A

Aquest examen conté 20 qüestions d'opció múltiple. En cadascuna d'elles només una de les respostes és correcta. Les contestacions han de presentar-se en una fulla entregada a part. Les respostes correctes aporten 0.5 punts a la nota del parcial mentre que las incorrectes resten 0.167 punts. En la fulla de respostes has d'emplenar la casella triada acuradament. Utilitza un llapis o un bolígraf (negre o blau fosc).

TEORIA

1. En el pla de desplegament d'una aplicació distribuïda…

a	La plantilla del pla estableix com connectar els components, llistant tant les dependències a resoldre com els endpoints exposats. *Vertader. Entre altres coses la plantilla del pla ha d'establir els aspectes esmentats.*
b	Les plantilles de configuració de cada component contindran els mateixos valors per a totes les instàncies del component. *Fals. Una plantilla de configuració no ha de declarar els valors a utilitzar en els paràmetres a considerar per a desplegar les instàncies d'un component. La plantilla solament ha de contenir el conjunt de paràmetres a considerar i l'administrador (o desplegador) emplenarà la plantilla assignant els valors adequats. Normalment, instàncies diferents tindran valors diferents, almenys per a alguns dels paràmetres.*
c	Algunes instàncies de components poden tenir dependències que no siguen resoltes en el desplegament, sinó després, durant la prestació del servei. *Fals. Normalment les dependències de les instàncies dels components seran resoltes durant el desplegament. Seria perillós postergar la seua resolució fins a quan la instància ja estiga participant en l'execució del servei ja que, si hi haguera algun problema en fer-ho, moltes peticions en curs podrien perdre's i alguns components del servei podrien no iniciar-se o no funcionar correctament.*
d	La plantilla del pla estableix la ubicació (en nodes) de totes les instàncies de tots els components, respectant la restricció que, en cada node, s'execute una i només una instància de cada component. *Fals. La plantilla no ha de restringir-se d'aquesta manera. No té cap sentit que no es puga situar més d'un component en un mateix node.*

A

2. En els models de fallades en un sistema distribuït, és correcte dir que…

a	Tant en el model de fallades de ***parada*** com en el model de fallades de ***caiguda***, la fallada d'un procés sempre és detectada pels altres processos. *Fals. Les fallades dels processos solament poden ser detectades amb seguretat per altres processos en el model de fallades de parada. Aquesta característica no està contemplada en el model de caiguda. Observe's que el model de parada sol assumir un major nivell de sincronia que el model de caiguda, precisament perquè la detecció de fallades siga possible.*
b	És fàcil implantar un sistema distribuït on les úniques fallades possibles siguen les pròpies del model de fallades de ***parada*** o les del model de fallades de ***caiguda***. *Fals. De fet és molt difícil que un sistema real assegure que els seus processos es comportaran perfectament fins parar o caure. Per a fer això necessitaríem programes perfectes (sense cap error i amb una configuració que es corresponguera a la perfecció amb el seu entorn real en tot moment) per a tots els processos en execució al sistema, al costat d'un conjunt perfecte de components per a construir el propi sistema (és a dir: sistemes operatius de cada màquina, sistemes middleware, etc.). És impossible aconseguir aquest nivell de qualitat en totes les tasques de desenvolupament i desplegament.*
c	El model de fallades de ***partició de la xarxa*** és equivalent al model de fallades d'omissió ***general*** (és a dir, omissió tant d'enviaments com de recepcions). *Fals. Una partició de la xarxa no permet que els processos situats en diferents subgrups de la xarxa intercanvien missatges entre si, però sí que admet que aquells processos situats en un mateix subgrup es comuniquen sense problemes. D'altra banda, el model d'omissió general assumeix que els processos tindran problemes quan intenten enviar o rebre missatges independentment de quins siguen, respectivament, els receptors o emissors d'aquests missatges. Com pot veure's, tots dos conceptes no són equivalents.*
d	Si es proporciona transparència a les fallades pròpies del model de ***fallades bizantines***, llavors també es proporciona transparència a les fallades pròpies de tots els altres models. *Vertader. Si es pot garantir transparència de fallades assumint un model de fallades bizantines això implica que qualsevol fallada podrà ser ocultada pels nostres programes. Observe's que el model bizantí inclou a tots els altres models. Per aquesta raó, les situacions de fallada assumides en tots els altres models ja estan considerades en el model bizantí i segur que no seran observables quan es proporcione transparència de fallades per al model de fallades bizantines.*

