Minicurso SBRC 2010

Novas Arquiteturas de *Data Center* para *Cloud Computing*

Fábio Luciano Verdi – UFSCar
Christian Esteve Rothenberg – Unicamp
Rafael Pasquini – Unicamp
Maurício F. Magalhães – Unicamp

Gramado, Maio de 2010

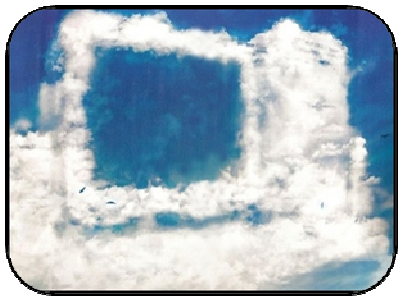
Organização do minicurso

- Introdução
  - O que é Cloud computing?
  - Definições e características essenciais
  - Custos e eficiência energética
- Caracterização dos *data centers* para serviços em nuvem
  - Infraestrutura de um *data center*
  - Limitações das arquiteturas de redes atuais
  - Caracterização do tráfego
  - Requisitos das arquiteturas de rede para novos *data centers*
- Novas arquiteturas para *data centers*
  - *Monsoon*
  - *VL2*
  - *Portland*
  - *BCube e MDCube*
- Conclusão
  - Cenários futuros
  - Desafios e oportunidades

Organização do minicurso

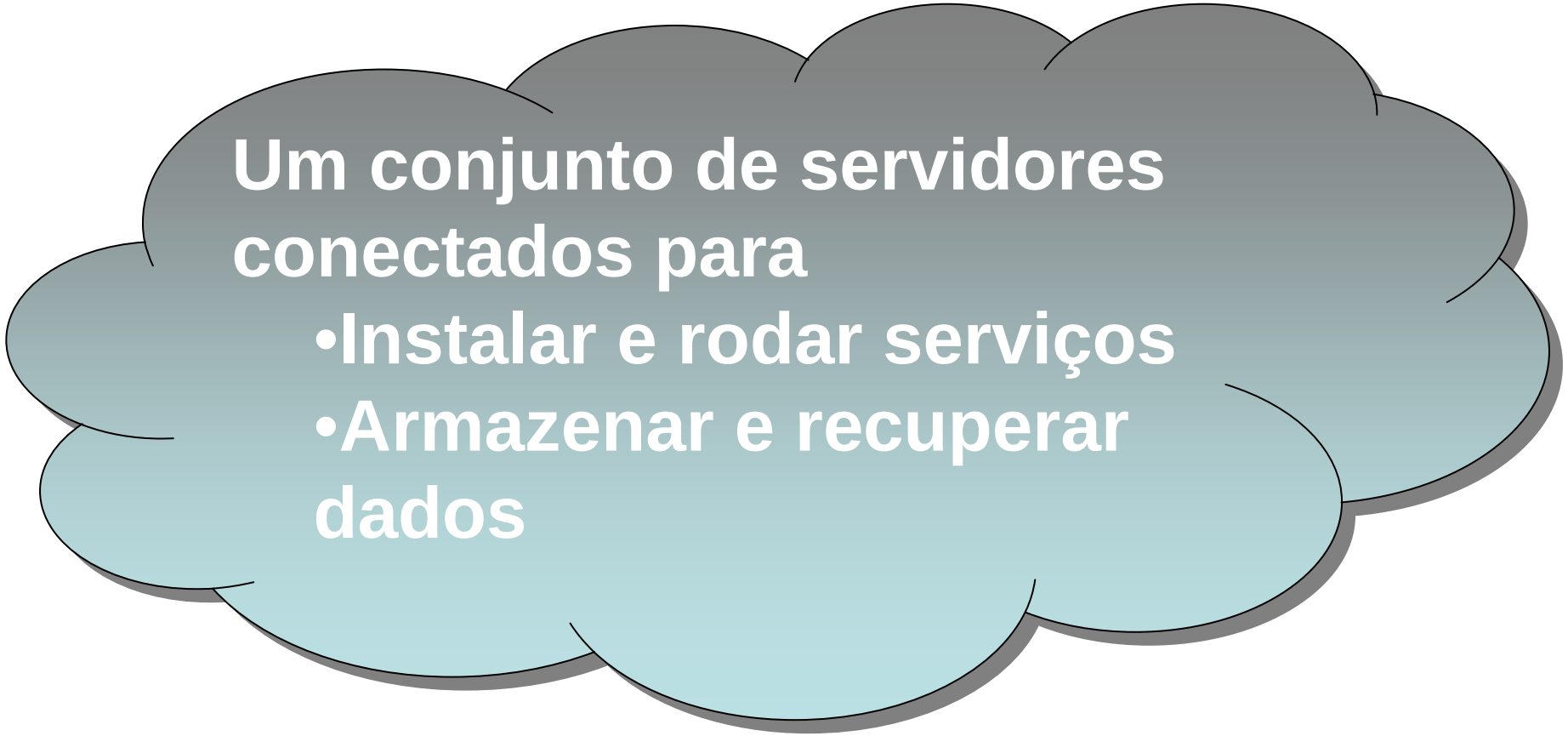
- Introdução
  - O que é Cloud computing?
  - Definições e características essenciais
  - Custos e eficiência energética
- Caracterização dos *data centers* para serviços em nuvem
  - Infraestrutura de um *data center*
  - Limitações das arquiteturas de redes atuais
  - Caracterização do tráfego
  - Requisitos das arquiteturas de rede para novos *data centers*
- Novas arquiteturas para *data centers*
- Conclusão

O que é Cloud Computing?

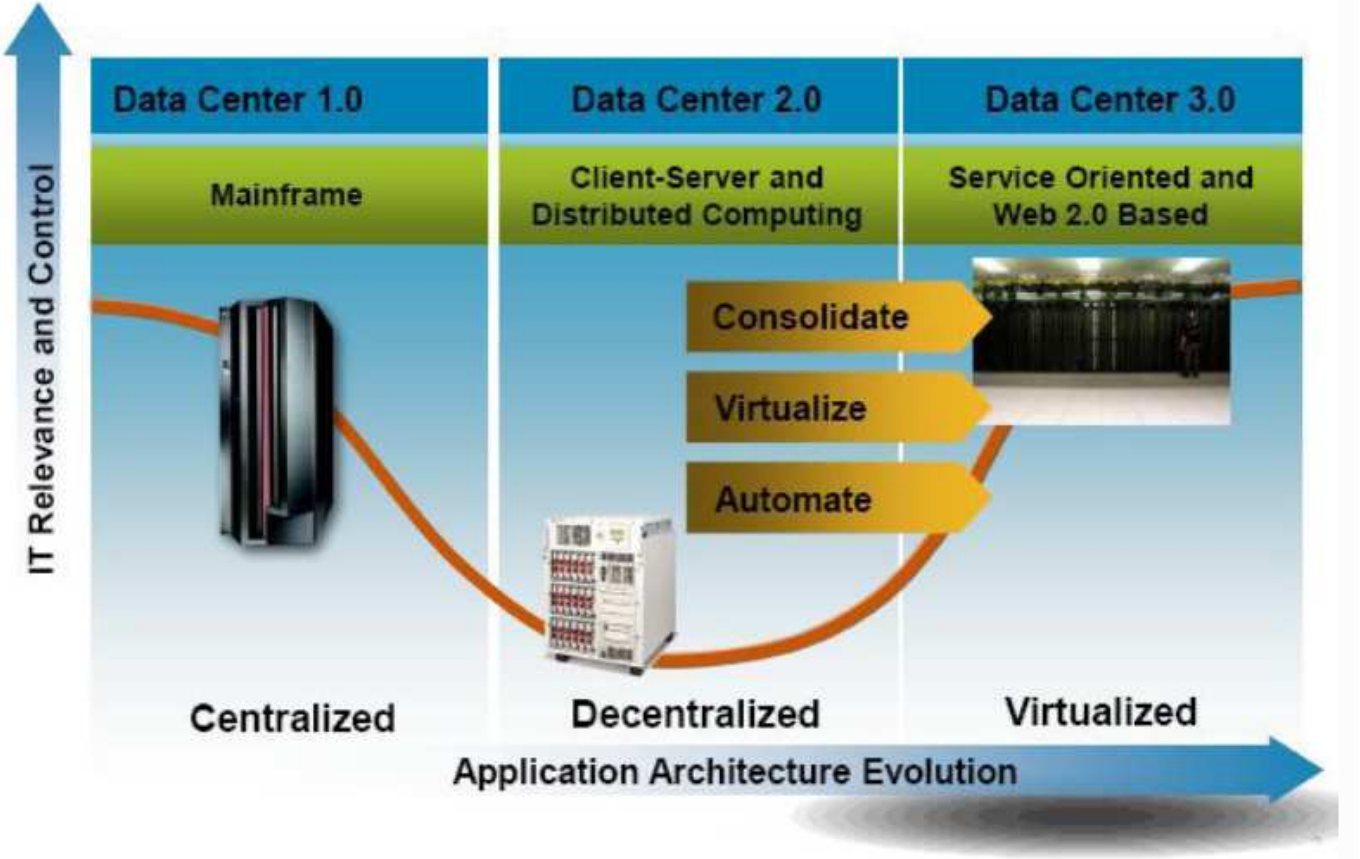


- O que é *cloud computing* [Vaquero et al. 2009]
  - *"Cloud computing é um conjunto de recursos virtuais facilmente usáveis e acessíveis tais como hardware, plataformas de desenvolvimento e serviços. Estes recursos podem ser dinamicamente re-configurados para se ajustarem a uma carga variável, permitindo a otimização do uso dos recursos. Este conjunto de recursos é tipicamente explorado através de um modelo pay-per-use com garantias oferecidas pelo provedor através de acordos de nível de serviço (Service Level Agreements-SLAs)."*

O que é Cloud Computing?

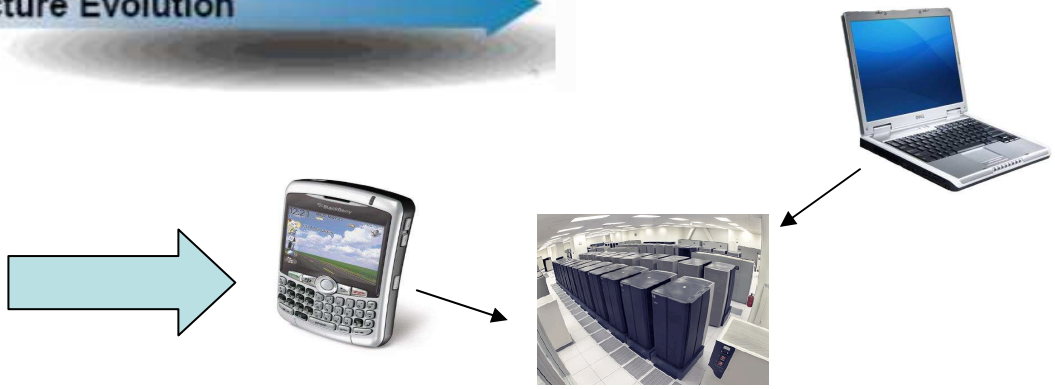


Data center: evolução



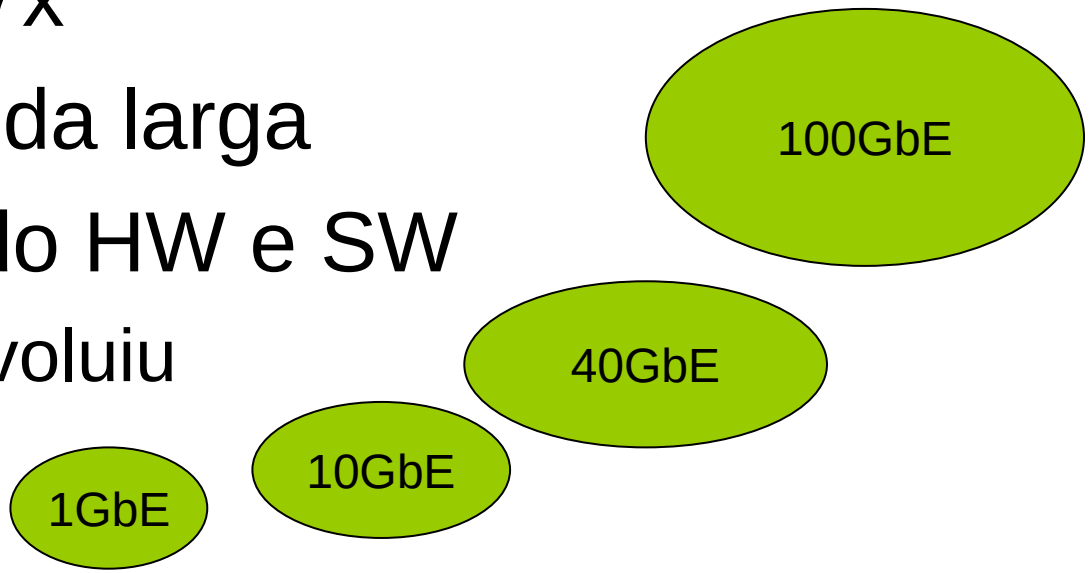
Fonte: Cisco Data Center 3.0

- “Novo” modelo, com a aplicação hospedada remotamente



Cloud computing: porque agora?

- Maior demanda, novos usuários
- Há *expertise* operacional: tolerância a falhas, segurança, serviços web
- Mais barato: 5-7x
- Internet em banda larga
- Barateamento do HW e SW
  - Virtualização evoluiu



Alguns números

- 30 mil servidores
- 350 milhões de usuários registrados. Em breve 500M!
- 80 bilhões de fotos
- Produz 24 TB de dados de logs por dia

myspace.com

- 130 milhões de usuários por mês
- 40% da população dos EUA possuem conta no MySpace
- 300 mil novos usuários por dia
- 3 mil servidores web + 800 servidores de cache

twitter

- 105 milhões de usuários
- 300 mil novos usuários por dia
- 180 milhões de usuários usam o twitter por mês
- 3 bilhões de requisições por dia via sua API
- 37% dos usuários usam seu telefone para postar tweets

Google

- Possui mais de 1 milhão de servidores
- Estima-se que possui 2% do total de servidores no mundo
- Contribui com 6-10% de todo tráfego atual da Internet

Organização do minicurso

- Introdução
  - O que é Cloud computing?
  - Definições e características essenciais
  - Custos e eficiência energética
- Caracterização dos *data centers* para serviços em nuvem
  - Infraestrutura de um *data center*
  - Limitações das arquiteturas de redes atuais
  - Caracterização do tráfego
  - Requisitos das arquiteturas de rede para novos *data centers*
- Novas arquiteturas para *data centers*
- Conclusão

Características essenciais de Cloud computing

- Serviço sob-demanda
  - Funcionalidades fornecidas sem interação humana
- Elasticidade
- Amplo acesso aos serviços
  - Geo-distribuição, acesso ubíquo
- *Resource pooling*
  - Recursos compartilhados

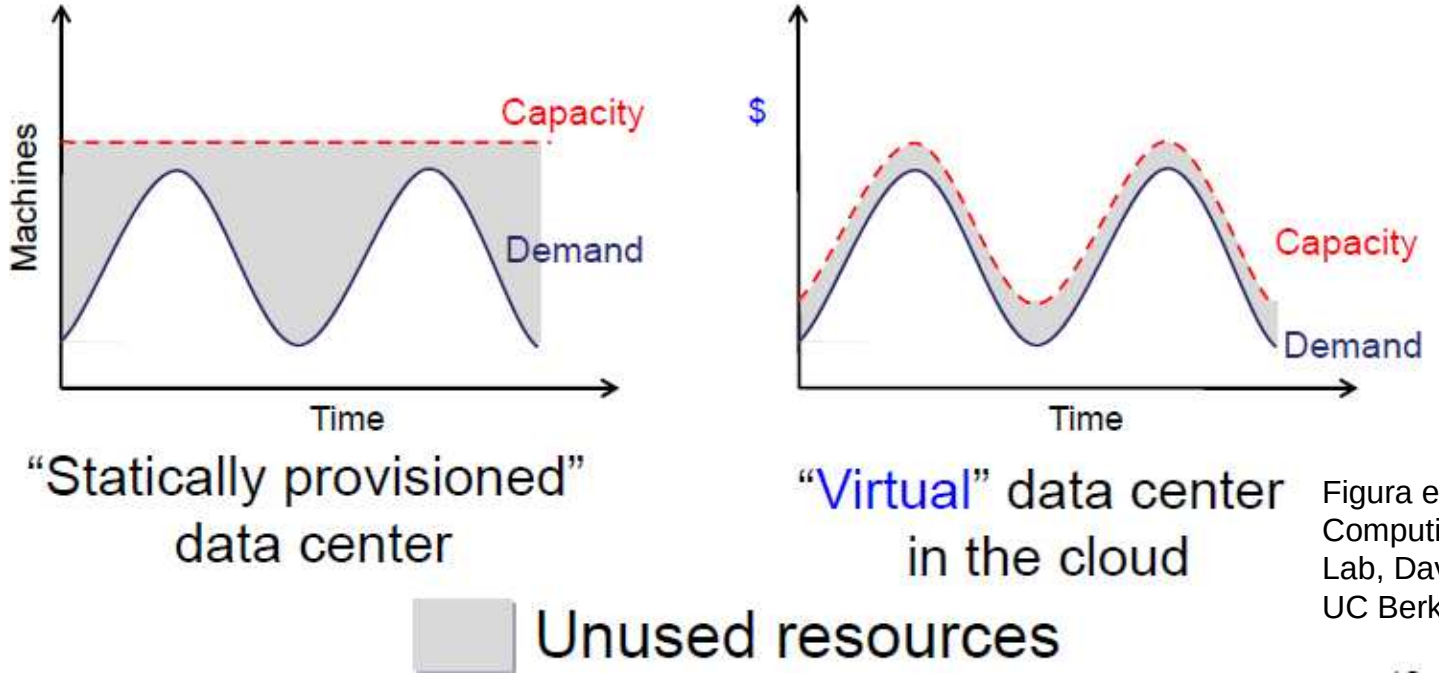
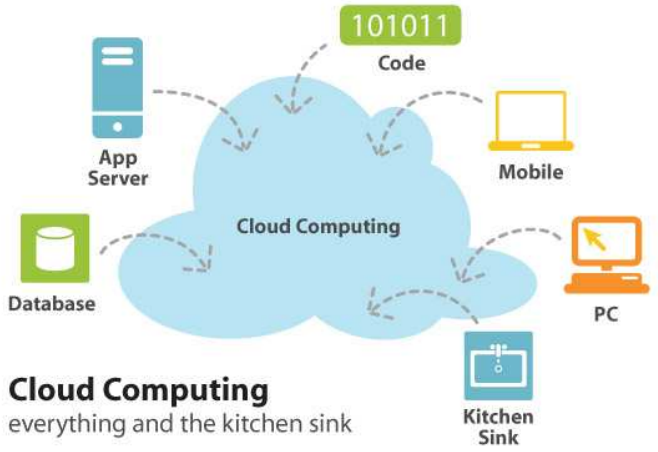


Figura extraída de Cloud Computing and the RAD Lab, David Patterson, UC Berkeley.

Características essenciais (cont.)

- Orientado a serviços
  - SLAs
- Auto-organização
- Medição dos serviços
  - Cobrança baseada no modelo *pay-as-you-go*



Modelos de serviços

- **Software como um serviço (*Software as a Service* - SaaS):** aplicações hospedas na nuvem. O Google Apps e o SalesForce são exemplos de SaaS.

Modelos de serviços

- **Plataforma como um Serviço (*Plataform as a Service* - PaaS):** capacidade oferecida pelo provedor para o usuário desenvolver aplicações que serão executadas e disponibilizadas em nuvem. AppEngine e Microsoft Azure são exemplos de PaaS.

Modelos de serviços

- **Infraestrutura como um Serviço (*Infraestructure as a Service* - IaaS):** é a capacidade que o provedor tem de oferecer uma infraestrutura de processamento e armazenamento de forma transparente. Exemplos de IaaS incluem a Amazon EC2, o GoGrid e o Eucalyptus (*open source*).

Arquitetura da computação em nuvem

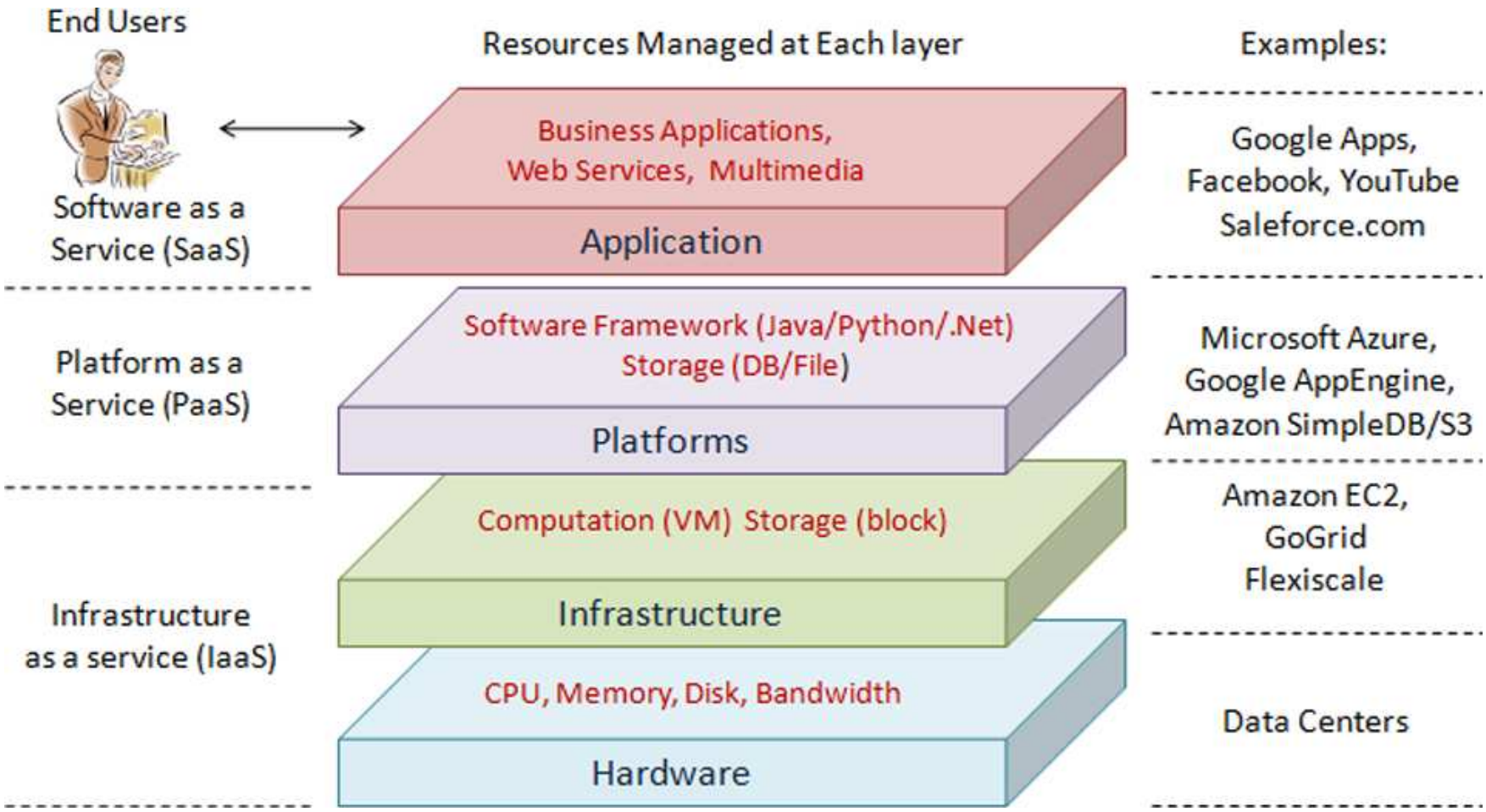
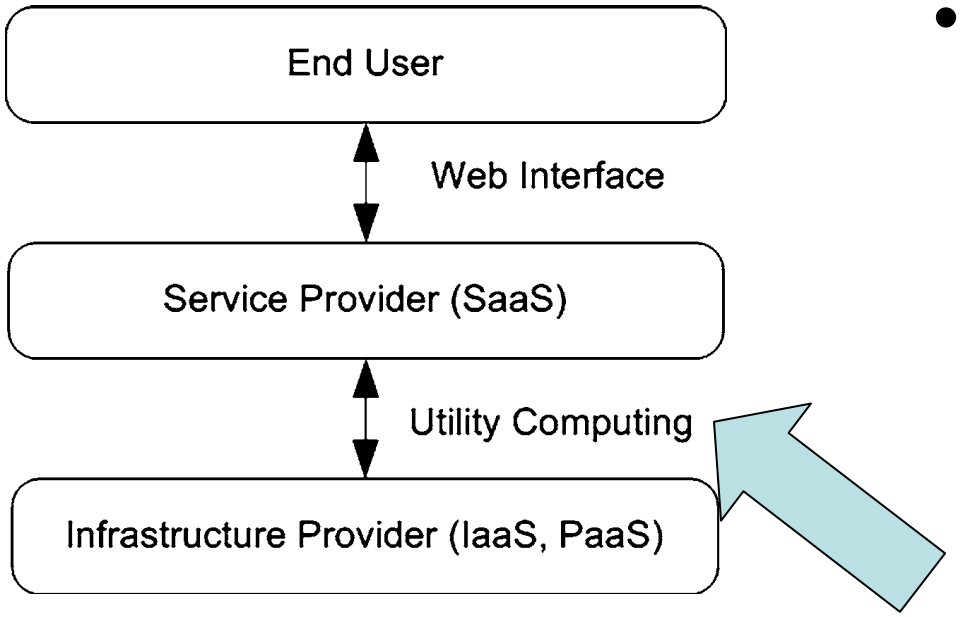


Figura extraída de Cloud computing: state-of-the-art and research challenges
Springer Journal of Internet Services and Applications, April 2010.

Modelo de negócios



- Normalmente o provedor da infraestrutura (IaaS) e o provedor da plataforma (PaaS) pertencem à mesma organização

Figura extraída de Cloud computing: state-of-the-art and research challenges
Springer Journal of Internet Services and Applications, April 2010.

Modelos de implantação

- Nuvem privada (*private clouds*): operada unicamente por uma organização. Os serviços são oferecidos para serem utilizados internamente pela própria organização, não estando disponíveis publicamente para uso geral. Muitas vezes criticadas pois são similares ao modelo já existente de servidores internos;

- Nuvem comunidade (*community cloud*): fornece uma infraestrutura compartilhada por uma comunidade de organizações com interesses em comum;

- Nuvem pública (*public cloud*): a nuvem é disponibilizada publicamente através do modelo *pay-per-use*. Tipicamente, são oferecidas por companhias que possuem grandes capacidades de armazenamento e processamento. Alguns problemas com segurança;

Modelos de implantação (cont.)

- Nuvem híbrida (*hybrid cloud*): a infraestrutura é uma composição de duas ou mais nuvens (privada, comunidade ou pública) que continuam a ser entidades únicas porém, conectadas através de tecnologia proprietária ou padronizada;

- Nuvem privada virtual (*virtual private cloud*): disponibilizada sobre uma nuvem pública através do uso de VPNs. Permite ao usuário criar sua própria topologia, virtualizar servidores e a infraestrutura de rede.

Grades e Computação em Nuvem

- É uma evolução. Muitos conceitos se unem em *cloud computing*
- Algumas diferenças:
  - Modelo de pagamento e origens
    - Grades➔ aplicações científicas
    - Nuvem➔ comercial
  - Compartilhamento de recursos
    - Grades➔ usuário individual pode obter a maioria dos recursos (senão todos)
    - Nuvem➔ usuário individual pode obter uma fração pequena do total de recursos
  - Virtualização
    - Grades➔ simples
    - Nuvem➔ uso intenso, dinâmico, migração

Grades e Computação em Nuvem (cont.)

- É uma evolução. Muitos conceitos se unem em *cloud computing*
- Algumas diferenças:
  - Escalabilidade e gerenciamento
    - Grades➔ aumenta-se o número dos nós
    - Nuvem➔ redimensionamento do hardware virtualizado
  - Padronização
    - Grades➔ madura
    - Nuvem➔ inicial, *intercloud,* migração de máquinas virtuais
  - Aplicações
    - Grades➔ são poucas com muito desempenho
    - Nuvem➔ são muitas, heterogêneas e de longa duração (serviços)

Computação em nuvem e HPC
Um exemplo comparativo

Sistema "Abe" do National Center for Supercomputing Applications

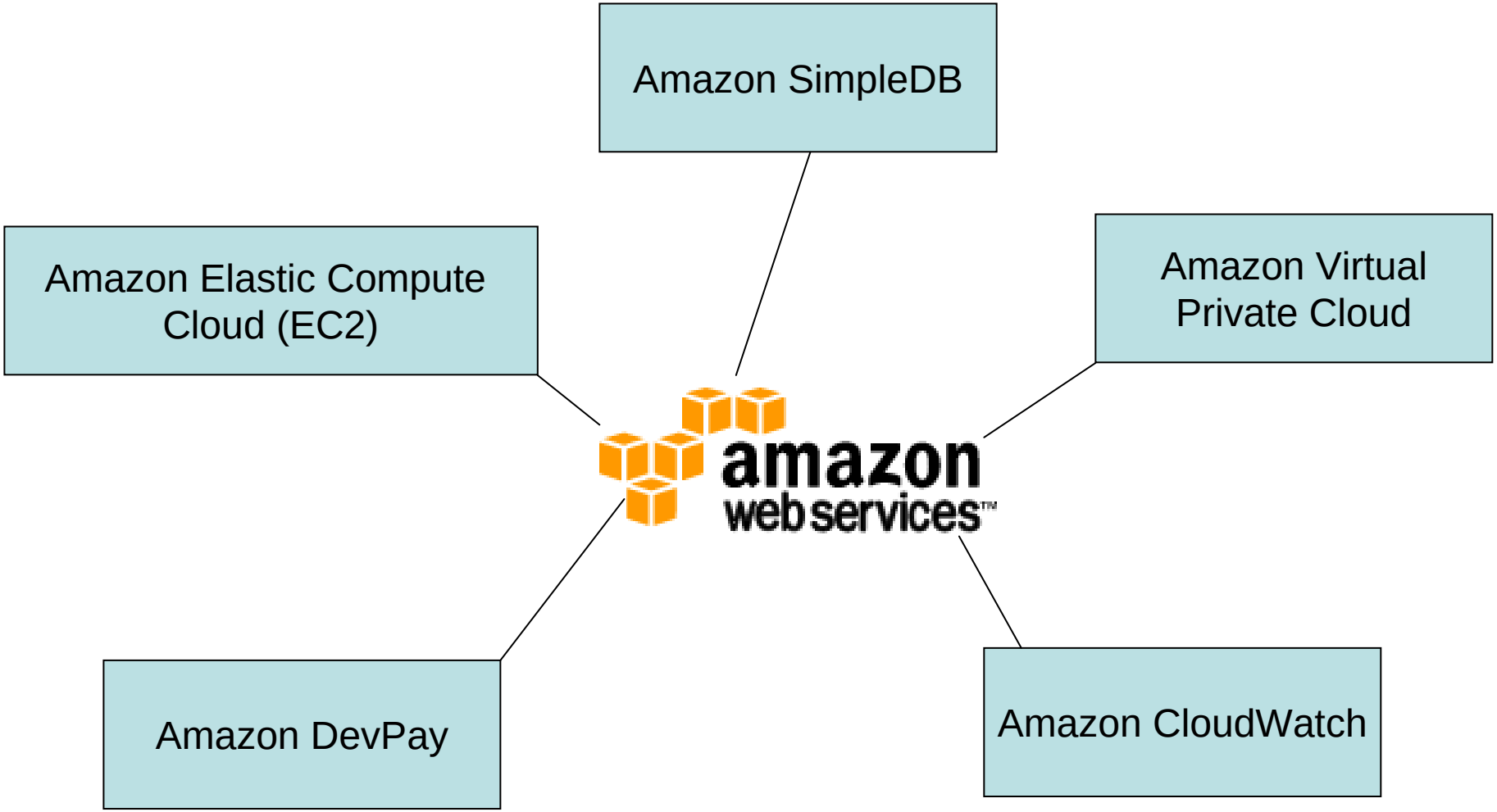
- 32 processadores
- 25 segundos para finalizar a tarefa
- A probabilidade de se obter em 400 segundos 32 processadores utilizáveis por ~20 segundos no supercomputador mencionado é de 34%, uma taxa relativamente baixa

EC2 da Amazon

- 32 imagens
- 100 segundos
- Tempo para obter as 32 imagens: 300 segundos
- Tempo total: 300 + 100 = 400 segundos

Algumas *utility computing*
PaaS + IaaS

Algumas *utility computing*
PaaS + IaaS



EC2 - Amazon

EC2 (IaaS)

- Oferece a maior liberdade de acesso aos recursos
- Oferece ao usuário um conjunto de recursos físicos como se fosse uma máquina real, podendo controlar praticamente todo o software a partir do *kernel*
- Uma instância EC2 é uma máquina virtual rodando sobre o Xen
- Permite criar instâncias em múltiplas localizações

EC2 - Amazon

EC2 (IaaS)

- Simple Storage Service (Amazon S3) para armazenamento
  - Armazena objetos de dados. Cada objeto pode conter de 1 byte a 5 GB
- Amazon Virtual Private Cloud (VPC)
  - Usa VPN para conectar a infraestrutura de TI aos serviços em nuvem
- Amazon SimpleDB
  - Abordagem NoSQL
  - Métodos put/get
- Amazon DevPay
  - Faturamento, pagamento
- Amazon CloudWatch
  - Ferramenta de monitoramento: utilização da CPU, operações de escrita/leitura, consumo de banda de rede, etc.

