

Replicação em sistemas web

Sistemas Distribuídos

abril de 2018



Servidores Web

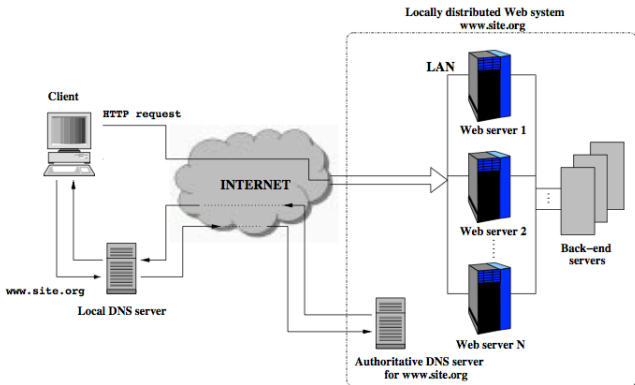
- tolerância a falhas
- desempenho/escalabilidade

desempenho:

- uso de servidores mais potentes (*scale-up x scale-out*)
- caching
- uso de servidores replicados

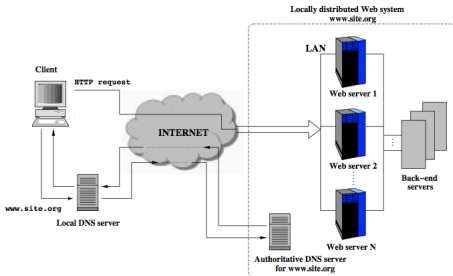


replicação local



- V. Cardellini, E. Casalicchio, M. Colajanni, P. Yu. The state of the art in locally distributed Web-server systems. *ACM Computing Surveys (CSUR)*, 34 (2), 263-311, 2002.

replicação local



- servidores web tipicamente acessam serviço de dados (outra *camada*)
- busca de uma url pelo cliente pode se traduzir em diversos acessos
 - podem ocorrer na mesma conexão ou em conexões diferentes (HTTP/1.1)

Escolha de servidor

- 1 cliente
- 2 DNS
- 3 sistema web

componentes

- mecanismo de roteamento de requisições
- algoritmo de seleção do servidor apropriado

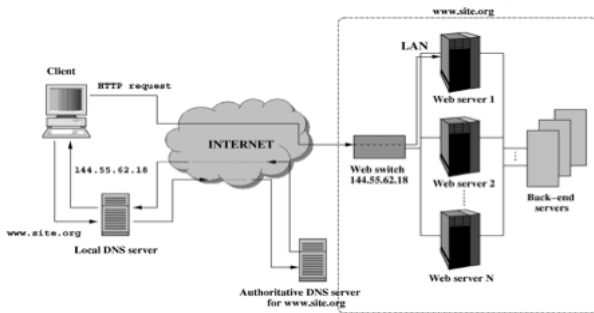


DNS

- tradução url→IP
- serviço distribuído



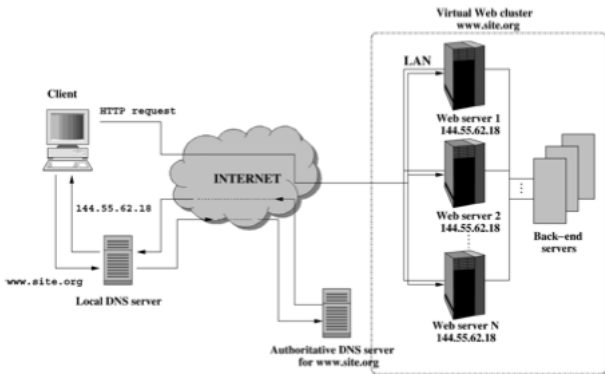
Arquiteturas baseadas em um ponto de entrada único



uso de um *web switch*

- um único nó (e um único endereço IP) faz a interface do cluster com clientes
- ponto central de distribuição

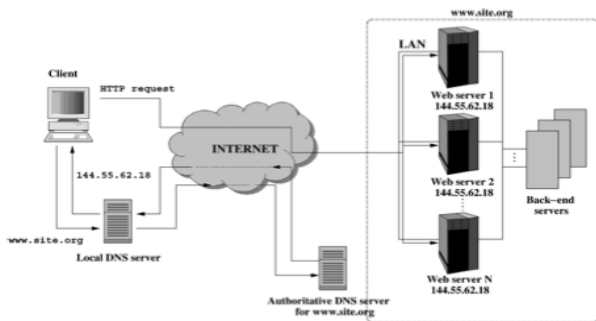
Arquiteturas baseadas em um ponto de entrada único



uso de virtualização

- o mundo externo vê vários nós com o mesmo endereço IP
- cada nó recebe todas as msgs e filtra as que irá tratar

