

1 – Qual é a diferença entre HTTP e HTTPS?

O Http é inseguro se comparado com o Https, por que foi criado somente com a intenção de obter dados da rede usando o método get, trabalhando com a transferência de texto, passa os dados e senhas como texto. Com o avanço da tecnologia, houve a necessidade de criar um protocolo mais seguro para passar dados, então foi criado o protocolo Https. O Https pode-se dizer que é uma evolução do Http, pois ele faz o processo de criptografia dos dados, usando o método da chave simétrica, onde é usada apenas uma chave. Outro modo utilizado é o método de chave assimétrica, onde se tem chaves públicas e privadas para recepção do conteúdo.

Conclusão:

- Http: menos seguro;
- Https: mais seguro.

2 – Qual o formato de uma requisição HTTP?

O formato é composto nos seguintes campos:

- Linha inicial(Request-line);
- Linhas de cabeçalho(Request-header);
- Linha em branco obrigatória;
- Corpo de mensagem opcional.

A linha inicial é composta em três partes:

- Método(method);
- Identificação do URI(Request-URI);
- Versão do Http(HTTP-Version).

3 – Qual o formato de uma resposta HTTP?

O formato é composto pelos seguintes campos:

- Linha inicial(Status-Line);
- Linhas de cabeçalhos(ResponseHeader);
- Linha em branco obrigatória;
- Corpo de mensagem opcional.

4 – Quando um servidor não encontra um recurso, quais os principais códigos de status que existem? Por exemplo 404? O que significa?

No Http temos vários códigos para retornar a requisição, respondendo a solicitação do cliente, entre eles:

301 - moved permanently- Redireciona o navegador de forma permanente para uma URL diferente, o cliente deverá usar a nova URL em vez da original;

400 – indica que o cliente apresentou uma solicitação Http inválida;

401- indica que o servidor requer autenticação Http antes de ser atendido;

403 – indica que ninguém está autorizado a acessar o recurso solicitado;

404 – recurso solicitado não encontrado;

5 – Quais as principais diferenças do HTTP v1 e HTTP v2?

O Http v1 é um protocolo sequencial, abrindo apenas uma conexão TCP, solicitando um arquivo por vez, necessitando que os servidores abrem múltiplas conexões, que pode travar o sistema e prejudicar outros usuários que requisitem o mesmo recurso. Já o Http v2 teve seu mecanismo de codificação foi redesenhado sem alterar a semântica dos métodos verbos e cabeçalhos. Com a comunicação quebrada em frames, em uma única conexão TCP.

As requisições Http v1 possuem cabeçalhos com informações sobre o recurso e suas propriedades. O Http v2 possui os cabeçalhos comprimidos através do algoritmo HPACK, reduzindo o tamanho de cada transferência, além de manter atualizada uma lista indexada de campos de cabeçalhos