AppEngine - Google

AppEngine (PaaS)

- Voltada para aplicações web tradicionais
- Suporta as linguagens Java e Python
- O armazenamento utiliza a MegaStore, solução proprietária da Google baseada na BigTable
  - A BigTable é responsável por armazenar dados de mais de 60 produtos da Google, incluindo Google Analytics, Google Finance, Orkut e Google Earth

AppEngine - Google

AppEngine (PaaS)

- Integração com o Google Web Toolkit possuindo um *plug-in* para o Eclipse
- Eclipse permite o desenvolvimento completo de aplicações AJAX
- Gratuito até 5 milhões de visitantes por mês!

Windows Azure - Microsoft

Windows Azure (PaaS)

- Suporta o desenvolvimento de aplicações de objetivo geral
- Sistema operacional Windows Azure
- Serviços para o desenvolvedor que podem ser usados individualmente ou em conjunto:
  - SQL Azure
  - Azure AppFabric

Windows Azure - Microsoft

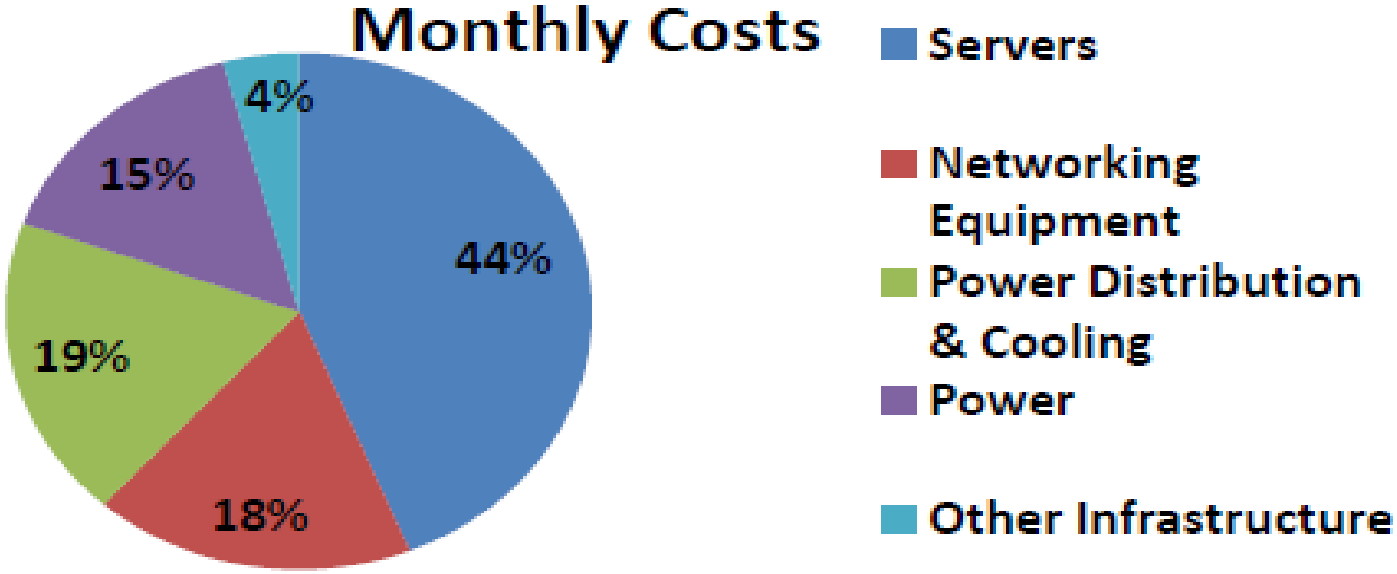
Windows Azure (PaaS)

- Sistema operacional Windows Azure
  - Plataforma para a execução de aplicações Windows e o armazenamento de seus dados na nuvem
  - O armazenamento do Windows Azure não é um sistema relacional, e sua linguagem de consulta não é a SQL
- SQL Azure
  - Banco de dados relacional
  - “Huron” Data Sync
    - sincroniza os dados relacionais dos diversos bancos de dados
- App Fabric
  - Conjunto de serviços que oferecem blocos construtivos para o desenvolvimento de aplicações distribuídas na nuvem
  - App Fabric Labs disponível gratuitamente

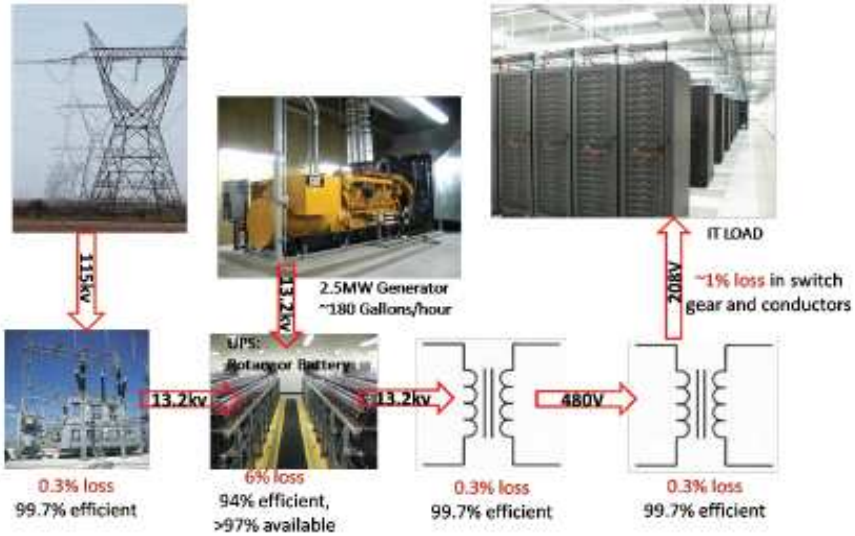
Organização do minicurso

- Introdução
  - O que é Cloud computing?
  - Definições e características essenciais
  - Custos e eficiência energética
- Caracterização dos *data centers* para serviços em nuvem
  - Infraestrutura de um *data center*
  - Limitações das arquiteturas de redes atuais
  - Caracterização do tráfego
  - Requisitos das arquiteturas de rede para novos *data centers*
- Novas arquiteturas para *data centers*
- Conclusão

Distribuição de custos: Custos mensais (50 mil servidores)



62% em TI



3 anos de amortização para os servidores
e 15 anos para infraestrutura

Power distribution & cooling

Distribuição de custos

- Valores altos!!!
  - Atualmente há uma baixa utilização dos servidores: 10%. Ótimo quando atinge 30%!!!
  - Tendência: aumento nos custos com energia e infraestrutura e redução nos custos dos servidores
- Como reduzir os custos?
  - Comoditização
  - Buscar alta utilização dos recursos
    - Agilidade: any server to any service
  - Relaxar os requisitos de tolerância a falhas individuais

Modular routers
- Chassis $20K
- Supervisor card $18K

Ports
- 8 port 10G-X - $25K
- 1GB buffer memory
- Max ~120 ports of 10G per switch

Total price in common configurations: $150-200K (+SW&maint)
Power: ~2-5KW

Integrated Top of Rack switches
- 48 port 1GBase-T
- 2-4 ports 1 or 10G-x
- $7K

Load Balancers
- Spread TCP connections over servers
- $50-$75K each
- Used in pairs

• Um data center com 100K servidores pode custar até $12 milhões/mês
• A maior parte dos recursos no data center passam mais tempo ociosos do que realizando tarefas

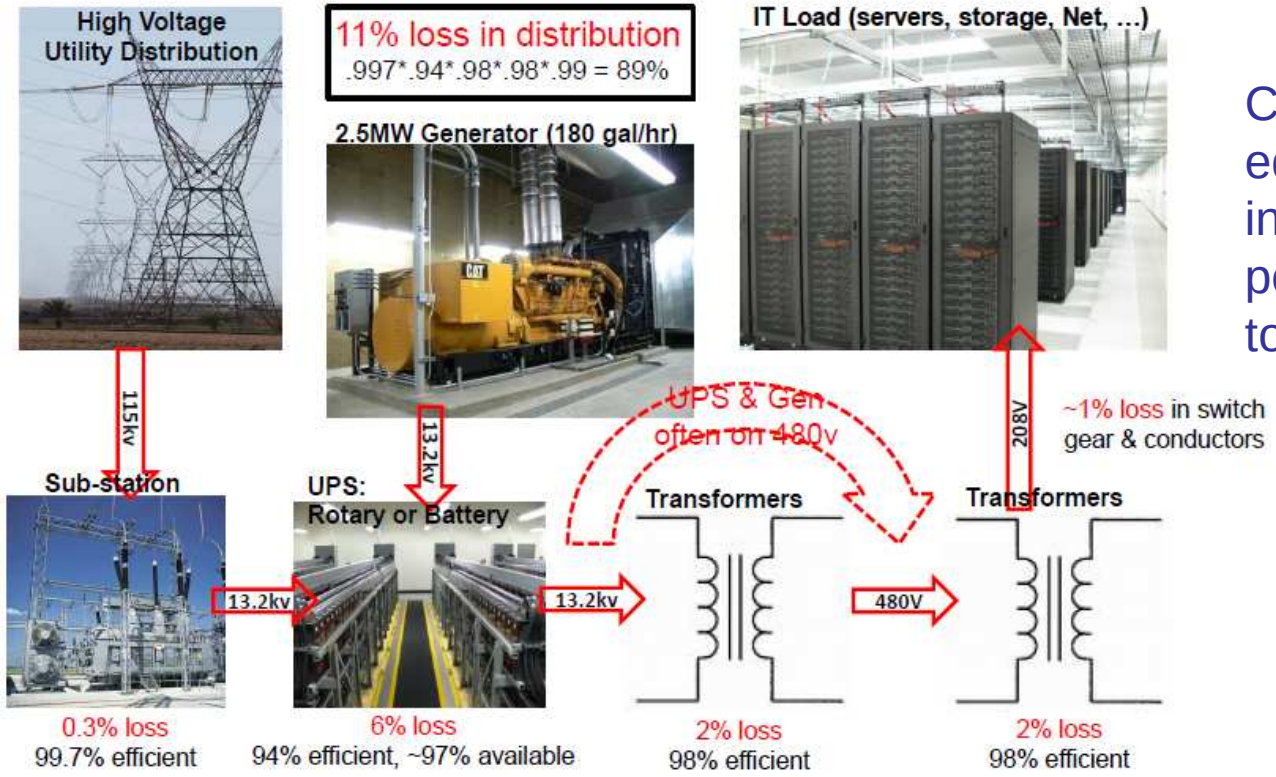
Eficiência energética

- Fonte de grande despesa mensal
  - 34% incluindo o consumo + infraestrutura
- Métrica: Power Usage Effectiveness – PUE
- PUE = Total Facility Power/IT Equipment Power
  - Total Facility Power: energia total do data center➔inclui distribuição e refrigeração
  - IT Equipment Power: equipamento de rede, armazenamento, processamento, etc.
  - Quanto maior o PUE pior é a relação de energia consumida➔PUE ideal é 1

Exemplo: PUE de 3.0➔ indica que a demanda do *data center* é três vezes maior do que a energia necessária para alimentar o equipamento de TI

Eficiência energética

- 85% dos data centers possuem um PUE médio de 3.0
- Em um data center com PUE de ~1.5
  - De cada watt consumido:
    - 67% vai para o equipamento de TI➔ 5.8% pertencem ao equipamento de rede (parte dos 67%)
    - 11% é utilizado na distribuição, conversão
    - 22% é utilizado na refrigeração



Consumo de energia dos equipamentos de rede é ineficiente individualmente, porém não é grande no todo

Custos + eficiência energética

- Custos com servidores dominam
- Equipamento de rede representa
  - 18% do custo mensal
  - 5.8% da energia consumida
- Maximizar o uso dos servidores:
  - Desligar os servidores aumenta o PUE
  - O melhor PUE é atingido quando todos os servidores no *data center* estão funcionando perto da capacidade máxima
- Agilidade/elasticidade
- Comoditização: scale out ao invés de scale up

- 2/3 da energia total do data center é gasta com servidores
- Metade da energia fornecida é consumida fazendo nada
- Evitar conversões (menos transformadores)
- Comprar energia em grandes volumes
- *Energy proportionality:* N% de carga deveria consumir N% de energia
- Há espaço para melhora nos mecanismos de refrigeração e redução de perdas
- Custos de refrigeração estão 100% relacionados com a dissipação de calor do data center
- Alta voltagem perto do consumo
- Usar ar de fora ao invés de AC

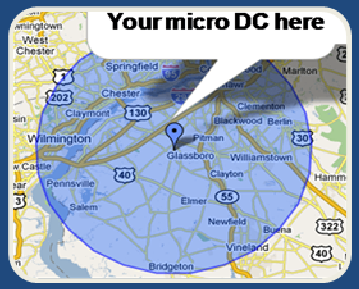
Organização do minicurso

- Introdução
  - O que é Cloud computing?
  - Definições e características essenciais
  - Custos e eficiência energética
- Caracterização dos *data centers* para serviços em nuvem
  - Infraestrutura de um *data center*
  - Limitações das arquiteturas de redes atuais
  - Caracterização do tráfego
  - Requisitos das arquiteturas de rede para novos *data centers*
- Novas arquiteturas para *data centers*
- Conclusão

Tipos de data centers

Macro Data Center

- Specially dedicated facilities
- 100.000 or more servers and 10s of Mega-Watts of power at peak
- Computation in the cloud
  (e.g., Amazon EC2, Windows Azure, Google AppEngine)



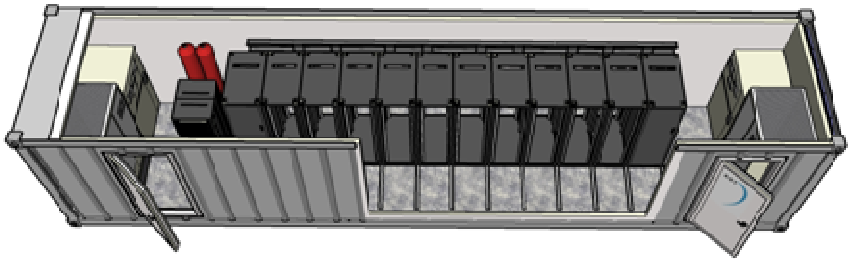
Micro Data Center

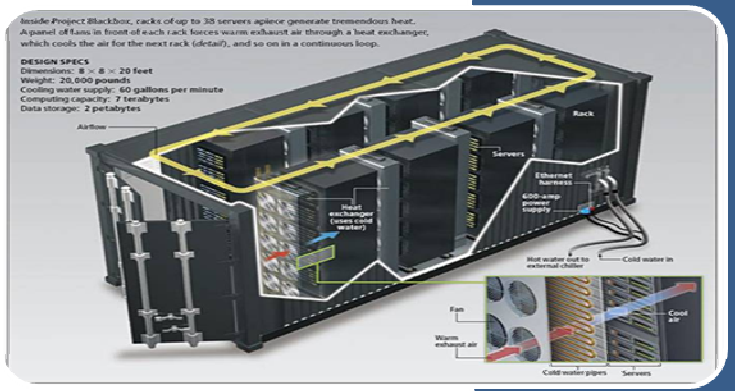
- Geo-diverse placed close to major population centers (e.g. CDN nodes)
- 1000s of servers and 100s of kilowatts
- Higher degree of independence between physical data center outages
- Opportunity to economically reach data center customers with low latency (e.g., front-end cloud apps)

Nano Data Center

- Located in the customer premises equipment (e.g., set-top-box)
- "Why don't we try to take the functionality that we have now in the data center, and distribute it across hundreds of thousands of set top boxes so that we have these 'Nano Data Centers" [EU FP7 NADA]
- P2P-like resource management. Low latency. Low cost.

Data center in a box





Container-based modular DC

- Efficient way to deliver computing and storage services
- 1000-2000 servers in a single container
- Sun Project Black Box (242 systems in 20')

Rackable Systems Container
2800 servers in 40'

Core benefits:

- Easy deployment
  - High mobility
  - Just plug in power, network, & chilled water
- Increased cooling efficiency
- Manufacturing & H/W Admin. Savings
- Push modularity throughout the DC

IEEE Spectrum Feb.

Noção de escala dos data centers

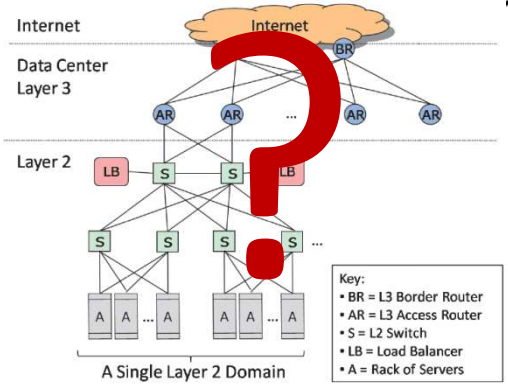
Extensão dos data centers:

10 x

50.000 – 200.000 servidores

100s serviços

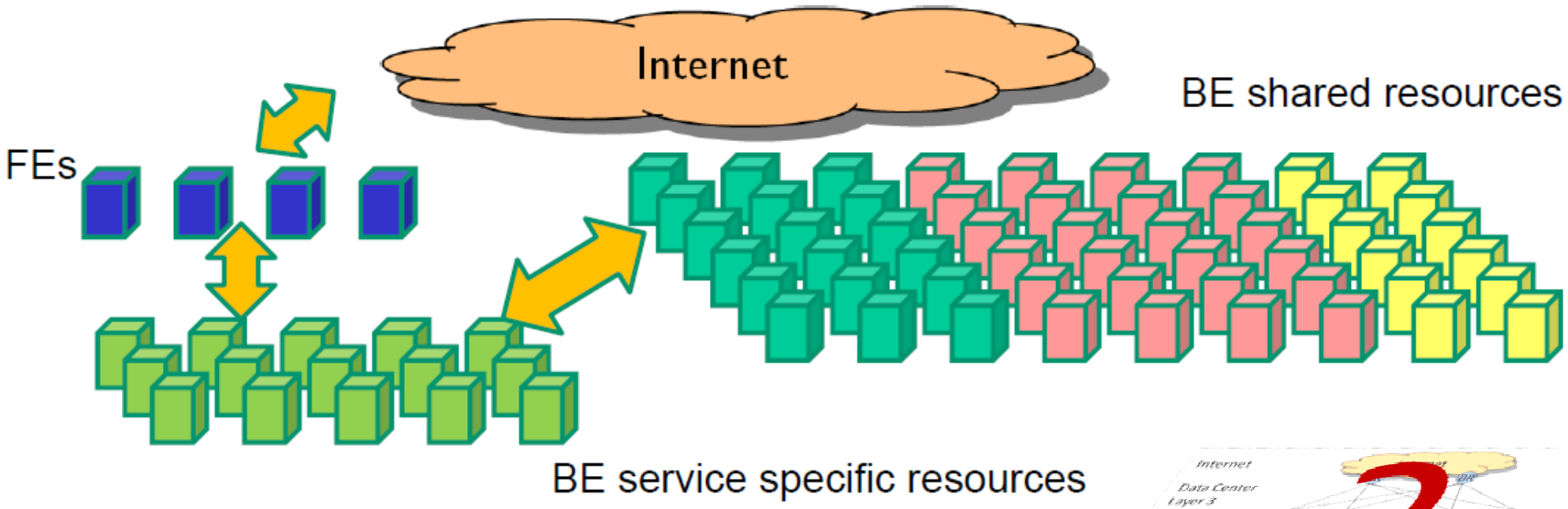
Big picture



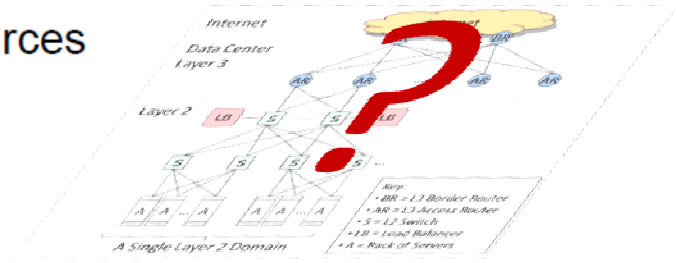
Source: Albert Greenberg, ICDCS 2009

Big Picture

- Componentes das aplicações em nuvem:
  - Front Ends (FE) localizados em data centers satélites ou em *Content Distribution Networks*
  - Back Ends (BE) localizados em grandes data centers servindo recursos específicos do serviço e outros recursos compartilhados

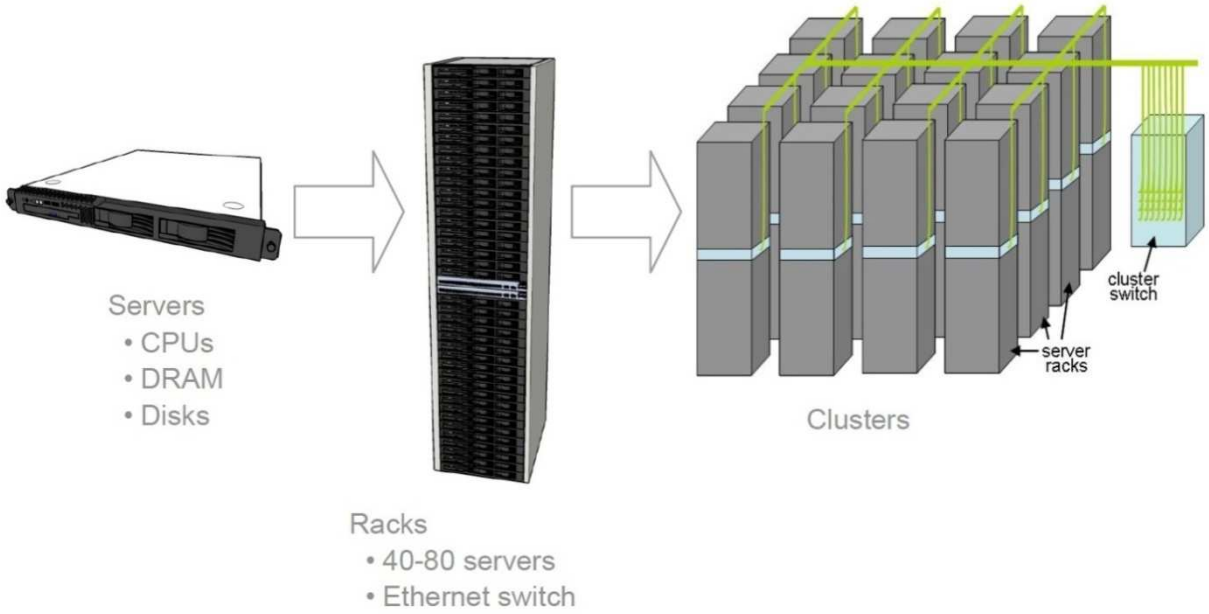


Complexidade! Muitos componentes: balanceadores de carga, DNS, BGP, middleware, servidores, racks, falhas de SW e HW.

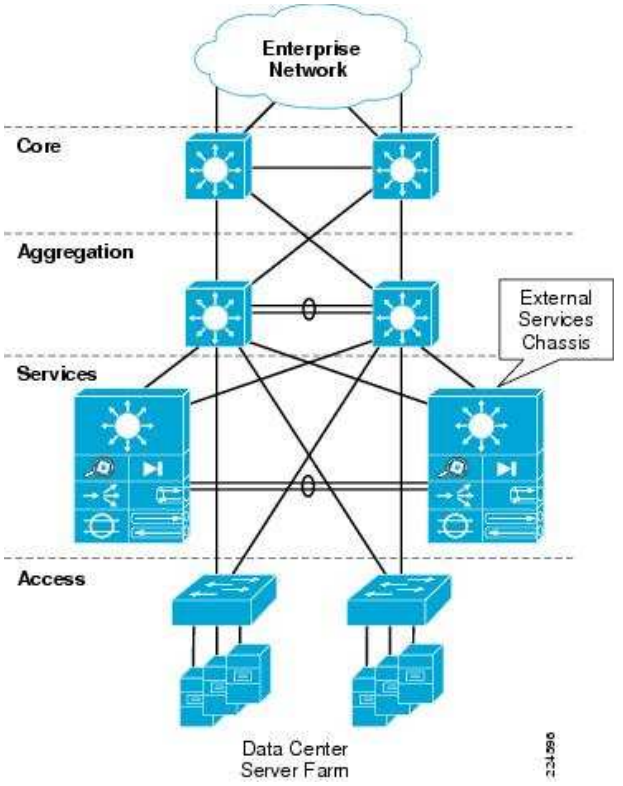


Componentes do data center

- Servidores, racks, clusters, e topologia em camadas

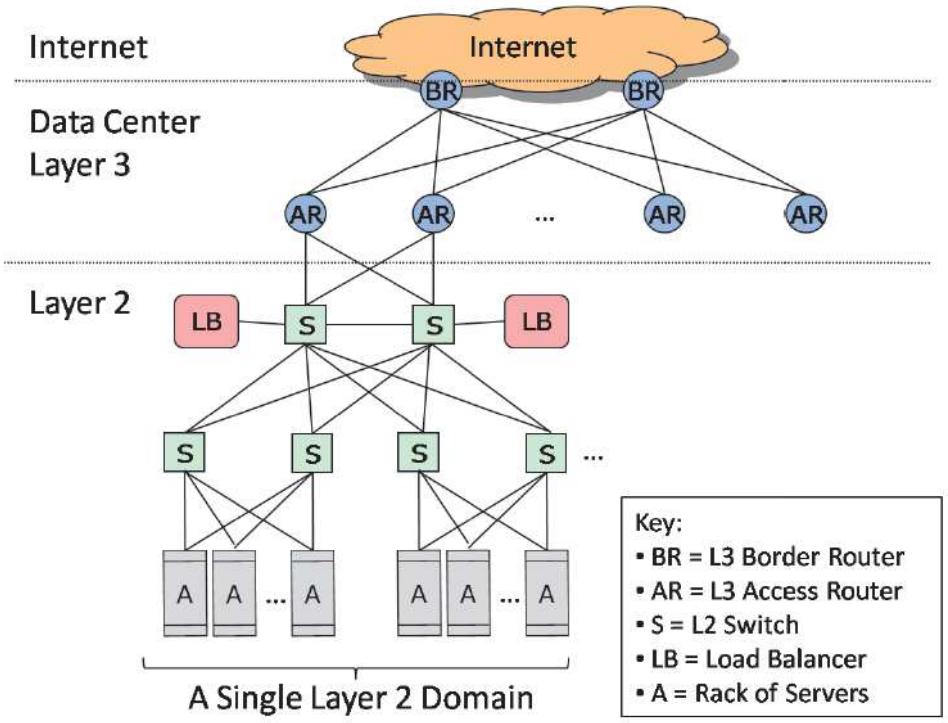


Fonte: Luiz André Barroso and Urs Hölzle, The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines



Fonte: Cisco Data Center Design Guide

Abordagem tradicional



~ 4,000 servers/pod

Tipicamente 20 a 40 servidores / Top of Rack (ToR) switch com 1 Gbps.

- Estrutura hierárquica (IP subnets + fragment. VLANs)
- Balanceadores de carga (LB): Direct_IP em Virtual_IP
- Redundância 1+1

Desenho *scale-up:*

- Equipamento no nível alto da herarquia carrega mais tráfego, maior funcionalidade e disponibilidade - mais caro.

E ainda assim um fraco desempenho de rede e problemas de escala….

