MO655 - Comunicação em Datacenters

Leandro Souza da Silva - RA: 191082
Luís Felipe Mattos - RA: 107822

15 de Dezembro de 2016

Conteúdo

1 Introdução **2**
1.1 Requisitos de Rede . . . 3
1.1.1 Escalabilidade . . . 3
1.1.2 Tolerância a Falhas . . . 4
1.1.3 Latêcia . . . 4
1.1.4 Capacidade da Rede . . . 4
1.1.5 Virtualização . . . 4

2 Motivação **5**

3 Topologias **8**
3.1 Tradicionais . . . 8
3.1.1 Baseadas em Árvores . . . 8
3.1.2 Recursivas . . . 10
3.2 SDN . . . 14

4 Protocolos **17**

5 Tendências **18**

6 Conclusão **19**

7 Referências **20**

1

Introdução

Com o crescimento da demanda dos usuários por poder computacional e armazenamento, cada vez mais as grandes empresas estão investindo em estruturas próprias de datacenters. Esta estrutura inclui tanto os computadores em si, os discos de armazenamento e os racks como também inclui a própria sala que ficarão estes racks. Estas salas devem tem uma arquitetura própria, como por exemplo, o piso elevado, sistema de refrigeração e circulação de ar. Um exemplo por ser visto na figura 1.1.

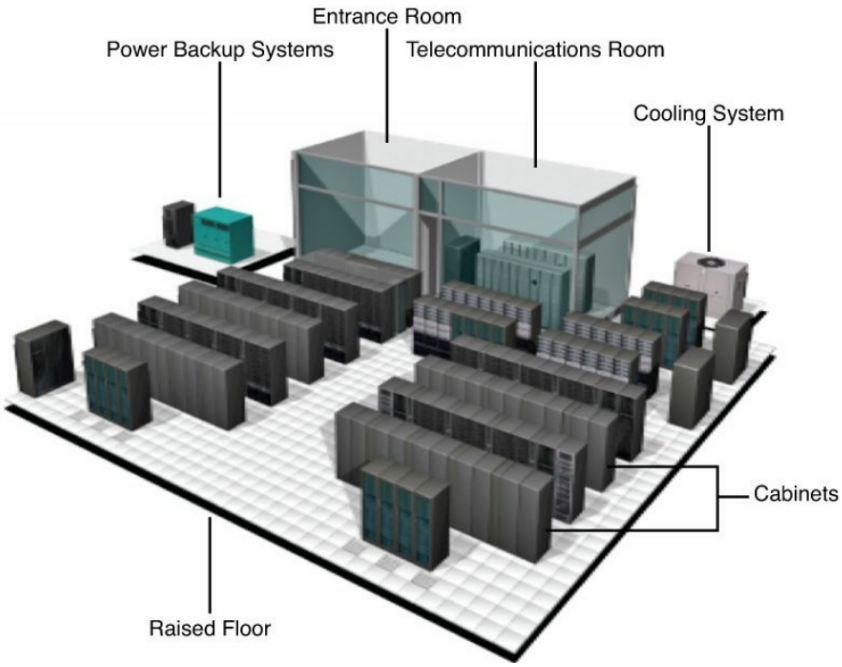

Figura 1.1: Visão geral de um Datacenter

Além da estrutura física, a parte computacional, principalmente relacionada à comunicação interna e externa do datacenter possui alguns requisitos básicos para que possa oferecer um serviço de qualidade para os usuários. Estes requisitos são citados a seguir:

- Escalabilidade
- Tolerância a Falhas

- Latência
- Capacidade da Rede
- Virtualização

A seguir, os requisitos citados serão mais detalhados.

1.1 Requisitos de Rede

1.1.1 Escalabilidade

O sistema deve ser construido de tal forma que seja possível haver uma expansão, caso a demanda aumente. Este requisito diz respeito tanto ao hardware como ao software. Para o hardware, a estrutura das máquinas deve permitir que o sistema seja melhorado e também deve haver espaço físico para a inclusão de novas máquinas. Atualmente, existem alguns sistemas modulares que possuem uma fácil integração de novos módulos.

Um exemplo é a utilização de datacenters em containers, cada container possui um sistema completo com refrigeração própria e é facilmente transportado. Com isso, pode-se expandir facilmente uma estrutura de um datacenter. Um exemplo de container pode ser visto na figura 1.2.

