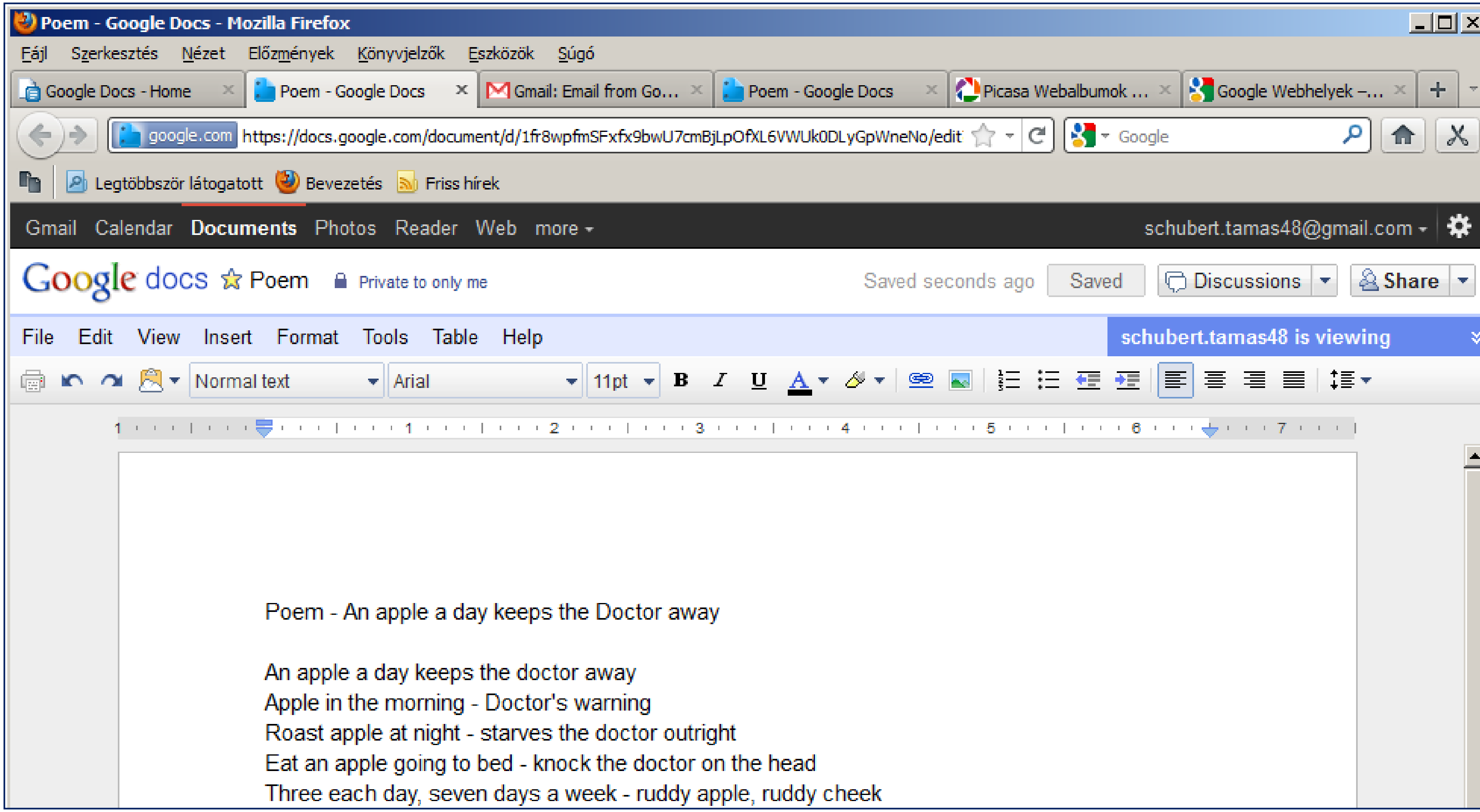
SaaS applications - Google Apps - Google Docs

- Free Web-based Google services:
  - Word processor
  - Spreadsheet
  - Slide show
  - Data storage service
- Document
  - create
  - edit
  - import/export
  - send e-mail
  - store on a Google server
- Real-time cooperation of users. Concurrent
  - open
  - edit
  - e-mail notification of users in the case of modification
- Support Microsoft .doc, .xls, .ppt forms
- Manage .pdf documents

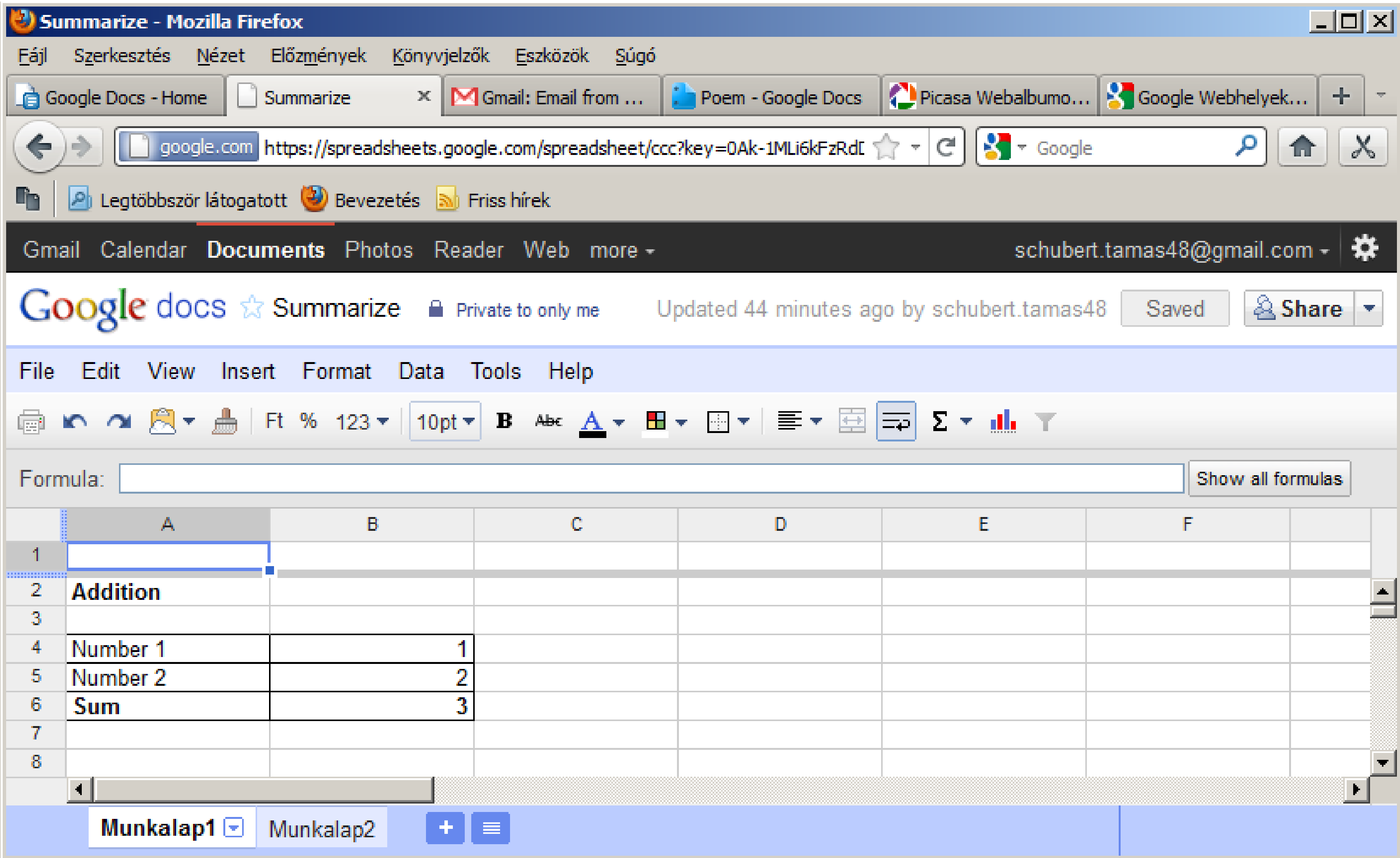
Google docs – Documents



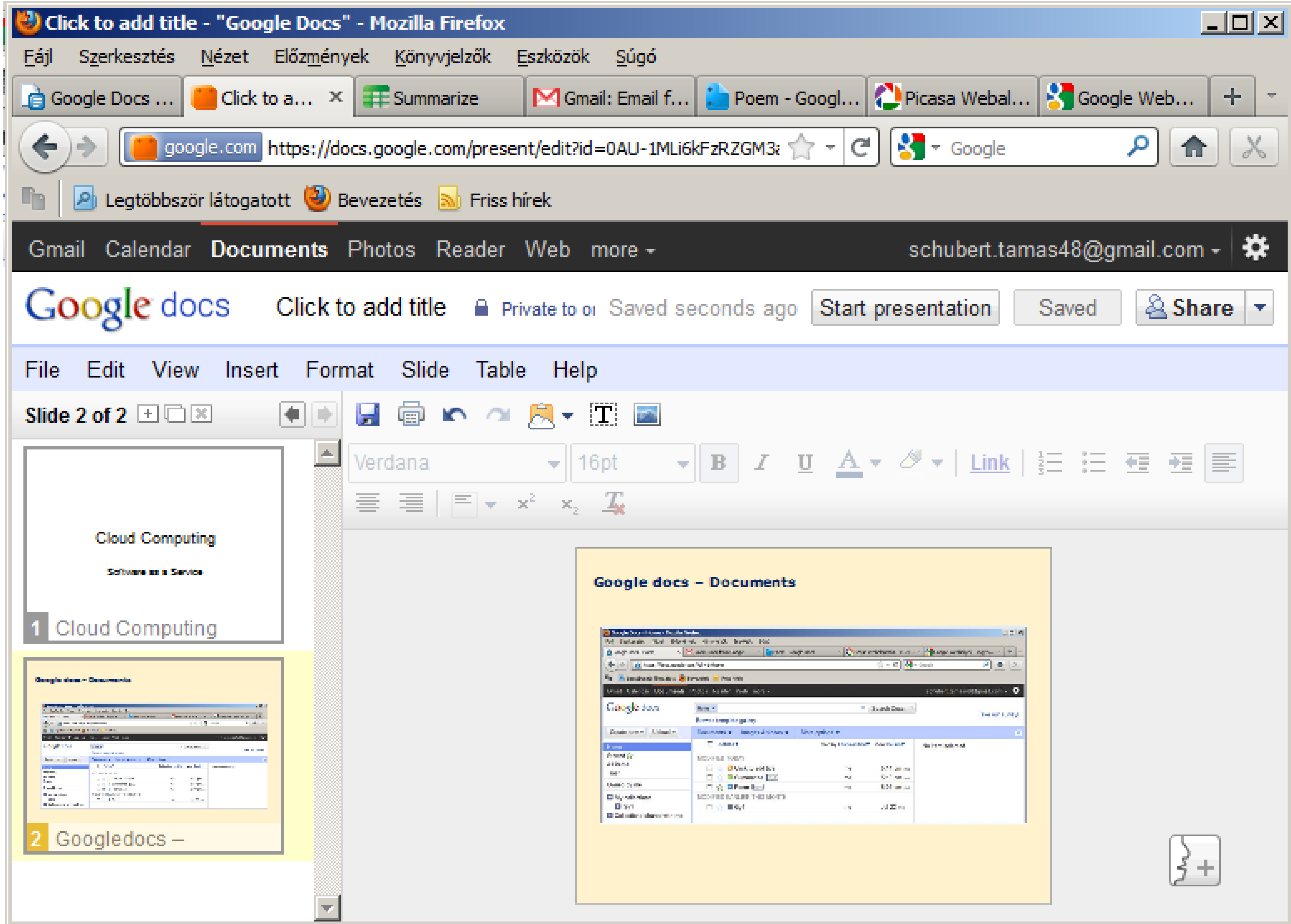
Google docs – Document



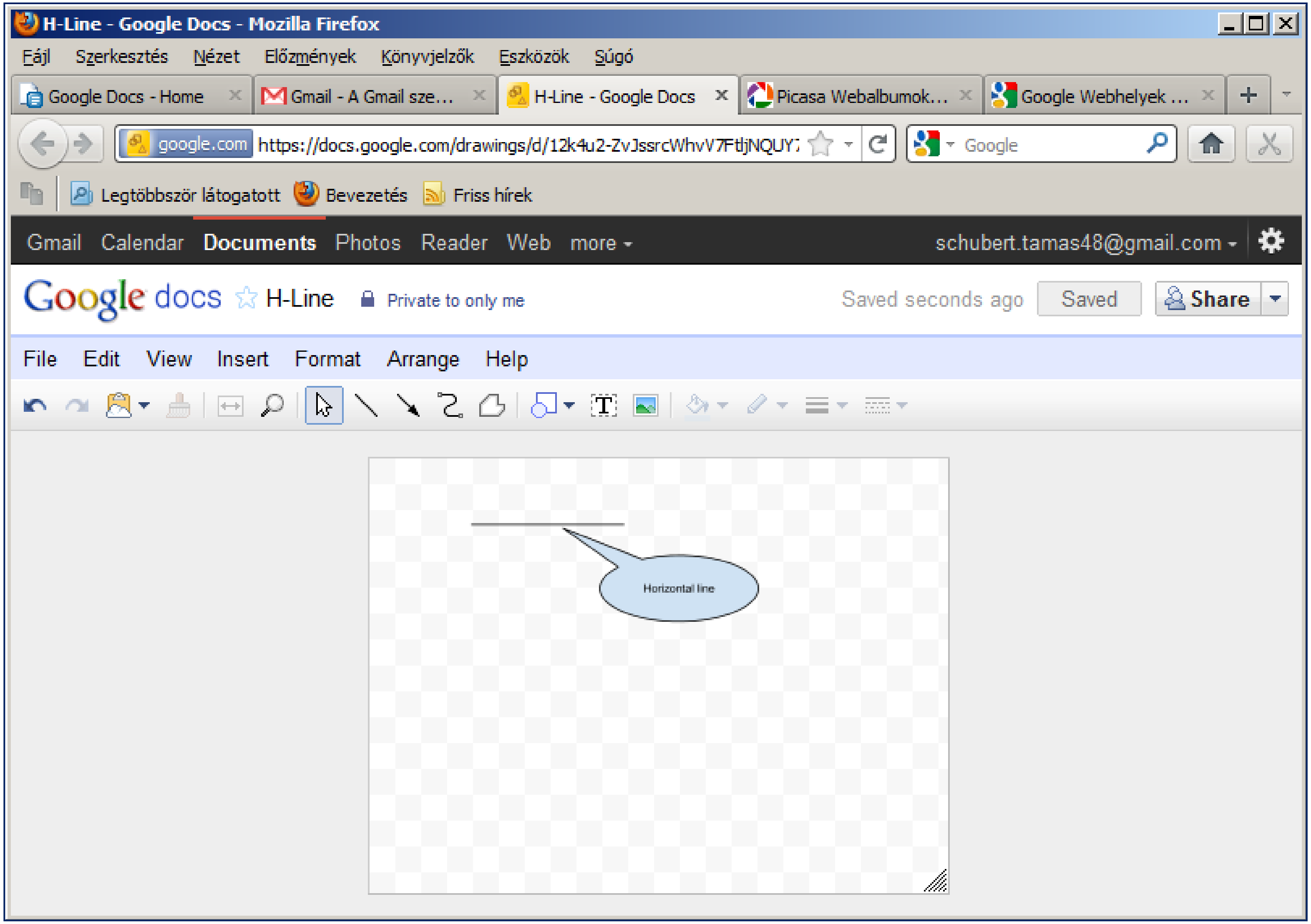
Google docs – Spreadsheet



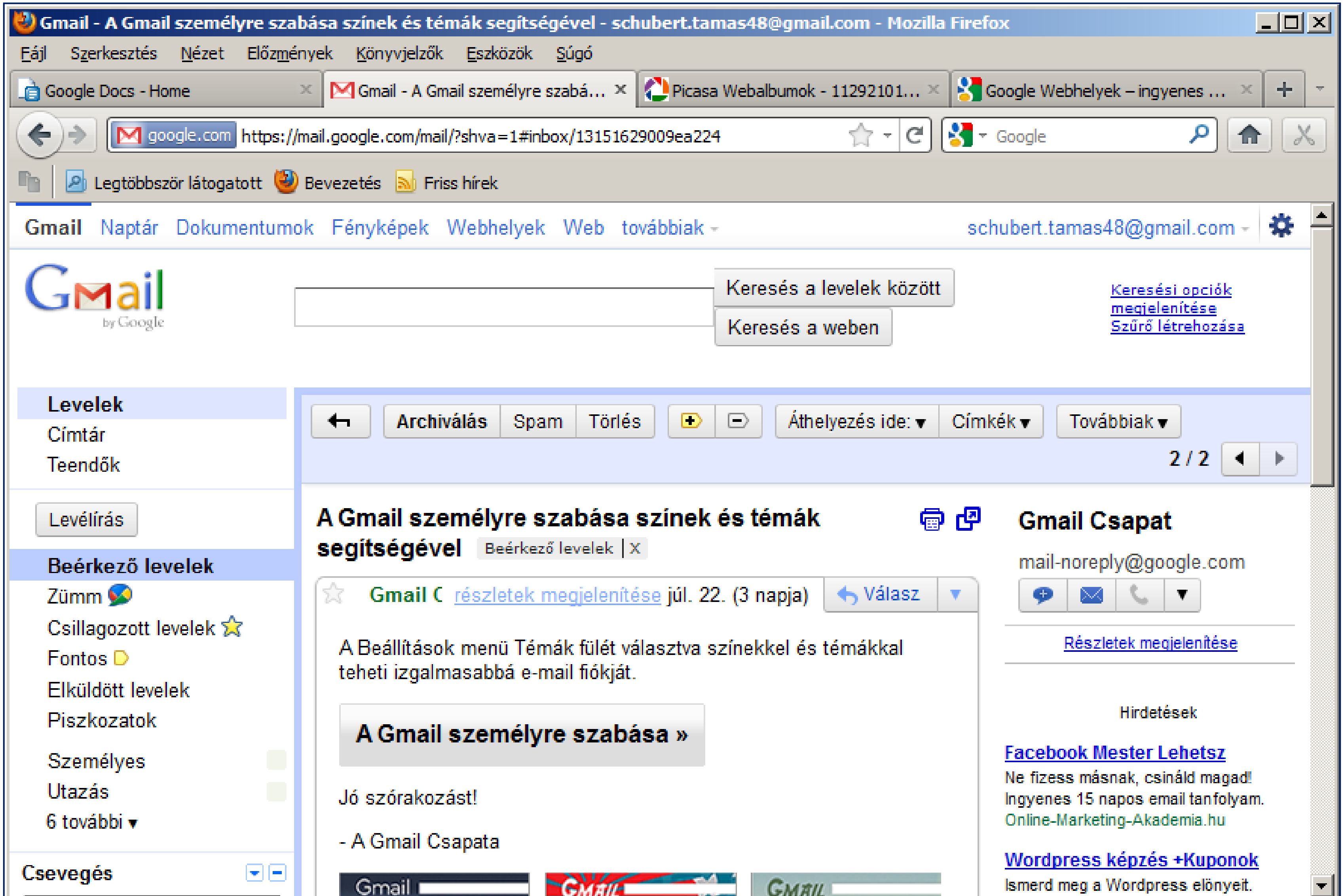
Google docs – Presentation



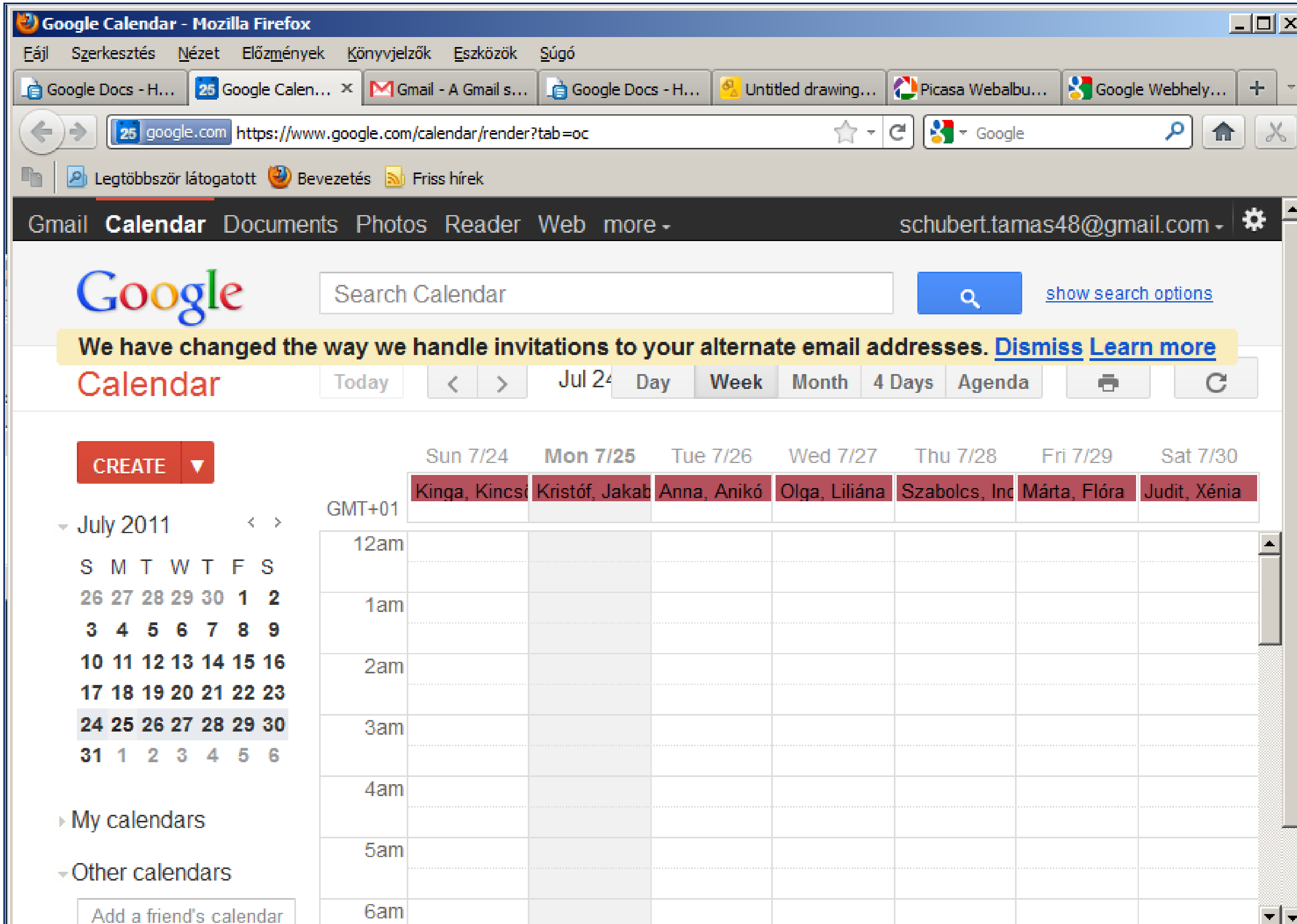
Google docs – Drawing



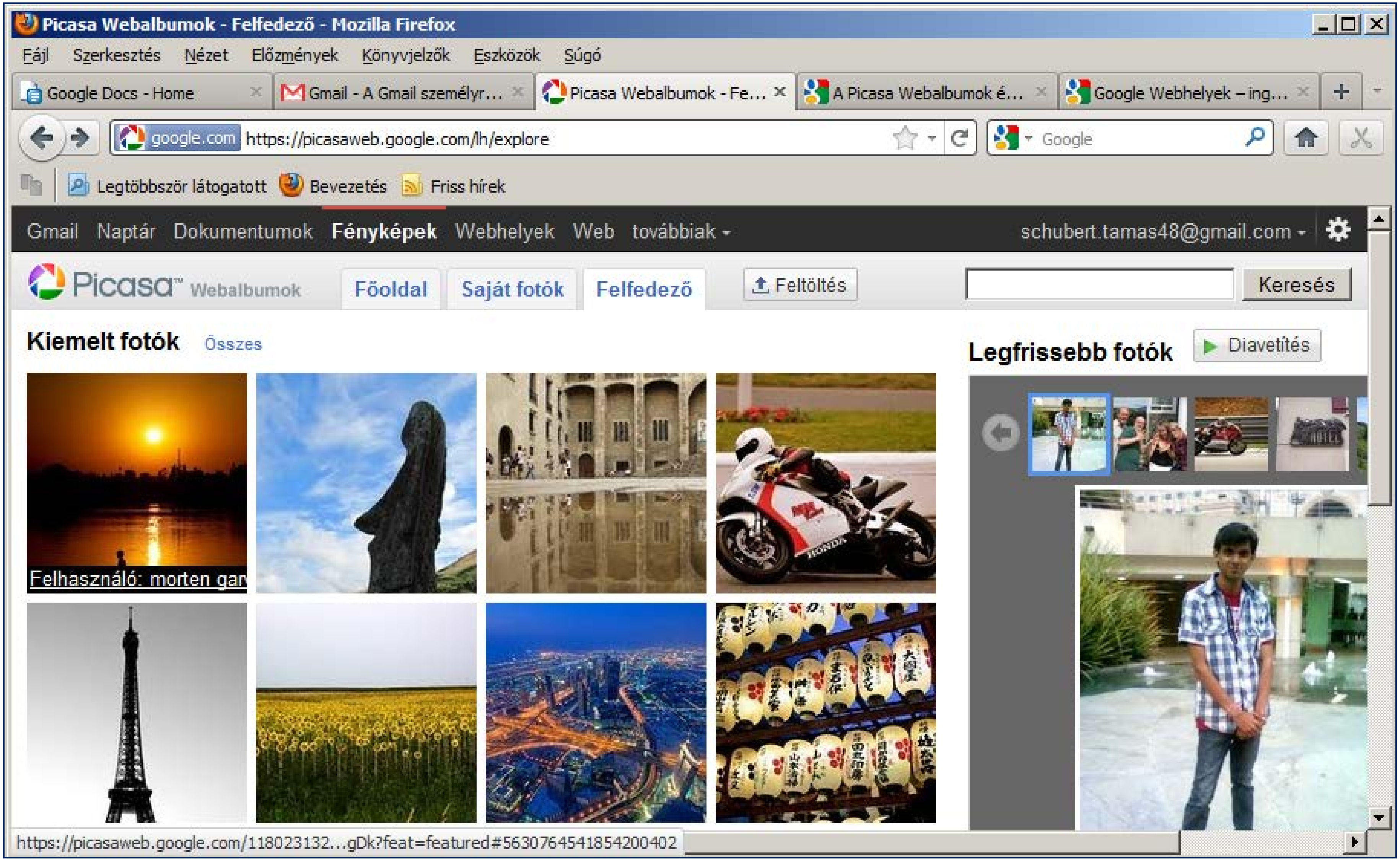
Google Apps – Gmail



Google Apps – Calendar



Google Apps – Picasa Web album



2. Salesforce.com – Customer Relationship Management (CRM) [7]

Salesforce.com – CRM

- Salesforce.com is a leader in cloud computing Customer Relationship Management (CRM) applications
- Its CRM consists of the Sales Cloud and the Service Cloud and can be broken down into five core applications:
  - Sales
    - The most popular cloud computing sales application
    - Salesforce.com says that CRM Sales is used by more than 1.1 million customers around the world
    - Its claim to fame is that it is comprehensive and easy to customize
    - Its value proposition is that it empowers companies to manage people and processes more effectively, so reps can spend more time selling and less time on administrative tasks

Salesforce.com – CRM (Cont.)

- Marketing
  - With Salesforce.com CRM Marketing, marketers can put the latest web technologies to work building pipeline while collaborating seamlessly with their sales organization
  - The application empowers customers to manage multichannel campaigns and provide up-to-date messaging to sales
  - And since the application is integrated with the Salesforce.com CRM Sales application, the handoff of leads is automated

Salesforce.com – CRM (Cont.)

- Service
  - The Service Cloud is the new platform for customer service
  - Companies can tap into the power of customer conversations no matter where they take place
  - Because it's on the Web, the Service Cloud allows companies to instantly connect to collaborate in real time, share sales information, and follow joint processes
  - Connecting with partners is made to be easy: companies instantly share leads, opportunities, accounts, contacts, and tasks with their partners

Salesforce.com – CRM (Cont.)

- Collaboration
  - Salesforce.com CRM can help an organization work more efficiently with customers, partners, and employees by allowing them to collaborate among themselves in the cloud
  - Some of the capabilities include:
    - ✓ Create and share content in real time using Google Apps and Salesforce.com
    - ✓ Track and deliver presentations using Content Library
    - ✓ Give the community a voice using Ideas and Facebook
    - ✓ Tap into the collective wisdom of the sales team with Genius

Salesforce.com – CRM (Cont.)

- Analytics
  - Force.com offers real-time reporting, calculations, and dashboards so a business is better able to optimize performance, decision making, and resource allocation
- Custom Applications
  - Custom applications can be quickly created by leveraging one data model, one sharing model, and one user interface

3. SaaS Cloud Architecture

Software as a Service (SaaS) [7]

- Applications are available and can be managed via the internet
- Applications can be accessed exclusively by an internet browser, local installation isn't necessary
- The data structure of the application (distributed model) and the program architecture permit, that the application be used by several people at the same time (multi-tenancy)
- Uniform applications can be easily migrated to the cloud. The SaaS application needs to be generalized enough so that lots of customers will be interested in the service
- Customization can be achieved (without code change) by parameterization
- The security of the communication can be achieved by using SSL
- Customers needn't buy software licenses (on demand licensing), customers only pay for the service (e.g. per-month, per-user fee)
- An SaaS application needs to include measuring and monitoring so customers can be charged actual usage

Software as a Service (SaaS) (Cont.)

- An SaaS application must have a built-in billing service
- SaaS applications need published interfaces and an ecosystem of partners who can expand the company's customer base and market reach
- SaaS applications have to ensure that each customer's data and specialized configurations are separate and secure from other customer's data and configurations
- SaaS applications need to provide sophisticated business process configurators for customers
- SaaS applications need to constantly provide fast releases of new features and new capabilities
- SaaS applications have to protect the integrity of customer data

Software as a Service (SaaS) (Cont.)

- Software licenses are managed by the cloud provider
- The costs are shared by several customers
- Software maintenance is managed by the cloud provider
- Version tracking are made by the provider
- Hardware costs decrease at the customer
- Hardware scaling can be more easily managed at the provider in the case of mass utilization
- Possible disadvantages:
  - Network problems (bandwidth shortage)
  - Security deficiency
  - Provider dependency
  - Limited customization

The Requirements of SaaS Service Delivery Infrastructures [4]

- User perspective:
  - quick access to information
  - resource availability
  - access to latest software features
  - stored information security
  - transaction security

The Requirements of SaaS Service Delivery Infrastructures (Cont.)

- Administrator perspective:
  - easy scalable by adding new machines
  - rapid detection of infrastructure failures
  - fast replacement of damaged machines
  - monitoring capabilities
  - automatic reorganization of the infrastructure on failure
  - upgrade software without service interruptions
  - ability to statistically determine when and how the infrastructure can fail or lose data

The Requirements of SaaS Service Delivery Infrastructures (Cont.)

- Developer perspective:
  - separation of the application code from the infrastructure code
  - availability of structured and unstructured storage services
  - availability of communication services in the infrastructure
  - infrastructure complexity not exposed to the application

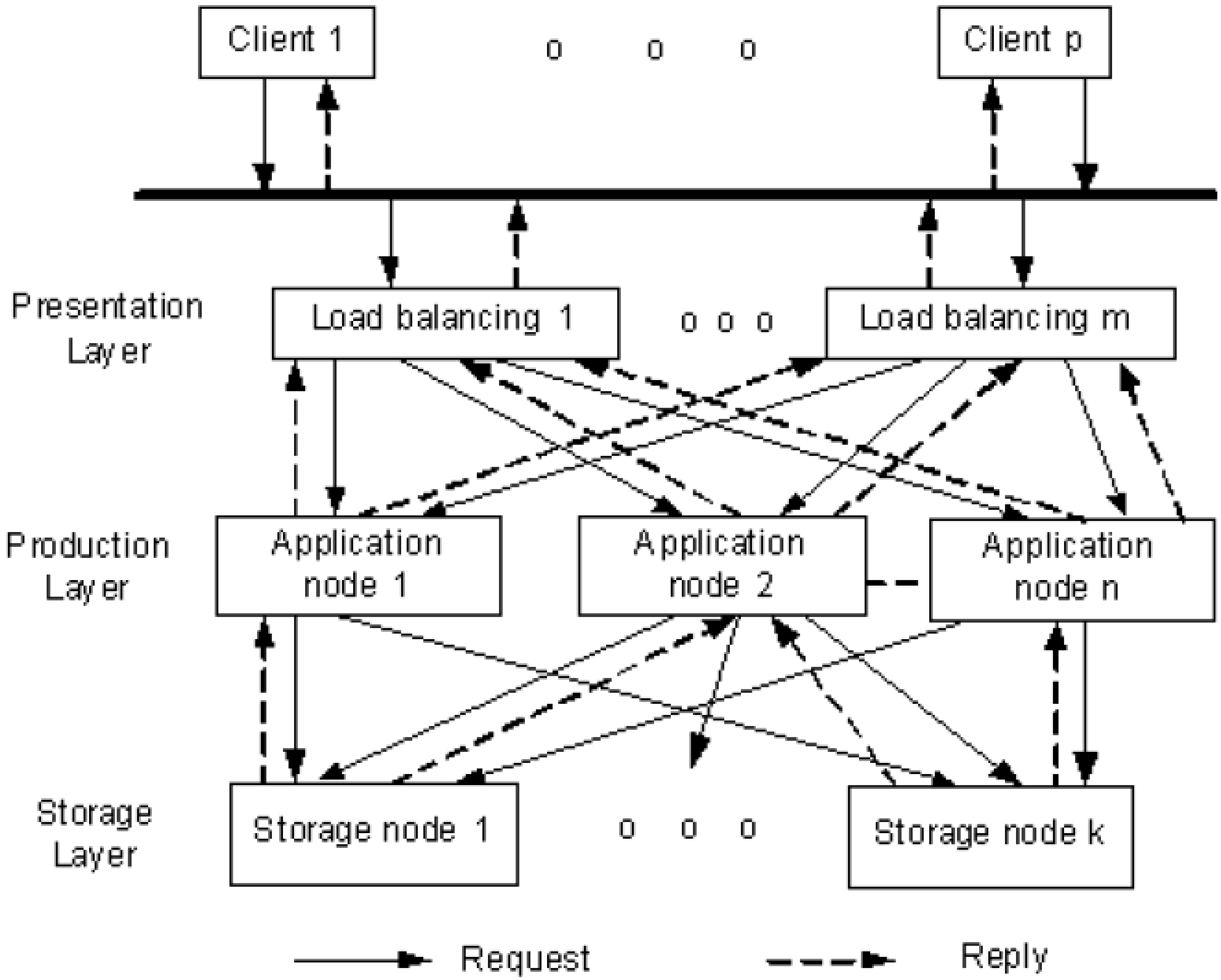
The Requirements of SaaS Service Delivery Infrastructures (Cont.)

- Other perspective:
  - provider's independence from the hardware vendor
  - customer independence from the cloud provider
  - proprietary software independence
  - resistance to disasters
  - predictable operating cost

SaaS Cloud Architecture [4]

- The cloud is built using virtualization because it provides the necessary methods to divide resources and to adjust them in real-time
- In most cloud implementations nodes are virtual machines, abstracted through the virtualization technology
- Virtual servers provide resource control features, which is very important especially when powerful hardware machines with many processing units are used
- This logical structure of the infrastructure allows a good scalability, reduces maintenance costs, and enables the application's integration in the hardware
- The infrastructure is divided in three different layers:
  - Presentation layer
  - Production layer
  - Storage layer

SaaS Cloud Architecture [4] (Cont.)



SaaS Cloud Architecture – Presentation layer

- The presentation layer has the nodes that take the requests of the clients and distribute them to the Production layer
- There are utilization scenarios that allow the presentation nodes to answer to requests directly instead of forwarding them to the Production Layer
- Nodes are able to process a large number of requests because all operations are very fast as they do not require important computing resources
- A single node is able to distribute requests to tens or even hundreds of node on the Production layer
- To dimension the Presentation layer it is necessary to know the number of requests received by the infrastructure and the capacity of the nodes on this layer
- Scaling can be performed while infrastructure is operating without service interruptions
- There are several techniques used on this layer to perform request distribution and load balancing
- Most of the solutions operates on the Transport (Layer 4) and the Application (Layer 7) OSI layer

SaaS Cloud Architecture – Presentation layer (Cont.)

- Implementation in the Transport OSI layer:
  - This is considered the least sophisticated technique, but it also consumes less resources
  - The node distributes the requests received on a specific port and protocol to the Production layer nodes
  - For some higher level protocols, problems can occur with this type of distribution because the requests need to preserve session information
  - This type of distribution can be implemented in hardware devices that use specialized processors designed for this purpose (there are devices capable of distributing up to 10 gbps)

SaaS Cloud Architecture – Presentation layer (Cont.)

- Implementation in the Application OSI layer:
  - The logic used to implement this technique is more complex
  - To distribute HTTP requests, a specialized server for load balancing, a reverse proxy, a redirect server etc. is used
  - Implementation is done using session cookies, browser applications recognize them and assigns them to a session
  - Requests from the same client will get distributed to the same node on the production layer
  - Performance is moderate because there is a lot of processing involved in order to perform the distribution

SaaS Cloud Architecture – Presentation layer (Cont.)

- The distribution techniques can be combined
- Initially, distribution was done mainly using dedicated network equipment, because regular servers were not powerful enough to handle the traffic
- The dedicated load balancing devices are complex
- These solutions are not very flexible in configuration
- There is no problem to implement a cloud with one node on the Presentation layer and several nodes on the Production layer
- Adding new nodes and removing existing nodes is a difficult and non-automated process, especially on large implementations with thousands of nodes

SaaS Cloud Architecture – Presentation layer (Cont.)

- Many implementations consider that a single node on the Presentation layer is enough to handle all the traffic, which is not correct for all deployments
- When multiple nodes need, the distribution of requests between them is usually performed using the DNS service
- The DNS does not know anything about the availability of the Presentation layer nodes, and even if notified, it would be unfeasible to change DNS records due to client record caching
- A more effective solution is for a note on the same layer to take over the IP from the failed machine and to distribute requests on its behalf. The operation is implemented using gratuitous ARP
- Choosing the most appropriate solution for the Presentation layer depends on the project
- It needs that the Presentation layer gets information about the load of the Production layer nodes in order to help the balancing algorithms to perform a fair distribution

SaaS Cloud Architecture – Production layer

- Production layer nodes implement the application logic
- This logic is responsible for handling the requests received from the Presentation layer
- This layer can auto-scale to handle the load during peak times
- This layer is the least affected by the infrastructure organization because it receives the requests forwarded by the Presentation layer and uses the services made available by the Storage layer
- Some differences compared to traditional architectures:
  - more nodes answer to requests in parallel
  - the communication between the nodes on the Production layer should be avoided
  - access to storage structures is unified
  - the layer is isolated from the outside

SaaS Cloud Architecture – Storage layer

- Storage layer is responsible for data storage
- It acts as a service for the Production layer
- It is a vital component of the infrastructure because information can be lost or tampered only at this level
- Challenges are:
    - the huge amount of information accessible in real-time
    - scalability expectations
    - security constraints
    - high availability
- Traditional web applications use the structured storage: SQL relational databases
- These databases don’t scale linearly on multiple servers (horizontal scaling)
- There are databases which can scale vertically (by upgrading hardware)

SaaS Cloud Architecture – Storage layer (Cont.)

- Applications started to avoid using SQL structured databases, but there is still a huge demand on relational databases
- This is satisfied by SQL as a service cloud offers, although such offers are limited to a maximum number of transactions per unit time
- The SQL provided as a service on the Storage layer might be highly available, but it cannot scale. Such services are usually implemented on virtual machines, in a very similar way to traditional hosting (examples: Amazon RDS MySQL, PostgreSQL, Heroku).
- Alternate solutions provide important advantages such as availability, speed and scalability, but sacrifice consistency
- E.g.: Google BigTable is a non-relational, distributed database
- Characteristics:
  - It exposes a low number of operations
  - It doesn't have JOIN operations
  - Its storage format is not column oriented
  - Information is distributed on multiple nodes and localized by the client application

SaaS Cloud Architecture – Storage layer (Cont.)

- Many cloud applications use local storage, but the cloud storage layer should provide support for both local and remote storage via the Internet
- Cloud storage platforms implement different techniques and answer to different needs
- For example, Amazon's Simple Storage Service (S3) provides an unstructured storage facility accessible using the Internet
- The model is:
  - It stores objects as binary content in configurable containers
  - Software applications can create, read and delete objects and containers
  - Objects can also be updated, but in reality they are completely replaced
  - This simple, but limited storage service is easily scalable
  - Application developers have access to cheap storage space in the cloud, but they must work harder to use it

SaaS Cloud Architecture – Storage layer (Cont.)

