Trabalho Prático 2 (TP 02) – Disciplina de Sistemas Distribuídos

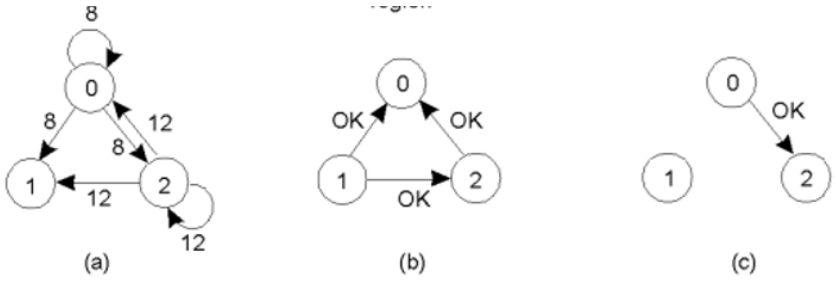
Neste trabalho o grupo de alunos (até 3 alunos) irá implementar um protocolo de sincronização distribuída. O grupo poderá implementar qualquer solução distribuída da literatura, ou seja, sem controle central de um nó do cluster, que geralmente mantém uma pilha de autorização de acesso ao recurso. ESTE TIPO DE SOLUÇÃO NÃO SERÁ VÁLIDA.

Qualquer linguagem será aceita. Não serão aceitas soluções apenas multicore.

Preliminares: Independente do protocolo a ser implementado, o seguinte cenário deve ser adotado: criar um conjunto de clientes (em containers ou VMs ou em máquinas reais mesmo) que farão acesso de escrita a um recurso R hipotético. Mínimo de 5 clientes. Cada cliente conhece apenas um elemento do cluster denominado CLUSTER DE SINCRONIZAÇÃO DE ACESSO AO RECURSO R (abreviação Cluster Sync). Cada cliente possui um ID único e a cada pedido de acesso ao recurso R, o mesmo envia o timestamp para garantir pedidos únicos. O Cluster Sync deve ser composto por, no mínimo, 5 processos encapsulados em containers ou VMs ou máquinas reais. O cliente recebe apenas a mensagem COMMITTED de um elemento do Cluster Sync após o acesso a R ser concluído. Após receber uma mensagem COMMITTED, o cliente entra em estado de espera (Sleep de 1 a 5 segundos calculado aleatoriamente) e volta a pedir acesso de escrita ao recurso R. Cada cliente deve pedir entre 10 e 50 acessos de escrita ao recurso R, forçando, desta forma, a concorrência e, consequentemente, a validação do protocolo de exclusão mútua.

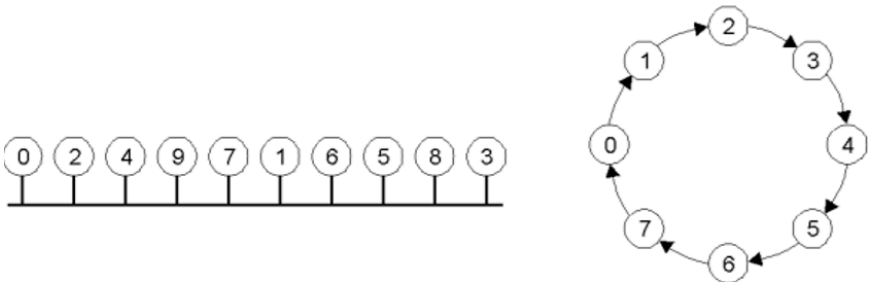
Alternativa 1 de Protocolo:

Qualquer elemento do Cluster Sync ao receber uma solicitação de um cliente obtém seu timestamp e o envia aos demais membros do Cluster Sync (incluindo ele mesmo). Também é enviado o ID (Ex. Peer1, Peer2, etc.) do elemento do Cluster Sync a cada solicitação do cliente, assim como a própria solicitação do cliente. Cada elemento do Cluster Sync avalia os timestamps recebidos e os respectivos IDs dos elementos do cluster, sabendo, com isto, sua posição temporária no acesso ao recurso R. Cada elemento do Cluster Sync responde com OK aos pares do cluster com timestamp menor do que o seu próprio. Elementos do Cluster Sync sem pedidos de cliente respondem a todos que enviaram pedido de acesso a R. O elemento do cluster com 5 OKs entra na seção crítica por um tempo (Sleep de 0.2 a 1 segundo) e depois responde tanto ao elemento do Cluster Sync com timestamp imediatamente superior ao seu e ao cliente. Vejamos a figura ilustrativa:



