**Client-server and Peer-to-peer**

**Client-server:**

1. Describe the client-server network architecture.
   1. A network architecture where one or more computers / devices (the clients) are connected to a powerful central computer (the server). Each client may have it’s own data and files and resources and can also access resources on the server. The server handles the data transmissions.
2. What are the advantages and disadvantages of a client-server network?
   1. Security is better since all files are stored in a central location and access rights to the central location are managed by the server
   2. Backups are done centrally so clients don’t have to individually update. Hard for data to be permanently accidentally lost
   3. Costly to setup, one point of failure, if too many computers, could overload because server cant handle that many
3. What is Cloud computing and what are its benefits?
   1. Services that provide access to files and software on the internet, so you don’t need to store data with hardware, it is all done online. It is good because you can then access data remotely.

**Peer-to-peer:**

1. Describe the peer-to-peer network architecture.
   1. Where two or more computers are connected to each other. Unlike a client-server model, there is no central computer, all devices are just connected to each other
2. What are the advantages of a peer-to-peer network, and when would it be used?
   1. It is simple and cheap to set up because you don’t need a server or a lot of hardware. It is also quite easy to maintain. The fact that it also doesn’t have one single point of failure helps since if one computer breaks down, and you still need to for example, share files, and you need to send data through the broken down computer, since its peer to peer and all computers connected to each other, there are other routes data can take. However, this poses a security risk since if one computer is connected and secretly a hacker or unauthorised person, they can receive the files and data and steal it.
3. What are the downsides of peer-to-peer networks?
   1. Security risk above

**Client-side processing:**

1. What is client-side processing, and what are the benefits and limitations of this approach?
   1. Where processing takes place at the client, so if the computers on the network handle the processing, rather than a server.this can be good as it maintains security since data doesn’t need to be sent to a server, so it cant be intercepted during a transmission. Also it is faster since at a server, there might be traffic already due to other requests, and also no transmission time since data isnt sent anywhere else. However, this is bad for processing that needs to be handled by someone else, since on a client, the client can just process it themselves however they want, so for example if you want to buy a plane ticket, you can just process it like you get the plane ticket, without paying for it.
2. What types of tasks can be performed on the client side in a client-server architecture?
   1. Processing / loading websites, any form of task that doesn’t require third party, validation

**Server-side processing:**

1. What is server-side processing, and how does it differ from client-side processing?
   1. Processing is done at a server or central powerful computer, so instead of taking place at the client, it is done elsewhere.
2. What types of tasks are typically handled by a server in a client-server architecture?
   1. Online transactions / purchases, retrieving data from elsewhere, verification
3. What are the main security concerns in client-server networks, and how can they be mitigated?
   1. When using server side processing, since data has to be sent to the server to be processed, It could be intercepted and changed or stolen etc. The server itself could also be hacked, if it is compromised, everyone’s data is compromised since everyone is connected to it.
4. What is a Distributed Denial of Service (DDoS) attack, and how can a server defend against such attacks in a client-server architecture?
   1. When an overly large amount of requests are made to the server, since server can only handle so many requests at a time, and the server handles requests in order, so if a genuine client trying to request from the server, their request is so far back in the line it basically is never attended to. Can be avoided by limiting number of requests from a single IP, or filtering requests to check for genuine requests. Load balancing can be used to distribute requests to one server to multiple other servers, slows down processing a little since now other servers have more requests, but is used in conjunction with other methods.

**API (Application Programming Interface):**

1. What is an API?
   1. Set of protocols and rules that governs how two applications should interact with one another
2. Describe an example of how a website may utilise an API.
   1. For example, you might use a weather API to get information on the weather to put into your application.

**Thin- versus Thick-client computing:**

1. What is meant by thin- or thick-client computing?

Amount of processing handled by the client, so thin is the server handles most of the processing, whereas thick client computing is where the client does most of the processing

1. What are the advantages and disadvantages of each?

For thick:

* + - Server doesn’t need to be as powerful, useful if you have a lot of people trying to use the server, it is hard for the server whereas if the clients do most of it, the server doesn’t need to be as strong
    - Don’t need continuous connection to server to process, so if something goes wrong with the cabling, most of what you need to process can still be processed
    - Can be more expensive for clients because clients need a higher specification and be able to do more processor intense actions
    - It therefore doesn’t make much sense for every client to have to have a super good computer

Thin:

* + - Less load on client since most of the processing is done by server rather than client so client doesn’t need a super powerful device
    - Frees up more space for client since don’t need to store as many things that are being processed since the server will hold most of the data that needs to be used, like worldwide maps, a client wouldn’t want to have a whole super detailed map on their phone of the world because it would take too much space
    - Need more powerful server since if you have many clients using thin computing, the server needs to have plenty of storage and be powerful enough to handle all the processing.
    - Therefore can be a lot more expensive due to hardware and needing specialists to maintain more powerful servers.