TSR: Segundo Parcial

Este examen consta de 20 cuestiones de opción múltiple. En cada una, solo una respuesta es correcta. Debe responderse en otra hoja. Las respuestas correctas aportan 0.5 puntos a la calificación del examen. Las erróneas descuentan 0.167 puntos.

TEORÍA

1. El despliegue incluye la instalación inicial y la configuración de una aplicación. Además de esas tareas, el despliegue de un servicio también comprende…

A	La depuración de los programas.
B	La gestión del ciclo de vida del servicio.
C	El desarrollo de la aplicación.
D	El diseño de la aplicación.

2. El objetivo principal de la inyección de dependencias es…

A	Resolver las dependencias entre componentes utilizando ficheros de configuración.
B	Eliminar todas las dependencias entre componentes durante la etapa de diseño.
C	Que la resolución de dependencias sea lo más transparente posible para el desarrollador de los componentes.
D	Evitar el uso de contenedores, pues estos penalizan el rendimiento.

3. Imaginemos un servicio que necesite 400 ms para procesar localmente cada petición que modifique su estado. Esas peticiones invierten 20 ms para transmitir a otras réplicas el estado modificado y 30 ms en aplicar esas modificaciones en ellas. Un mensaje de petición puede ser difundido (en orden total) en esa red en 3 ms. Una petición de lectura puede ser gestionada localmente en 20 ms. La proporción de accesos es: 80% accesos de lectura y 20% accesos de modificación.
Para escalar este servicio, la mejor aproximación será…

A	Replicarlo utilizando el modelo activo.
B	Replicarlo utilizando el modelo pasivo, procesando todas las solicitudes en la réplica primaria.
C	Replicarlo utilizando el modelo pasivo, pero permitiendo que las solicitudes de lectura sean procesadas por las réplicas secundarias.
D	No replicarlo, pues la replicación introduce demasiada coordinación y eso impide un escalado eficiente.

4. En los modelos de consistencia centrados en datos, podemos decir que el modelo A es más fuerte que el modelo B en los siguientes casos:

A	A: causal, B: caché.
B	A: FIFO, B: caché.
C	A: causal, B: secuencial.
D	A: causal, B: FIFO.

5. Los almacenes escalables NoSQL no soportan el modelo relacional porque…

A	El modelo relacional no admite replicación.
B	El modelo relacional no admite particionado horizontal ("*sharding*").
C	Los datos relacionales deben mantenerse en disco.
D	Las transacciones en el modelo relacional necesitan mecanismos de control de concurrencia que pueden ser complejos.

6. El teorema CAP…

A	…exige que los servicios escalables y disponibles utilicen siempre el modelo de consistencia estricto para tolerar así las particiones de la red.
B	...permite que los servicios escalables relajen su consistencia mientras la red permanezca particionada, asegurando así su disponibilidad.
C	...no permite la implantación de servicios altamente disponibles utilizando modelos de consistencia fuertes.
D	...no tiene sentido en los centros de datos de computación en la nube, pues jamás habrá particiones de la red en ellos.

7. Respecto a la escalabilidad de servicios se puede afirmar que…

A	Un mismo servicio no puede escalar horizontal y verticalmente.
B	Los algoritmos descentralizados mejoran la escalabilidad de distancia.
C	El particionado horizontal ("*sharding*") mejora la escalabilidad administrativa.
D	Evitar la contención es un factor clave para mejorar la escalabilidad de un servicio.

8. Los objetivos principales de un subsistema de seguridad son:

A	Protección, control de acceso y seguridad física.
B	Protección, gestión de la confianza y un buen soporte para mecanismos de cifrado.
C	Contabilidad, integridad, confidencialidad y disponibilidad.
D	Políticas robustas, mecanismos eficientes y garantías correctas.

SEMINARIOS

9. ¿Cuál de las siguientes tareas NO se realiza al utilizar esta orden Docker?

```
docker run –it ubuntu /bin/bash
```

A	Ejecutar el programa /bin/bash en un contenedor.
B	Descargar la imagen "ubuntu:latest" desde Docker Hub si no la teníamos en el depósito de imágenes local.
C	Recoger la salida del contenedor que está siendo utilizado para ejecutar esa orden. Esa salida puede mostrarse mediante la orden `docker logs`.
D	Eliminar automáticamente este contenedor una vez su ejecución haya terminado.

10. La orden `docker commit a b` …

A	Crea un nuevo contenedor llamado "a" utilizando el Dockerfile ubicado en la carpeta "b".
B	Crea una nueva imagen "b" utilizando el Dockerfile de la carpeta "a".
C	Crea una nueva imagen "b" con el contenido actual del contenedor cuyo nombre o identificador es "a".
D	Realiza el "commit" de una transacción "a" que fue iniciada con una orden `docker pull` o `docker push`, generando un contenedor con ID "b".

11. El módulo "cluster" de NodeJS se utiliza para…

A	…desplegar un conjunto de programas NodeJS en un "*cluster*" de ordenadores.
B	…gestionar múltiples hilos en un proceso NodeJS.
C	…gestionar un conjunto de procesos NodeJS para que puedan compartir algunos recursos; p.ej., un puerto en el que escuchar y un mismo programa a ejecutar.
D	…implantar fácilmente múltiples modelos de consistencia de memoria.

12. MongoDB utiliza los siguientes mecanismos para mejorar su escalabilidad:

A	Control de concurrencia distribuido.
B	Algoritmos descentralizados.
C	Replicación pasiva y particionado horizontal (o "*sharding*").
D	Escalabilidad administrativa.