- Cloud information can be stored in structured or unstructured format
- Structured information is strictly formatted and it is stored in such a way that various operations can be performed on it (e.g.: XML / XHTML, SQL databases)
- Structured information is organized to identify and separate content from context information
- Unstructured information is not organized in a way to allow any type separation
- From unstructured information it is not possible to automatically extract properties and relationships
- Unstructured information is the generated content such as audio, video, graphics and documents
- Each type of content, structured or unstructured has different particularities when it comes to how storage and search operations are performed

[1]: http://www.cloud.com, Cloud Computing Outlook, 2011

[2]: http://cloudtaxonomy.opencrowd.com/taxonomy, 2005

[3]: Sarna, D.E.Y.: Implementing and Developing Cloud Computing Applications, Auerbach Publications, 2011

[4]: Cârstoiu, B.: Cloud SaaS infrastructure, Control and Computers Faculty, University Politehnica of Bucharest, Romania

[5]: Dean J., Ghemawat S.: MapReduce: Simplified Data Processing on Large Clusters, Google, Inc., 2004

[6]: Hurwitz J,. Bloor R., Kaufman M., Halper F.: Cloud Computing For Dummies, Wiley Publishing, Inc., 2010

[7]: Velte A. T., Velte, T. J., Ph.D., Elsenpeter R., Cloud Computing: A Practical Approach, McGraw-Hill, 2010

[8]: http://docs.google.com/: Google Docs

[9]: Amazon Simple Storage Service, Quick Reference Card (Version 2006-03-01), http://awsdocs.s3.amazonaws.com/S3/latest/s3-qrc.pdf

[10]: Running Databases on AWS: http://aws.amazon.com/running_databases/

[11]: http://www.cloudsecurityalliance.org/guidance/csaguide.v2.1.pdf: Security Guidance for Critical Areas of Focus in Cloud Computing V2.1, Cloud Security Alliance, 2009

[12]: http://hadoop.apache.org/: Apache™ Hadoop™

[13]: http://code.google.com/intl/hu-HU/appengine/docs/whatisgoogleappengine.html: Google App Engine, 2011

Cloud Computing

Infrastructure as a Service

Gergely Windisch

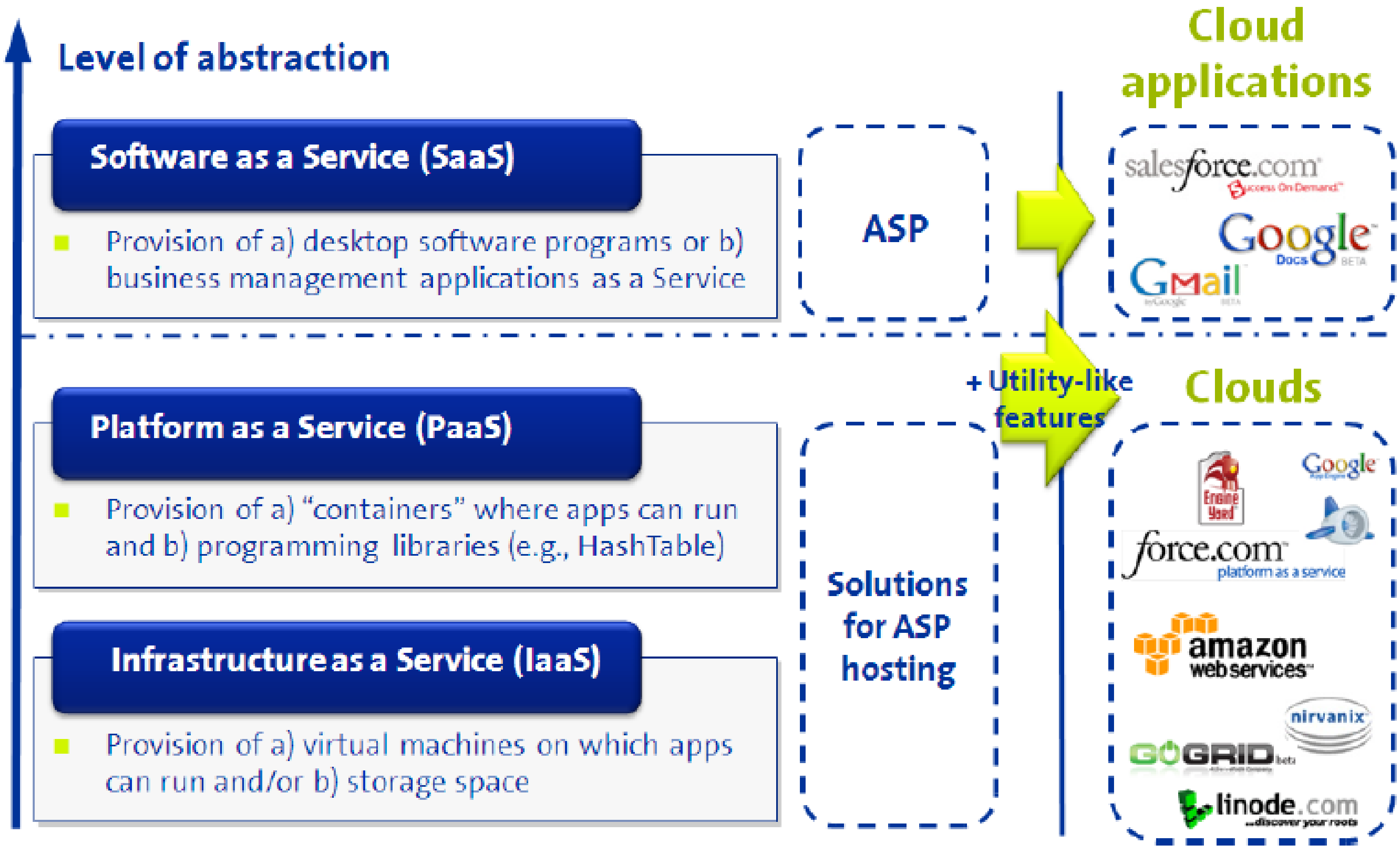
Table of Contents

Introduction to the class

- 6 classes - 2 hours each
- Classes
  - 1st: Introduction to clouds in practice
    - Recap on what clouds are (30 min)
    - Try a public cloud infrastructure (StratosLab) (60 min)
  - 2nd: Planning and implementing OpenNebula infrastructure
    - Introduction to OpenNebula, planning the infrastructure (30 min)
    - Create the virtual machines, install the operating systems (60 min)
  - 3rd: Implementing the infrastructure
    - Continue installing the operating systems (45 min)
    - Configure the operating systems (45 min)
  - 4th: Implementing the infrastructure
    - Configure the operating systems (45 min)
    - Install the OpenNebula toolkit (45 min)
  - 5th: Using the OpenNebula toolkit
    - VM lifecycle management
  - 6th: Using the OpenNebula toolkit
    - Advanced management, template image creation

1 Infrastructure as a Service

Infrastructure as a Service (1)



Source: 4CaaS Project [7]

Infrastructure as a Service (2)

- Cloud computing is an elastic infrastructure model
- Treats IT as a service provider
  - I want my company to appear on the web
    - 1 month of website presence: 4 $
    - 0.1$ / 10000 visitors
    - 1$ / 5 GB downloaded
    - Total for January: 5.4 $
  - As opposed to the traditional method: buy server, put it in the data center, install OS, install web server, buy new parts when more power is needed etc.
  - Utility computing - IT like water or electricity
- More of about a way of thinking than technology
  - The technology has been around for a long time
  - Cloud is a buzzword - they use if for everything

Infrastructure as a Service (3)

- Infrastructure as a Service
  - The cloud provider delivers a computer infrastructure
  - The parameters of that infrastructure be whatever we need (and want to pay for)
    - Number of computers
    - Virtual or physical (depending on the provider)
    - Number and performance of the CPU-s
    - Memory size
    - Disks
  - The provider usually charges on a utility computing basis: the cost reflects the amount of resources consumed (i.e. you pay by the hour (whenever the machine was turned on))
  - Payment models
    - On demand: switch it on for two days, pay only for that
    - Reserved: lease the machine for a given period of time (months), pay for the whole period-cheaper in the long run
- Public IaaS providers
  - Lots, the most well known is Amazon EC2

Why clouds are nothing new (at least as far as the technology is concerned)

- SaaS
  - Yahoo webmail has been around for 15 years. We used to call it webmail. Now it would be called Software as a Service.
- IaaS
  - Virtualization is not a new concept, the roots go back to the '60s
  - It has been possible for a long time to have a powerful computer in a datacenter, run virtualization on it. The administrator would create a new virtual machine, install the OS on it, when it is ready, the user could log in and use it.
    - Same thing happens in a cloud, but it happens automatically
- The base line for cloud computing is
  - Automation
  - Self service
  - Saving cost on employees (with the automated datacenters large companies need about 5 administrators for 10000 machines)
    - May not be too late for you to start a different carrier :)

Cloud computing (IaaS) types (1) [1], [2]

- Public cloud
- Private cloud
- Hybrid cloud

Cloud computing (IaaS) types (2)

- Public cloud
  - Implementation
    - Virtual machines are running at a cloud provider
    - Data is stored in their servers
  - Pricing
    - Cheap to start - no need for any investment in computers / datacenters
    - More expensive in the long run than private clouds
  - Security
    - Data security can be an issue (the sensitive data of the company sitting in the datacenters of a cloud provider might not be a good idea)
- Private cloud
- Hybrid cloud

Cloud computing (IaaS) types (3)

- Public cloud
- Private cloud
  - Implementation
    - The company runs it's own cloud infrastructure.
    - The virtual machines are in the local data centers
    - Datacenter management is up to the administrators
      - backups
      - updates
      - physical security
  - Pricing
    - Large initial costs of installing the data centers
    - Gets cheaper over time as there is no need to pay rent
  - Security
    - Data security is better than for the public clouds - sensitive data never has to leave the borders of the company
- Hybrid cloud

Cloud computing (IaaS) types (4)

- Public cloud
- Private cloud
- Hybrid cloud
  - Part of the infrastructure is in the local datacenter
  - Whenever more computation power / storage is needed, it can be rented from a public provider
  - Essential to have compatible technologies locally and publicly

Evolution of computing that lead to IaaS

- Supercomputers
  - very expensive
  - huge capacity in one coherent unit
  - one platform
  - few specialized applications
- Clusters
  - tightly coupled array of "ordinary" computers, tuned to different needs
    - Load balance cluster
    - HA cluster
    - Compute cluster
- Grid computing
  - Large number of loosely coupled computers
  - Capable of executing a large number of specialized jobs through a middle layer

IaaS

- With Infrastructure as a Service you rent a virtual/physical machine
- During the rest of the semester we will
  - Try a public IaaS as a user
  - Create our own IaaS infrastructure
  - Provide virtual machine infrastructure to users

Scalability

- One of the big advantage of using a (public) cloud infrastructure is the possibility of unlimited scaling
- On the Infrastructure as a Service level scaling comes from two factors
  - The machine itself
  - The whole infrastructure [15]
- In an IaaS environment the users can choose the performance of their computers
  - A small, slow virtual machine usually costs a lot less, but even that can be enough for a company that is just starting up
  - When the company grows and gets more and more visits on the web site, the machine can be reconfigured to be more powerful
- There is also a scaling potential at the infrastructure level
  - when one server is not enough, the client can ask for additional servers
  - This however does not come cheap, as the underlying software also has to be scalable

Use case examples (1)

- Education
  - Operating systems class for example (linux scripting)
  - Each student needs access to a linux machine
  - Possible solutions
    - Thick clients - 1 computer with linux installed as the operating system
      - Problems with management (how to separate different users of the same computer, what if one computer fails, what if the next group needs different OS for another subject)
    - Thick clients - any host OS, virtualized linux running on each machine
      - Solves the issue of different OS needs (next group runs a different virtual machine)
      - Still not easy to manage (takes a long time to copy the necessary virtual machines to the hosts, hard to update them etc.)
    - Thick clients - any host OS, virtualized linux stored on a network share, but running on the client
      - Solves the problem of updating, copying, but requires an enormous bandwidth

Use case examples (2)

- Education
  - Operating systems class for example (linux scripting)
  - Each student needs access to a linux machine
  - Possible solutions (2) - Thin clients
    - Thin clients, one big linux server the students are connected to via ssh
      - Easy to manage (if the admin updates the server, the students see the updated version)
      - Hard to separate the students from each other

Use case examples (3)

- Education
  - Operating systems class for example (linux scripting)
  - Each student needs access to a linux machine
  - Possible solutions (3) - Thin clients
    - Thin clients, each student has their own virtual machine running somewhere in a datacenter
      - Possibly the best solution
      - Not tied to any specific computer room / device
        - » Need to go from one room to the other? No problem, users log out, and then connect again using ssh from the other room
        - » Students can use any client device as long as it has an ssh client
      - We could put the "Cloud" and "IaaS" labels on this approach if it is necessary

2 Public IaaS

Public IaaS (1)

- Private cloud means running the virtualized computers in the datacenters of a company
- There are advantages and disadvantages to this solution
  - Which one is better depends on
    - what the company want
    - how long they need the resources
    - what kind of resources do they need
    - whether they can afford large spending on infrastructure up front
    - whether they have the right personnel to administer the infrastructure efficiently
- All this has already been discussed in the first part of the semester, so now we will only talk about clouds in the technical sense

Public IaaS (2)

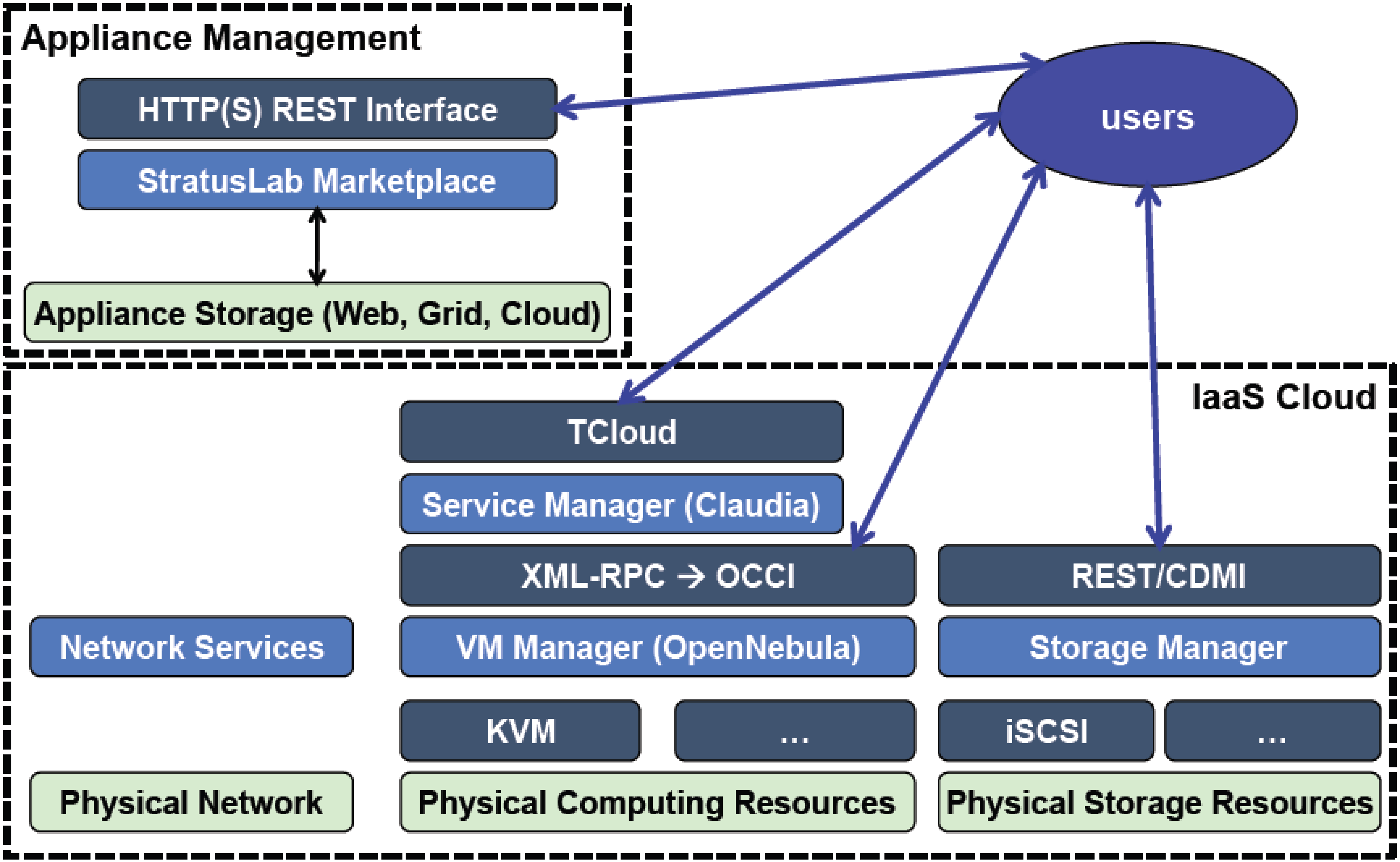
- There are many cloud providers that offer their services in the IaaS field
- For example
  - Amazon EC2 [3]
  - Salesforce [8]
  - Canonical [5]
  - RackSpace [4]
  - StratusLab [6]
- The method is usually the same, one can visit their website, choose the type of virtual machine to use, deploy it, and start using it within a few minutes.
- Some offer free trials
  - Amazon EC2 gives new users 1 whole year free for a micro instance (1GHz CPU, 600MB RAM, Ubuntu linux)
    - They ask for a credit card, but don't charge anything (at least as long as you are not overusing your account)
  - StratusLab allows research institutions to access their infrastructure and they were kind enough to provide us with usernames and passwords, so we'll use that infrastructure

2.1 StratusLab

Introduction to StratusLab [9]

- StratusLab is an Open Source cloud platform which can be downloaded and installed as a private cloud
  - It is also the name of the company that created it and provides public IaaS
- Goals
  - Create comprehensive and open source IaaS cloud distribution
- Implementation
  - StratusLab is an enhancement to the open source OpenNebula toolkit
    - OpenNebula provides the basic services (start, stop, kill)
  - Enhancements over the standard OpenNebula
    - Quarantine of stopped images
    - Improved logging
    - Improved fault tolerance
    - Improved management of network addresses
    - Support for user groups and roles
- Contact
  - http://stratuslab.eu

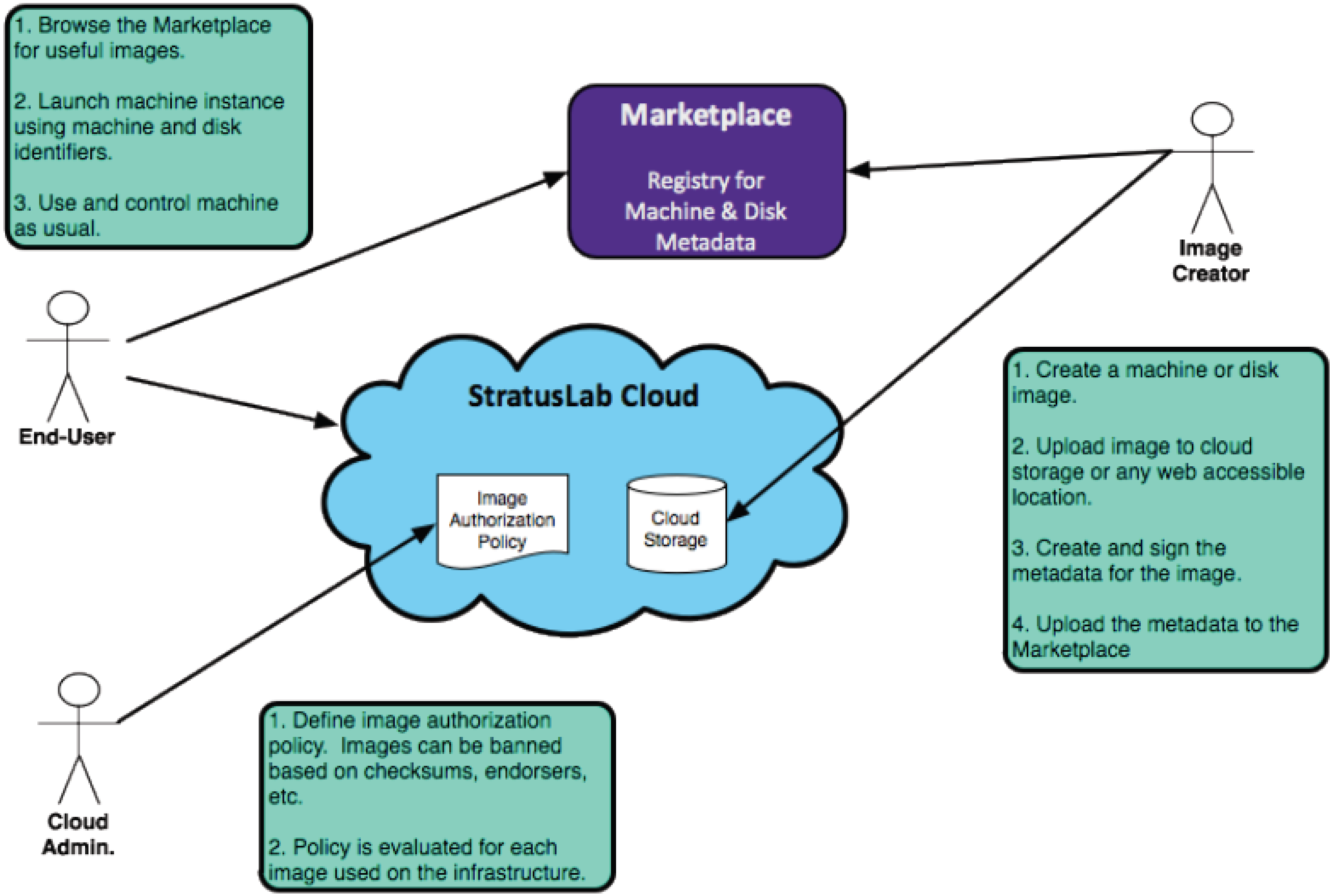
Architecture of StratusLab



StratusLab Services

- Storage services
  - Persistent disks
    - Allows storage for users and VMs
    - Can be mounted to one VM at a time
  - Static disks
    - Read only disk images
- Network services
  - Public IP
  - Local IP
- Marketplace
  - Images can be stored at the StratusLab marketplace
  - Easier to find and use the image required
- Claudia - Service manager
- Authentication through proxy service
- Registration service
- Accounting / monitoring

Marketplace workflows



Goals of StratusLab

- Offer remote access to computing resources
- Advantages
  - Customized environments
  - Rapid access via simple API
  - Complete control with root access
- Disadvantages
  - Interface is not standardized
  - It is difficult to create new virtual machines

Getting started with the StratusLab public IaaS

- In order to use the service provided by StratusLab, one has to
  - register at StratusLab.eu
    - click on the Support button, and write an email requesting an account to support@stratuslab.eu
    - We can skip this step as a group account has already been required for the whole class
  - install the StratusLab client
    - StratusLab client is a command line toolkit that can manage the lifecycle of the virtual machines
  - visit the marketplace to see the available images, and get the URL or marketplace ID that you have to use
    - http://appliances.stratuslab.eu:8081/metadata
  - use the client toolkit to create virtual machines
  - use the virtual machines

VM Lifecycle Overview

- StratusLab commands to manage VMs
  - `stratus-run-instance <VM URL>`
    - deploy new instance using the URL of the machine
  - `stratus-run-instance <VM Marketplace ID>`
    - deploy new instance using the Marketplace ID
  - `stratus-describe-instance [VM ID]`
    - get information about the running instances
  - `ssh root@IP`
    - connect to the machine
  - `stratus-kill-instance <VM ID>`

Installing the client (1)

- The client can be installed on many platforms (linux, Mac OSx, Windows)
- Dependencies
  - Python 2.6+
  - Java 1.6+
  - SSH client
  - Certificate
    - you can download a sample certificate from http://nik.uni-obuda.hu/gwindisch/cloud/cert
- Install the dependencies on the Ubuntu virtual machine
  - `sudo apt-get install python-2.6`
  - `sudo apt-get install jvm`
- Download and untar the tarball containing the toolkit from
  - http://repo.stratuslab.eu:8081/content/repositories/fedora-14-releases/
  - `mkdir $HOME/stratuslab`
  - `tar xvzf xxxx.tar.gz $HOME/stratuslab`

Installing the client (2)

- Configure path variables
  - `export PATH=$HOME/stratuslab/bin:$PATH`
  - `export PYTHONPATH=$HOME/stratuslab/lib/stratuslab/python`
- Test the installation
  - stratus-run-instance --help
  - if it returns with information on how to use the command, then everything is fine.
  - if it gives an error message, try again (make sure the dependencies are installed)

Installing the client (3)

- Set up keyless authentication
  - In order to log in you'll need an ssh public/private keypair
  - use ssh-keygen to generate the keys
    - do not set passphrase - it is required for automatic management
- Set up the environment
  - `export STRATUSLAB_KEY=$HOME/.ssh/id_rsa.pub`
  - `mkdir $HOME/.stratuslab`
  - `cp $HOME/stratuslab/conf/stratuslab-user.cfg.ref ~/.stratuslab/stratuslab-user.cfg`
  - Edit the file to set the following parameters
    - endpoint = cloud-gmet.stratuslab.eu
    - username = YourUserName
    - password = YourPassword
    - key = $HOME/.ssh/id_rsa.pub
- We will use password authentication. StratusLab also supports LDAP and grid certificates

Test the installed client

- stratus-describe-instance
  - This command logs in to the cloud with your credentials, and prints the status of your account
- You should see something similar:

```
$ stratus-describe-instance
id    state    cpu     memory   ip       name
```

- The list is currently empty, because you are not running any virtual machines.
- If you see something else, raise your hand

StatusLab Marketplace

- The Marketplace makes it easy to reuse virtual machines (either public or ones you've created)
  - Stores authenticated, secure virtual machines
  - Images are stored in cloud, grid or web storage
- End users can browse the existing images
- Creators can upload new images
- Cloud administrators evaluate the trustworthiness of the images

StatusLab Marketplace interfaces

- REST interface
  - The marketplace offers an HTTP-based REST interface
  - Can be easily manipulated with different programming languages
- Web interface
  - the REST interface allows access via the web browsers

- Access the marketplace at
  - http://appliances.stratuslab.eu:8081/metadata

Web Portal (1)

http://appliances.stratuslab.eu:8081/metadata

Home | Endorsers | Query | Upload | About

Metadata

Show 10 entries

Search all columns:

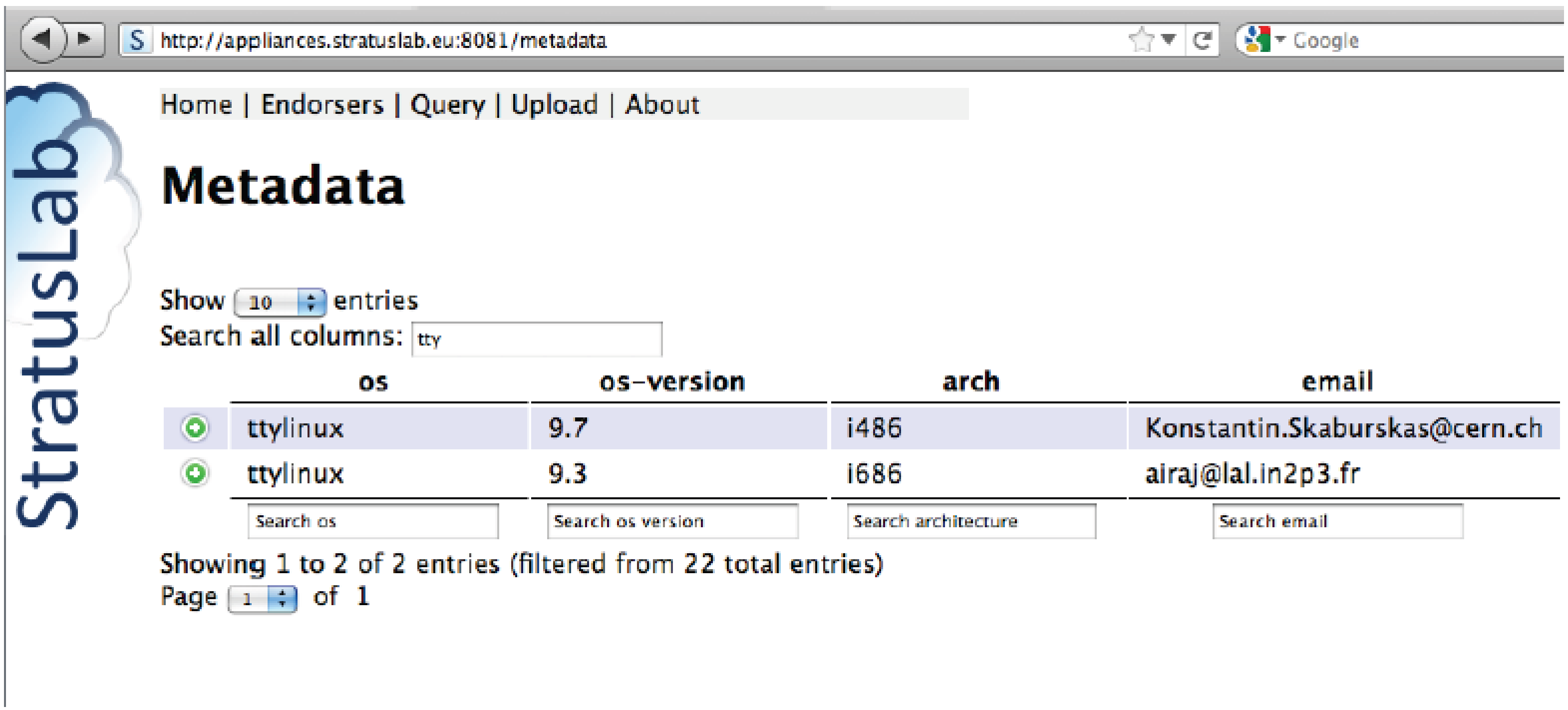
	os	os-version	arch	email
	CentOS	5.5	x86_64	Konstantin.Skaburskas@cern.ch
	centos	5.5	x86_64	vfloros@admin.grnet.gr
	CentOS	5.5	x86_64	Konstantin.Skaburskas@cern.ch
	CentOS	5.5	x86_64	vfloros@admin.grnet.gr
	CentOS	5.5	x86_64	christophe.blanchet@ibcp.fr
	CentOS	5.5	x86_64	Konstantin.Skaburskas@cern.ch
	centos	5.5	x86_64	Konstantin.Skaburskas@cern.ch
	centos	5.5	x86_64	christophe.blanchet@ibcp.fr
	CentOS	5.5	x86_64	christophe.blanchet@ibcp.fr
	CentOS	5.5	x86_64	airaj@lal.in2p3.fr
	Search os	Search os version	Search architecture	Search email

Showing 1 to 10 of 23 entries

Page 1 of 3

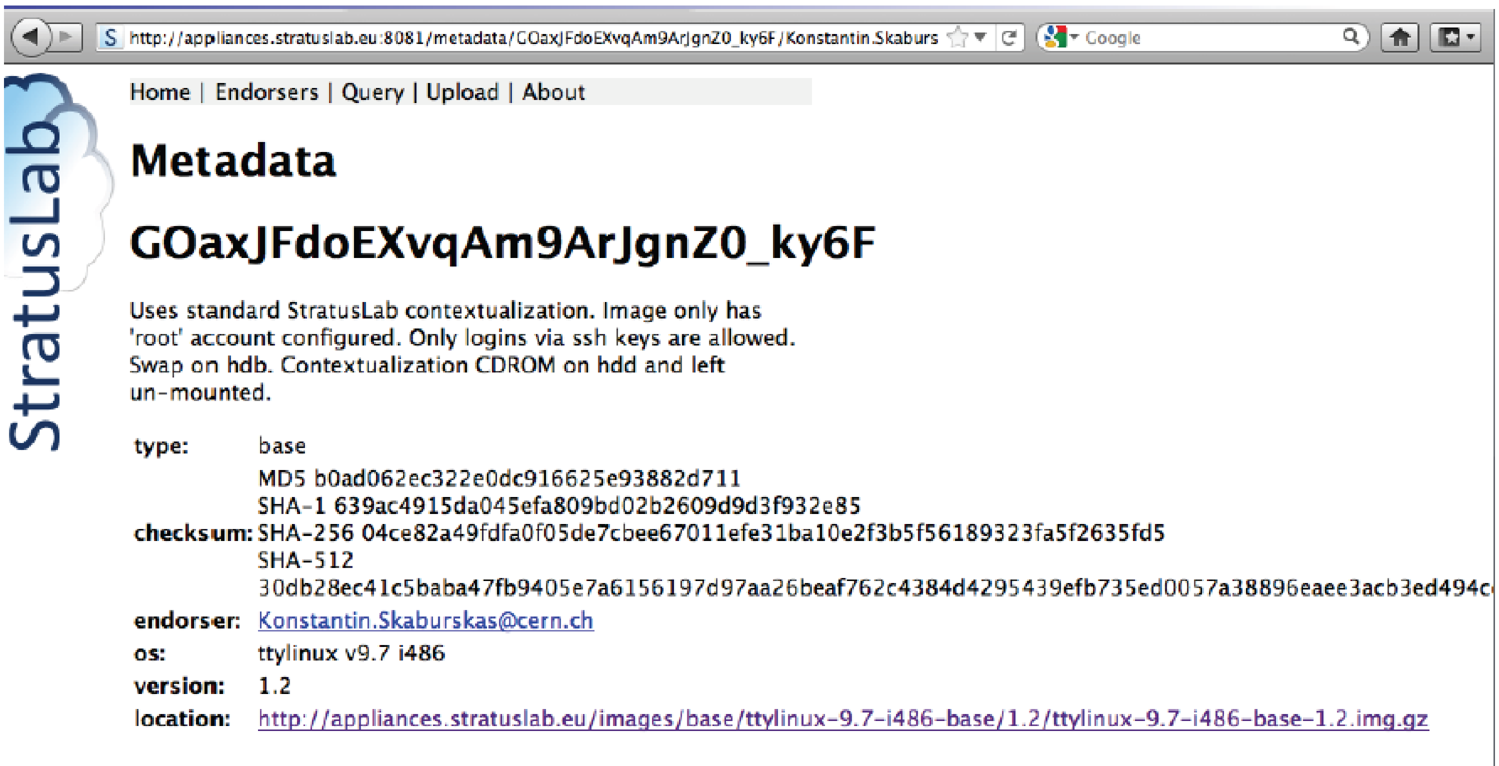
Web Portal (2)

- Use the search box to look for a specific image



Web Portal (3)

- Click on any image to see the details
  - The GOax… is the Marketplace ID, you can use that to create your VM (Contextualize the image)
  - The endorser shows who created the image
  - The OS denotes what the image operating system is
  - The location is the URL that you can also use to contextualize the VM



StratusLab VM lifecycle (1)

- Deploying a virtual machine
  - We will try to contextualize the image called ttylinux
  - export FIRSTIMAGE=http://appliances.stratuslab.org/images/base/ttylinux-9.7-i486-base/1.2/ttylinux-9.7-i486-base-1.2.img.gz
  - status-run-instance $FIRSTIMAGE
- If all goes well, the response should be

```
:::::::::::::::::::::::
:: Starting machines ::
:::::::::::::::::::::::
:: Starting 1 machine
:: Machine 1 (vm ID: 11)
   Public ip: 134.158.75.33
:: Done!
```

- Note the vm ID and the public IP. We need the IP to connect to the machine and we can use the vm ID to manage the instance

StratusLab VM lifecycle (2)

- Status of the vm can be checked using the stratus-describe-instance command

```
$ stratus-describe-instance 11
id    state    cpu      memory   ip                 name
11    Running  1        128      134.158.75.33      one-11
```

- The state column shows the current state of the virtual machine.
- There are many states, the most important are
  - Prolog, boot
    - creating the virtual machine, copying the image from the marketplace to the computer and booting the image
  - Running
    - The image is up and running. You can log in now
  - Failed
    - Something has gone horribly wrong. Check the logs. Could be anything from bad image, wrong URL to network issues
  - Unknown
    - As the name suggests (usually it means that the agent is not functional)

StratusLab VM lifecycle (3)

- Deploy image using the marketplace ID

```
export SECONDIMAGE=GOaxJFdoEXvqAm9ArJgnZ0_ky6F
stratus-run-instance $SECONDIMAGE

:::::::::::::::::::::::
:: Starting machines ::
:::::::::::::::::::::::
:: Starting 1 machine
:: Machine 1 (vm ID: 12)
   Public ip: 134.158.75.34
:: Done
```

- Check the status

```
$ stratus-describe-instance
id state        cpu      memory   ip               name
11 Running      1        128      134.158.75.33    one-11
12 Running      1        128      134.158.75.34    one-12
```

StratusLab VM lifecycle (4)

- Check if the OS is running correctly

```
$ ping 134.158.75.33
PING 134.158.75.33 (134.158.75.33): 56 data bytes
64 bytes from 134.158.75.33: icmp_seq=0 ttl=63 time=0.780 ms
64 bytes from 134.158.75.33: icmp_seq=1 ttl=63 time=0.704 ms
```

- Log in as root

```
$ ssh root@134.158.75.33
#
# echo $USER root
root
# ifconfig
...
#hostname -a
...
```

StratusLab VM lifecycle (5)

- Shutting down the virtual machine
- The nice way (initiating shutdown from within the virtual machine)

```
# shutdown -h
# Connection to 134.158.75.33 closed by remote host.
Connection to 134.158.75.33 closed.
```

- The not so nice way (similar to ripping the cord out of the wall)

```
$ stratus-kill-instance 12
$ stratus-describe-instance 12
id state         cpu     memory   ip              name
12 Done          1       128      134.158.75.34   one-12
```

- Warning! Shutting the machine down also means destroying it, so all the data will be lost. If you need the data, you should use persistent storage

Exercise

- Imagine that you are a company that needs to have a website up and running as soon as possible
- Your task is to contextualize an Ubuntu virtual machine, install Apache, and deploy your website to the root directory of Apache

Exercise hints

- Look for the Ubuntu image in the marketplace
- Check the meta data for the information that is required to deploy the VM
- Use the commands learned in the previous slides to start and connect to the machine.
- Use apt-get to install apache2
- Look for the file /etc/apache/sites.available/sites.conf, to see where to put your html files

3 Private IaaS

Private IaaS

- As opposed to public IaaS, private IaaS means that the cloud infrastructure runs in the datacenters of the company itself
- It is similar to having virtualization in the datacenters, but it adds even more flexibility
- Some companies like this solution because the sensitive data never has to leave the premises
- Disadvantages of private clouds
  - The company has to handle everything
    - Servers, installation, cooling, backup, network, redundancy etc.
- There are some commercial products out there, they usually require commodity hardware to work and cost a lot of money, and there are free or open source alternatives. They sometimes lack the grace and elegance of a commercial product, but in pure performance, they are usually on par with the more expensive solutions

Creating your own private cloud

- Competing technologies and providers
  - HP Matrix (this is the system we have at Obuda University)
  - VMware vCloud
  - Microsoft Hyper-V cloud
  - OpenNebula
  - Openstack
  - Eucalyptus
- Most of these options are commercial
- They cost a lot of money
- We will implement a cost effective, open source cloud solution called OpenNebula

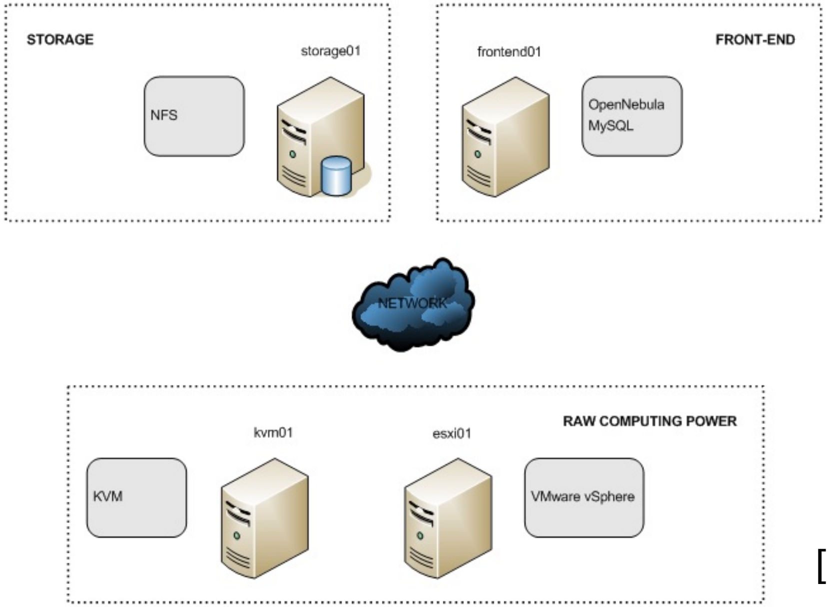
4 OpenNebula

4.1 Introduction to OpenNebula

What is OpenNebula

- From the OpenNebula website:

  "OpenNebula is a Virtual Infrastructure Manager that orchestrates storage, network and virtualization technologies to enable the dynamic placement of multi-tier services (groups of interconnected virtual machines) on distributed infrastructures, combining both data center resources and remote cloud resources, according to allocation policies."
- OpenNebula is a free, open source cloud platform that can be used as a
  - public cloud
  - private cloud
  - hybrid cloud



[12]

Why OpenNebula (in general)

- http://opennebula.org/about:why
  - "OpenNebula is an open and flexible tool that fits into existing data center environments to build any type of IaaS Cloud deployment. OpenNebula can be primarily used as a virtualization tool to manage your virtual infrastructure in the data-center or cluster, which is usually referred as Private Cloud. OpenNebula supports Hybrid Cloud to combine local infrastructure with public cloud-based infrastructure, enabling highly scalable hosting environments. OpenNebula also supports Public Clouds by providing Cloud interfaces to expose its functionality for virtual machine, storage and network management."

Why do we use OpenNebula for our class?

- Because it is free (not having to pay for something is quite good)
- Because it is open (no vendor lock in)
- Because it is open source (you can actually see how features are implemented)

- But mainly because it does not really matter which one you choose
  - The different cloud provider APIs differ in the commands, features, performance, but basically they are very similar.
  - We are learning about private clouds in general, not the vendor specific parts

How to get OpenNebula?

