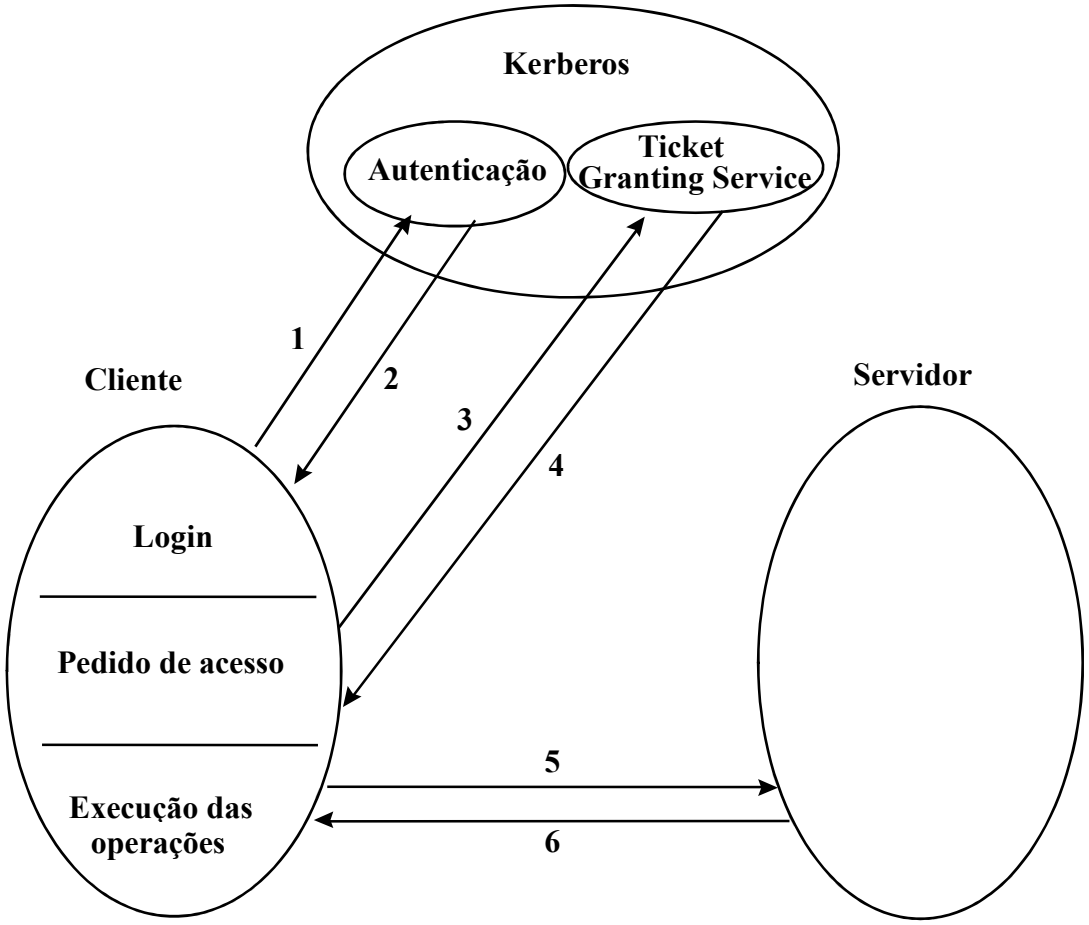
LETI/LEIC 2015-2016, 2º Teste de Sistemas Distribuídos

14 de junho de 2016

Responda no enunciado, usando apenas o espaço fornecido. Identifique todas as folhas.
Uma resposta errada numa escolha múltipla desconta 1/(N-1) do valor da pergunta (sendo N alternativas).
Duração da prova: 1h30m

Grupo I [7,9 valores]

1) Considere o seguinte diagrama que representa um sistema **Kerberos**.



a) [0,8] Complete a seguinte legenda para descrever o que acontece em cada passo do protocolo, evidenciando todos os artefactos relevantes:

1. O cliente identifica-se enviando um pedido de início de sessão para o SAut
2.
3.
4.
5.
6. O servidor envia o resultado da operação ao cliente, cifrado com Kcs

b) [0,6] Que chaves devem ser partilhadas antes do protocolo se poder executar?
Para cada chave referida, indique quem as conhece.