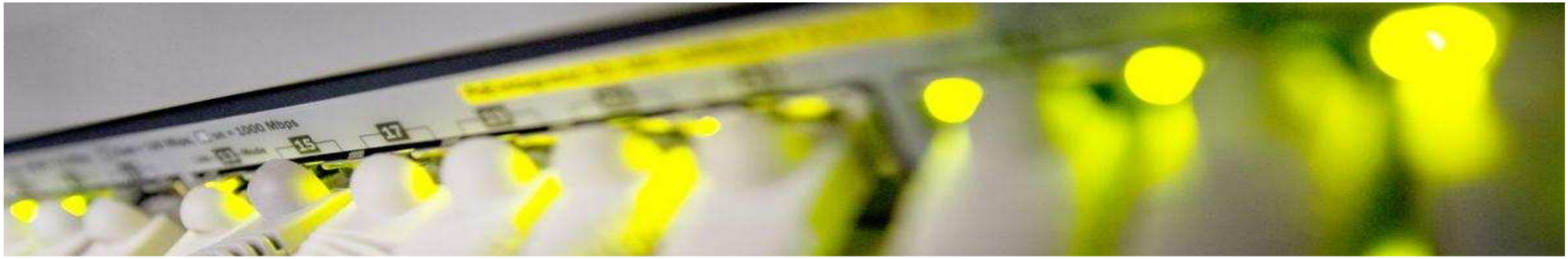
Organização do minicurso

- Introdução
  - O que é Cloud computing?
  - Definições e características essenciais
  - Custos e eficiência energética
- Caracterização dos *data centers* para serviços em nuvem
  - Infraestrutura de um *data center*
  - Limitações das arquiteturas de redes atuais
  - Caracterização do tráfego
  - Requisitos das arquiteturas de rede para novos *data centers*
- Novas arquiteturas para *data centers*
- Conclusão

Alguns problemas com os projetos convencionais de redes de data center

Limitações de rede considerando a organização hierárquica tradicional L2/L3:

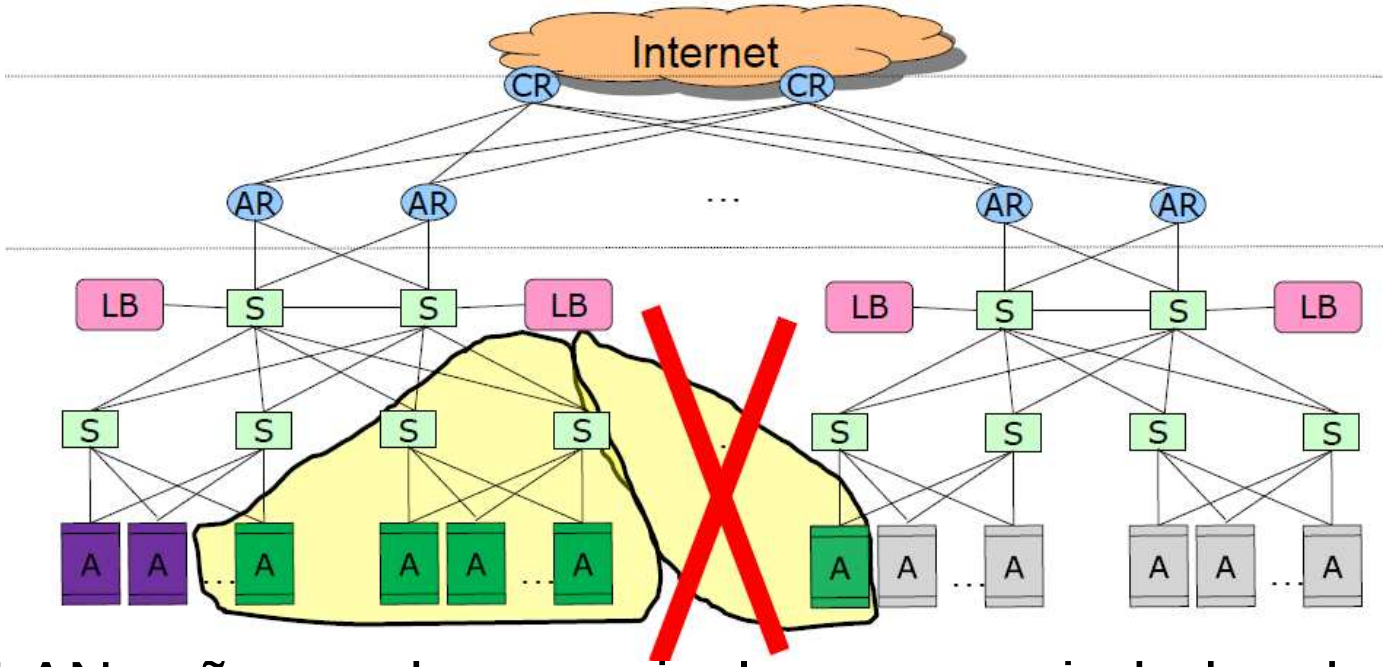
- Fragmentação de recursos (VLAN, subnetting)
- Capacidade limitada de conexão entre servidores (high oversubscription)
- Escalabilidade Ethernet (tamanho da FIB, STP, flooding, ARP)
- Baixo desempenho para o tráfego de aplicações em nuvem
- Confiabilidade: 1+1 é um valor baixo de redundância



Agilidade: Any server, any service

- Transformar os servidores em um único pool de recursos
  - Permitir que os serviços “respirem”: expandir e contrair o uso dos recursos conforme necessário
- Requisitos para implementar a agilidade
  - Prover mecanismos para instalar rapidamente novos códigos em um servidor
    - *Máquinas virtuais, imagens de SOs*
  - Prover mecanismos para que o servidor acesse dados persistentes
    - *Sistemas de arquivos distribuídos (e.g., blob stores)*
  - Prover mecanismos para comunicação eficiente entre servidores, independente de onde eles estejam localizados no data center
    - *Network : ?*

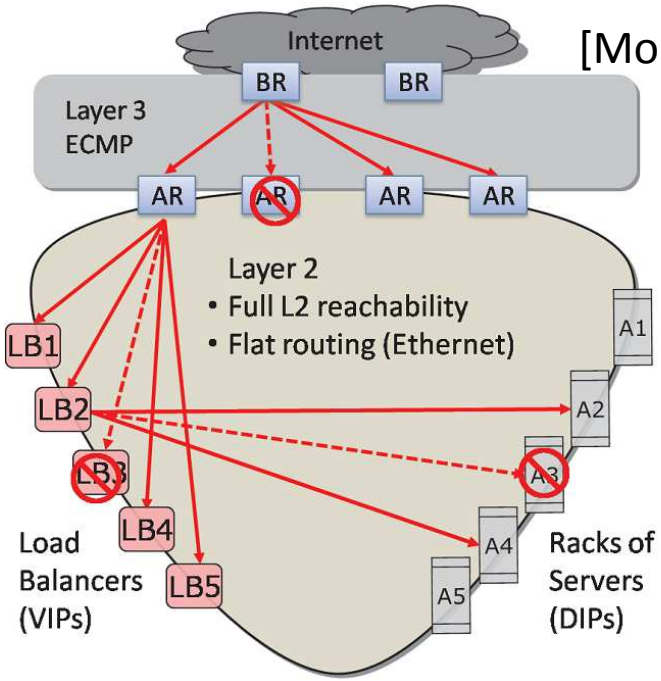
Fragmentação interna dificulta a alocação e liberação de recursos dinâmicamente



- VLANs são usadas para isolar as propriedades das aplicações
- Endereços IP topologicamente determinados pelos ARs
- Reconfiguração de IPs e VLAN são difíceis, manuais e sensíveis a erros humanos

Source: Albert Greenberg, ICDCS 2009

Arquiteturas de camada 2 (Ethernet) não são escaláveis



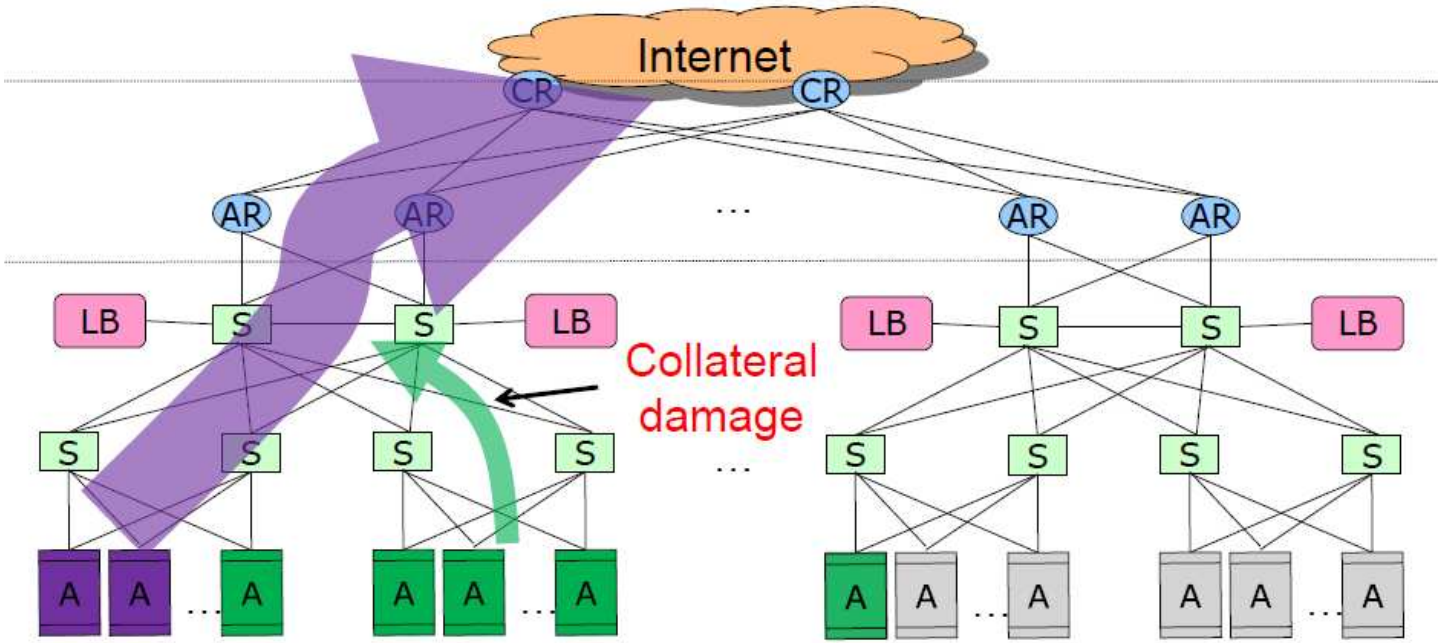
[Monsoon]

Key:
- CR = L3 Core Router
- AR = L3 Access Router
- S = L2 Switch
- LB = Load Balancer
- A = Rack of 20 servers with Top of Rack switch

- Tamanho da tabela de encaminhamento nos switches é limitado (Mac-based flat routing)
- *Broadcast* do ARP e *flooding de pacotes* limitam as VLANs a centenas de hosts
- *Limitações de desempenho* (bisection BW), i.e., protocolos padrões de spanning tree dificultam os mecanismos de tolerância a falhas e encaminhamento multipath

Source: Albert Greenberg, ICDCS 2009

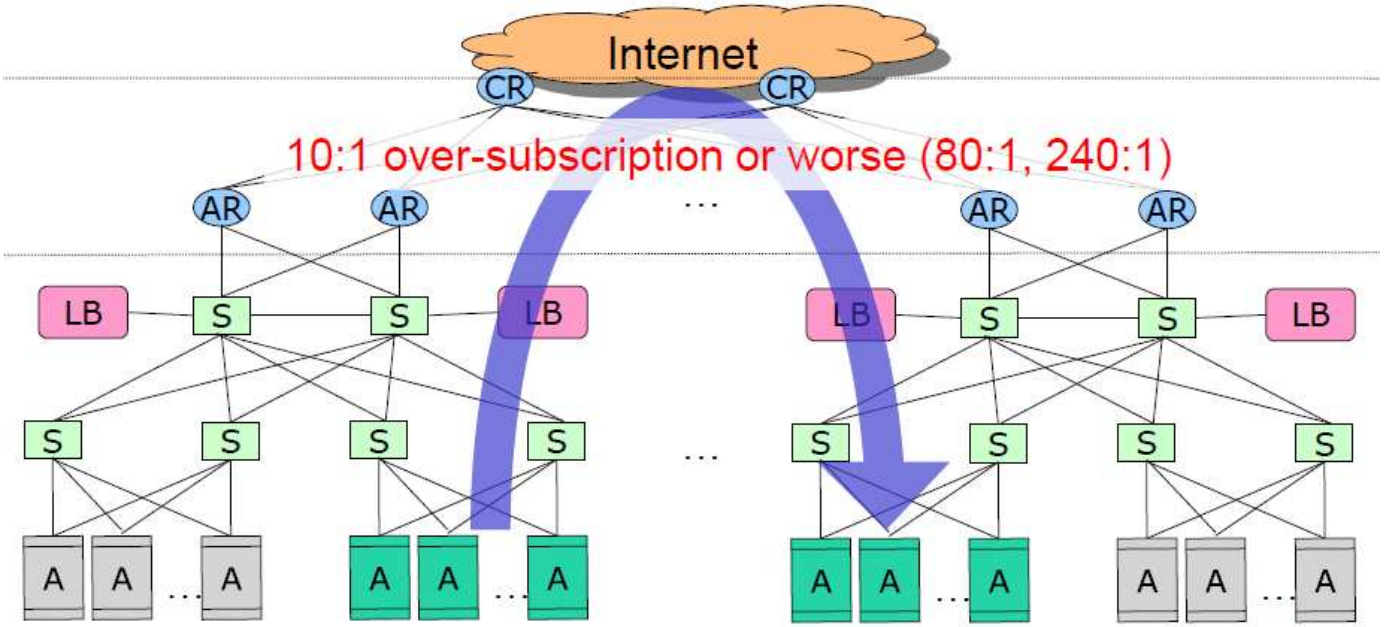
Não há isolamento de desempenho



- VLANs tipicamente fornecem somente isolamento de alcançabilidade
- Um serviço enviando ou recebendo muito tráfego pode prejudicar outros serviços na mesma sub-rede

Source: Albert Greenberg, ICDCS 2009

A rede precisa de muita Bisection BW e engenharia de tráfego para usar o que estiver disponível

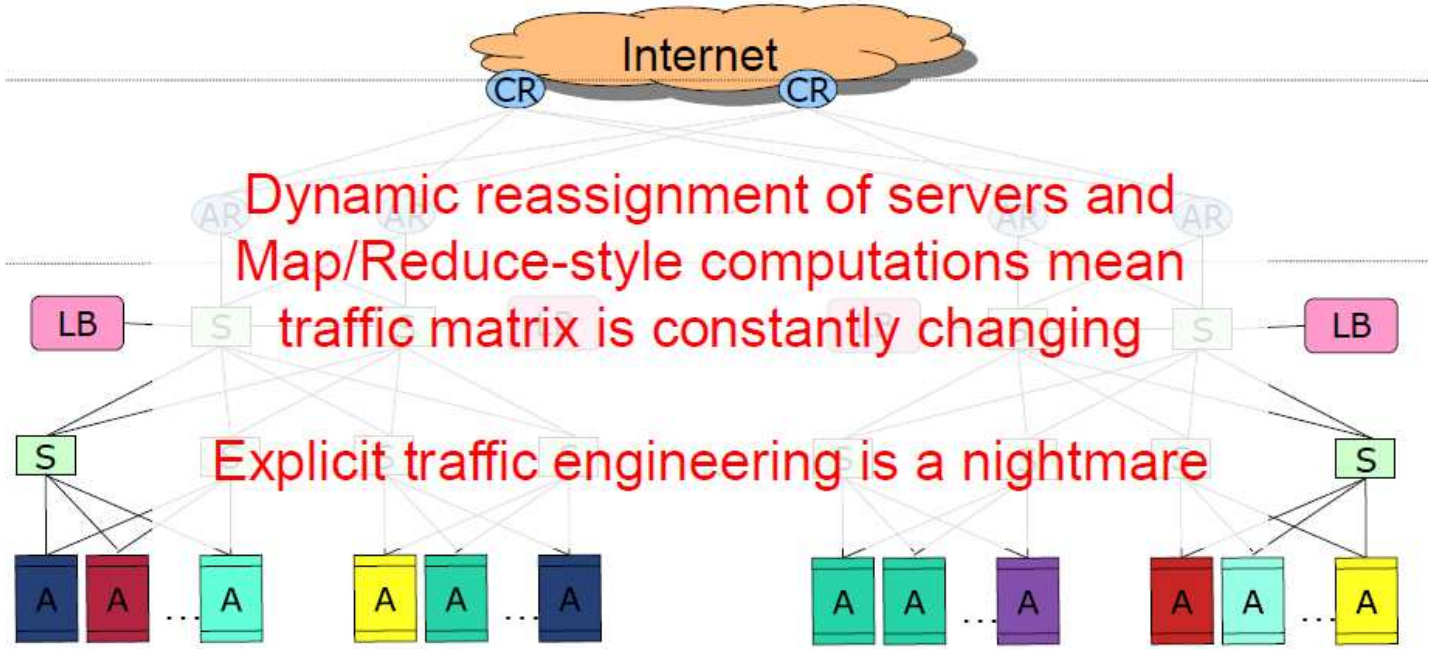


Os data centers executam dois tipos de aplicações:

- Aplicações externas (servir páginas web para os usuários)
- Computação interna (computar indexação de documentos, searching, HPC)

Source: Albert Greenberg, ICDCS 2009

A rede precisa de muita Bisection BW e engenharia de tráfego para usar o que estiver disponível



Os data centers executam dois tipos de aplicações:

- Aplicações externas (servir páginas web para os usuários)
- Computação interna (computar indexação de documentos, searching, HPC)

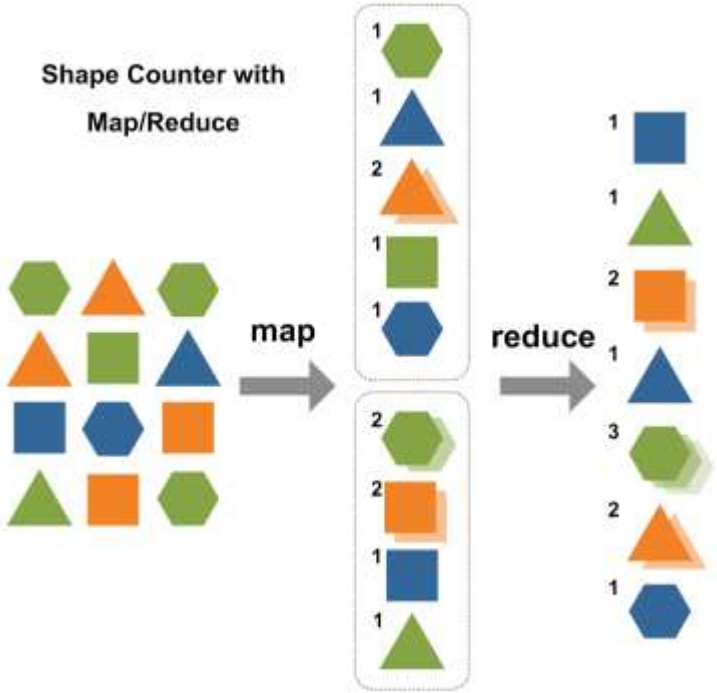
Source: Albert Greenberg, ICDCS 2009

Organização do minicurso

- Introdução
  - O que é Cloud computing?
  - Definições e características essenciais
  - Custos e eficiência energética
- Caracterização dos *data centers* para serviços em nuvem
  - Infraestrutura de um *data center*
  - Limitações das arquiteturas de redes atuais
  - Caracterização do tráfego
  - Requisitos das arquiteturas de rede para novos *data centers*
- Novas arquiteturas para *data centers*
- Conclusão

MapReduce

Large Scale Data Processing



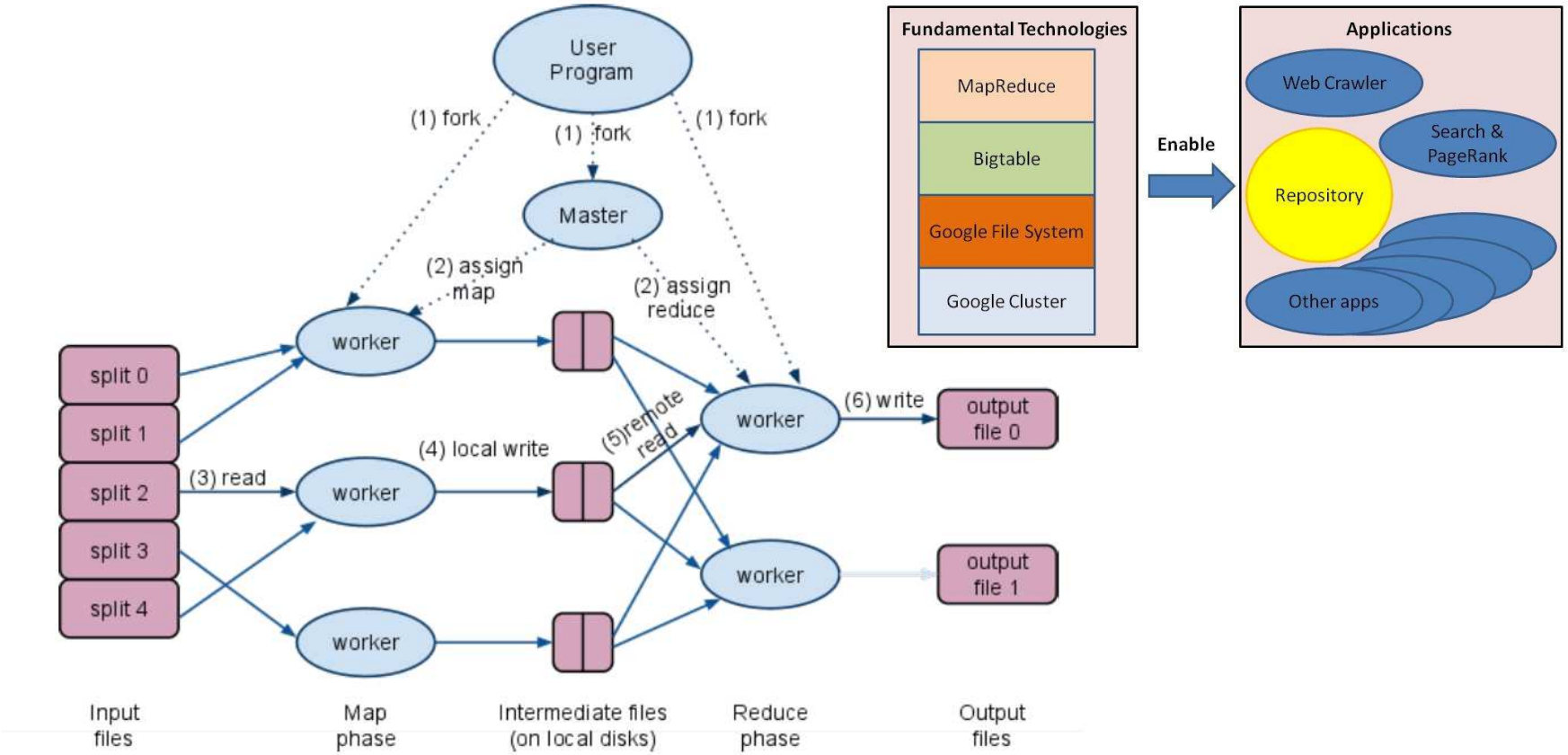
Example applications

- distributed grep
- distributed sort
- web link-graph reversal
- term-vector per host
- web access log stats
- inverted index construction
- document clustering
- machine learning
- statistical machine translation
- …

- 6000 MR apps at Google

Tutorial:
http://static.googleusercontent.com/external_content/untrusted_dlcp/research.google.com/en//archive/papers/mapreduce-sigmetrics09-tutorial.pdf

MapReduce traffic is M:N



- On-line: A typical Google search involves 700 to 1,000 machines
- Off-line: Six hours and two minutes to sort 1PB

(10 trillion 100-byte records) on 4,000 computers.

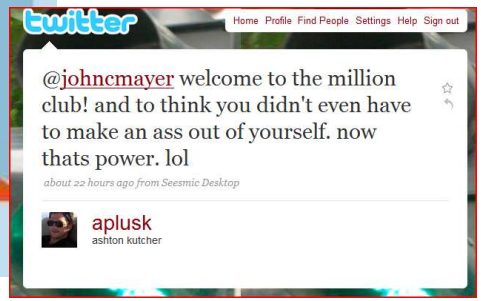
Online Social App traffic is 1:N

Hard to scale:

- *All data is active* all the time.
- Hard to *partition data* because everyone is connected (social graph).
- Everything must be kept in *RAM* cache for fast data access.
- Compare with traditional Web apps (E-Mail, content)
  - Easy data partition
  - Only a % of online users

- Gather the status of all 200 of your friends at the same time
- Plus trigger third-party services
- Large-scale apps struggle with high latency:
  - Facebook: can only make 100-150 internal requests per page

Online Social Networks scale bad



Example: Why Twitter's Engineers Hate the @replies feature

References:

Pujol *et al.* "The Little Engine(s) that Could: Scaling Online Social Networks“ In SIGCOMM’10.

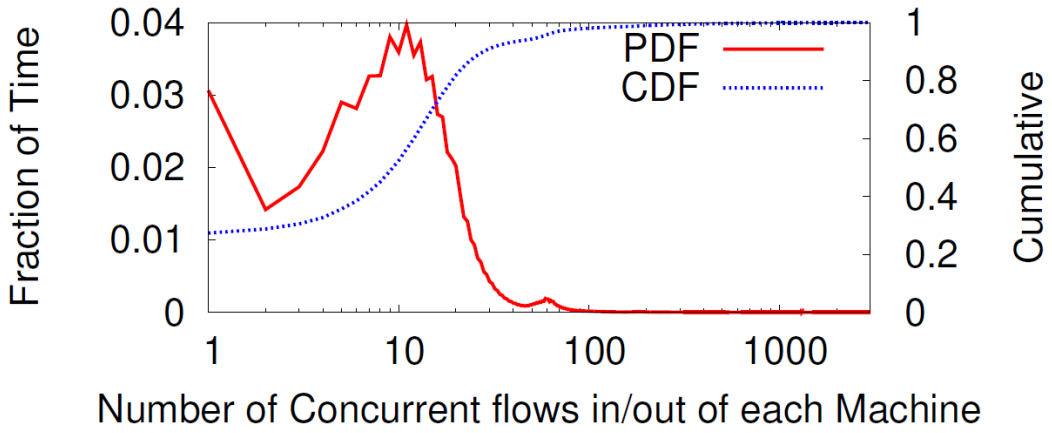
“Scaling Online Social Networks without Pains”

http://highscalability.com/blog/2009/10/13/why-are-facebook-digg-and-twitter-so-hard-to-scale.html

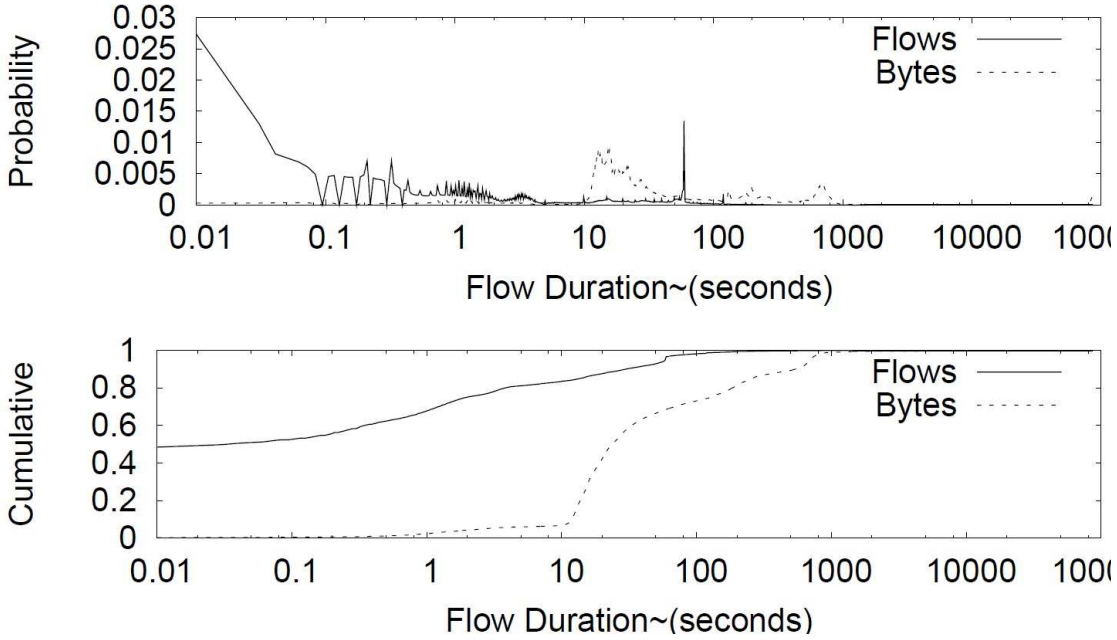
http://www.25hoursaday.com/weblog/2009/05/15/WhyTwittersEngineersHateTheRepliesFeature.aspx

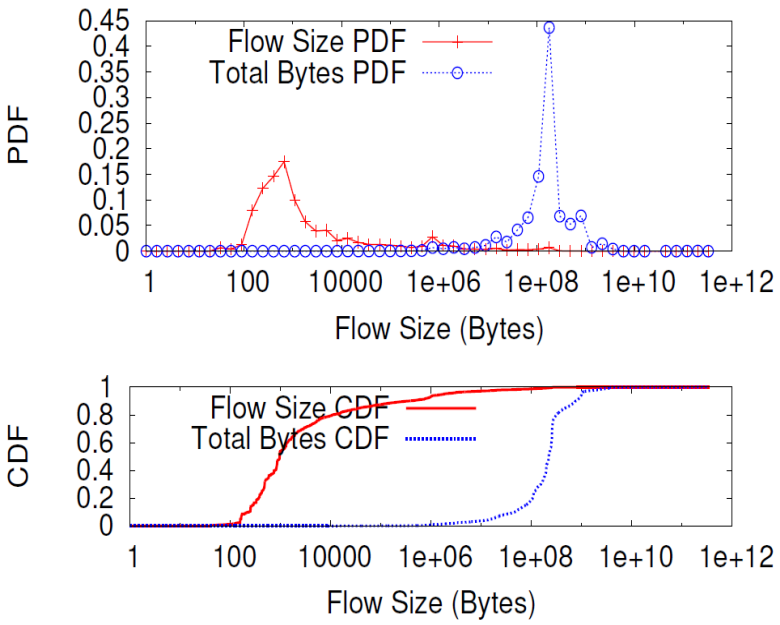
Perfil do tráfego

- 80% dos pacotes ficam dentro do data center
  - Data mining, computações de indexação, back end to front end
  - A tendência é aumentar ainda mais as comunicações internas
- Tráfego do DC != tráfego da Internet
- Média de 10 fluxos ao mesmo tempo por servidor [VL2]



Caracterização do tráfego

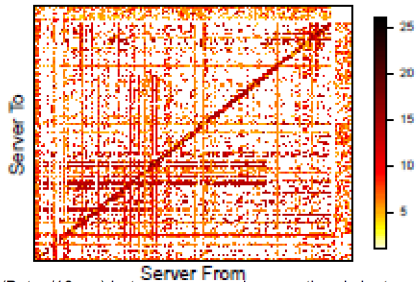




A maioria dos fluxos são “ratões”

- 80% duram menos de 10s
- poucos de longa duração (menos de 0.1% duram mais de 200s).
- mais de 50% dos bytes estão em fluxos que duram menos de 25 segundos.