- Visit the OpenNebula website
  - http://www.opennebula.org
- Download the source or the executable

Basic components of OpenNebula

- The OpenNebula Daemon (oned) controls the operation of the modules that manage the lifecycle of the VMs
- The drivers access the specific parts of the infrastructure (hypervisors, storage, network)
- The scheduler decides on VM placement (where to put a newly created virtual machine)

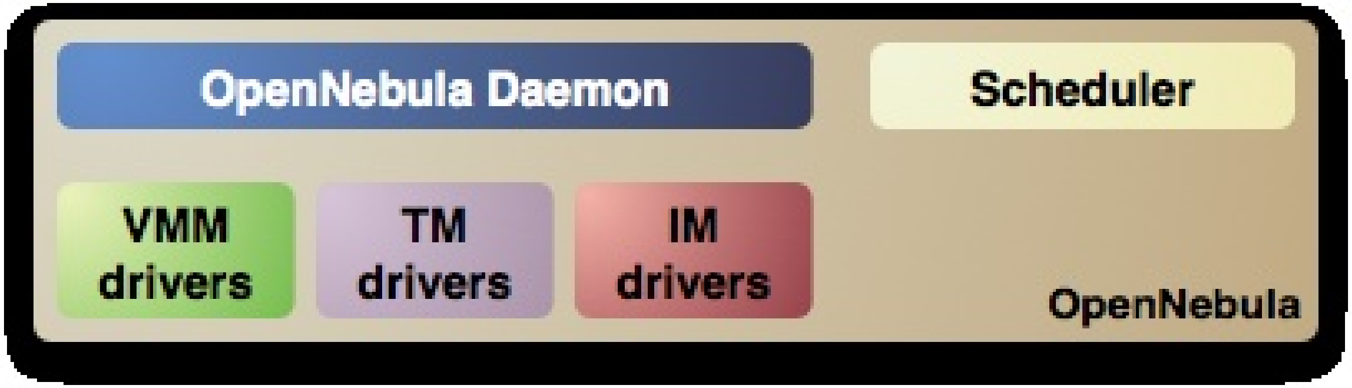


Image source: http://opennebula.org/documentation

4.2 Creating an OpenNebula cloud

4.2.1 Setting up the infrastructure

Infrastructure required by OpenNebula (1)

- Controller
  - This machine runs the OpenNebula daemon (oned), which is responsible for the lifecycle of the virtual machines (it creates them, allocates them to nodes, stops them, destroys them etc.)
  - This is a linux based computer that has all the OpenNebula scripts and programs installed
  - This machine does not need to be too powerful and it does not require virtualization support
- Node (host)
  - At least one node machine is needed - this is the computer that executes the virtual machines
  - The more nodes we have the more powerful our cloud will be
  - OpenNebula supports KVM, XEN, Vmware ESXi hypervisors for the nodes. Support for Oracle VirtualBox is on it's way

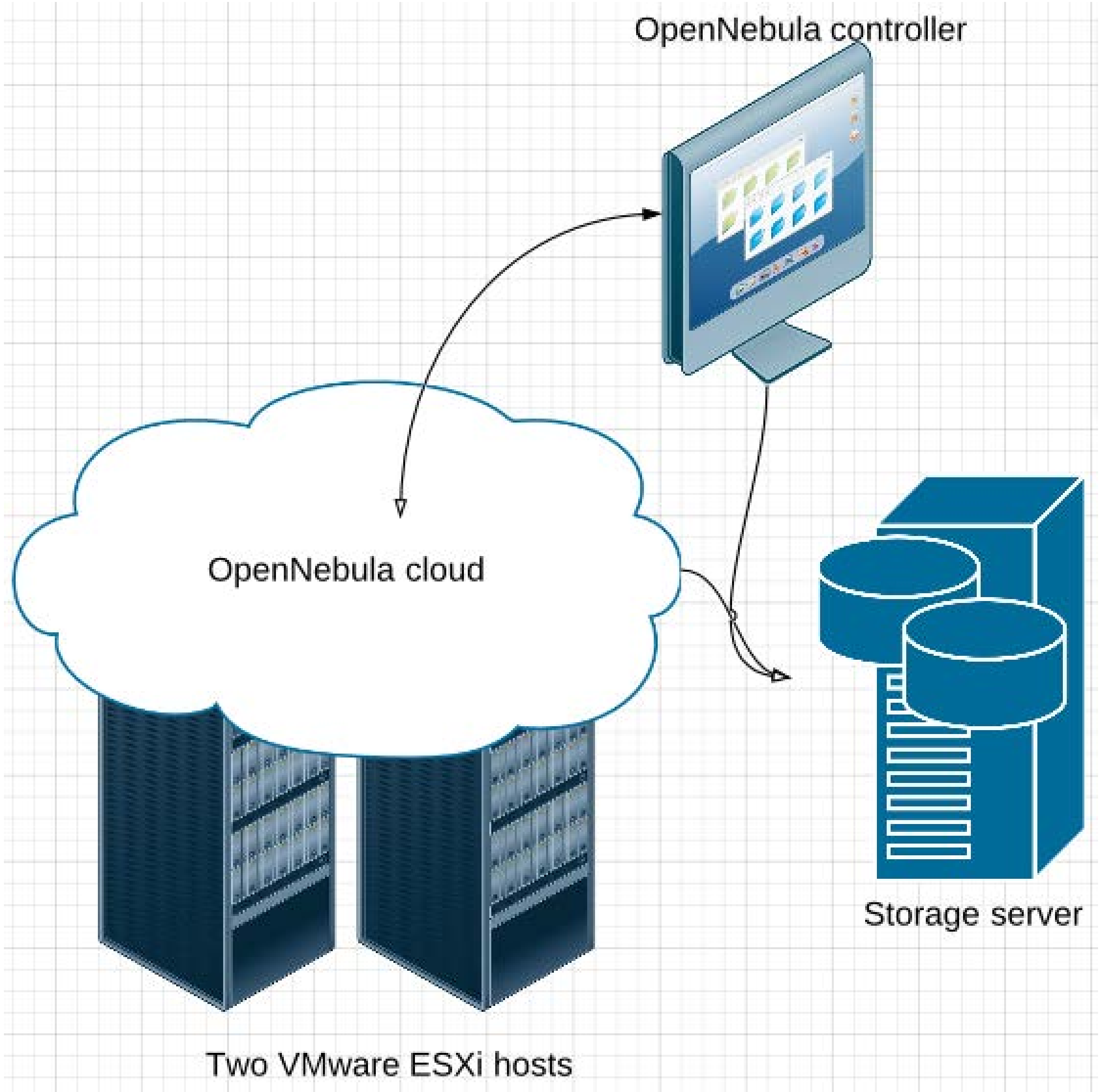
Infrastructure required by OpenNebula (2)

- Storage
  - An OpenNebula infrastructure has to store
    - Virtual Machine template images
    - Actual running instances of the virtual machines
    - Persistent data of the vm users
  - Possibilities for storage
    - Centralized storage (SAN, NAS, NFS)
      - The template images, the running instances and the persistent data are stored in one location
      - Requires dedicated storage infrastructure but offers the best performance and the more advanced features
    - Decentralized storage
      - The template vm's are stored on the controller, the running instances are stored on the nodes, the images are copied from one place to the other over ssh
      - Easy to set up, does not require storage infrastructure, but no live migration is possible

Infrastructure required by OpenNebula (3)

- Additional services
  - Mailing server
  - DNS
    - For easy access of the computers in the infrastructure
  - LDAP
- The additional services are optional, there are ways to avoid having to use them

Our planned infrastructure (1)



Our planned infrastructure (2)

- In the class we will be implementing an entirely virtualized OpenNebula cloud*
- Each student has their own high performance host
  - CPU: Core2 Quad, 2.400 GHz
  - RAM: 8 GB
  - HDD: 120 GB available
  - OS: Windows 2003
  - Virtualization technology: Vmware Workstation 7.1.4
- We will install OpenNebula 2.2 on virtual machines created over vmware
  - Controller: Ubuntu 11.04 desktop
    - 1 CPU, 786 MB RAM, 10 GB disk
    - Will handle mailing, DNS
  - Nodes: VmWare ESXi 4.1
    - 2 CPUs, 3 GB RAM, 10 GB disk each
  - Storage: Ubuntu 11.04 server with NFS support
    - 1 CPU, 768 MB RAM, 70 GB disk

*: the main purpose of a Cloud infrastructure is to have multiple high performance elements connected to achieve very high overall performance. Creating the whole cloud on one virtualized host create a huge performance penalty, so this setup is not ideal for production purposes

4.2.1 Setting up the infrastructure (6)

Preparing the infrastructure

- There are certain things that have to be done before installing OpenNebula. Here is a short list on the steps we are about to take, you can find detailed instructions on the next slides
  - create the new virtual machines, download the OS and install them
  - install some additional software, update the existing on the Ubuntu hosts
  - set up DNS service using the /etc/hosts files
  - create the NFS share on the storage server
  - install the NFS client on the controller
  - create the oneadmin user and the cloud group on both machines
  - set up public key authentication between the hosts
  - install MySQL
  - install necessary packages
- Install OpenNebula

Preparation - Download the installation images

- Software required for our project
  - Ubuntu 11.04 (64-bit) Desktop edition - OpenNebula controller
  - Ubuntu 11.04 (64-bit) Server edition - NFS storage server
  - Vmware ESXi - Cluster nodes
- Download the installation disks
  - Controller
    - http://www.ubuntu.com/download/ubuntu/download
    - Ubuntu 11.04 - Latest version
    - 64-bit
  - Storage
    - http://www.ubuntu.com/download/server/download
    - Ubuntu 11.04 - Latest version
    - 64-bit
  - Nodes
    - http://downloads.vmware.com/d/details/hypervisor_41_installable_free/ZCV0YmRoaHBiZHdldA==
    - VMware ESXi 4.1 - 60 day trial (free version will not work)
    - VMware vSphere Center - 60 day trial

Preparation - Create the virtual machines

- Create the four VMware virtual machines according to the infrastructure presented on the "Our planned infrastructure" slide
- 4 vm-s:
  - Ubuntu 11.04 desktop
  - Ubuntu 11.04 server
  - 2 VMware ESXi machines
- For a detailed, step-by-step guide on creating the virtual machines please refer to Appendix A

4.2.1 Setting up the infrastructure (9)

Preparation - Install the base virtual machines

- Install the operating systems on virtual machines
- For a detailed, step-by-step guide on installing Ubuntu 11.04 desktop, server and VMware ESXi please refer to Appendices B,C,D respectively

Preparation - Install additional software on controller and storage

- The new installation of ubuntu comes with a lot of software, but there are a few that are missing which are recommended
- Using apt-get, it is easy to install any software (called package) on Ubuntu
- Install the following packages
  - `sudo apt-get install mc`
  - `sudo apt-get install vim`
  - `sudo apt-get install nano`
  - `sudo apt-get install openssh-server`

Set up the name server

- It would be possible to use IP addresses in our project, but that could lead to confusion, so it is better to provide some kind of name service
- You could either set up a DNS server (you could use either one of the newly installed Ubuntu machines), or
- You can manually configure the hostnames manually. For this, do

```
$ sudo vim /etc/hosts
```

```
192.168.150.101     controller
192.168.150.102     storage
192.168.150.103     node01
192.168.150.104     node02
```

- Try pinging the machines to make sure that they are working properly
  - `ping controller`
  - `ping storage`
  - `ping node01`
  - `ping node02`
- Ping from each machine

4.2.2 Setting up the storage

Set up the storage

- For the shared storage, we will be using NFS
- Execute the following commands on the storage server

```
(1) user@storage:~ $ sudo mkdir -p /srv/cloud/one

(2) user@storage:~ $ sudo groupadd --gid 1001 cloud
(3) user@storage:~ $ sudo useradd --uid 1001 -g cloud -s /bin/bash -d
   /srv/cloud/one oneadmin
(4) user@storage:~ $ sudo chown -R oneadmin:cloud /srv/cloud
(4_2) user@controller:~$ sudo vim /etc/group
   admin:x:120:oneadmin
(5) user@storage:~ $ sudo apt-get install nfs-kernel-server

(6) user@storage:~ $ sudo vim /etc/exports
/srv/cloud      192.168.1.0/255.255.255.0(rw,anonuid=1001,anongid=1001)

(7) user@storage:~ $ sudo /etc/init.d/nfs-kernel-server restart
```

Command description

- (1): mkdir creates the /srv/cloud/one directory. It will be used to store all the virtual machine images (both the templates and the running instances). We will share these later on
- (2),(3): Create the group called cloud, create the user called oneadmin and assign it to the cloud group
- (4): set the oneadmin user and the cloud group as owners of the dir. cloud
- (4_2): add the user oneadmin to the admin group. You can do it by writing the username after the : on the line of the admin group. After this, it is possible to use sudo for oneadmin as well
- (5): apt-get install downloads and installs the necessary software for nfs server
- (6): edit the /etc/exports file - this is the configuration file for the nfs server. Add the line to the file (it basically states that any machine from the 192.168.1 subnet can access the /srv/cloud directory when the user asking has the UID of 1001.
- (7): the nfs-kernel-server service needs to be restarted for the changes in the export file to take an effect

4.2.3 Setting up the controller

Set up the storage (2)

- We have to set the shared storage on the controller machine as well
- Execute the following commands on the controller server

```
 (8)  user@controller:~$ sudo apt-get install nfs-common
                            sudo modprobe nfs
 (9)  user@controller:~$ sudo mkdir -p /srv/cloud
(10) user@controller:~$ sudo vim /etc/fstab
          add the following line:
      storage:/srv/cloud /srv/cloud      nfs4    _netdev,auto    0 0
(11) user@controller:~$ sudo mount -a
(12) user@controller:~$ ls -l /srv/cloud
```

Command description

- (8): install the nfs-common package which is needed to access the nfs share, then load the module with modprobe
- (9): create the /srv/cloud directory on the machine
- (10): add the line to the /etc/fstab file. The fstab file contains the drives (regular and network) that can be mounted (attached) automatically. We are setting that the share named /src/cloud on the storage machine should be mounted to the /srv/cloud directory automatically upon boot
- (11): mount -a mounts all the automatic mounts - this only has to be done if we don't want to restart the system as that would result in mounting the nfs share
- (12): make sure that the share is working - if you see the directory called one, then it works

Setting up the oneadmin user on the controller

- The OpenNebula administrator is called oneadmin. It has to be created on the controller as well
- It has to have the same user ID and group ID as that on the storage server

```
(13) user@controller:~$ sudo groupadd cloud
(14) user@controller:~$ sudo useradd -s /bin/bash -d /srv/cloud/one -g
   cloud oneadmin
(15) user@controller:~$ sudo id oneadmin
   uid=1001(oneadmin) gid=1001(cloud) groups=1001(cloud)
(15_2) user@controller:~$ sudo vim /etc/group
   admin:x:120:oneadmin
```

Important! The home directory of oneadmin is set to be on a network share. It is essential that the share is available when we log in with oneadmin. Always turn on the storage server first.

Command description

- (13),(14): Create the group called cloud, create the user called oneadmin and assign it to the cloud group
- (15): check the properties of the new user
- (15_2): add the user oneadmin to the admin group. You can do it by writing the username after the : on the line of the admin group. After this, it is possible to use sudo for oneadmin as well

Setting up environmental variables for the users

- Log in as the newly created oneadmin
  - sudo su - oneadmin
- Add the following lines to .bashrc

```
oneadmin@controller:~$ vim .bashrc
```

- `export ONE_AUTH=$HOME/.one/one_auth`
- `export ONE_LOCATION=/srv/cloud/one`
- `export ONE_XMLRPC=http://localhost:2633/RPC2`
- `export PATH=$ONE_LOCATION/bin:$PATH`

4.2.3 Setting up the controller (6)

Create the directories

- (16)   mkdir .one
- (17)   vim .one/one_auth

oneadmin:xxxxxxx

Generate keys for oneadmin

- The management of machines is done using ssh - a secure shell that is used to log in to remote machines.
- By default, a username and password is needed for logging in
- Another option is to use private / public key pairs for authentication*. It is a more secure way of authentication, and it also enables logging in without a password. Passwordless authentication is essential for OpenNebula
- Execute the following commands to have passwordless authentication

```
(18)  oneadmin@controller:~$ ssh-keygen
   When it asks anything, just press enter. Do not provide passphrase!
(19)  oneadmin@controller:~$ cat .ssh/id_rsa.pub >> .ssh/authorized_keys
(20)  oneadmin@controller:~$ vim .ssh/config
Host *
    StrictHostKeyChecking no
(21)  oneadmin@controller:~$ touch .hushlogin
```

*: More information on public key authentication: http://www.debian-administration.org/articles/530

Command description

- (18): ssh-keygen creates the public/private keypair that can be used for authentication. It asks for the location where the keypair should be stored (leave the default value), and asks for a passphrase. Do not use a passphrase, because then it is possible to log in to the remote machines without providing a password
- (19): The authorized_keys file contains the public keys that can be used for passwordless login
- (20): The .ssh/config file configures extra parameters for the ssh client. If we add the line Host * StrictHostKeyChecking no, the client will not give error messages, saying that the key of the server cannot be verified.
- (21): .hushlogin is a file that is needed for silent logins - no banner will be shown - it is essential for the automatic login methodology of OpenNebula

Check to see if everything works

- What we should have
  - network connection between the two servers
    - `oneadmin@controller ~$ ping storage`
    - `oneadmin@storage ~$ ping controller`
  - user oneadmin and group cloud on both servers, with the same UID and GID
    - both machines: grep oneadmin /etc/passwd
      grep cloud /etc/group
    - should have identical lines on the two servers
  - shared home directory at /srv/cloud
    - both machines: echo $HOME
      touch ~/newfile - on the other machine, make sure newfile exists
  - correct filenames and permissions on the shared storage
    - both machines: ls -l ~ - owner and group should be the same
  - public key authentication working for the user oneadmin
    - `oneadmin@controller ~$ ssh storage`
    - `oneadmin@storage ~$ ssh controller` - works, if logs in without password

Install the MySQL database

- The MySQL database will be used to store the data of the infrastructure
- MySQL server will be installed, and a database called opennebula has to be created
- Execute the following commands

```
(22) oneadmin@controller:~$ sudo apt-get install mysql-server
   libmysql++-                                             dev libxml2-
   dev
```

It will ask for a password. It is possible to set whatever you want, but please try not to forget what you set here. I recommend using the very secure root as the password.

```
(23) oneadmin@controller:~$ sudo mysql_secure_installation
(24) oneadmin@controller:~$ mysql -u root -p
(25) mysql> GRANT ALL PRIVILEGES ON opennebula.* TO 'oneadmin'
   IDENTIFIED BY 'xxxxxx';
Query OK, 0 rows affected (0.00 sec)
```

*: More information on public key authentication: http://www.debian-administration.org/articles/530

Command description

- (22) Installs the MySQL server and other packages that are necessary
- (23) Configures MySQL
- (24) Logs in as root to mysql database
- (25) Creates the opennebula database

Synchronizing the time between the clients

- Synchronized clocks are important in the OpenNebula cloud, so it is a good idea to set up ntp clients on them.
- Execute the following commands and add the specified lines

```
oneadmin@controller:~$ sudo crontab -e
...
0 * * * * ntpdate pool.ntp.org
```

```
oneadmin@storage:~$ sudo crontab -e
...
0 * * * * ntpdate pool.ntp.org
```

*: More information on public key authentication: http://www.debian-administration.org/articles/530

Installing the tools required to compile OpenNebula

- Execute the following command

```
oneadmin@controller:~# sudo apt-get install build-essential ruby
   libxmlrpc-c3-dev scons libopenssl-ruby libssl-dev flex bison ruby-dev
   rake rubygems libxml-parser-ruby libxslt1-dev libnokogiri-ruby
   libsqlite3-dev
```

- After this step, we are ready to compile and install the OpenNebula toolkit. Before doing so, it is recommended to configure the ESXi nodes, so we will start with that

*: More information on public key authentication: http://www.debian-administration.org/articles/530

4.2.4 Installing the OpenNebula toolkit

Download and untar the toolkit source

- By now, everything should be configured and set up to install OpenNebula
  - if you are not sure, please consult the previous slides for guidance
- Download the toolkit
  - oneadmin@controller:/tmp$ wget http://dev.opennebula.org/attachments/download/395/opennebula-2.2.1.tar.gz
- Untar the contents of the tar.gz
  - oneadmin@controller:/tmp$ tar xvzf opennebula-2.2.1.tar.gz ; cd opennebula-2.2.1

Fix the bug with the ESXi host

- There is a bug in the current version of OpenNebula (2.2.1) which makes it impossible to use VMware ESXi hosts
  - Thanks to OpenNebula being open source, we are able to correct the bug. Note: It is possible that the bug will be fixed in future versions, so chances are, that your version is not affected. I would suggest you to check, just to make sure. If the script looks something like the one I have here, it has already been fixed
- In the file /tmp/opennebula-2.2.1/src/vmm/LibVirtDriverVMware.cc
  - modify the area that contains the driver name line to look similar to this:

```
if ( emulator != "vmware" ){
                file << "\t\t\t<driver name='";
        if ( !driver.empty() ){ file << driver << "'/>" << endl;  }
                else{  file << default_driver << "'/>" << endl;  }
        }
```

Source of this solution: http://redes-privadas-virtuales.blogspot.com/2011/06/opennebula-installation-on-ubuntu-ii.html

Compile and install OpenNebula

- This is the step we have all been waiting for - compilation of the tools
- Issue the following commands
  - oneadmin@controller:/tmp/opennebula-2.2.1$ scons mysql=yes parsers=yes
    - this one compiles OpenNebula
  - oneadmin@controller:/tmp/opennebula-2.2.1$ ./install.sh -d /srv/cloud/one
    - this is the command that actually installs OpenNebula

Set up the database

- The default database behind OpenNebula is SQLite
- We are planning on using MySQL, so we have to modify the configuration file

```
oneadmin@controller:/tmp/opennebula-2.2.1$  vim
   /srv/cloud/one/etc/oned.conf
...
# DB = [ backend = "sqlite" ]

DB = [ backend = "mysql",
   server  = "localhost",
   port    = 0,
   user    = "oneadmin",
   passwd  = "xxxxxx",
   db_name = "opennebula" ]
...
```

Start the OpenNebula daemon

- one start
  - starts opennebula
- one stop
  - stops opennebula

- It should not return any error messages. If it does, consult the previous slides

Auto starting the daemon (1)

- The one command is not issued automatically on boot. If you want the opennebula daemon to start automatically each time the machine is turned on, you have to create an rc file and place it in the init.d directory.
- Copy the contents of the next slides to /etc/init.d/opennebula

```
root@controller:~# vim /etc/init.d/opennebula
#!/bin/bash

### BEGIN INIT INFO
# Provides:          OpenNebula
# Required-Start:    $remote_fs $syslog $network
# Required-Stop:     $remote_fs $syslog $network
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start daemon at boot time
# Description:       Enable service provided by daemon.
### END INIT INFO

export ONE_LOCATION=/srv/cloud/one
export ONE_AUTH=$ONE_LOCATION/.one/one_auth
export ONE_XMLRPC=http://localhost:2633/RPC2
export PATH=$ONE_LOCATION/bin:$PATH
```

Auto starting the daemon (2)

```
RETVAL=0

start()
{
     su oneadmin -s /bin/bash -c '$ONE_LOCATION/bin/one start' ; RETVAL=$?
     return $RETVAL
}

stop()
{
     su oneadmin -s /bin/bash -c '$ONE_LOCATION/bin/one stop' ; RETVAL=$?
}

case "$1" in
     start)
             sleep 5
             start
             ;;
     stop)
             stop
             ;;
     restart)
             stop
             start
             ;;
     *)
             echo $"Usage: service opennebula {start stop restart}"
esac

exit $RETVAL
```

Auto starting the daemon (3)

- The rc script that has been just created needs to be set for different run levels
- This can be achieved by running the following commands

```
root@controller:~# chmod +x /etc/init.d/opennebula
root@controller:~# update-rc.d opennebula start 90 2 3 4 5 . stop 10 0 1
   6 .
```

- To test your installation and scripts, simply restart the machine, and check to see if the one command is running (ps ax | grep one)

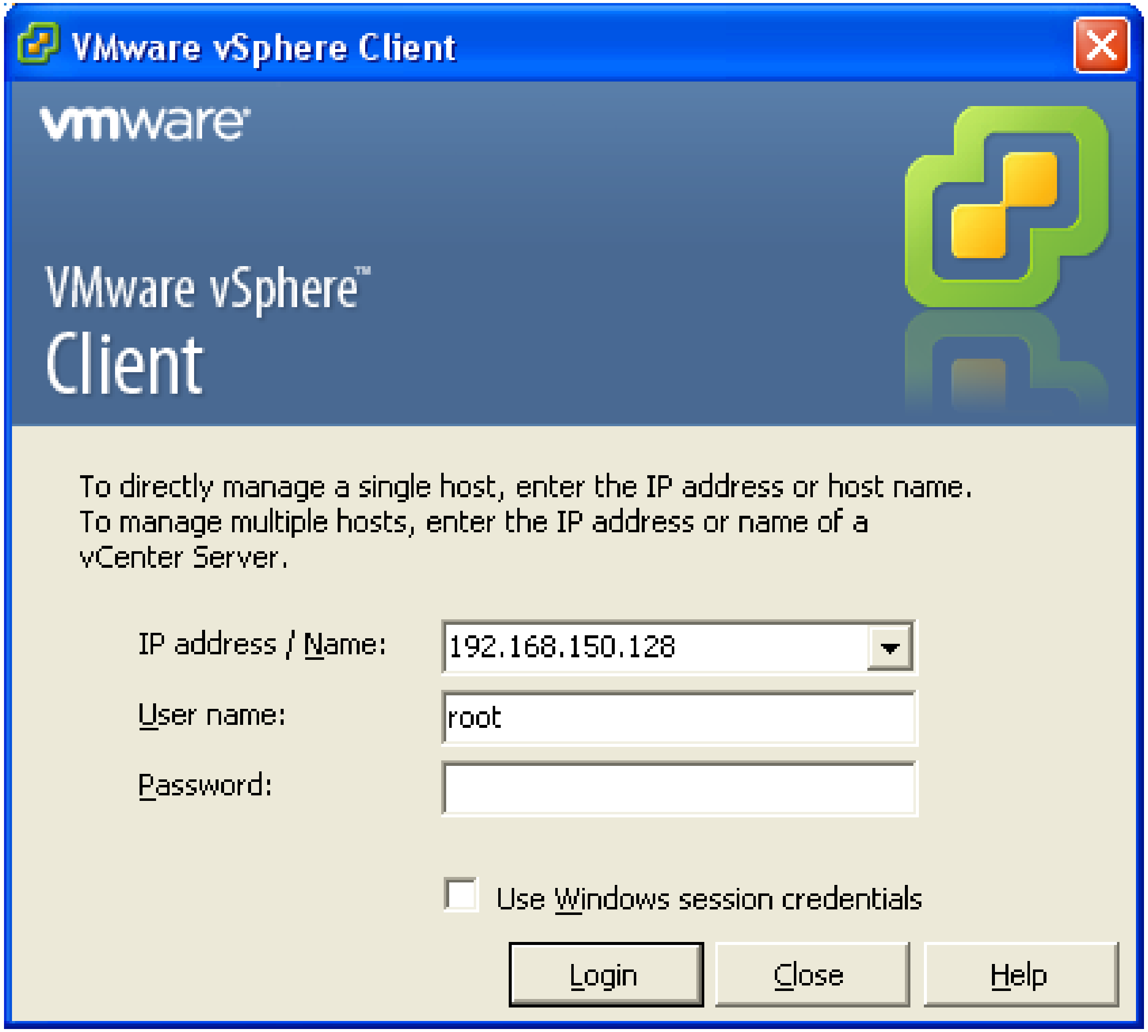
4.2.5 Configuring the ESXi nodes

Configuring the ESXi nodes (1) [10]

- In order to use OpenNebula with VMware ESXi nodes, the ESXi servers have to be configured properly
  - They can be set up using the vSphere client
- For a step-by-step tutorial on installing ESXi, please see the Appendix
- Configurating the ESXi servers
  - add the cloud group
  - add the oneadmin user
  - set the permission
  - set up the ntp time server
  - attach the storage

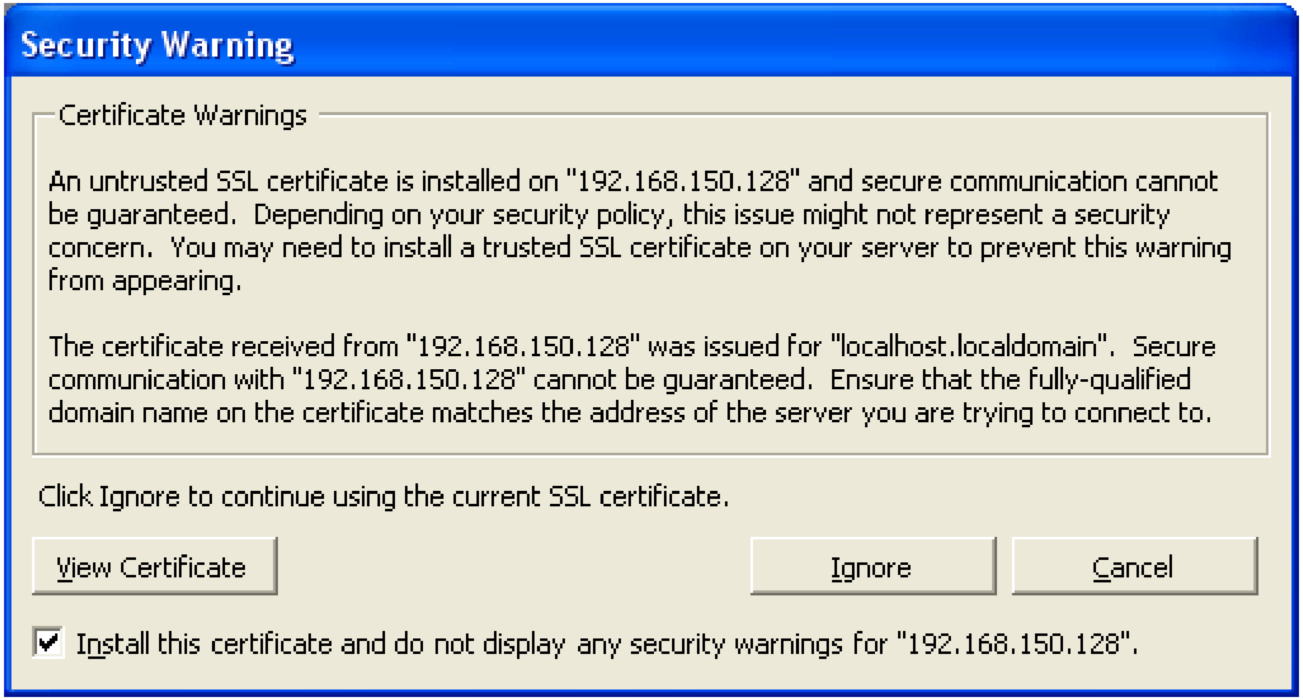
Configuring the ESXi nodes (2)

- Connect to the nodes using vSpehere Client - it has to be done with all the nodes



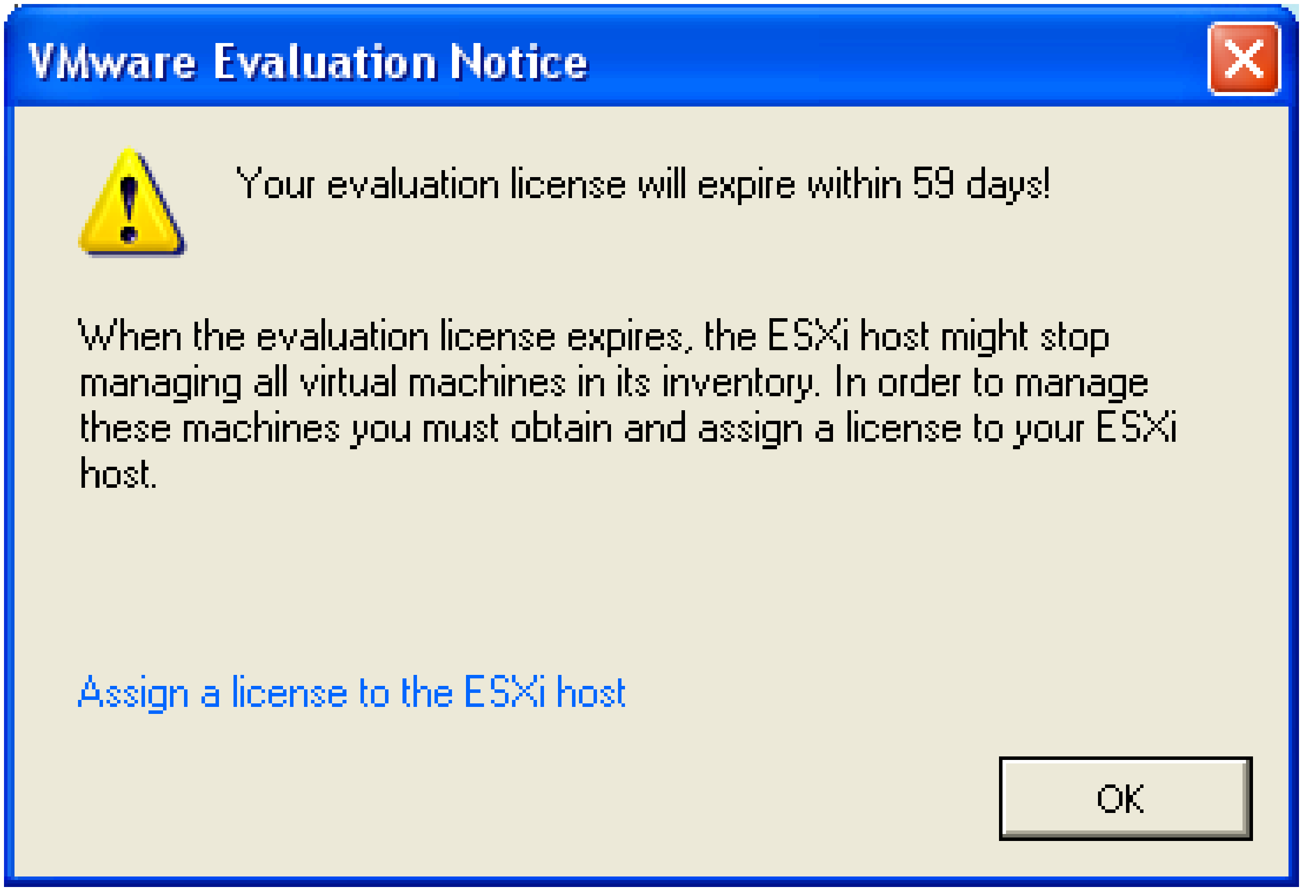
Configuring the ESXi nodes (3)

- Ignore the security warning



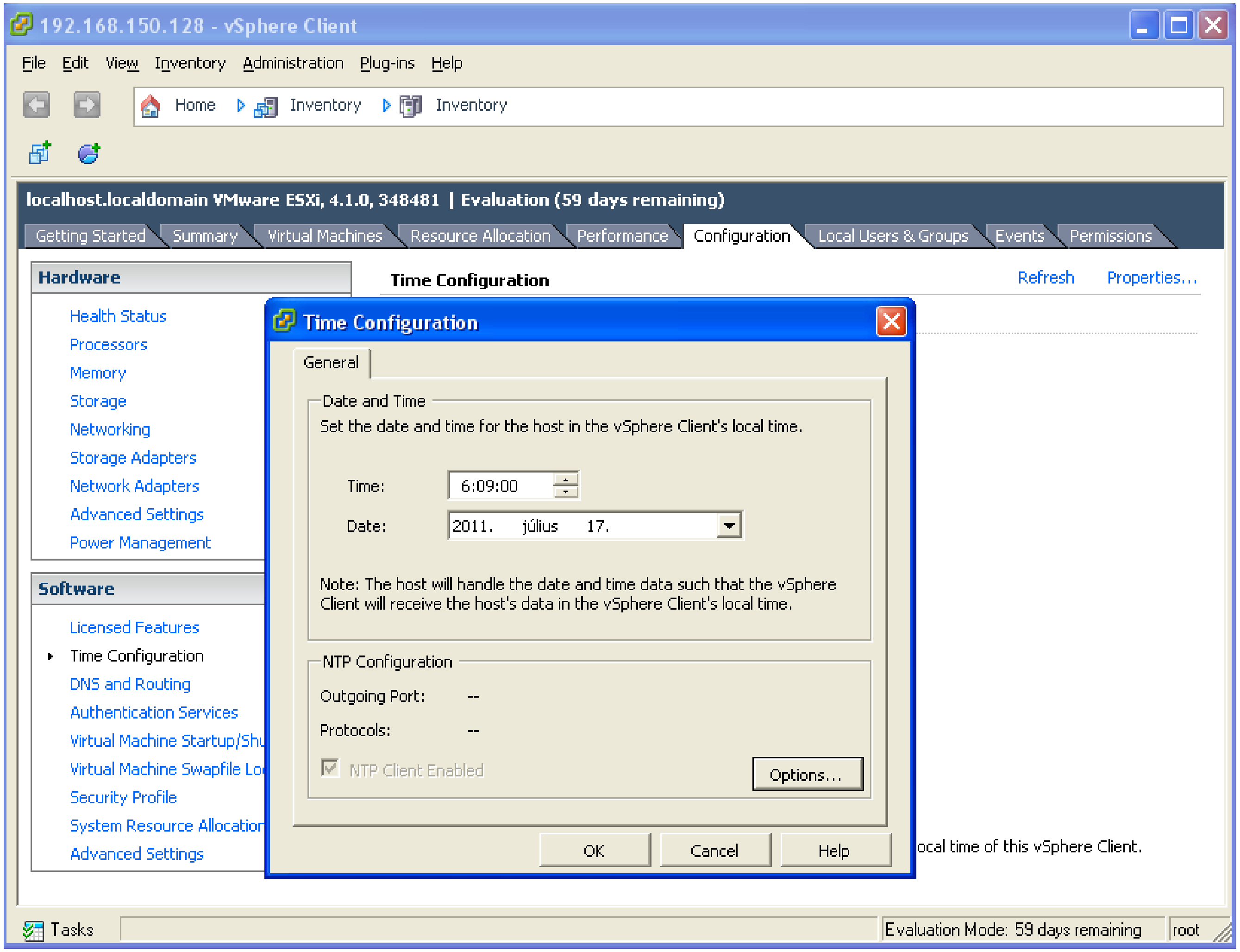
Configuring the ESXi nodes (4)

- Since we are using evaluation version for the ESXi hosts, we are greeted with this message. It is safe to ignore it



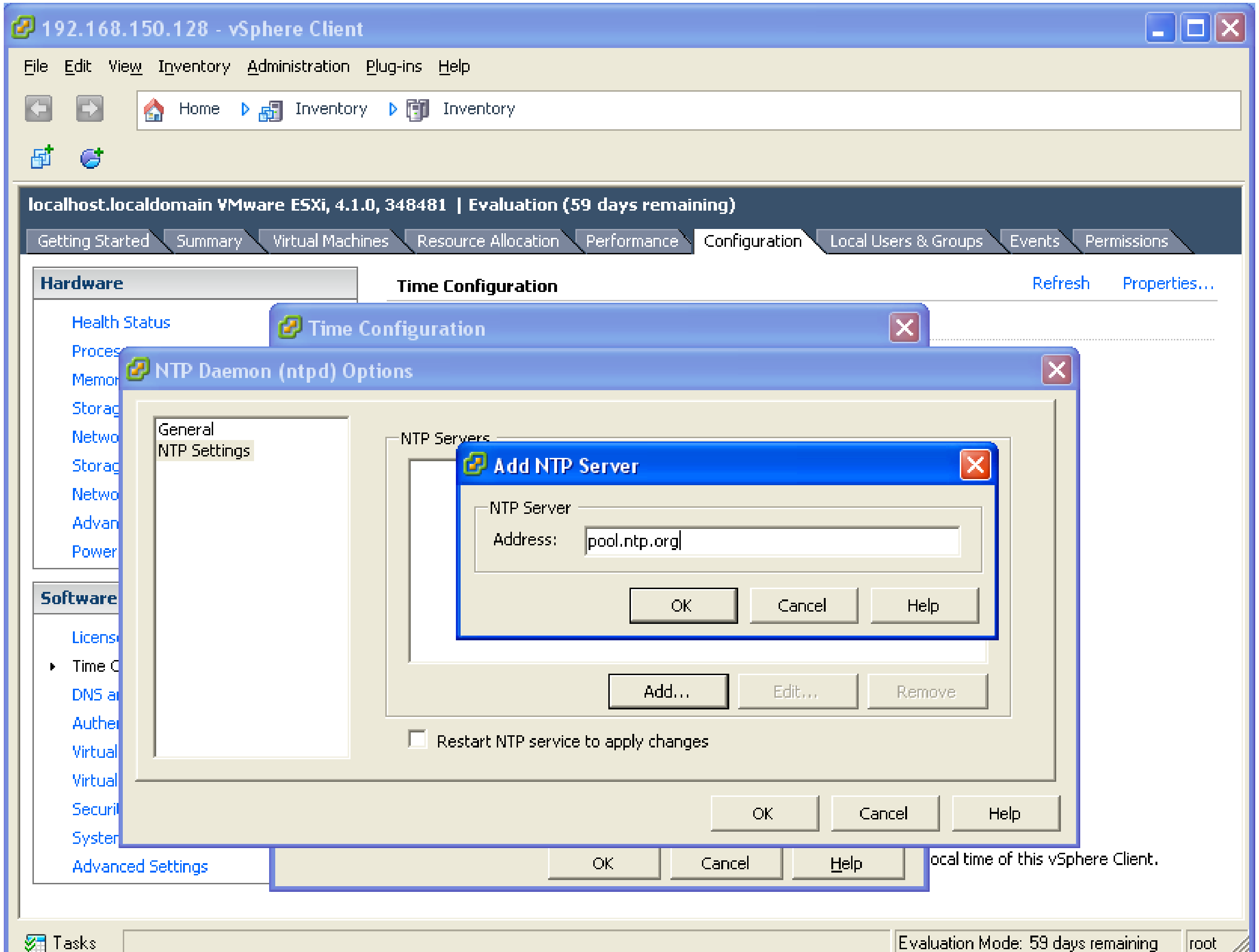
Configuring the ESXi nodes (5)

