MT 3 Estudo

20 de junho de 2023 13:46

Protocolo HTTP

- Protocolo de aplicação mais usado na Internet
- Linguagem comum entre clientes e servidores
- Funciona como protocolo de pedido-resposta
  Um browser envia um pedido HTTP para um servidor **->** O servidor retorna uma resposta HTTP para o cliente
  -> A resposta contém dados de informação e o conteúdo requerido pelo cliente
- Protocolo Bastante Simples para trocar mensagens de texto entre duas maquinas
- Foi lançado há 30 anos

HTTP 0.9

- Proposto por Tim Berners-Lee para implementar a World Wide Web
- Tinha como principais objetivos:
  - permitir transferência de ficheiros de texto
  - permitir pesquisas em ficheiros hypertexto
  - permitir negociação de formatos de ficheiros
  - permitir referenciar outros servidores ao cliente.
- O pedido do cliente é um conjunto de caracteres terminados com um CRLF(Carriage Return(\r) Line Feed(\n))
- A resposta é um conjunto de caracteres em HTML

HTTP 1.0

- Surgiu no boom da internet para colmatar limitações anteriores
- Servir mais formatos de ficheiros (não só documentos HTML)
- Providenciar metadatos sobre os pedidos e respostas
- Permite Negociação de conteúdos
- Nunca chegou a ser um standard

HTTP 1.0 - Métodos

- **GET** - Pede um recurso dado por um determinado url
- **HEAD** - Semelhante ao GET, mas sem o conteúdo do recurso. Útil para obter apenas os headers da resposta
- **POST** - Requer ao servidor que aceite o conteúdo que vai em anexo

HTTP 1.0 - Status codes

- **200 Ok** - Resposta bem sucedida
- **301 Moved Permanently** - Pedidos futuros devem ser redirecionados
- **400 Bad Request** - Pedido mal formado
- **404 Not Found** - Recurso pedido não foi encontrado.

HTTP 1.1

- Standard que resolve algumas ambiguidades das versões anteriores e otimizações de performance
- Reutiliza conexões TCP(evita TCP 3-way handshake)
- Pedidos de bytes específicos (útil para streaming)
- Limitações:
  - Pedidos e repostas são sequencias
  - Para haver paralelismo tem de haver múltiplas conexões ao mesmo servidor

HTTP 1.1 - Novos métodos

- **PUT** - Requer ao servidor que aceite o conteúdo que vai em anexo para um URI. Se o recurso já existir, modifica-o.
- **DELETE** - Remover o recurso especificado pelo URI.
- **TRACE** - Faz eco o pedido recebido (para testes).
- **OPTIONS** - Retorna os métodos HTTP disponibilizados.
- **PATCH** - Aplica alterações parciais ao recurso no URI.
- **CONNECT** - Encaminha o pedido.

HTTP 2.0

- Surgiu a partir do protocolo experimental SPDY da Google com seguintes objetivos:
  - Melhorar latência percebida sobre HTTP/1.1.
  - Não requere múltiplas conexões paralelas.
  - Retém semântica do HTTP/1.1.
- É um protocolo binário
- Todas as comunicações são feitas numa única conexão TCP.
- Uma stream é um canal virtual dentro de uma conexão e tem um identificador (1, 2, .. n).
- Uma mensagem é uma mensagem HTTP, (request - response) e pode estar dividida em várias frames.
- Multiplexagem: Permite enviar mensagens em várias streams paralelamente.
- Milhões de servidores têm de ser adaptados para usar o protocolo binário, e biliões de clientes terão de atualizar os seus browsers
- Browsers atuais já têm mecanismos de atualização automáticos, mas alguns utilizadores não podem atualizar os seus browsers
- Routers intermédios na rede irão suportar HTTP/1.1 durante pelo menos mais uma década

HTTP/3

- Apesar do protocolo HTTP/2 ter uma utilização mundial de cerca de 34%, já está a ser criado a versão 3
- Baseado no protocolo experimental da Google (QUIC)
- Utiliza o UDP como protocolo de transporte (invés de TCP)

Gestão de nomes

- A gestão de nomes tem um papel importante em sistemas distribuídos. Estes são utilizados para:
  - Identificar entidades
  - Referir localizações
  - Partilhar recursos
  - Associar nomes a objetos

Nomes

- Sequencias de caracteres usadas para denominar recurso como
  - servidores (server1.google.com)
  - impressoras (myprinter1)
  - discos (Macbook HD)
  - ficheiros num sistema de ficheiros (C:/myfile.txt)
- Um recurso pode ter 1 ou mais nomes mas um nome só esta associado a um recurso

Endereços

- Um recurso pode ter um ou mais pontos de acessos designados por endereços
- Endereços podem ser:
  - Endereços IP (86.129.34.12)
  - Endereços MAC (00:0a:95:9d:68:16)
  - Endereços de Memória (0x7fff9575c05f)

Resolução de nomes em sistemas - Nomeação plana

- Sistemas onde identificadores são nomes não-estruturados:
  - PCSALA1 , IMPRESSORA_241 ou Iphone do Manuel
- Não contêm informação nenhuma sobre como localizar o ponto de acesso do recurso

