IF678 - Infraestrutura de Comunicação
Prof. Kelvin Lopes Dias
Pro Sistema de votação

Projeto Sockets: Sistema de votação tolerante a falhas

- Data de lançamento: 31/01/2025
- Data de entrega: 12/03/2022 **(60% da nota – repositório Git com códigos fonte e 40% da nota - relatório);**
- Equipe: <= 3 integrantes
- Criar um repositório privado no GitHub ou GitLab e adicionar os professores como colaboradores a fim de acompanhar o desenvolvimento!
    - Kelvin - kld@cin.ufpe.br
    - Katarine - mksb@cin.ufpe.br
    - Monitores

1. **Regras:**
    a. Os projetos devem ser implementados em Python 3 (preferencialmente), Java, C ou C++;
        i. Os projetos devem rodar obrigatoriamente no sistema operacional Linux.
    b. Frameworks ou bibliotecas que facilitam a manipulação com sockets não podem ser utilizados;
    c. Deve ser elaborado um relatório detalhado para cada projeto desenvolvido;
        i. Utilize diagramas e trechos do código;
        ii. O relatório deve conter uma seção com os passos que devem ser seguidos para executar o código fonte de forma satisfatória;
        iii. Apresentar telas com a execução do projeto.
        iv. O relatório também deve conter capturas do wireshark comprovando o devido funcionamento do projeto.
    d. O código-fonte deve ser entregue;
        i. O repositório deve conter um README.md fornecendo o passo a passo para a execução dos códigos.
    e. Cópias acarretarão em nulidade dos projetos das equipes envolvidas;
        i. Um algoritmo de controle de autoria será utilizado para verificar as cópias;

2. **Descrição do Sistema de Votação**

Etapa 1: Criar uma aplicação cliente-servidor

a. O sistema de votação deve conter 2 candidatos e 15 eleitores (clientes), sendo que cada eleitor pode votar apenas uma vez. Após votar, o eleitor deve ter a opção de se desconectar do sistema. Contudo, pelo menos um eleitor deve permanecer sempre conectado, para continuar acompanhando os votos dos candidatos. Caso o cliente se desconecte, ao se reconectar, ele deve receber uma notificação informando que já votou e não poderá votar novamente, garantindo que o voto do eleitor é único.
b. O servidor de votação deve armazenar os candidatos, os votos recebidos e o histórico de eleitores que já votaram.
c. Quando o sistema detectar que os 15 votos foram recebidos, a votação deve ser encerrada, e os eleitores que ainda estiverem logados devem receber o resultado.

Etapa 2: Serviço de Autenticação (HTTP)

a. O sistema de votação deve ser seguro contra fraudes, ou seja, deve garantir que o voto do eleitor X foi realmente feito por ele.
b. Ao ser inicializado, o eleitor deve gerar um par de chaves pública e privada. A chave privada deve ser armazenada no sistema do eleitor e utilizada durante o processo de votação, gerando uma assinatura digital, que assegura que o voto foi realmente dado pelo eleitor.
    i. Vale destacar que, ao se desconectar, o eleitor não pode gerar um novo par de chaves; ele deve usar a chave gerada inicialmente.
c. As chaves públicas devem ser armazenadas em um serviço de autenticação. Assim, após a geração das chaves, o eleitor deve usar o método HTTP-POST para enviar sua chave pública ao serviço de autenticação, onde ela será armazenada para permitir consultas de validação da assinatura digital gerada.
d. Os servidores de votação, ao receberem um voto, devem validar a assinatura digital do usuário. Para isso, devem usar o método HTTP-GET para consultar a chave pública do eleitor e verificar a assinatura. Uma vez que a assinatura seja validada, o voto pode ser armazenado.
e. O serviço de autenticação também deve estar disponível no servidor de nomes.

Etapa 3: Sistema de descoberta

a. O cliente não deve conhecer inicialmente o endereço (IP, porta) do servidor de votação.
b. Crie seu próprio Servidor de Nomes: Quando o serviço de votação for iniciado, ele deve ser registrado no servidor de nomes para ficar disponível para os clientes.
    i. Quando o cliente solicitar um serviço, o servidor de nomes deve retornar o conjunto (IP, porta).
    ii. Caso o serviço não exista, o servidor de nomes deve retornar uma mensagem "Not Found".

Etapa 4: Tolerância a falhas

a. Criação de redundância no sistema de votação: Devem ser inicializados dois servidores de votação, e ambos devem ser registrados no servidor de nomes.
b. Inicialmente, apenas um servidor deve ser retornado ao cliente. Após o primeiro servidor receber 8 votos, ele deve parar a execução, e os usuários devem ser redirecionados para o segundo servidor.
c. Mesmo com o redirecionamento, os usuários que estavam conectados devem permanecer conectados, e os dados de votação não devem ser perdidos, impedindo que os votos sejam descartados ou que eleitores votem novamente.
    i. Dica: Crie uma base de dados centralizada.

3. Dicas:

a. Não deixe para começar os projetos mais tarde. Comece logo! São apenas 40 dias!
b. É (quase) impossível fazer os projetos de “virada”... Mesmo em duas semanas de “viradas” ;-)
c. É interessante fazer um cronograma de atividades de desenvolvimento dos projetos;
d. Considere épocas de provas e o desenvolvimento de outros projetos em outras disciplinas;
e. Não se esqueçam de dar atenção ao relatório!
f. Lembrar de adicionar os professores e o monitor como colaborador do repositório Git para acompanhamento do desenvolvimento!
g. Marcaremos alguns dias de acompanhamento dos projetos;
    i. Entretanto, a responsabilidade do desenvolvimento dos projetos é da equipe;