3. Sobre els models de replicació, és correcte dir que…

a	El model de replicació passiva suporta el model de fallades bizantines, però no suporta el model de fallades de caiguda i enllaç. *Fals. Per a suportar el model de fallades bizantines es necessita saber quantes fallades arbitràries podran ocórrer simultàniament. Assumim que va a ser "f" aquest valor. Així, necessitarem 2f+1 rèpliques actives per a tolerar fallades arbitràries en el millor dels casos i 3f+1 rèpliques actives quan s'utilitzen missatges per als quals no es puga autenticar el seu emissor. En aquest context considerem que una rèplica és activa quan pot interactuar directament amb els clients del servei. En el model de replicació passiva només existeix una rèplica*

	activa per cada entitat replicada: la seua rèplica primària. A causa d'això, el model de replicació passiva no podrà suportar el model de fallades bizantines. *Un altre argument per a considerar que aquesta afirmació és falsa consisteix a observar que el model de fallades de caiguda i enllaç és menys sever (és a dir, més fàcil de gestionar) que el model de fallades bizantines. Per això, si cert programa pot suportar les fallades bizantines també podrà suportar amb total seguretat el model de fallades de caiguda i enllaç.*
b	El model de replicació passiva planteja serioses dificultats per a la transmissió de les actualitzacions d'estat satisfent un ordre total. *Fals. La propagació d'actualitzacions en ordre total és fàcilment implantable en aquest model. Com només hi haurà un únic propagador (la rèplica primària), aquest només ha de numerar (i, així, seqüenciar) els missatges d'actualització que propague. Per a fer això només necessita un comptador local.*
c	El model de replicació activa planteja serioses dificultats en cas de reconfiguració després de la fallada d'alguna de les rèpliques. *Fals. Com totes les rèpliques són idèntiques en aquest model, normalment no es necessitarà cap reconfiguració quan alguna d'elles falle.*
d	En el model de replicació activa es necessiten més rèpliques per a donar suport al model de fallades bizantines que per a donar suport al model de fallades de caiguda. *Vertader. En el model de replicació activa tots dos tipus de fallades poden suportar-se. Per a suportar "f" fallades simultànies de caiguda solament es necessiten f+1 rèpliques. Per a suportar "f" fallades bizantines simultánies es necessitaran 2f+1 o 3f+1 rèpliques, depenent de si es pot autenticar o no a l'emissor de cada missatge (per a evitar així suplantacions d'identitat).*

4. Els magatzems de dades NoSQL milloren l'escalabilitat perquè…

a	Garanteixen la integritat referencial. *Fals. La majoria dels magatzems de dades NoSQL no garanteixen integritat referencial.*
b	Redueixen la redundància de la informació emmagatzemada. *Fals. Les tècniques de normalització van ser introduïdes en les bases de dades relacionals per a reduir la redundància de dades. No obstant això, els magatzems NoSQL no utilitzen aquestes tècniques i no es preocupen per tenir una mateixa dada registrada múltiples vegades en una mateixa base.*
c	Eliminen o simplifiquen les transaccions. *Vertader. Les transaccions que garanteixen les propietats ACID (utilitzades en les bases de dades relacionals) utilitzen implícitament alguns mecanismes de control de concurrència que poden bloquejar a les transaccions en curs. Qualsevol tipus de sincronització de tasques posa en perill l'escalabilitat d'un servei. Per això molts magatzems NoSQL han eliminat la gestió de transaccions. A causa d'això, aquests gestors renuncien a l'aïllament i atomicitat transaccionals en favor del rendiment i l'escalabilitat.*
d	Garanteixen que no hi haurà mai particions en la xarxa. *Fals. Les particions en la xarxa ocorren quan hi ha problemes de configuració o funcionament en els equips de comunicació (routers, switches, etc.). Els magatzems NoSQL no tenen cap efecte directe sobre aquests aspectes.*

