

Sistemas Distribuídos

Introdução

Prof. Tales Viegas

<https://fb.com/ProfessorTalesViegas>

O que é um Sistema Distribuído?

- ▶ Conjunto de computadores ligados em rede, com software que permita o compartilhamento de recursos e a coordenação de atividades, oferecendo idealmente um sistema integrado.

Características

- ▶ Comunicação através de mensagens
- ▶ Concorrência
- ▶ Compartilhamento de recursos
- ▶ Sistema Assíncrono
- ▶ Falhas Independentes
- ▶ Heterogeneidade

Características

- ▶ Comunicação através de mensagens
 - Não existem variáveis globais compartilhadas
 - Modelos de Programação:
 - Cliente/Servidor
 - Modelo baseado em objetos
- ▶ Concorrência
 - Vários usuários utilizando o sistema ao mesmo tempo
 - Necessário coordenar o acesso aos recursos compartilhados (HW/SW/Dados...)

Características

- ▶ Compartilhamento de Recursos
 - Impressoras
 - Ferramentas para trabalho cooperativo
 - Bases de Dados
 - Gerência de recursos para controlar o acesso aos recursos compartilhados
- ▶ Sistema Assíncrono
 - Não existe um “relógio global”
 - Diferentes velocidades de processamento
 - Não existe um limite de tempo para comunicação

Características

- ▶ Falhas Independentes
 - Falhas na rede
 - Perdas de mensagens
 - Duplicação
 - Reordenação
 - Falhas em unidades de processamento
 - A falha em um componente não impede o outro de funcionar

Características

▶ Heterogeneidade

- Um sistema distribuído pode possuir:
 - Diferentes tipos de rede
 - Diferentes tipos de hardware
 - Diferentes representações de dados
 - Diferentes códigos de máquina
 - Diferentes sistemas operacionais
 - Diferentes interfaces para os protocolos de comunicação
 - Diferentes linguagens de programação
 - Diferentes representações de estruturas de dados, como arrays ou registros
- Para resolver o problema, define-se uma camada de Software intermediária: middleware

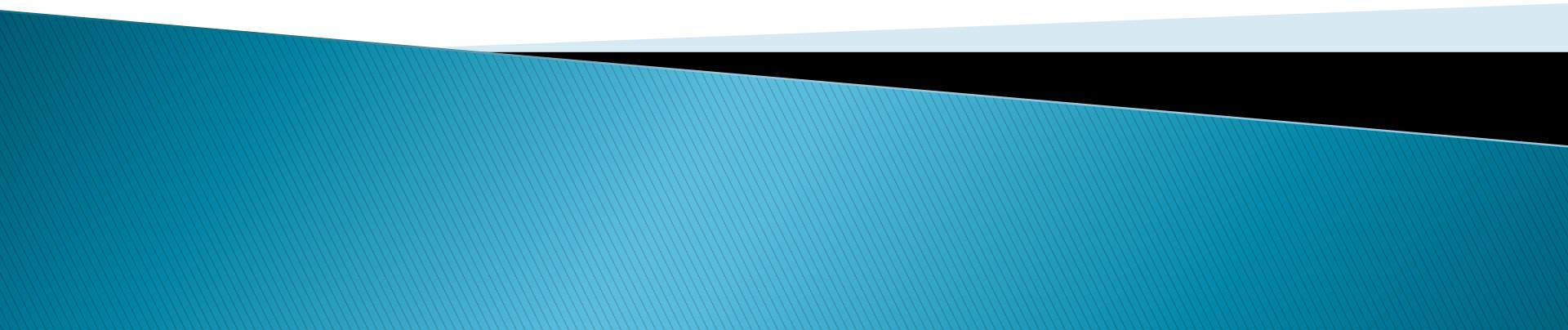
Eight Fallacies of Distributed Computing

- ▶ “Essentially everyone, when they first build a distributed application, makes the following eight assumptions.
- ▶ All prove to be false in the long run and all cause big trouble and painful learning experiences” (Peter Deutsch, 1992)