- Enable NTP - Configuration \ Time Configuration \ NTP Configuration \ Options



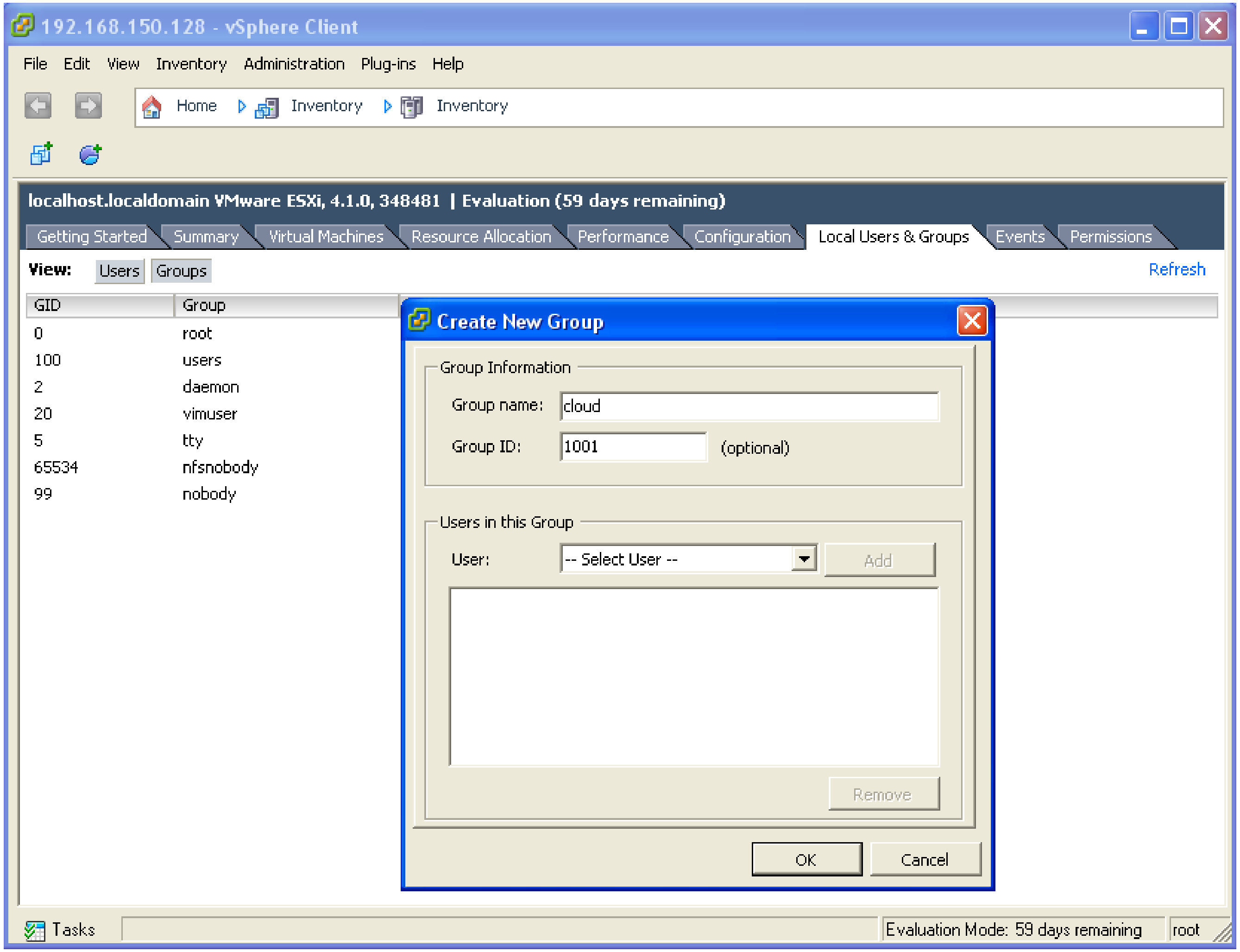
Configuring the ESXi nodes (6)

- Enable NTP: NTP Settings \ Add \ pool.ntp.org



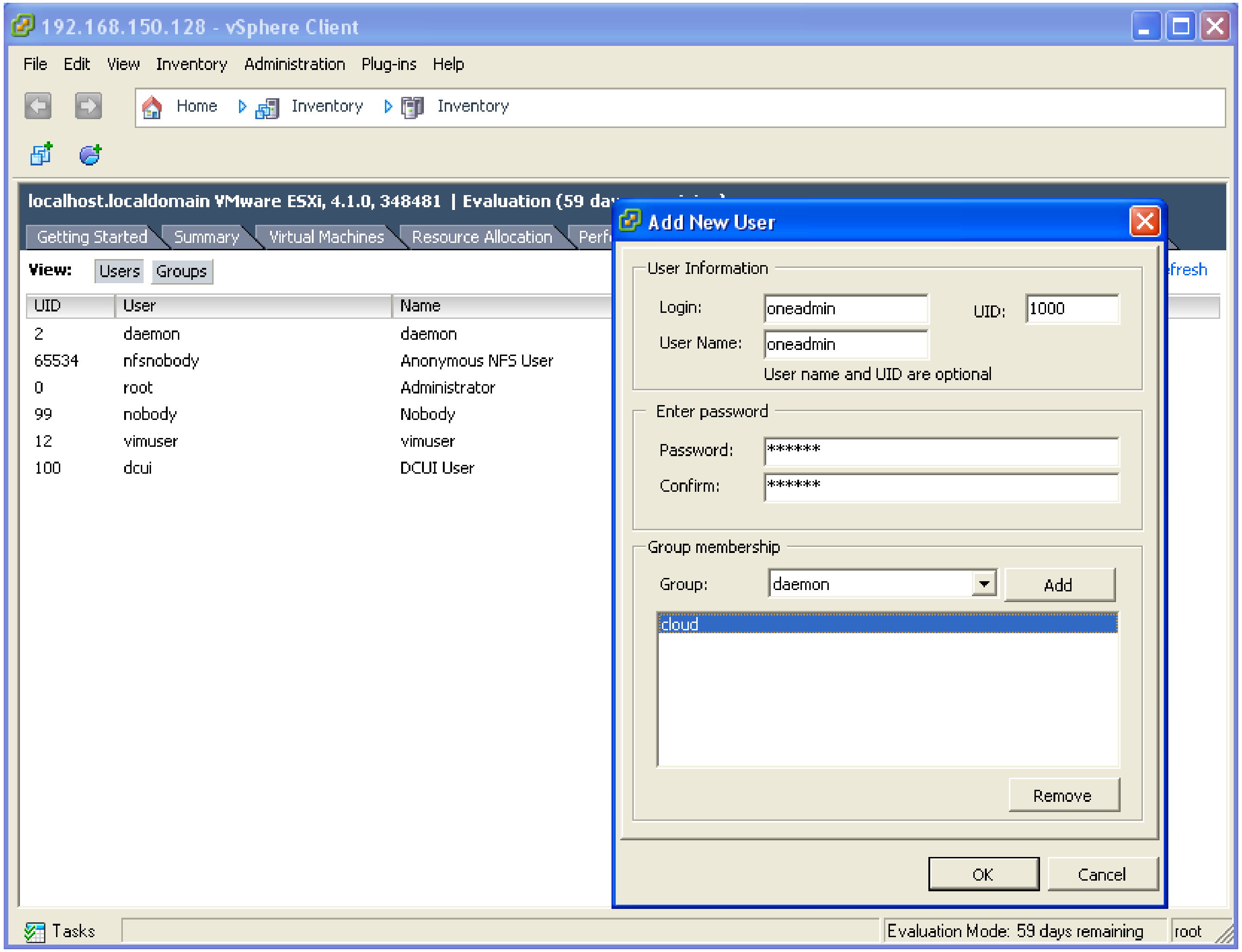
Configuring the ESXi nodes (7)

- Create the new group - Local Users and Groups tab, right click -> add new group. Group name: cloud, GID: 1001 (same as the GID on the controller)



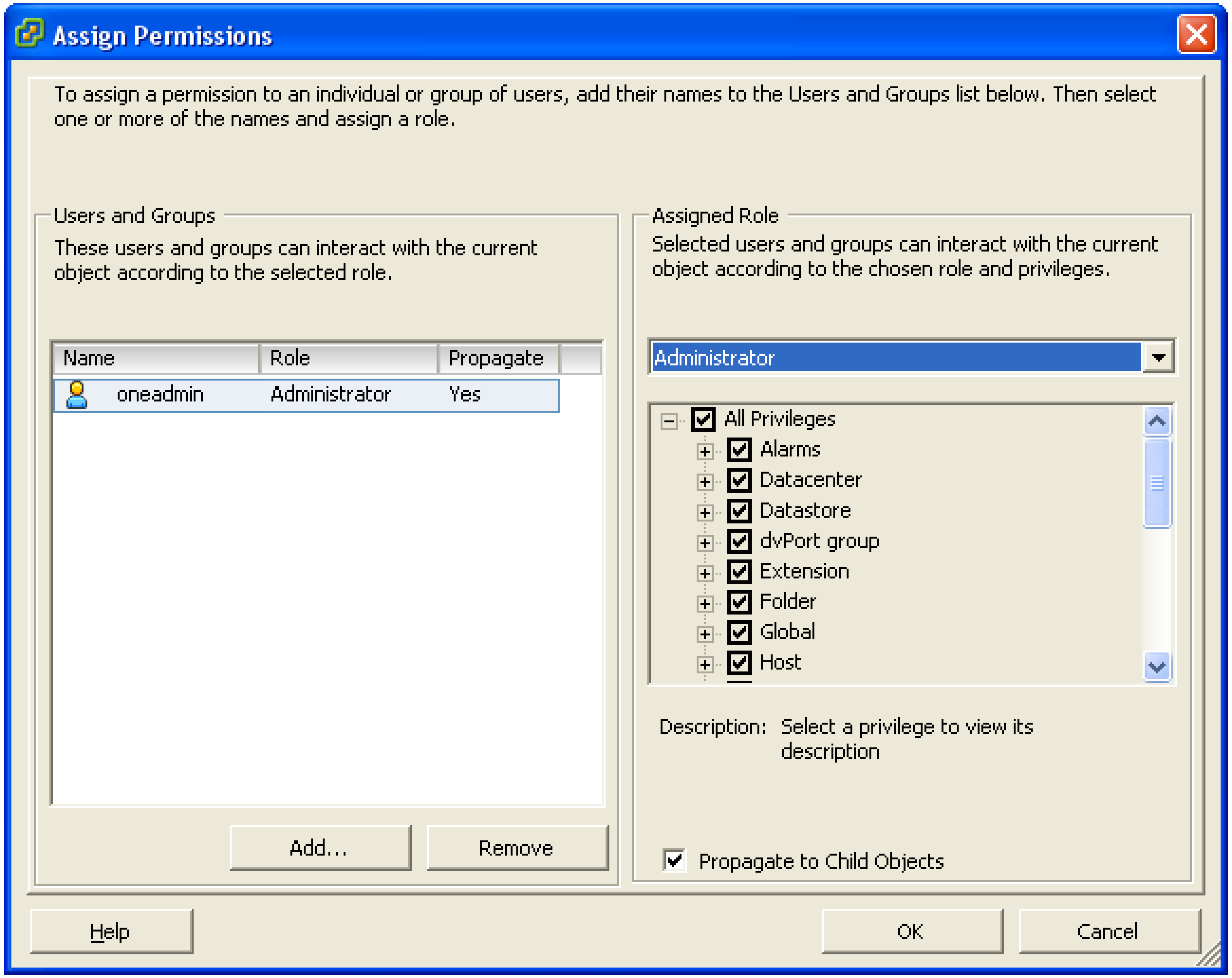
Configuring the ESXi nodes (8)

- Create the user oneadmin. Add to cloud group Passwords have to have more than 8 characters. UID has to be the same as on the controller



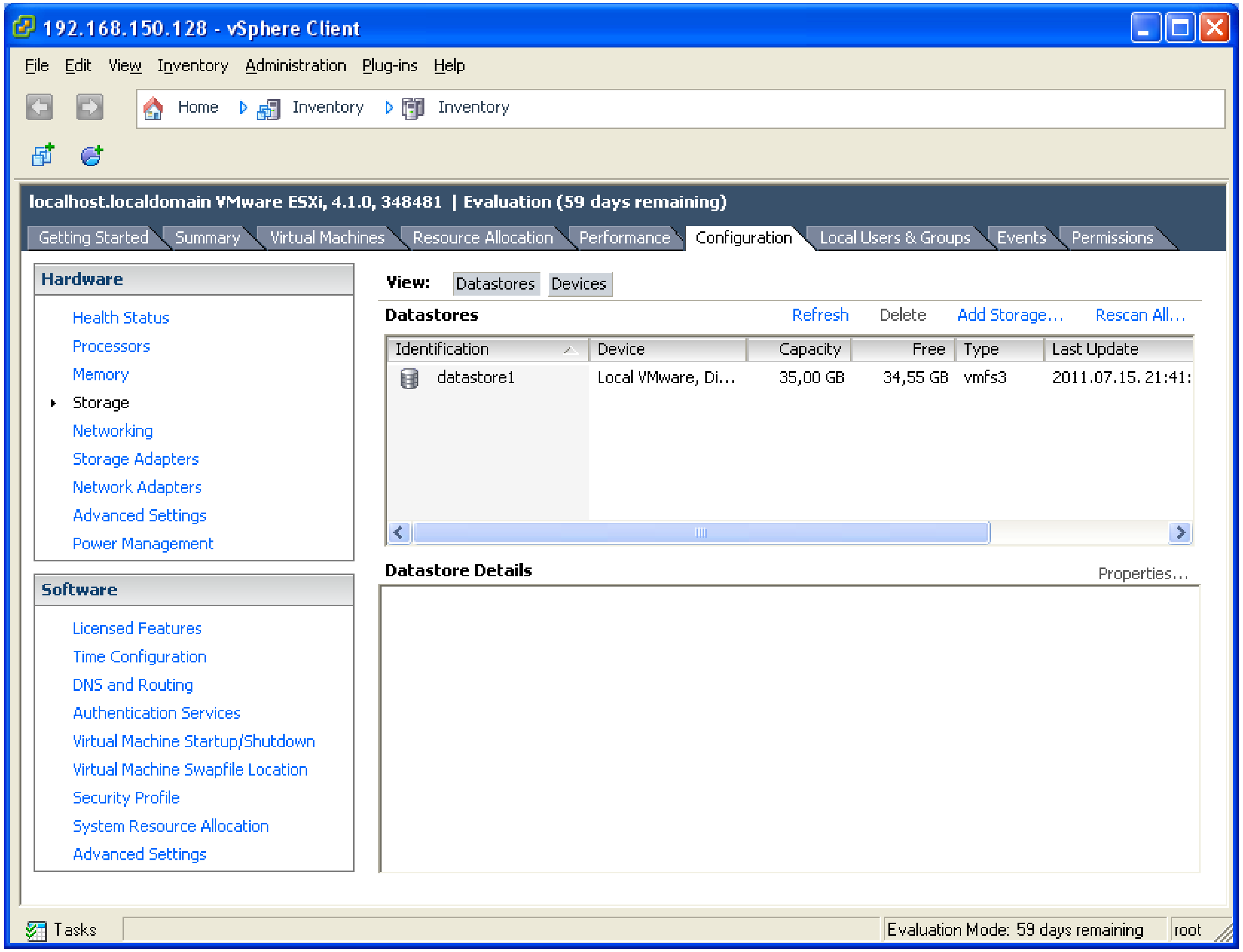
Configuring the ESXi nodes (9)

- Assign the permissions - Set administrator role for oneadmin



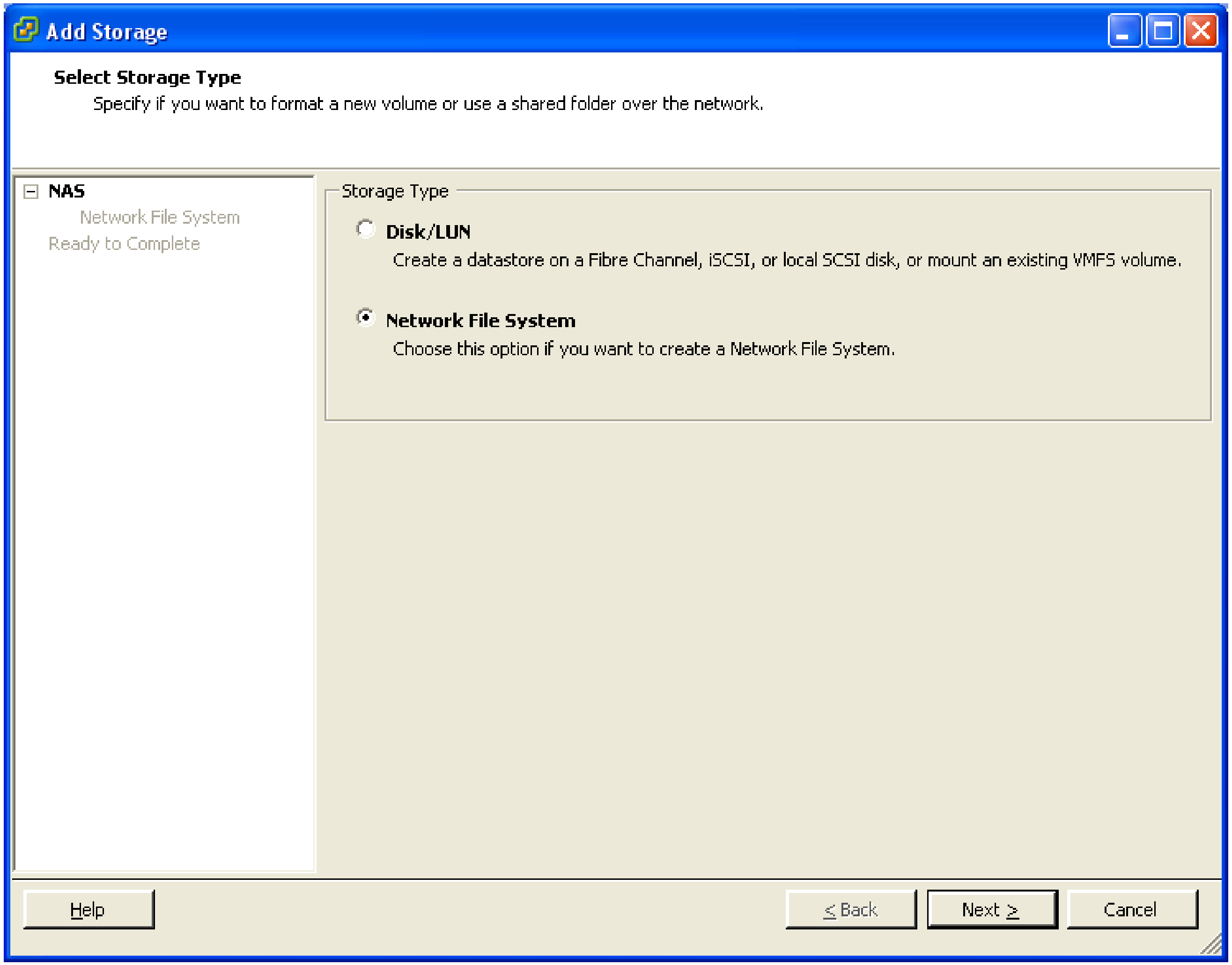
Configuring the ESXi nodes (10)

- Go to Configuration \ Storage \ Datastores \ Add Storage to configure the NFS access



Configuring the ESXi nodes (11)

- Select Network File System

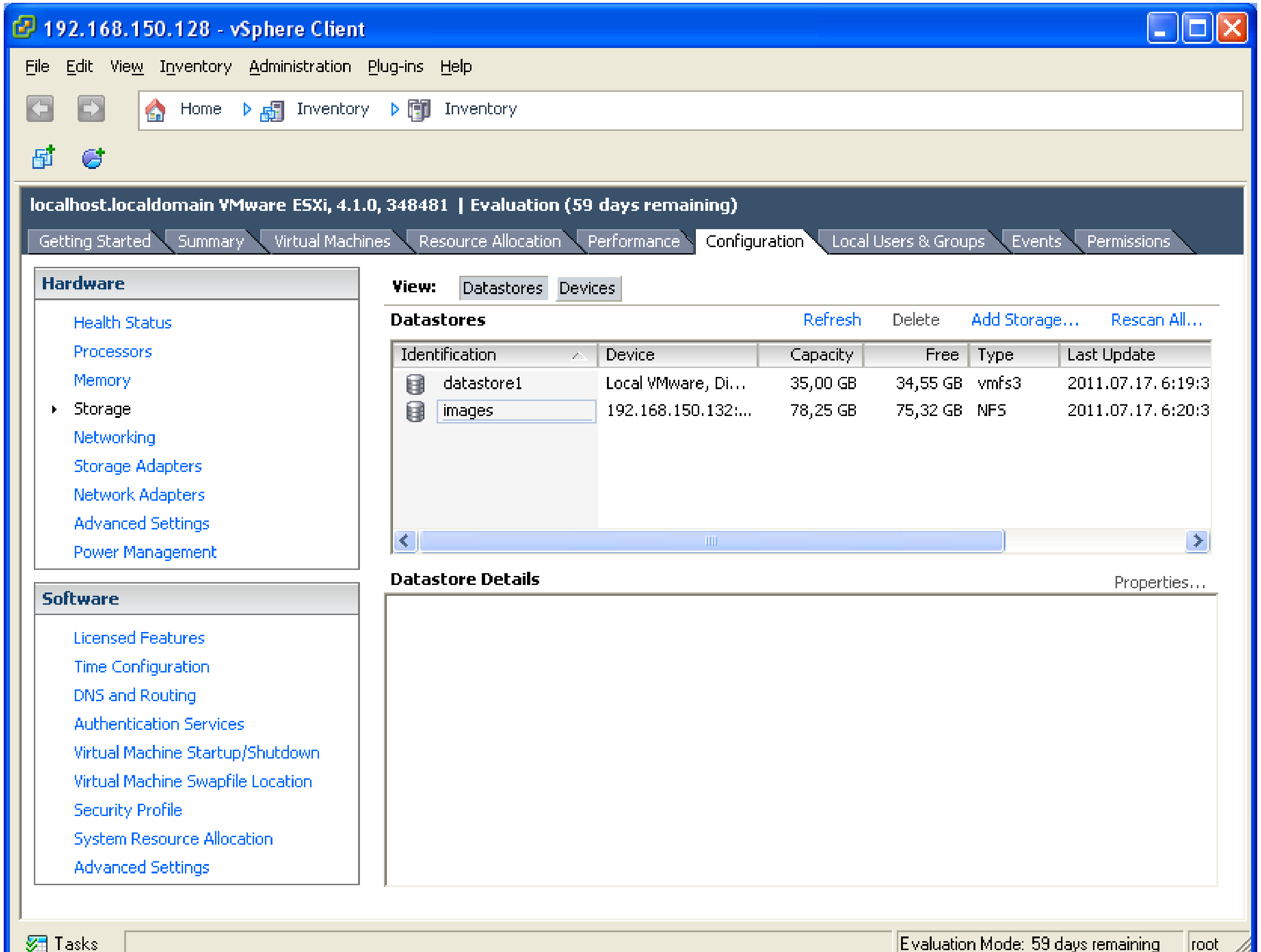


Configuring the ESXi nodes (12)

- Set the parameters: IP address of the storage server, username, password, path

Configuring the ESXi nodes (13)

- The new share should appear on the list



Configuring the ESXi nodes (14)

- Repeat the above procedure for all the nodes. Make sure you set everything up identically (addresses, usernames, passwords, IDs, everything must match)

- By now, you should have the controller, the storage and all the ESXi nodes installed and configured correctly. It is time to install the OpenNebula toolkit

Setting up the controller to work with ESXi hosts (1)

- To use OpenNebula, libvirt has to be installed. It is present in the Ubuntu repository, but it has to be installed from source, because we need the ESXi support which is not enabled by default

```
root@controller:~# apt-get install libgnutls-dev libdevmapper-dev
   libcurl4-gnutls-dev python-dev libnl-dev libapparmor-dev

root@controller:/tmp# wget http://libvirt.org/sources/libvirt-
   0.9.2.tar.gz

root@controller:/tmp# tar xvzf libvirt-0.9.2.tar.gz ; cd libvirt-0.9

root@controller:/tmp/libvirt-0.9# ./configure --with-esx --with-apparmor
   --sysconfdir=/etc --libdir=/usr/lib --sbindir=/usr/sbin --
   datarootdir=/usr/share --localstatedir=/var --
   libexecdir=/usr/lib/libvirt

root@controller:/tmp/libvirt-0.9# make ; make install
```

Setting up the controller to work with ESXi hosts (2)

- In the previous slide we have configured libvirt to have apparmor support, so we have to add libvirt to the apparmor configuration

```
root@controller:/tmp/libvirt-0.9# mkdir -p /etc/apparmor.d/libvirt
root@controller:/tmp/libvirt-0.9# cp -a examples/apparmor/usr.*
   /etc/apparmor.d/
root@controller:/tmp/libvirt-0.9# cp -a examples/apparmor/TEMPLATE
   /etc/apparmor.d/libvirt/
root@controller:/tmp/libvirt-0.9# cp -a examples/apparmor/libvirt-qemu
   /etc/apparmor.d/abstractions/

root@controller:/tmp/libvirt-0.9# vim /etc/apparmor.d/usr.sbin.libvirtd
   #Append the next line in front of the final }
   owner /srv/cloud/one/var/** rw,
   }

root@controller:/tmp/libvirt-0.9.2# /etc/init.d/apparmor restart
```

Install the VMware Driver Addon (1)

- VMware Driver Addon has to be downloaded and installed as well
- This is the module that enables communication between the controller and the ESXi hosts
- Execute the following commands as oneadmin user

```
oneadmin@controller:/tmp$ wget
   http://dev.opennebula.org/attachments/download/350/vmware-
   2.2.0.tar.gz

oneadmin@controller:/tmp$ tar xvzf vmware-2.2.0.tar.gz ; cd vmware-2.2.0

oneadmin@controller:/tmp/vmware-2.2.0$ ./install.sh
VMWare Drivers Addon successfully installed
```

- You should see the "VMWare Drivers Addon successfully installed" message appear on the screen

Install the VMware Driver Addon (2)

- After the driver addon is installed, you it will print a few lines that have to be copied into the opennebula configuration file (oned.conf)

```
#-------------------------------------------------------------------------------
#  VMware Driver Addon Virtualization Driver Manager Configuration
#-------------------------------------------------------------------------------
VM_MAD = [
name       = "vmm_vmware",
executable = "one_vmm_sh",
arguments  = "vmware",
default    = "vmm_sh/vmm_sh_vmware.conf",
type       = "vmware" ]

IM_MAD = [
name       = "im_vmware",
executable = "one_im_sh",
arguments  = "vmware" ]

TM_MAD = [
name       = "tm_vmware",
executable = "one_tm",
arguments  = "tm_vmware/tm_vmware.conf" ]
```

Install the VMware Driver Addon (3)

- install.sh does not copy all the files necessary, so it needs to be done manually
- Execute the following commands

```
oneadmin@controller:/tmp/vmware-2.2.0$ mkdir -p
   $ONE_LOCATION/var/remotes/im/vmware.d && cp -r im/remotes/*
   $ONE_LOCATION/var/remotes/im/vmware.d

oneadmin@controller:/tmp/vmware-2.2.0$ mkdir -p
   $ONE_LOCATION/var/remotes/vmm/vmware && cp -r vmm/remotes/*
   $ONE_LOCATION/var/remotes/vmm/vmware
```

Install the VMware Driver Addon (4)

- OpenNebula needs to be able to log in to the ESXi host, so it is essential that we set the password in the etc/vmwarerc configuration file
  - There are two lines that are important to us
    - USERNAME = "oneadmin"
    - PASSWORD = "xxxxxx"
  - fill these to be the same as the user that have been created on the ESXi hosts
- OpenNebula has to be able to create an OS hook - which needs root permissions. The easiest way to do this is to add the user to the sudoers file

  sudo visudo
  - add the following line
- `oneadmin ALL=NOPASSWD:/srv/cloud/one/share/hooks/fix_owner_perms.sh ""`

- Restart OpenNebula daemon
  - oneadmin@controller:~# sudo /etc/init.d/opennebula restart

5 Using the OpenNebula toolkit

5.1 Command line toolkit

Basic commands in the OpenNebula toolkit

- onevm
  - submit, control and monitor virtual machines
- oneuser
  - add, delete and monitor users
- onehost
  - add, delete and monitor hosts
- onevnet
  - add, delete and monitor virtual networks
- oneimage
  - add, delete and monitor images
- onecluster
  - add, delete and monitor clusters
- oneauth
  - manage authentication settings

onevm command

- Description (from the man page)

```
"This command enables the user to manage virtual machines in OpenNebula.
The user can allocate, deploy, migrate, suspend, resume and shutdown a
virtual machine with the functionality present in onevm."
```

- Main uses
  - create
  - list
  - deploy
  - migrate
  - stop
  - delete
- For more information, consult the man page
  - man onevm

oneuser command

- Description (from the man page)

"This command enables the OpenNebula administrator to manage users, adding, listing and deleting them."

- Main uses
  - create
  - list
  - delete
  - migrate
  - passwd
  - delete
- For more information, consult the man page
  - man oneuser

onehost command

- Description (from the man page)

```
"This command enables the user to manage hosts in the Open Nebula
server. It provides functionality to allocate, get information and
delete a particular host or to list all the available hosts."
```

- Main uses
  - create
  - show
  - list
  - enable
  - disable
  - stop
  - top
  - sync
- For more information, consult the man page
  - man onehost

onevnet command

- Description (from the man page)

```
"This command enables the user to manage virtual networks in the
OpenNebula server. It provides functionality to create, get information
and delete a particular network or to list available and used IP's."
```

- Main uses
  - create
  - show
  - list
  - publish
  - unpublish
  - delete
  - addleases
  - rmreleases
- For more information, consult the man page
  - man onevnet

oneimage command

- Description (from the man page)

```
"This command enables the user to manage images."
```

- Main uses
  - register
  - update
  - enable
  - disable
  - publish
  - unpublish
- For more information, consult the man page
  - man oneimage

onecluster command

- Description (from the man page)

```
"This command enables the OpenNebula administrator to manage clusters.
The administrator can create, delete, as well as add and  remove  hosts
from them. Any user can list available clusters."
```

- Main uses
  - create
  - delete
  - list
  - addhost
  - removehost
- For more information, consult the man page
  - man onecluster

oneauth command

- Description (from the man page)

```
"This command contains a set of utilities to manage authorization
module"
```

- Main uses
  - quota set
  - login
  - key
- For more information, consult the man page
  - man oneauth

5.2 OpenNebula Sunstone - web interface

OpenNebula Cloud Operations Center

- The primary interface of Opennebula is the toolkit with all the commands that are required for vm lifecycle management
  - Using the CLI is highly effective, yields instant results, the commands can be scripted, but it can be daunting to some users who prefer a graphical interface
- With version 2.2 of the toolkit, the users can also use OpenNebula Cloud Operations Center [11] (aka Sunstone)
  - Sunstone is capable of the same settings and management features that the CLI
  - Easier to use for beginners
  - Advanced users should stick to the command line as it is a lot more effective
    - Starting 10 virtual machines on the CLI takes as little as issuing the command onehost <name> start, on the GUI, each vm takes dozens of mouse clicks and scrolling in different menus

Installing the web interface

- The web interface is installed at the same time as the OpenNebula toolkit, so it is necessary that the required packages are installed beforehand
  - Three packages are needed
    - json - a json parsers library
    - sinatra - web framework
    - thin - web server
  - execute the following commands
    - sudo gem install json
    - sudo gem install sinatra
    - sudo gem install thin

Accessing the web interface (1)

- The OpenNebula Sunstone can be started with the command
  - $ONE_LOCATION/bin/sunstoner-server start
- Once the server is running, it will be listening to incoming connections on the URL
  - http://localhost:4567
  - You can specify the port at start time using the -p switch
- By default, the server can only be connected to from localhost. If you want to access it from a different host, you have to use the -H switch and provide the public IP address of that computer
  - For example, to start the server, have it listen on port #4411 and access connections from 192.144.224.211, the command to run is

```
$ONE_LOCATION/bin/sunstoner-server start -p 4411 -H 192.144.224.211
```

- To stop the web service, use the command

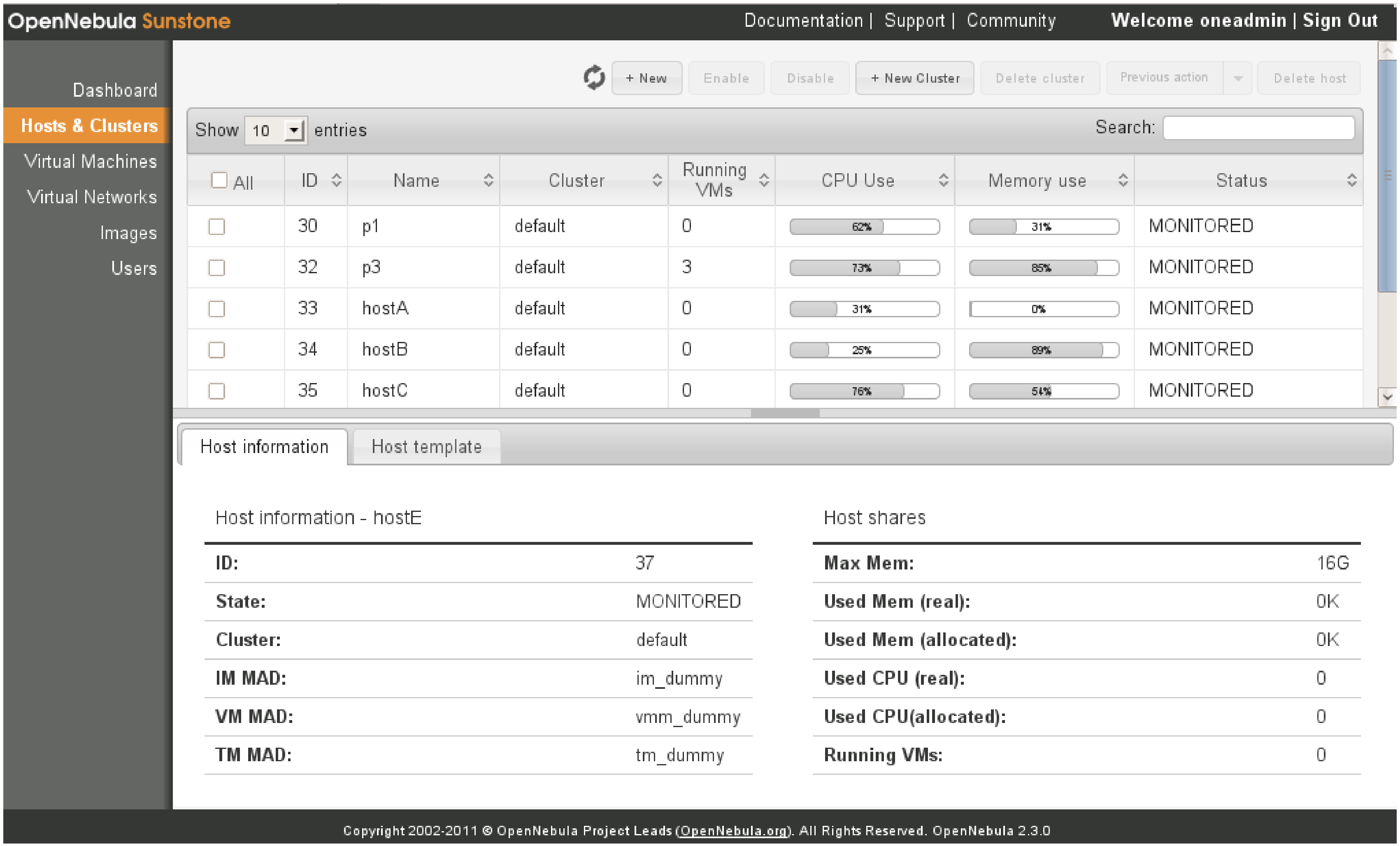
```
$ONE_LOCATION/bin/sunstoner-server stop
```

Accessing the web interface (2)

- OpenNebula Sunstone login screen - log in with oneadmin

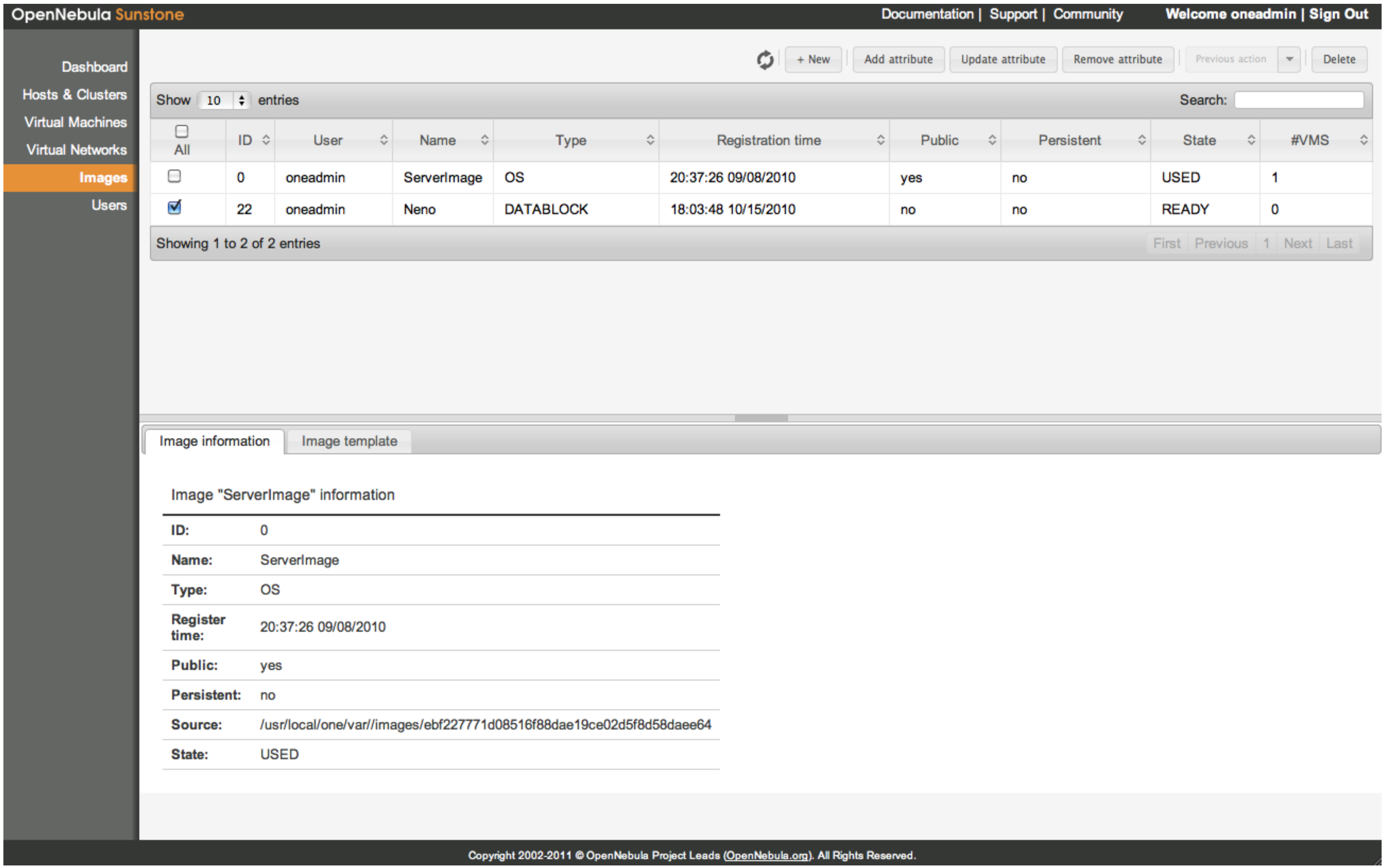
Managing hosts

- Click on Hosts and Clusters to manage / monitor the hosts and clusters in your cloud - CPU use, Memory use, running VMs etc.