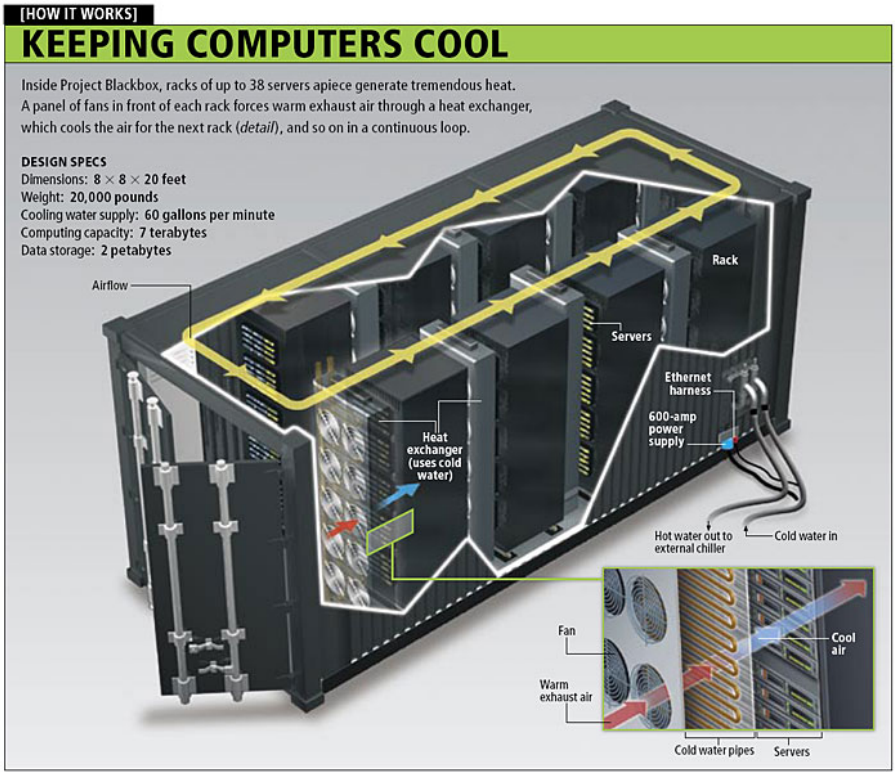

Figura 1.2: Datacenter em container

1.1.2 Tolerância a Falhas

O sistema deve ser capaz de prevenir e corrigir falhas. Por causa disso, a maioria dos sistemas de datacenters possuem redundâncias em quase todos aspectos do datacenter. Existem backups dos dados dos usuários, a comunicação interna é feita de modo que existam vários caminhos possíveis da fonte para o destino e além disso, alguns datacenters possuem backups em outros datacenters. Apesar disso, existe o custo de manter estas cópias atualizadas.

Mais a frente falaremos um pouco mais sobre as redundâncias dos caminhos de comunicação interna dos datacenters, tanto relacionados à topologia como relacionado aos protocolos de comunicação e roteamento.

1.1.3 Latêcia

Um dos principais desafios dos datacenters é possuir baixa latência, assim, a performance do sistema como um todo se mantém em um nível aceitável pelos usuários. Para isso, a topologia é muito importante, uma vez que quanto menor o caminho entre a fonte e o destino, menor a latência. Porém, outro fator que influencia muito a latência é o nível de congestionamento da rede, mais a frente iremos tratar sobre os protocolos e como estes controlam o nível de congestionamento da rede.

1.1.4 Capacidade da Rede

A capacidade da rede está quase que diretamente ligada à latência, quanto maior a capacidade da rede, menor a latência. O requisito é que a capacidade da rede seja suficiente para que atinja a demanda de forma que o desempenho e a qualidade de serviços não sejam afetados, independente da quantidade de usuários.

1.1.5 Virtualização

Outro requisito é a virtualização. Este requisito é relacionado à escalabilidade, onde permite que haja uma capacidade elástica da rede e além disso, que o sistema seja capaz de mover sistemas virtualizados de uma máquina física para outra, sem que haja problemas de compatibilidade. Isso só é possível com a virtualização, que executa o mesmo sistema em cima da camada de hardware de modo transparente para os usuários.

2

Motivação

O crescimento da demanda dos usuários tem sido exponencial nos últimos anos, o gráfico da figura 2.1. Com isso, cada vez mais as empresas precisam investir na expansão e atualização dos datacenters para acompanhar a demanda. A figura 2.2 mostra o crescimento do número de racks com servidores entre 2011 e 2013 no leste do Estados Unidos.