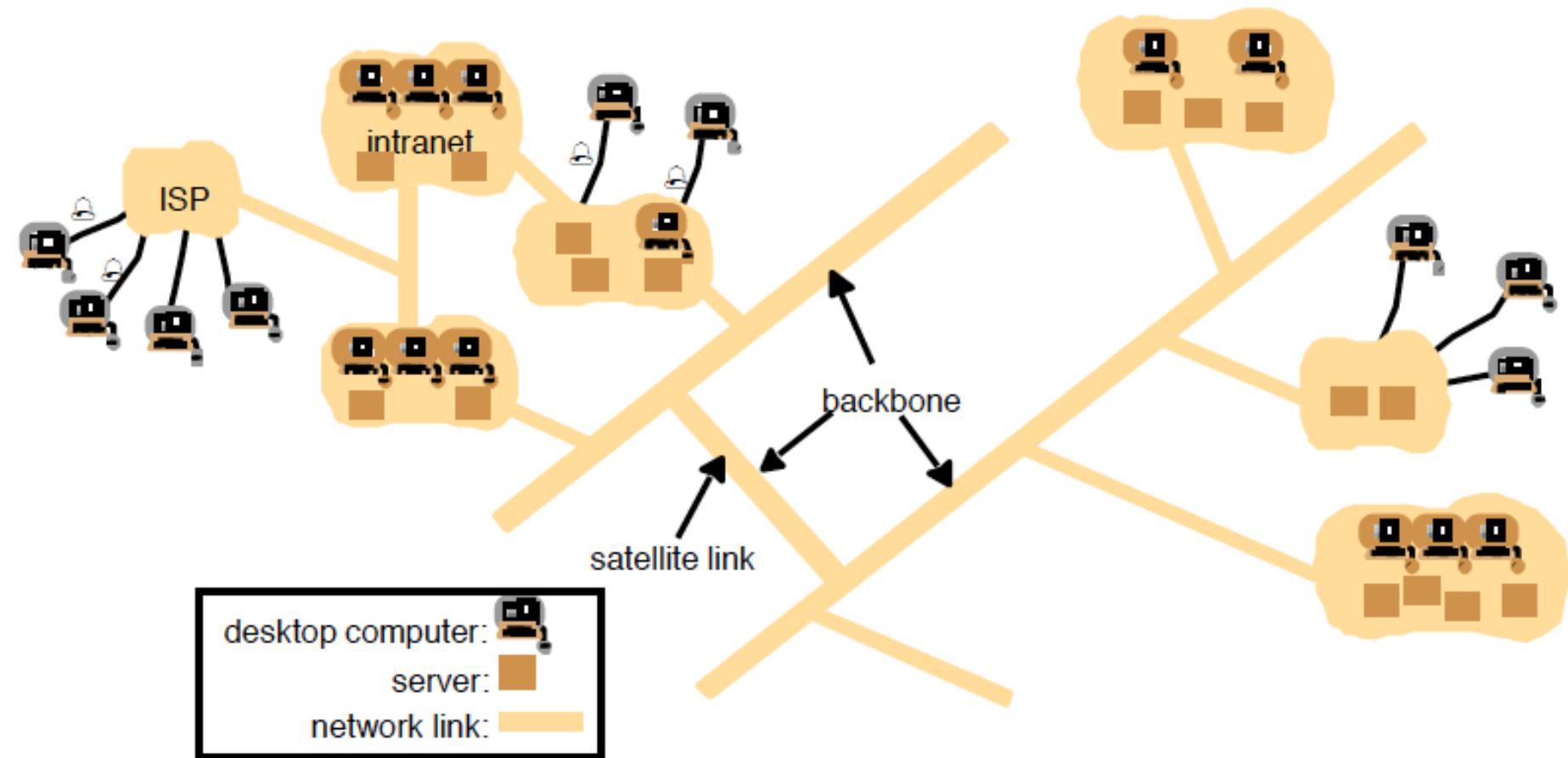
Eight Fallacies of Distributed Computing

1. A rede é confiável
2. A latência é zero
3. A largura de banda é infinita
4. A rede é segura
5. A topologia de rede não vai mudar
6. Há apenas 1 administrador
7. O custo de transporte é zero
8. A rede é homogênea