Padrões de tráfego



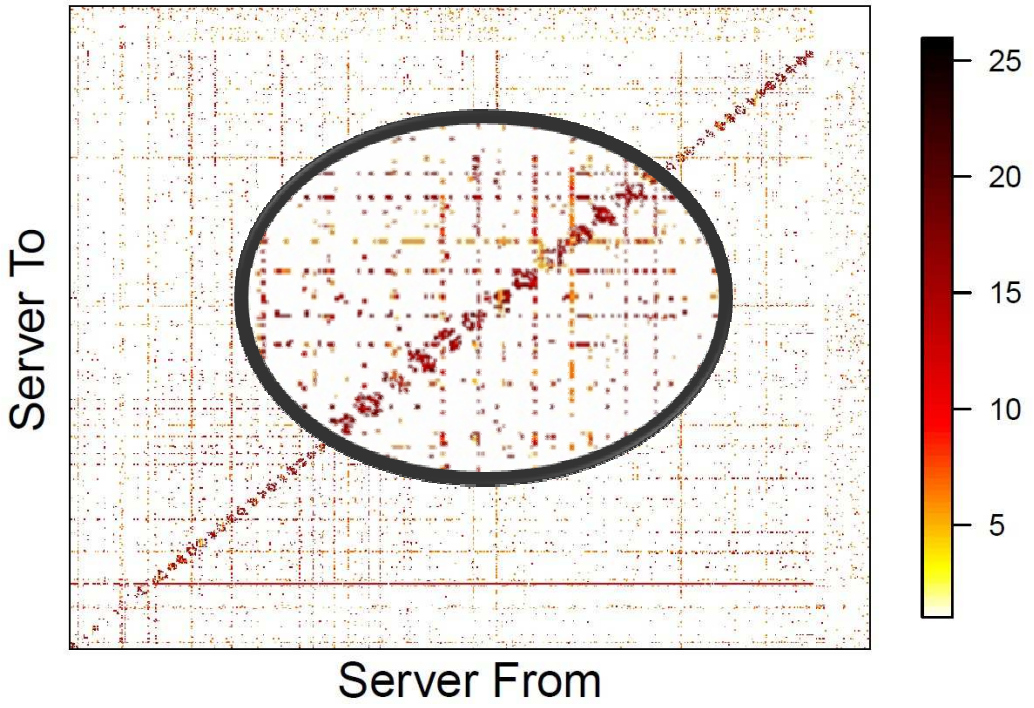
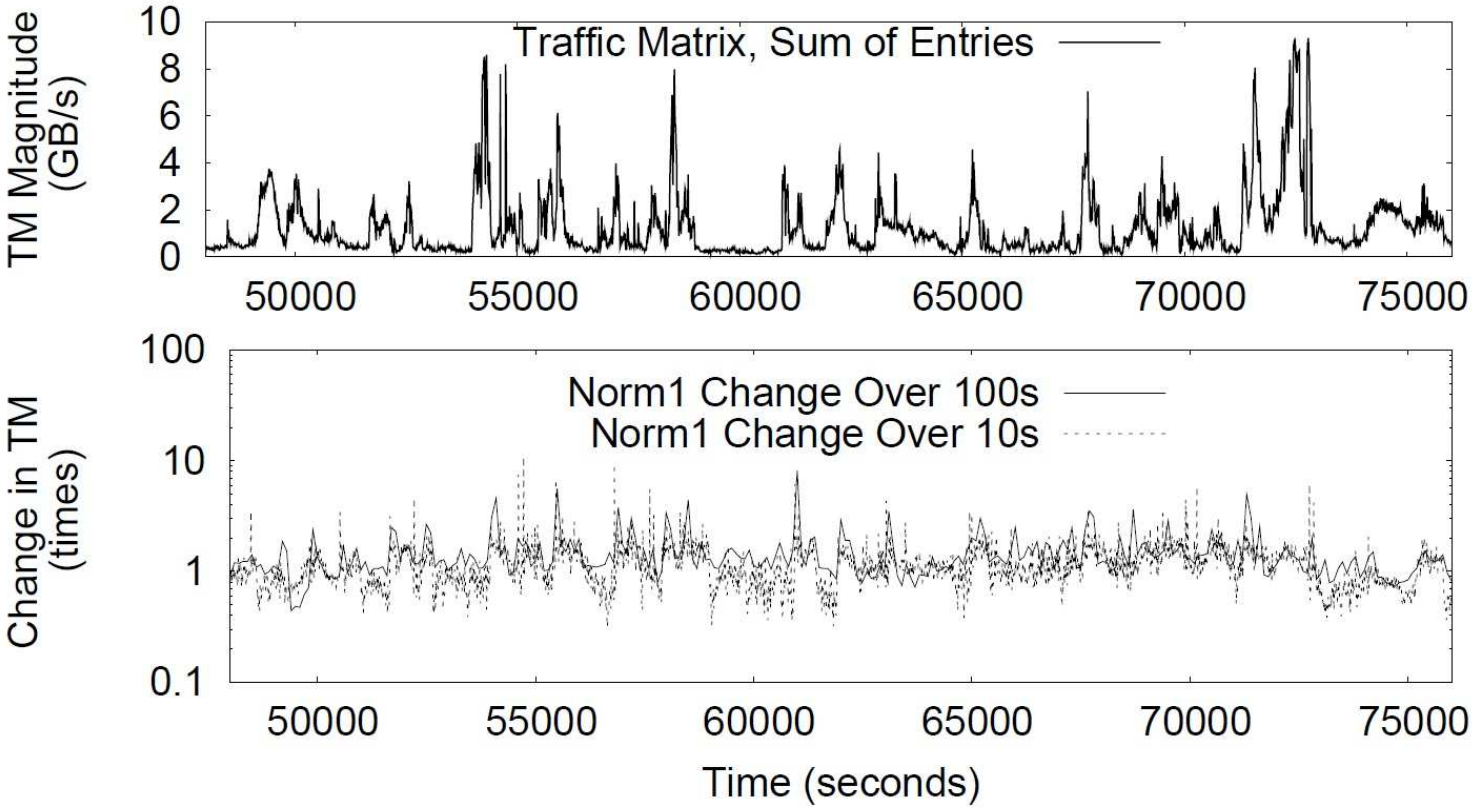


Figura: Em (l*og(Bytes)*/10s) entre servidores num cluster operacional da MS [S. Kandula and Patel 2009]

1.- Work-Seeks-Bandwidth: Maior parte do tráfego na linha diagonal (sem escala log só ficaria um linha diagonal). Requer lógica nas aplicações para alocar servidores no mesmo ToR.

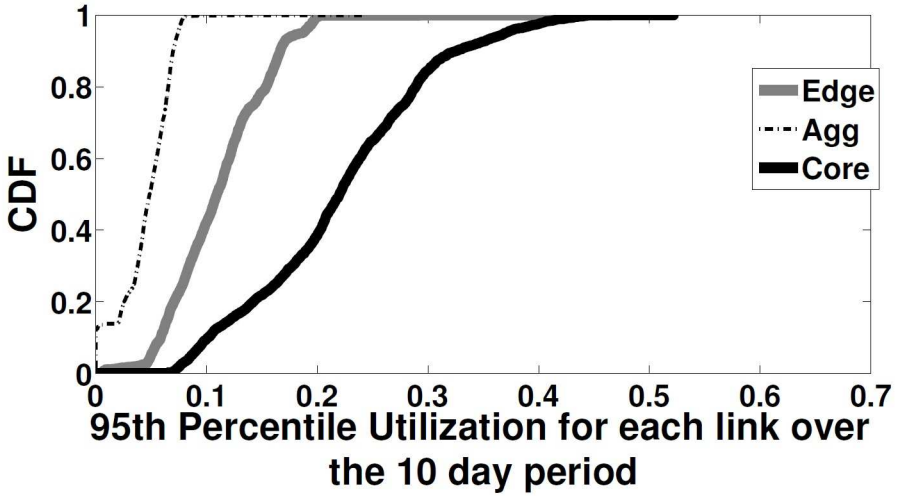
2.- Scatter-Gather: Linhas verticais e horizontais, onde um servidor empurra (ou puxa), dados a vários servidores dentro de todo o *cluster, i.e., MapReduce*

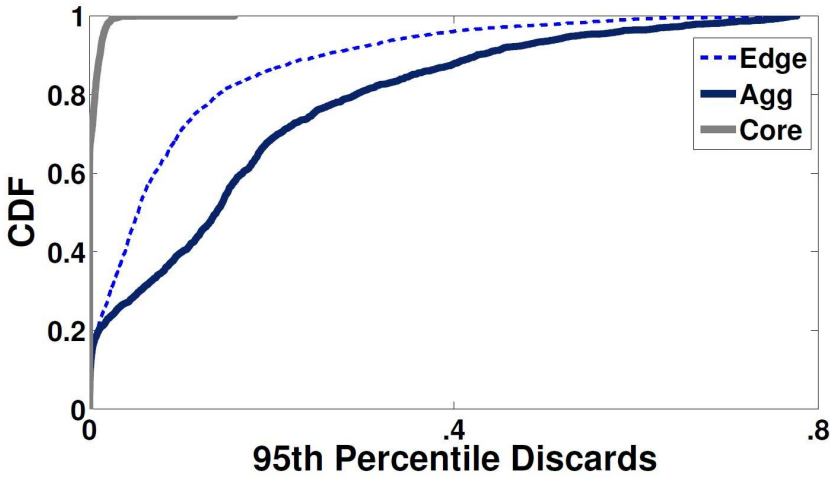
Padrão do tráfego varia constantemente



- Padrões de tráfego do tipo ON/OFF
- Volume do tráfego e participantes mudam constantemente

Utilização dos links e perda dos pacotes por camadas

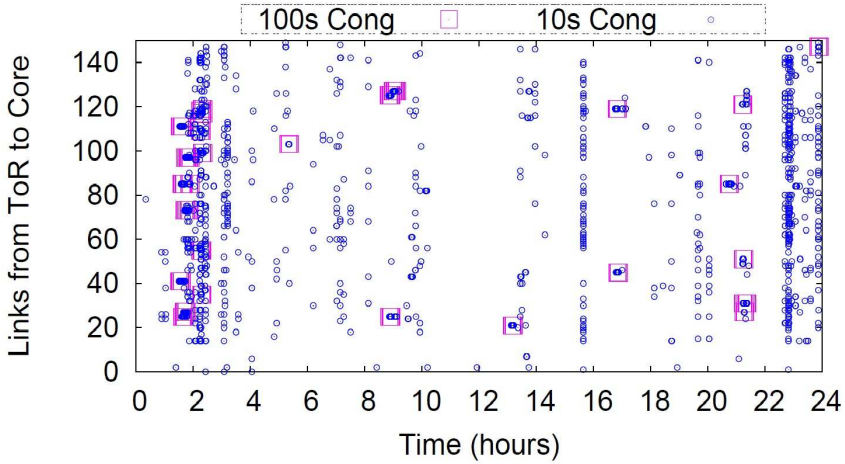


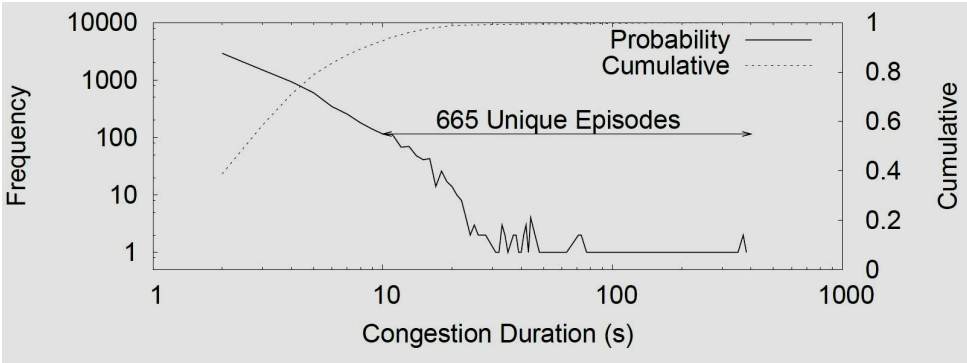


- A carga média nos Core é superior e diminui em direção aos switches ToR
- As maiores perdas, em média, são superiores nos ToRs e mais reduzidas no core, sugerindo tráfego em rajadas nos enlaces Agg e ToR
- Uma pequena fração dos enlaces apresenta perdas muito maiores

  -> Seria possível achar rotas por caminhos alternativos evitando a maior parte das perdas por switches congestionados.

Congestionamento: Hits Hard When it Hits

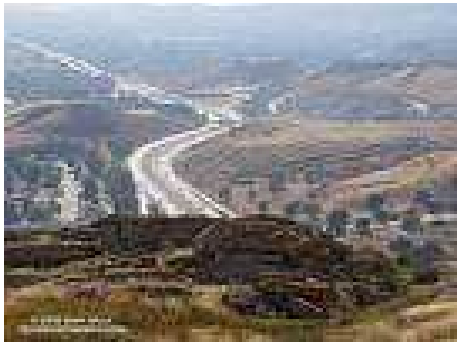




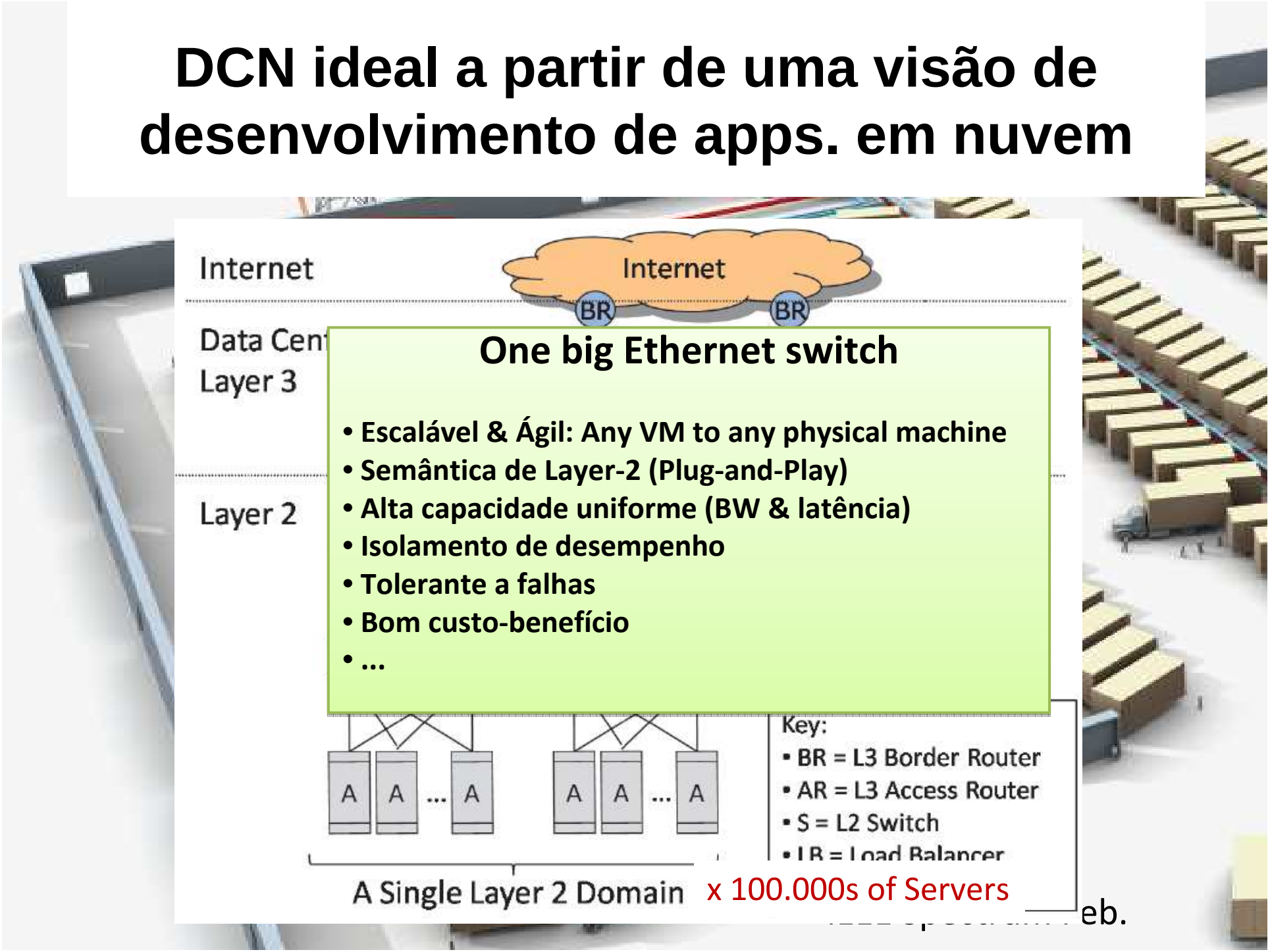
- 86% dos enlaces sofreram congestionamento de pelo menos 10 segundos
- Períodos curtos de congestionamento (círculos azuis, 10s de alta utilização) correlacionados com dezenas de enlaces -> picos de demanda da aplicação
- Períodos largos tendem a se localizar em um número mais reduzido de links

Organização do minicurso

- Introdução
  - O que é Cloud computing?
  - Definições e características essenciais
  - Custos e eficiência energética
- Caracterização dos *data centers* para serviços em nuvem
  - Infraestrutura de um *data center*
  - Limitações das arquiteturas de redes atuais
  - Caracterização do tráfego
  - Requisitos das arquiteturas de rede para novos *data centers*
- Novas arquiteturas para *data centers*
- Conclusão



DCN ideal a partir de uma visão de desenvolvimento de apps. em nuvem



Objetivos e requisitos

Goals	Requirements	Features
Resource Pooling (servers and network eq.) & Agility	R1: Any VM to any physical machine. - Let services “breathe”: Dynamically expand and contract their footprint as needed - L2 semantics	· ID/loc split · Scalable L2
	R2: High network capacity - Uniform BW and latency for various traffic patterns between any server pair - 1:1, 1:M, N:N efficient communications along any available physical paths	· Multipath support · New TE (load-balancing)
Reliability	R3: Design for failure. - Failures (servers, switches) will be common at scale.	· Fault-tolerance
Low Opex	R4: Low configuration efforts - Ethernet plug-and-play functionality	· Auto-config.
	R5: Energy efficiency - Networking design for idle link/server optimization	· Energy/Cost-awareness
Low Capex	Use commodity hardware	· Scaling-out
Control	Include middlebox services in the data path as required	· Network ctrl.

Café!!!

Organização do minicurso

- Introdução
- Caracterização dos *data centers* para serviços em nuvem
- Novas arquiteturas para *data centers*
  - *Monsoon*
  - *VL2*
  - *Portland*
  - *BCube e MDCube*
- Conclusão
  - Cenários futuros
  - Desafios e oportunidades

Metas da arquitetura *cloud data centers*

- Redução da dependência de switches de grande porte no core da rede;
- Simplificação do software de rede (plano de controle, pilha de protocolos);
- Aumento da confiabilidade do sistema como um todo;
- Evitar que a rede seja o gargalo do sistema e simplificar o trabalho do desenvolvimento de aplicações;
- Redução dos custos de capital e operacionais.

Resumo de requisitos

Goals	Requirements	Features
Resource Pooling (servers and network eq.) & Agility	R1: Any VM to any physical machine. - Let services “breathe”: Dynamically expand and contract their footprint as needed - L2 semantics	· ID/loc split · Scalable L2
	R2: High network capacity - Uniform BW and latency for various traffic patterns between any server pair - 1:1, 1:M, N:N efficient communications along any available physical paths	· Multipath support · New TE (load-balancing)
Reliability	R3: Design for failure. - Failures (servers, switches) will be common at scale.	· Fault-tolerance
Low Opex	R4: Low configuration efforts - Ethernet plug-and-play functionality	· Auto-config.
	R5: Energy efficiency - Networking design for idle link/server optimization	· Energy/Cost-awareness
Low Capex	Use commodity hardware	· Scaling-out
Control	Include middlebox services in the data path as required	· Network ctrl.

Organização do minicurso

- Introdução
- Caracterização dos *data centers* para serviços em nuvem
- Novas arquiteturas para *data centers*
  - *Monsoon*
  - *VL2*
  - *Portland*
  - *BCube e MDCube*
- Conclusão
  - Cenários futuros
  - Desafios e oportunidades

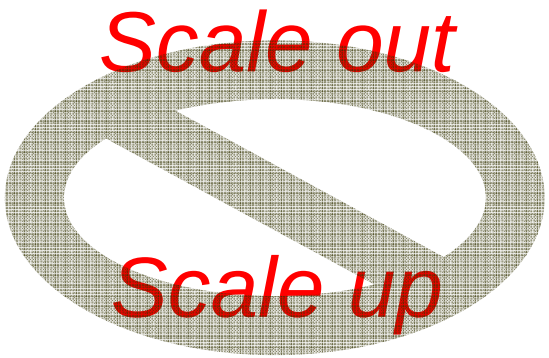
Albert Greenberg, Parantap Lahiri, David A. Maltz, Parveen Patel, Sudipta Sengupta

(Microsoft Research, Redmond, WA, USA)

Monsoon

Filosofia

- Comoditização da infraestrutura como forma de obter:
  - Escalabilidade
  - Baixo custo
    - Portas de 1Gbps < $100



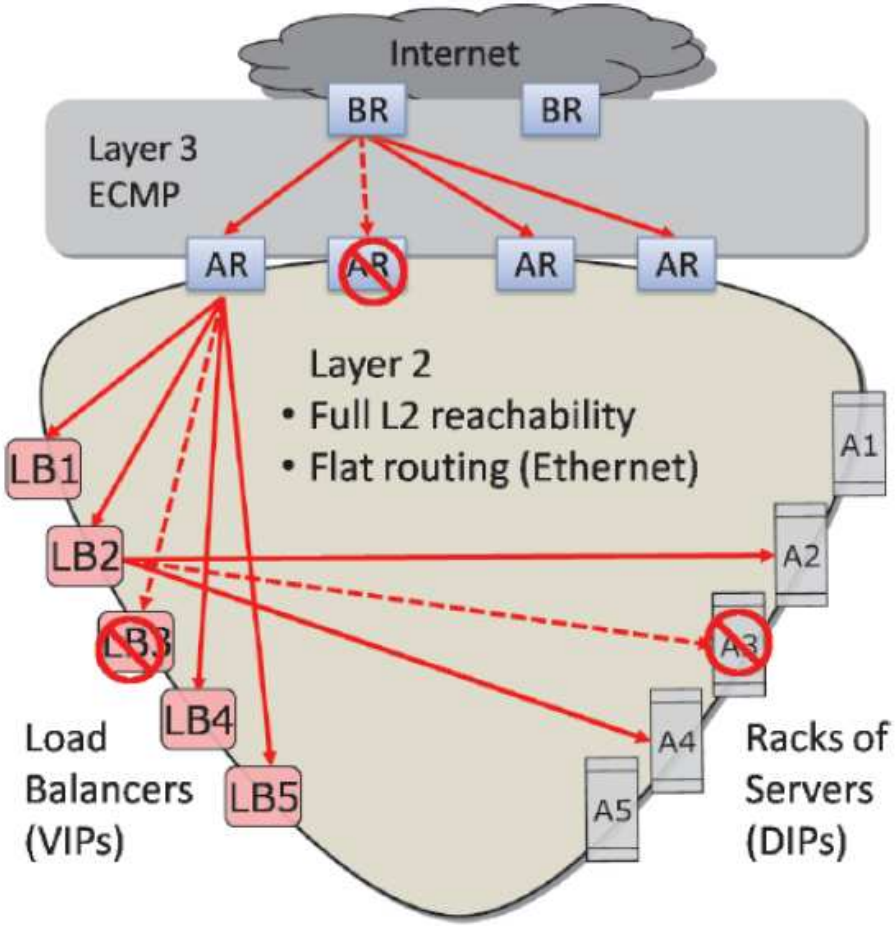
Aspectos principais

- Engenharia de tráfego utilizando uma malha de camada 2
  - Valiant Load Balancing (VLB)
  - Acomodar qualquer matriz de tráfego
- Escalabilidade para criar grandes domínios de camada 2
  - Suportar mais de 100.000 servidores em um único domínio de camada 2
- Espalhamento de carga utilizando toda a estrutura de switches
  - Melhor aproveitamento da infraestrutura
  - Evitar usar Load Balancers em hardware

Arquitetura

Camada 3 X Camada 2
Camada 2 é mais barata
Camada 2 evita fragmentação de recursos

- Uma porta em camada 2 custa de 10 % a 50% de uma porta em camada 3
- Camada 2
  - Prover a comunicação entre todos os servidores
  - Taxa de 1 Gbps (taxa da interface)
- Camada 3
  - Conectar o *data center* à Internet



Visão geral da arquitetura do Monsoon. Extraída de [Greenberg et al. 2008].

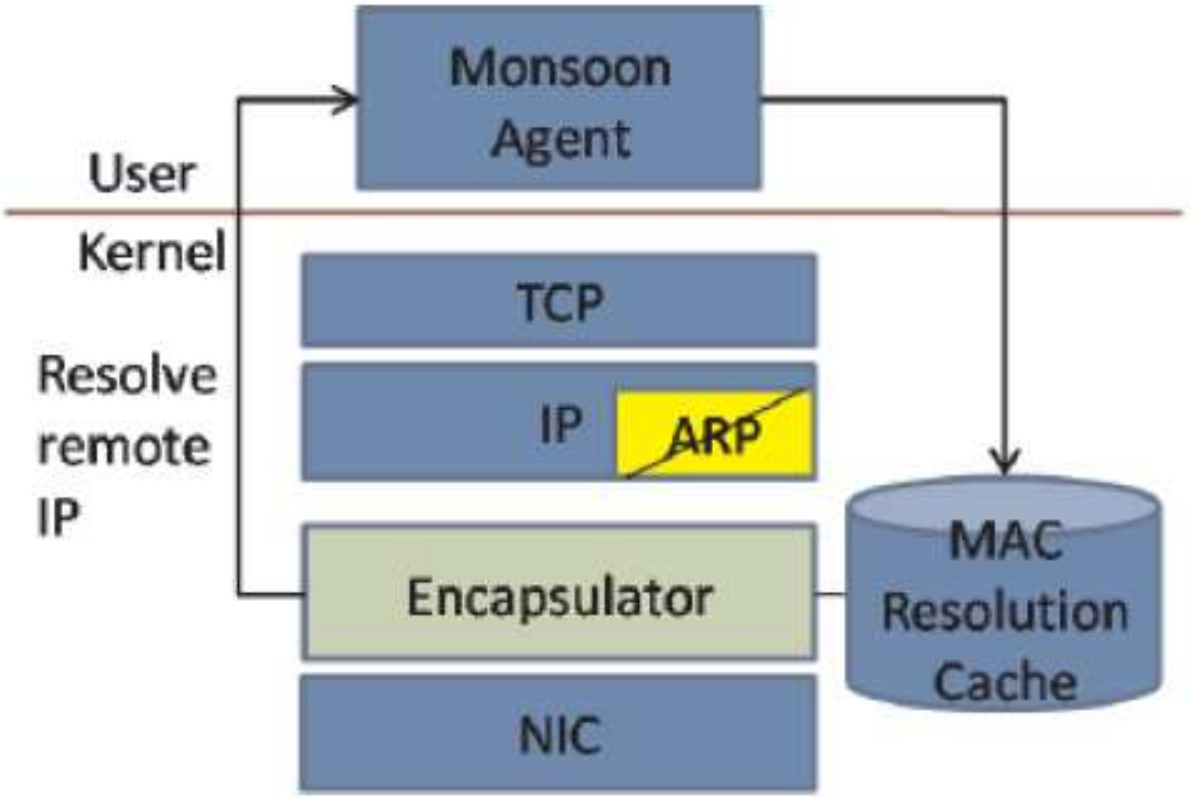
- *Border Routers* (BR)
  - Admitir as requisições vindas da Internet
- *Equal Cost Multi Path* (ECMP)
  - Espalhar as diversas requisições entre os roteadores de acesso (AR)
- *Access Routers* (AR)
  - Distribuir as requisições entre os diversos balanceadores de carga (LB)
- *Load Balancers* (LB)
  - Implementado em servidores via software programável
  - Espalhar as requisições entre os servidores responsáveis pelo tratamento da requisição
- *Health Service*
  - Monitorar continuamente o estado de todos os componentes do *data center*

Componentes exigidos

- MAC-in-MAC
- 16K entradas MAC
- Dois tipos de switches
  - Tof-of-Rack (TOR): 20 links de 1Gbps conectando 20 servidores em cada rack
  - Core: switches com 144 portas de 10Gbps

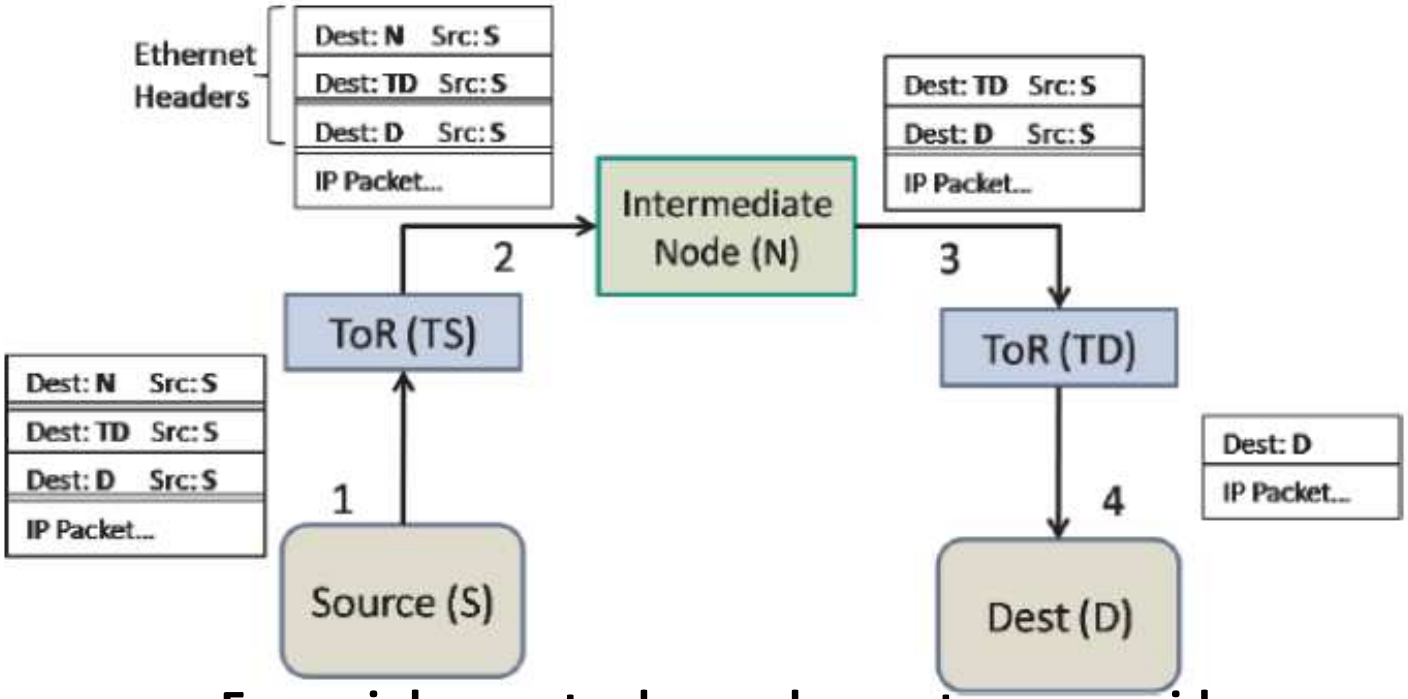
Encaminhamento Servidor-a-Servidor

- Engenharia de tráfego
  - VLB garante a transferência à taxa da interface
  - Escolhe um switch intermediário para enviar os dados (bounce off)
- Espalhamento da carga
  - MAC rotation: escolhe um servidor da lista de MACs associados a um serviço
  - Usa consistent hashing por fluxo para evitar reordenação de pacotes



Pilha de rede implementada pelos servidores do Monsoon.
Extraída de [Greenberg et al. 2008].

- **Serviço de diretório traduz o IP em**:
  - Lista de MACs associados a um serviço
  - MAC dos switches TOR nos quais os servidores estão conectados
- **Alteração na pilha de protocolos dos nós finais**
  - ARP desativado e substituído pelo Monsoon *Agent*
  - Nova interface MAC virtual (*Encapsulator*)



Encaminhamento de quadros entre os servidores de origem e destino via encapsulamento MAC-in-MAC. Extraída de [Greenberg et al. 2008].

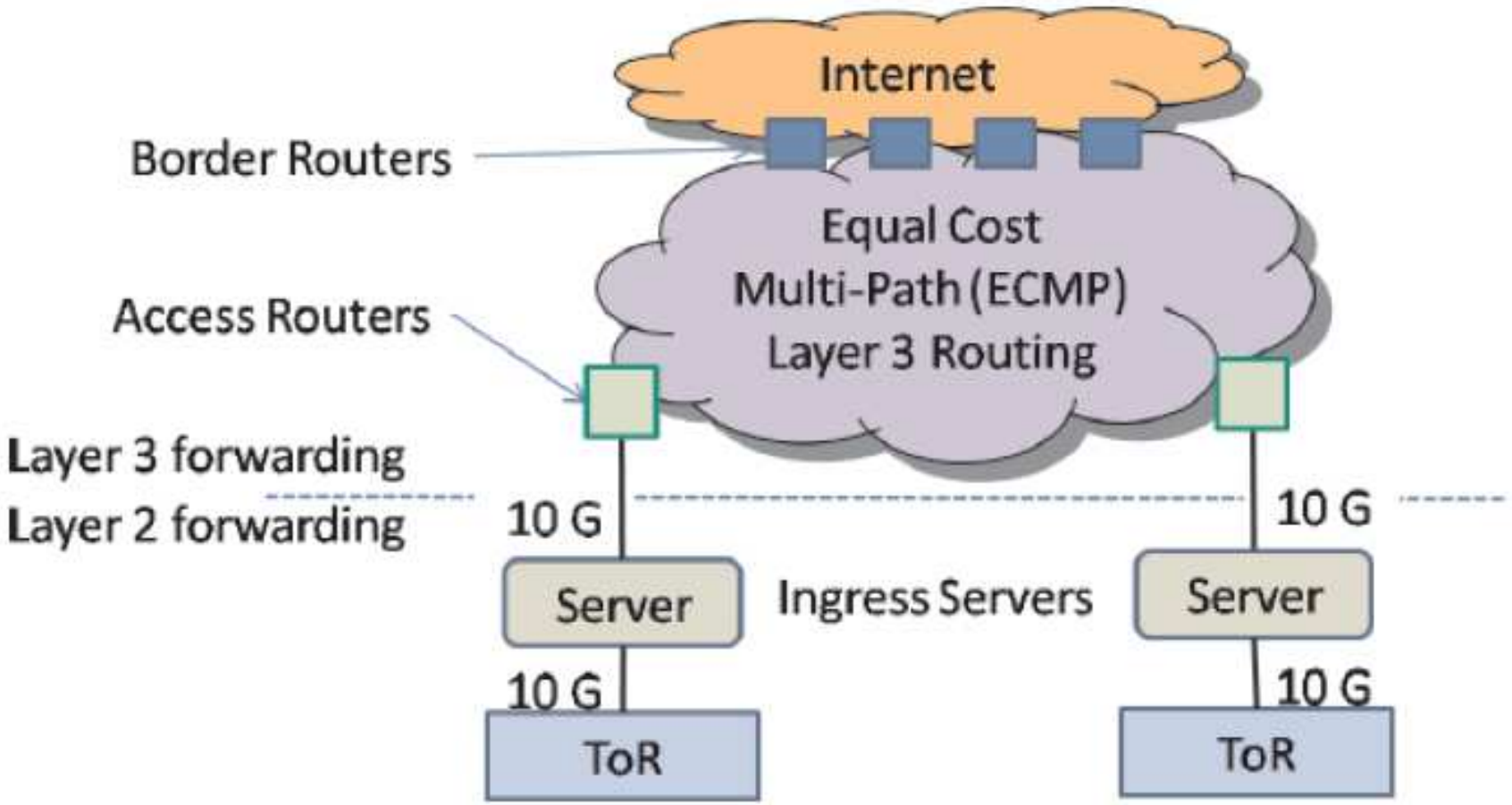
O que é necessário?