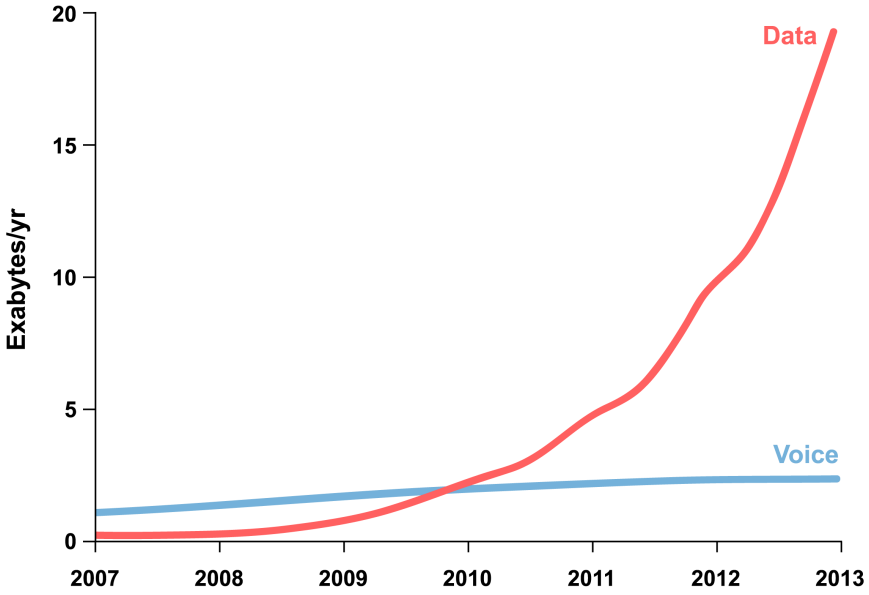

Figura 2.1: Crescimento do consumo de dados

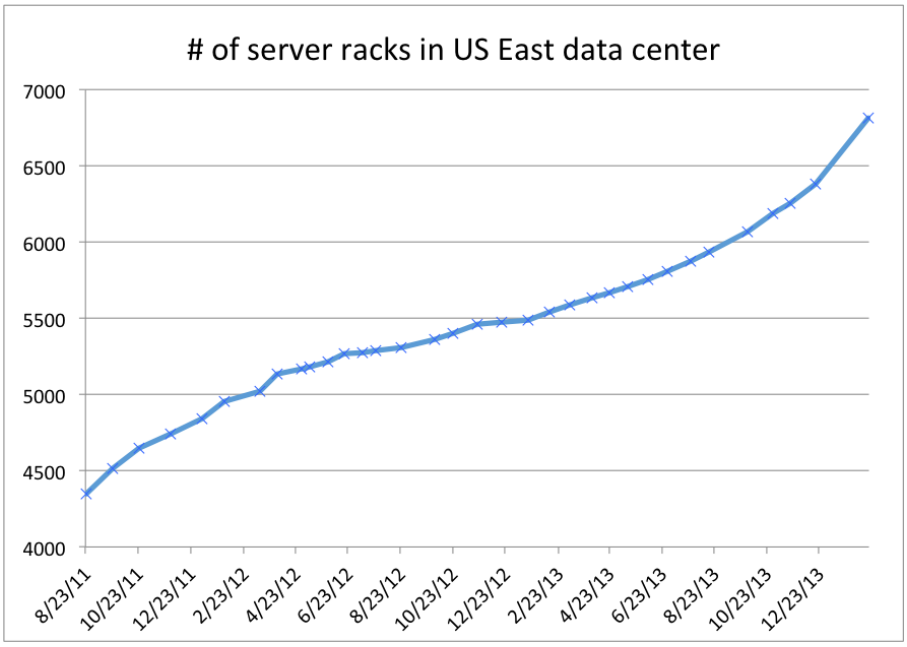

Figura 2.2: Crescimento do número de servidores

Apesar desse crescimento exponencial, o custo não aumenta, uma vez que com o passar dos anos, novas tecnologias fazem com que o preço dos componentes diminua, como mostra a figura 2.3.

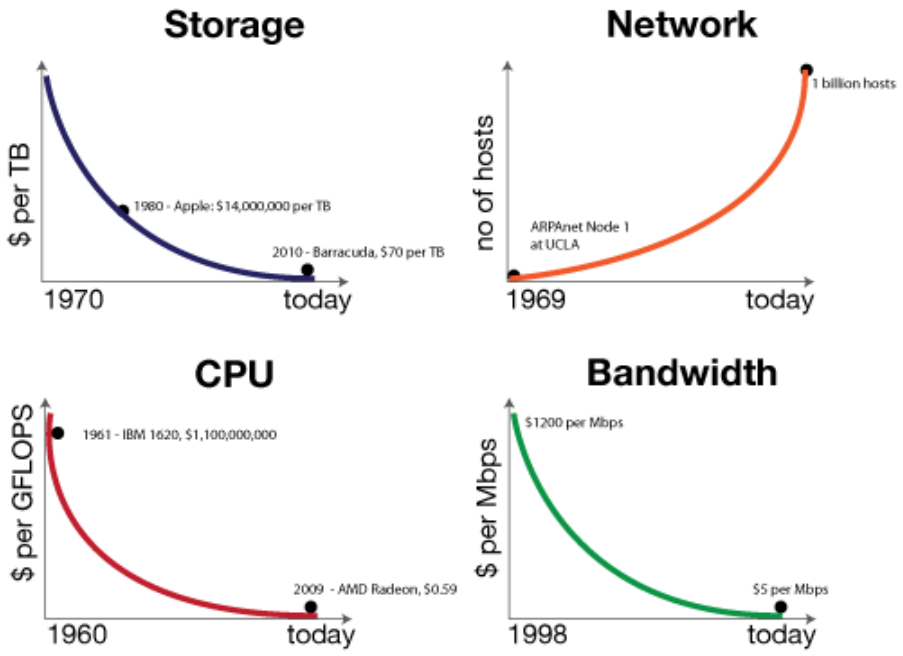

Figura 2.3: Queda do custo dos componentes

Com toda a facilidade de expansão vem um custo. Este custo é relacionado com a perda de desempenho quando há subutilização da rede, ou seja, quanto maior e mais complexa a rede,

mais difícil de gerenciar e com isso, há o aumento da latência. A figura 2.4 mostra um pouco este desequilíbrio. A linha azul clara mosta a latência da rede e a linha azul escura mostra o número de servidores necessários para que a latência se mantenha constante com o aumento da quantidade de tráfego de dados.

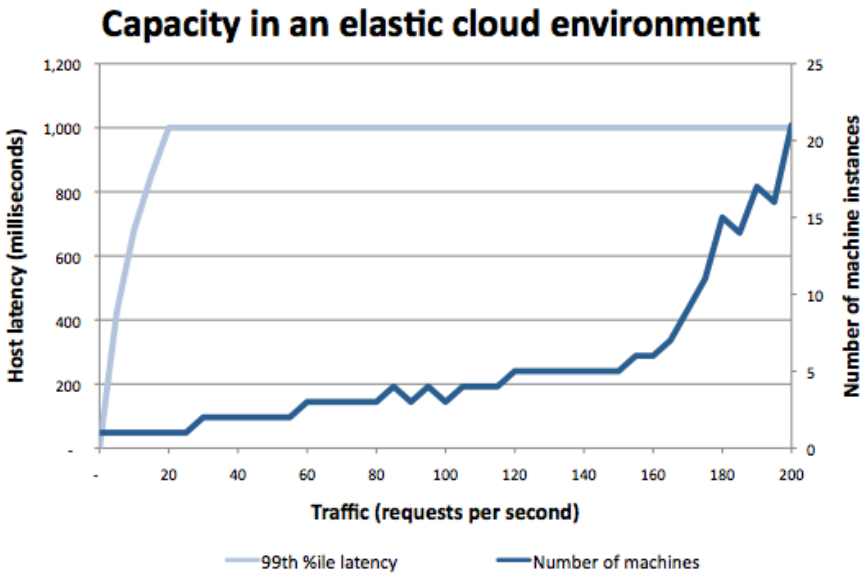

Figura 2.4: Latência da rede x Aumento do tráfego

Com isso, as topologias e os protocolos são muito importantes para equilibrar todos os requisitos sem que haja perda de qualidade de serviço para o usuário. A seguir, vamos apresentar as topologias das redes de datacenters mais utilizadas.

3

Topologias

A seguir, mostraremos algumas topologias, algumas são mais escaláveis, outras mais redundantes, porém todas possuem vantagens e desvantagens.

3.1 Tradicionais

As topologias tradicionais são aquelas que são fisicamente montadas e necessitam manutenção de hardware.