Nomeação plana Soluções simples

- **Broadcast**
  - Pedidos de resolução são enviados para todos os nós da rede
  - Cada nó verifica se é o destinatário
  - O destinatário responde com o seu endereço
  - Consome demasiados recursos na rede
- **Multicast**
  - Pedidos são enviados apenas para nós pertencentes ao grupo
  - O destinatário responde com o seu endereço
  - Mais escalável que broadcast, mas ainda limitado
- **Tabelas de dispersão distribuídas**
  - Estrutura de dados que associa chaves de pesquisa a valores
  - Numa tabela de dispersão distribuída, cada nó tem apenas uma pequena parte da tabela total
- **Tabelas de dispersão distribuídas (Chord)**
  - Algoritmo baseado em tabelas de dispersão distribuídas:
    - Cada nó tem um nó sucessor e um nó predecessor
    - Cada nó reconhece n sucessores e n predecessores
  - Para resolver um nome:
    - O nó inicial verifica se sabe o endereço do destinatário senão, pergunta aos seus sucessores

Resolução de nomes em sistemas - Nomeação estruturada

- Nomes não-estruturados tendem a ser pouco convenientes para redes mais complexas.
- Nomes estruturados tendem a ser compostos por nomes simples e legíveis, organizados num "espaços de nomes" (namespaces)

Nomeação estruturada Soluções simples

- Espaço de nomes
  - O espaço de nomes pode agrupar os vários componentes do nome num formato tipo árvore
  - Raiz: Nome de mais alto nível
  - Diretórios: Subdividem um nome em subnomes
  - Folhas: Representam os recursos que se procura
- Distribuição de nomes
  - O espaço de nomes para redes de larga escala estão normalmente organizados hierarquicamente em camadas lógicas
    - **Camada global**: Nós de mais alto nível e subnomes importantes. Representam grupos de organizações como países, etc
    - **Camada administrativa**: Nós intermédios, que podem representar organizações e pertencer às mesmas organizações
    - **Camada final**: Nós que representam os recursos finais
  - Nós de mais alto nível devem ter maior disponibilidade que os de mais baixo nível.

Resolução de nomes estruturados

- Iterativamente
  - O servidor resolve a parte do nome que conseguir e devolve o restante ao cliente,
  - que reencaminha o pedido para o próximo servidor
  - Consome recursos da rede (latência)
- Recursiva
- O servidor resolve a parte do nome que conseguir e reenvia o pedido a outro servidor, até completar a resolução do nome
- Consome recursos do servidor

Exemplo: DNS

- Protocolo de gestão de nomes na Internet.
- Um nome de domínio é constituído por subnomes separados por pontos.
- Hierarquia
  - O nome mais à direita é o domínio de topo (com, org, pt, etc.)
  - Os domínios dos países (pt, br, us) são geridos pelos países, enquanto genéricos (com, net, etc.)são geridos pela IANA
- Zonas
  - Subárvores da árvore anterior
  - Cada zona é gerida por uma autoridade e está associado a um servido
- Dado a complexidade dos sistemas e número de pesquisas, a resolução de nomes é essencialmente iterativa

Resolução de nomes em sistemas - Nomeação baseada em atributos

- Em certos casos, pode-se querer indexar os nomes dos recursos por atributos:
  - Serviços que oferecem
  - Capacidades que possuam
  - Características que possuam
- Ao pesquisar-se por atributos, reduz-se efetivamente o espaço de pesquisa

Diretórios de serviços

- Em certos casos, pode-se querer indexar os nomes dos recursos por atributos:
  - Serviços que oferecem
  - Capacidades que possuam
  - Características que possuam.
- A pesquisa é feita com base nos atributos pretendidos.
  - Exemplo de uma impressora:
    - Tipo de papel suportado (A3, A4, A5, etc.)
    - Tipo de tecnologia (Inkjet, Laser, etc.)
    - Cores ou Preto-e-Branco

LDAP (Lightweight Directory Access Protocol)

- Mistura conceitos de nomes estruturados com atributos.
- Consiste num número de registos com uma coleção de (atributo, valor)
- A parte estruturada consiste na hierarquização dos vários atributos como se fossem domínios. Esta estrutura chama-se DIT (Directory Information Tree).

Webservices

- Serviços criados de modo a suportar a interação entre máquinas sobre uma rede
- Tendo em conta as operações disponíveis no Webservice, a aplicação solicita uma dessas operações.
- O Webservice efetua o processamento e envia os dados para a aplicação.
- Exemplos:
  - Google Maps API
  - Yahoo Weather API

Recursos na web

- Inicialmente documentos e ficheiros identificados por urls
- Atualmente é um conceito mais abstrato (documentos, ficheiros, vídeos ,serviços ,etc)

Tecnologias

- Algumas das tecnologias mais usadas
  - **AJAX**: Permite aos browsers obterem informação de um servidor de forma assíncrona sem reler a página toda novamente
  - **SOAP**: Protocolo de mensagens que usa XML para trocar informação com um servidor
  - **REST**: Arquitetura para interoperabilidade entre computadores na Internet

REST (Representational State Transfer)

- Servidores expões serviços através de recursos na web
- Exige o mínimo de manutenção de estados no servidor
- É suposto ser mais leve e simples que outros esquemas de serviços AJAX e SOAP

REST Métodos

- **GET** - Obtém Recurso
- **POST** - Cria Recurso
- **PUT** - Atualiza um recurso
- **DELETE** - Elimina um recurso

REST Status Codes

- **200 OK** | Resposta a GETs, PUTs, DELETEs
- **201 Created** | Resposta a POSTs bem sucedidos
- **400 Bad Request** | Dados enviados mal formatados
- **403 Forbidden** | Não tem permissões para aceder a recurso
- **404 Not** Found | Recurso não encontrado
- **500 Server Erro** |Internal Server error