- Lista de MAC para os servidores que implementam o serviço
- Switches intermediários para VLB
- Encapsulator escolhe um MAC (MAC rotation) e o switch TOR correspondente
- Encapsulator escolhe um switch intermediário

Conectividade externa

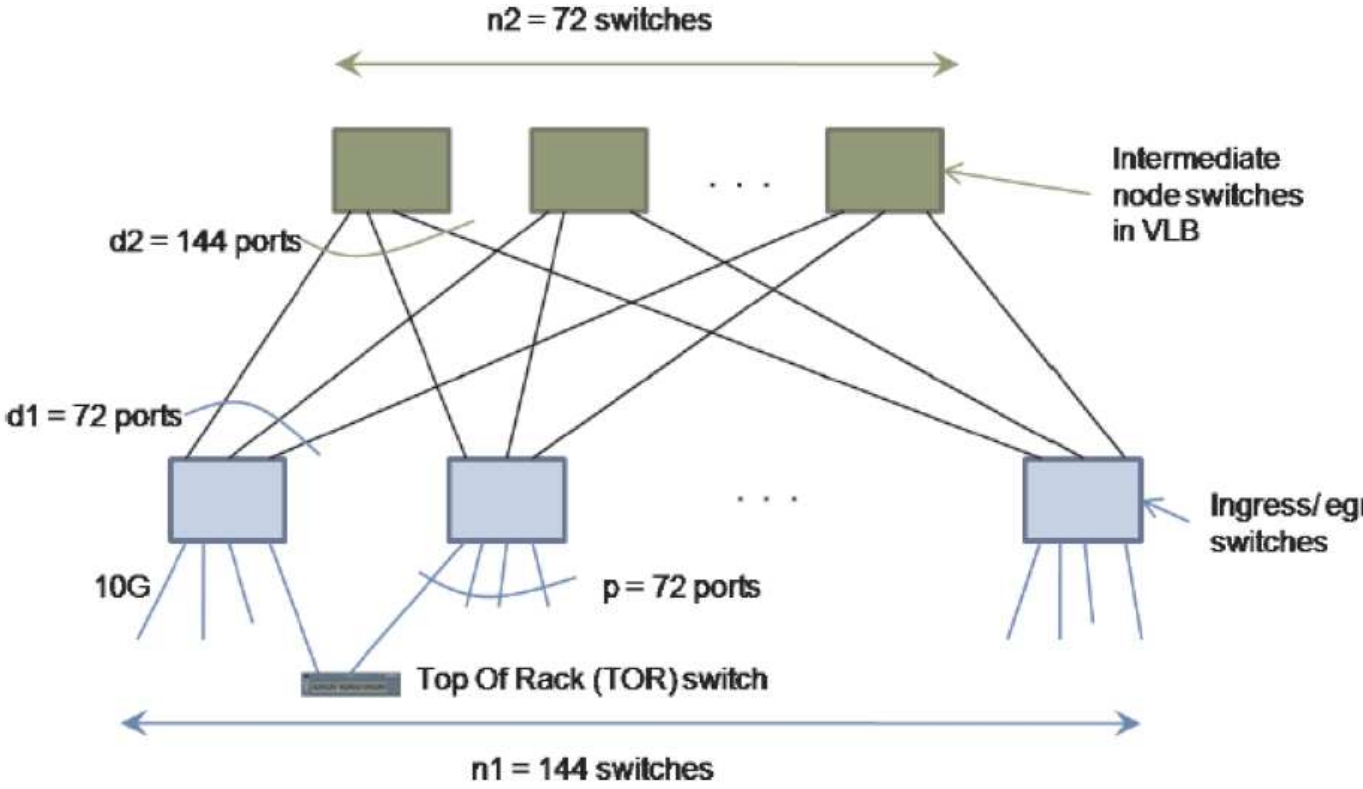
- Malha de camada 3
  - Roteadores de borda
  - Roteadores de acesso
    - Possuem um servidor de ingresso associado a eles, de forma a efetuar o tratamento adequado das requisições segundo as especificações do Monsoon
  - ECMP
  - Servidor de ingresso
    - Possuem duas interfaces de rede
- Roteador de Acesso
  - Switch TOR

Conectividade externa



Caminho utilizado pelas conexões originadas ou destinadas à Internet. Extraída de [Greenberg et al. 2008].

- Exemplo de topologia com 103.680 servidores
- TOR com 2 interfaces de 10Gbps e 24 interfaces de 1 Gbps
  - 10Gbps→ conectadas a dois switches de ingresso/egresso
  - 1Gbps→ conectadas aos servidores
- *Core* em dois níveis (n1 e n2)
  - n1 com 144 switches com 144 portas de 10Gbps
  - Estão conectados aos switches em n2
  - n2 com 72 switches com 144 portas de 10Gbps
    - Estão conectados aos switches em n1



144 switches x 72 portas

10368

/

2

5184 TORs

x

20

103.680 servidores

Albert Greenberg James R. Hamilton Navendu Jain, Srikanth Kandula Changhoon Kim Parantap Lahiri, David A. Maltz Parveen Patel Sudipta Sengupta

(Microsoft Research)

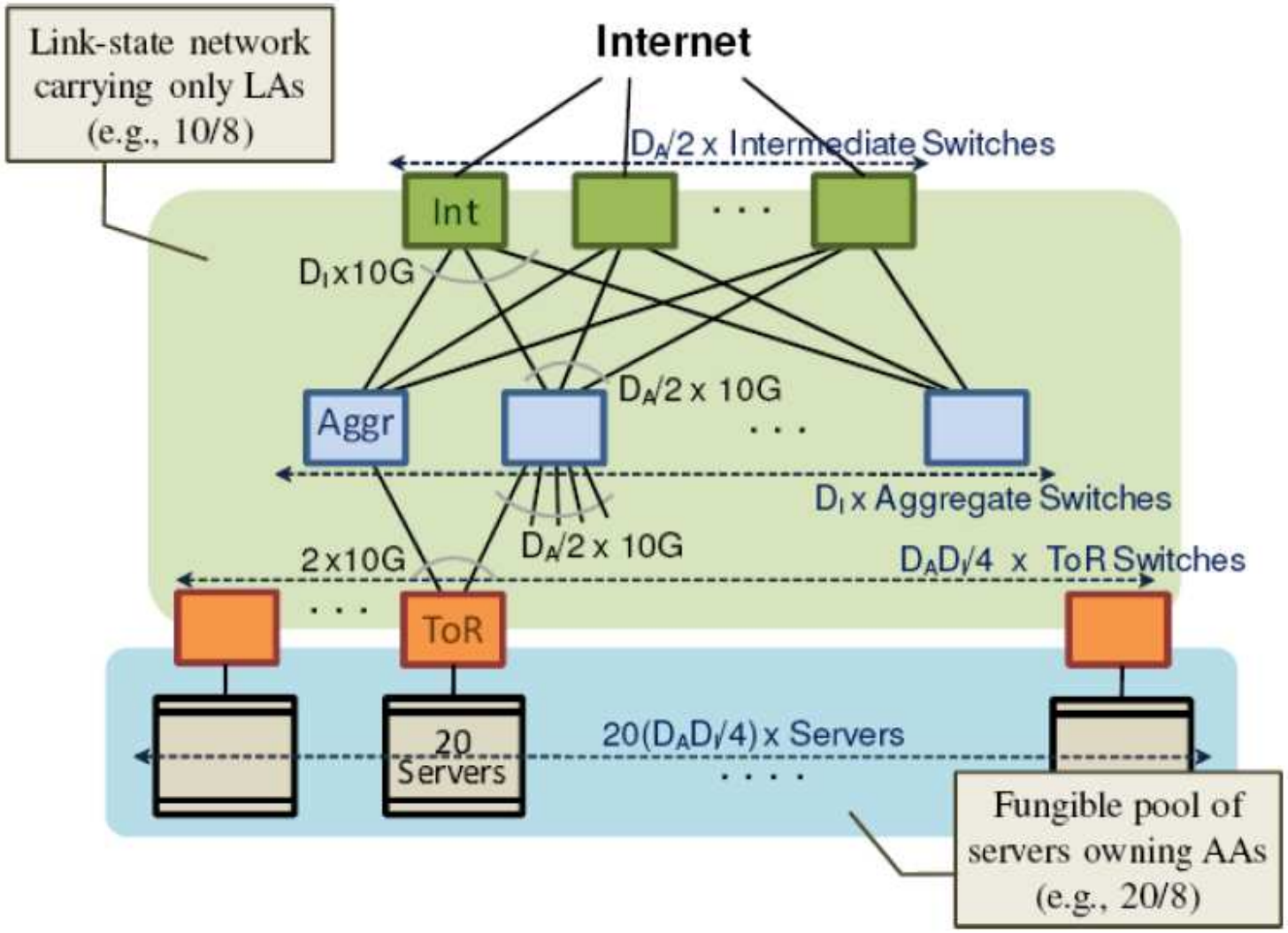
VL2

Filosofia

- Uma evolução do Monsoon, mesmo grupo
  - Mais realista
- Criar uma camada 2 virtual (VL2 – *Virtual Layer 2*)
  - Criar a ilusão de que todos os servidores estão conectados a um único switch de camada 2, independentemente do tamanho do *data center* (1 ou 100.000 servidores!)
- Investigar que tipo de solução poderia ser construída utilizando os recursos disponíveis atualmente
  - Oferecer transparência a aplicações legadas

Aspectos principais

- A comunicação servidor-a-servidor só pode ser limitada pela taxa de transmissão das interfaces de rede de cada servidor (1Gbps) = sem oversubscription
- Isolamento de desempenho
- Agilidade
- VLB para engenharia de tráfego
  - Escolhe um switch intermediário para espalhar a carga

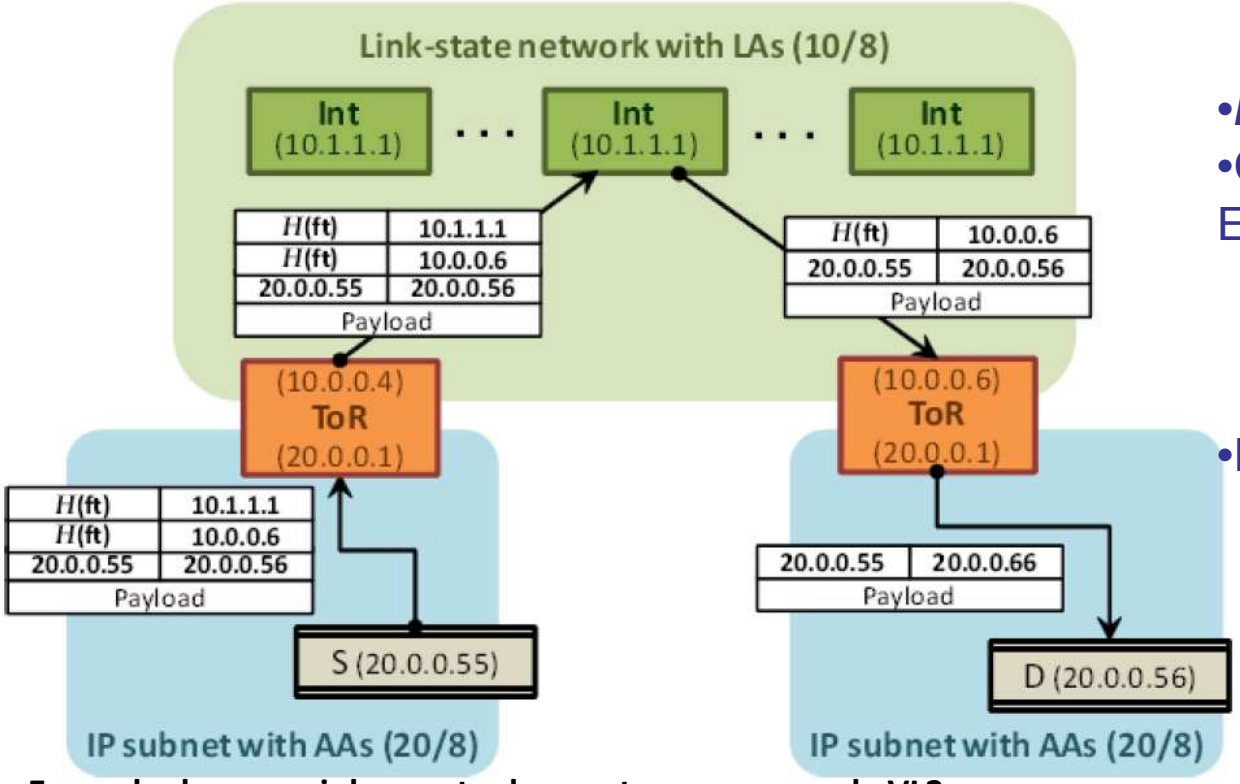


Exemplo de *backbone* utilizado pelo VL2. Extraída de [Greenberg et al. 2009b].

- Mesma topologia do Monsoon!
- Rede Clos: alta conectividade

Endereçamento e roteamento

- São utilizadas duas classes de endereços
  - Com significado topológico (LA – *Locator Addresses*)
    - Switches
    - LAs são disseminados pelos switches através de um protocolo de estado de enlace
  - Endereços planos de aplicação (AA – *Application Addresses*)
    - Servidores
    - Permanecem inalterados, independentemente da localização do servidor no interior do data center
  - Para cada AA é atribuído um LA do switch TOR



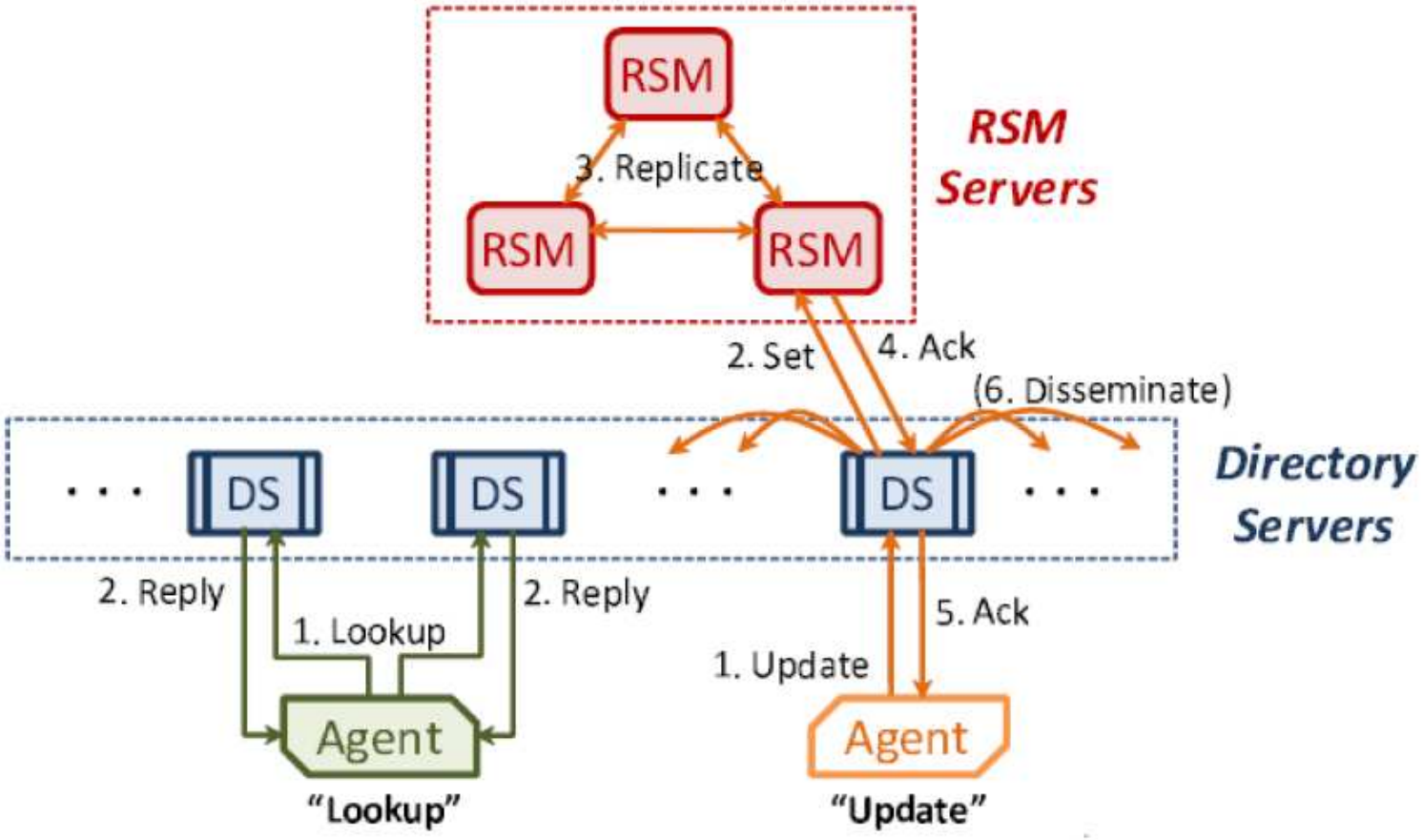
Exemplo de encaminhamento de pacotes em uma rede VL2. Extraída de [Greenberg et al. 2009b].

- *Brodcast* ARP
- Combinação entre o VLB e o ECMP
  - O VL2 define o mesmo endereço IP para os switches intermediários
- ECMP
  - Garantir a entrega dos pacotes a um switch intermediário
  - Ideal para a tolerância a falhas

- Utiliza encapsulamento IP-in-IP
  - Agente VL2 executado nos servidores intercepta os pacotes e os encapsula
    - Um cabeçalho destinado ao servidor de destino
    - Um cabeçalho destinado ao switch TOR de destino
    - Um cabeçalho destinado a um swtich intermediário (VLB)

Serviço de diretório

- Provê 3 funcionalidades
  - Consultas
  - Atualizações de mapeamento AA/LA
  - Atualização de cache reativo
- Arquitetura em dois níveis
  - Nível 1 constituído por servidores de diretório (DS) otimizados para leitura (consultas)
  - Nível 2 constituído por servidores de diretório (RSM – *Replicated State Machine*) otimizados para escrita



Arquitetura do serviço de diretório do VL2.
Extraída de [Greenberg et al. 2009b].

Atualização do cache nos servidores é reativa. TOR avisa o DS que então corrige via unicast a cache do servidor.

Avaliação

- VL2 implementado e testado em um protótipo de 80 servidores
- Alta capacidade uniforme de transmissão
  - 11x mais rápido quando comparado com a arquitetura tradicional➔2.7 TB de dados
  - VLB eficiente
  - Isolamento de desempenho garantido
  - Boa convergência após falhas

Radhika Niranjan Mysore, Andreas Pamboris, Nathan Farrington, Nelson Huang, Pardis Miri, Sivasankar Radhakrishnan, Vikram Subram

(University of California San Diego, San Diego, USA)

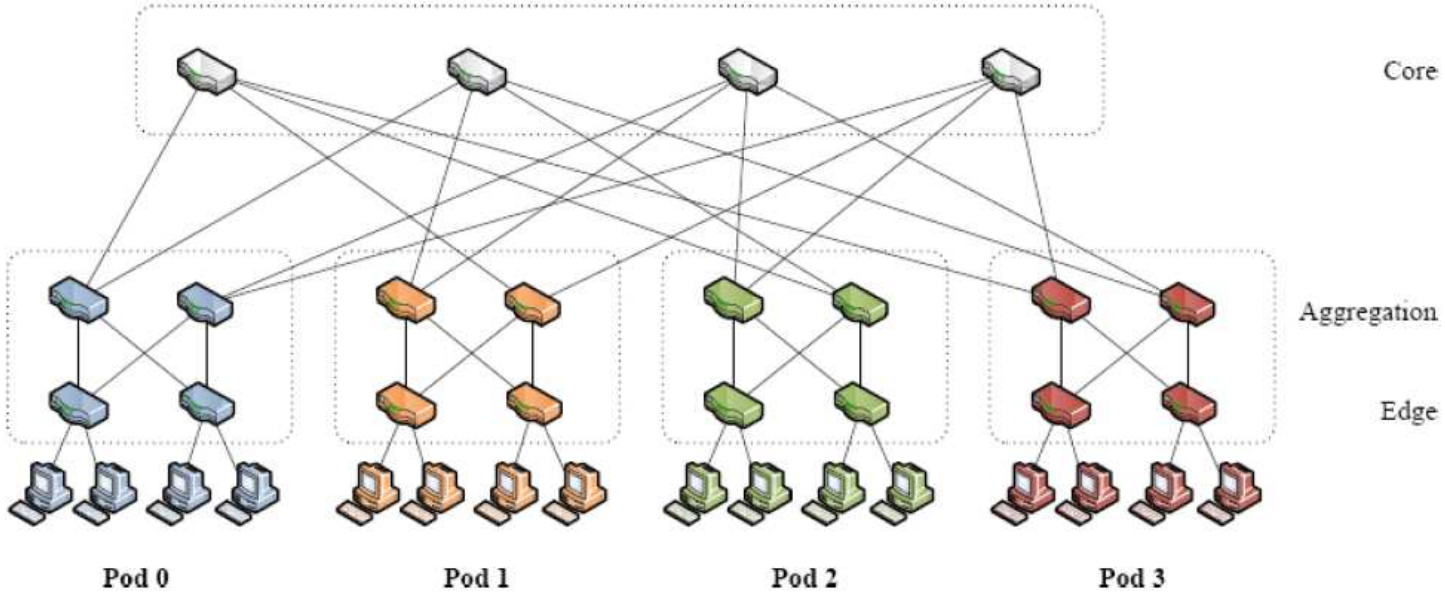
PORTLAND

Abordagem

- Utiliza um protocolo que possibilita aos switches descobrirem suas posições na topolgia da rede
- Atribui Pseudo endereços MAC (PMAC) para todos os nós finais, de forma a codificar suas posições na topologia
- Utiliza um serviço centralizado de gerenciamento de infraestrutura de rede (*Fabric Manager*)
- Utiliza um serviço de *Proxy* para contornar o *broadcast* inerente ao ARP

Arquitetura

- Utiliza uma estrutura *fat tree* em 3 níveis
  - Parâmetro *k* define o número de portas em cada switch
  - A topologia é organizada em *k* conjuntos de servidores (*pods*)
  - Permite a comunicação não bloqueante entre $k^3/4$ servidores utilizando $5k^2/4$ switches.

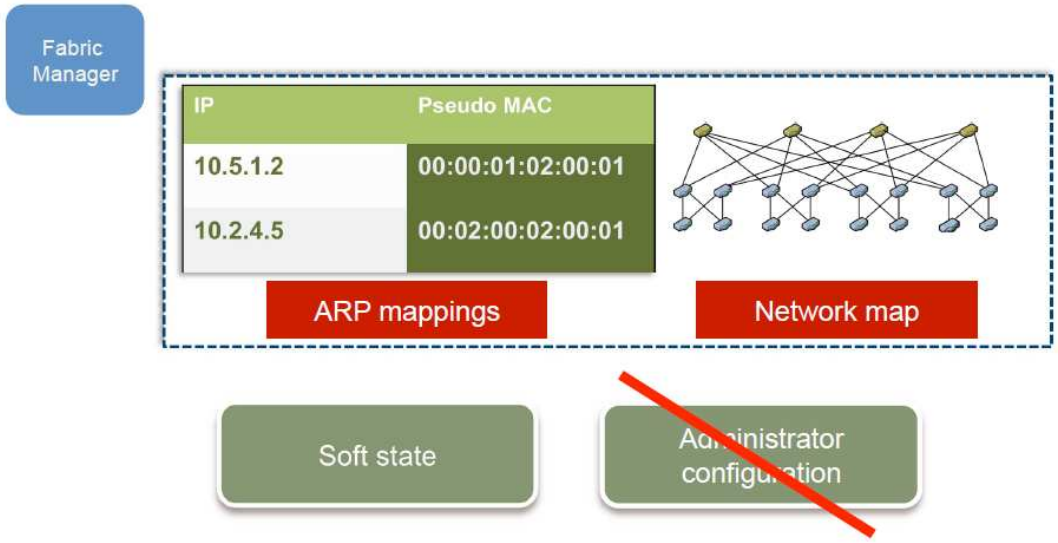


[Mysore et al. 2009].

Fabric manager

Gerenciador centralizado de infraestrutura

- Mantém o estado atual da rede
  - Configurações
  - Topologia
- Executado no espaço do usuário em uma máquina dedicada. Auxiliam no tratamento de:
  - Requisições de ARI
  - Tolerância a falhas
  - *Multicast*

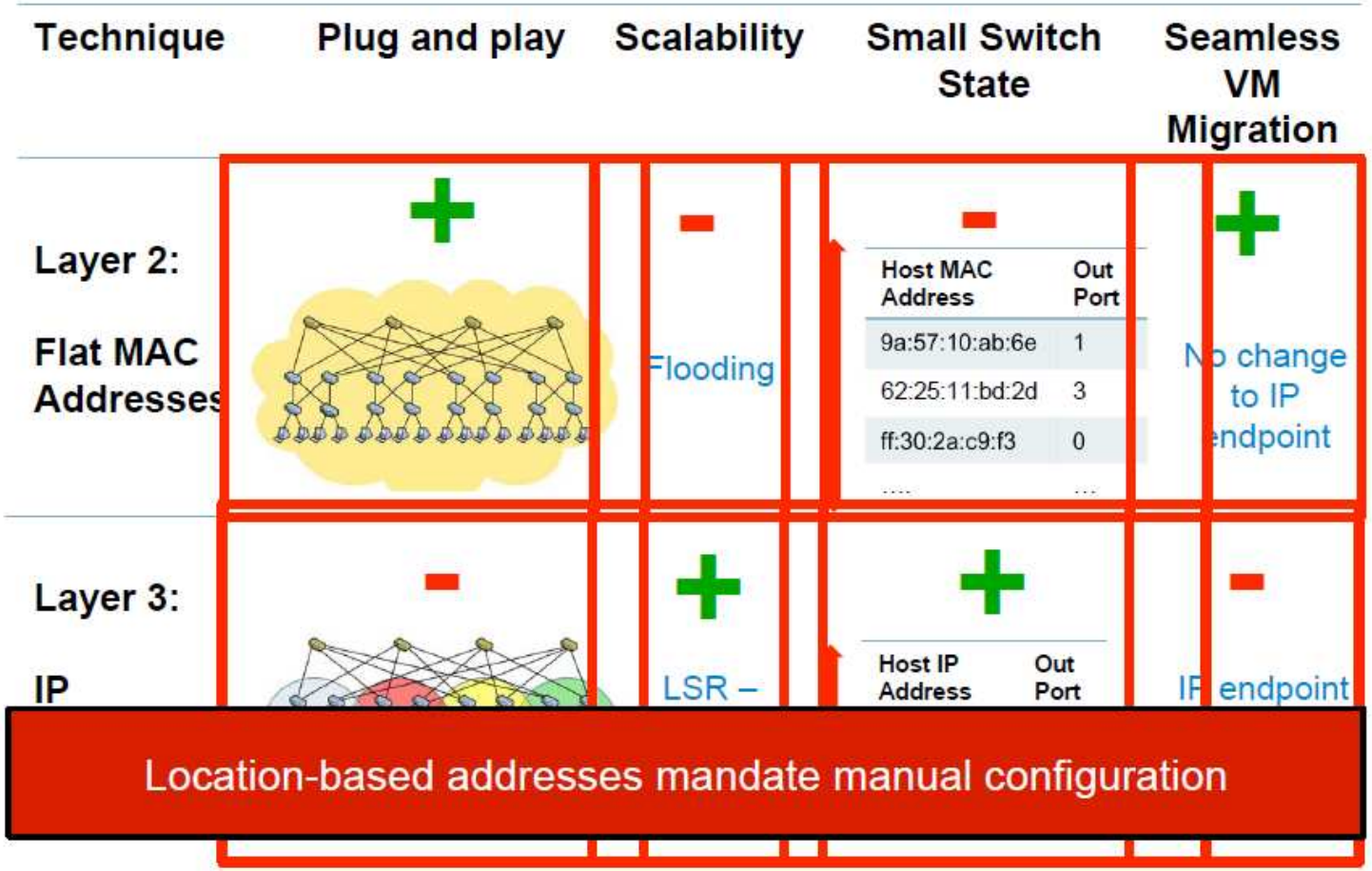


Layer 2 versus Layer 3 Data Center Fabrics

Technique	Plug and play	Scalability	Small Switch State	Seamless VM Migration
Layer 2: Flat MAC Addresses	+	-	-	+
Layer 3: IP Addresses	-	+	+	-

[Source: http://bnrg.eecs.berkeley.edu/~randy/Courses/CS268.F09/lectures/08a-Portland.pdf]

Layer 2 versus Layer 3 Data Center Fabrics



[Source: http://bnrg.eecs.berkeley.edu/~randy/Courses/CS268.F09/lectures/08a-Portland.pdf]

Cost Consideration: Flat Addresses vs. Location Based Addresses

- Commodity switches today have ~640 KB of low latency, power hungry, expensive on chip memory
  - Stores 32 – 64 K flow entries
- Assume 10 million virtual endpoints in 500,000 servers in data center
- Flat addresses ➔ 10 million address mappings ➔ ~100 MB on chip memory ➔ ~150 times the memory size that can be put on chip today
- Location based addresses ➔ 100 – 1000 address mappings➔ ~10 KB of memory ➔ easily accommodated in switches today

[Source: http://bnrg.eecs.berkeley.edu/~randy/Courses/CS268.F09/lectures/08a-Portland.pdf]

Endereços PMAC (Pseudo MAC)

- Endereços hierárquicos, únicos para cada nó final.
  - Representam a localização de cada nó na topologia
  - Nós finais permanecem inalterados, utilizam o AMAC (*Actual* MAC)
  - Tradução dos pacotes (AMAC – PMAC) são feitas pelos switches de ingresso/egresso
    - Regras do OpenFlow para tratamento de “fluxos”

