TSR: Segundo parcial

Este examen comprende 10 preguntas de opción múltiple. En cada caso solamente una respuesta es correcta. Debe responderse en una hoja aparte. Las respuestas correctas aportan 1 punto a la calificación de esta prueba. Las incorrectas reducen la calificación 0.33 puntos.

1. Para desplegar adecuadamente un servicio replicado, debemos asegurar que:

A	Sus réplicas se ubican en diferentes ordenadores que no dependen de una misma fuente de fallos.
B	El código de sus réplicas se ha escrito en JavaScript.
C	Sus réplicas usan ZeroMQ como su middleware de comunicaciones.
D	El despliegue se realiza con las herramientas de Docker.

2. Considerando el acoplamiento en servicios distribuidos, deberíamos asegurar que los componentes muestren:

A	Acoplamiento débil, pues implica que los componentes son fiables.
B	Acoplamiento fuerte, pues implica que los componentes utilizan algoritmos descentralizados.
C	Acoplamiento fuerte, pues implica que los componentes están replicados.
D	Acoplamiento débil, pues implica que los mensajes entre componentes serán pequeños y que la sincronización entre componentes será muy relajada.

3. ¿Cuál de los siguientes modelos de servicio "cloud" se preocupa por la gestión de la elasticidad de los servicios desplegados por sus clientes?

A	SaaS.
B	IaaS.
C	PaaS.
D	Todos ellos.

4. ¿Cuál de las siguientes afirmaciones sobre seguridad en sistemas distribuidos es FALSA?

A	Los sistemas distribuidos son potencialmente vulnerables debido a su necesidad de comunicación entre nodos.
B	Los ataques de denegación de servicio consisten en "inundar" las instancias servidoras con más solicitudes de las que puedan gestionar.
C	Los sistemas distribuidos son potencialmente vulnerables porque no pueden proporcionar mecanismos de control físico de acceso a todos sus nodos y canales.
D	Los sistemas distribuidos tienen una "base informática de confianza" (TCB) fácilmente identificable.

5. Sea esta ejecución en un sistema con tres procesos (P1, P2 y P3):
W1(x)1, R2(x)1, W2(y)3, W1(y)2, R1(y)3, R3(y)2, R3(y)3, R3(x)1,…
…donde P2 no acepta el valor 2 de "y" escrito por P1 en la cuarta acción de la traza. El modelo de consistencia de memoria más fuerte soportado por esta ejecución es:

A	FIFO.
B	Secuencia.
C	Caché.
D	Causal.

6. Sea esta ejecución en un sistema con tres procesos (P1, P2 y P3):
W1(x)1, R2(x)1, W2(y)3, R1(y)3, W1(y)2, R2(y)2, R3(x)1, R3(y)3, R3(y)2,…
El modelo de consistencia de memoria más fuerte soportado por esta ejecución es:

A	FIFO.
B	Secuencial.
C	Caché.
D	Estricto.

7. El componente "mongod" en MongoDB es un…:

A	Proxy mantenido por los procesos clientes para interactuar con los servidores de configuración de MongoDB.
B	Equilibrador de carga que redirige las peticiones clientes al servidor "mongos" apropiado. Este gestiona localmente una instancia de la base de datos MongoDB.
C	Un módulo JavaScript que debe ser incluido en los programas NodeJS para acceder a la base MongoDB.
D	El servidor que localmente gestiona una instancia de la base de datos MongoDB.

8. Si somos desarrolladores, para reducir las posibles vulnerabilidades de los servicios distribuidos que escribamos... ¿cuál es la mejor de las siguientes alternativas?

A	Comprobar periódicamente que nuestras herramientas de desarrollo, las bibliotecas y middleware utilizados y los nodos donde se realicen los despliegues hayan recibido sus actualizaciones relacionadas con seguridad.
B	Desplegar nuestros servicios con docker-compose.
C	Asignar privilegios de administrador a todos los usuarios de nuestros servicios.
D	Confiar en la reputación del sistema operativo utilizado en los nodos anfitriones, comprobando que todos los nodos utilicen el mejor sistema en este aspecto.

9. Considerando este programa...

```
var cluster = require('cluster');
var http = require('http');
var numCPUs = require('os').cpus().length;
function processRequest(req,resp) {
  // Some code for processing an HTTP request and generate its response.
}
if (cluster.isMaster) {
  for (var i=0; i < numCPUs; i++) cluster.fork();
  cluster.on('exit', function(worker) {
    cluster.fork();
  });
} else {
  http.createServer(processRequest).listen(8000);
}
```

¿Qué afirmaciones son ciertas?

A	Si un proceso trabajador muere, el maestro genera otro.
B	Este programa crea tanto procesos trabajadores como núcleos de procesador haya en la máquina.
C	Los procesos trabajadores utilizan la función "processRequest" para gestionar cada petición HTTP recibida.
D	Todas las anteriores.

10. Supongamos que un programador ha escrito un componente broker en NodeJS que escucha en múltiples puertos: 8000 (mensajes clientes), 8001 (conexión para modificar la configuración del broker) y 8002 (mensajes hacia los trabajadores). Cuando los clientes y los trabajadores se ubiquen en otros nodos, se recomienda asignar su puerto 8000 al puerto 80 del anfitrión y su puerto 8002 al 82 del anfitrión. El programa del broker se llama Broker.js y ese fichero está en la misma carpeta que este Dockerfile:

```
FROM zmq-devel
COPY ./Broker.js /Broker.js
CMD node /Broker.js
```

Tras usar la orden "docker build –t myBroker ." en esa carpeta, se ha intentado ejecutar el broker, pero es incapaz de interactuar con algunos clientes y trabajadores locales. Para corregir ese problema, se debe:

A	Añadir esta línea en el Dockerfile: PORT 8000 8001 8002
B	Añadir estas líneas en el Dockerfile: PORT 80:8000 82:8002 PORT 8001
C	Añadir esta línea en el Dockerfile: EXPOSE 8000 8001 8002
D	No se necesita hacer nada en el Dockerfile. El problema está relacionado con los argumentos y opciones utilizados en la orden "docker run myBroker".