Kc partilhada por C e SAut
Ks partilhada por S e TGS
Ktgs partilhada por SAut e TGS.

c) [0,6] Como pode o Servidor ter a certeza que o *ticket* que recebe é autêntico?

Porque o ticket vem cifrado com Ks e apenas TGS (kerberos) conhece a chave Ks além de S

d) Um *ticket* Kerberos contém os seguintes itens: X, Y, T1, T2, K.

i) [0,4] O que são os valores X e Y ?

ii) [0,5] Qual é a função da chave K?

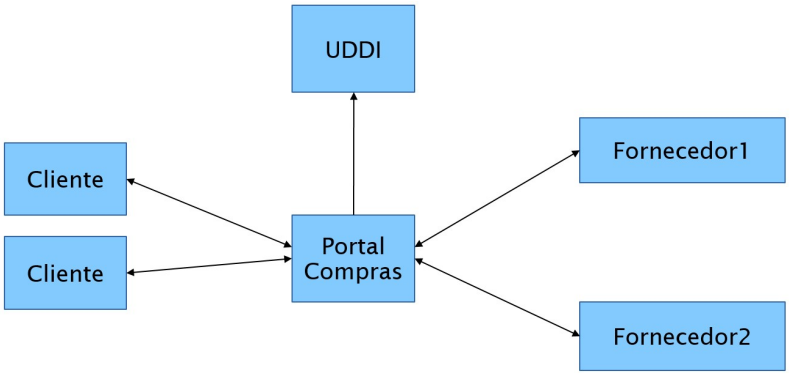
iii) [0,7] Considera que T1 e T2 têm alguma relação com ataques de "força-bruta"? Justifique.

T1 e T2 definem o intervalo de validade do ticket.
Ao definir um tempo limite para a duração do ticket, vão também garantir que
mesmo que um atacante faça um ataque de força bruta o resultado prático não irá compensar.

iv) [0,6] Qual seria a vantagem de cifrar o *ticket* Kerberos com uma cifra por blocos em modo CBC (Cipher-Block Chaining) em vez de modo ECB (Electronic Code Book)? Justifique.

ECB cifra um bloco de cada vez. CBC realimenta o bloco anterior na cifra do bloco seguinte.
CBC é mais seguro porque permite esconder padrões na informação cifrada.
Dado que os tickets kerberos têm campos que podem ser repetidos, CBC seria mais aconselhável
para não os expor padrões nos tickets.

2) Considere um cenário de um Portal de Compras, baseado na tecnologia de **Web Services**, no qual se pretende garantir **segurança** para a comunicação entre o Portal e os Fornecedores.



A segurança do sistema é baseada em certificados digitais de chave pública seguindo a norma **X.509**.

a) [0,7] Em que consiste um certificado digital de chave pública?
Refira especificamente o que contém o certificado.

b) [0,7] O UDDI da figura permite o acesso sem autenticação.
Mesmo assim, o UDDI poderia ser usado para distribuir os certificados evitando ataques de *"man-in-the-middle"*? Justifique como.

Sim, porque a proteção dos certificados não depende do canal de informação.
A verificação da assinatura da CA permite validar o certificado
mesmo que o man-in-the-middle tentasse trocar a chave contida no certificado.

c) [0,9] Descreva de forma esquemática o conteúdo de uma mensagem SOAP que seria enviada do Portal para o Fornecedor de modo a autenticar o emissor.
Assuma que o Fornecedor já conhece o certificado digital do Portal.

```
<soap:envelope>
  <soap:header>
    <id-emissor>…
    <marca-temporal e/ou nonce>…
    <assinatura digital> resumo da mensagem, cifrado com chave privada do emissor
  <soap:body>
    (conteúdo normal)
</soap:envelope>
```

d) Considere um "*replay attack*" à mensagem construída na alínea anterior.
   i) [0,4] O que deve fazer o emissor para impedir o ataque?

   ii) [0,4] O que deve fazer o recetor para impedir o ataque?

e) [0,6] O que será necessário para garantir o **não-repúdio** dos pedidos de encomenda do Portal para o Fornecedor?