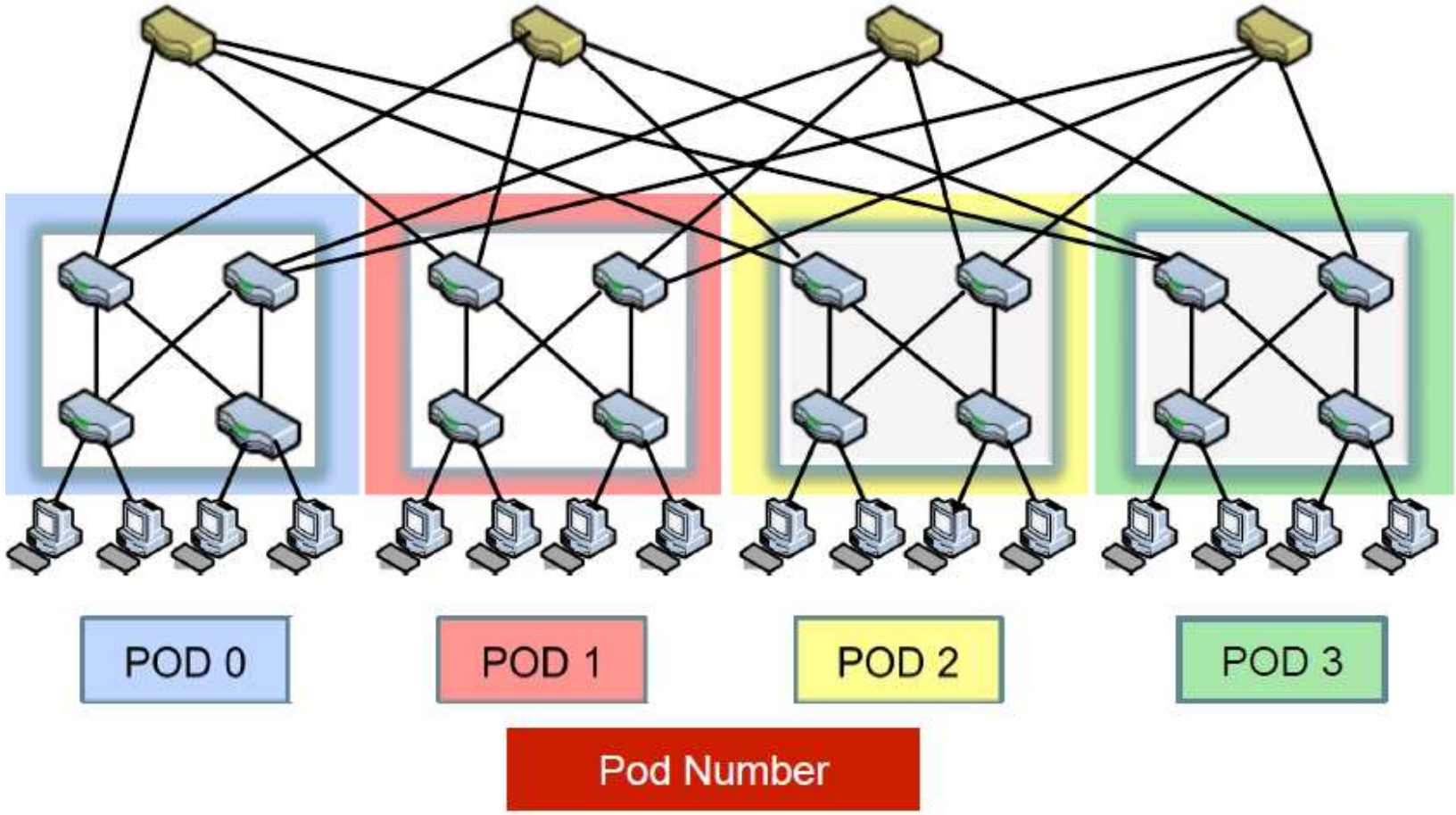
Endereços PMAC (Pseudo MAC)

- Formato do PMAC:

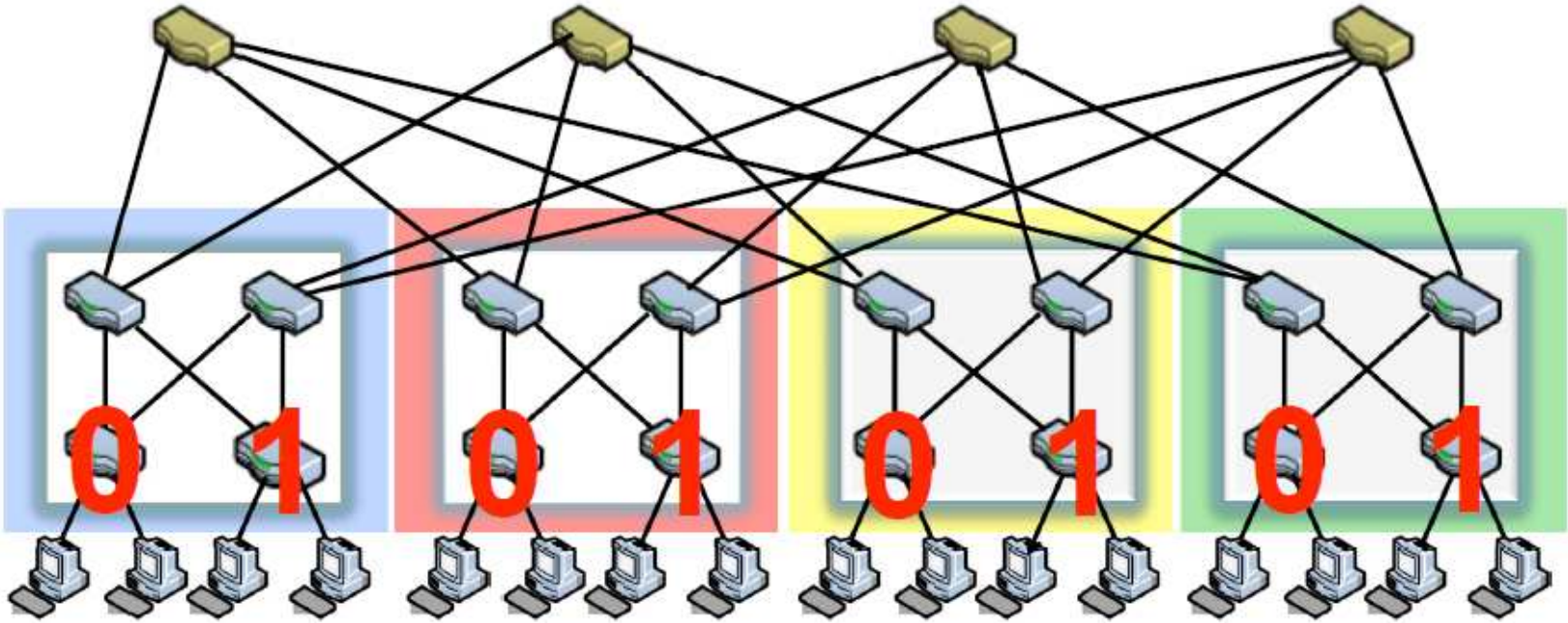
pod.posição.porta.vmid

- **pod** – 16 bits e identifica o pod onde os nós estão localizados
- **posição** - 8 bits e indica a posição do switch dentro do pod
- **porta** - 8 bits e representa a porta na qual o nó final está ligado ao switch
- **vmid** – 16 bits e serve para multiplexar máquinas virtuais em uma mesma máquina física

pod – 16 bits e identifica o pod onde os nós estão localizados

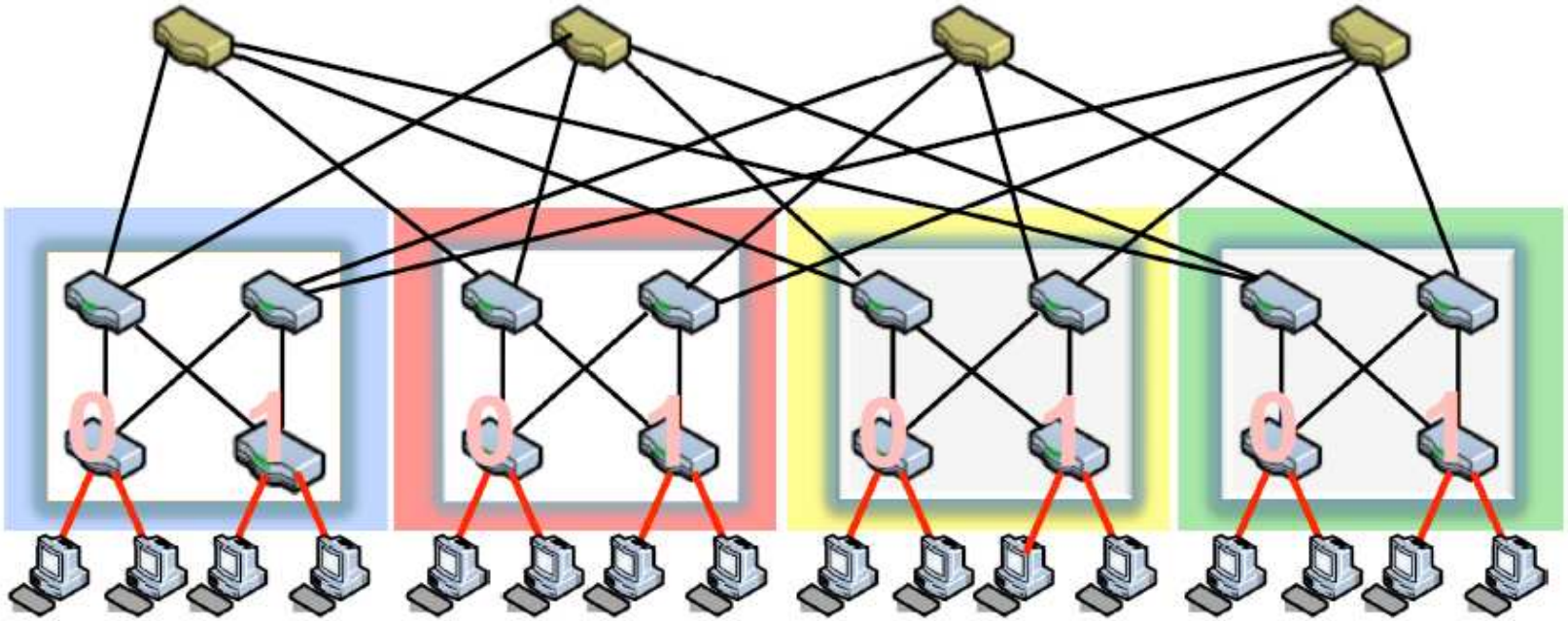


posição - 8 bits e indica a posição do switch dentro do pod



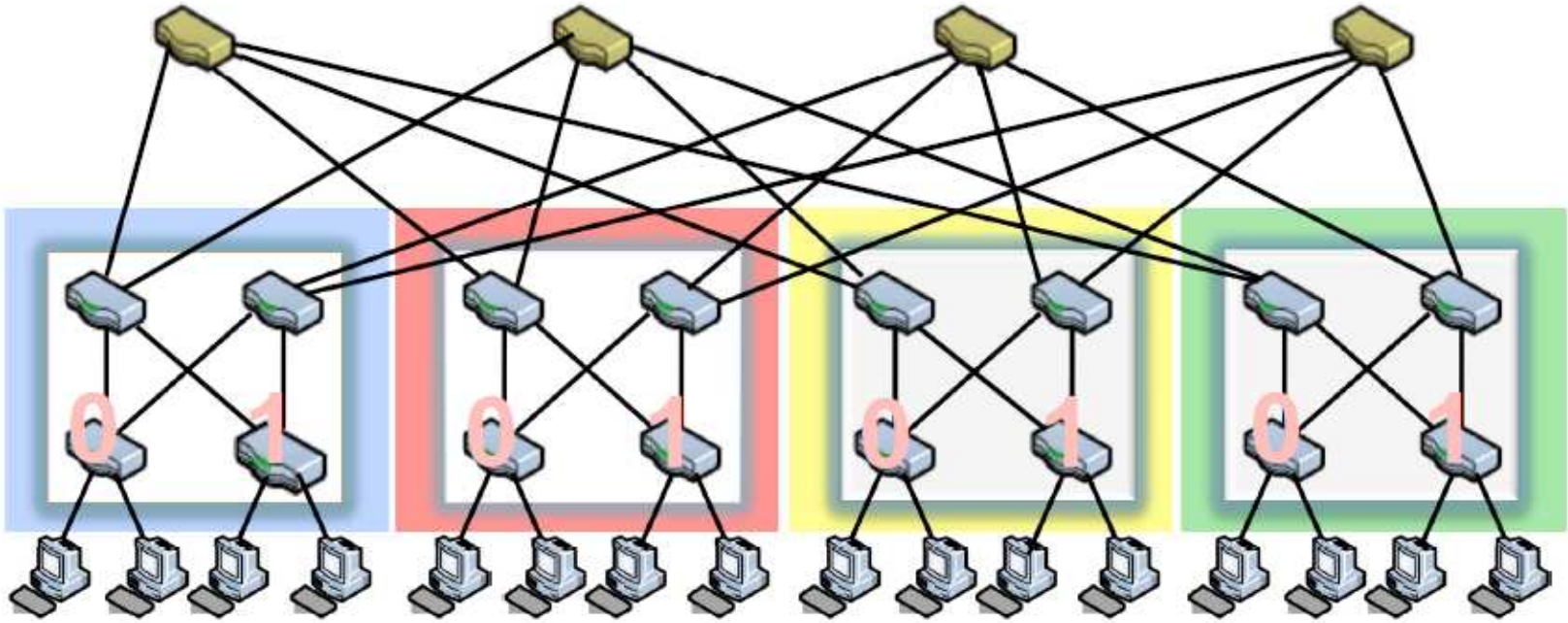
PMAC: pod.position.port.vmid

porta - 8 bits e representa a porta na qual o nó final está ligado ao switch



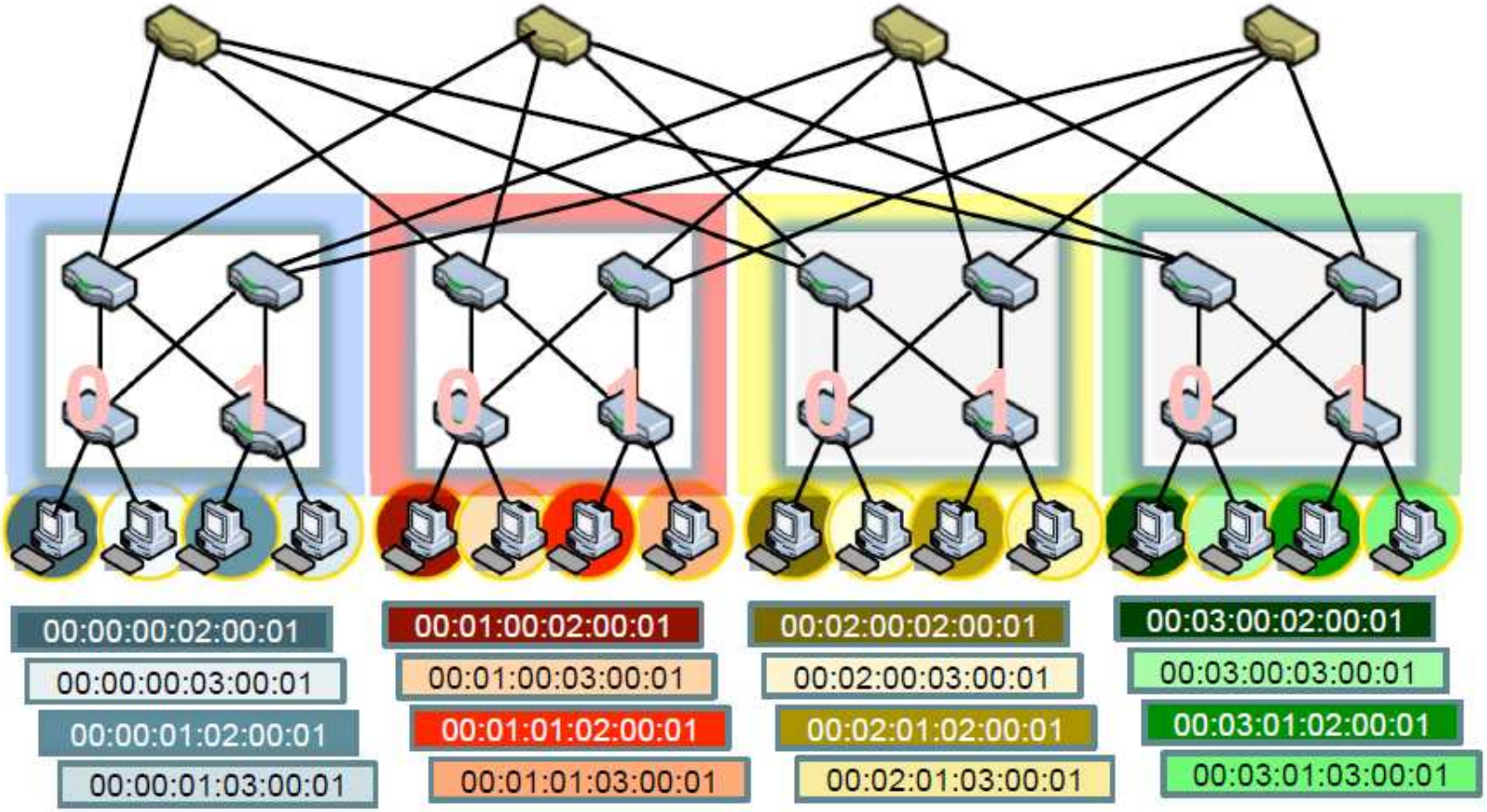
PMAC: pod.position.port.vmid

vmid – 16 bits e serve para multiplexar VMs em uma mesma máquina física



PMAC: pod.position.port.vmid

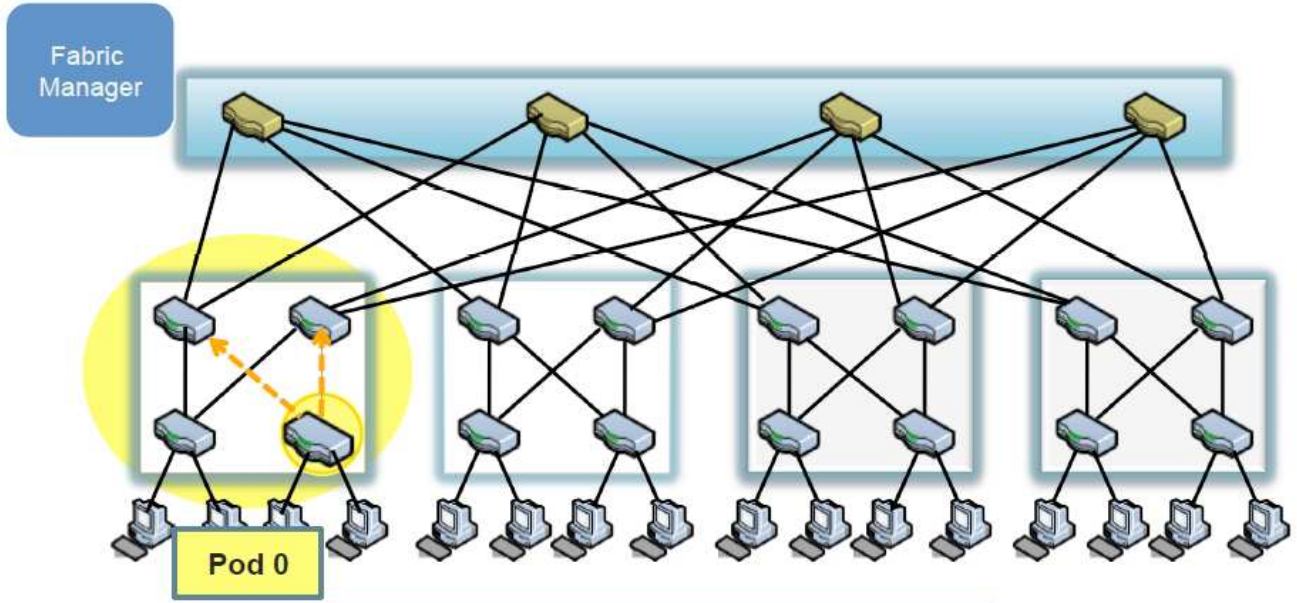
Endereços hierárquicos Pseudo MAC



pod:posição:porta:vmid

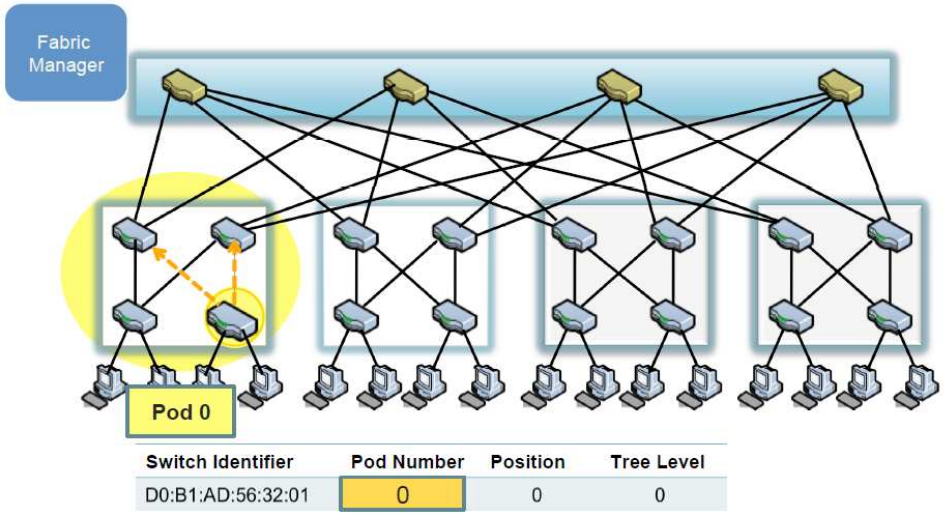
LDP – Location Discovery Protocol

- Envio periódico de LDMs (*Location Discovery Messages*) em todas as portas do switch para:
  - Definir a posição dos switches
  - Monitorar o estado das conexões fisicas

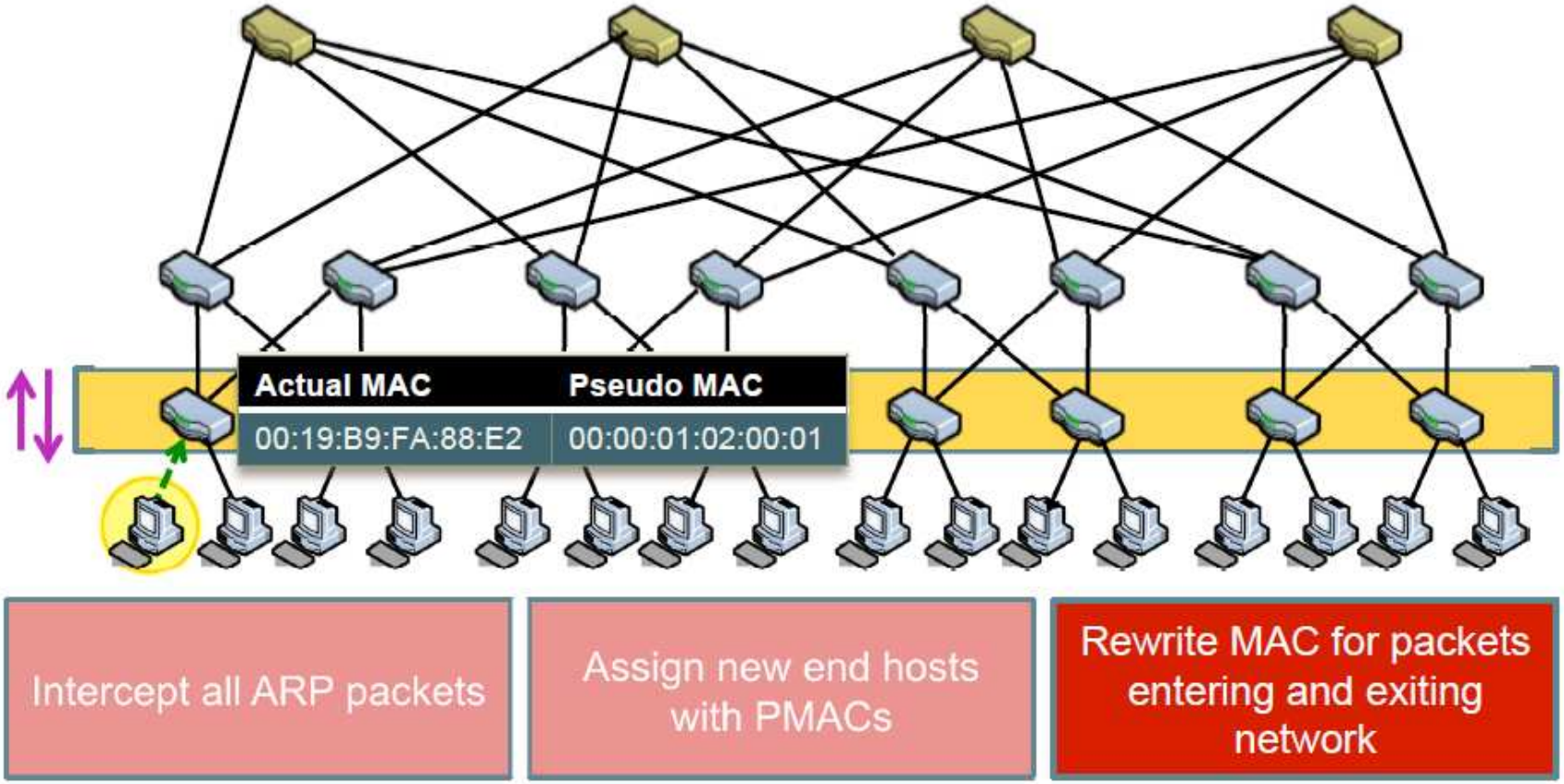


Protocolo de descoberta de posição na topologia

- Como definir quais os switches de borda (*Edge*)?
  - Recebem LDMs apenas nas portas conectadas aos switches de agregação. Servidores não geram LDMs.
  - Passam a gerar LDMs identificando seu nível (*Edge*)
- Como definir a posição do restante dos switches?
  - **Aggregation** → recebem mensagens identificadas dos switches de *Edge* em um conjunto de portas e conseguem aferir seu nível
  - ***Core*** → recebem mensagens dos switches de *Aggregation* em todas as portas

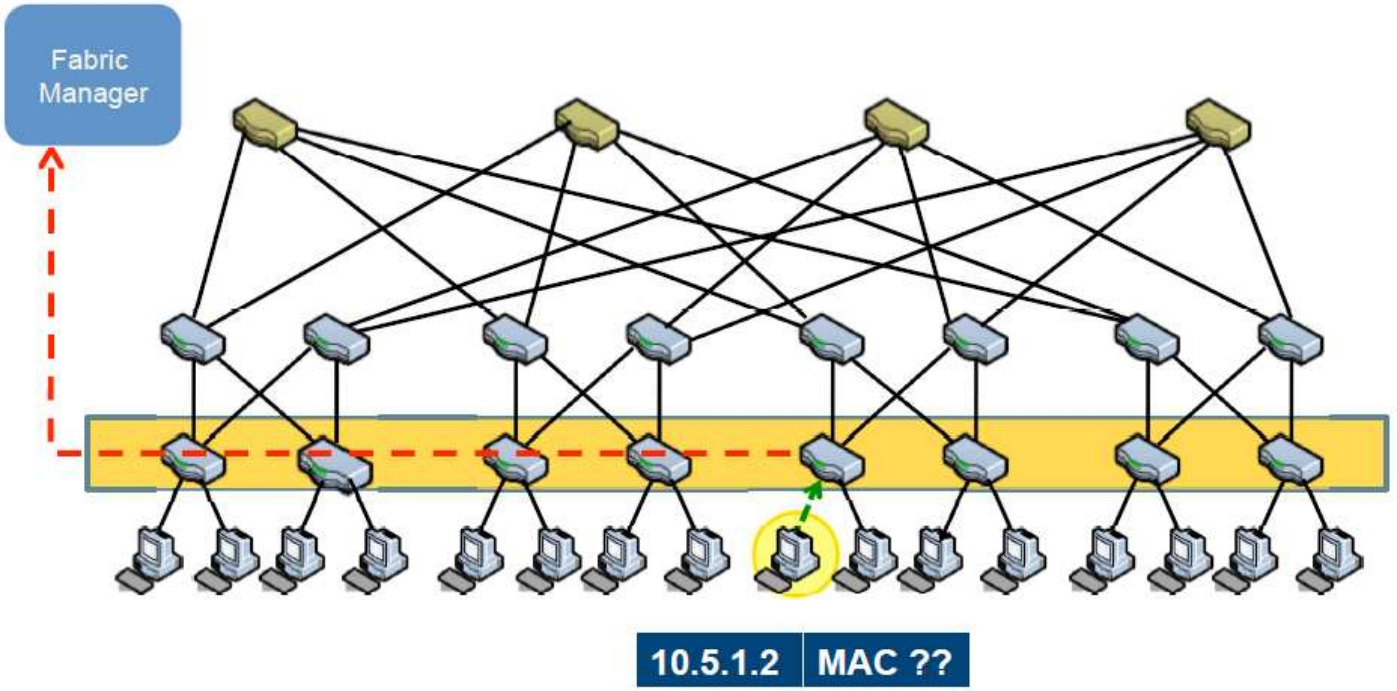


PortLand: Name Resolution



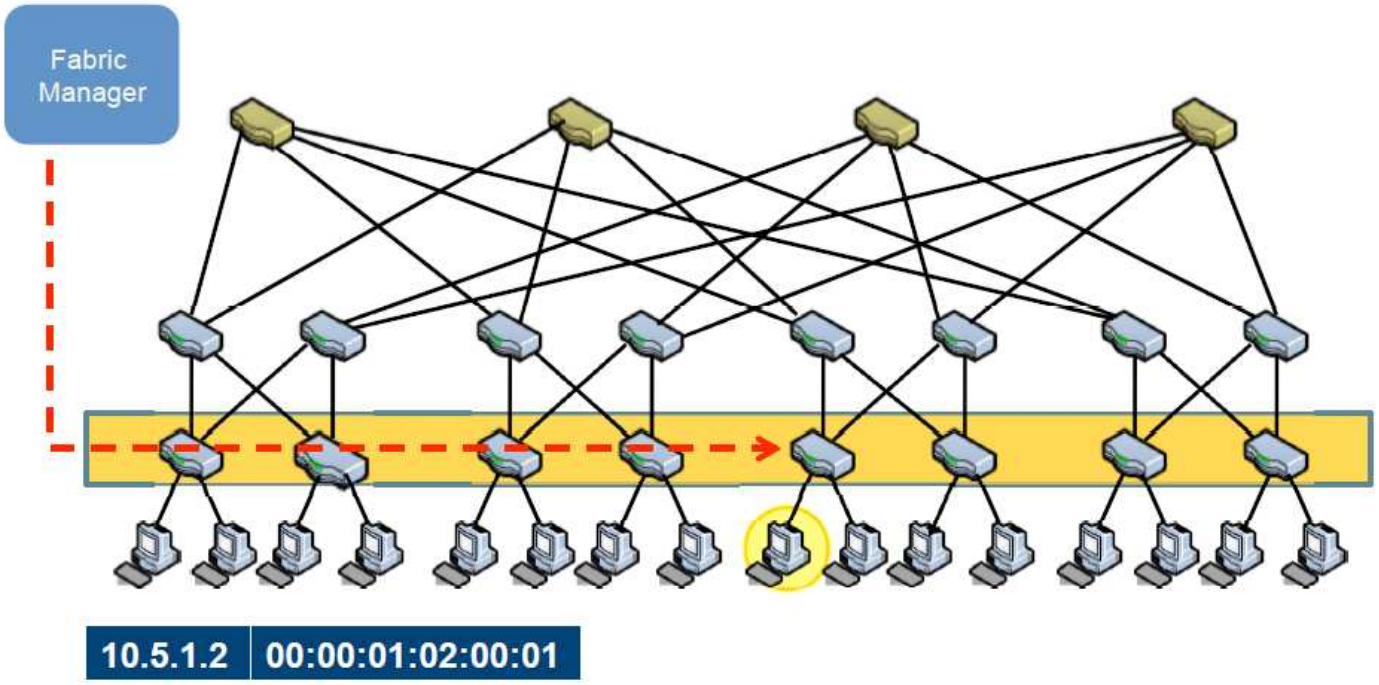
Mecanismo de proxy para requisições ARP

1. Switch de ingresso intercepta mensagem de ARP
2. Encaminha a requisição para o *Fabric Manager*



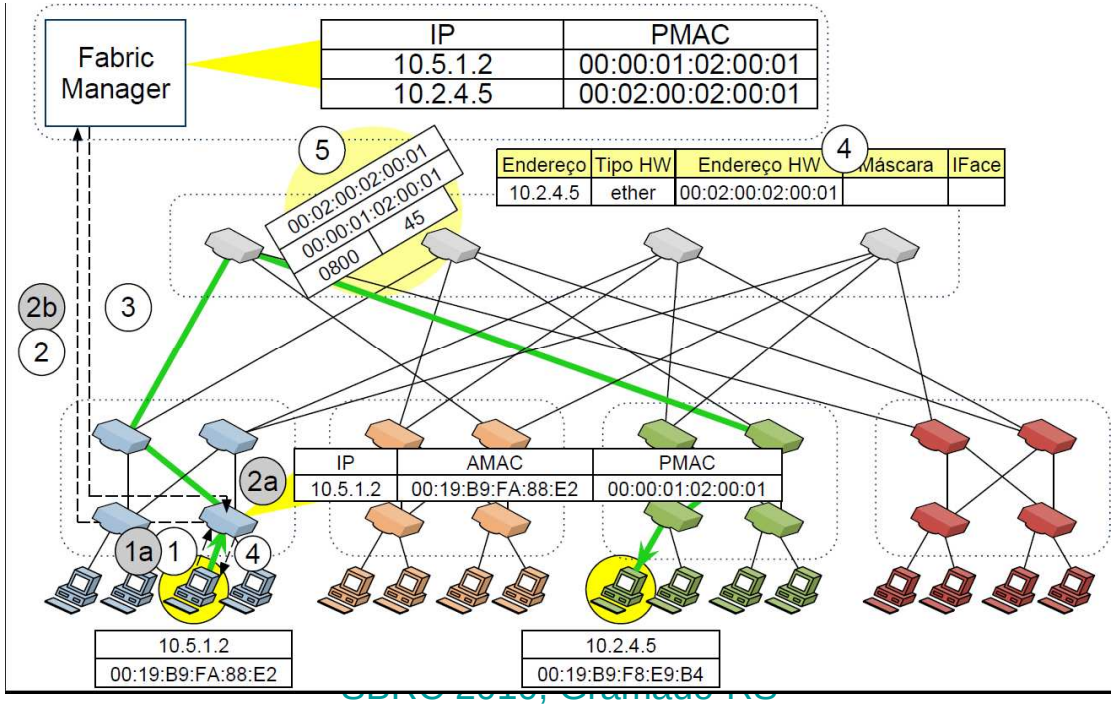
Mecanismo de proxy para requisições ARP

1. Switch de ingresso intercepta mensagem de ARP
2. Encaminha a requisição para o *Fabric Manager*
3. O *Fabric Manager* e retorna ao switch o mapeamento IP/PMAC
4. O switch e envia ao nó requisitante a resposta ARP



Mecanismo de proxy para requisições ARP

1. Switch de ingresso intercepta mensagem de ARP
2. Encaminha a requisição para o *Fabric Manager*
3. O Fabric Manager e retorna ao switch o mapeamento IP/PMAC
4. O switch e envia ao nó requisitante a resposta ARP
5. Os pacotes trafegam na rede utilizando PMACs



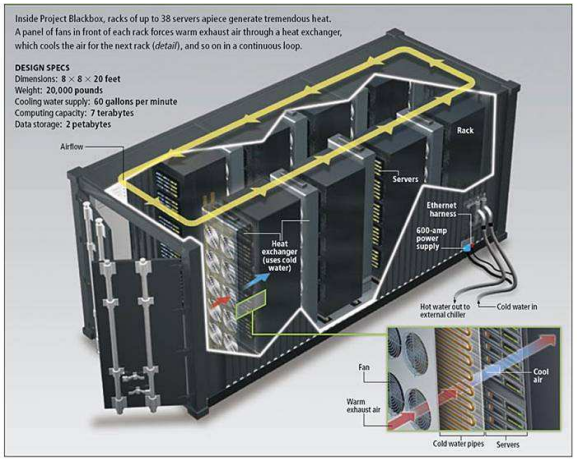
C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu.

