NIST - Rec_Second Partial. 2025-01-30

This exam is worth 10 points, and consists of 17 questions. Each question poses 4 alternatives and has only one correct answer. Once discarded the worst answer, each correct answer earns 10/16 points, and each error deducts 10/48 points. You must answer on the answer sheet.

C

1. ***This client program was used in Unit 3 to illustrate that the REQ/REP pattern could block client/server interactions when one of the servers (for example, the one using port 8888) aborted after receiving a request but before returning its corresponding response. In that case, the second message was never sent to the second server.***

```
const zmq = require('zeromq')
const rq = zmq.socket('req')
rq.connect('tcp://127.0.0.1:8888')
rq.connect('tcp://127.0.0.1:8889')
rq.send('Hello')
rq.send('Hello again')

rq.on('message', function(msg) {
   console.log('Response: ' + msg)
})
```

What would happen if the corresponding server used ROUTER sockets instead of REP sockets (along with the other relevant changes to properly receive requests and correctly send responses) and the same failure situation occurred?

A. The client would no longer block in this situation
B. There would be executions in which the client would block and others in which it would not
C. The situation initially described in the statement could never occur when using the REQ/REP pattern
D. No change would be observed: the client would continue to block

2. ***This client program was used in Unit 3 to illustrate that the REQ/REP pattern could block client/server interactions when one of the servers (for example, the one using port 8888) aborted after receiving a request but before returning its corresponding response. In that case, the second message was never sent to the second server.***

```
const zmq = require('zeromq')
const rq = zmq.socket('req')
rq.connect('tcp://127.0.0.1:8888')
rq.connect('tcp://127.0.0.1:8889')
rq.send('Hello')
 rq.send('Hello again')

rq.on('message', function(msg) {
   console.log('Response: ' + msg)
})
```

What would happen if this client used DEALER sockets instead of REQ sockets (along with the other relevant changes to properly send requests and correctly receive responses) and the same failure situation occurred?

A. There would be executions in which the client would block and others in which it would not
B. The situation initially described in the statement could never occur when using the REQ/REP pattern
C. No change would be observed: the client would still block
D. The client would no longer block in this situation

3. ***What advantages does a virtual machine offer when compared to a container in a deployment?***

A. Lower resource consumption
B. Greater isolation, along with less dependency on the host operating system
C. Simpler configuration files, providing easier deployment
D. None; a container will always be a better option

4. ***This is an example of a Dockerfile used in both Practice 3 and Unit 4:***

```
FROM tsr-zmq
COPY ./tsr.js tsr.js
RUN mkdir broker
WORKDIR broker
COPY ./broker.js mybroker.js
EXPOSE 9998 9999
CMD node mybroker 9998 9999
```

Select the true statement about the contents of this file:

A. The first line indicates that the image to be generated will be saved in the `tsr-zmq` directory
B. The `RUN` and `WORKDIR` instructions used in this file allow the `mybroker.js` file to locate the `tsr.js` module in the expected location
C. Wherever this file resides, we must also have the `tsr.js` and `mybroker.js` files so that there are no errors when using this Dockerfile
D. All the statements are true

5. ***A Dockerfile allows you to:***

A. Specify which instructions should be used to deactivate, stop, and delete containers used in the deployment of a service.
B. Deploy multiple instances of various related components on a single host computer.
C. Specify which instructions should be used to generate a virtual machine.
D. Specify which instructions should be used to build an image that will allow you to start containers.

6. ***How many segments and their contents does a DEALER socket add or expect (and remove) when sending or receiving a message?***

A. It does not add, expect, or remove any segments in either the sending or receiving operation
B. It adds a leading delimiter when sending a message, and expects and removes a leading delimiter when receiving a message
C. It adds the identity of the sending socket when sending a message, and does not expect, add, or remove segments when receiving a message
D. It expects, uses, and removes, a segment with the identity of the receiver when sending a message, and adds the identity of the sending socket when receiving a message

7. ***Which of the following stages of the software life cycle is not included in deployment?***

A. Activation
B. Analysis
C. Upgrade
D. Installation

8. ***Select the true statement about the `docker compose up` command.***

A. All the statements are true
B. It assumes that there will be a `docker-compose.yml` file in the directory where it is used
C. It deploys a distributed service, starting an instance (i.e. a container) of each of its components, in a certain order
D. It allows you to use the `--scale compo=X` option to run `X` instances of the `compo` component

9. ***Select the true statement about passive replication:***

A. Once the request to be managed is received, it requires fewer interactions between replicas than active replication
B. Recovering from a failure of your primary replica is more complex than from the failure of a secondary replica
C. It offers better performance than active replication, especially when your operations generate many high-volume modifications
D. It offers better performance than multi-master replication

10. ***Select the true statement about active or multi-master replication:***

A. Multi-master replication is more appropriate than active replication if we intend to tolerate Byzantine failures
B. Multi-master replication is generally faster at recovering services after a failure than active replication
C. Multi-master replication is appropriate if we intend to replicate non-deterministic services
D. Multi-master replication is less appropriate than active replication for highly scalable environments