5. Segons el teorema CAP, quan ocórrega una partició de la xarxa...

A

a	...tots els subgrups continuaran contestant les peticions dels clients i els serveis replicats mantindran com a mínim una consistència seqüencial. *Fals. Quan es dóna una partició de la xarxa els processos que pertanyen a diferents subgrups no poden intercanviar missatges entre si. En aquest cas, si tots els subgrups continuaren, les modificacions aplicades en un subgrup no podrien ser propagades als altres. Així els processos situats en diferents subgrups no podrien mantenir ni veure la mateixa seqüència d'escriptures sobre les seues variables compartides. Com a resultat d'això, la consistència mantinguda en el sistema no podria ser seqüencial.*
b	...s'assumirà un model particionable i els serveis replicats mantindran una consistència final. *Vertader. Si s'assumeix un model particionable els processos situats en cada subgrup podran continuar amb la seua execució. Així no podran mantenir una consistència forta però la consistència resultant podrà respectar les condicions de la consistència final. Això implica que quan la connectivitat es recupere les rèpliques podran intercanviar les modificacions que hagen aplicat i totes elles convergiran cap a un mateix estat (és a dir, cadascuna d'elles tindrà el mateix valor en cada variable).*
c	...s'assumirà un model de partició primària i totes les rèpliques del servei continuaran servint als seus clients. *Fals. Hi ha una contradicció en aquesta oració. Si s'assumeix el model de partició primària llavors solament el subgrup majoritari (en cas que n'hi haja algun) podrà continuar. Aleshores no pot ser cert que "totes les rèpliques del servei continuen servint als seus clients" posat que aquelles que residisquen en subgrups minoritaris no respectaran aquesta condició (és a dir, hauran de parar).*
d	...pararem totes les rèpliques del servei fins que es repare la partició. No es permetrà cap activitat del servei durant aquest interval. *Fals. Tal com s'ha explicat en l'apartat "b", res obliga a que les rèpliques del servei romanguen parades mentre dure la partició.*

6. La contenció, o dificultat per a l'escalabilitat, pot deure's a…

a	L'ús d'algorismes descentralitzats per a atendre les tasques pesades. *Fals. Si una tasca perllongada es gestiona mitjançant un algorisme descentralitzat serà dividida en múltiples subtasques i cada subtasca serà executada per un agent diferent. En aquest cas, la tasca global serà acabada prompte. Per tant, aquesta aproximació no introdueix contencions.*
b	Una mala distribució dels recursos que cause un major tràfic. *Vertader. Si els recursos necessaris per a gestionar alguna activitat s'han distribuït d'una manera inadequada, els agents que participen en l'activitat necessitaran molts més missatges per a sol·licitar i alliberar els recursos. Aquests missatges introduiran una sincronització addicional entre els agents. Com a conseqüència, els agents participants arribaran a bloquejar-se durant intervals perllongats esperant l'arribada d'alguns missatges i això comportarà certa contenció, reduint així l'escalabilitat del sistema.*
c	L'ús d'un middleware de comunicacions asincrònic. *Fals. La comunicació entre agents ha de ser el més asincrònica possible per a minimitzar la durada dels intervals de bloqueig que arriben a donar-se.*
d	La replicació dels components responsables del repartiment de càrrega. *Fals. Si un component es replica amb cura podrà incrementar el seu rendiment i reduir els seus intervals de bloqueig.*

