Senhas Descartáveis (OTP)

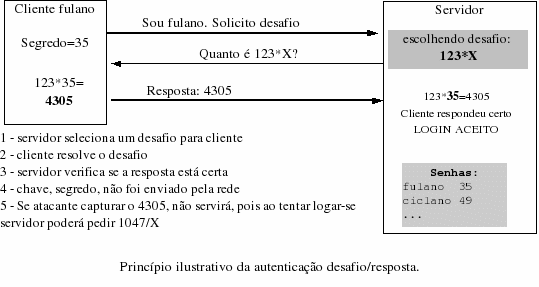
Encaixa-se no modelo de identificação positiva baseando-se no que o usuário sabe, um segredo que o usuário tenha conhecimento. Senha é um artefato sigiloso que é conhecido apenas por aquele que a utiliza, caso este cenário deixe de ser verdade sua utilidade é perdida e a segurança comprometida. É considerado um modelo fraca devido as falhas do usuário, entre elas podemos citar: escolha de senhas fracas (fáceis de serem descobertas), quebra de sigilo (usuário conta sua senha para alguém ou gravam em algum local de fácil acesso).

A evolução das senhas foi a criação de senhas descartáveis (One-Time Password) que possuem um tempo finito de vida (um único acesso) e assim que expiram deixam de ser válidas. As senhas são geradas através de hardwares ou softwares com diferentes lógicas e algoritmos de criptografia e enviadas para locais que o usuário tenha fácil acesso como celular, e-mail ou qualquer outro dispositivo conhecido pelo gerador da senha. Devido a alta volatilidade da senha torna-se mais difícil acertar por acaso com ataques de *brute force* ouaplicar engenharia social dado que o usuário não influência na geração do segredo. Porém é necessário que a rede pela qual a informação fora enviada seja segura para evitar a interceptação da informação. Há casos conhecidos, no qual, senhas descartáveis envidas por mensagem para *smartphones* através de falhas de segurança do protocolo SS7 foram interceptadas e facilmente descriptografadas por ataques hackers.

Desafio/Resposta

O modelo de desafio/resposta consiste em gerar um desafio relacionado ao segredo. O desafio assim como a senha testa conhecimento do usuário, entretanto, não expõem o segredo diretamente.

O protocolo passa por três etapas simples, dentro de cada uma ainda há outros requisitos para aumentar a segurança. Um usuário qualquer inicia uma solicitação a um servidor querendo ser autenticado, o servidor então envia um desafio cuja senha do usuário é a chave para resolvê-lo. Na segunda etapa o usuário insere sua senha, porém a senha não é enviada, a partir dela é feito um calculo e enviado a solução deste calculo, quando chega no servidor ele realiza o mesmo calculo com a senha que está em sua base e verifica se as respostas foram iguais. Se as respostas foram idênticas o usuário acessa o sistema, senão acesso negado.



O exemplo acima é ingênuo e utilizado apenas para fácil compreensão, este modelo trabalha com funções *Hashs* de alta complexidade como por exemplo o algoritmo SHA-2 criado pela agência nacional de segurança dos Estados Unidos. Portanto não importa quantas vezes for interceptado a resposta, esta informação sempre será inútil.

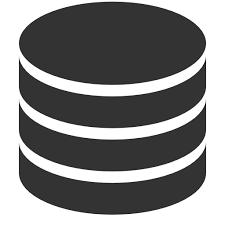
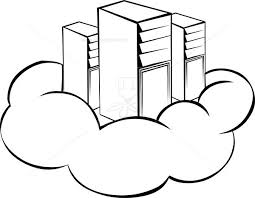
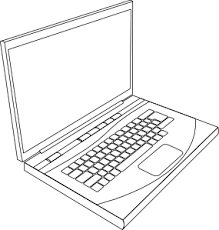
O modelo desafio/resposta é muito utilizado nos dias atuais como podemos notar através do protocolo SSH (Secure Shell) que permite login de acesso remoto de forma segura mesmo em redes não seguras, é atualmente o protocolo utilizado pelo github para acesso aos repositórios remotos. Preferencialmente utilizado em sistemas UNIX, porém com alguma compatibilidade em sistemas windows.

Token de autenticação

O conceito de *token* vem de um dispositivo físico que gera senhas que podem ser utilizadas por um tempo finito ou quantidade de utilizações. Muito parecido com o conceito de senhas descartáveis podemos notar que a evolução da autenticação por *token* tenta unir vários modelos diferentes para criar um ambiente mais seguro e menos propenso a falhas.

Os *tokens* digitais amplamente utilizados na atualidade autenticam e transmitem informação em ambientes seguros com o auxílio de protocolos de segurança, criptografias e certificados digitais. Há diferentes tipos de implementações e arquiteturas que vão desde o servidor realizar as operações de autenticação, gerar o *token* e ler/gravar informações até arquiteturas que dividem estas tarefas criando APIs que apenas autenticam, que geram o *token* e leem e gravam dados da base.