11. ***Consider the following example `docker-compose.yml` file:***

```
version: '2'
services:
  ca:
    image: ima
    build: ./dira/
    links:
      - cc
    environment:
      - C_HOST=cc
      - C_PORT=9998
  cb:
    image: imb
    build: ./dirb/
    links:
      - cc
    environment:
      - C_HOST=cc
      - C_PORT=9999
  cc:
    image: imc
    build: ./dirc/
    expose:
      - "9998"
      - "9999"
```

Select the true statement about that file:

A. When the service described in that file is deployed, the first component to be started is `ca`, since it is specified first

B. The Dockerfile present in the `dirc` subdirectory uses two environment variables called `C_HOST` and `C_PORT`

C. All other statements are false

D. When the service described in that file is deployed, the first component to be started is `cc`

12. ***Suppose a system consisting of three processes P1, P2 and P3, where the following execution has occurred: `W1(x)2, R2(x)2, W2(y)1, R1(y)1, R3(y)1, R3(x)2`. This execution respects, among others, the following consistency:***

A. Strict

B. Sequential

C. Causal

D. Cache

13. ***Select the true statement about active replication:***

A. It manages non-deterministic operations without generating inconsistencies between replicas

B. It can offer better performance than passive replication, especially when the operations to be managed generate large volume modifications

C. It uses a primary replica and several secondary ones

D. It is the replication model used in the component that manages persistent data on Wikipedia

14. ***Let us suppose a CBW system to which we will add two `logger` components, to collect the trace messages generated in the frontend and backend sockets of the broker respectively. To do this, the broker will use an additional PUB socket, on which it will apply a `bind` and with which it will emit messages with two segments, with "frontend" or "backend" in the first. The ports to be used by this broker will be received from the command line: first the PUB one, second the client ROUTER one and third the worker ROUTER one. If the initial Dockerfile for the broker has this content:***

```
FROM tsr-zmq
COPY ./tsr.js tsr.js
RUN mkdir broker
WORKDIR broker
COPY ./broker.js mybroker.js
```

What lines should be added to properly handle that new configuration? (due to layout, some lines may have been split in two)

A. `EXPOSE 9997 9998 9999`
`CMD node mybroker 9997 9998 9999`

B. `EXPOSE 9998 9999`
`CMD node mybroker 9998 9999 $LOGGER_HOST $LOGGER_PORT`

C. `EXPOSE 9998 9999`
`CMD node mybroker 9998 9999 localhost 9997`

D. `EXPOSE 9997 9998 9999`
`CMD node mybroker 9997 9998 9999 $LOGGER1_HOST $LOGGER2_HOST`

15. ***Select the true statement about the CAP theorem:***
    - **A.** Partition tolerance, as mentioned in the CAP theorem, means that tolerating partitions is equivalent to using the primary partition model.
    - **B.** The CAP theorem indicates that it is most common to sacrifice availability in highly scalable systems.
    - **C.** If we have a system where the primary partition model is adopted and we have high consistency, we will be sacrificing availability.
    - **D.** If we have a system that follows the causal model, we will be guaranteeing strong consistency, therefore we will have to sacrifice support for partitions or availability.

16. ***The command to find out the IP address of a container that provides a service, after it has been started is (assuming we already know the container ID or name):***
    - **A.** None of the other options are true
    - **B.** `docker ps`
    - **C.** `docker inspect <ID>`
    - **D.** `docker logs <ID>`

17. ***Let us assume a CBW system to which we will add two `logger` components, to collect trace messages generated in the broker's frontend and backend sockets respectively. To do this, the broker will use an additional PUB socket, on which it will apply a `bind` and with which it will emit messages with two segments, with "frontend" or "backend" in the first. For their part, the two `logger` components share the same program and the same image. Each one will subscribe to a different prefix. Based on that prefix, a different log file name will be generated. The directory in which they will save those files in their containers will be the same for both: `/tmp/cbwlog`. The `logger.js` program will need to receive, in this sequence, these three arguments from the command line: `BrokerHost brokerPort prefixToSubscribe`. If the initial Dockerfile for the `logger` has this content:***

```
FROM tsr-zmq
COPY ./tsr.js tsr.js
RUN mkdir logger
WORKDIR logger
COPY ./logger.js mylogger.js
```

What should be appended to that Dockerfile to properly handle that scenario? (due to layout, some lines may have been split in two)

- **A.**
```
VOLUME /tmp/cbwlog
CMD node logger $BROKER_HOST $BROKER_PORT
$PREFIX
```
- **B.**
```
VOLUME /tmp/cbwlog
CMD node logger localhost $BROKER_PORT
$PREFIX
```
- **C.**
```
VOLUME /tmp/cbwlog
CMD node mylogger $BROKER_HOST $BROKER_PORT
$PREFIX
```
- **D.**
```
VOLUME /tmp/cbwlog
EXPOSE 9997
CMD node mylogger 9997 /tmp/cbwlog/logs
```