Grupo II [5,9 valores]

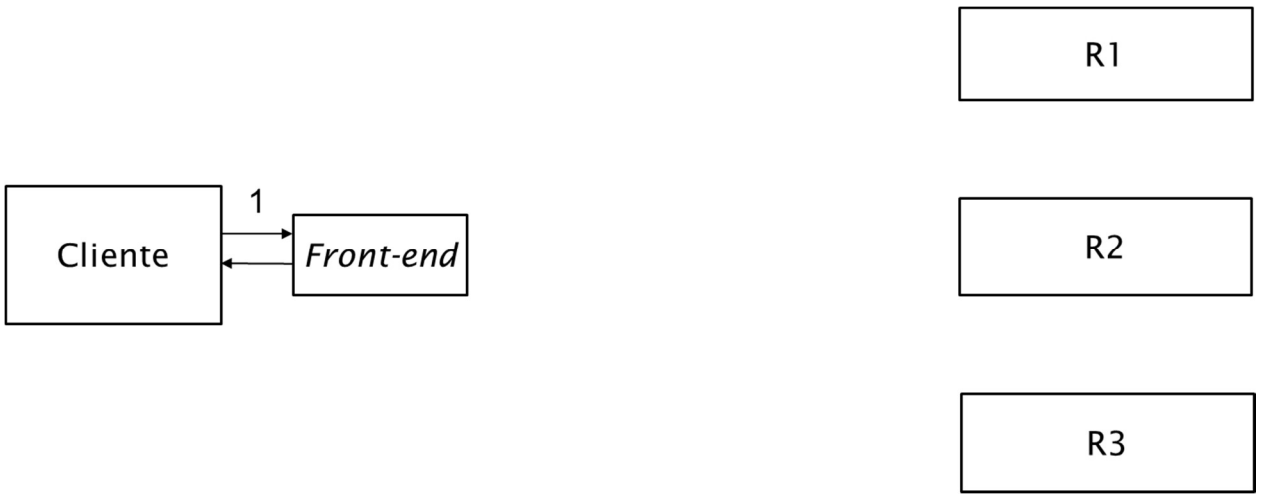
Considere um sistema replicado em que um cliente (C) comunica através de um *Front-end* (FE) com um conjunto de servidores: R1, R2, R3. As operações são apenas de leitura (R) e de escrita (W).

1) O protocolo usado é o ***primary-backup*** semelhante ao das aulas teóricas. R1 é o primário.

   Assuma que os servidores são de falha silenciosa, que o sistema é síncrono, que a rede não tem falhas permanentes e que garante que as mensagens são processadas pela ordem de envio (FIFO).
   O primário contacta diretamente os secundários e envia a cada secundário em cada período P uma mensagem de "I'm alive". Considere que existe um protocolo para na ausência de mensagem de "I'm alive" do primário, os secundários decidirem qual é o novo primário.

   a) [0,5] Dê um exemplo de uma falta silenciosa neste protocolo.

b) [0,8] Considere que o cliente quer escrever na variável X o valor 22 – W(X,22).
Complete o diagrama abaixo indicando as mensagens através de setas numeradas e acrescente a legenda necessária para cada número. Não represente as mensagens de "I'm alive".