Um *token* funciona da seguinte maneira, dada uma solicitação de autenticação do usuário em uma rede protegida por um protocolo de segurança qualquer, o servidor responde com uma concessão de acesso e devolve ao usuário o *token* criptografado com ID gerado aleatoriamente com um tempo finito de validade. A partir daí toda comunicação e transmissão de dados entre aplicação e servidor será feita com este *token* criptografado o que protege os dados do usuárioqueperiodicamente altera o ID gerado dificultando a descoberta da assinatura.



**Conexão Segura (SSL)**

Usuário

Senha

Valida informações no banco

Devolve um token criptografado (RSA)

Requisição POST

Resposta a requisição

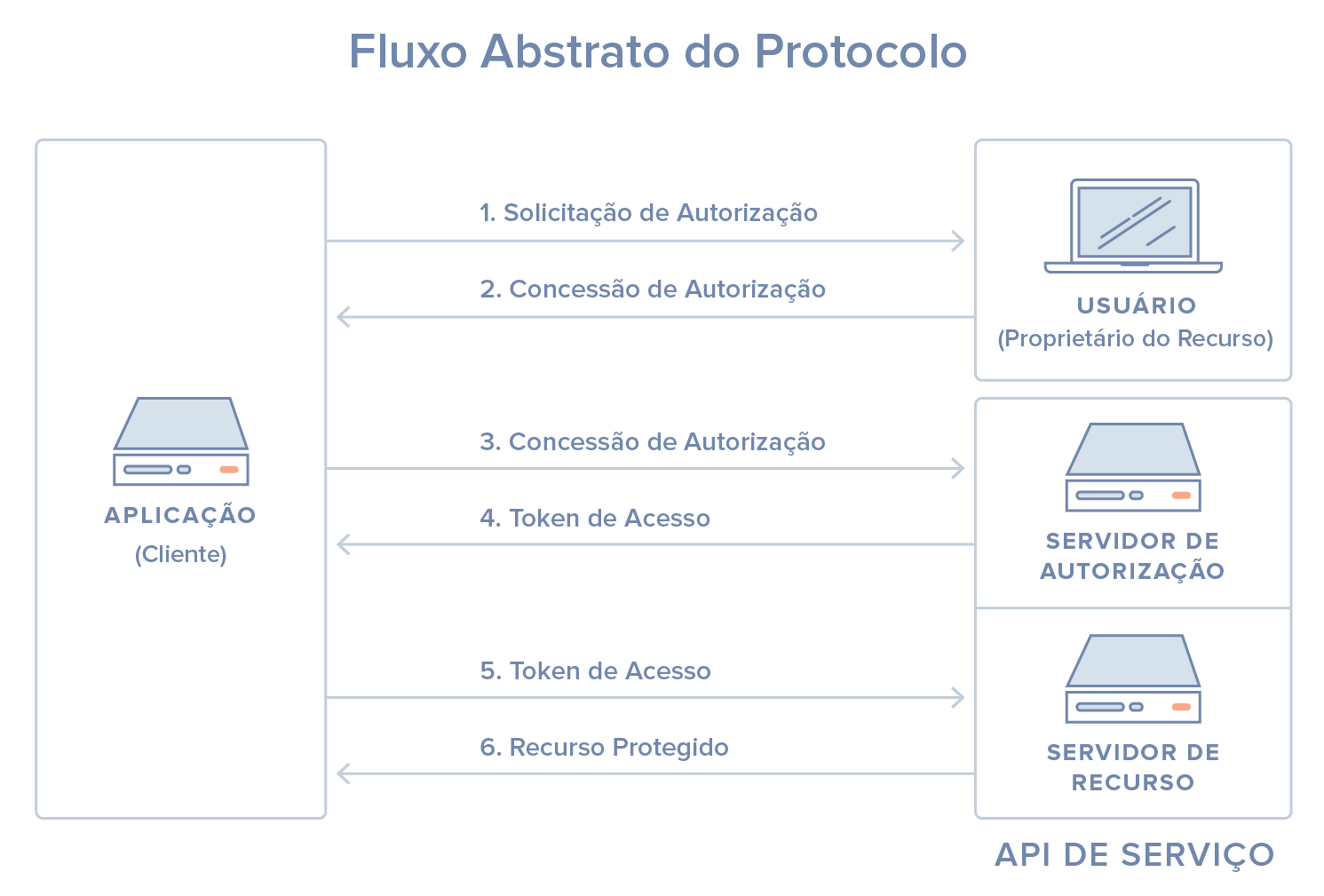
Token vai no cabeçalho da requisição

Requisição POST

Resposta a requisição

Valida assinatura do token e busca informação pedida

Um outro modelo seria baseado em uma aplicação que terceiriza o trabalho de validações e acesso a recursos como é o modelo OATH que realiza a autenticação através de logins da google, facebook, microsoft, windows... e então liberar o *token* de acesso.



A desvantagem clara deste modelo é quantidade excessiva de requisições necessárias para se obter a liberação de acesso, consumindo mais tempo e recursos. Além disso o controle da informação fica totalmente na mão do terceiro.

Há diversos outros tipos de *tokens* com suas vantagens e desvantagens, em sua grande maioria seguem o mesmo pedrão base alterando apena os algoritmos de geração de códigos pseudoaleatórios e algoritmos de criptografia.