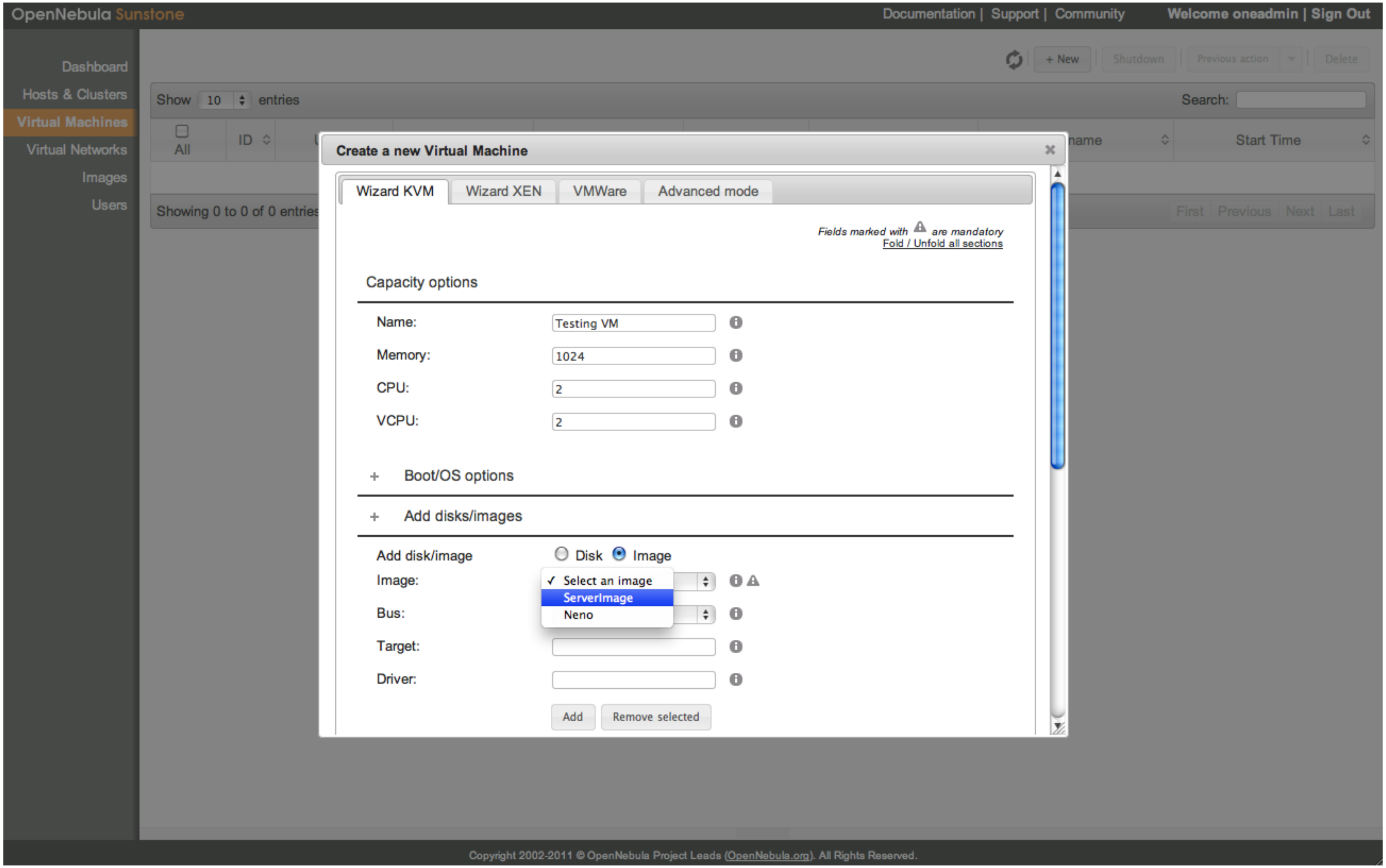
Managing images

- Images tab contains the settings concerning the images installed to your cloud



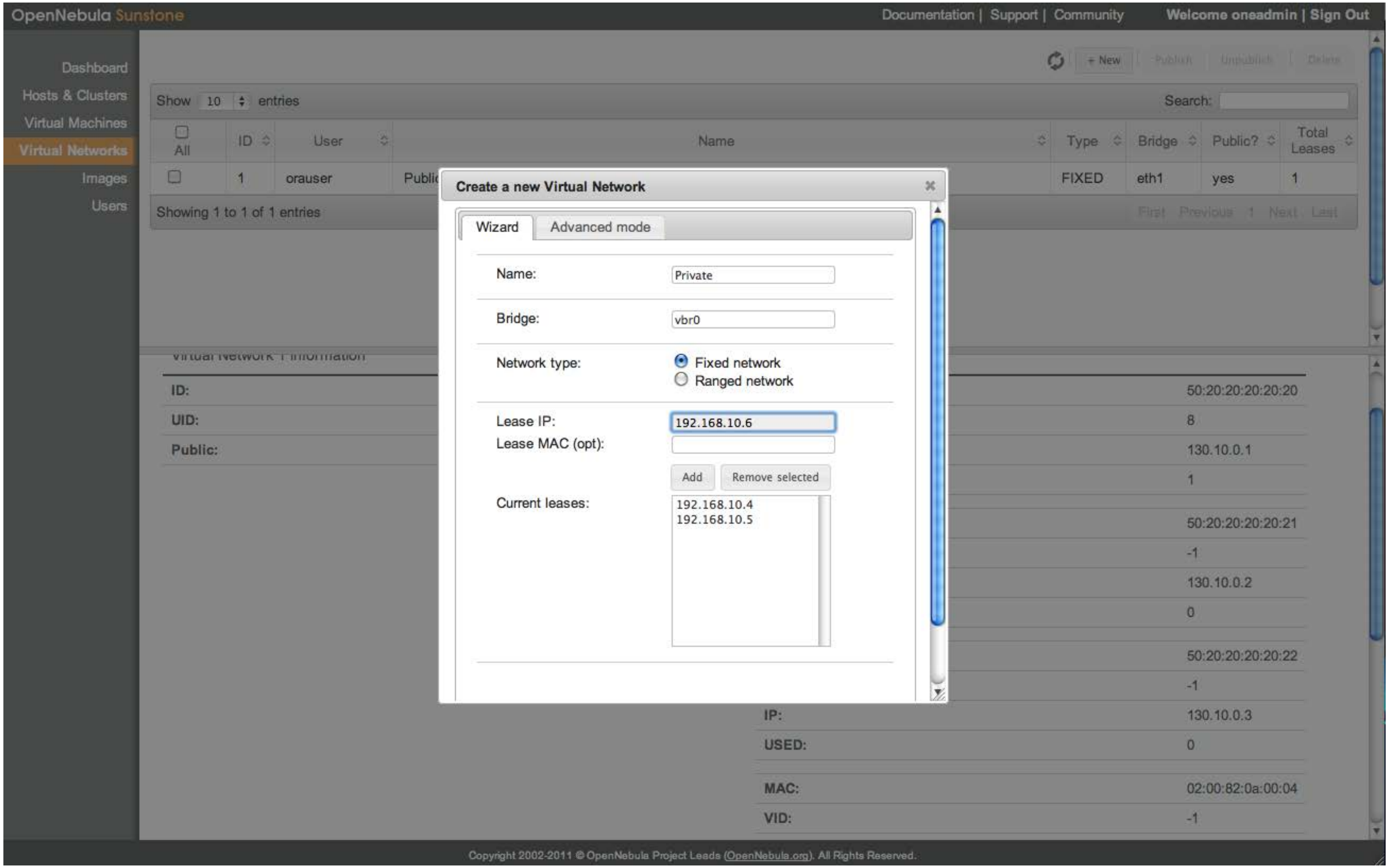
Managing virtual machines

- Use the Virtual Machines tab to contextualize new virtual machines. You have to select the type and hardware parameters, and click add



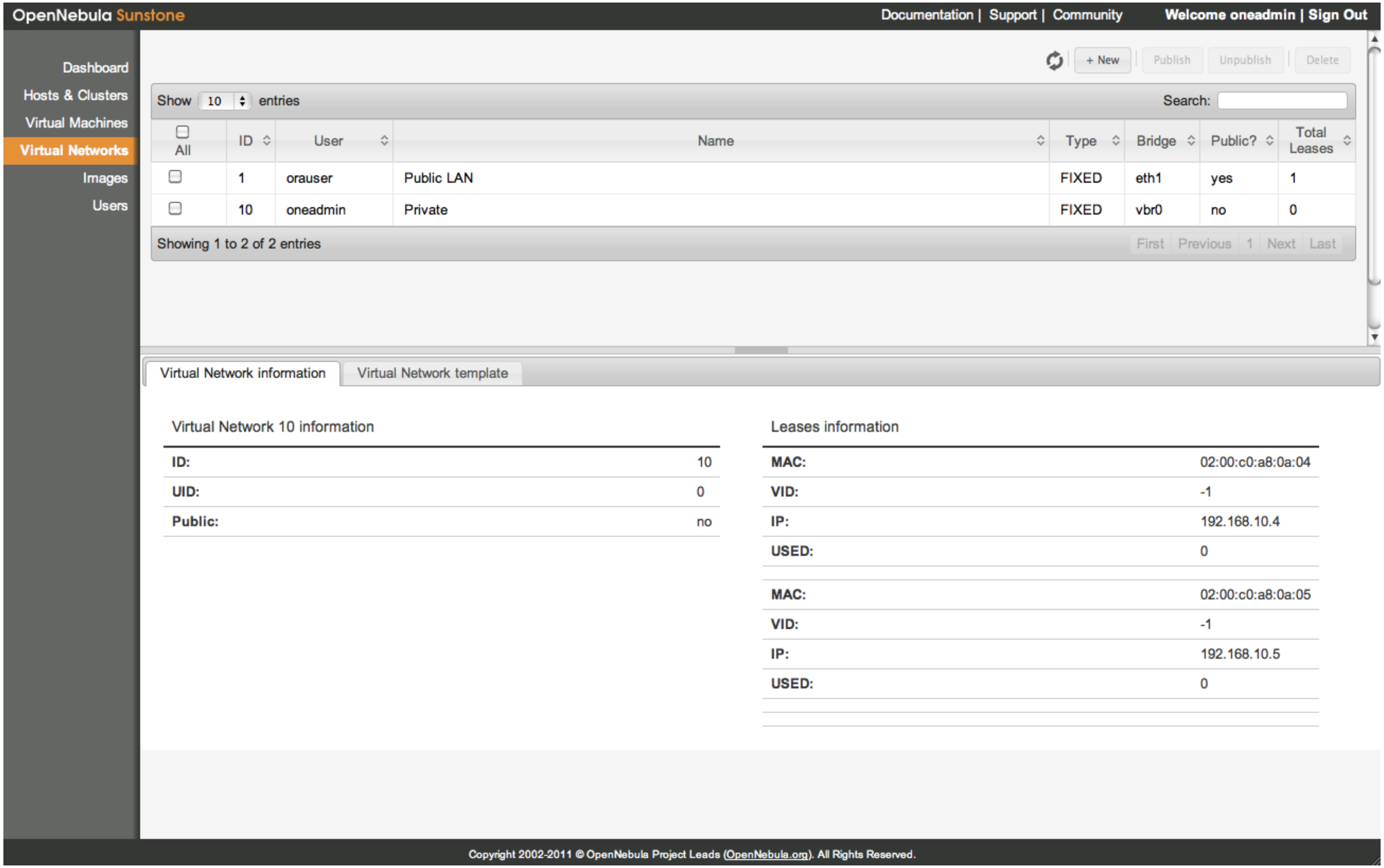
Managing virtual networks with Sunstone (1)

- Click on the network tab to manage your virtual networks. You can create, enable, disable, delete the virtual networks



Managing virtual networks with Sunstone (2)

- Click on the network tab to manage your virtual networks. You can create, enable, disable, delete the virtual networks



Exercise

- Take a few minutes to get to know the web interface

5.3 Managing users

Users in OpenNebula

- There are two types of users in an OpenNebula cloud
  - oneadmin
    - the account created the first time
    - it is the root account - oneadmin has enough privileges to control everything in the cloud
  - regular users
    - must be created by oneadmin
    - can only manage their own objects (images, virtual machines, networks)
    - can use public images of the others
- Each user needs to have the following variables set
  - ONE_AUTH
    - points to a file with a single line: username:password
  - ONE_LOCATION
    - set to the OpenNebula installation folder
  - ONE_XMLRPC
    - http://localhost:2633/RPC2
  - PATH
    - $ONE_LOCATION/bin:$PATH

Adding and removing users

- Creating users
  - oneuser create john password
    - john now has to edit his auth file

```
$ export ONE_AUTH="/home/john/.one/one_auth"
$ cat $ONE_AUTH
john:password
```

- List the current users
  - oneuser list

```
UID NAME           PASSWORD                                       ENABLE
  0 oneadmin       c24783ba96a35464632a624d9f829136edc0175e         True
  1 john           e727d1464ae12436e899a726da5b2f11d8381b26         True
```

- Delete users
  - oneuser delete john

5.4 Logging and debugging

Logging

- When something goes wrong, the logs should be consulted in case they have more details on the problem
- Each component logs to different places
  - ONE Daemon
    - core component, the logs are at $ONE_LOCATION/var/oned.log
  - Scheduler
    - $ONE_LOCATION/var/sched.log
  - Virtual machines
    - Different logs are under the $ONE_LOCATION/var/<VID>/ folder
      - vm.log
      - deployment.<execution>
      - transfer.<execution>.<operation>
  - Drivers
    - $ONE_LOCATION/var/name_of_driver

5.5 Managing hosts and clusters

Hosts

- Hosts are the main pillars of your cloud - they are the computers that actually run the virtual machines
- There can be multiple hosts in a cloud, and each one can use different hypervisor - OpenNebula is able to handle a heterogeneous setup
- When adding a host to the cloud, the following information is required
  - Hostname (or IP address)
  - Information driver - module to monitor the host e.g. im_kvm
  - Storage driver - module that handles all the storage functions (like copying, cloning, deleting images). e.g. tm_nfs
  - Virtualization driver - module that knows how to manage the VM lifecycle. e.g. vmm_kvm
- Before adding a host, make sure that you can ssh into it without a password

Clusters

- A cluster is a new concept that appeared first with t version 2.0 of OpenNebula. It is basically a group of hosts that can be managed together
- The grouping can be based on anything at all,
  - For example if we are a web developer company, we might have many servers. We use five for testing and development, and we use the other five for production websites.
    - In this case, it is a good idea to create two clusters, production and development, and assign each host to either one of these groups.
    - The clusters can be managed together - so we can stop all the hosts that belong to the development cluster

Managing hosts

- In this example, we create three hosts. They use kvm

```
$ onehost create host01 im_kvm vmm_kvm tm_nfs
$ onehost create host02 im_kvm vmm_kvm tm_nfs
$ onehost create host03 im_kvm vmm_kvm tm_nfs
```

```
$ onehost list
 ID NAME          CLUSTER        RVM  TCPU  FCPU  ACPU   TMEM    FMEM STAT
  0 host01        default          2   100    90    90 523264  205824   on
  1 host02        default          7   100    99    99 523264  301056   on
  2 host03        default          0   100    99    99 523264  264192  off
```

- Onehost list lists the hosts that are currently available
  - We can see their names, the clusters they are in, information about the available resources on each host, and the status (in this specific example, we can see that host03 is turned off

Managing hosts

```
$ onehost show host01
HOST 0 INFORMATION
ID                      : 0
NAME                    : host01
CLUSTER                 : default
STATE                   : MONITORED
IM_MAD                  : im_kvm
VM_MAD                  : vmm_kvm
TM_MAD                  : tm_nfs

HOST SHARES
MAX MEM                 : 523264
USED MEM (REAL)         : 317440
USED MEM (ALLOCATED)    : 131072
MAX CPU                 : 100
USED CPU (REAL)         : 10
USED CPU (ALLOCATED)    : 20
RUNNING VMS             : 2

MONITORING INFORMATION
ARCH=i686
CPUSPEED=1995
FREECPU=90
FREEMEMORY=205824
HOSTNAME=host01
HYPERVISOR=xen
MODELNAME=Intel(R) Xeon(R) CPU L5335  @ 2.00GHz
NETRX=0
NETTX=0
TOTALCPU=100
TOTALMEMORY=523264
USEDCPU=10
USEDMEMORY=317440
```

The onehost show command shows detailed information about each host

Managing hosts

- When a host is not needed anymore, it can be disabled or deleted using the onehost command
- $ onehost disable host01
  - disabled hosts show up in the list as status=off
- $ onehost delete host01

Managing clusters (1)

- A cluster is a new feature in OpenNebula
- It is a group that can hold one or multiple hosts that can be managed together
- The clusters are managed using the onecluster command

```
$ onecluster list
  ID     NAME
   0  default
```

- The default cluster is created automatically
- It is a good idea to separate your hosts
  - it makes management easier
  - and also more secure (cannot accidentally switch off one host from the wrong group)

Managing clusters (2)

```
$ onecluster create testing
$ onecluster create production

$ onecluster addhost host1 production
$ onecluster addhost host2 testing
$ onecluster addhost host3 testing

$ onehost list
  ID NAME          CLUSTER       RVM    TCPU    FCPU    ACPU     TMEM     FMEM STAT
   0 host5        default       0       0       0     100       0        0    on
   1 host1        production    0       0       0     100       0        0    on
   2 host4        default       0       0       0     100       0        0    on
   3 host2        testing       0       0       0     100       0        0    on
   4 host3        testing       0       0       0     100       0        0    on
```

- From now on, we could set a requirement for the VMs to only accept a host on the testing cluster

```
REQUIREMENTS = "CLUSTER = \"testing\""
```

Managing clusters (3)

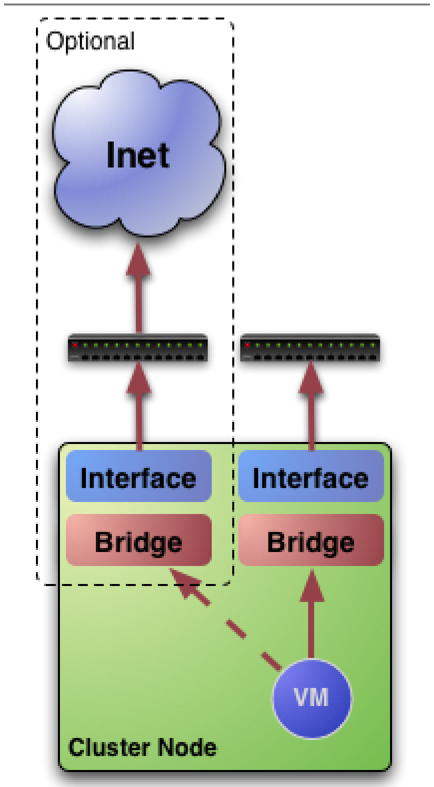
- You can delete the clusters when you don't need it anymore

```
$ onecluster delete testing

$ onehost list
  ID NAME          CLUSTER       RVM   TCPU   FCPU   ACPU    TMEM    FMEM STAT
   0 host5         default       0      0      0     100      0       0   on
   1 host1         production    0      0      0     100      0       0   on
   2 host4         default       0      0      0     100      0       0   on
   3 host2         default       0      0      0     100      0       0   on
   4 host3         default       0      0      0     100      0       0   on
```

5.6 Virtual networks

Managing virtual networks [13]

- The hosts are connected to one or more network interfaces that are accessible to the virtual machines as well, using the bridge interfaces that are set up on the host
- Virtual network types
  - Fixed
    - fix IP-MAC pair
  - Ranged
    - defines a range of IP addresses to be used
- Fixed virtual networks
  - A fixed set consists of IP - MAC address pairs
  - Information we need to define a fixed VN:
    - Name: name of the network
    - Public: yes or no (can others use it?)
    - Type: Fixed
    - Bridge: name of the bridge interface in the host to be used
    - Leases: definition of the IP-MAC addresses
      - If only the IPs are provided, the MAC will be generated in a deterministic manner



Managing virtual networks (2)

- Fixed virtual network template example

```
NAME = "Public"
TYPE = FIXED

#We have to bind this network to ''virbr1'' for Internet Access
BRIDGE = vbr1

LEASES = [IP=130.10.0.1, MAC=50:20:20:20:20:20]
LEASES = [IP=130.10.0.2, MAC=50:20:20:20:20:21]
LEASES = [IP=130.10.0.3]
LEASES = [IP=130.10.0.4]

#Custom Attributes to be used in Context
GATEWAY = 130.10.0.1
DNS     = 130.10.0.1

LOAD_BALANCER = 130.10.0.4
```

Managing virtual networks (3)

- Ranged virtual network example
  - Information we need to define a fixed VN:
    - Name: name of the network
    - Public: yes or no (can others use it?)
    - Type: Ranged
    - Bridge: name of the bridge interface in the host to be used
    - Network_Address: base network address for generating IPs
    - Network_Size: C: Class of the network, which in turn controls the number of available hosts in a network. It can either be a number (number of ones in the bitmask), or B or C

Managing virtual networks (4)

- Ranged virtual network template example

```
NAME = "Red LAN"
TYPE = RANGED

# This vnet can be only used by the owner user
PUBLIC = NO

#Now we'll use the cluster private network (physical)
BRIDGE = vbr0

NETWORK_SIZE    = C
NETWORK_ADDRESS = 192.168.0.0
#Custom Attributes to be used in Context
GATEWAY = 192.168.0.1
DNS     = 192.168.0.1

LOAD_BALANCER = 192.168.0.3
```

Managing virtual networks (5)

- Virtual networks are managed with the onevnet command
- We have created two templates for the networks. Using that template, it is possible to add the network to the cloud

```
$ onevnet -v create public.net
$ onevnet -v create red.net
```

```
$ onevnet list
  ID USER     NAME                TYPE BRIDGE P #LEASES
   2 oneadmin Public             Fixed   vbr1 Y       0
   3 oneadmin Red LAN           Ranged   vbr0 N       0
```

- To delete a virtual network just issue the delete command

```
$onevnet delete 2
$onevnet delete 'Red LAN'
```

- To use a network, don't forget to set it in the VM template

```
NIC=[NETWORK="Public"]
NIC=[NETWORK="Red LAN"]
```

5.7 Creating your own image

Creating your own image

- To create your own image on the cloud, all you have to do is create the virtual machine, install the operating system on it, create the virtual machine Definition File, and upload it using oneimage

Virtual Machine Definition File (1)

- The Virtual Machine Definition File is a main configuration file for each cloud OS image
- This file controls every setting of the runnable instance, for example
  - name
  - hardware resources
    - CPU (amount of virtual cores and clock frequency), Memory
  - OS and boot options
    - path to kernel, path to boot loader, architecture etc.
  - storage
    - disks, cdroms, network shares
  - I/O devices
    - input, graphics
  - Context section
    - hostname, IP address, MAC address
  - Requirements
- Syntax is straightforward
  - OPTION=VALUE — single attribute
  - OPTION=[OP1=VALUE1,OP2=VALUE2] — vector attribute

Virtual Machine Definition File (2)

- We can only set parameters that actually exist
  - e.g in order to use a network connection in the file, it has to be created beforehand using the onevnet command
- The basic parameters are quite straightforward, so we'll only talk about the tricky ones
- Context section
  - The context section contains the parameters for each individual running virtual machine that has been instantiated using this image. (which is also called contextualization)
    - Parameters like IP address, hostname, mac address etc.
  - For each of these parameters, it is possible to set a value, like
    - hostname = sampleubuntu
    - ip = 192.168.0.1
    - What is the problem with this approach?
      - Right, all the machines will have the same ip and the same hostname. Which might just lead to a small problem.
  - Use variables to set parameters that can change

Virtual Machine Definition File (3)

- Context section
  - hardcoded parameters
    - HOSTNAME = "MAINHOST
  - variable parameters
    - IP_GEN = "10.0.0.$VMID"
  - This example assigns the VMID as the last digit of the IP address
    - as the VMID is guaranteed to be different for all the running instances, there will be no IP address collision

Virtual Machine Definition File (4)

- Placement section
  - When a new virtual machine is created, it is assigned to a host automatically (it is possible to specify the host on creation, but that is not recommended)
  - We can give hints and specify rules that will be evaluated when OpenNebula selects a host for a specific VM
  - There are two keywords that handle this
    - RANK
      - tells the system which parameter to check when selecting a host - sorting parameter
      - RANK = FREECPU
        - » the host with the highest available CPU count will be selected
    - REQUIREMENT
      - hard parameter - a host has to match this to be eligible candidate for the virtual machine

Virtual Machine Definition File (5)

- Placement section examples
  - REQUIREMENTS = "CPUSPEED > 1000"
  - RANK = FREECPU
    - VM can only go on machines where the CPUSPEED is higher than 1000, and it will choose the one with the most available CPU
  - REQUIREMENTS = "HYPERVISOR=\"vmware\""
    - VM can only go on hosts that run vmware as the hypervisor
- For more information on the templates and definition file, please refer to the OpenNebula documentation
  - http://opennebula.org/documentation:rel2.2:template

Virtual Machine Definition Sample File

```
#---------------------------------------
# VM definition example
#---------------------------------------

NAME = vm-example

CPU    = 1
MEMORY = 512

# --- kernel & boot device ---

OS = [
  kernel   = "/vmlinuz",
  initrd   = "/initrd.img",
  root     = "sda" ]

# --- 3 disks ---

DISK = [ IMAGE_ID  = 3 ]
DISK = [ IMAGE_ID  = 5 ]
DISK = [
  type     = swap,
  size     = 1024,
  readonly = "no" ]

# --- 1 NIC ---

NIC = [ NETWORK_ID = 6 ]

# --- Placement options ---

REQUIREMENTS = "CPUSPEED > 1000"
RANK         = FREECPU

# --- Contextualization ---

CONTEXT = [
  files     = "/service/init.sh /service/certificates /service/service.conf"  ]
```

Important considerations when installing your own OS

- Creating a virtual machine and installing the OS is easy
  - Download the iso install image
  - Click next a few times
  - You have your shiny new OS
- When creating an image for use in the cloud however, things are not that simple. There are some special circumstances that has to be considered
  - There is no local console
    - make sure to install something that enables remote access
      - openssh server
        - » don't forget passwordless authentication
      - vnc server
      - (in case of Windows) enable remote desktop
  - Do not hardcode any ID as there will be multiple instances with the same ID
  - Hardware parameters can change between boots
    - users can select the HW parameters (CPU number, memory size etc), so your system has to be able to handle those changes

Creating your own image (1)

- The first step would be to install a virtual machine that will be used as a "golden" image
  - this will be the parent of all the running instances
  - we will skip this step and download an ubuntu server virtual machine from the vmware marketplace
- Visit http://www.vmware.com/appliances/directory/ and download any image you want.
  - Ubuntu 10.04 LTS
  - http://www.vmware.com/appliances/directory/711773
- Unzip the image you've downloaded
  - Note the names of the files that are inside the compression
  - *.vmdk are the virtual disk files - you'll need the names of those

Creating your own image (2)

- The next step is to create an image definition template for this image
  - An image template is a standardized meta file that specifies the name of the image and other essential information
  - http://opennebula.org/documentation:rel2.2:img_template
  - example of the image definition template

```
NAME          = "Ubuntu Web Development"
PATH          = /home/one_user/images/ubuntu_desktop.img
PUBLIC        = YES
DESCRIPTION   = "Ubuntu 10.04 desktop for Web Development students.
Contains the pdf lessons and exercises as well as all the necessary
programming tools and testing frameworks."
```

Creating your own image (3)

- Write your own Image Template!
  - vim ubuntuserver.img

```
NAME        = "Ubuntu Server 10.04"
DESCRIPTION = "Ubuntu Server 10.04 LTS (32 bits)"
```

- When the image template is done, you have to register your image with OpenNebula

```
oneadmin@controller:/tmp/ubuntu-server-10.04-i386$ onevmware register --disk-vmdk
    ubuntu-server-10.04.1-i386.vmdk --disk-flat ubuntu-server-10.04.1-i386-
    s001.vmdk,ubuntu-server-10.04.1-i386-s002.vmdk,ubuntu-server-10.04.1-i386-
    s003.vmdk,ubuntu-server-10.04.1-i386-s004.vmdk,ubuntu-server-10.04.1-i386-
    s005.vmdk ubuntuserver.img
```

- Now you should have an image in your cloud that can be deployed

```
oneadmin@controller:~$ oneimage list
ID     USER                 NAME TYPE              REGTIME PUB PER STAT  #VMS
0 oneadmin   Ubuntu Server 8.04   OS   Jul 02, 2011 10:34  No  No  rdy     0
```

Setting up the virtual network

- The next step is to create the virtual network the VM can use to connect to the network
- For a detailed guide, please visit
  - http://opennebula.org/documentation:rel2.2:vgg
- For our example, we will create a simple ranged network template

```
oneadmin@controller:~$ mkdir templates ; cd templates

oneadmin@controller:~/templates$ vim esxi.net
    NAME            = "ESXi Network"
    TYPE            = RANGED
    PUBLIC          = NO
    BRIDGE          = "VM Network"
    NETWORK_ADDRESS = 192.168.1.160
    NETWORK_SIZE    = 16
    NETMASK         = 255.255.255.0
    GATEWAY         = 192.168.1.1
    DNS             = 194.30.0.1
```

NOTE! You have to use the IPs of your actual setup

Setting up the virtual network (2)

- Use onevnet create the create the virtual network you've just created the configuration file for

```
oneadmin@controller:~/templates$ onevnet create esxi.net
```

- Test your virtual network

```
oneadmin@controller:~/templates$ onevnet list
ID USER     NAME            TYPE BRIDGE P #LEASES
 0 oneadmin ESXi Network    Ranged VM Net N      0
```

Create your own VM instance template

- The next step is to create an instance template for your virtual machine
  - The instance template describes the properties of the virtual machine

```
oneadmin@controller:~/templates$ cat ubuntuserver01.vm
NAME   = "UbuntuServer-01"
CPU    = 1
MEMORY = 512

DISK   = [ IMAGE  = "Ubuntuserver",
           TARGET = hda ]

NIC    = [ NETWORK = "ESXi Network" ]
```

5.8 Deploying the virtual machines

Deploy your virtual machine

- When everything is set up, all you need is to contextualize your virtual machine

```
oneadmin@controller:~/templates$ onevm create ubuntuserver01.vm
```

```
oneadmin@controller:~/templates$ onevm list
ID      USER      NAME STAT CPU     MEM        HOSTNAME        TIME
 0 oneadmin UbuntuSe pend   0      0K                   00 00:00:07
```

- As we can see, the machine state is pend (pending), which means the VM has been created, it has to be deployed
- To deploy the virtual machine, issue the command onevm deploy <VMID>
  - If we check the list now, we would see that it is no longer in the pending state

```
oneadmin@controller:~/templates$ onevm deploy 0 0
```

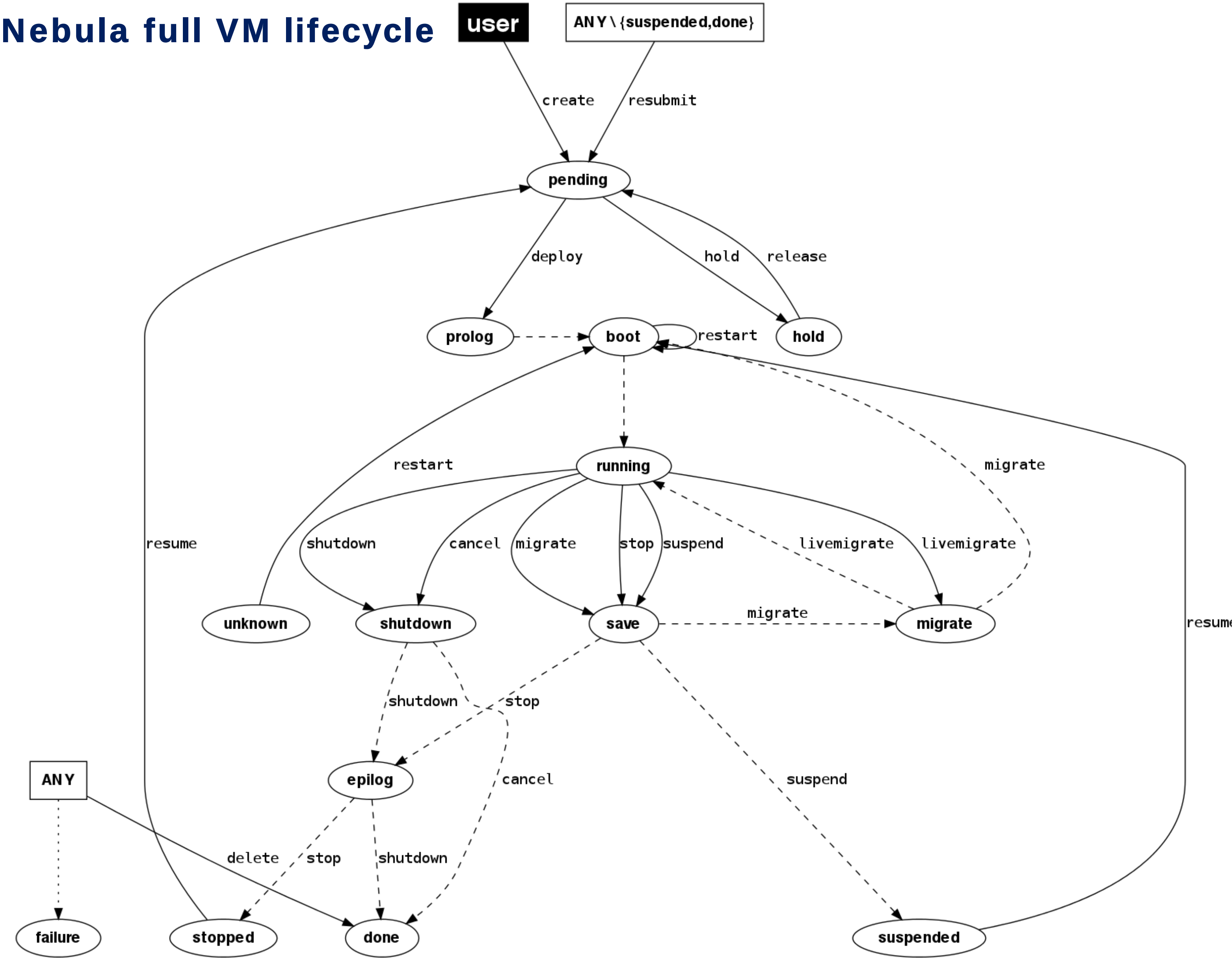
Virtual Machine Lifecycle

- When we created the virtual machine, it was created in the pending state. After deployment, the status changed
  - by running the onevm list command multiple times we can see how the state changes every time the machine enters into a new stage

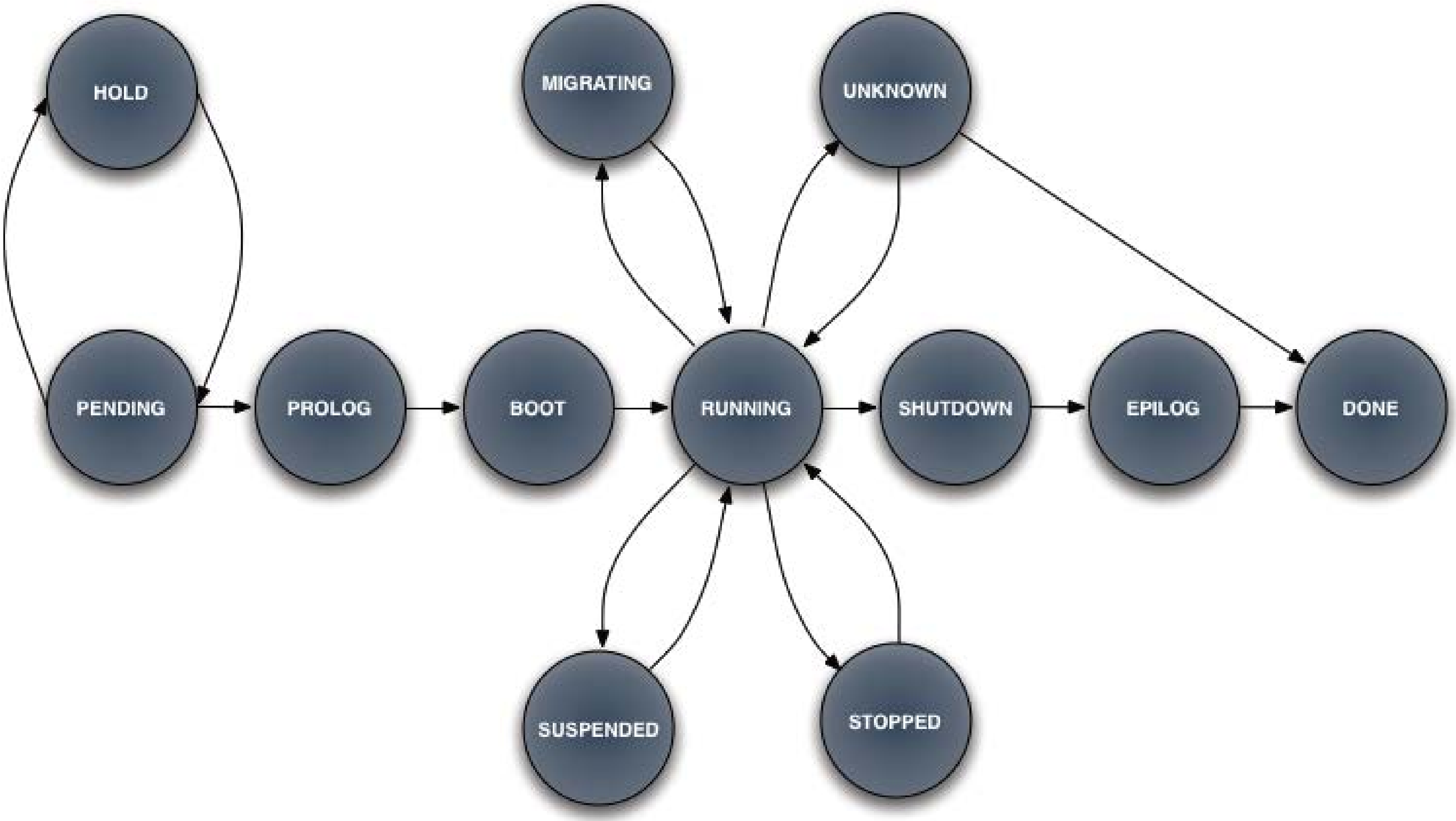
```
oneadmin@controller:~/templates$ onevm list
 0 oneadmin UbuntuSe prol    0      0K          esxi01 00 00:00:36
 0 oneadmin UbuntuSe boot    0      0K          esxi01 00 00:00:50
 0 oneadmin UbuntuSe runn    0      0K          esxi01 00 00:01:07
```

  - prol (prolog): the image is being copied from the storage to it's final location (could be on the host, could be just a different directory on the storage
  - boot: the virtual machine is booting
  - runn: the virtual machine is running (is ready to use)
- This ever changing state is called the lifecycle of the virtual machine
  - The lifecycle defines different states and ways to get from one state to the next

OpenNebula full VM lifecycle



OpenNebula simplified VM lifecycle



OpenNebula lifecycles (1)

- Pending (pend): By default a VM starts in the pending state, waiting for a resource to run on. It will stay in this state until the scheduler decides to deploy, or the user deploys it using the onevm deploy command
- Hold (hold): The owner has held the VM and it will not be scheduled until it is released. It can be, however, deployed manually
- Prolog (prol): The system is transferring the VM files (disk images and the recovery file)
- Running (runn): The VM is running (note that this stage includes the internal virtualized machine booting and shutting down phases). In this state, the virtualization driver will periodically monitor it
- Migrate (migr): The VM is migrating from one resource to another. This can be a life migration or cold migration (the VM is saved and VM files are transferred to the new resource)
- Epilog (epil): In this phase the system cleans up the cluster node used to host the VM, and additionally disk images to be saved are copied back to the cluster front-end

OpenNebula lifecycles (2)

- Stopped (stop): The VM is stopped. VM state has been saved and it has been transferred back along with the disk images
- Suspended (susp): Same as stopped, but the files are left in the remote resource to later restart the VM there (i.e. there is no need to re-schedule the VM)
- Failed (fail): The VM failed
- Unknown (unknown): The VM couldn't be reached, it is in an unknown state
- Done (done): The VM is done. VMs in this state won't be shown with “onevm list” but are kept in the database for accounting purposes

Migration

- The cloud is a very elastic infrastructure. It is easy to move a virtual machine from one host to the other (given of course that we have multiple hosts)
  - The method is called migration
- Two types of migration
  - off line migration
    - the machine is shut down, the important data is transferred from one host to the other, and the machine is turned on again
  - live migration
    - the machine is not turned off, it keeps on running while it is transferred to the other host
    - the memory contents are copied, then the changes made during the copy are copied again, and again, and again, until there is nothing left to copy. The machine is paused, the CPU state is transferred, the machine is restarted on the other host. When done correctly, the users don't note anything
- Live migration requires shared storage among the hosts
- Migration is handled by the onevm command

Exercise (1)

- By now the infrastructure is up and running, the rest of the class is for the exercises
- If you are not sure how to solve a problem, consult the man page for the given command, or the OpenNebula documentation
- Hint: check the effect of all the commands you issue using one* list

- Exercises
  - Add the second ESXi host to the cloud
  - Check the status of the cloud now
  - Deploy two instances of the ubuntuserver virtual machine
    - On which host do each of them run?
  - Destroy all your virtual machine instances
  - Create two clusters
  - Add one host to one cluster and the other to the other
  - Create new virtual machine image templates. Now you have three. Set one to require one cluster, set the other to require the other cluster.
  - Deploy three vm-s - one of each, and check if they went to the right place
  - Check what happens if you require a cluster that does not exist

Exercise (2)

- Remove one host from it's cluster
- Use the Web Interface to add the removed host to the other cluster (the one containing the other vm)
- Use the Web Interface to deploy a virtual machine instance
- Use the Web Interface to create a new VM template
- Use the Web Interface to configure a new virtual network
  - The settings are not important
- Use the Web Interface to register a new image
- Use the Web Interface to destroy all the running instances

Back to the CLI

- Deploy 5 instances of one VM
- Check on which host they are running
- Migrate one VM to a different host
- Live-migrate one VM to a different host
  - try to note when the migration happens - either by doing something on the migrated machine or pinging it
- See what happens when you remove a host from the cloud that has VMS running on it
  - what happens?