3.1.1 Baseadas em Árvores

As topologias baseadas em árvores são as seguintes:

- Basic Tree
- Fat-Tree
- VL2

Basic Tree A primeira estrutura hierárquica que foi pensada foi uma árvore básica, geralmente com 3 níveis, onde a raiz possui a tarefa de controlar os fluxos que entram e saem do datacenter. Os switches do segundo nível fazem a atribuição dos domínios, roteamento e balanceamento de carga. O terceiro nível é também responsável por um balanceamento de carga entre os servidores, mas em um nível menos. Além disso, este nível consegue controlar um pouco do congestionamento da rede.

A figura 3.1 mostra o esquema básico para esta árvore de 3 níveis.

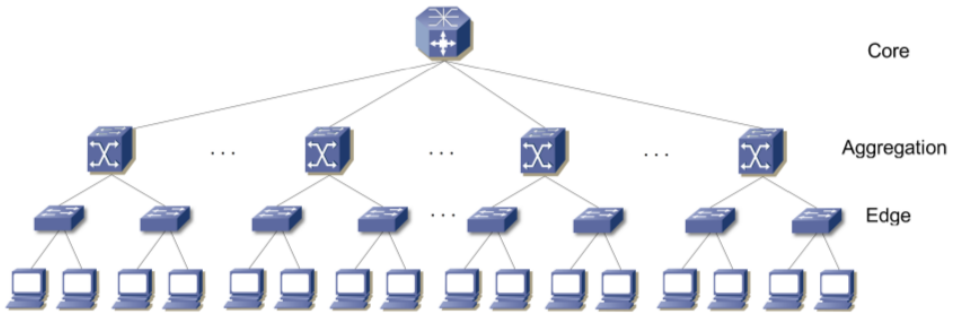

FIGURE 1. A traditional 3-level tree-based data center network topology.

Figura 3.1: Latência da rede x Aumento do tráfego

O crescimento do Oversubscription na direção do Core da rede pode causar alguns problemas, como por exemplo o incast, onde os servidores tentam transmitir ao mesmo tempo na direção do Core, isto causa um congestionamento na rede.

Fat-Tree Para tentar resolver os problemas da topologia anterior, foi desenvolvida a Fat-Tree, que possui mais do que um switch Core e possui conexões cruzadas na camada de agregação. Isso faz com que seja possível os servidores utilizarem vários caminhos diferentes até um switch Core. Esta ideia da redundância de caminhos vem para evitar o congestionamento da rede e tentar resolver o problema do incast que ocorria no caso anterior.

Além disso, é uma estrutura rearranjável não bloqueante e fornece uma relação de Oversubscription de 1:1 a todos os servidores. No entanto, a complexidade da fiação é O(n3), que é um desafio sério.

A figura 3.2 mostra a topologia básica para uma fat-tree de 3 níveis.

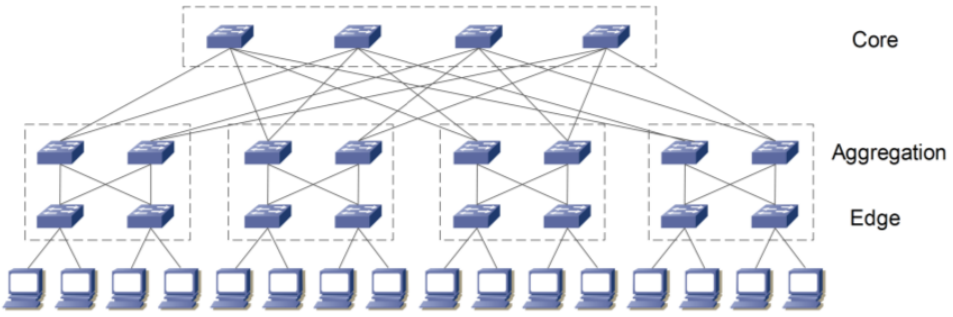

FIGURE 2. A simple 3-level Fat-Tree topology.

Figura 3.2: Latência da rede x Aumento do tráfego

VL2 Esta topologia possui switches em uma topologia de rede CLOS que Usa o VLB (Valiant Load Balancing) para distribuir tráfego entre os caminhos da rede. Além disso, usa uma modificação do protocolo ARP (Address Resolution Protocol) para que seja escalável para um número grande de servidores, onde o broadcast é feito somente na região local de cada servidor.

3.1.2 Recursivas

As topologias recursivas são as seguintes:

- Dcell
- Bcube
- FiConn
- FlatNet
- SprintNet

Dcell A topologia Dcell utiliza a ideia de células compostas por 1 switch e servidores conectados neste switch. As células são interligadas então através dos servidores com outras células. Assim, o número de células é limitado pelo número de portas do servidor. Esta topologia é meio limitada quanto à expansão, uma vez que para que haja a redundância de conexões, cada servidor está conectado com outro servidor em outra célula. Para expandir o número de células é necessário que haja uma troca do switch por outro com um número maior de portas.

Neste caso, o roteamento dentro da própria célula é feito pelo switch e o roteamento com outras células é feito pelos próprios servidores, o que causa um overhead e consequentemente um pequeno atraso do redirecionamento dos fluxos.

A figura 3.3 mostra um exemplo desta topologia para switches com 4 portas.