(Microsoft Research Asia, Universities of China, UCLA)

BCUBE/MDCUBE

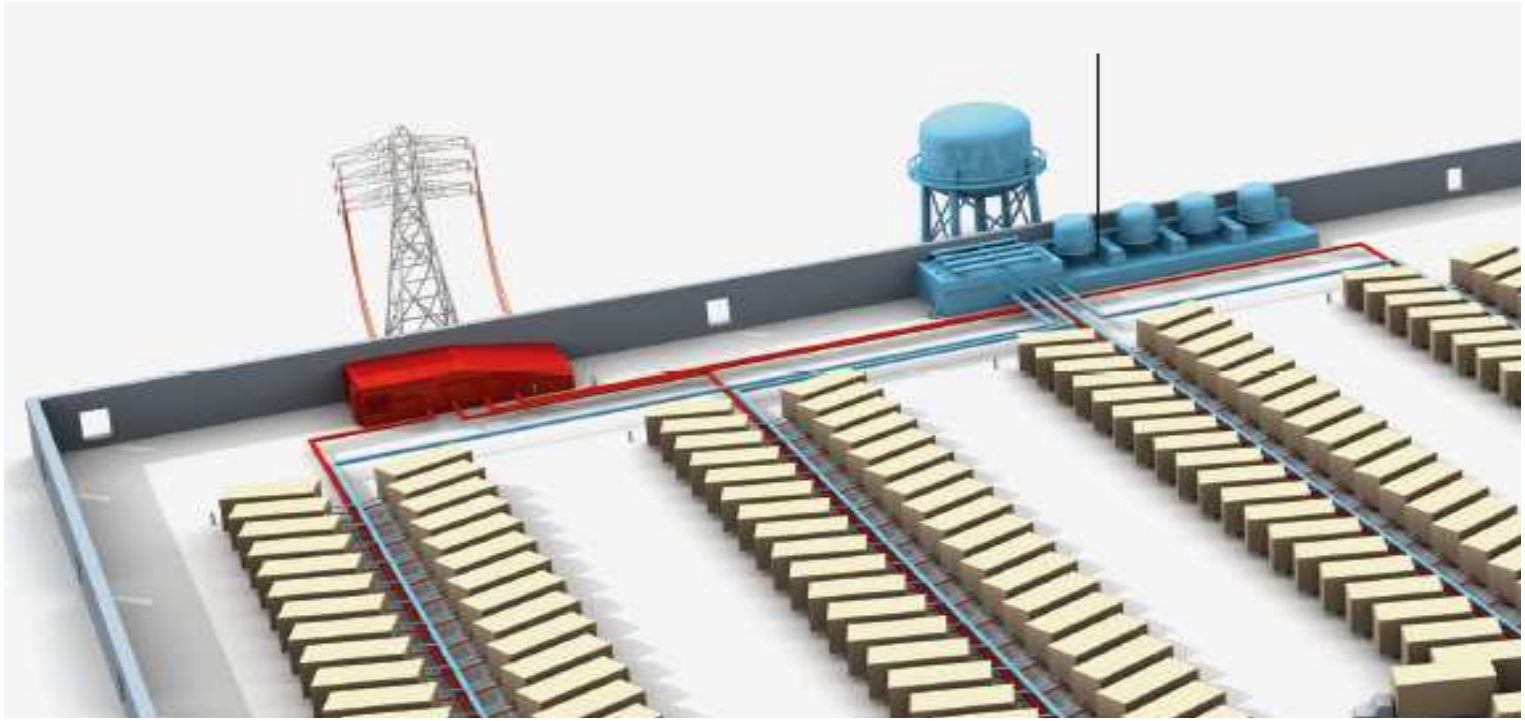
Filosofia BCube

- Utilizar contêineres para construir e implantar *data centers*. MDCs – *Modular Data Centers*
  - 1000-2000 servidores e pequenos switches empacotados em contêineres mercantes de 20 ou 40



Filosofia MDCube

- Cada BCube é visto como sendo um nó em uma estrutura de *mega data center*
  - Os diversos contêineres possuem um conjunto de portas de alta velocidade para efetuar a interconexão

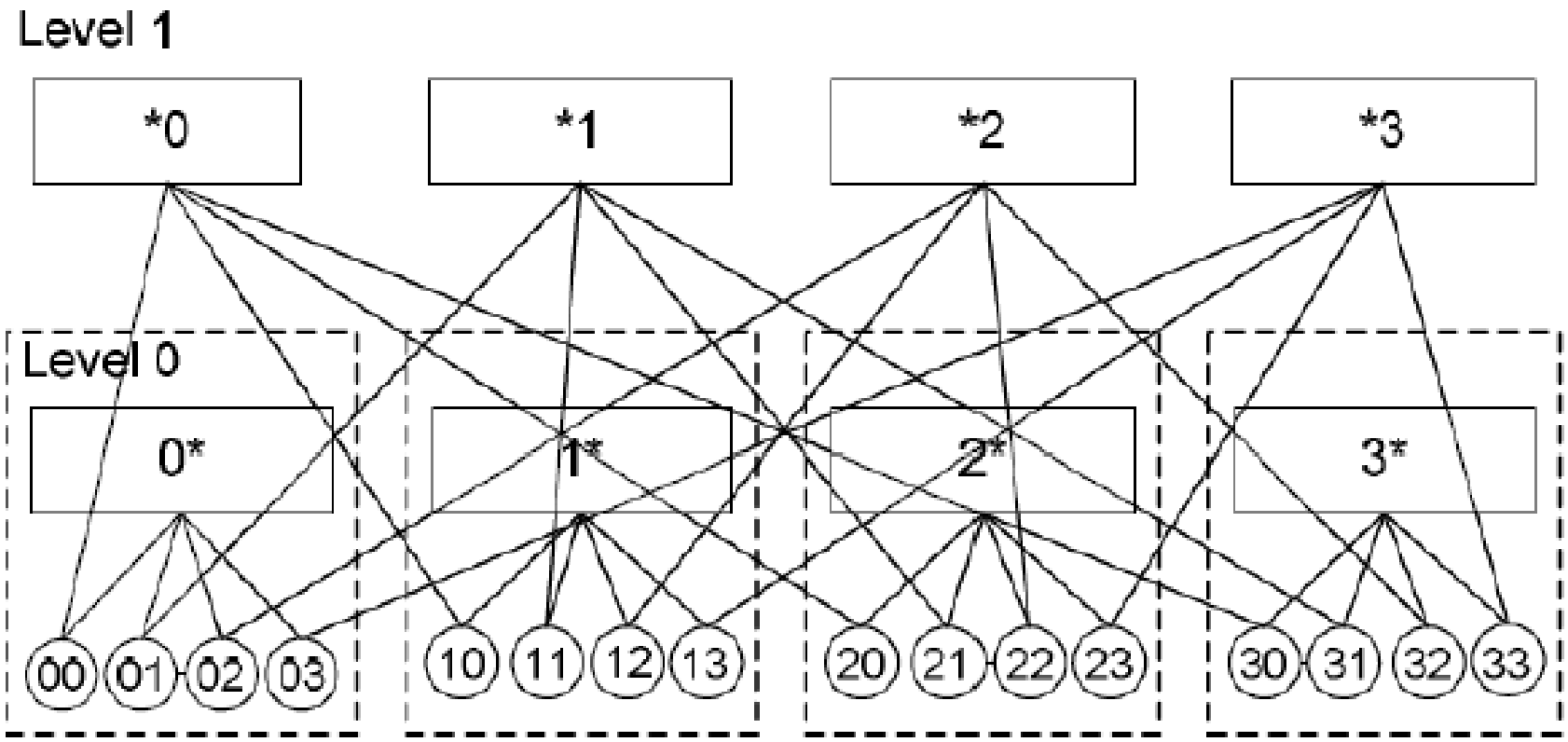


Aspectos principais

- Facilitam o posicionamento de data centers
  - Realocação geofráfica de data centers
  - Menor tempo de implantação
  - Maior densidade de equipamentos
  - Menores custos com resfriamento
- ***Server-centric***
  - Inteligência de roteamento nos servidores
  - Permite a utilização de switches comoditizados
  - Switches só interligam interfaces de servidores

Arquitetura BCube

- Utiliza servidores com multiplas interfaces de rede e switches com poucas portas
- Estrutura recursiva
  - $BCube_0$ = *n* servidores conectados a um switch de *n* portas
  - $BCube_1$ = *n* $BCubes_0$ e *n* switches de *n* portas
- Expressão genérica:
  - $BCube_k$ ($k \geq 1$) = *n* $Bcubes_{k-1}$ e n^k switches de *n* portas
  - Cada servidor em um $BCube_k$ possui *k*+1 portas

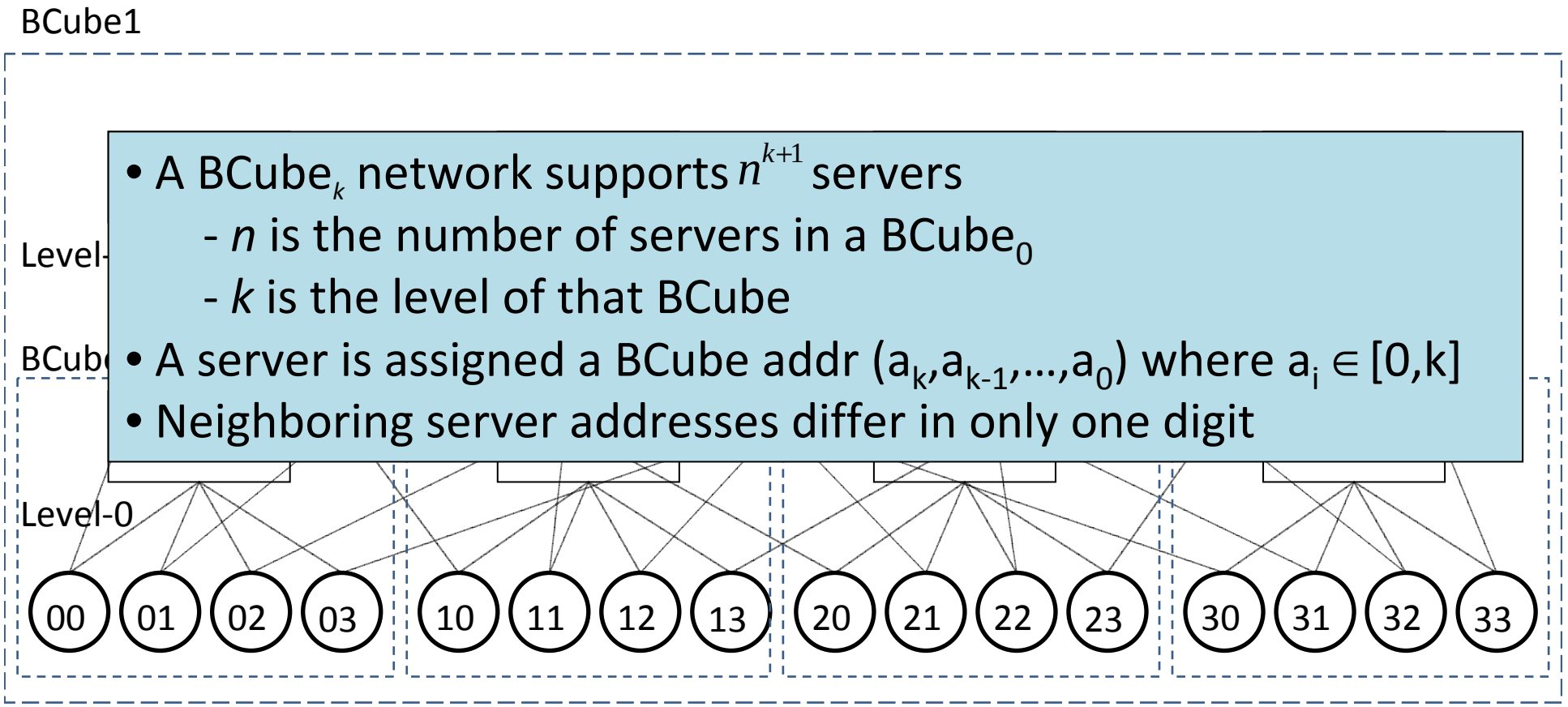


Exemplo de um BCube1 com *n = 4*.
Extraída de [Guo et al. 2009].

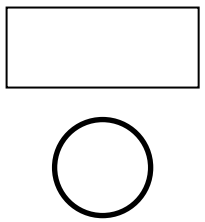
BCube parcial

- Como criar um BCube parcial?
  1. Criar todos os $BCubes_{k-1}$ desejáveis
  2. Conectar estes $BCubes_{k-1}$ utilizando uma camada *k* completa de switches

BCube structure



- A BCube$_k$ network supports n^{k+1} servers
  - *n* is the number of servers in a BCube$_0$
  - *k* is the level of that BCube
- A server is assigned a BCube addr (a_k,a_{k-1},…,a_0) where $a_i \in [0,k]$
- Neighboring server addresses differ in only one digit



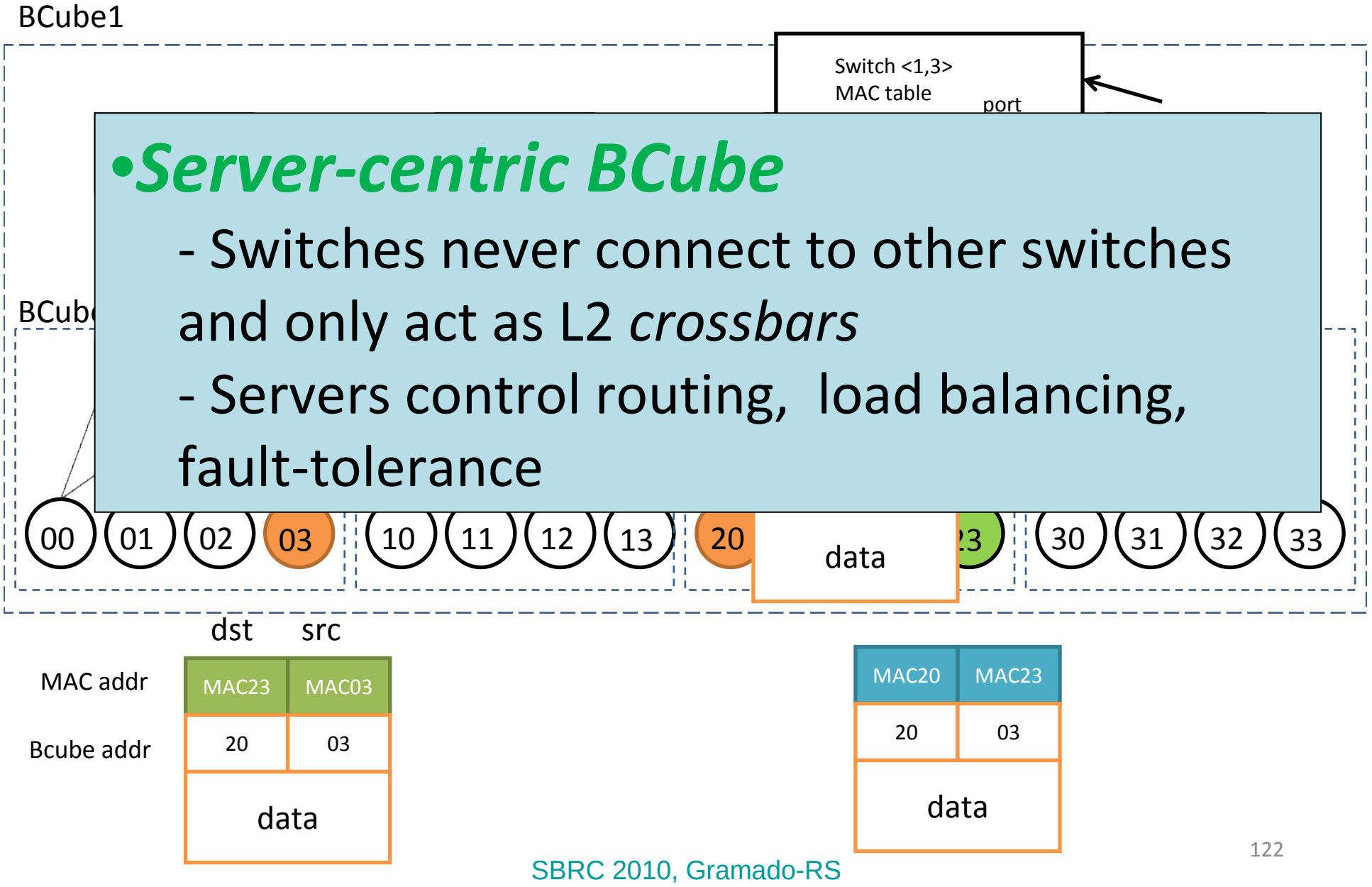
- ***Connecting rule***
  - The *i*-th server in the *j*-th BCube$_0$ connects to the *j*-th port of the *i*-th level-1 switch

Mecanismo de rota na origem do BCube

- BSR – *BCube Source Routing*
  - Utiliza mensagens de teste (*probing*) para descobrir o melhor caminho (rota na origem)
  - Path compression into a shim-header-based Next Hop Index (NHI) Array
- Por que o mecanismo de rota na origem?
  - Cada servidor de origem é livre para escolher toda a rota para seus pacotes
  - Os servidores intermediários apenas efetuam o encaminhamento dos pacotes

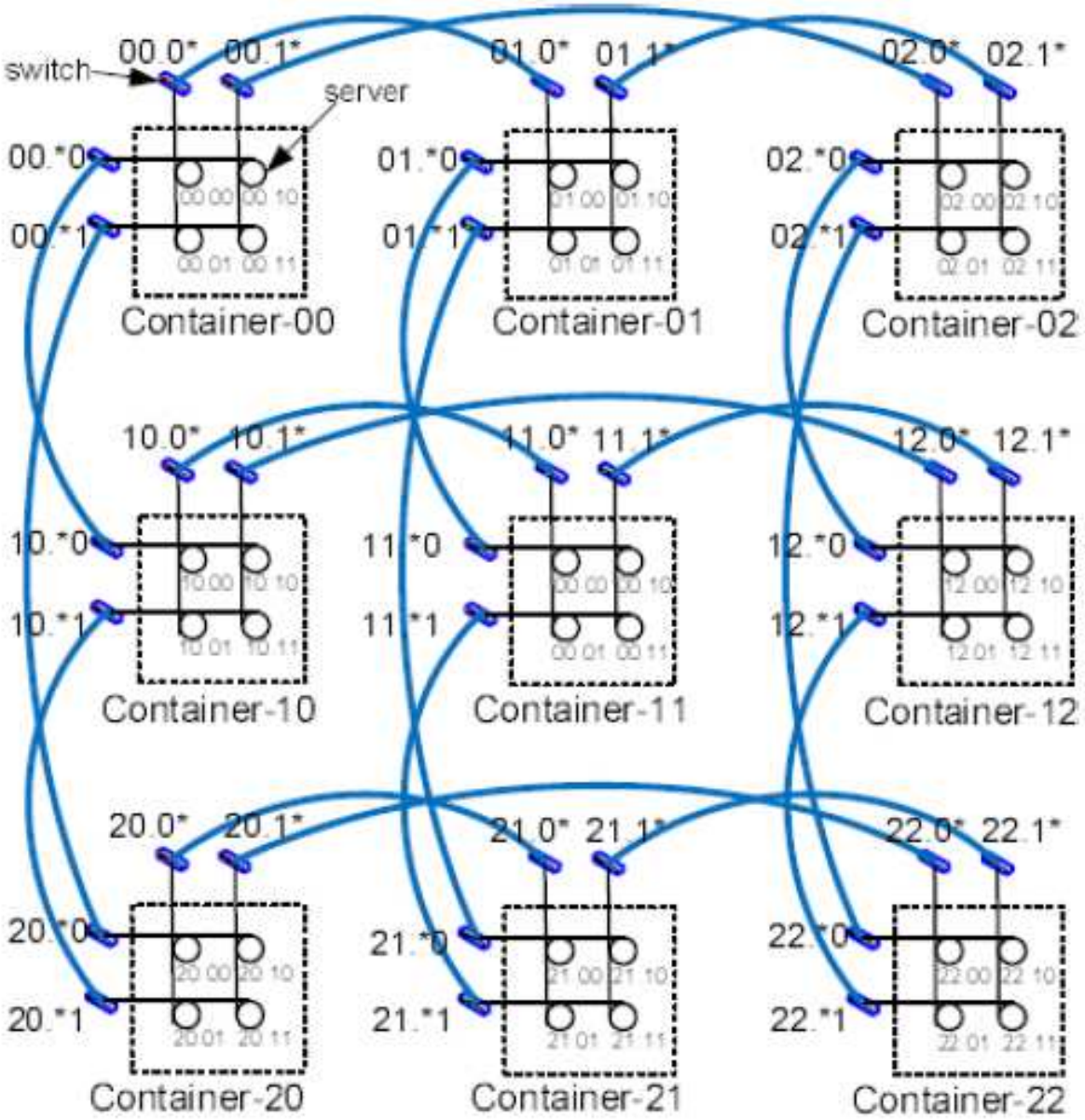
SBRC 2010, Gramado-RS

BCube: Server centric network



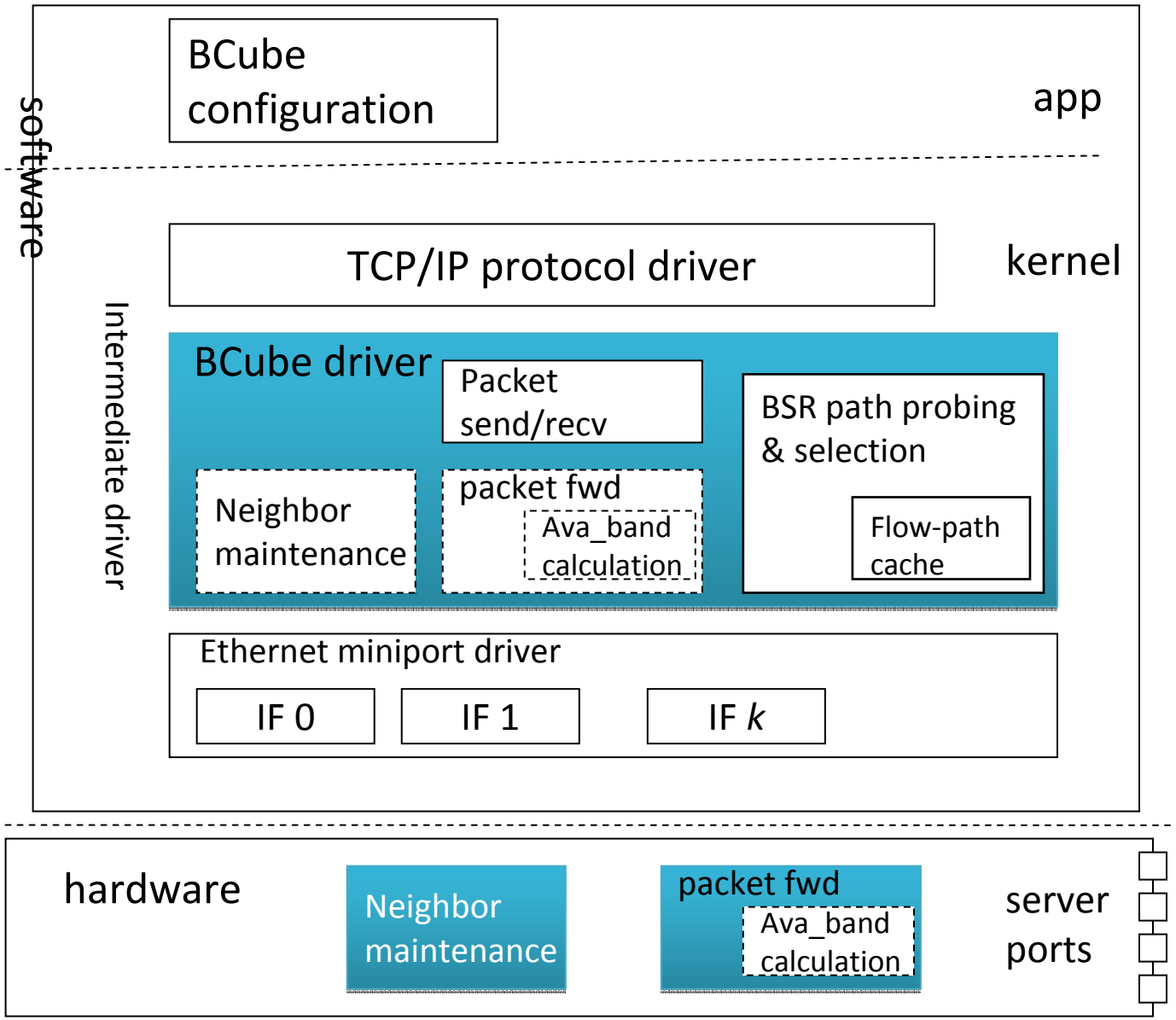
Arquitetura do MDCube

- As diversas interfaces de alta velocidade de um BCube são vistas como uma única interface virtual
  - Exemplo:
    - 1 BCube que possua 4 interfaces de 10Gbps é visto como possuidor de apenas uma interface virtual de 40Gbps
- *Server-centric*
  - Roteamento na origem (BSR) extendido para suportar o cenário de *mega data center*

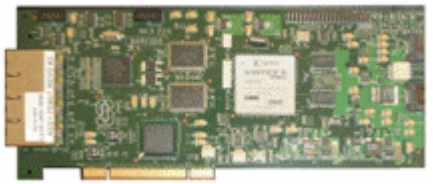


Exemplo de um MDCube de 2 dimensões formado a partir de nove (3x3) BCubes. Extraída de [Wu et al. 2009].