6 Final exam

Final exam

- The final exam for the hands-on lab consists of two parts
  - Theoretical questions on clouds, OpenNebula (infrastructure, installation, configuration)
  - Hands-on exercises: using the OpenNebula cloud

7 References

[1]: Velte A. T., Velte, T. J., Ph.D., Elsenpeter R., Cloud Computing: A Practical Approach, McGraw-Hill, 2010

[2]: Hurwitz J,. Bloor R., Kaufman M., Halper F.: Cloud Computing For Dummies, Wiley Publishing, Inc., 2010

[3]: Amazon Elastic Cloud Computing, http://www.aws.amazon.com

[4]: Rackspace Hosting, http://www.rackspace.co.uk

[5]: Canonical inc - http://www.canonical.com/enterprise-services

[6]: StratusLab - OpenSource IaaS Cloud, http://stratuslab.org

[7]: 4CaaS Project - http://4caast.morfeo-project.org

[8]: Rackspace Cloud - http://www.rackspace.com

[9]: OpenNebula Project - http://www.opennebula.org

[10]: ESXi Installable and vCenter Server Setup Guide, VMware Inc, 2010 http://www.vmware.com/support

[11]: http://opennebula.org/documentation:rel2.2:sunstone

[12]: Redes Privadas Virtuales, Javier Andrés Alonso, http://redes-privadas-virtuales.blogspot.com/2011/06/cloud-computing-with opennebula.html

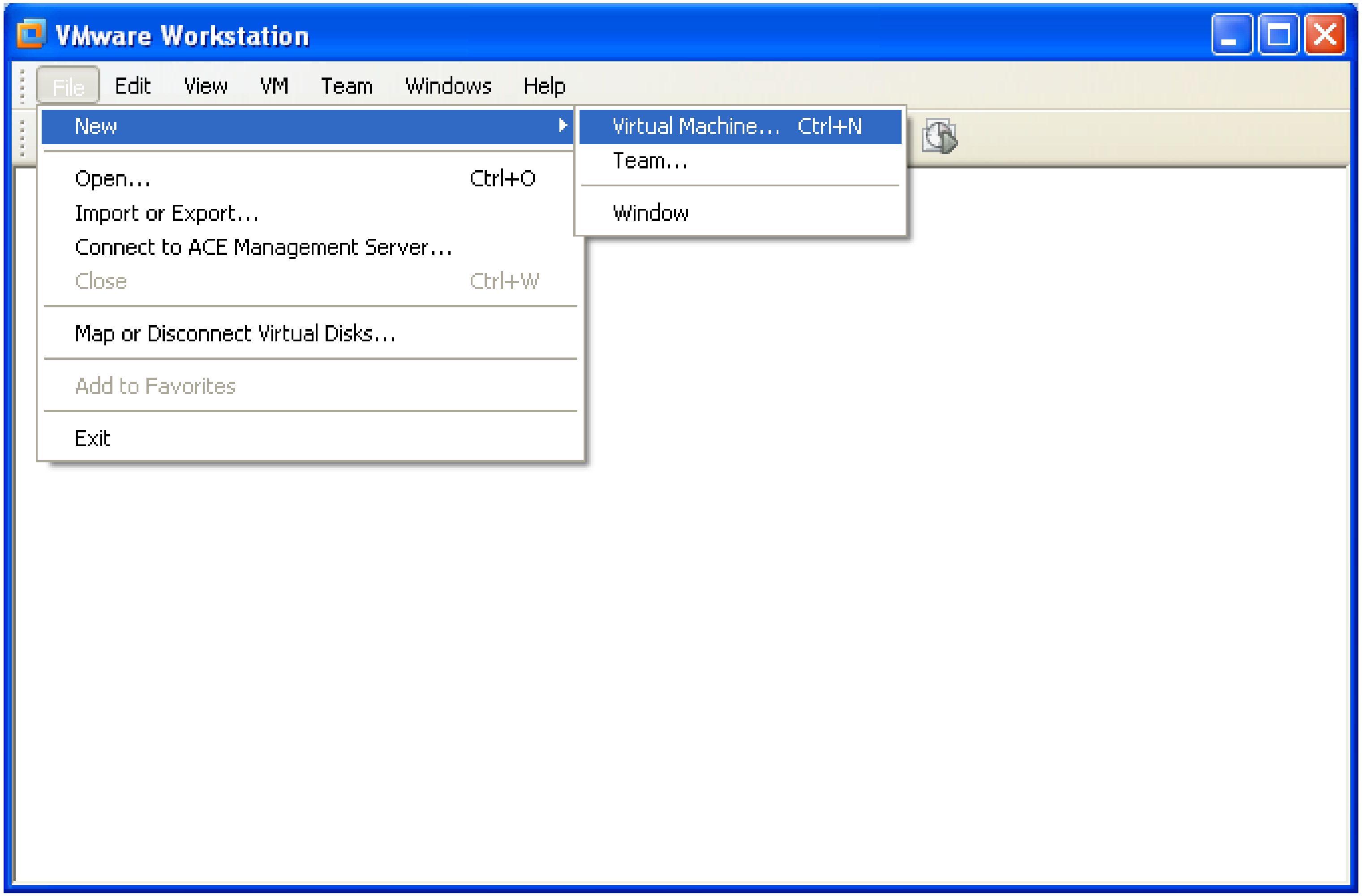
[13]: Virtual Networking Guide, OpenNebula, 2011 http://opennebula.org/documentation:rel2.2:vgg

[14]: http://www.cloudtweaks.com/2011/02/iaas-and-paas-to-disappear-by-2012

Appendix A - Creating virtual machines

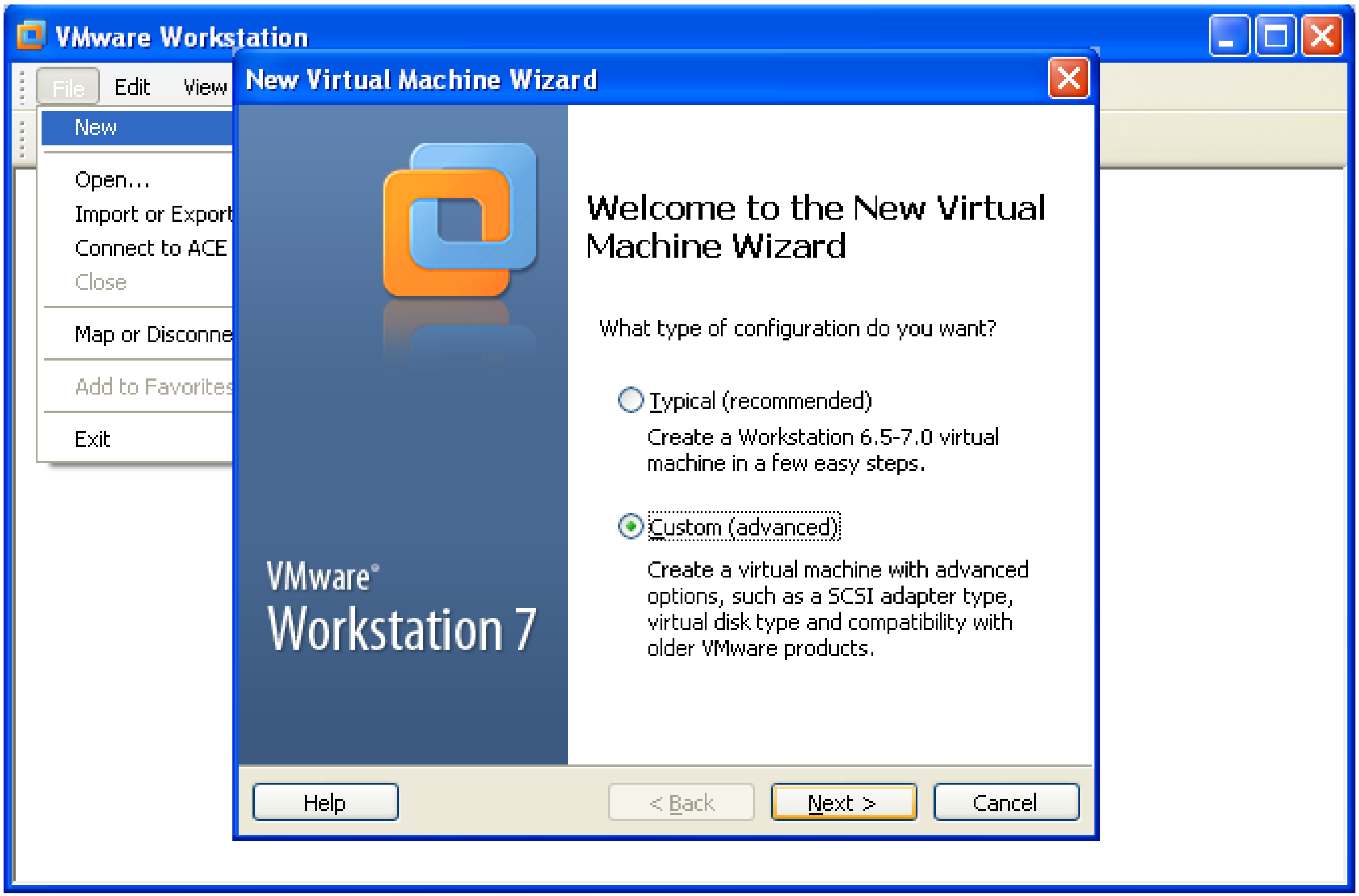
Preparation - Creating the virtual machines (1)

- Most of the steps are the same for all the machines



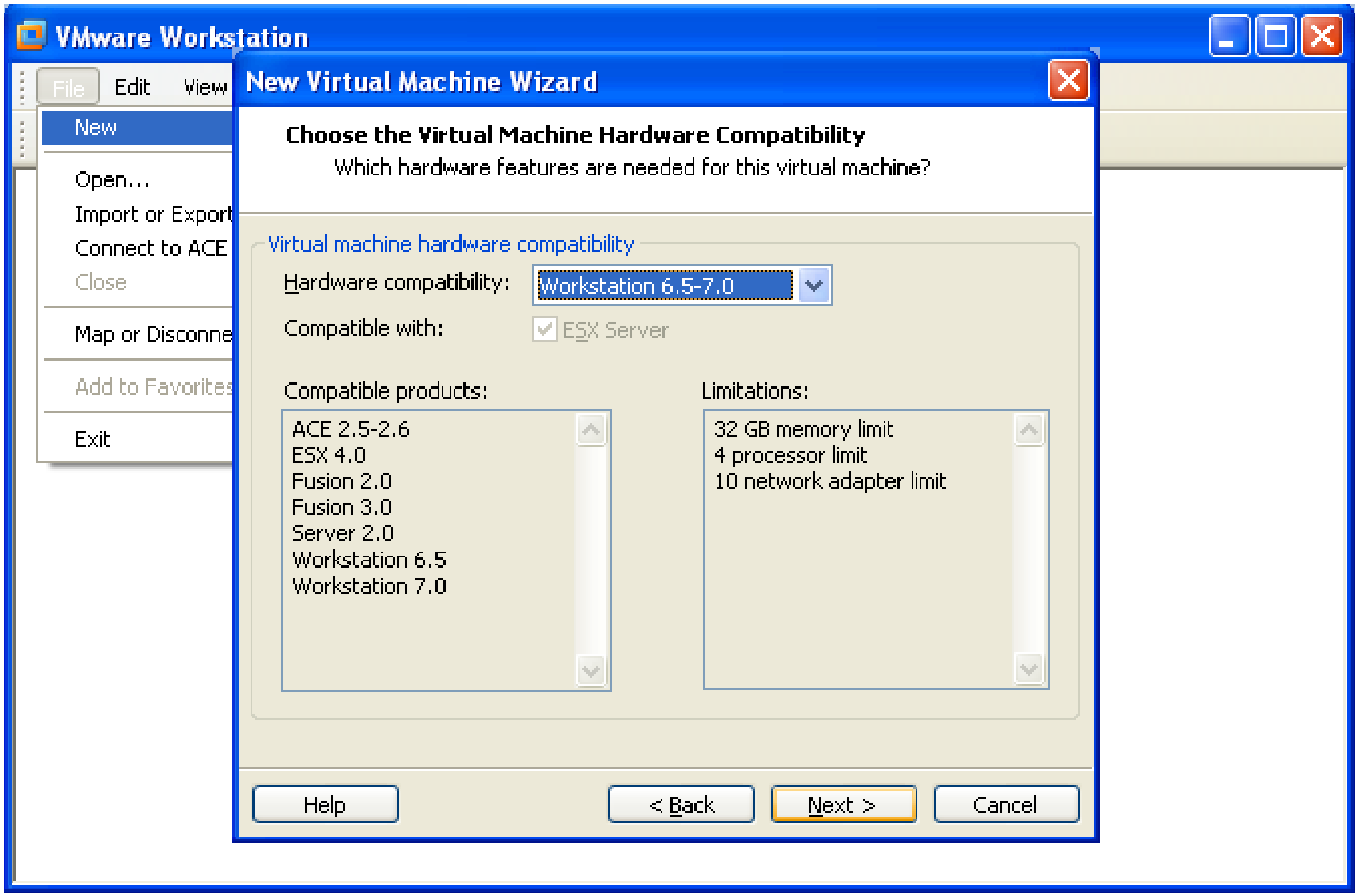
Preparation - Creating the virtual machines (2)

- Select "Custom (advanced)" as it enables you to customize the vm



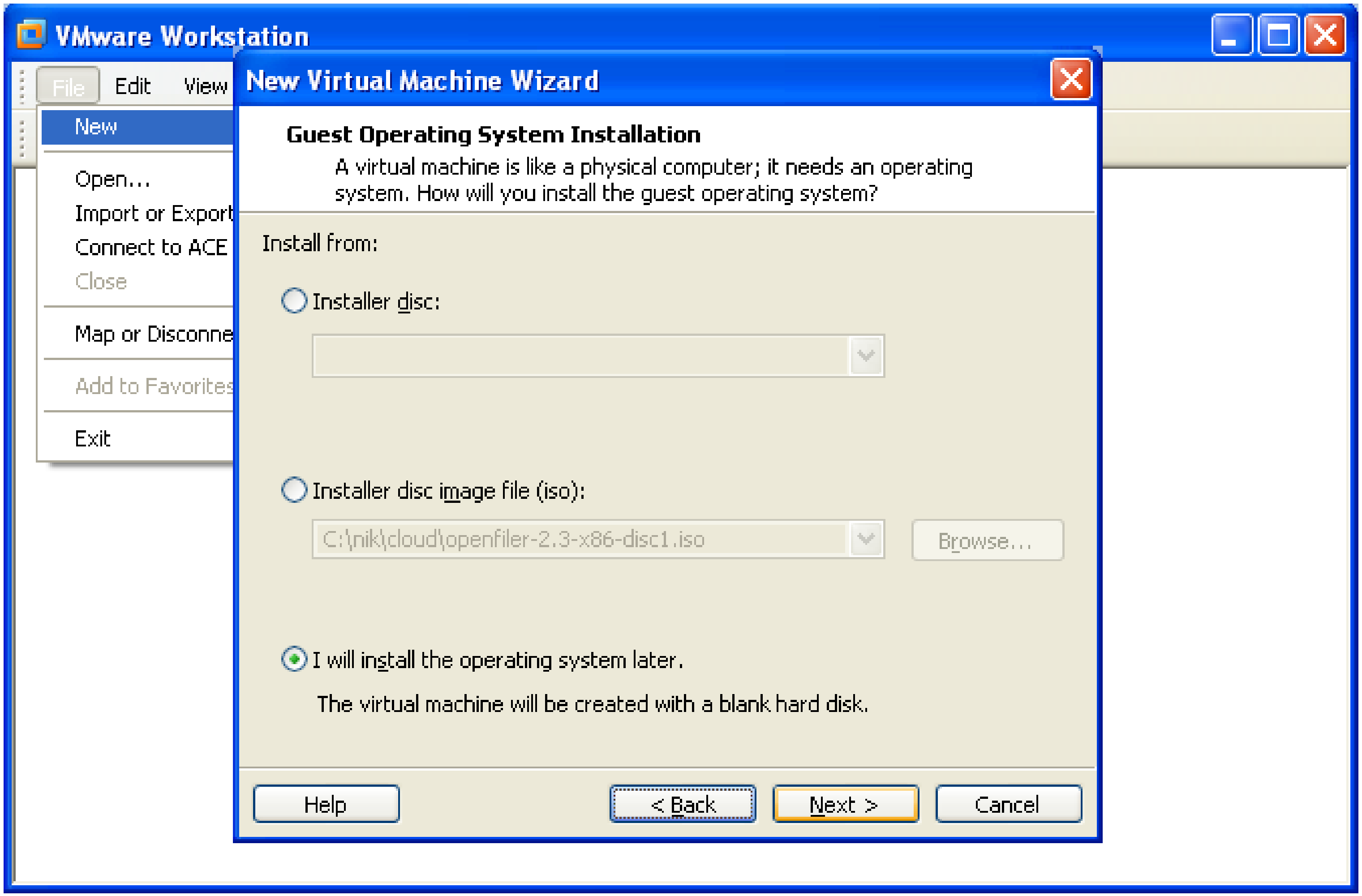
Preparation - Creating the virtual machines (3)

- Set *Hardware compatibility* to Workstation 6.5-7.0 (default)



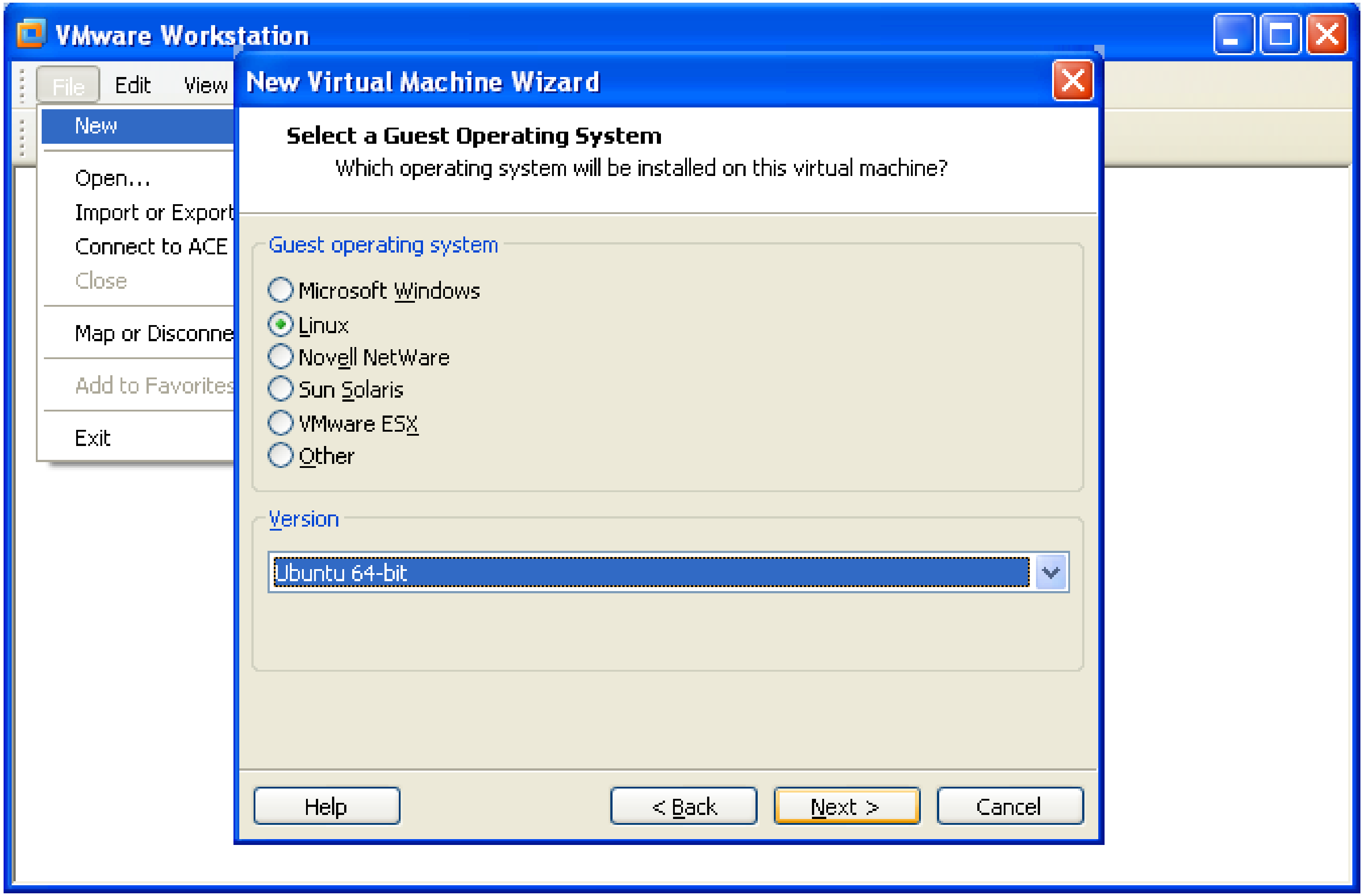
Preparation - Creating the virtual machines (4)

- Set *I will install the operating system later*



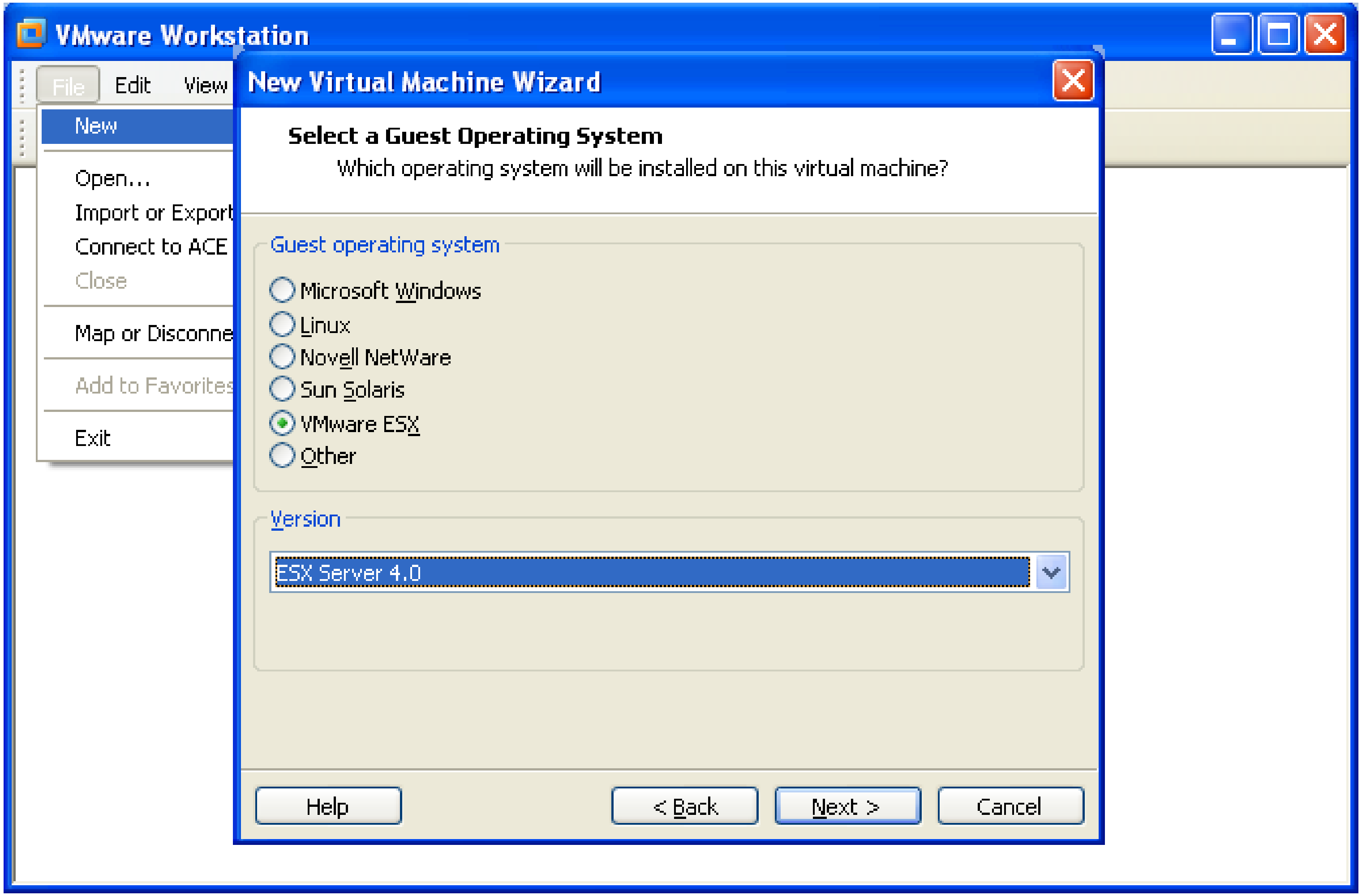
Preparation - Creating the virtual machines (5)

- For the controller and storage set Ubuntu 64-bit,
  For the nodes set ESX Server 4.0



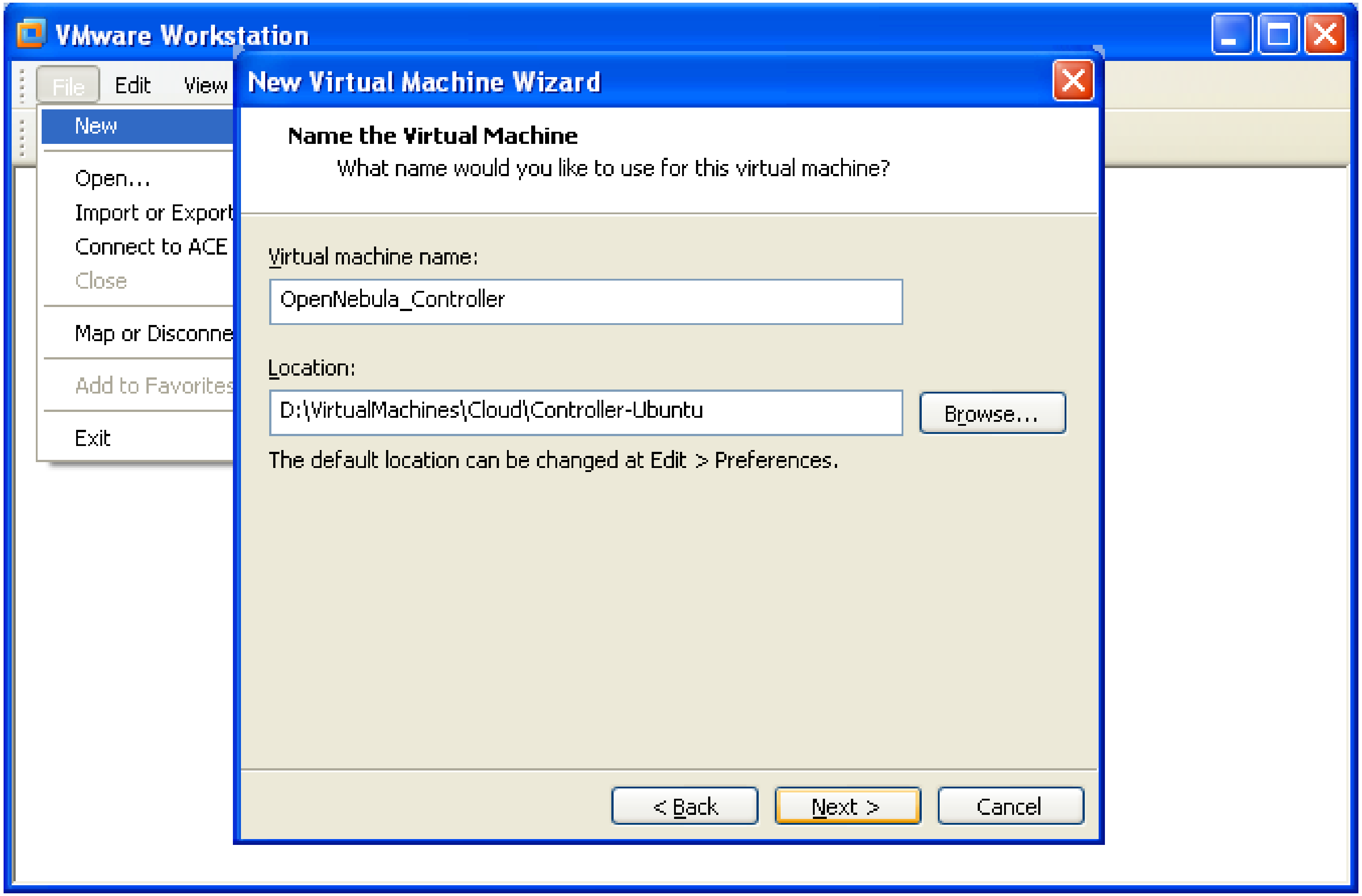
Preparation - Creating the virtual machines (6)

- For the controller and storage set Ubuntu 64-bit,
  For the nodes set ESX Server 4.0



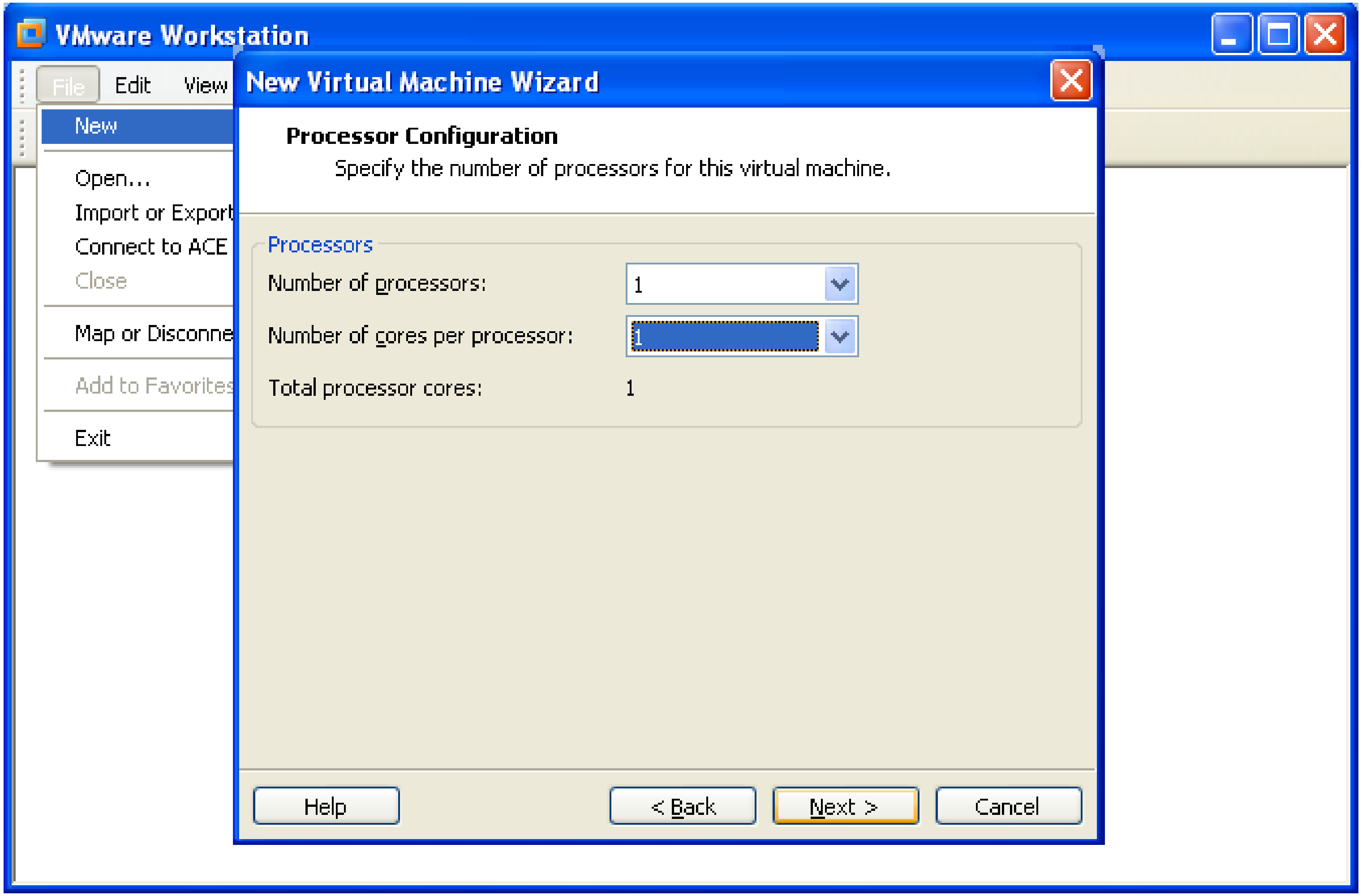
Preparation - Creating the virtual machines (7)

- Set the name and directory. In the lab, please use D:\VirtualMachines\Cloud\... directory as a base for the virtual machines



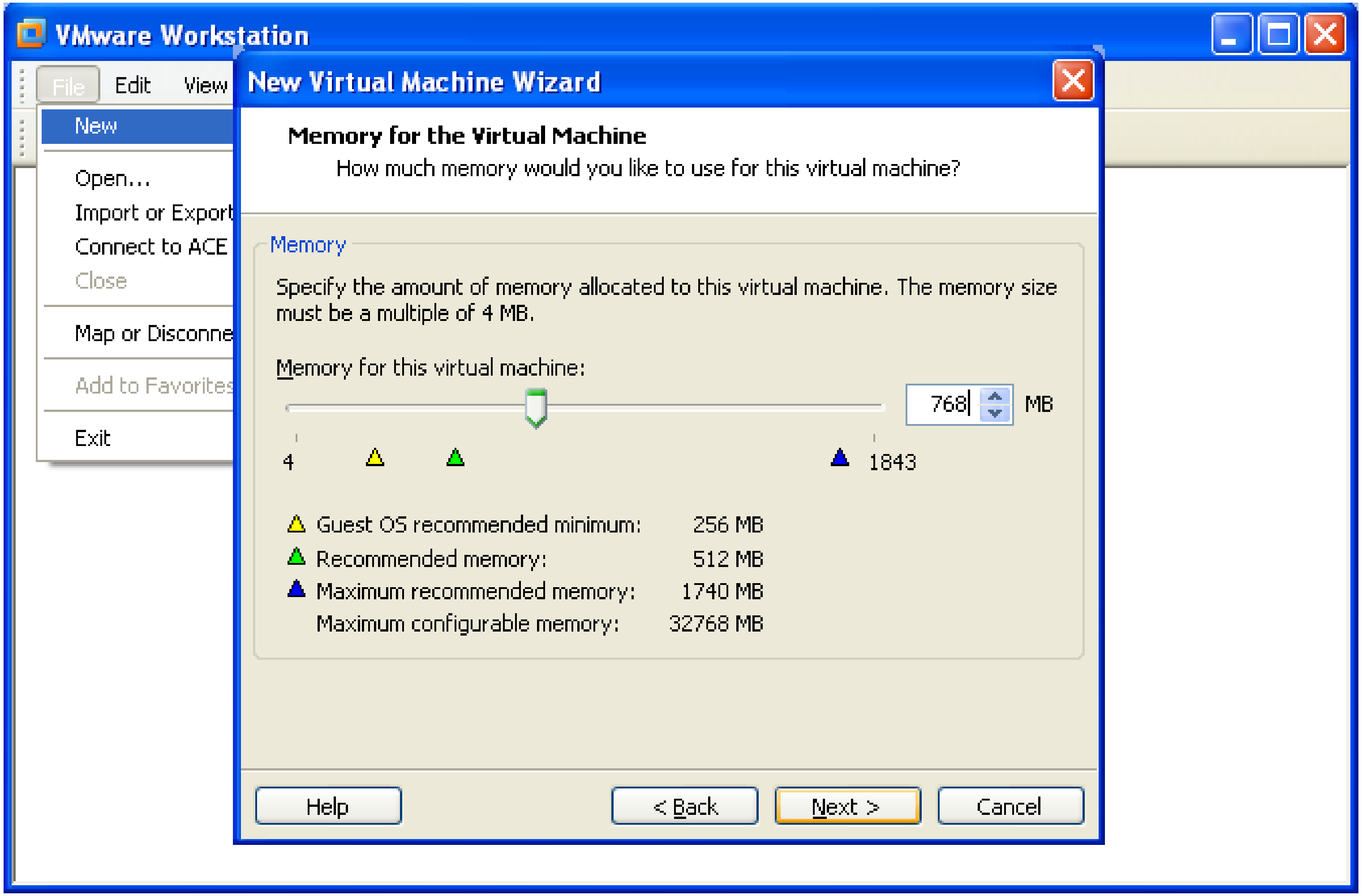
Preparation - Creating the virtual machines (8)

- Set the number of cores according to the proposed configuration (Number of processors: 1 for controller and storage, 2 for nodes)



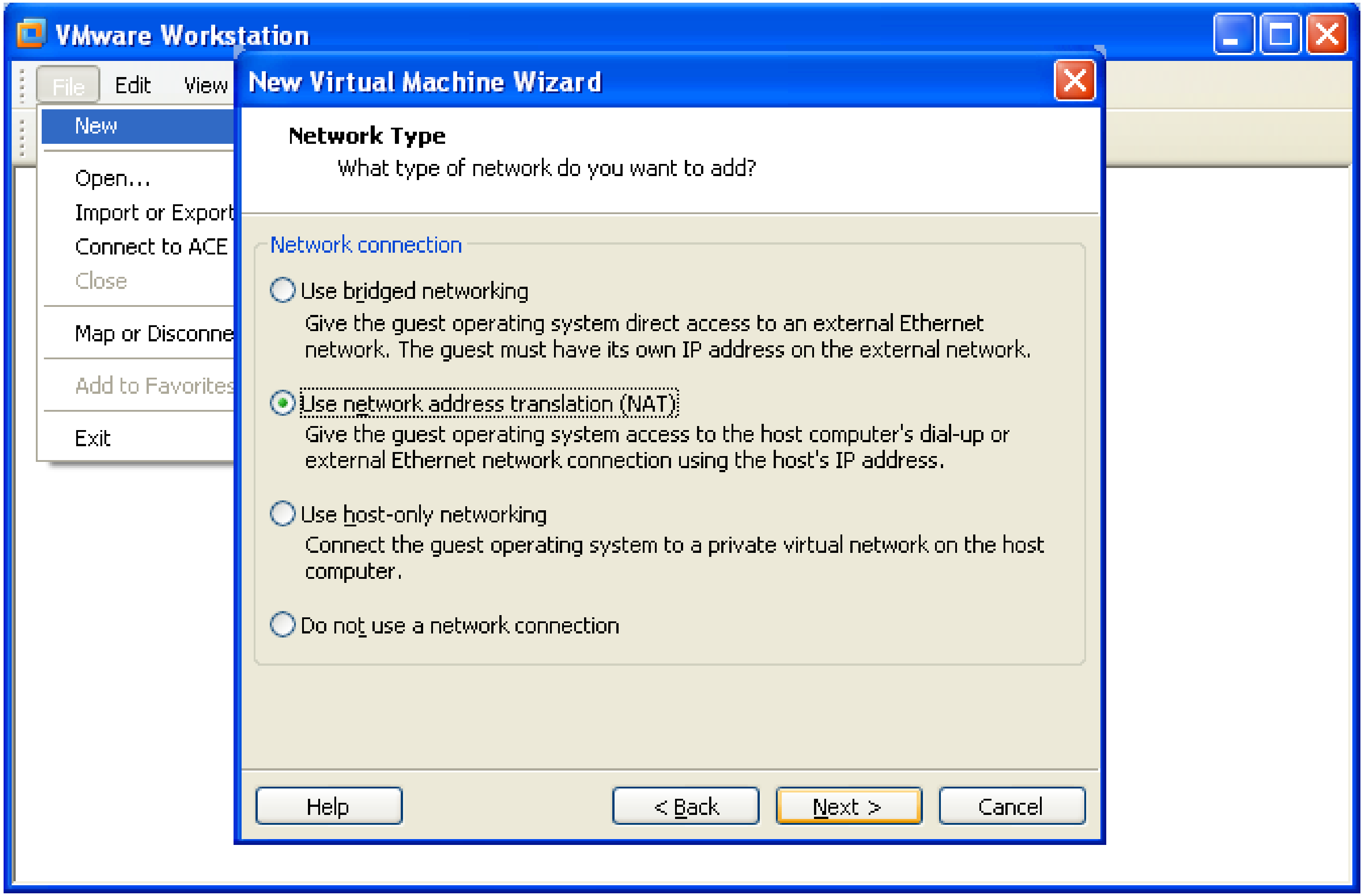
Preparation - Creating the virtual machines (9)

- Set the amount of memory according to the proposed configuration (768 for the controller and storage, 2048 for the nodes)



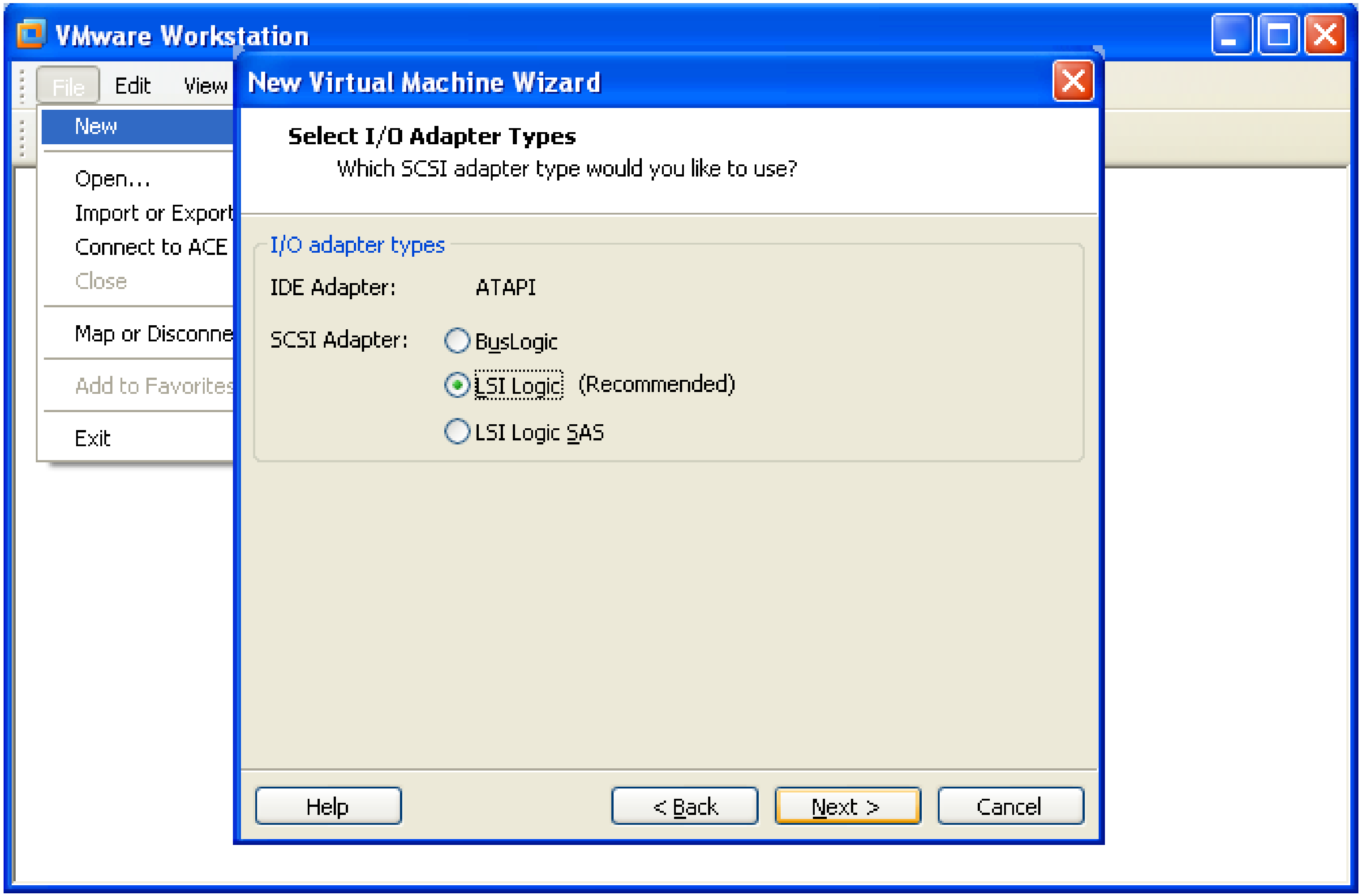
Preparation - Creating the virtual machines (10)