13. El objetivo principal de los servidores de configuración en MongoDB es:

A	Adoptar el papel de árbitros cuando una réplica falle.
B	Controlar la distribución de datos entre las múltiples particiones horizontales ("shards") existentes.
C	Respetar el teorema CAP cuando se dé una partición en la red.
D	Detectar fallos en los nodos, iniciando un protocolo de recuperación cuando se dé algún fallo.

14. Considerando la clasificación temática de vulnerabilidades vista en el Seminario 8, se puede considerar cierta la siguiente afirmación:

A	La explotación de defectos en políticas de seguridad no puede automatizarse tan fácilmente como la explotación de contraseñas débiles.
B	El "*phishing*" es una vulnerabilidad de tipo "error software".
C	La protección física es un ejemplo de vulnerabilidad de ingeniería social.
D	Un defecto en una política de seguridad de protección personal requiere menor interacción para ser explotada que un error software en el sistema operativo.

15. Asumiendo este Dockerfile…

```
FROM zmq
RUN mkdir /zmq
COPY ./broker.js /zmq/broker.js
WORKDIR /zmq
EXPOSE 8000 8001
CMD node broker.js
```

¿Cuál de las siguientes afirmaciones es FALSA?

A	Necesitamos tener el fichero "broker.js" en el directorio del anfitrión en el que se encuentre este Dockerfile.
B	El programa a ejecutar en estos contenedores utiliza el puerto 8000 del contenedor y lo asocia al puerto 8001 del anfitrión.
C	Por omisión, los contenedores generados a partir de este Dockerfile ejecutarán la orden "**node broker.js**".
D	Este Dockerfile asume la existencia de una imagen llamada "zmq" con una instalación válida del intérprete de JavaScript "node".

16. Imaginemos que el componente broker de la cuestión 15 se incluye en un fichero docker-compose.yml con estos contenidos (entre otros que correspondan a otros componentes):

```
version: '2'
services:
...
 bro:
  image: broker
  build: ../broker/
```

¿Cuál de las siguientes afirmaciones es FALSA?

A	Podemos iniciar al menos una instancia del componente **broker** con la orden **docker-compose up –d**
B	Una vez se ha iniciado el servicio, podemos tener 5 instancias del componente **broker** utilizando **docker-compose scale broker=5**
C	Este fichero "docker-compose.yml" asume que el Dockerfile mostrado en la cuestión 15 se encuentra en la carpeta "../broker/".
D	Podemos utilizar la orden **docker-compose stop bro** para parar todas las instancias de este componente **broker**.

17. Supongamos un protocolo de replicación basado en un proceso secuenciador que utiliza un socket PUB ZeroMQ para propagar todos los eventos "write" a los procesos participantes, en orden de recepción. Esos eventos han sido recibidos mediante canales PUSH-PULL, cuyo socket PULL está en el secuenciador. Ese protocolo de replicación soporta los siguientes modelos de consistencia:

A	Solo el modelo estricto.
B	Solo el modelo caché.
C	Solo el modelo causal.
D	Secuencial, procesador, causal, caché y FIFO.

18. Dada la siguiente ejecución:
W1(x)1, R4(x)1, W2(y)2, W1(y)3, W2(x)4, R3(y)2, W3(x)5, R1(x)5, R2(x)1, R3(x)1, R4(y)3, R1(y)2, R3(y)3, R4(x)5, R3(x)4, R2(y)3, R4(y)2, R1(x)4, R2(x)5, R4(x)4.
Esa ejecución soporta estos modelos de consistencia:

A	Solo el modelo FIFO.
B	Solo el modelo caché.
C	Procesador, FIFO y caché.
D	Secuencial, procesador, causal, caché y FIFO.

19. **Dado el siguiente programa servidor de descarga de ficheros…**

```
var cluster = require('cluster');
var fs = require('fs');
var path = require('path');
var zmq = require('zmq');
var os = require('os');
const ipcName = 'Act2.ipc';
const dlName = 'ipc://'+ipcName;
if (cluster.isMaster) {
   var numCPUs = os.cpus().length;
   var rt = zmq.socket('router');
   var dl = zmq.socket('dealer');
   rt.bindSync('tcp://127.0.0.1:8000');
   dl.bindSync(dlName);
   rt.on('message', function() {
         msg = Array.apply(null, arguments);
         dl.send(msg);   });
   dl.on('message', function() {
         msg = Array.apply(null, arguments);
         rt.send(msg);   });
} else {
  var rep = zmq.socket('rep');
  rep.connect(dlName);
  rep.on('message', function(data) {
    var request = JSON.parse(data);
    fs.readFile(request.path, function(err,
data) {
     if (err) data = ' NOT FOUND';
      rep.send(JSON.stringify({
          pid: process.pid ,
          path: request.path ,
          data: data ,
          timestamp: new Date().toString()
      }))
   })
  })
}
```

Hemos tratado de ejecutar el programa, pero no parece hacer nada útil. Su principal problema es…

A	Los sockets ZeroMQ no admiten "**ipc://**" como transporte.
B	No se ha creado ningún proceso trabajador del módulo "cluster".
C	Se está tratando de propagar mensajes internos del módulo "cluster" a través de un socket DEALER ZeroMQ.
D	Un servidor no puede utilizar un socket ROUTER como su "endpoint".

20. **La vulnerabilidad OpenSSL Heartbleed descrita en el Seminario 8 es un ejemplo de vulnerabilidad que pertenece a las clases siguientes:**

A	"Defecto en la política de trabajo" en su categoría y "Personal humano" en su origen.
B	"Defecto en la lógica del software" en su categoría y "Biblioteca/middleware" en su origen.
C	"Ingeniería social" en su categoría y "Desarrollador" en su origen.
D	"Defecto en la lógica del software" en su categoría y "Personal humano" en su origen.