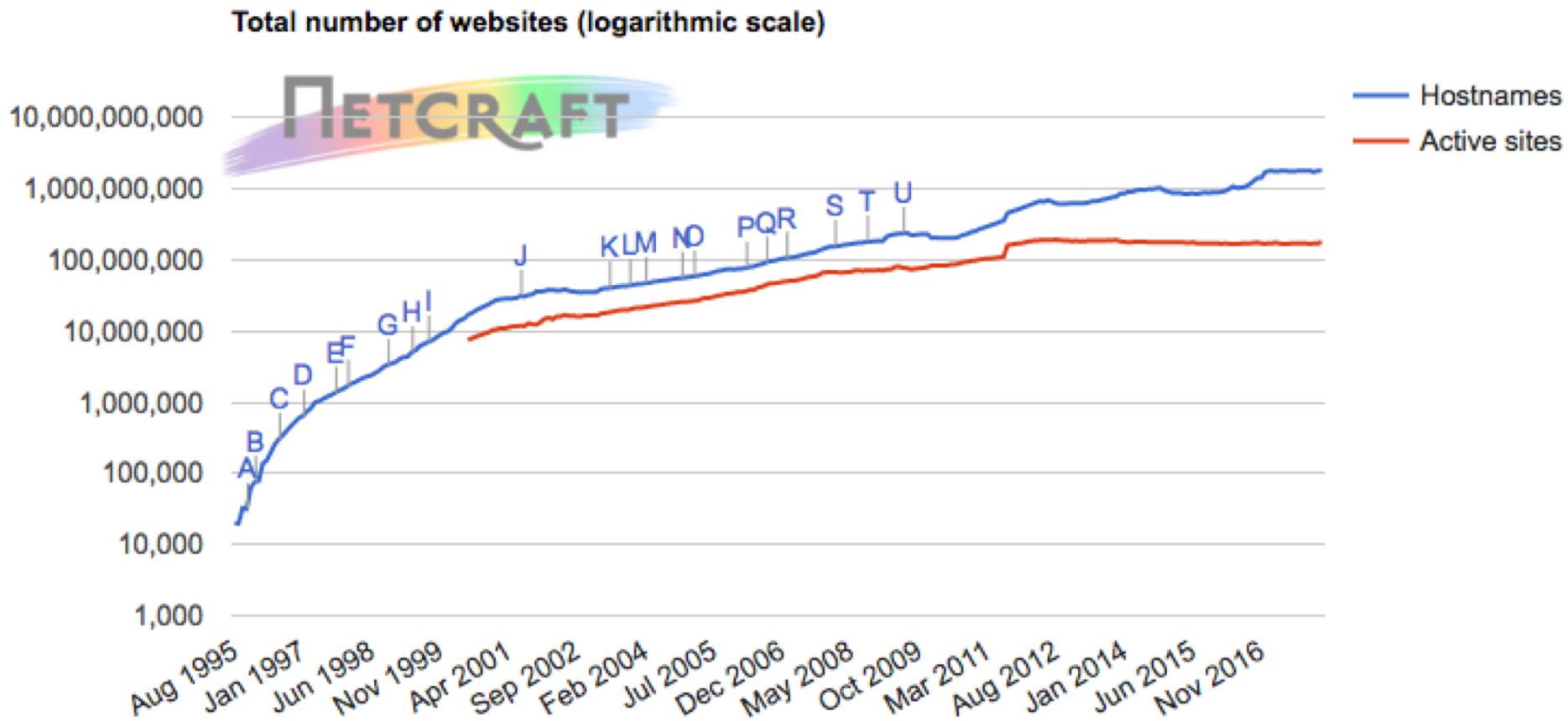
Exemplos de Sistemas Distribuídos



Internet

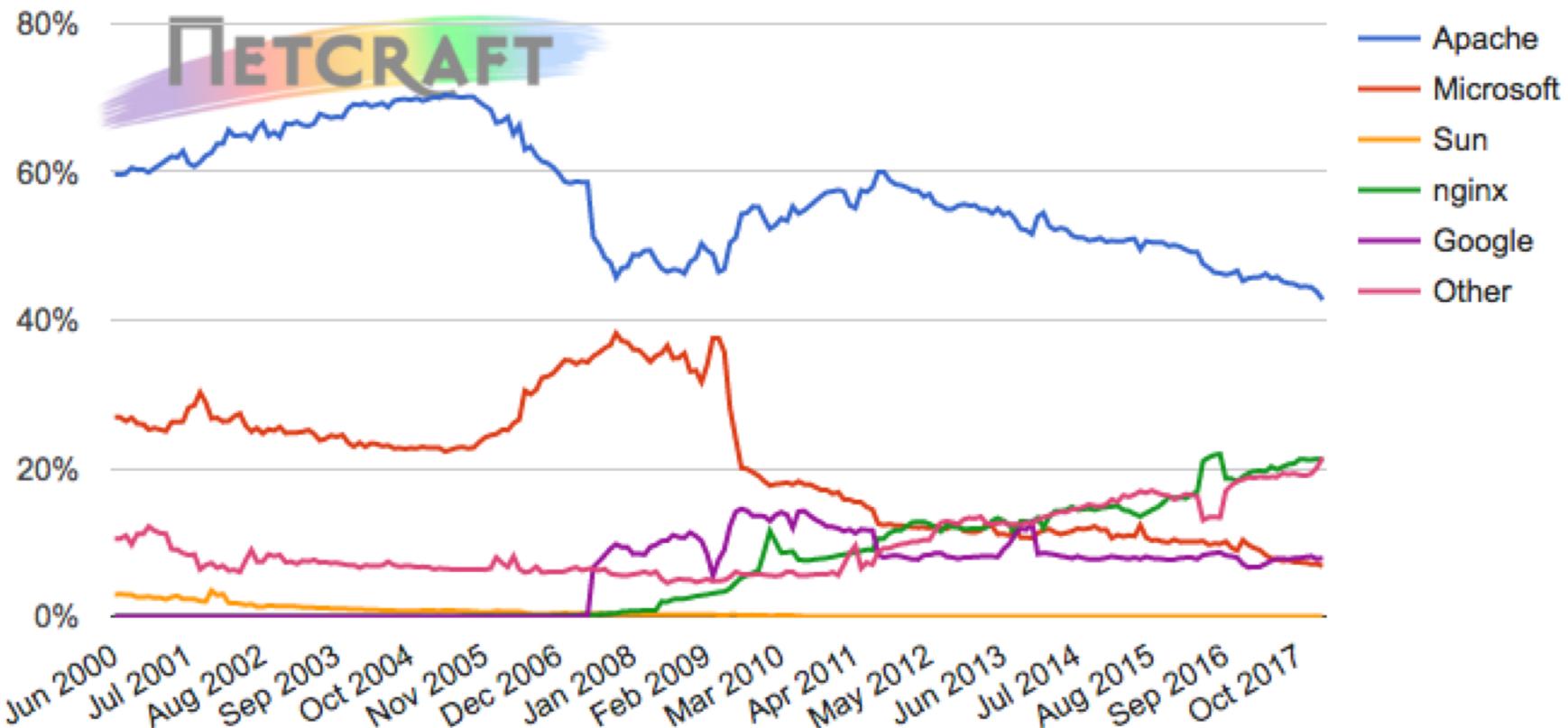


Total Sites Across All Domains

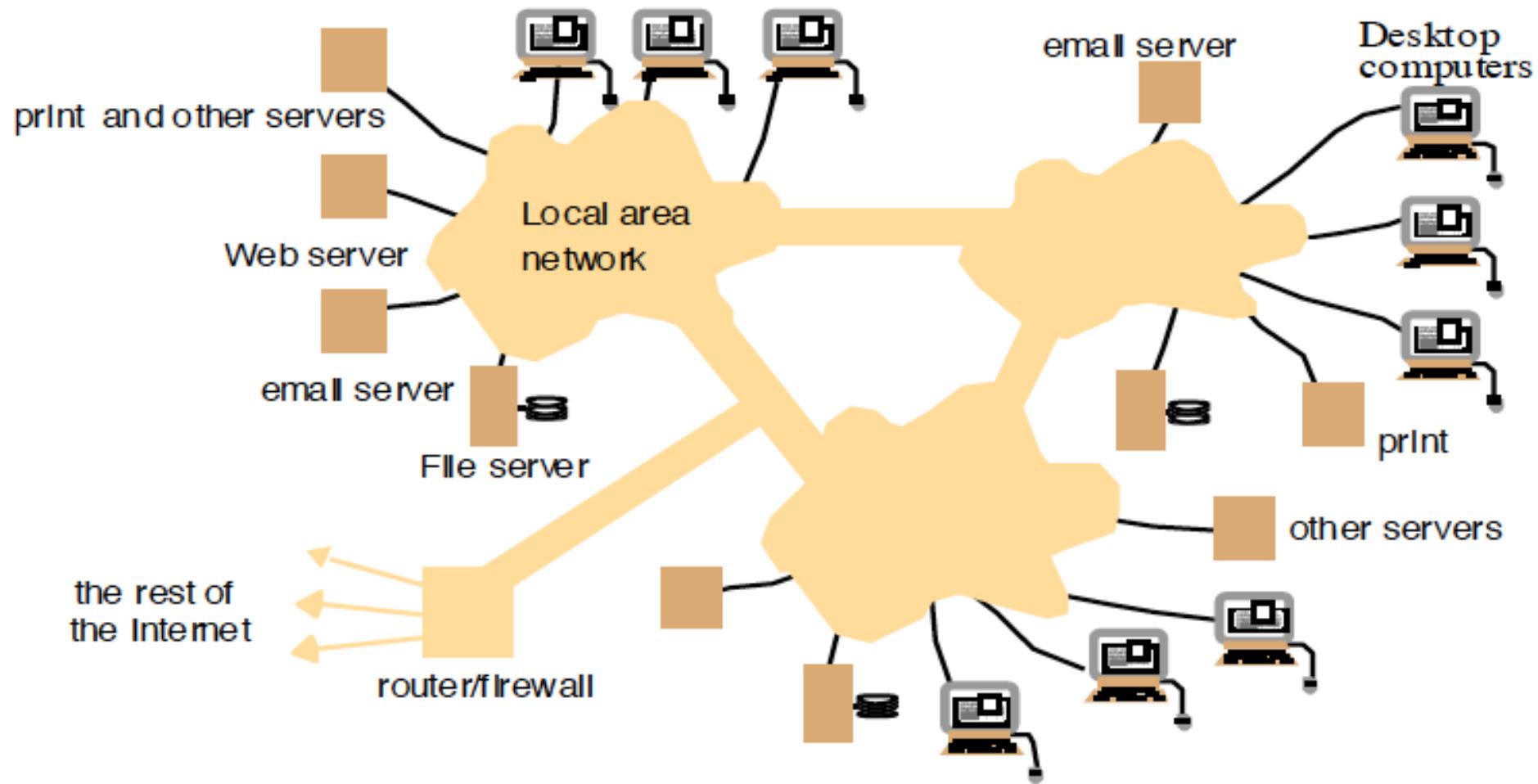


<http://news.netcraft.com/archives/category/web-server-survey/>
Fevereiro/2018

Market Share for Top Servers