- Set the network type (Choose NAT for all the machines)



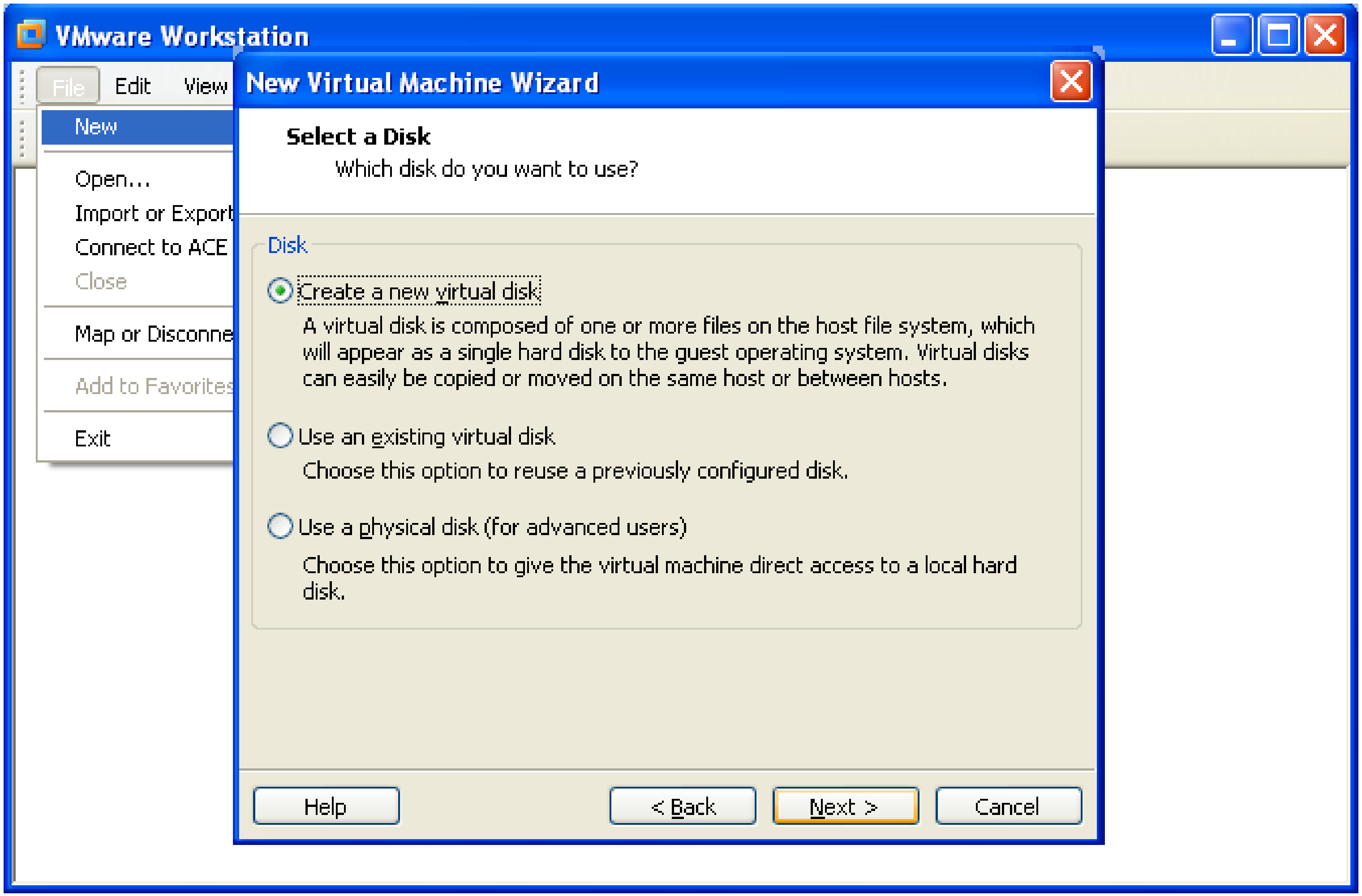
Preparation - Creating the virtual machines (11)

- Select the I/O Adapter Types - set whatever is the default for the template



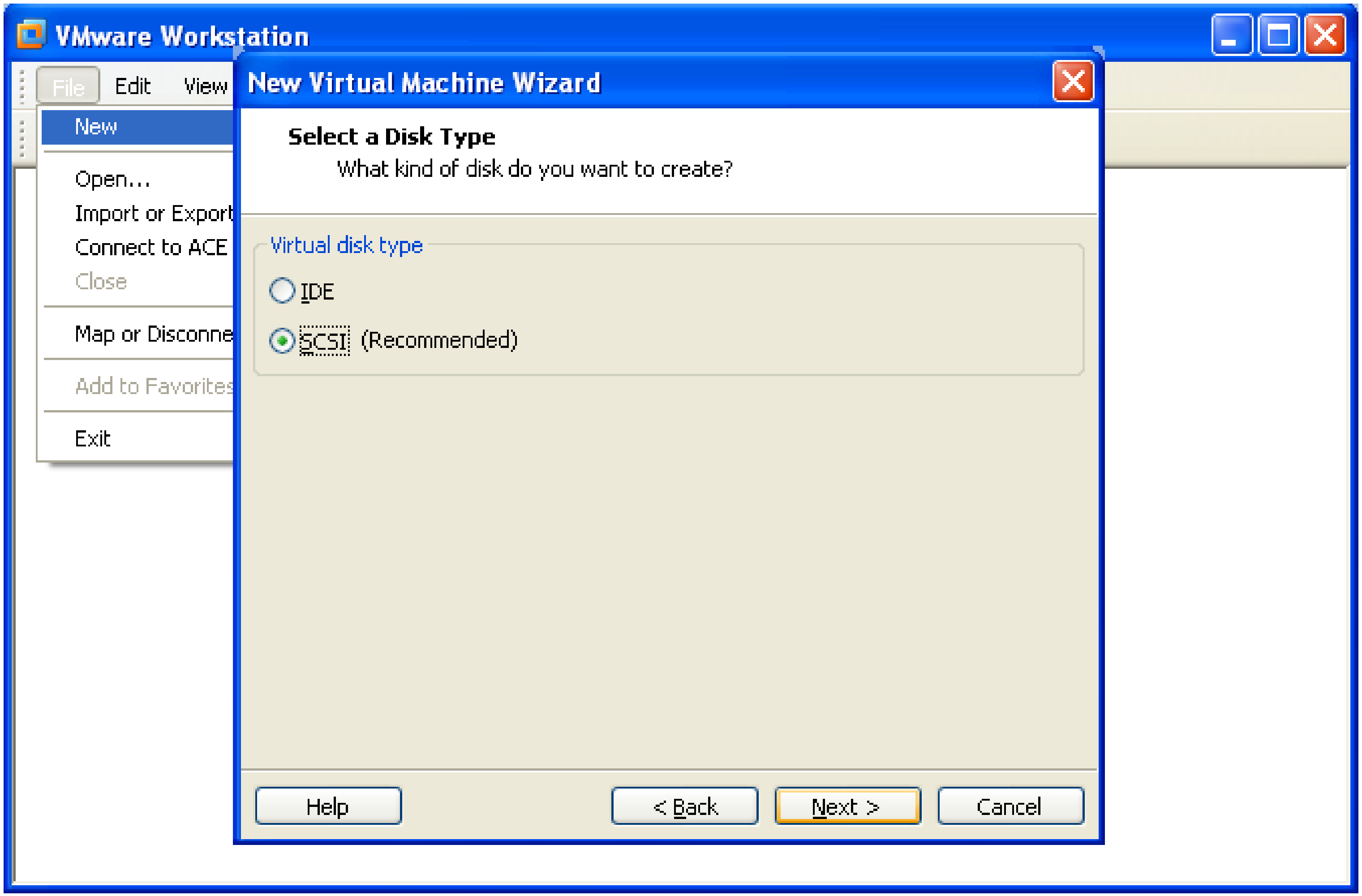
Preparation - Creating the virtual machines (12)

- Select a disk on which to install - *Create a new virtual disk*



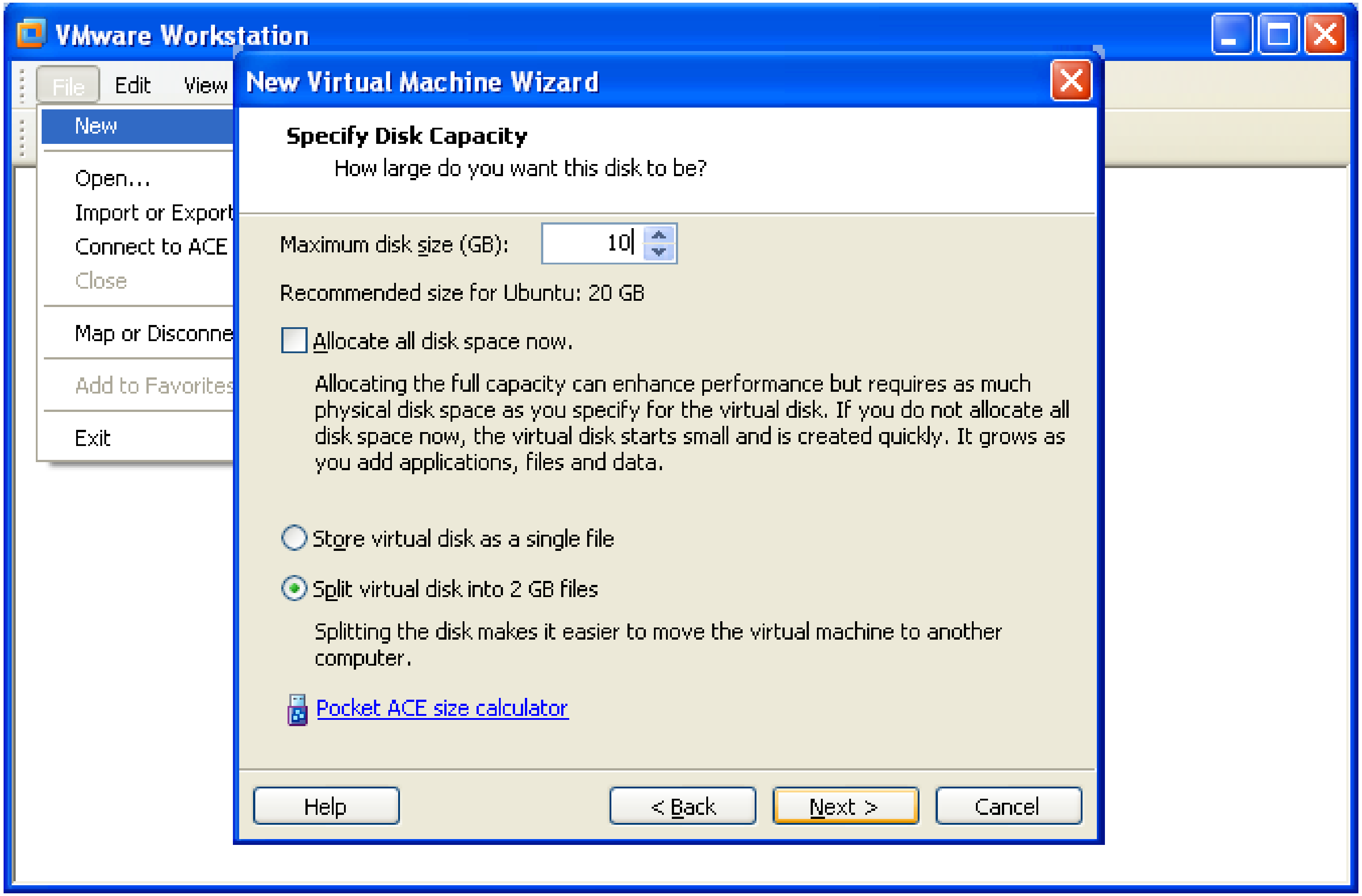
Preparation - Creating the virtual machines (13)

- Select a disk type - Choose the Recommended for the template



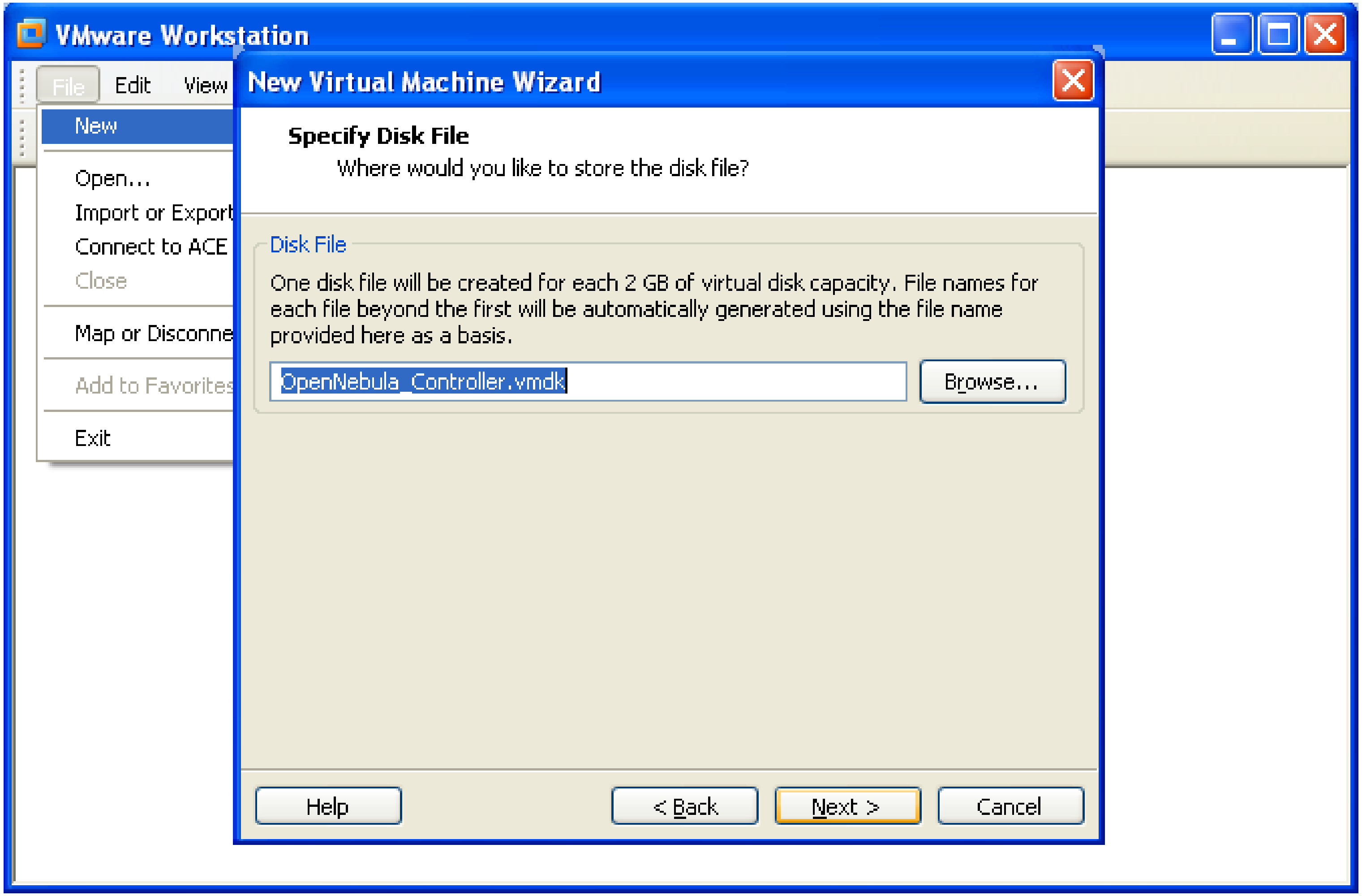
Preparation - Creating the virtual machines (14)

- Specify disk capacity - 10 GB for Controller and the Nodes
  70 GB for the Storage



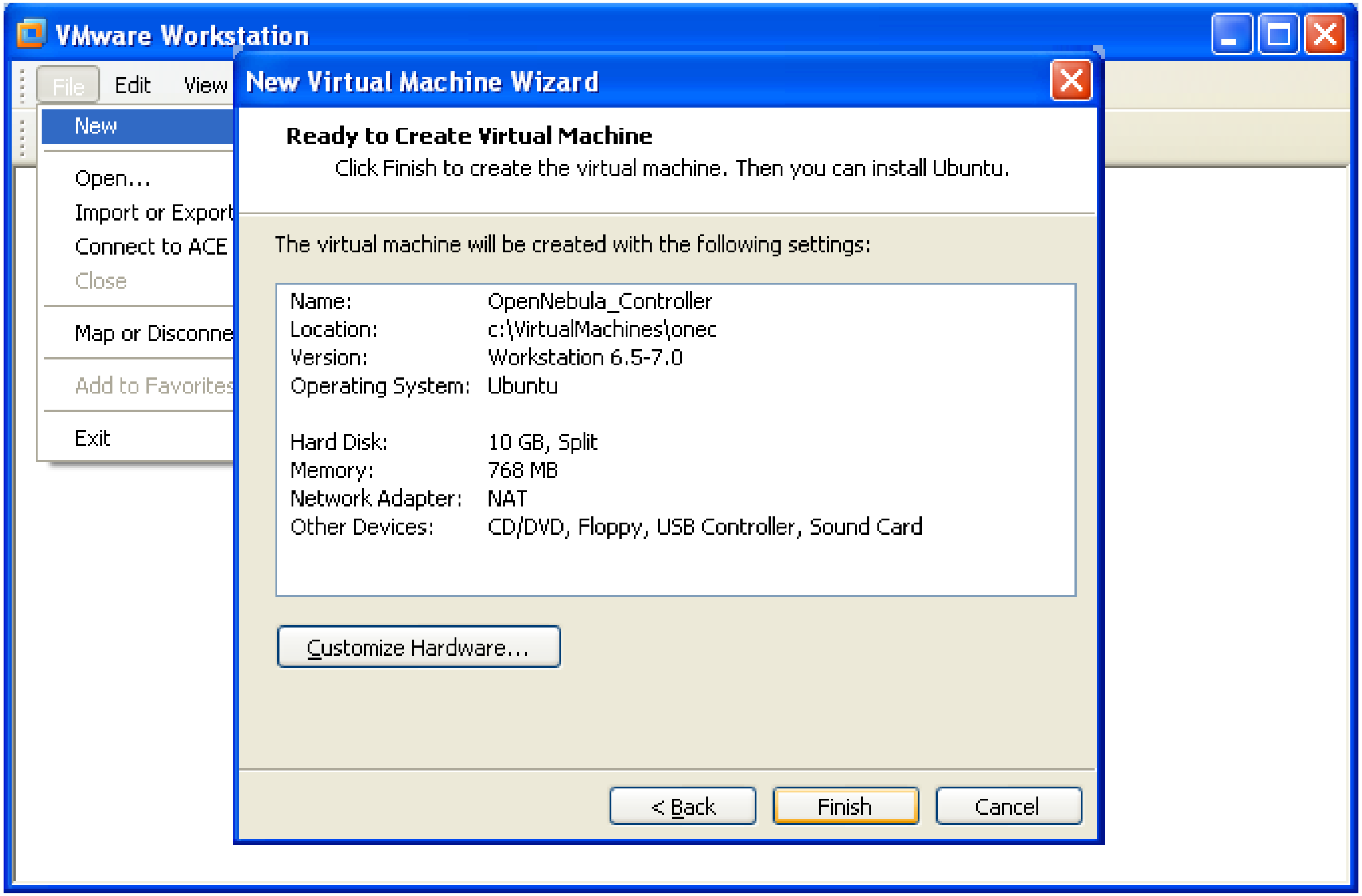
Preparation - Creating the virtual machines (15)

- Specify disk file - the default value is good



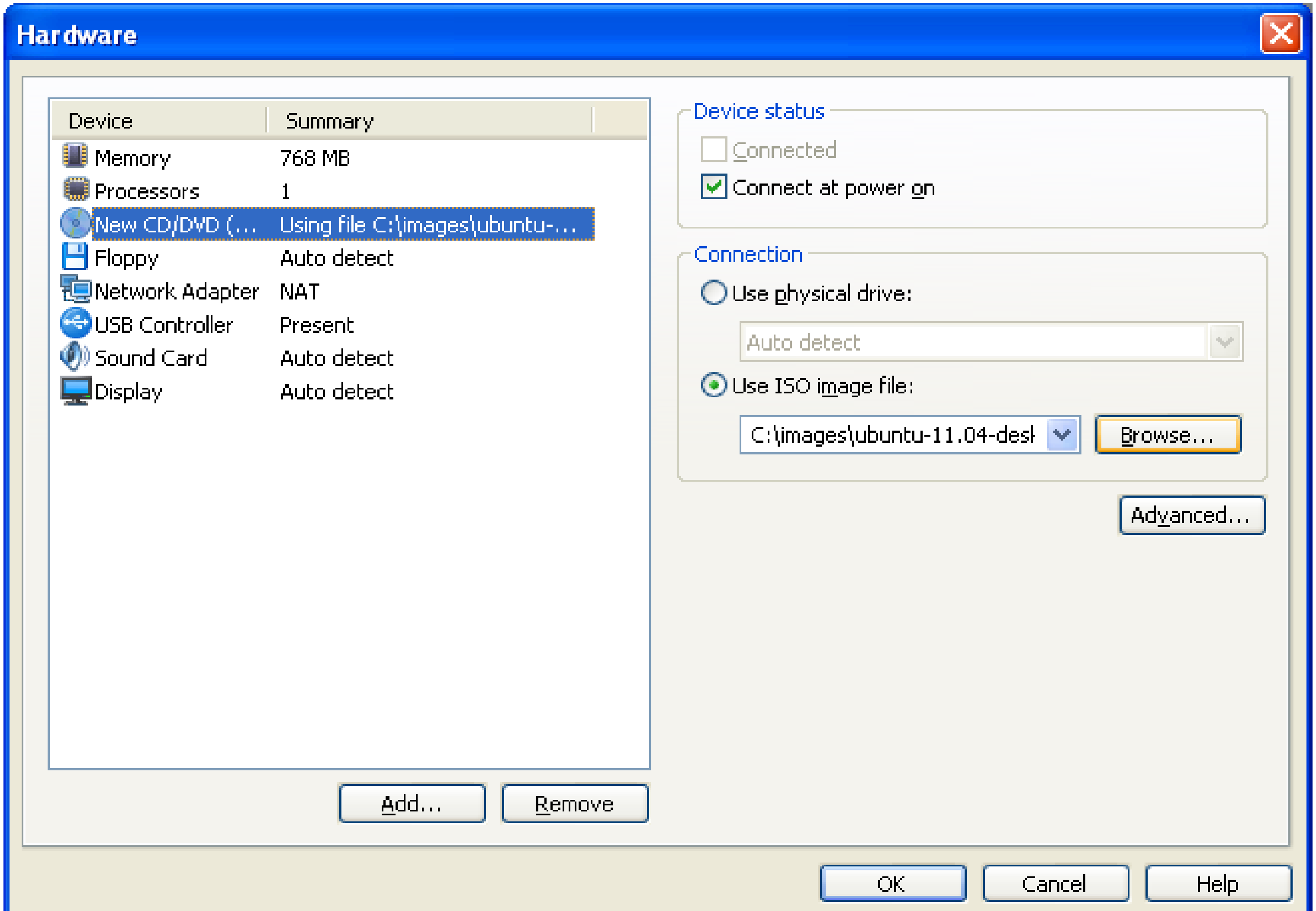
Preparation - Creating the virtual machines (16)

- Click Customize Hardware... button to set the install CD image



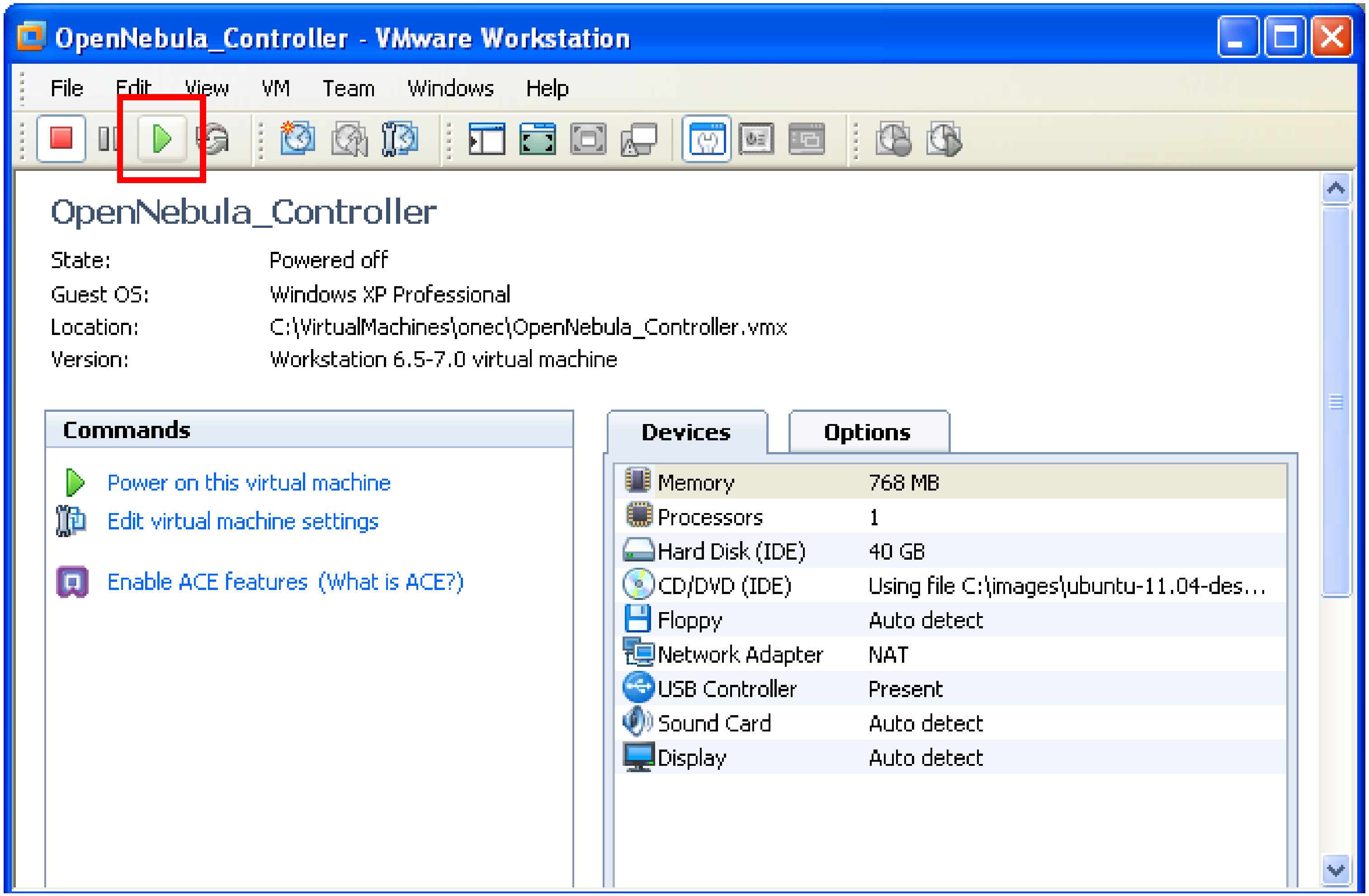
Preparation - Creating the virtual machines (17)

- Select the CD, Use ISO image, browse for the appropriate image



Preparation - Creating the virtual machines (18)

- Click on the green play button to start the virtual machine



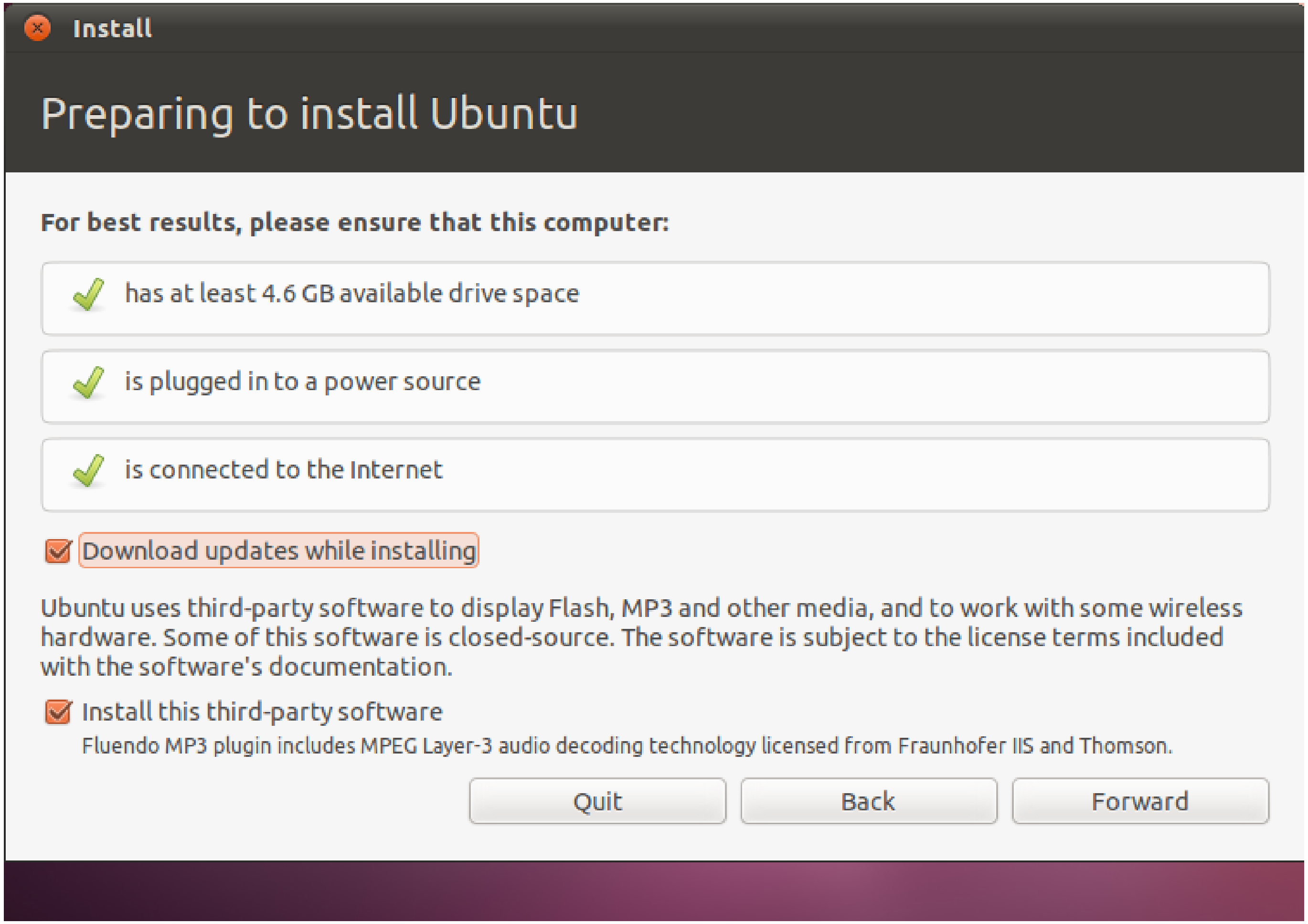
Appendix B - Installing Ubuntu 11.04 Desktop

Installing the Controller

- We will use ubuntu 11.04 Desktop image for our OpenNebula Controller
- Download installation disk from http://www.ubuntu.com
  - http://www.ubuntu.com/download/ubuntu/download
  - Ubuntu 11.04 - Latest version
  - 64-bit

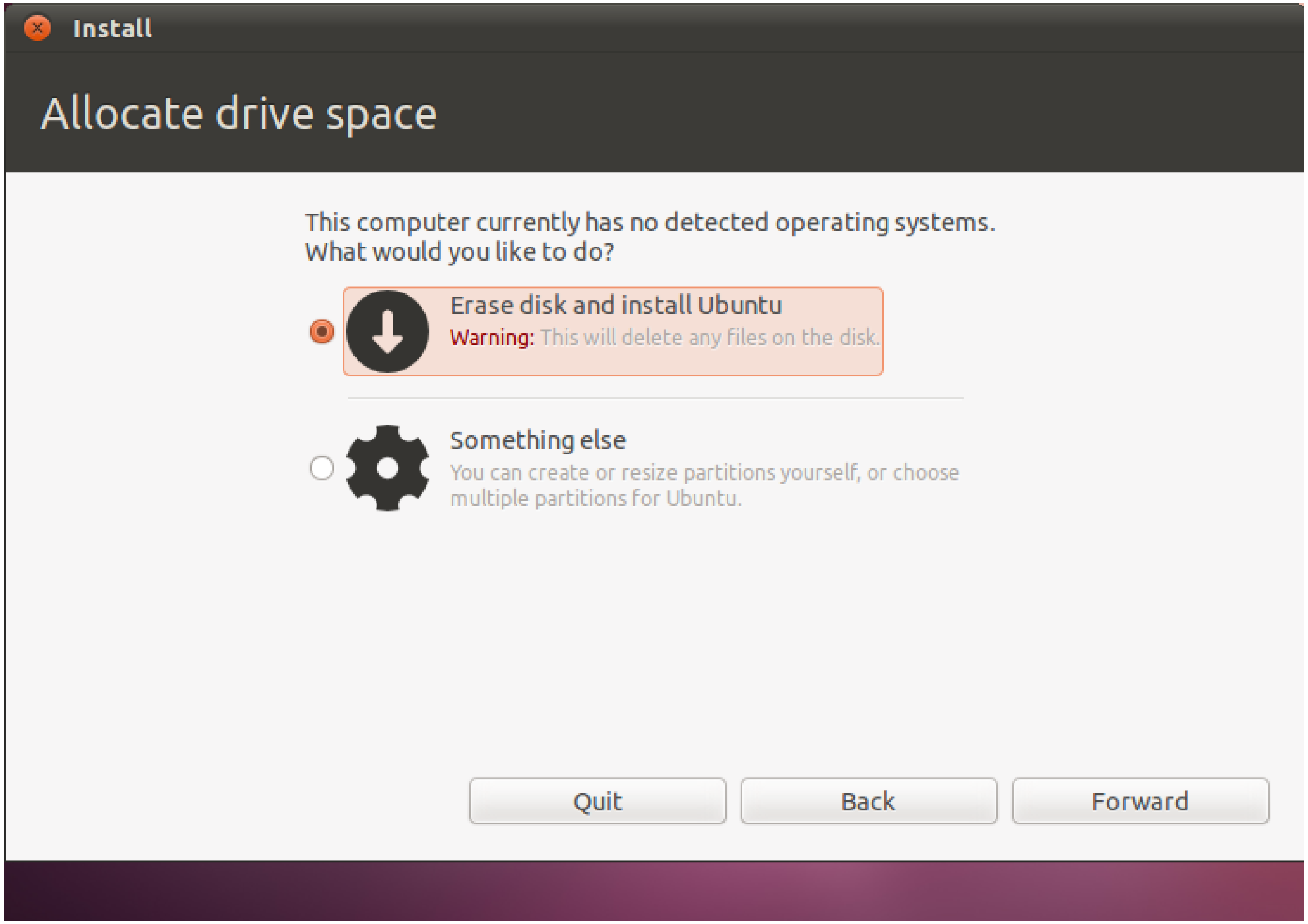
Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (1)

- Select Install Ubuntu at the boot, set Download Updates and click on *Forward*



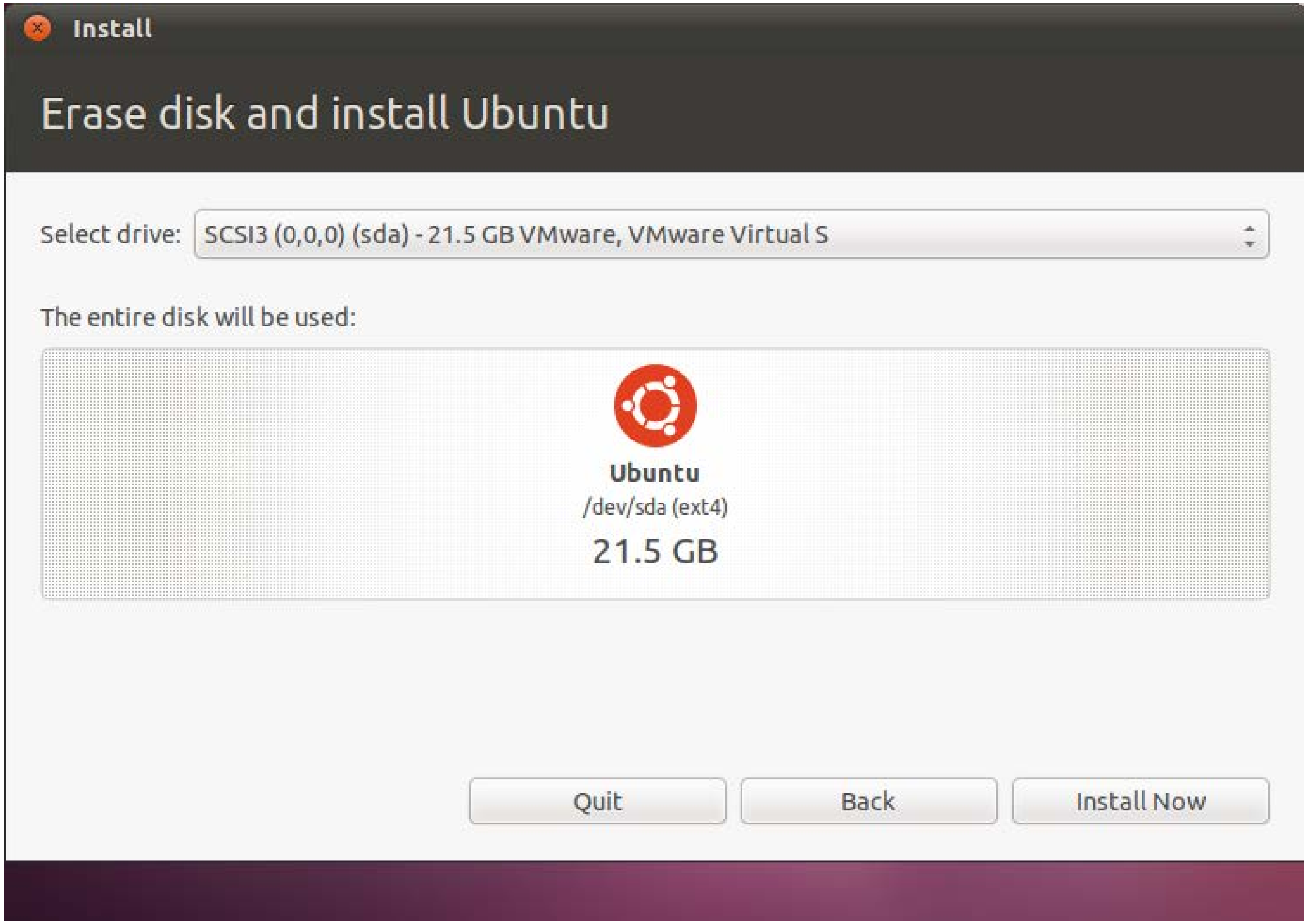
Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (2)

- Since this will be the only OS on the machine, it is safe to say Erase disk



Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (3)