Implementation



Intel® PRO/1000 PT Quad Port Server Adapter



NetFPGA

RESUMO COMPARATIVO DAS ABORDAGENS

	Monsoon	VL2	Portland	BCube e MD-Cube
Realocação dinâmica de servidores (agilidade)	sim	sim	sim	sim
Transmissão à taxa máxima das interfaces/ *oversubscription*	sim (1Gbps) / 1:1	sim (1Gbps) / 1:1	sim (1Gbps) / 1:1	sim (1Gbps) / 1:1
Topologia	*fat tree*	*fat tree*	*fat tree*	hipercubo
Mecanismo de Roteamento/ Encaminhamento	tunelamento MAC-in-MAC	tunelamento IP-in-IP	baseado na posição hierárquica dos nós (PMAC)	rota na origem gerada por mensagens de *probing*
Balanceamento de Carga	VLB + ECMP	VLB + ECMP	Não especificado	trocas periódicas de *probing* modificam a rota
Modificação nos Nós Finais	sim	sim	não	sim
Modificação nos *Switches*	sim	não	sim	não
Serviço de diretório	sim	sim	sim	não

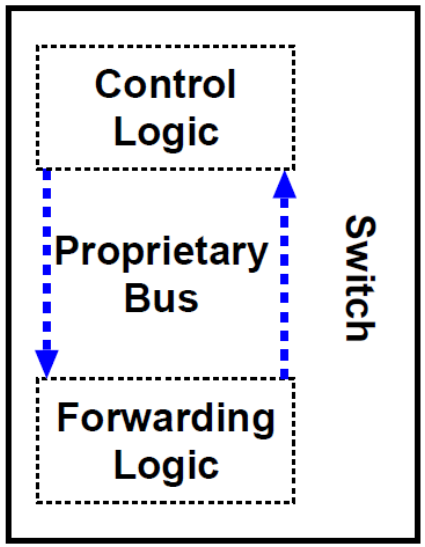
Organização do minicurso

- Introdução
- Caracterização dos *data centers* para serviços em nuvem
- Novas arquiteturas para *data centers*
  - *Monsoon*
  - *VL2*
  - *Portland*
  - *BCube e MDCube*
- Conclusão
  - Cenários futuros
  - Desafios e oportunidades

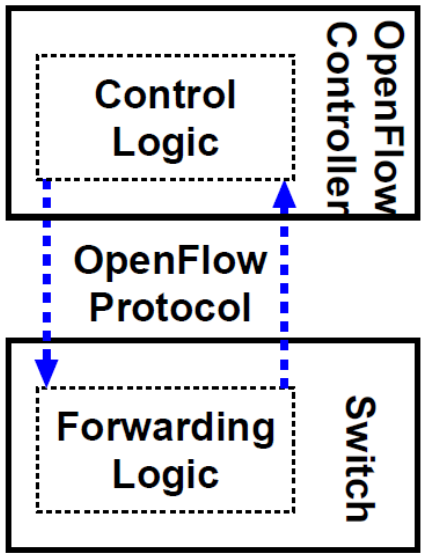
Agenda

- Tendências
  - *Software-Defined Networking* com OpenFlow
  - Arquiteturas baseadas em memória
  - Bases de dados NoSQL
  - Padrões para computação em nuvem
  - Interação entre nuvens (*Inter-Cloud)*
- Conclusões

OpenFlow: Software-Defined Networking



Classical Architecture



OpenFlow Architecture

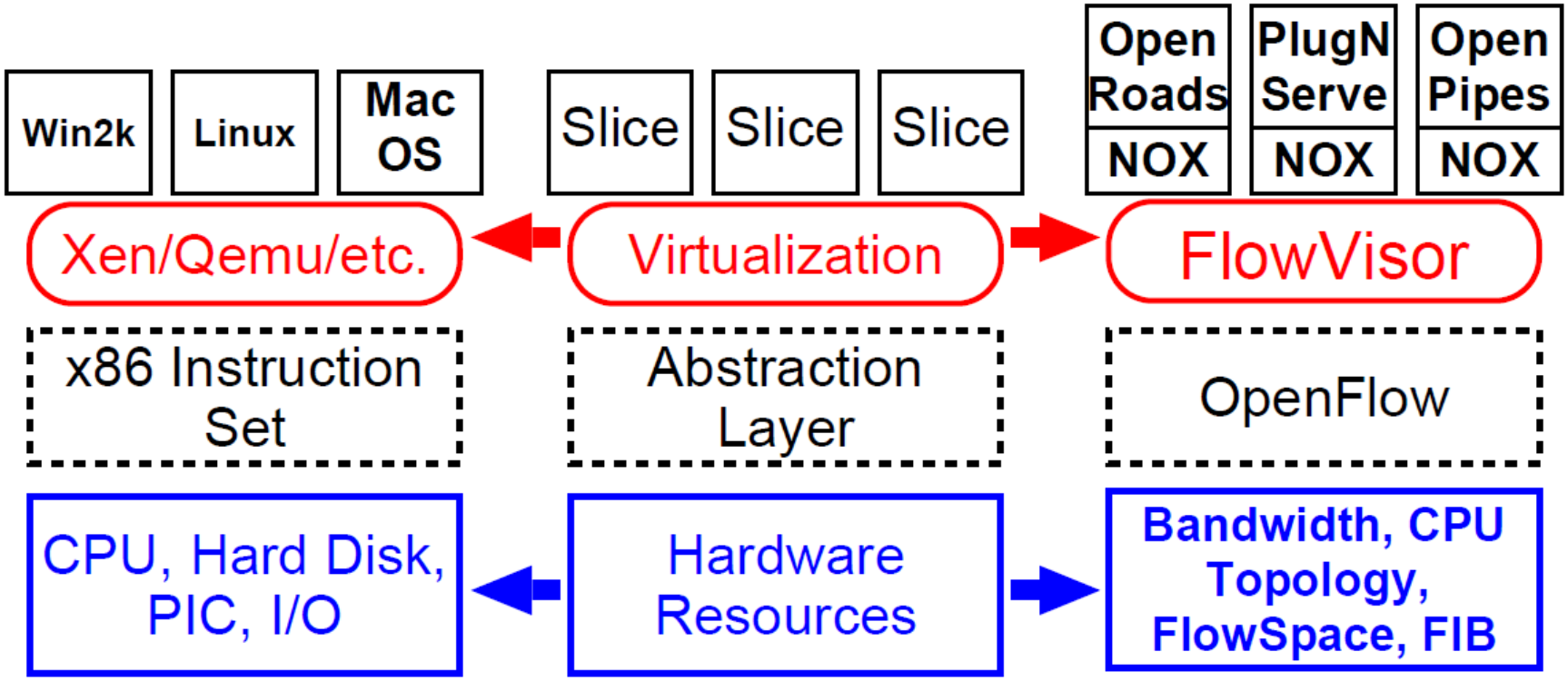
Open-source software on general-purpose computer

Open, clean hardware interface.

Same hardware (at first)

Value proposition of OpenFlow:
Simple, common, stable, hardware substrate
+ Virtualization + Open operating system + Competition above (App Store)
→ Competition at all layers + Rapid innovation

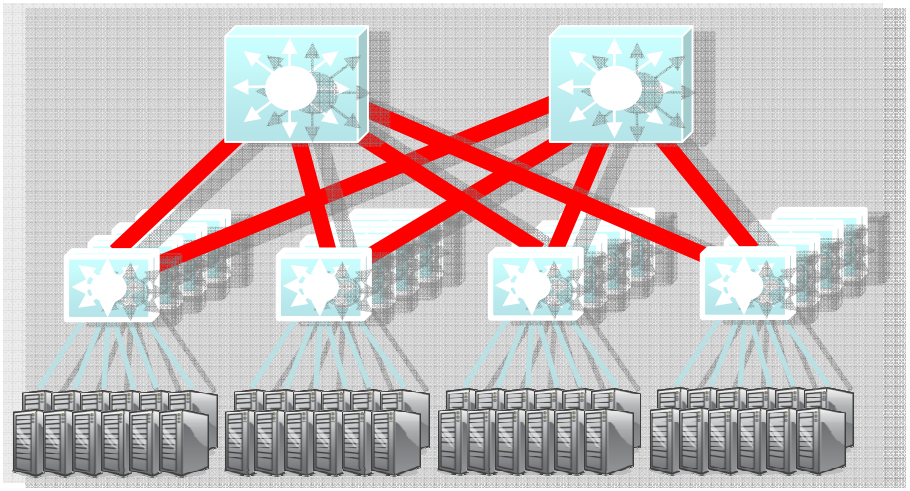
OpenFlow: Software-Defined Networking



OpenFlow: Three Stories

- A platform for innovations within
  - Enterprise, backbone, *data center networks (this short-course)*
- An architectural direction for Future Internet
  - That enables innovations *(WPEIF on 27th and 28th)*
- An architecture that unifies
  - Packet and circuit networks across wired, wireless, and optical technologies

OpenFlow in Data Center Networks



Cost

500,000 servers ⇨ 10,000 switches
$10k commercial switch ⇨ $100M
$1k custom-built switch ⇨ $10M

Savings in 10 data centers = $900M

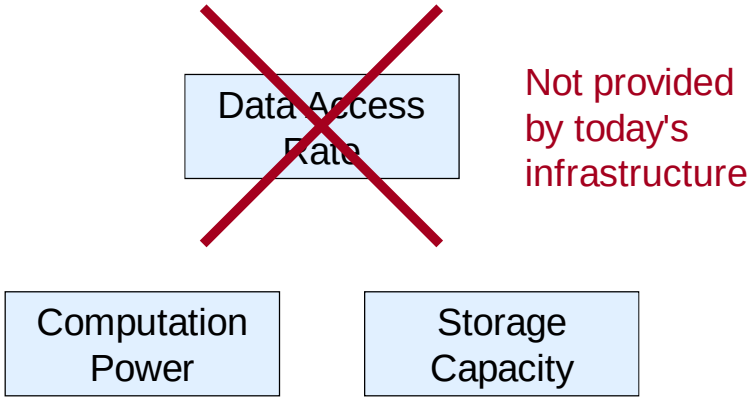
Control

1. Optimize for features needed
2. Customize for services & apps
3. Quickly improve and innovate

- Examples from academia: Portland, SiBF (SBRC´10), etc.
- Think Google (scale plus programming power)

Arquiteturas baseadas em memória: RAM is the New Disk

• Google: complete index in memory
• LinkedIn and Digg: graph of network social network in-memory
• Facebook: 800 memcached servers (28 TB, 99% cache hit rate)



- Beyond memcached (high-performance, distributed memory object caching system)
- Native memory-based data center architectures and apps.
- Many research issues:
  - FAWN [Andersen et al. 2009], Gordon [Caulfield et al. 2009], Stanford´s RAMCloud [Ousterhout et al. 2009]
  - [http://highscalability.com/are-cloud-based-memory-architectures-next-big-thing]

Bases de dados NoSQL

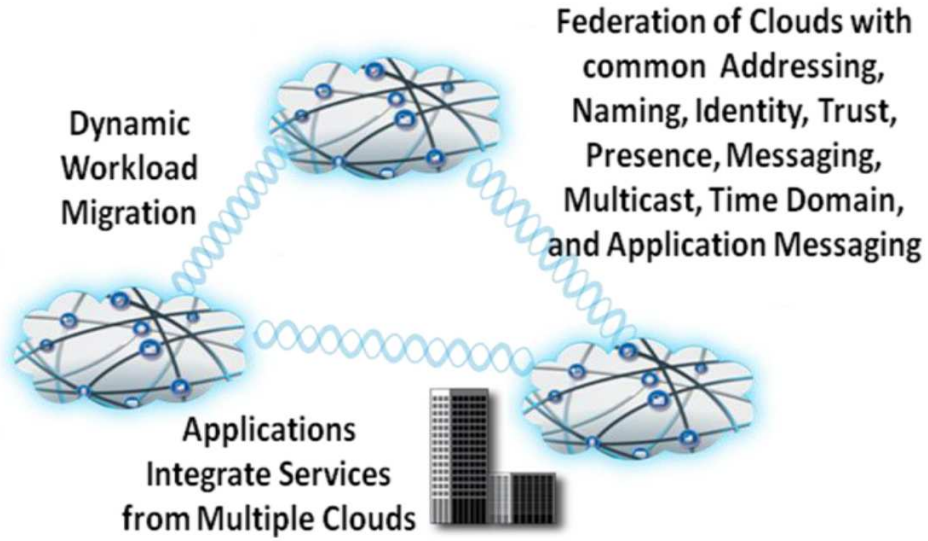
- Focus on particular classes of problems:
  - more flexible about stored data (document stores)
  - relationships (graph databases)
  - data aggregation (column databases)
  - simplifying database job to simply storing a value (key/value stores).
- SQL databases still retain their advantages in terms delivering ACID and relational capabilities
- NoSQL databases represent is a new disruptive force in the data storage and retrieval business.

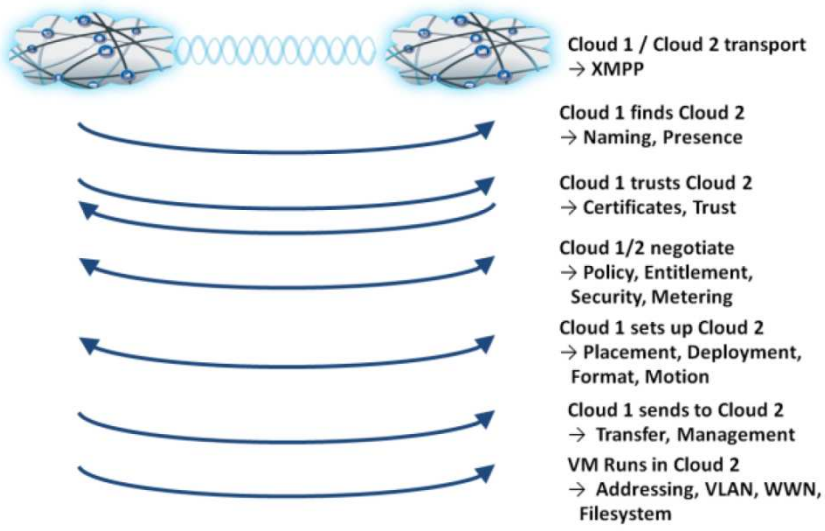
- **Move towards a more mixed and much richer data storage and retrieval ecosystem.**

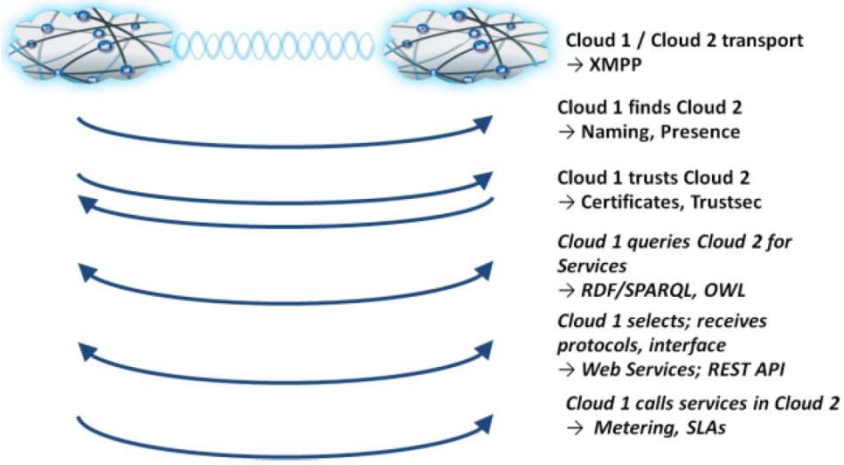
[http://nosql-database.org/]


Inter-Cloud

Inter-Cloud

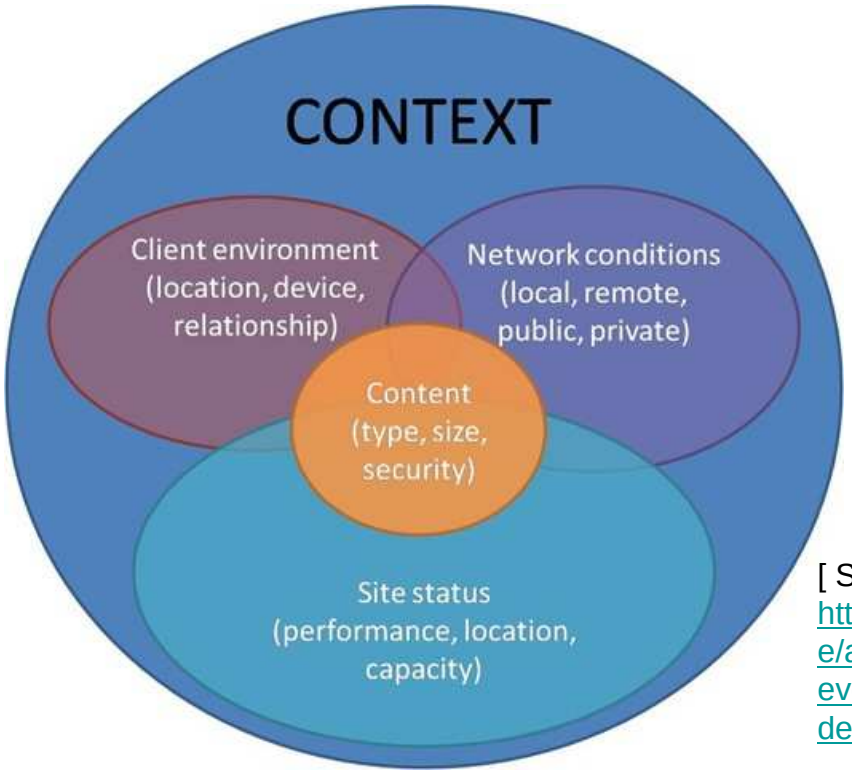






Bernstein *et al.* "Blueprint for the Intercloud - Protocols and Formats for Cloud Computing Interoperability,"

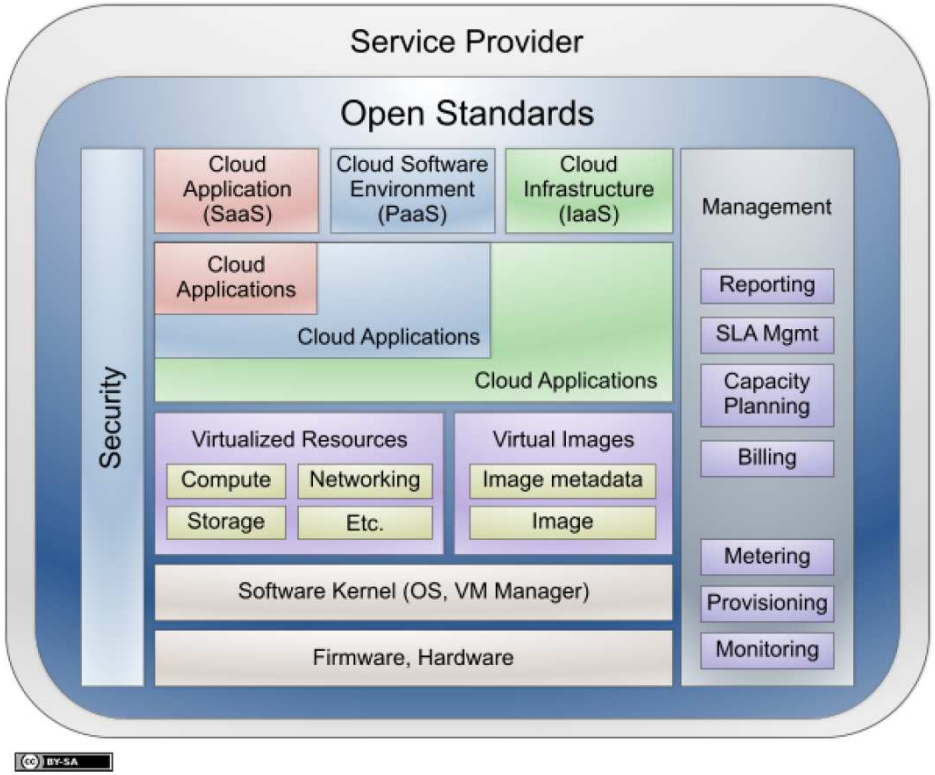
Research in the Inter-Cloud



[Source: http://devcentral.f5.com/weblogs/macvittie/archive/2009/06/30/intercloud-the-evolution-of-global-application-delivery.aspx]

- It is all about *context* (again, i.e., like mobility) but strongly driven by business (aka BGP but on a higher level)
- Expect the emergence of a market and cloud resource brokers

Padrões para computação em nuvem

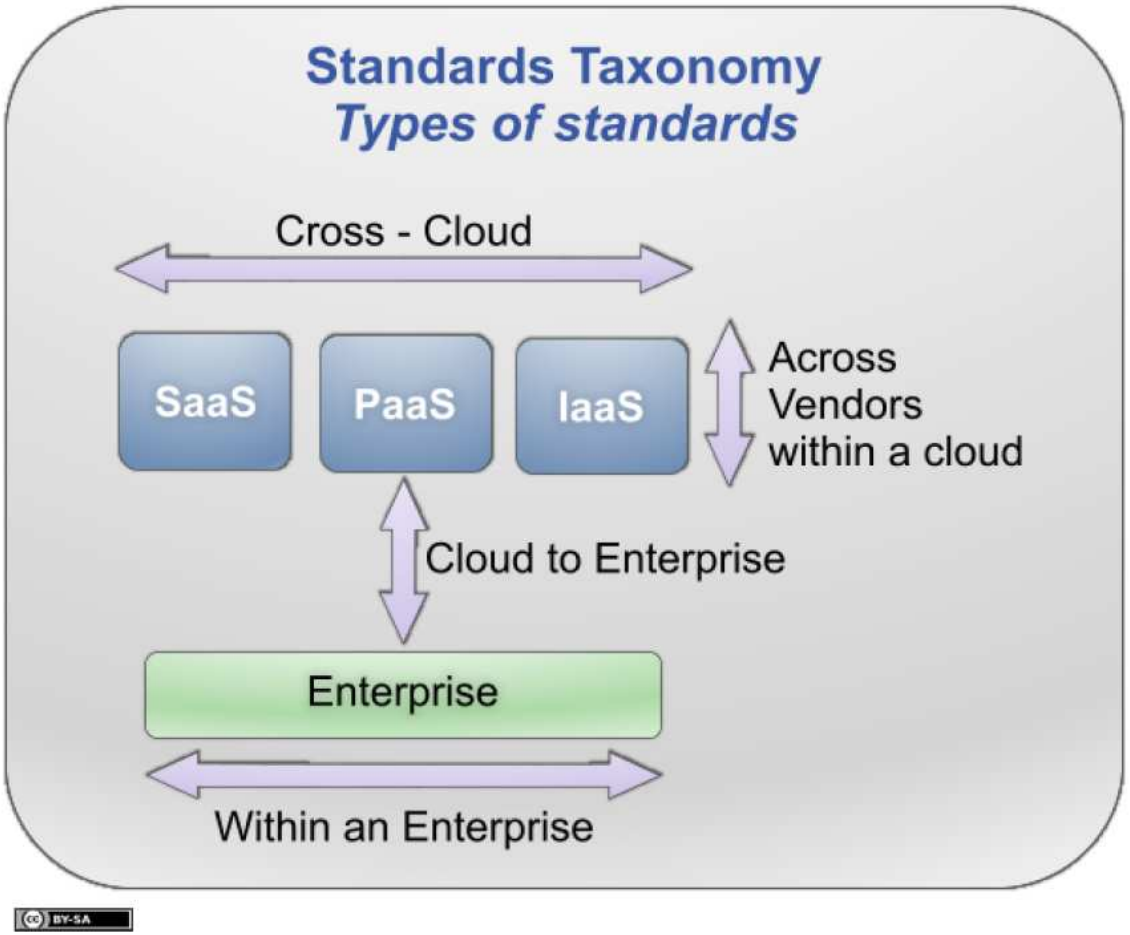


- VM image formats and metadata
- API to storage, DB, etc.
- Naming
- Security
- ...

[Source: http://www.opencloudmanifesto.org/Cloud_Computing_Use_Cases_Whitepaper-2_0.pdf]

- Open Source developments (Eucalyptus, Ubuntu Enterprise Cloud)
  and de facto APIs (e.g., Amazon EC2 and S3)

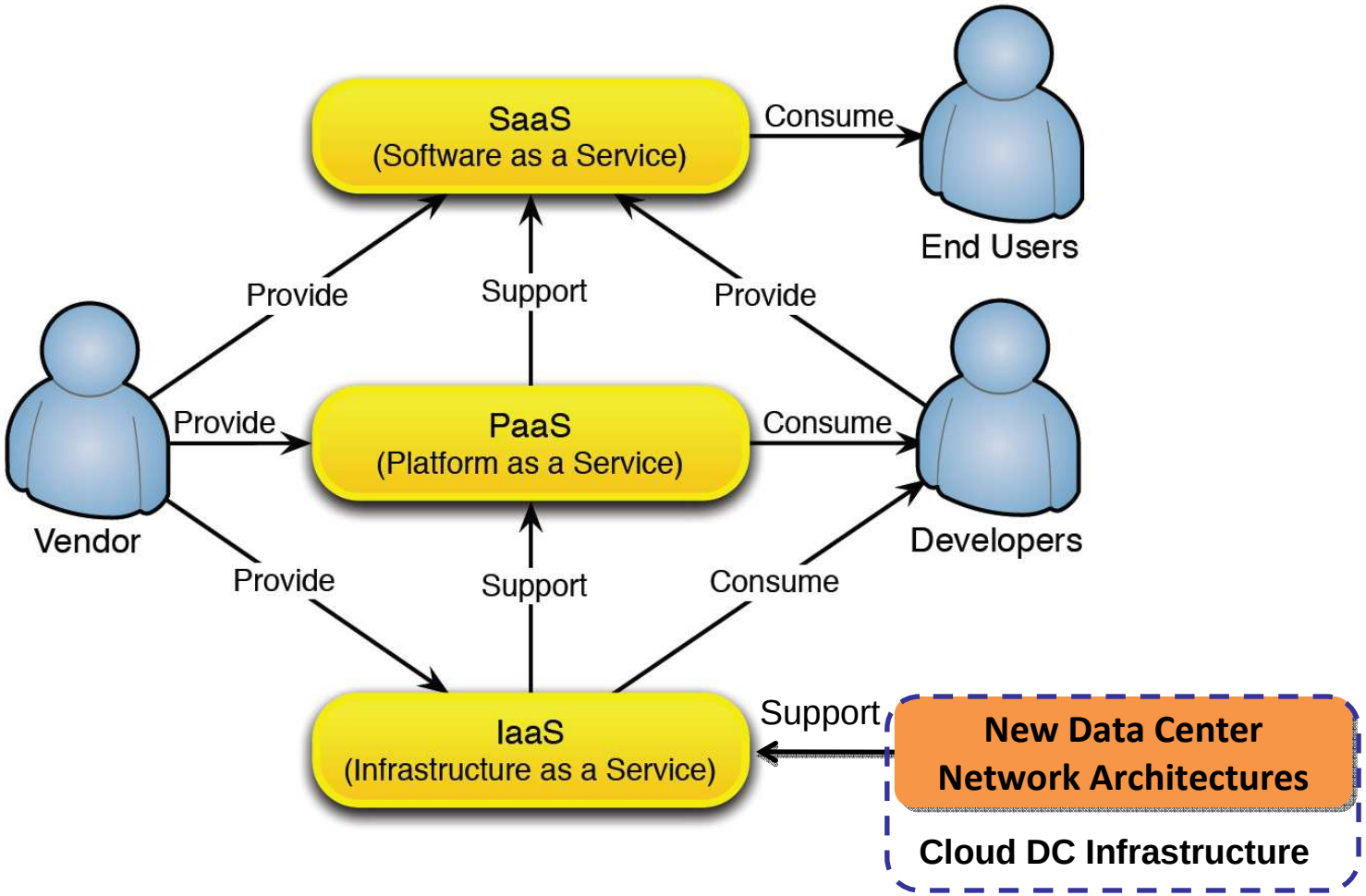
Padrões para computação em nuvem



[Source: http://www.opencloudmanifesto.org/Cloud_Computing_Use_Cases_Whitepaper-2_0.pdf]

Conclusões

Networking for the Cloud



Conclusões

- Modelo evolucionário
- Há muitos questionamentos: segurança?
- Mas há também muitas oportunidades
- Comunidade de redes no Brasil está se inserindo e atuando neste tema
- Administrações públicas, indústria, órgãos do governo, ISPs devem começar a usar e oferecer serviços em nuvem
- Não há ainda oferta real de cloud computing no Brasil

Agradecimentos

- Aos co-autores Maurício e Rafael
- Ericsson, CNPq, FAPESP
- Albert Greenberg e todas as outras fontes utilizadas no minicurso

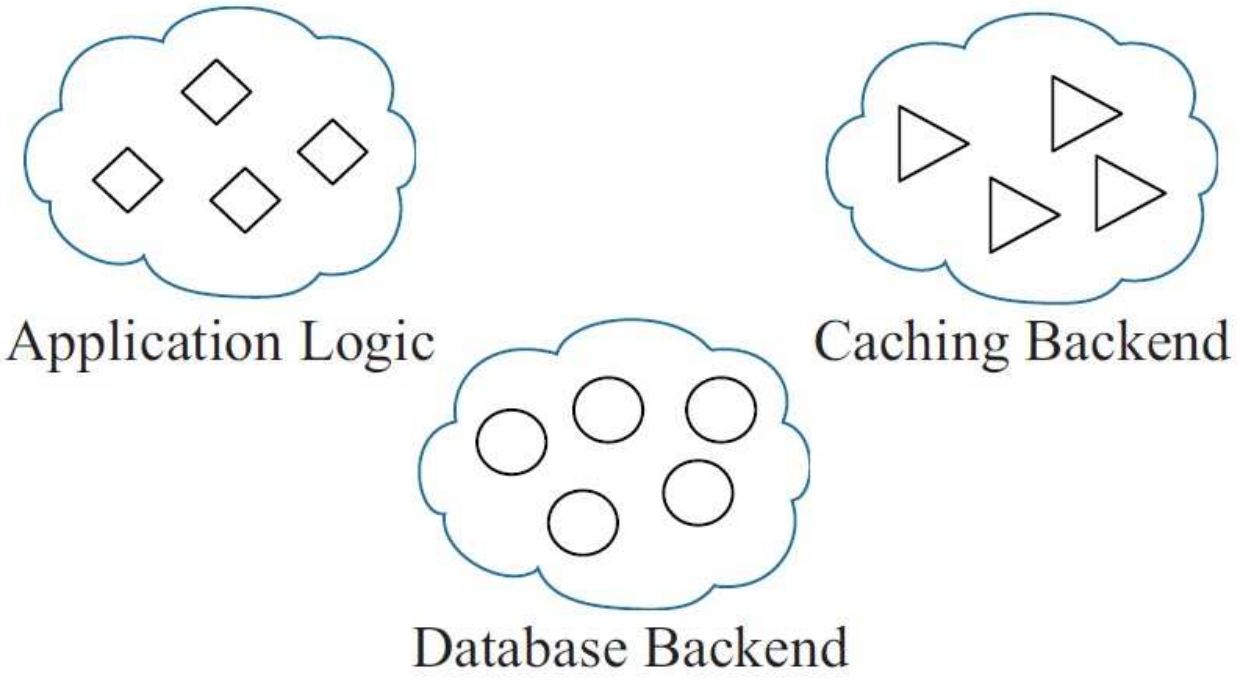
http://www.sor.ufscar.br/~verdi

http://www.dca.fee.unicamp.br/~chesteve

BACKUP

Typical Architecture

(Each Layer - Separate Cloud)



Equipamento de rede convencional

Modular routers

- Chassis $20K
- Supervisor card $18K

Ports

- 8 port 10G-X - $25K
- 1GB buffer memory
- Max ~120 ports of 10G per switch

Total price in common configurations: $150-200K (+SW&maint)

Power: ~2-5KW

Integrated Top of Rack switches

- 48 port 1GBase-T
- 2-4 ports 1 or 10G-x
- $7K

Load Balancers

- Spread TCP connections over servers
- $50-$75K each
- Used in pairs

Latencia

- Propagation delay in the data center is essentially 0
- –Light goes a foot in a nanosecond; 1000’ = 1 usec

- End to end latency comes from
- –Switching latency
- 10G to 10G:~ 2.5 usec(store&fwd); 2 usec(cut-thru)
- –Queueinglatency
- Depends on size of queues and network load

- Typical times across a quiet data center: 10-20usec
- Worst-case measurement (from our testbed, not real DC, with all2all traffic pounding and link util> 86%): 2-8 ms
- Comparison:
- –Time across a typical host network stack is 10 usec
- –Application developer SLAs > 1 ms granularity

Server Costs

- **Ugly secret: 10% to 30% utilization considered “good” in DCs**
- Causes:
- •Uneven application fit:
- –Each server has CPU, memory, disk: most applications exhaust one resource, stranding the others
- •Long provisioning timescales:
- –New servers purchased quarterly at best
- •Uncertainty in demand:
- –Demand for a new service can spike quickly
- •Risk management:
- –Not having spare servers to meet demand brings failure just when success is at hand

- If each service buys its own servers, the natural response is hoarding

Portland evaluation

Measurements	Configuration	Results
Network convergence time	Keepalive frequency = 10 ms Fault detection time = 50 ms	65 ms
TCP convergence time	RTO_{min} = 200ms	~200 ms
Multicast convergence time		110ms
TCP convergence with VM migration	RTO_{min} = 200ms	~200 ms – 600 ms
Control traffic to fabric manager	27,000+ hosts, 100 ARPs / sec per host	400 Mbps
CPU requirements of fabric manager	27,000+ hosts, 100 ARPs / sec per host	70 CPU cores