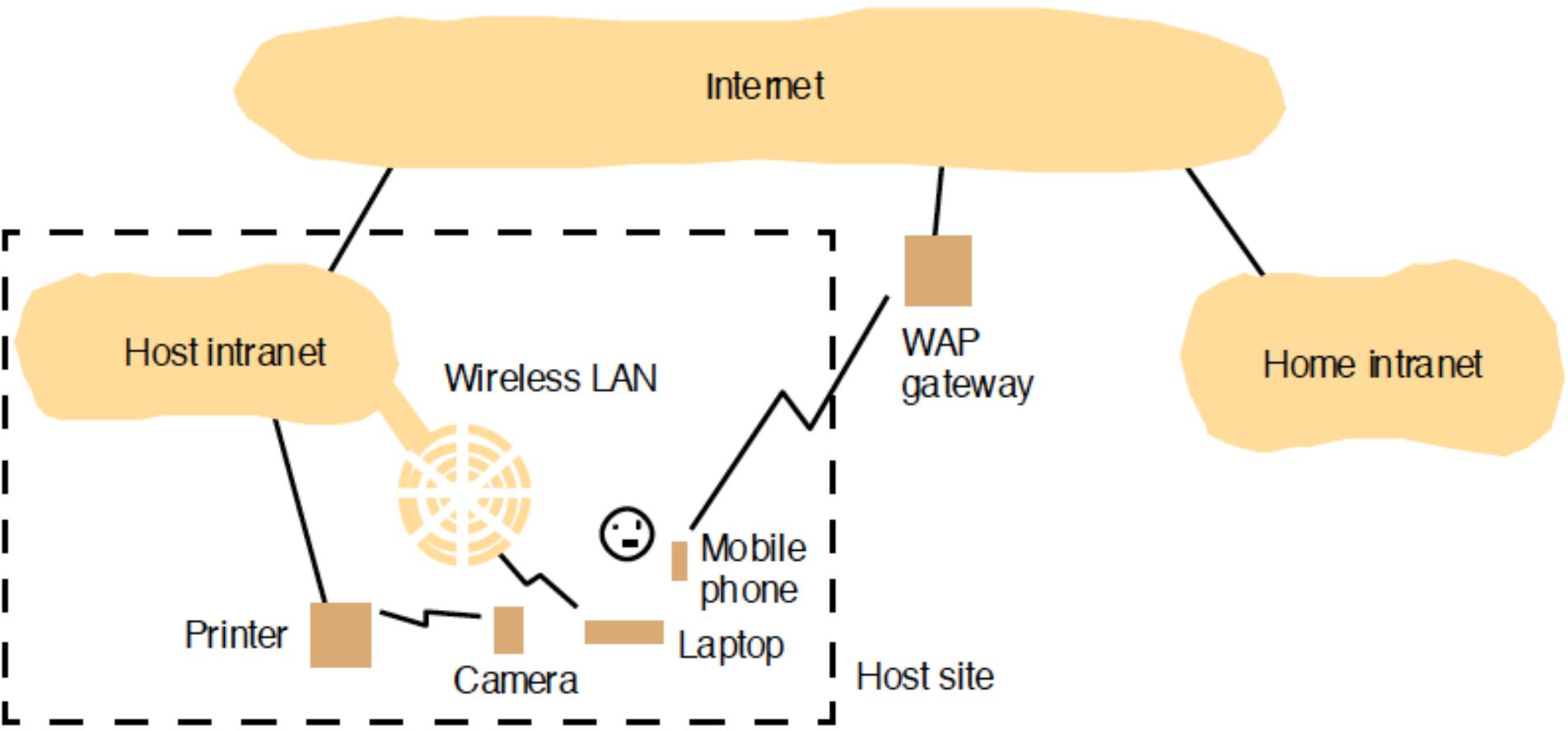


Intranets



Computação Móvel e Ubíqua

- ▶ Integração de equipamento portátil em sistemas distribuídos



Computação Móvel e Ubíqua

- ▶ Computadores portáteis
- ▶ Telefones móveis
- ▶ Máquinas fotográficas digitais
- ▶ Impressoras
- ▶ Wearable devices
- ▶ Sistemas embarcados

Outros Sistemas Distribuídos

- ▶ Correio Eletrônico
- ▶ Sistemas de Arquivos Distribuídos
- ▶ FTP/Telnet/SSH
- ▶ Chat
- ▶ Mensagens Instantâneas

- ▶ Sistemas de Comércio Eletrônico
- ▶ Caixas Eletrônicos
- ▶ ...

Desafios em Sistemas Distribuídos

- ▶ Escalabilidade
- ▶ Extensibilidade
- ▶ Tolerância a Falhas
- ▶ Segurança
- ▶ Transparência

Escalabilidade

- ▶ Capacidade do sistema funcionar de forma correta e com o desempenho desejado independente do número de usuários
- ▶ É necessário
 - Especificar o software de modo que o aumento de usuários não exija grandes alterações
 - Evitar algoritmos e estruturas de dados centralizadas (replicação de dados, se necessário)
 - ...