- Leave everything on the default setting and press *Install Now*

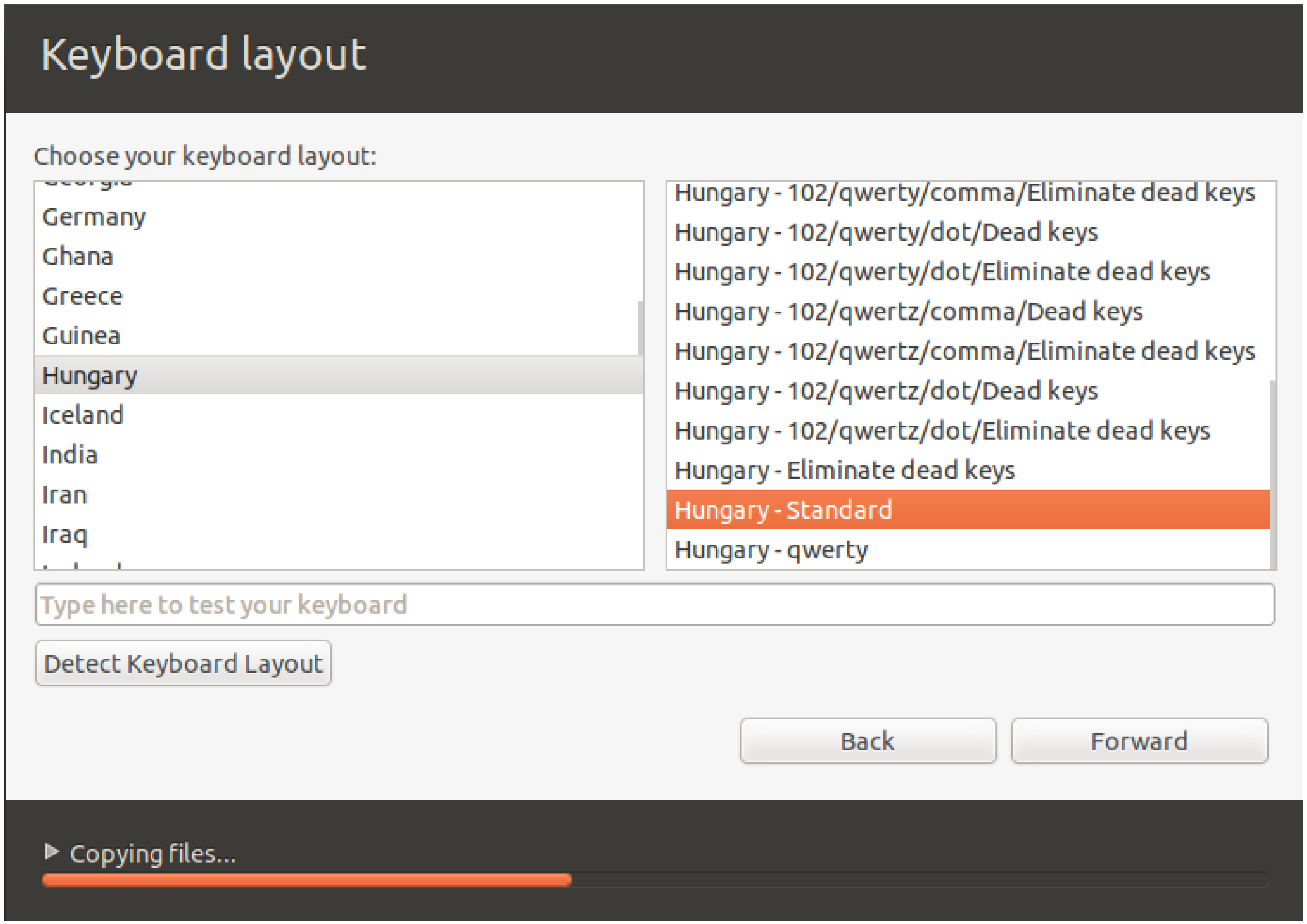


Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (4)

- Set the appropriate time zone

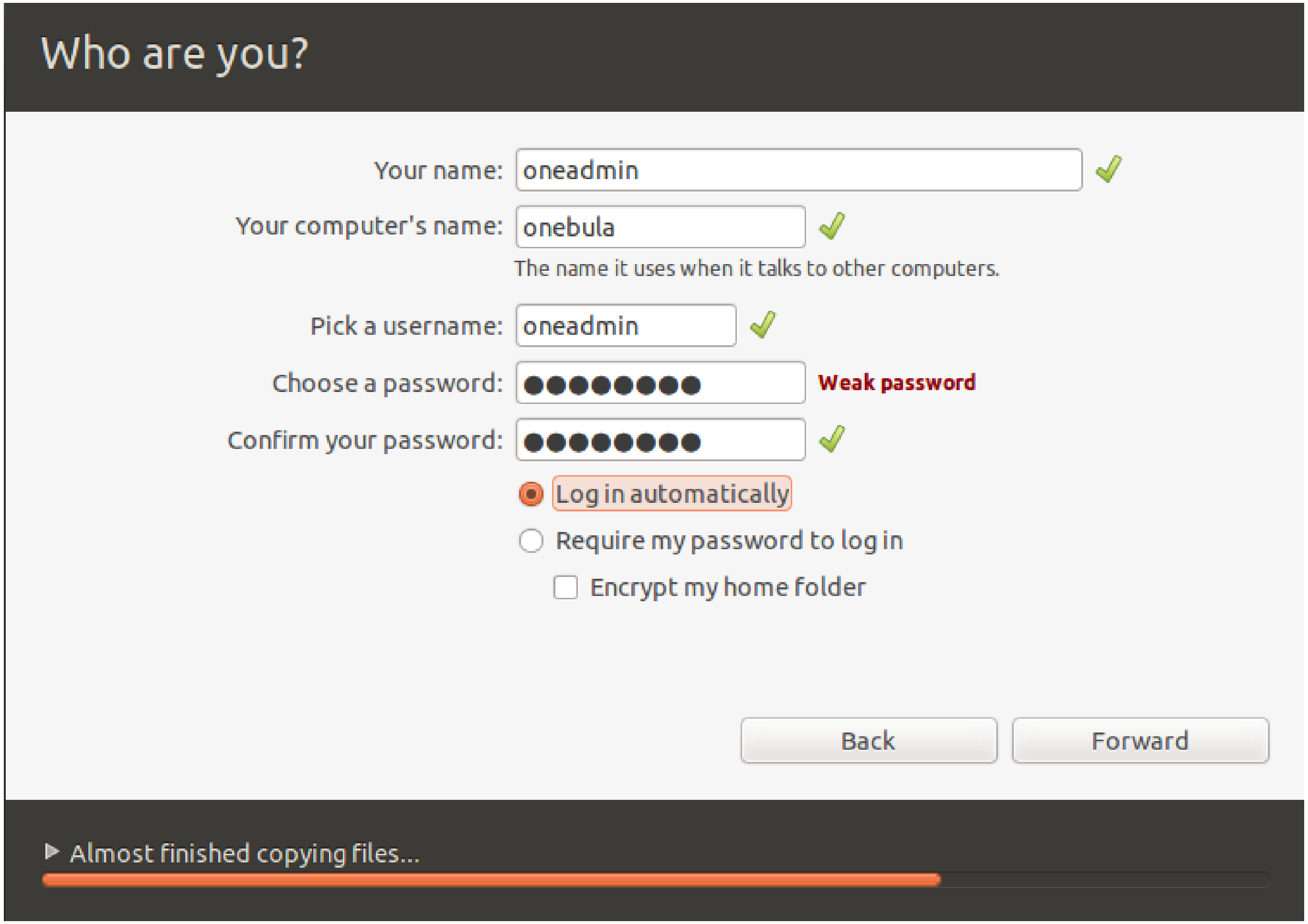
Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (5)

- Set the keyboard layout



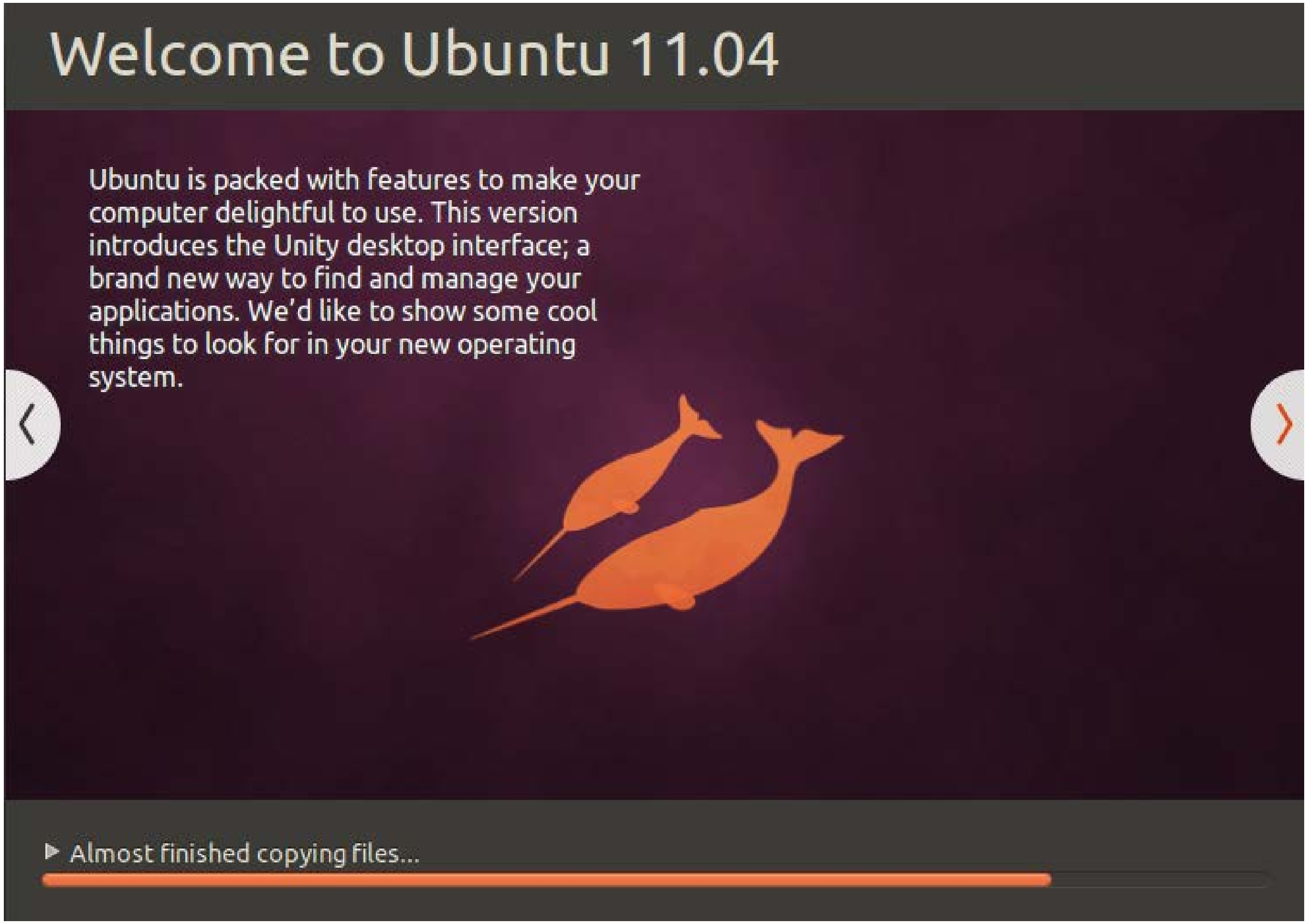
Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (6)

- Set the primary user: oneadmin - it has to be oneadmin



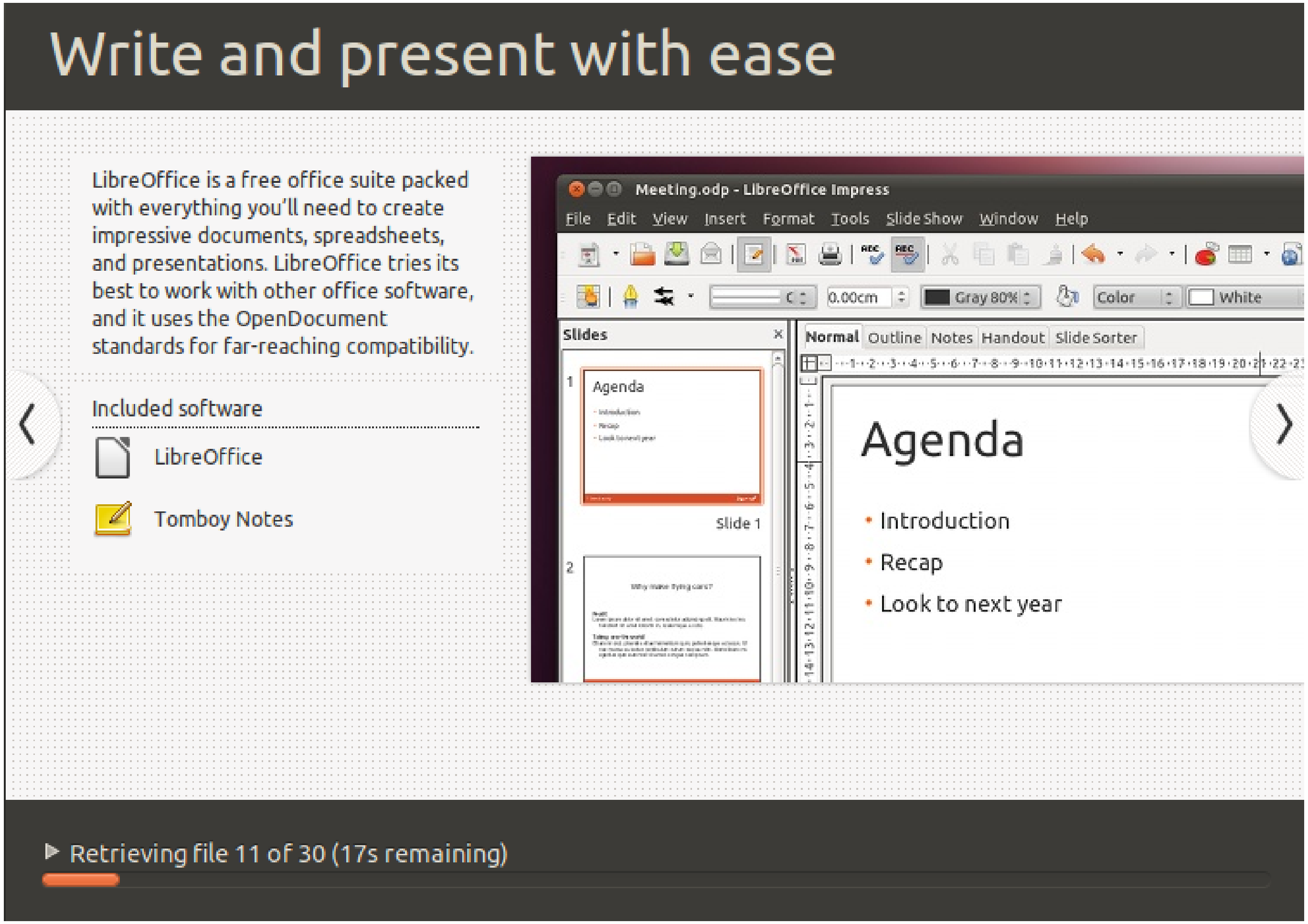
Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (7)

- Wait, wait, wait. Read the presentation to kill time



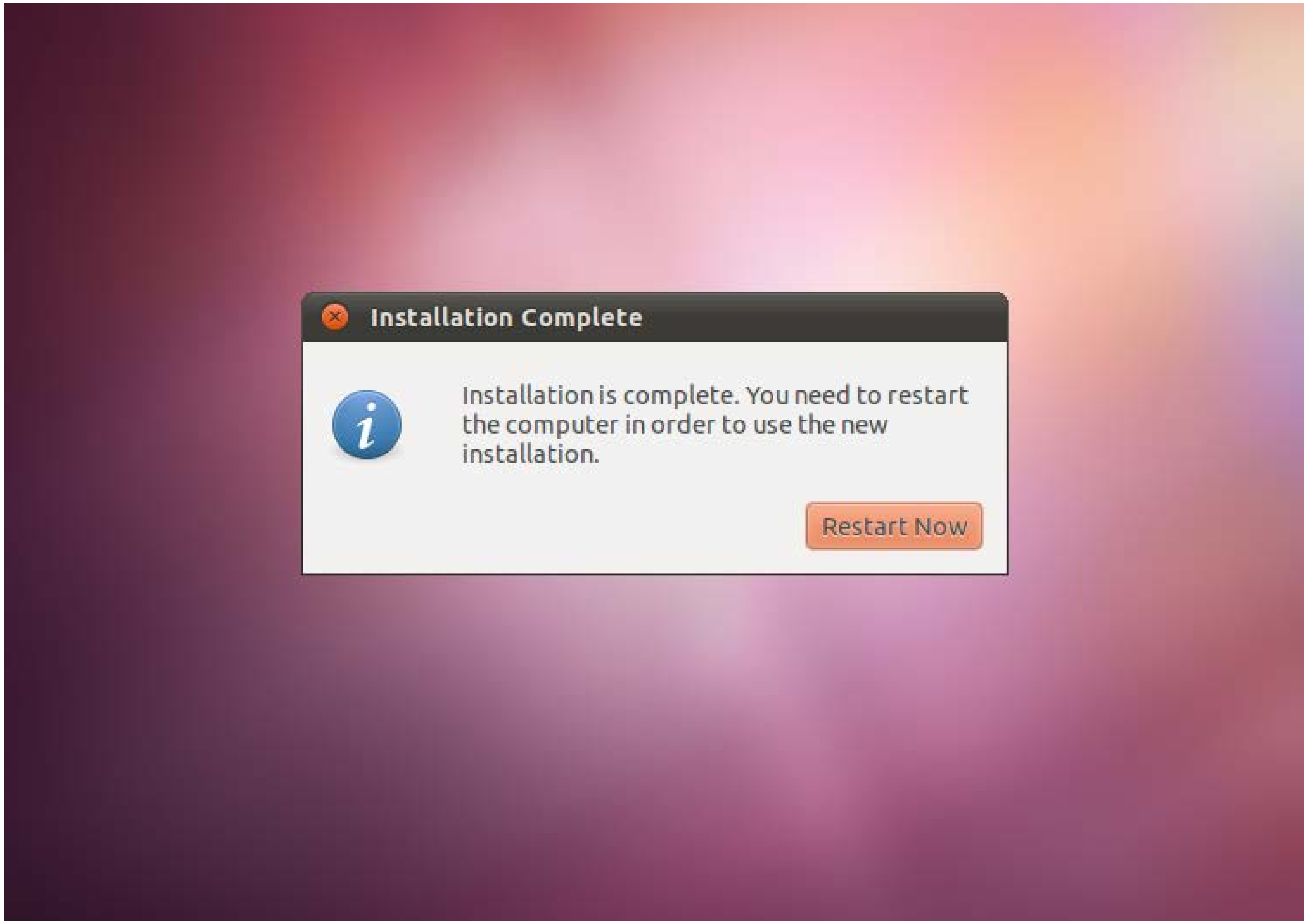
Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (8)

- Wait, wait, wait some more. The system downloads the updates



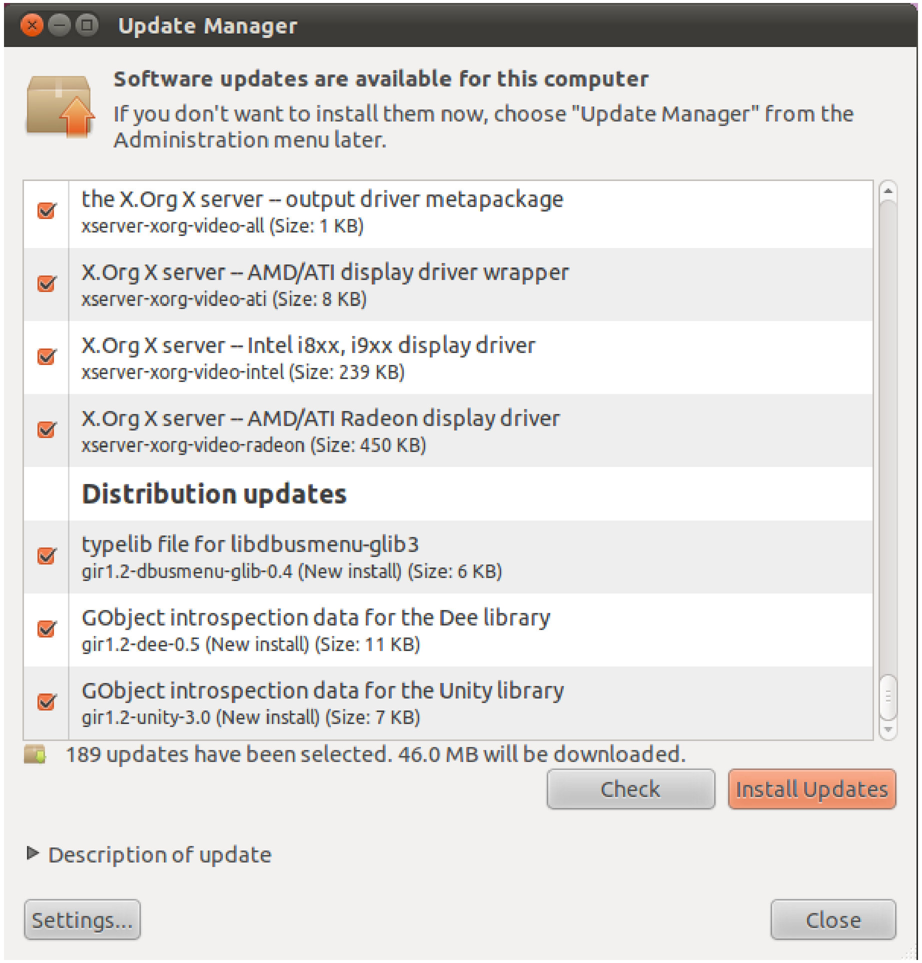
Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (9)

- Press *Restart Now* when the installation is complete



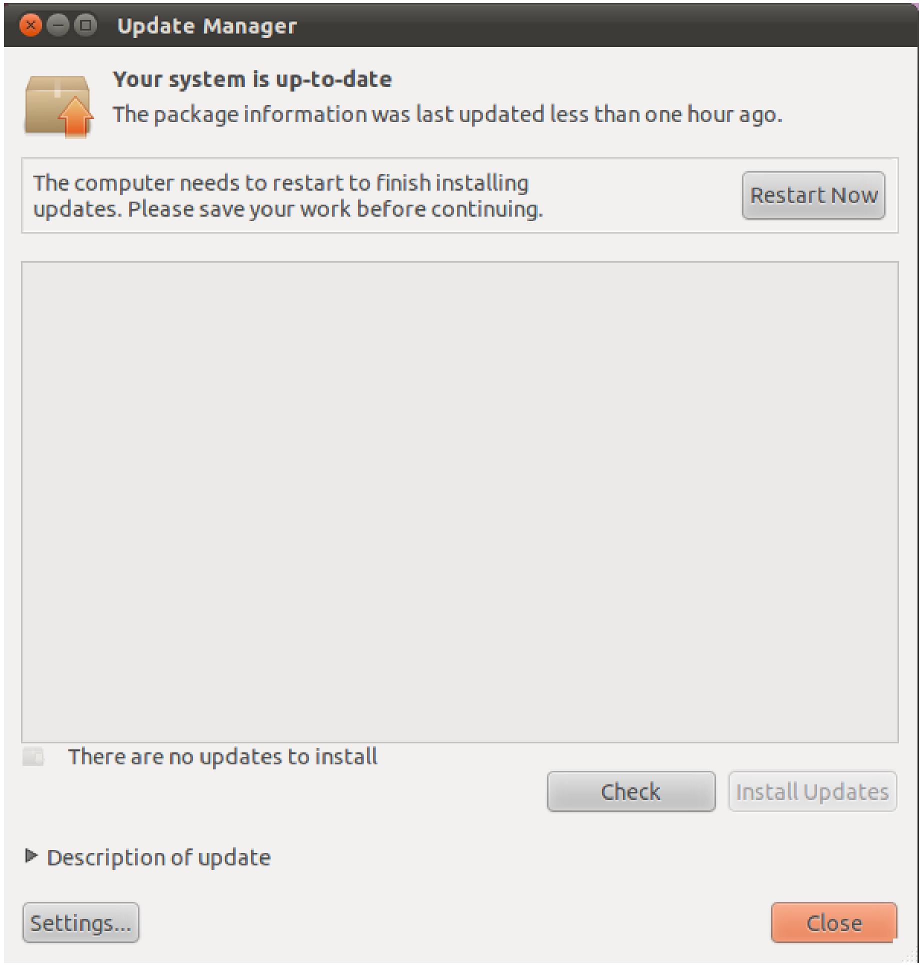
Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (10)

- After the first boot the system asks you to install updates



Preparation - Installing the controller - Ubuntu 11.04 Desktop edition (11)

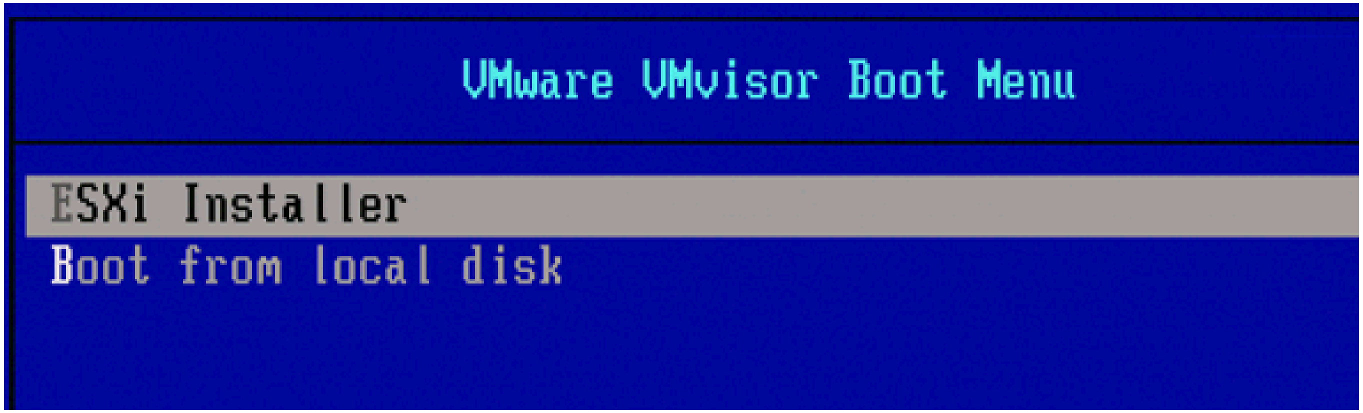
- Click on Restart Now when the updates are installed
  You are ready to install OpenNebula now



Appendix C - Installing Vmware ESXi

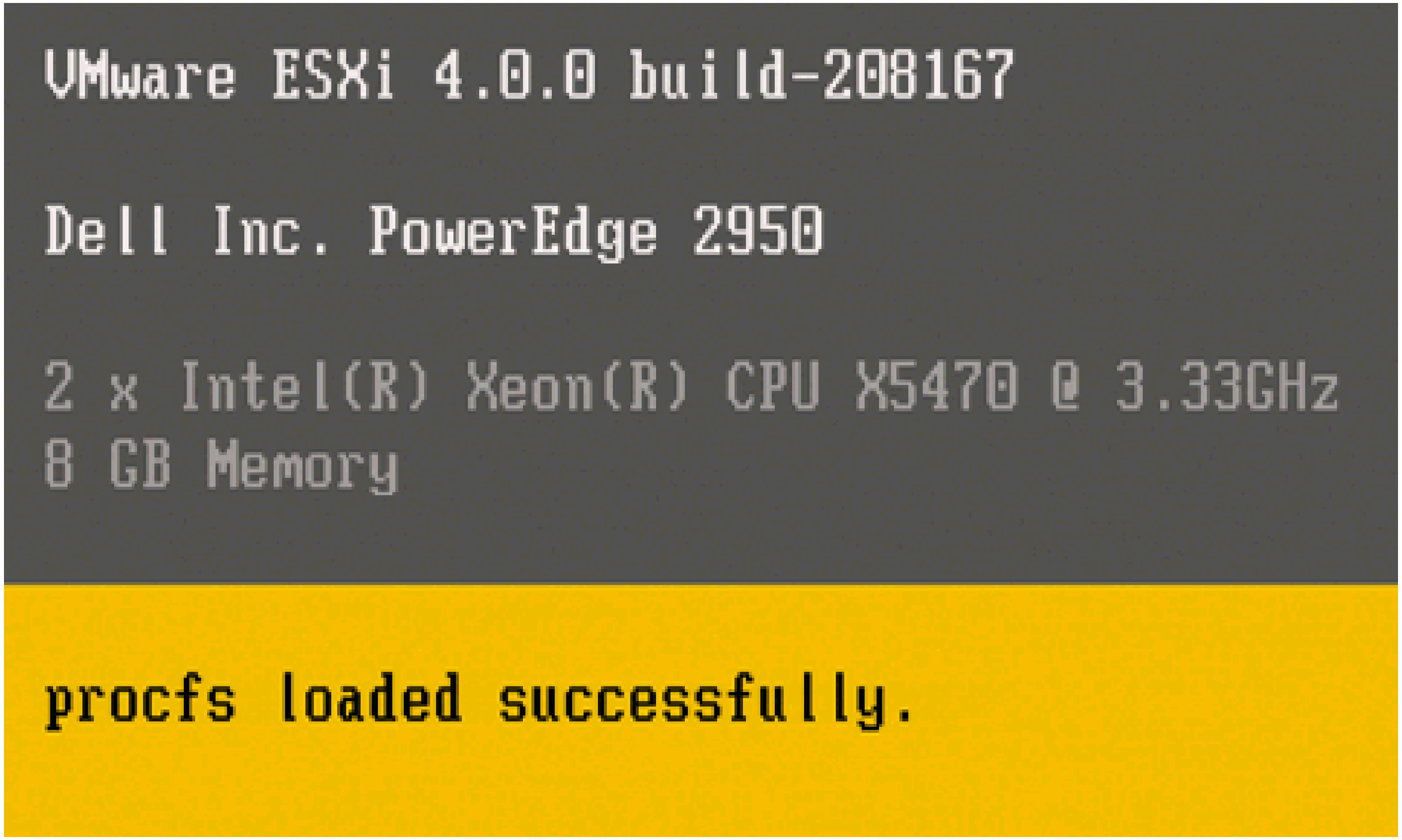
Preparation - Installing the ESXi hosts (1)

- Choose ESXi Installer at the boot prompt
  - Images in the guide were taken from an older version of ESXi



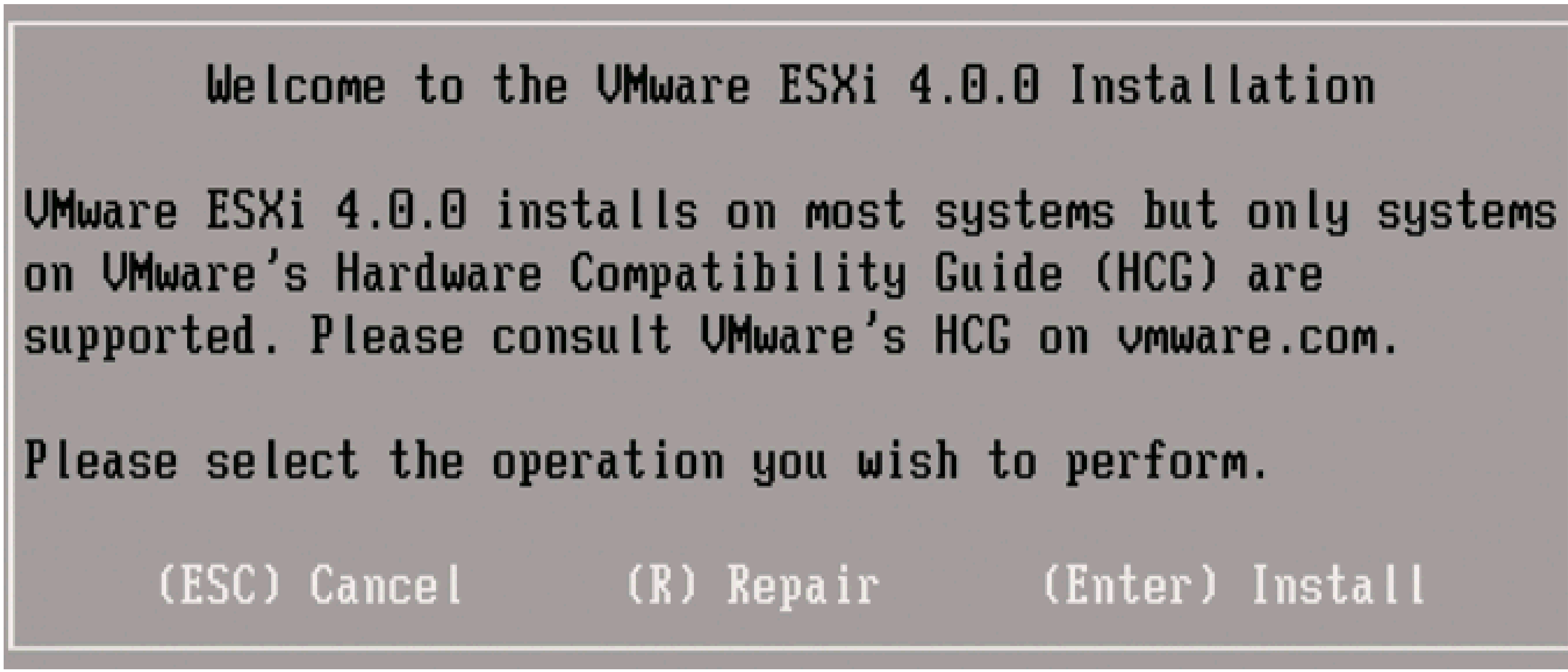
Preparation - Installing the ESXi hosts (2)

- Load screen



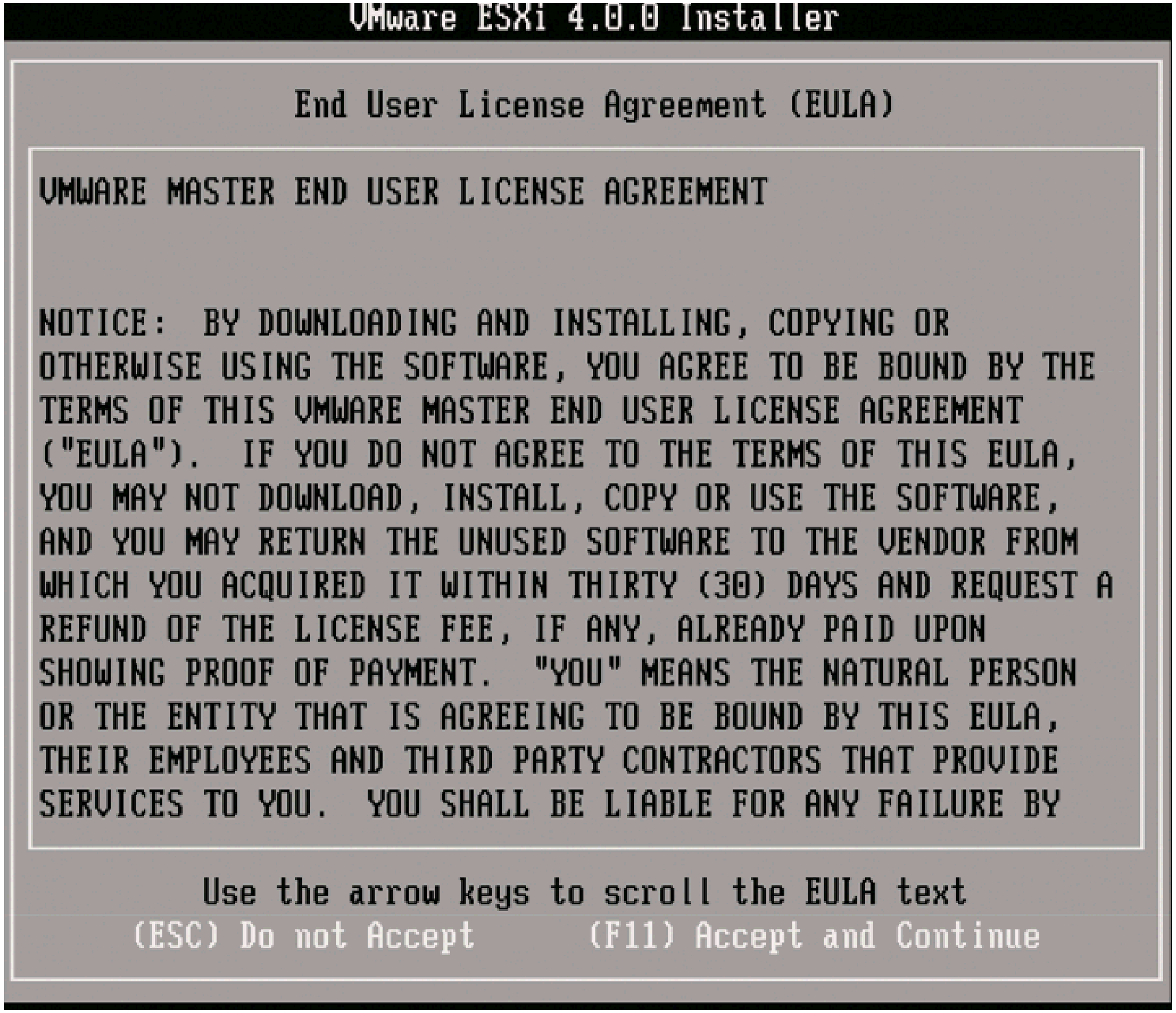
Preparation - Installing the ESXi hosts (3)

- Press Enter to install ESXi



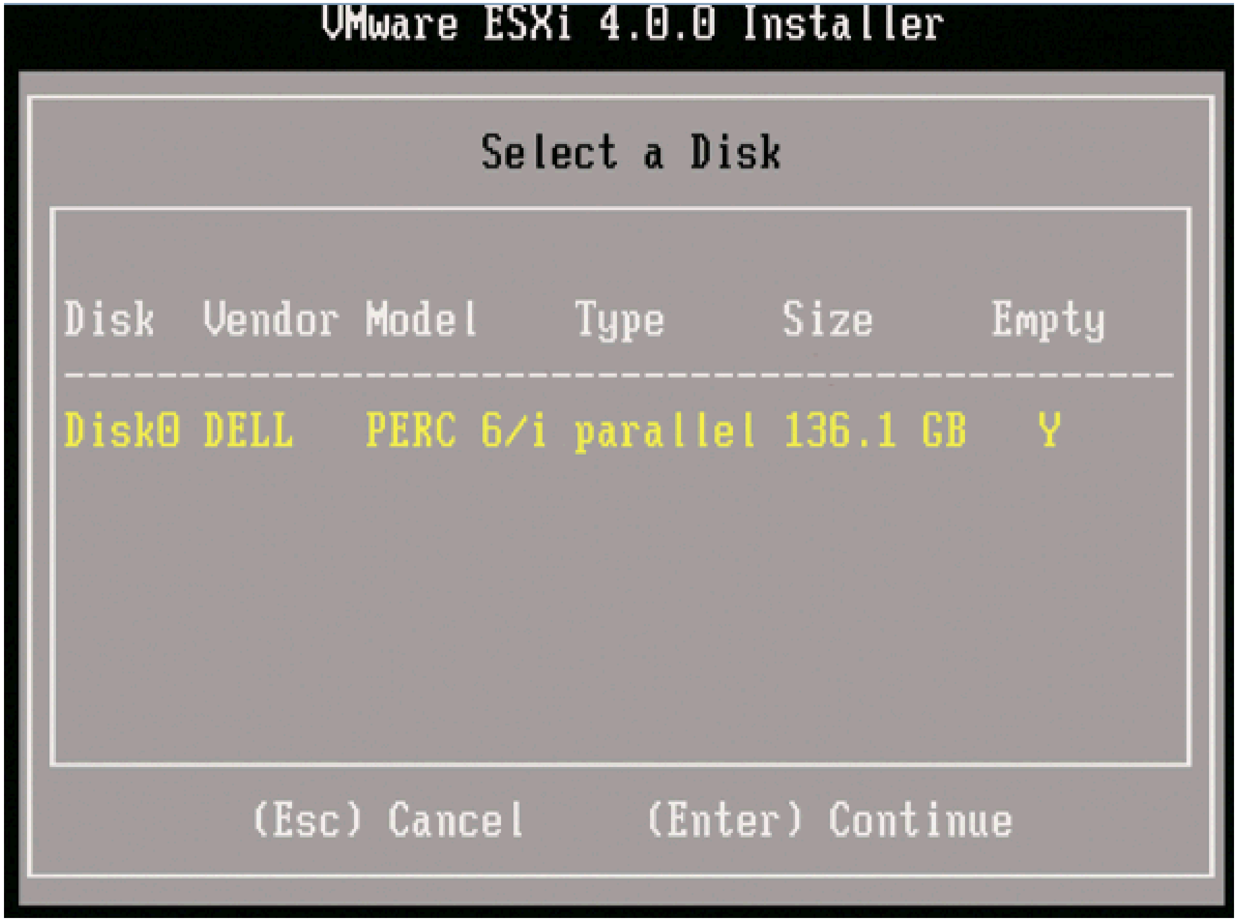
Preparation - Installing the ESXi hosts (4)

- Press F11 to accept EULA



Preparation - Installing the ESXi hosts (5)

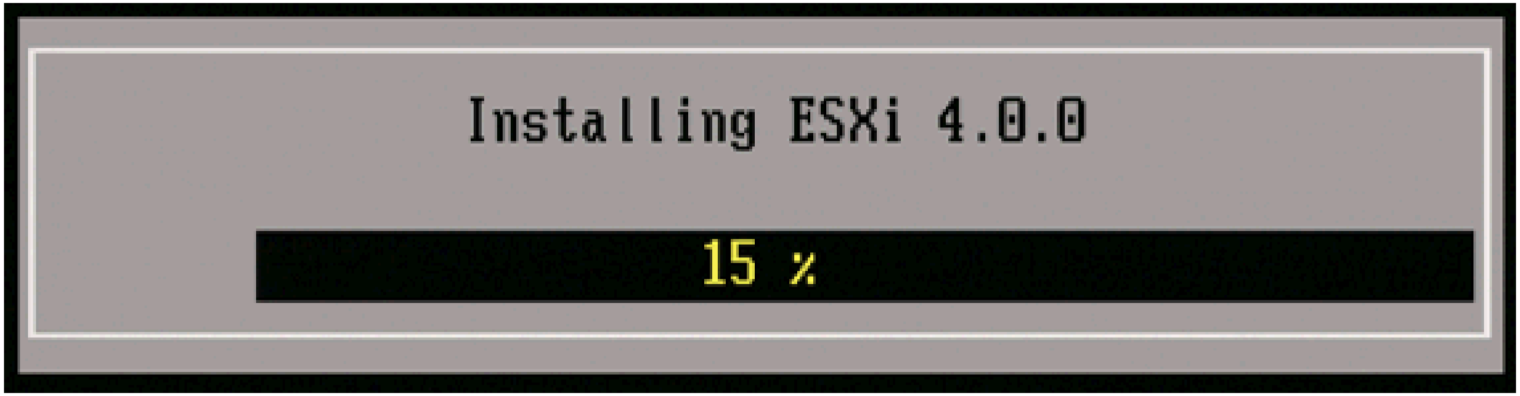
- Select disk



Preparation - Installing the ESXi hosts (6)

- Confirm installation

Preparation - Installing the ESXi hosts (7)



Preparation - Installing the ESXi hosts (8)

- If everything goes fine, this is the window you should see

Preparation - Installing the ESXi hosts (9)

- When the installation is done, it shows information on how to access the server