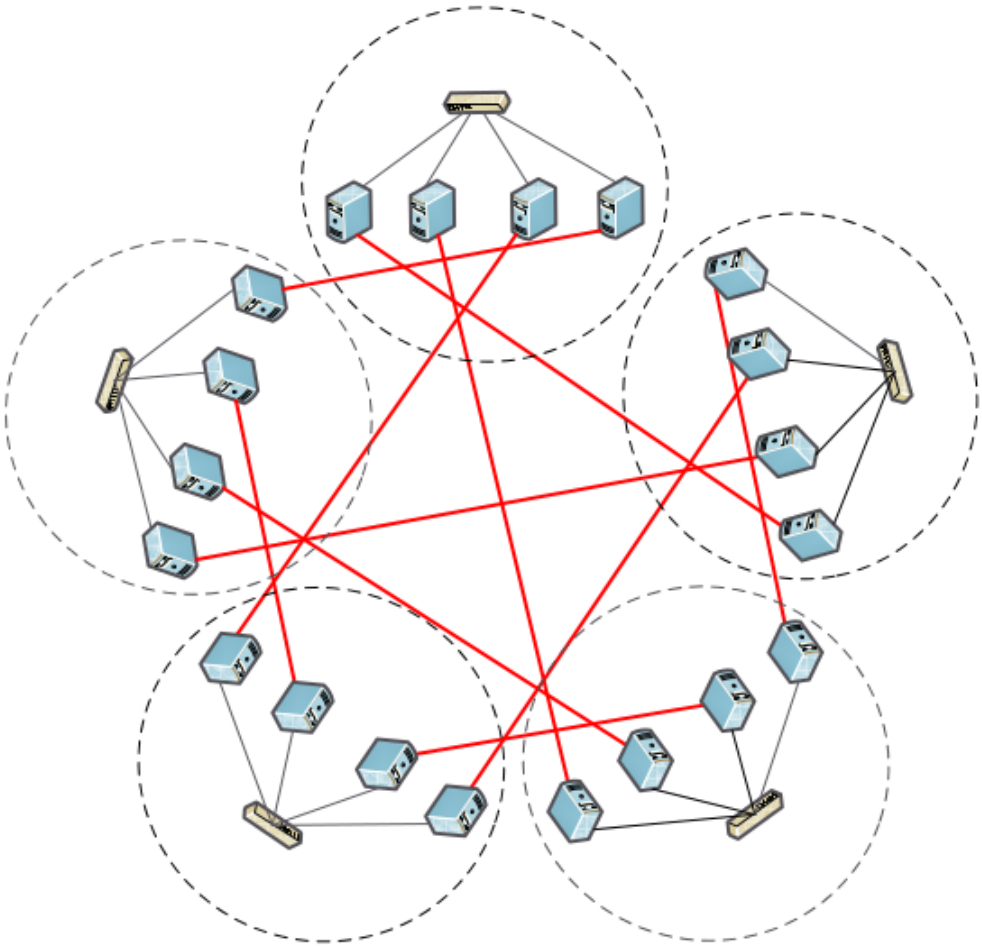

Figura 3.3: Topologia Dcell para switches de 4 portas

Bcube Assim como no caso anterior, esta topologia se baseia na ideia de células interconectadas mas deixa a complexidade do redirecionamento dos fluxos para os switches, uma vez que neste caso, as células são conectadas por switches ao invés dos próprios servidores. Além disso, esta topologia também sofre limitações por conta do número de portas dos switches.

A figura 3.4 mostra esta topologia com 3 níveis e switches de 4 portas.

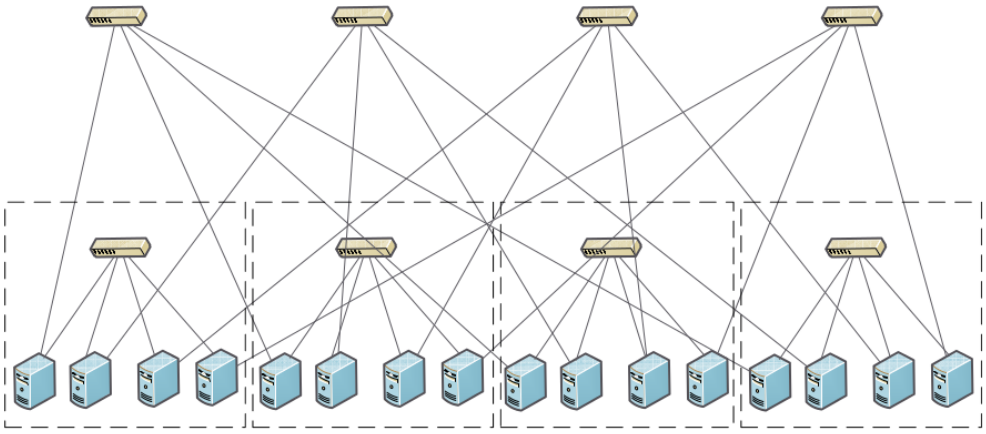

Figura 3.4: Topologia Bcube para switches de 4 portas

FiConn Esta topologia é muito semelhante à Dcell, porém possui a limitação de ter somente 2 links conectados em cada célula, ou seja, diminui a complexidade de fios que esta topologia necessita. Porém, a capacidade da rede é diminuida, uma vez que menos links significam menos largura de banda e pode ser que ocorram congestionamentos mais facilmente.

A figura 3.5 mostra a topologia da rede.

