1. Briefly explain each of the following distributed system challenges.

For each challenge, give a practical system example that illustrates it.

i. Heterogeneity

Distributed systems use hardware and software resources of varying characteristics (heterogeneous resources). For example, in Grid computing clusters computers with various operation systems they should work together to execute jobs.

ii. Scalability

A system is considered to be scalable if it can handle the growth of the number of users. For example, the number web servers of the specific website should be increase when total number of requests on that website grows.

iii. Openness

Openness refers to the ability of extending the system in different ways by adding hardware or software resources. The process is akin to the standardization of interfaces but it often bypasses official standardization. A web application which is implemented based on the microservices architecture and use RESTful APIs for communicating components is open.

2. Compare threads and processes and provide a practical example for each. Explain what is Context Switching?

Processes are typically independent, while threads exist as subsets of a process
Processes carry considerably more state information than threads, whereas multiple threads within a process share process state as well as memory and other resources
Processes have separate address spaces, whereas threads share their address space
Processes interact only through system-provided inter-process communication mechanisms
Context switching between threads in the same process typically occurs faster than context switching between processes
Example: Spell checker is a thread in Microsoft word
Example: Nginx runs at least one master process.
Each task is given a small amount of time to run before the next task is switched in. Context switches take care of all this. It saves all the information needed (program counter, values of variables), so the program can be returned to.

3. Which speedup could be achieved according to Amdahl´s law for infinite number of processors if 2.5% of a program is sequential and the remaining part is ideally parallel?

1 / (1-0.975) = 1/0.025 = 40

4. Compare Message passing and thread programming models.

Process
Sockets

5. What is the difference between the cold and live migration of VMs. Give an exemplary scenario that requires each of these migrations.

VM and all running applications in it will be stopped in cold migration. Example: we have an stopped VM that we would like to migrate it to another host for maintenance purposes of the physical sever.

Live migration refers to the process of moving a running virtual machine or application between different physical machines without disconnecting the client or application. Example we need to migrate a VM that hosts a web server that cannot be interrupted.

6. Explain role of service in Kubernetes. Provide an example that you need to use services.

Pods are not reliable and might die. Managed via Deployments or DaemonSets, they get replaced when they fail. Replacements come with totally different IPs. Services provide reliable networking for a set of Pods that is using selectors and label detects newly added pods and distributed traffic between them. Example, in assignment one, we had a web service for object detection in images running on multiple pods for scalability purposes. We designed a service that was detected added pods automatically based on the labels and distribute loads between them automatically.

7. You run a meme creation website where users can create memes and then download them for use on their own sites. The original images are stored in S3 and each meme's metadata in DynamoDB. You need to decide upon a low-cost storage option for the memes. Which storage solution should you use to store the non-critical, easily reproducible memes in the most cost-effective way?

S3 – 1Zone-Infrequent Access

8. Suggest an AWS service to a software engineer of a company to make communication of two components of a distributed application fail-safe? Justify your answer.

SQS

9. Compare Representational State Transfer (REST) and SOAP (Simple Object Access Protocol) based web services for implementing service-oriented architectures.

	REST	SOAP
	REST is an architectural style (more flexible)	SOAP is a protocol (highly specified)
	No official standard for RESTful APIs	Has an official standard
	REST is focused on accessing named resources through a single consistent interface	SOAP focuses on exposing pieces of application logic (not data) as services
Documentation	Easier to understand (generally speaking)	More complex documentation
Data formats	REST permits many different data formats	SOAP only permits XML
Performance	REST has better performance and scalability	SOAP not so scalable
Cacheable	REST reads can be cached	SOAP based reads cannot be cached
Security	REST supports SSL	SOAP supports SSL but also WS-Security which adds some enterprise security features (XML signatures, XML encryption)
Reliability	Limited to single HTTP transactions. Cannot implemented two-phase commit (2PC) protocol. REST expects the client to re-try if something fails.	Can ensure ACID transactions (Atomicity, Consistency, Isolation, Durability) - guarantees that database transactions are processed reliably. SOAP has successful/retry logic built in.
Useful for...	All types of system apart from where high security and high reliability is critical	"Enterprise" systems and high security / reliability systems e.g. banking systems

10. There are many open challenges in delivering secure clouds. Describe some the technical and non-technical issues that currently exists for development and delivery of security-oriented clouds (at least 4 issues).

Losing control of resources, lack of trust, Multi-tenancy challenges, lack of a standard language to convey one's policies and expectations, Secure data deletion, auditing and compliance

Also you can have a look at slides 26-29 in the cloud security slide deck, if you're interested to see a classification of attacks and vulnerabilities based on Cloud level (IaaS, PaaS, SaaS)