Escalabilidade

- ▶ É necessário:
 - Controlar o aumento de custos devido à disponibilização de mais recursos
 - Controlar a perda de performance
 - Evitar ultrapassar a quantidade de recursos disponíveis (ex: endereço IP de 32 bits)

Extensibilidade

- ▶ Capacidade do sistema ser extensível, tanto em software, como em hardware
- ▶ Novos componentes podem ser adicionados no sistema sem afetar os componentes já existentes, podendo se comunicar com eles

Extensibilidade

- ▶ É importante que:
 - Sejam conhecidas as interfaces dos novos componentes através da publicação da sua documentação
 - Utilizar protocolos e formatos de comunicação padrão
- ▶ RFC (Request for Comments)
 - <http://www.ietf.org>
 - Contém as especificações de protocolos desde o início dos anos 80

Tolerância a Falhas

- ▶ Tolerar uma falha significa conter os efeitos de forma que o sistema continue a funcionar
- ▶ Detecção de falhas
 - Dados corrompidos (mensagens ou arquivos) podem ser detectados através de checksums
- ▶ Localização de falhas
 - Se não houve resposta a um pedido o que significa?
 - Falha na rede?
 - Falha no nó destino?
 - Como distinguir?

Tolerância a Falhas

- ▶ Mascarar/Tolerar a falha
 - Algumas falhas podem ser ocultadas do usuário se for utilizada redundância suficiente
 - Quando uma mensagem não chega, pode ser retransmitida
 - Um arquivo pode ser escrito duplicado (um em cada disco)
 - Entre dois “routers” da internet, sempre devem existir duas rotas disponíveis
 - Uma base de dados pode ser replicada em vários servidores

Segurança

- ▶ Manter recursos computacionais seguros significa:
 - Manter o nível de confidencialidade exigido pelos usuários
 - proteção contra acessos não autorizados
 - Garantir a integridade dos dados
 - proteção contra alteração ou corrupção de dados ou programas
 - Manter a disponibilidade do sistema
 - proteção contra interferências com o meio de acesso aos recursos

Segurança

- ▶ Alguns problemas a resolver:
 - Ataques do tipo DOS (Denial of Service)
 - Segurança do código móvel

Transparência

- ▶ O sistema deve ser visto como um todo e não como uma coleção de componentes distribuídos
- ▶ No padrão ODP (Open Distributed Processing) foram definidos os seguintes tipos de transparência:
 - Acesso
 - Localização
 - Concorrência
 - Replicação
 - Falhas
 - Migração
 - Desempenho
 - Escalabilidade
- ▶ Acesso e localização são conhecidos como “Transparência de Rede”

Transparência

- ▶ Transparência de Acesso
 - Permite que o acesso a recursos locais e a recursos remotos seja feito através das mesmas operações
 - Ex: sistemas de arquivos distribuídos
- ▶ Transparência de Localização
 - Permite que os recursos possam ser acessados sem o conhecimento da sua localização
 - Ex: correio eletrônico

Transparência

- ▶ Transparência de Concorrência
 - Permite que vários clientes de um componente não necessitem ter em conta o acesso concorrente ao mesmo
- ▶ Transparência de Replicação
 - Permite que os clientes de um componente não percebam se existe replicação ou se estão usando uma réplica ou o original
 - A utilização de várias instâncias de um componente pode ocorrer por razões de desempenho ou confiabilidade
 - Os usuários do componente não precisam saber que o componente precisa ser replicado

Transparência

▶ Transparência de Falhas

- Permite que o sistema funcione na presença de falhas de software ou hardware sem que os usuários e programadores saibam como as falhas foram sanadas
- Ex: Um sistema de email pode retransmitir uma mensagem até que ela seja entregue com sucesso

▶ Transparência de Migração

- Permite que um recurso possa trocar sua localização sem que isto afete seu uso.
- Ex: celulares em movimento

Transparência

- ▶ Transparência de Desempenho
 - Permite que o sistema seja reconfigurado para melhorar o seu desempenho sem que os usuários percebam
- ▶ Transparência de Escalabilidade
 - Permite que o sistema seja expandido sem que os usuários percebam como isto foi realizado.