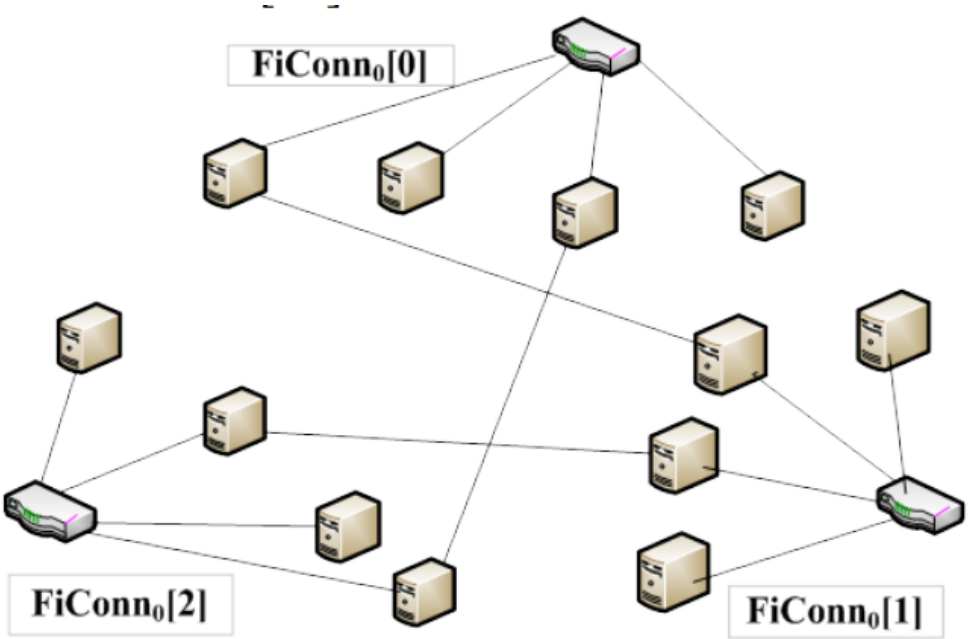

Figura 3.5: Topologia Bcube para switches de 4 portas

FlatNet Esta topologia é semelhante ao Bcube, porém é mais interconectada, porque cada célula está conectada com n switches, onde n é o número de portas do switch interno da célula. A segunda camada de switches por sua vez conecta n células. A diferença é que esta topologia divide a tarefa de roteamento com os switches e com os servidores, assim, os servidores podem

rotear fluxos para outras células.

A figura 3.6 mostra esta topologia para $n = 4$.

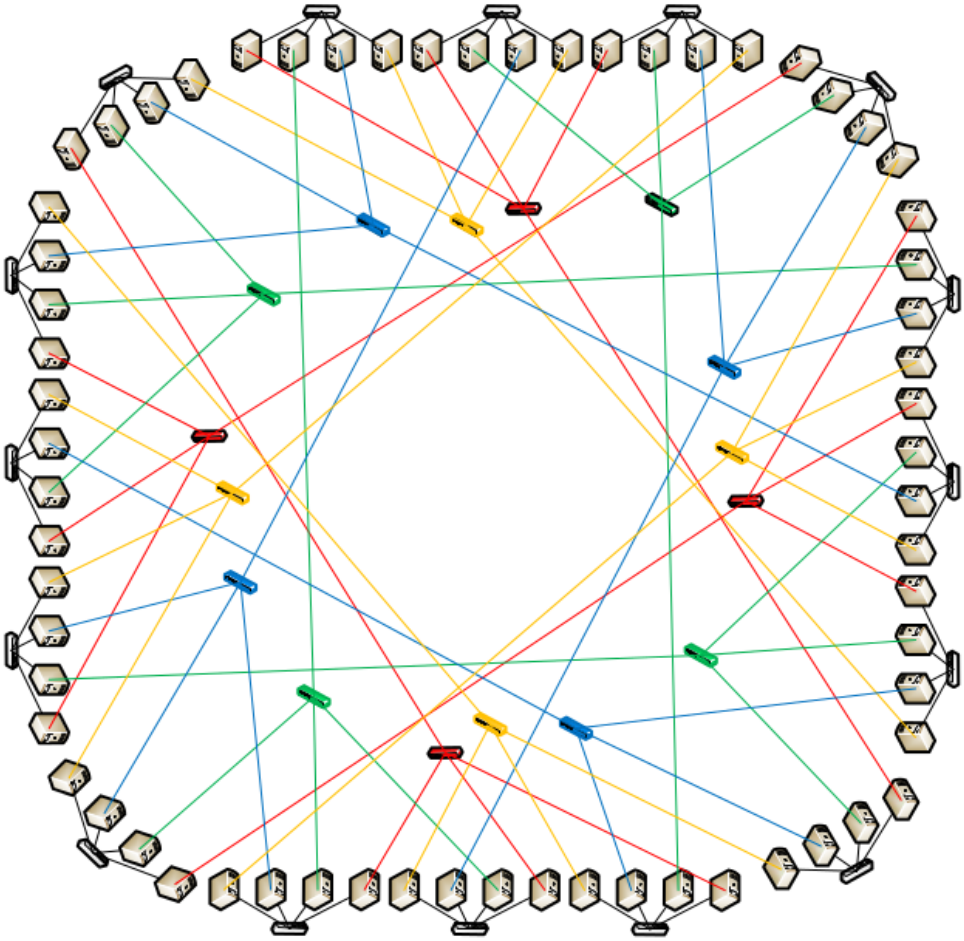

Figura 3.6: Topologia Bcube para switches de 4 portas

SprintNet Esta topologia utiliza células com 2 switches e 4 servidores e existe uma redundância nas conexões internas da célula. Neste caso, todas as conexões entre células são feitas pelos servidores conectados com switches. Isso faz com que todo fluxo interno seja roteado pelo switch e o fluxo externo seja roteado tanto pelos switches como pelos servidores. Assim, existem diversos caminhos que podem ser utilizados para a comunicação inter-células.

A figure 3.7 mostra esta topologia.