7. Principis generals per a millorar l'escalabilitat:

a	Incrementar sense límit el grau de concurrència. *Fals. Un increment en el grau de concurrència no sempre millorarà l'escalabilitat d'un servei. Si aquestes activitats concurrents accedeixen a algun recurs compartit necessitaran alguns mecanismes de control de concurrència i això pot conduir a bloquejar-les temporalment. En aquest cas, un increment en el grau de concurrència podria reduir l'escalabilitat del servei.*
b	Mantenir una consistència forta. *Fals. Com més fort siga la consistència més perllongats seran els intervals de suspensió que originarà el protocol de consistència que implante aquest model. Per tant, per a millorar l'escalabilitat convé utilitzar models de consistència relaxats en lloc de models de consistència forts.*
c	Evitar, tant com siga possible, la sincronització entre agents. *Vertader. La condició principal per a millorar l'escalabilitat consisteix a minimitzar la sincronització entre agents. Així, els agents invertiran la major part del seu temps en l'execució de les tasques que se'ls haja assignat en lloc de romandre suspesos en algun pas de sincronització.*
d	Mantenir totes les dades en emmagatzematge secundari per a assegurar la seua persistència. *Fals. Guardar les dades en emmagatzematge secundari introdueix llargs intervals d'E/S. Els agents que soliciten aquestes operacions romandran suspesos durant aquests intervals. Per tant, això no millora l'escalabilitat.*

8. Sergi és un expert en seguretat que treballa en l'empresa B. Va utilitzar ahir un rastrejador de paquets (o "*packet sniffer*") i va obtenir l'identificador i la contrasenya d'un administrador de sistemes de l'empresa A. També va esbrinar l'adreça pública d'un dels servidors d'A des d'on els seus administradors poden accedir a altres servidors de l'empresa. Per a l'empresa A, l'escenari actual és un exemple de...

a	...un atac de denegació de servei. *Fals. De moment la capacitat de servei de la companyia A no s'ha vist compromesa i no s'ha rebut cap atac d'aquest tipus.*
b	...una amenaça externa. *Vertader. És una amenaça ja que alguns dels mecanismes de seguretat utilitzats en A són ara coneguts per membres d'una altra empresa però aquest coneixement no s'ha traduït en cap acció que puga haver danyat a A. A més, és una amenaça externa perquè Sergi no pertany a A.*
c	...una feblesa dels seus protocols d'encaminament. *Fals. Els protocols d'encaminament no tenen cap implicació sobre com treballa un rastrejador de paquets. Per a evitar les amenaces introduïdes pels rastrejadors de paquets hauríem d'utilitzar comunicació xifrada i aquesta comunicació no depèn dels protocols d'encaminament. Observe's que la comunicació xifrada pot implantar-se en el nivell d'aplicació mentre les tasques d'encaminament són gestionades pel nivell de xarxa.*
d	...un mecanisme de seguretat física. *Fals. El que s'ha descrit en aquesta qüestió és una **amenaça** sobre la seguretat d'un sistema en lloc d'un **mecanisme** de seguretat.*

A

SEMINARIS

9. Donada la següent seqüència d'ordres Docker executades des de la CLI:

```
docker pull fedora
docker run --name fedora fedora dnf install –y nodejs
docker commit fedora node
docker push node
```

Quina de les següents accions NO s'ha fet?

a	Descarregar des del dipòsit públic (Docker Hub) una imatge, *fedora*. *Fals. Això es fa mitjançant l'ordre "docker pull fedora".*
b	Pujar al dipòsit públic (Docker Hub) una imatge, *node*. *Fals. Això es fa mitjançant l'ordre "docker push node".*
c	Crear un contenidor i executar en ell l'ordre *node*. *Vertader. El contenidor ha sigut creat mitjançant l'ordre "docker run" però en aquesta línia l'usuari no ha sol·licitat l'execució de l'intèrpret "node" al contenidor.*
d	Crear un contenidor, modificar-lo, i crear una imatge a partir d'aquest. *Fals. El contenidor ha sigut creat en la línia que comença amb "docker run". En la mateixa línia hem modificat el seu contingut amb l'ordre "dnf install –y nodejs" del sistema Fedora. Posteriorment la línia del "docker commit" crea una nova imatge amb aquest contenidor modificat.*