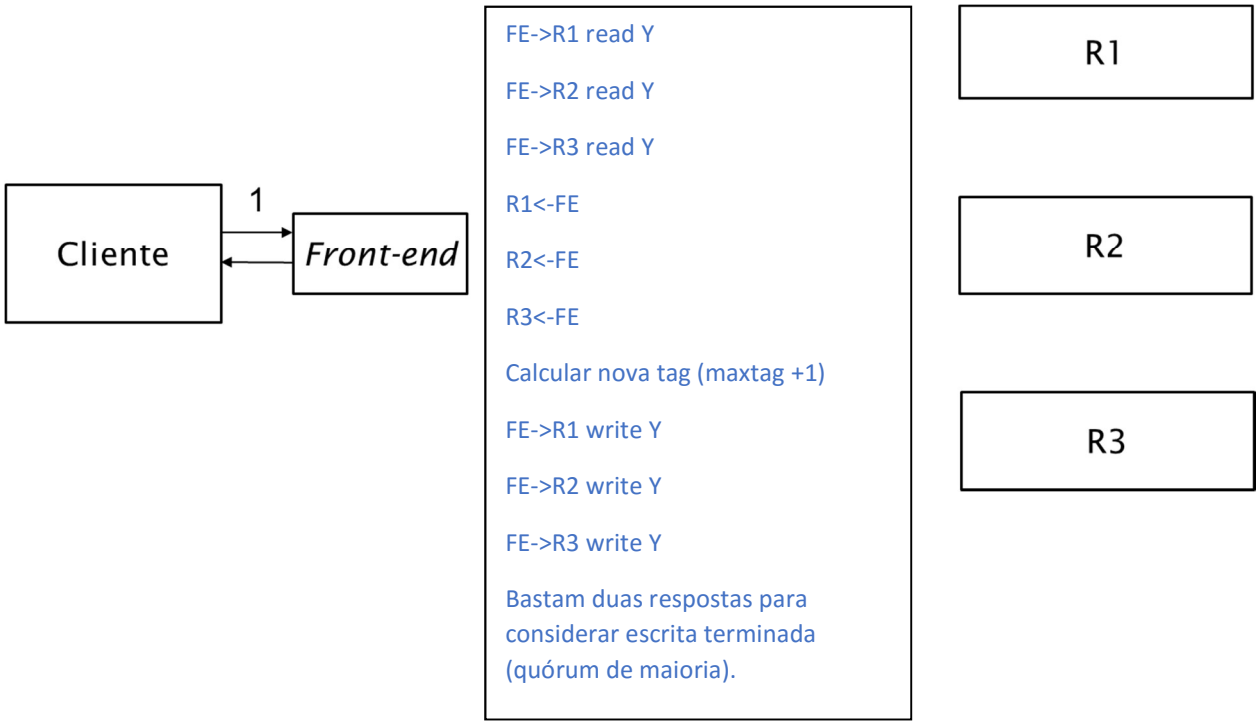


c) [0,6] Quantas faltas de servidores pode tolerar este sistema. Justifique explicitando os pressupostos que considerou.

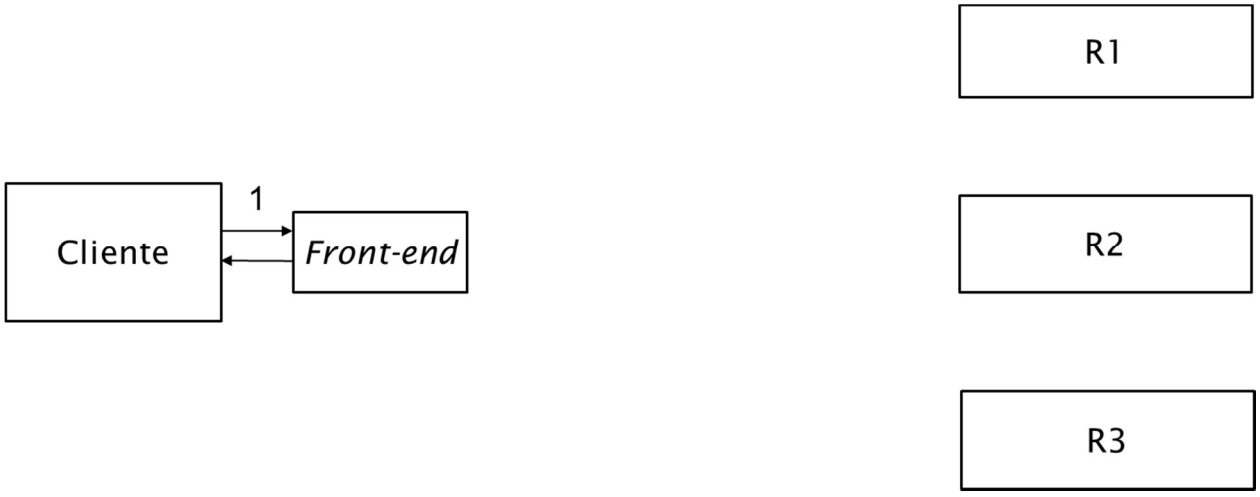
d) [0,8] Qual será o tempo máximo de recuperação do sistema em caso de falha de R1?
Use tMaxEscolha para representar o tempo máximo de escolha do novo primário.
Defina outros parâmetros de que necessite.