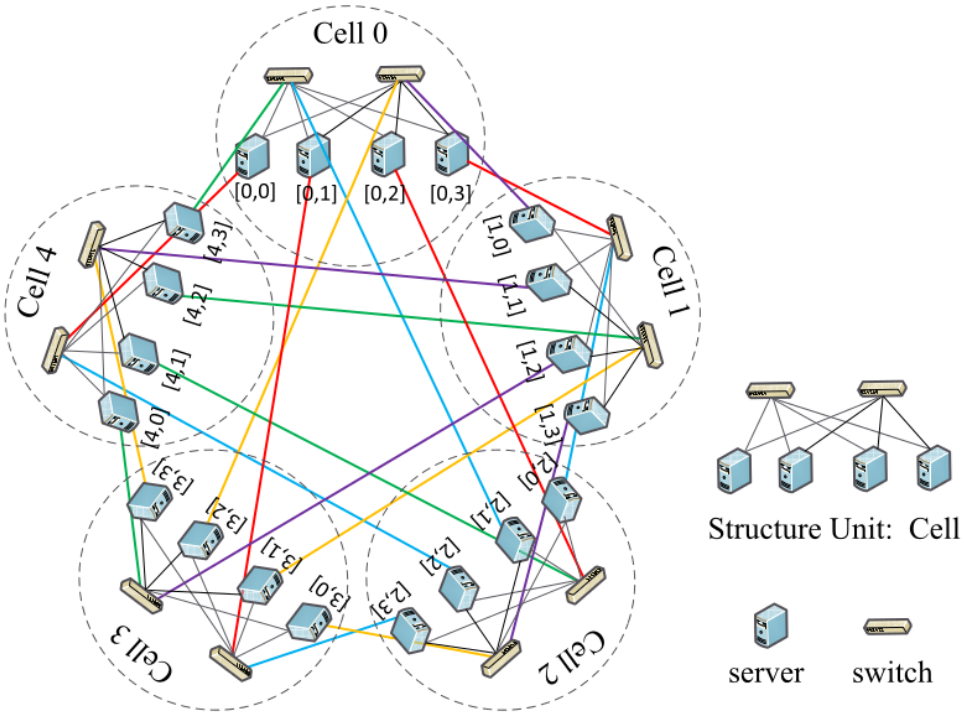

Figura 3.7: Topologia Bcube para switches de 4 portas

A tabela da figura 3.8 mostra a comparação da complexidade de algumas topologias mostradas. Como pode ser visto, as redes que suportam o maior número de servidores, também possuem uma complexidade muito grande de conexões, o que aumenta a complexidade de roteamento mas também aumenta o número de caminhos que podem ser escolhidos para um fluxo.

	Fat Tree (3 layers)	VL2 (3 layers)	DCell (2 layers)	BCube (2 layers)	FlatNet (2 layers)	SprintNet (2 layers)
Servers Number	$\frac{n^3}{4}$	$\frac{(n-2)n^2}{4}$	$n(n+1)$	n^2	n^3	$(\frac{c}{c+1})^2 n^2 + \frac{c}{c+1} n$
Links Number	$\frac{3n^3}{4}$	$\frac{(n+2)n^2}{4}$	$\frac{3n(n+1)}{2}$	$2n^2$	$2n^3$	$\frac{c^2 n^2}{c+1} + cn$
per Server	3	$\frac{n+2}{n-2}$	$\frac{3}{2}$	2	2	≥ 2
Switches Number	$\frac{5n^2}{4}$	$\frac{n^2}{4} + \frac{3n}{2}$	$n+1$	$2n$	$2n^2$	$\frac{c^2}{c+1} n + c$
per Server	$\frac{5}{n}$	$\frac{n+6}{n^2-2n}$	$\frac{1}{n}$	$\frac{2}{n}$	$\frac{2}{n}$	$\frac{c+1}{n}$
Bisection Bandwidth	$\frac{n^3}{8}$	$\frac{n^2}{4}$	$\frac{n^2}{4} + \frac{n}{2}$	$\frac{n^2}{2}$	$\frac{n^3}{4}$	$\frac{c^2 n^2}{2(c+1)^2} + cn$
per Server	$\frac{1}{2}$	$\frac{1}{n-2}$	$\approx \frac{1}{4}$	$\frac{1}{2}$	$\frac{1}{4}$	$\frac{1}{2} + \frac{(2c+1)(c+1)}{2(cn+c+1)}$
Network Diameter	6	6	5	4	8	4

Figura 3.8: Topologia Bcube para switches de 4 portas

3.2 SDN

Com o avanço do SDN, a ideia mais básica é definir servidores virtualizados e criar uma rede virtualizada. Este componente é a peça que faltava para criar datacenters totalmente definidos por software, uma vez que armazenamento e processamento virtualizados já existiam há um certo tempo. Com o SDN, é possível criar uma rede virtual com qualquer topologia e conectar

máquinas virtuais para o processamento e discos virtuais para o armazenamento. A figura 3.9 mostra a arquitetura de um datacenter virtualizado.