10. Donat el següent contingut d'un Dockerfile:

```
FROM zmq
RUN mkdir /zmq
COPY ./worker.js /zmq/worker.js
WORKDIR /zmq
CMD node worker $BROKER_PORT_8001_TCP
```

Si es genera una imatge a partir d'ell mitjançant l'ordre:

```
docker build –t worker .
```

Quina de les següents afirmacions és FALSA?

a	La imatge *worker* és una modificació de la imatge *zmq*. *Vertader. L'ordre "docker build" crea una imatge "worker" usant el Dockerfile presentat en aquesta qüestió. La seua primera línia indica que la nova imatge es construirà a partir d'una imatge "zmq" ja existent. A més, les instruccions RUN i COPY han estès i modificat la imatge "zmq" original.*
b	El directori de treball per a la instrucció CMD és */zmq*. *Vertader. Aquest és el resultat de la línia WORKDIR en aquest Dockerfile.*
c	Si es crea un contenidor a partir de la imatge *worker*, s'executarà el programa indicat en la instrucció CMD. *Vertader. Aquest és l'efecte de les instruccions CMD en els Dockerfile.*
d	Si es crea un contenidor a partir de la imatge *worker*, no s'executarà cap programa ja que ha d'indicar-se amb la instrucció ENTRYPOINT en lloc de CMD. *Fals. Tant ENTRYPOINT com CMD poden utilitzar-se per a especificar el programa a executar en el contenidor.*

A

11. Assumisca que s'ha instal·lat Docker en el nostre ordinador on hem creat una imatge "node2" on podrem utilitzar "node" des de la línia d'ordres. Imagine que volem executar en un contenidor Docker el programa "/tmp/exemple.js" que tenim en el nostre ordinador. Per a fer això, entre altres accions, haurem de...

a	Usar **docker run node2** des de la línia d'ordres, passant la ruta del programa com el seu últim argument; és a dir, **docker run node2 /tmp/exemple.js** *Fals. En l'ordre "docker run" el seu primer argument es refereix al nom de la imatge i el segon, quan s'utilitze, es referirà a l'ordre a executar en el contenidor. El segon argument no pot ser el nom de ruta d'un fitxer mantingut en l'amfitrió. En comptes d'això, ha de ser algun element que ja estiga al contenidor.*
b	No podrem fer res. Els fitxers del nostre ordinador no poden ser utilitzats des del contenidor i no hi ha manera de copiar-los en una imatge. *Fals. Hi ha diverses maneres de transferir recursos de l'amfitrió a la imatge utilitzada pel contenidor.*
c	Copiar el fitxer en una nova imatge basada en **node2**. Per a fer això utilitzarem la instrucció COPY en un Dockerfile. *Vertader. Aquest ha sigut el mecanisme explicat en els exemples mostrats al Seminari 4.*
d	Usar **docker cp /tmp/exemple.js node2**. *Fals. Existeix una ordre "docker cp" però utilitza una sintaxi diferent per a copiar fitxers en una imatge. La sintaxi correcta és:* **docker cp /tmp/exemple.js node2:** *Opcionalment, després dels dos punts que segueixen al nom de la imatge es pot especificar un nom de ruta. El caràcter dos punts és obligatori. Mitjançant ell s'indica quin dels dos arguments es refereix a la imatge. L'altre serà un nom de ruta vàlid en l'amfitrió.*

12. Considerant aquest Dockerfile...

```
FROM fedora
RUN dnf install -y nodejs
RUN dnf install -y zeromq-devel
RUN dnf install -y npm
RUN dnf install -y make
RUN npm install zmq
```