2) Considere agora que o protocolo usado é o ***quorum consensus*** e que o sistema é assíncrono.
O valor inicial da variável Y é 0.

a) [0,8] Considere que o cliente quer executar a operação W(Y, 6).
Complete o diagrama com todas as mensagens numeradas e respetiva legenda.
Considere que R1 está "em baixo" no momento da mensagem de escrita que lhe é dirigida.



b) [0,6] Considere agora a operação R(Y) que se executa posteriormente à escrita anterior.
R1 está agora "em cima", mas R3 está "em baixo". Complete o diagrama.

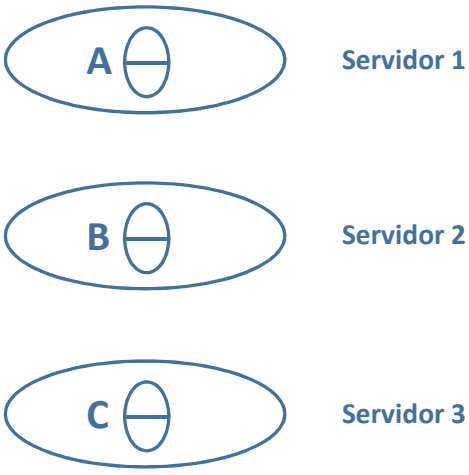


c) [0,7] Nesta situação, a leitura R(Y) vai devolver o valor correto? Justifique como pode o *front-end* escolher corretamente entre o valor 0 e o valor 6.

d) [0,6] Quantas réplicas no total seriam necessárias para tolerar 2 servidores em baixo?

e) [0,5] Dê um exemplo de uma falta bizantina neste protocolo.

Grupo III [3,1 valores]



Considere um sistema distribuído ilustrado na figura acima e os programas de dois clientes que utilizam **transações distribuídas**. Assuma que os servidores usam um sistema transacional com política de sincronização pessimista.
Os programas seguem uma sintaxe semelhante à do livro Coulouris et al., *Distributed Systems*.

Cliente 1

```
T = OpenTransaction;
        Value = A.read();
        B.deposit(Value*0,05);
CloseTransaction;
```

Cliente 2

```
T = OpenTransaction;
        Value = C.read();
        B.deposit(Value*0,05);
CloseTransaction;
```

1) Considere o seguinte Diário (*Log*) no Coordenador:

a) [0,5] O que pode inferir que aconteceu até esse instante. Justifique.

b) [0,4] O cliente 1 executa a operação remota de read.
   i) A operação é executada sobre o coordenador e deverá ser registada no seu log.
   ii) A operação já foi executada dando origem ao Join do servidor 1.
   iii) A operação read obriga a fazer Join e o servidor informa o coordenador do resultado do read.
   iv) O Join foi efetuado no início da transação por todos os servidores envolvidos.

   ii

c) [0,4] Suponha que ambos os clientes continuam a executar os seus programas em paralelo e executam a instrução B.deposit(). O que sucede:
   i) Não é possível porque apenas um pode aceder ao objeto B, uma das transações aborta.
   ii) A transação 272 que começou depois da 271 bloqueia-se.
   iii) Uma das transações bloqueia-se num trinco associado ao objeto B, não existindo qualquer situação de aborto.
   iv) Nesta situação os servidores avisam o coordenador de um conflito no acesso.

   iii

2) Considere agora que o cliente 2 chegou à execução da operação CloseTransaction().
   a) [0,6] Preencha o Log do coordenador desde o instante do CloseTransaction() até ao instante em que o coordenador pode tomar uma decisão de commit para finalizar a transação.
   Use uma linha para cada entrada no *log*. Procure evidenciar todos os registos necessários.

TX 272 C estado inicial 2PC; TX 272 C contacta S2 e pede voto; TX 272 C contacta S3 e pede voto;
TX 272 C recebe o voto de S2; TX 272 C recebe o voto de S3.
(Se os votos forem todos positivos, pode optar por commit)
TX 272 C regista decisão commit global.

   b) [0,4] Suponha que o coordenador envia a mensagem commit aos dois servidores e o servidor 2 não responde com ACK
      i) O coordenador decide abortar a transação.
      ii) O coordenador mantém a transação aberta até que o servidor 2 envie o ack ou o contacte.
      iii) O coordenador fecha a transação porque este ack final é facultativo uma vez que o commit já foi efetuado.
      iv) Esta situação não existe porque se o servidor vota favoravelmente tem de fazer ack em seguida.

      ii

3) O protocolo de *two-phase commit* tolera faltas temporárias da comunicação ou atrasos das mensagens na rede. Explique porquê:

a) [0,4] Tolera apenas faltas de **paragem temporárias da rede**.