Arquiteturas distribuídas



- múltiplos IPs podem ser vistos por clientes
- tipicamente é o DNS quem roteia a requisição



Nível do roteamento em arqs baseadas em clusters

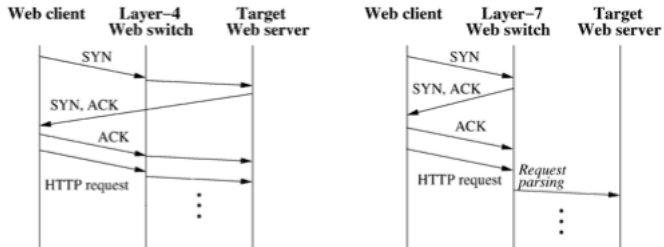
- *web switch* conhece endereço privado de cada nó

roteamento pode ocorrer em diferentes camadas:

- camada de transporte: roteamento ignora conteúdo
- camada de aplicação: roteamento considera conteúdo
 - maior sobrecarga de processamento mas pode implementar políticas mais sofisticadas



Roteamento nas diferentes camadas



- mecanismos mais complexos para roteamento de requisições
- alguns mecanismos proprietários exigem mudanças em sistemas operacionais



Retorno de resposta ao cliente

- two-way** : respostas a clientes também passam por *web switch*
- one-way** : servidor web escolhido envia diretamente resposta ao cliente



Roteamento na camada de transporte

- pacotes nas duas direções são reescritos no nível TCP/IP (NAT)
- *web switch* tem que reescrever cabeçalhos IP para enganar cliente e servidor sobre fonte de mensagens
 - *web switch* mantém tabela (sessão → servidor)



Uso de DNS para distribuição de requisições

- DNS responde fazendo *round-robin* entre servidores
- dificuldades com caches
 - caches ao longo da cadeia de consulta limitam papel do DNS-a
 - *ttl*, aumento de tráfego, caches em clientes



Distribuição de requisições pelo servidor Web

HTTP redirect

- primeiro servidor contactado responde com *redirect*
 - redireção pode levar em consideração o conteúdo da requisição
- ... *round-trip* extra

reescrita de URL

- primeiro servidor contactado altera as urls na página devolvida, indicando outro servidor
- custoso mas bastante flexível
 - pode ser implementado no nível de aplicação



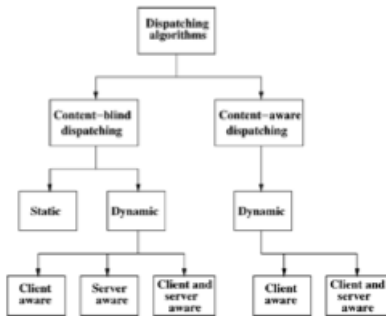
Dispatching e distribuição de carga

classificações tradicionais de algoritmos de distribuição

- centralizados X distribuídos
- distribuição X balanceamento
- estáticos X dinâmicos X adaptativos
 - sofisticação X sobrecarga (coleta de estado, algoritmo de decisão, ...)
- combinação de algoritmo estático no *web switch* com re-roteamento dinâmico pelo servidor?



Taxonomia



Algoritmos estáticos

- *round-robin*
- *random*
 - não há conhecimento prévio da carga a ser executada
 - algoritmos podem ser modificados para levar em conta heterogeneidade



Algoritmos dinâmicos ignorando conteúdo

informação utilizada

- cliente
 - informação sempre passa por *web switch*
- servidor
 - carga
 - *caches* em disco
 - coleta gera sobrecarga



Algoritmos dinâmicos considerando conteúdo

motivação

- otimização do uso de *cache*
- roteamento para servidores específicos (*streaming, ...*)



Algoritmos dinâmicos e informação de estado

índices frequentemente usados:

- utilização de CPU (no período anterior)
- fila de prontos (instantâneo)
- atividade de E/S
- *número de conexões ativas*
- tempo de resposta



aproveitamento de caches

- redirecionamento sempre para servidor que já tem aquele conteúdo
- algoritmos de *hash*



aproveitamento de caches

- redirecionamento sempre para servidor que já tem aquele conteúdo
- algoritmos de *hash*
 - entradas e saídas de servidores exigem rehash
 - balanceamento depende de distribuição de acessos



aproveitamento de caches

- entradas e saídas de servidores exigem rehash
 - uso de algoritmos de hash consistente
- balanceamento depende de distribuição de acessos
 - hash consistente com cargas limitadas