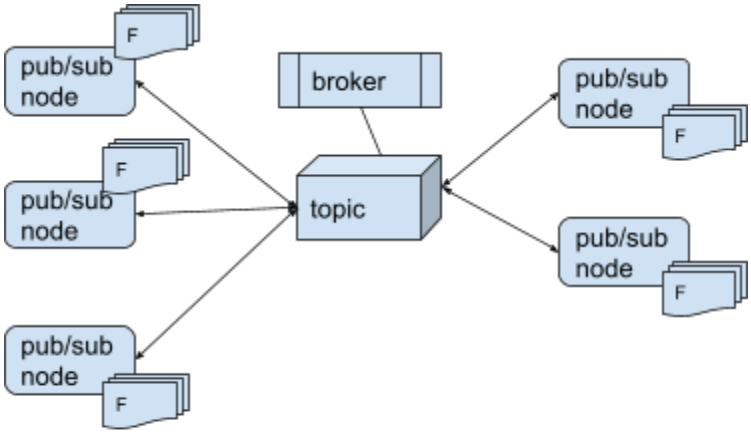
Alternativa 2 de protocolo:

Cada elemento do Cluster Sync conhece apenas um elemento de tal cluster, formando um anel. É criado um vetor com o mesmo tamanho do Cluster Sync e este passa a percorrer o anel tão logo o mesmo esteja criado. Vamos chamar tal vetor de token. Cada elemento do cluster escreve apenas na sua posição do token. Qualquer elemento do Cluster Sync ao receber uma solicitação de um cliente obtém seu timestamp, aguarda o token chegar, escreve no mesmo e o repassa. Após a inserção de um timestamp e o pedido do cliente no token, e o mesmo percorrer o anel completamente, o elemento do Cluster Sync que efetuou tal inserção já sabe se o seu timestamp é o menor gerado. Quando o token passa por um elemento do Cluster Sync e não há solicitação de um cliente, tal elemento escreve NULL na sua posição do token. Após o token retornar a cada elemento do Cluster Sync é possível saber se o mesmo entra ou não na seção crítica. Após entrar na seção crítica por um tempo (Sleep de 0.2 a 1 segundo), o elemento do Cluster Sync escreve NULL no token, responde ao cliente e repassa o token. Vejamos a figura ilustrativa:



Alternativa 3 de protocolo:

Cada elemento do Cluster Sync conhece apenas um broker pub/sub online 24x7 e tolerante a falha (Redis, RabbitMQ, Kafka ou qualquer outro). Qualquer elemento do Cluster Sync ao receber uma solicitação de um cliente envia uma mensagem de Pub a um tópico denominado R topic ou a uma fila denominada R Queue. Nesta mensagem de Pub é inserido apenas o ID do elemento do Cluster Sync, o pedido do cliente e a primitiva ACQUIRE. Cada elemento do Cluster Sync é também um consumidor, portanto recebe todas as mensagens enviadas pelo broker e o mais interessante: NA MESMA ORDEM FIFO, por exemplo. Cabe a cada elemento do cluster ficar avaliando sua fila F (PRIVADA) de mensagens do broker para saber se pode ou não entrar na seção crítica. Após entrar na seção crítica por um tempo (Sleep de 0.2 a 1 segundo), o elemento do Cluster Sync envia uma mensagem de Pub ao mesmo tópico ou fila do broker, mas neste caso contendo seu ID, o pedido do cliente e a primitiva RELEASE. Note que para entrar na seção crítica (acessar R para escrita), cada elemento do Cluster Sync deve olhar a sua ordem de ACQUIRE e também a ordem de RELEASES, assim como o pedido do cliente. Vejamos a figura ilustrativa:



A fila F com acquires e releases possui a mesma ordem que é garantida pelo broker