Porque se for uma falta permanente, o protocolo pode ficar bloqueado.
Por exemplo, quando o coordenador decide COMMIT e depois tem que informar os participantes.
Caso um participante tenha uma falta permanente a transação não termina.

b) [0,4] Tolera atrasos nas mensagens **apesar de usar timeouts**.

Se uma mensagem se atrasar, e der timeout,
o coordenador pode tomar decisão de abort ou então aguardar mais um pouco.
Entretanto pode chegar a mensagem atrasada ou então uma retransmissão da mesma.

Grupo IV [3,1 valores]

1) [0,4] Praticamente todos os sistemas cliente servidor têm associado um espaço de nomes e um gestor de nomes. Justifique a utilidade desta associação no caso do SUN-RPC. Procure mostrar que valor acrescenta ao DNS já existente.

O DNS permite traduzir o nome de máquina em endereço IP.
No entanto, o DNS não detalha o porto onde corre o programa servidor pretendido.
O serviço de nomes de um RPC permite tornar o porto do servidor dinâmico.

2) [0,4] Considere o nome BANCOPROG presente neste extrato de uma especificação em Sun-RPC:

```
program BANCOPROG {
  version BANCOVERS {
    criarRet CRIAR(criarIn) = 1;
    saldoRet SALDO(int) = 2;
    resultado DEPOSITAR(contaEvalor) = 3;
  } = 1;
} = 0x20000005;
```

A. Nome de um serviço local a um servidor, numa rede a executar Sun-RPC.
B. Nome impuro.
C. Não é um nome (no sentido informático) porque não é possível garantir a sua unicidade referencial.
D. Nome de um serviço global a uma rede a executar Sun-RPC.

A

3) [0,4] O URI `http://store.com:8080/StoreWS` indicado neste extrato de WSDL serve para:

```
<wsdl:service name="StoreWS">
      <wsdl:port name="StoreWSPort" binding="tns:StoreWSSoapBinding">
            <soap:address location="http://store.com:8080/StoreWS"/>
      </wsdl:port>
</wsdl:service>
```

A. Identificar o serviço.
B. Definir o *namespace* que deve estar presente nos pacotes SOAP de invocação.
C. Localizar o servidor.
D. Não tem uma real função porque é obrigatório usar o UDDI.

C

4) No DNS existe uma política de tolerância a faltas.
   a) [0,5] Explique em detalhe qual é.

Existe servidor *master* onde são definidos os nomes,
e existem servidores *slave* que têm cópias apenas de leitura.
Caso o servidor *master* falhe, os *slaves* continuam a responder a pedidos de leitura (mais frequentes)

   b) [0,5] Contudo o protocolo de replicação *primary- backup* não é utilizado no DNS. Explique qual a razão de não ser usado.

DNS apenas garante leituras, *primary-backup* implica leituras e escritas e provas de vida.
O envio de atualizações e provas de vida para todos os secundários iria limitar o número de réplicas.

   c) [0,5] No DNS são utilizadas *caches* nos clientes que não existem no *primary-backup*. Justifique a razão desta diferença.

As *caches* permitem evitar consultas aos servidores, reaproveitando respostas recentes.
No entanto, as *caches* podem estar inconsistentes porque só são atualizadas periodicamente.
Isto significa que se poderiam receber respostas inconsistentes, o que não acontece no primary/backup.

5) [0,4] No DNS compare o método de resolução iterativo e recursivo para otimizar o tempo de resposta dos servidores envolvidos a **futuros pedidos**:

A. Iterativo é mais eficaz.
B. Recursivo é mais eficaz.
C. Não tem influência sobre a resolução de nomes no futuro.
D. Iterativo e Recursivo são equivalentes neste aspeto.

B