...es pot afirmar que:

a	Aquest Dockerfile no té sentit perquè no inclou cap instrucció CMD o ENTRYPOINT. No fa res en absolut. *Fals. Un Dockerfile com aquest té sentit. Amb ell s'estén la imatge "fedora" del dipòsit públic instal·lant paquets addicionals en la imatge resultant. Les instruccions ENTRYPOINT i CMD no cal que s'utilitzen en tots els fitxers Dockerfile.*

b	Aquest Dockerfile és incorrecte perquè falla en la seua segona línia. No existeix cap instrucció "dnf" en Docker. *Fals. L'ordre "dnf" no és una instrucció de Docker sinó una ordre dels sistemes Fedora. Pot ser utilitzada sense problemes en els arguments de la instrucció RUN d'un Dockerfile quan la imatge presa com a base utilitze un sistema operatiu Fedora.*
c	El nom de la imatge creada amb aquest Dockerfile és "zmq". *Fals. El nom a assignar a la imatge resultant ha de ser especificat utilitzant l'ordre "docker build".*
d	Aquest Dockerfile crea una nova imatge basada en la imatge "fedora" del Docker Hub. La nova imatge afig almenys 4 paquets Fedora a la imatge base. *Vertader.*

13. En la implementació d'un model de consistència feble (activitat 1 del seminari 5, fitxers de codi: shared1.js, proc1.js), mitjançant un patró de comunicació PUB – SUB, es garanteix una consistència…

a	…seqüencial, perquè les escriptures d'un procés es comuniquen immediatament a la resta de processos mitjançant una difusió usant el socket PUB. *Fals. Com hi haurà diversos processos escriptors i cadascun d'ells utilitzarà un socket PUB diferent, els valors que ells escriguen podran arribar en un ordre diferent a cadascun dels altres processos. Així, els processos del sistema no veuran un mateix ordre de valors en les seues variables i la consistència resultant no serà seqüencial.*
b	… causal, perquè qualsevol procés efectua lectures dels valors escrits pels altres processos abans d'iniciar l'escriptura de la variable compartida. *Fals. Com els missatges necessiten algun temps per a ser propagats no es podrà garantir que un escriptor puga llegir els valors escrits prèviament per altres processos abans que la propagació del valor que ell escriga siga iniciada. Per tant, allò que s'esmenta en aquest apartat no pot garantir-se.*
c	… *cache*, atès que la subscripció a les escriptures de cada variable garanteix que tots lligen la mateixa seqüència de valors per a cada variable. *Fals. Com cada procés utilitza un socket PUB diferent i cadascun d'ells pot escriure valors sobre qualsevol variable, no hi ha cap garantia que els valors escrits sobre una mateixa variable arriben en el mateix ordre a tots els processos. Per tant, la consistència "cache" no pot garantir-se.*
d	… FIFO, atès que cada procés difon les seues escriptures en ordre, mitjançant el seu socket PUB, i atès que el protocol usat, TCP, respecta l'ordre FIFO. *Vertader. Els sockets PUB solen respectar l'ordre FIFO. Com cada procés té el seu propi socket PUB, els valors que escriga cadascun seran rebuts pels altres processos en l'ordre en què van ser escrits. Això respecta les condicions de la consistència FIFO.*

14. Considerant la implantació d'un protocol de replicació basat en un procés seqüenciador, com el descrit en el Seminari 5...

a	Quan executem el procés seqüenciador en el node més ràpid del nostre sistema, el model de consistència resultant serà ràpid. *Fals. Un model de consistència es considera ràpid quan existeix algun protocol que ho implanta en el qual no es necessita cap propagació de missatge per a considerar que una lectura o escriptura local ha sigut finalitzada. La utilització d'un procés seqüenciador implica que l'escriptor ha d'enviar un missatge al*