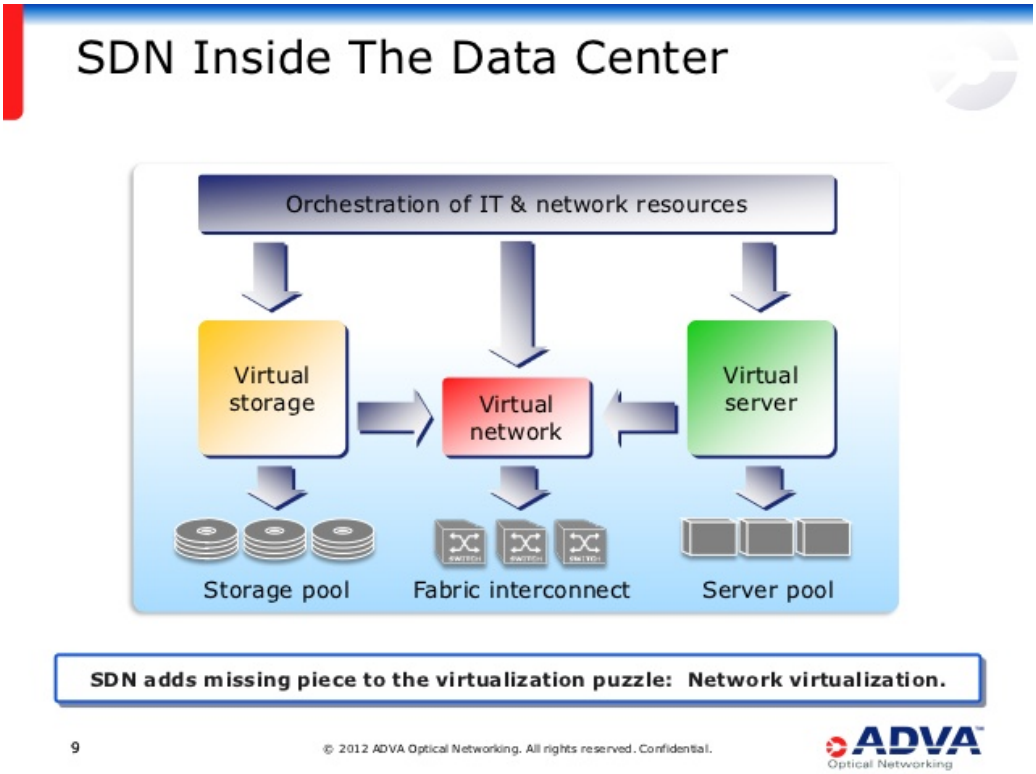

Figura 3.9: Topologia Bcube para switches de 4 portas

Para que isso seja possível, é necessário que haja uma modificação na camada dos servidores, onde os racks são substituídos por switches OpenFlow que escalonam as tarefas para máquinas virtuais na camada de borda. Esta técnica já é utilizada pelo Paypal e pode ser vista na figura 3.10.

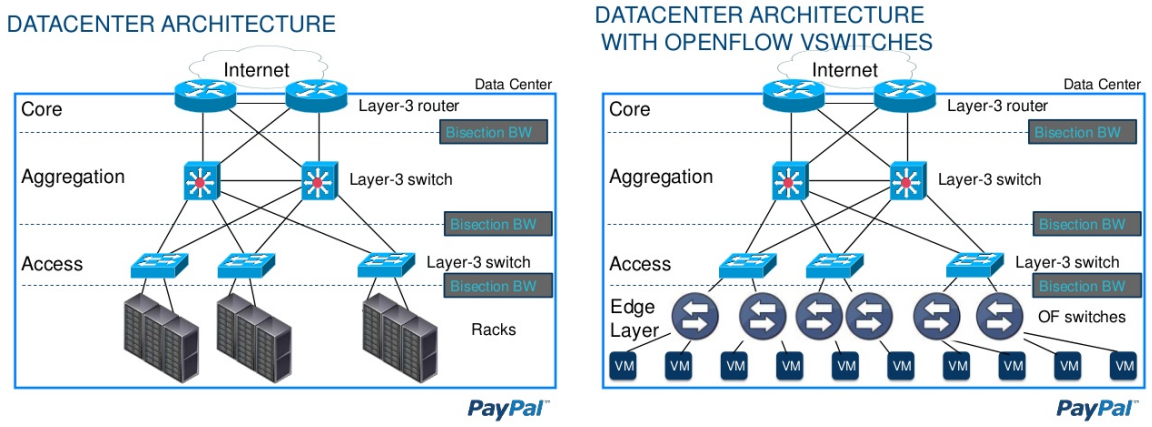

Figura 3.10: Topologia Bcube para switches de 4 portas

O sistema fica então com duas camadas, uma camada física e uma camada virtualizada,

como pode ser visto na figura 3.11. A camada superior contém todos os recursos físicos do datacenter enquanto a camada inferior contém os recursos virtualizados. A interface entre as camadas é feita por um controlador SDN que é responsável por atribuir recursos físicos para recursos virtualizados.

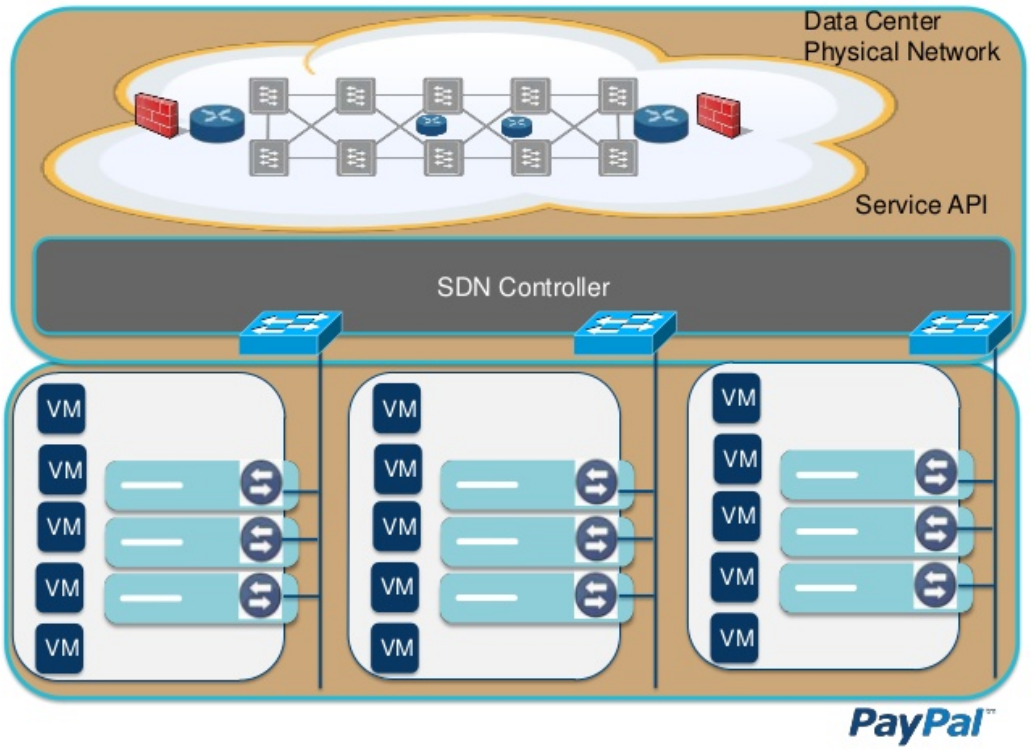

Figura 3.11: Topologia Bcube para switches de 4 portas

4

Protocolos

5

Tendências

6

Conclusão

7

Referências

- Manual de referência do NS-3

  `https://www.nsnam.org/docs/release/3.8/manual.pdf`

- Documentação

  `https://www.nsnam.org/doxygen/index.html`