	seqüenciador abans de completar la seua escriptura. Per això, cap model de consistència que necessite un seqüenciador serà ràpid.
b	Si usem diferents processos seqüenciadors per a cada variable, es mantindrà una consistència *cache* sense assegurar consistència seqüencial. *Vertader. Amb un seqüenciador diferent per a cada variable podrem assegurar que tots els processos estiguen d'acord en l'ordre de valors per variable. Aquesta és la condició a complir en el model de consistència "cache". El model seqüencial, per la seua banda, requereix acord sobre una seqüència comuna en tots els processos sobre totes les variables. Això no pot assegurar-se utilitzant més d'un seqüenciador.*
c	Si usem el mateix procés seqüenciador per a totes les variables, mantindrem consistència seqüencial sense garantir consistència *cache*. *Fals. Com el model de consistència seqüencial és més estricte que el model "cache", quan un protocol implante la consistència seqüencial estarà necessàriament implantant també la consistència "cache".*
d	Si usem el mateix procés seqüenciador per a totes les variables, mantindrem consistència seqüencial sense garantir consistència FIFO. *Fals. Com el model de consistència seqüencial és més estricte que el model FIFO, quan un protocol implante la consistència seqüencial estarà necessàriament implantant també la consistència FIFO.*

15. Considerant aquest programa...

```
var cluster = require('cluster');
var http = require('http');
var numCPUs = require('os').cpus().length;
if (cluster.isMaster) {
  for (var i=0; i < numCPUs; i++) cluster.fork();
  cluster.on('exit', function(who, code, signal) {
    console.log('Process '+ who.process.pid +' died');
  });
} else {
  http.createServer(function(req, res) {
    res.writeHead(200);
    res.end('hello world\n');
  }).listen(8000);
}
```

...es pot afirmar que:

a	El programa falla quan es crea el segon treballador ja que tots els treballadors tracten s'usar el mateix port (8000) i aquest ja està en ús. *Fals. Quan usem el mòdul "cluster" per a crear processos treballadors utilitzant la seua funció cluster.fork(), tots els treballadors poden compartir un mateix conjunt de ports. No generaran cap error del tipus "port ja en ús".*

b	El procés *master* crea tant processos treballadors com a processadors (o nuclis) hi haja en l'ordinador local. *Vertader. Aquest és l'objectiu del bucle "for" mostrat en aquest exemple.*
c	El mòdul "cluster" permet desplegar cada treballador generat en un ordinador diferent. *Fals. El mòdul "cluster" només pot generar processos treballadors en l'ordinador local.*
d	El programa mostra un missatge quan el procés *master* finalitza. *Fals. El programa mostra un missatge cada vegada que un procés treballador finalitze però no imprimeix res quan finalitza el procés "master".*

16. Sobre el programa de la qüestió anterior...

a	El primer treballador escriu un missatge en pantalla cada vegada que rep un missatge. *Fals. No s'escriu cap missatge en rebre una petició HTTP. En lloc d'això, s'envia al client una resposta HTTP.*
b	La resposta dels treballadors depèn del contingut de la petició HTTP enviada pel client. *Fals. La resposta HTTP retornada al client és sempre la mateixa ("hello world"), independentment de la petició que s'haja rebut.*
c	La comunicació entre el procés *master* i els processos treballadors utilitza un patró REQ/REP. *Fals. En aquest exemple no hi ha cap comunicació explícita entre el procés master i els processos treballadors.*
d	Cada treballador és un procés servidor HTTP. *Vertader. Qui haja desenvolupat aquest exemple ha utilitzat la funció http.createServer() per a escriure el codi dels processos treballadors. Per això, aquests processos són servidors HTTP.*

17. Per a millorar la seua escalabilitat, MongoDB utilitza...

a	...el model de replicació actiu. *Fals. Utilitza una variant de la replicació passiva en la qual les rèpliques secundàries poden atendre també peticions de només lectura i on pot haver-hi processos "virtuals" que participen en les votacions per a triar una nova rèplica primària quan el primari anterior haja fallat.*
b	...un model particionable per a gestionar les particions de la xarxa. *Fals. MongoDB utilitza un model de partició primària per a gestionar les situacions de partició en la xarxa de comunicacions.*
c	...repartiment de la base de dades (és a dir, "*sharding*"), combinat amb replicació passiva. *Vertader. Ja s'ha esmentat l'ús de replicació passiva en el primer apartat. Addicionalment, quan es gestione una base de dades gran, MongoDB pot distribuir el seu contingut entre múltiples servidors mongod utilitzant repartiment (o "sharding").*
d	...transaccions que respecten les quatre propietats ACID. *Fals. MongoDB no utilitza transaccions atòmiques que incloguen múltiples sentències d'accés a la base de dades.*

A

18. Quan es reparteix una base de dades MongoDB, tots els seus subconjunts (“*shards*”) han de tenir una grandària similar. Això s'aconsegueix usant...

a	...“*journaling*”. *Fals. MongoDB utilitza “journaling” per a una altra finalitat. El “journaling” es necessita per a assegurar que les modificacions (escriptures, esborrats, insercions...) en la base de dades siguen persistents. Per a fer això, l'accés s'anota primer en un fitxer de “log” i una vegada es completa aquesta escriptura s'escriu també en la taula o col·lecció indicada per la sentència a executar. Si el procés servidor fallara durant el servei d'aquesta modificació, la modificació podria completar-se satisfactòriament en iniciar de nou el servidor si l'anotació en el log va arribar a realitzar-se.*
b	...un algorisme de migració de fragments. *Vertader. Aquest algorisme s'utilitza per a migrar els fragments des dels subconjunts més grans cap als altres.*
c	...normalització. *Fals. MongoDB no utilitza normalització. La normalització sol utilitzar-se en les bases de dades relacionals.*
d	...un algorisme d'exclusió mútua. *Fals. Els algorismes d'exclusió mútua es necessiten en la programació concurrent per a evitar condicions de “carrera”. No guarden cap relació directa amb el repartiment de bases de dades ni amb l'ajust de la grandària dels seus subconjunts.*

19. Segons la classificació temàtica de vulnerabilitats vista en el Seminari 8...

a	Els atacs de “phishing” exploten vulnerabilitats relacionades amb la “enginyeria social”. *Vertader. Un atac de “phishing” consisteix a adoptar la identitat d'un altre servei per a obtenir l'identificador i la contrasenya (o qualsevol altra informació secreta) dels usuaris del servei. Explota vulnerabilitats d'enginyeria social.*
b	Les vulnerabilitats en el disseny de protocols pertanyen a la classe “enginyeria social”. *Fals. Les vulnerabilitats en el disseny de protocols pertanyen a la classe “errors de programari”.*
c	Les vulnerabilitats d'espionatge intern pertanyen a la classe “errors de programari”. *Fals. Les vulnerabilitats d'espionatge intern pertanyen a la classe “enginyeria social”.*
d	Les vulnerabilitats de protecció personal pertanyen a la classe “errors de programari”. *Fals. Les vulnerabilitats de protecció personal pertanyen a la classe “defectes en polítiques de seguretat”.*

20. La classificació de vulnerabilitats basada a l'origen...

a	...identifica quatre classes: enginyeria social, errors de programari, defectes en polítiques de seguretat i febleses generals. *Fals. Aquestes classes de vulnerabilitats s'identifiquen en la classificació temàtica.*

A

b	...considera l'interval necessari per a explotar una vulnerabilitat (i reaccionar a ella en cas d'atac) i el grau d'interacció necessari per a explotar-la. *Fals. Aquestes dues dimensions s'han considerat en la classificació temàtica.*
c	...considera que l'origen de les vulnerabilitats podrà facilitar una guia per a corregir-les. *Vertader. Aquest és el principal criteri o motiu que va conduir a l'adopció d'aquesta classificació de vulnerabilitats.*
d	...especifica que els atacs són la causa (és a dir, l'origen) de les vulnerabilitats. *Fals. Per a iniciar un atac l'atacant necessita identificar alguna vulnerabilitat en el seu sistema objectiu. Per tant, la relació existent és l'oposada: sense vulnerabilitats no podrà haver-hi atacs. Per tant, les vulnerabilitats són allò que permet els atacs.*