10. Tétel

a) Hálózati architektúrák és protokollok

Csomagkapcsolt hálózatok működése

- Csomagkapcsolás: Az információt kisebb darabokra, úgynevezett csomagokra bontják. Minden csomag tartalmazza a szükséges információkat, például a forrás- és célcímeket, amelyek lehetővé teszik, hogy önállóan eljussanak a címzetthez.

- Önálló útvonal: A csomagok különböző útvonalakon haladhatnak, és nem szükséges, hogy egy csomagnak ugyanazon az útvonalon kelljen eljutnia a célhoz, mint a többinek. Ezt a rugalmasságot biztosítja a hálózat.

- Újra összerakás: Amikor a csomagok megérkeznek a célállomásra, a hálózati protokollok segítségével újra összeállítják őket az eredeti üzenetté.

az OSI és TCP/IP modell összehasonlítása

- OSI részletesebben bontja le a hálózati kommunikációs folyamatot. TCP/IP modell gyakorlatiasabb és szélesebb körben használt

- OSI (7 réteg):

- Fizikai réteg: A legalsó réteg, amely a fizikai adathordozók (kábelek, rádióhullámok) és az átviteli technológiák (Ethernet, Wi-Fi) kezeléséért felel. Bit szinten továbbítja az adatokat.

- Adatkapcsolati réteg: Biztosítja az adatátvitel hibamentességét és a fizikai címzést (MAC cím). Feladatai közé tartozik a csomagok keretekre bontása, a keretek hibajavítása, valamint a hálózati hozzáférés szabályozása.

- Hálózati réteg: Felelős az útvonalválasztásért és a logikai címzésért (IP címek). Ez a réteg határozza meg, hogyan jutnak el a csomagok egyik hálózati csomópontból a másikba.

- Szállítási réteg: Megbízható adatátvitelt biztosít a hálózaton keresztül, például TCP protokoll segítségével. Feladata a hibajavítás, az adatcsomagok helyes sorrendben való összeállítása, valamint az áramlásvezérlés.

- Session: Kezeli a kapcsolatokat a hálózatban lévő alkalmazások között. Ez magában foglalja a kapcsolat létesítését, kezelését és befejezését.

- Prezentációs réteg: Az adatok átalakítása, tömörítése és titkosítása történik itt, hogy biztosítsák a hálózaton keresztül történő helyes értelmezést.

- Applikációs réteg: Ez a legfelső réteg, amely az alkalmazások közötti kommunikációt biztosítja. Példák: HTTP (web), FTP (fájlátvitel), SMTP (email)

- TCP/IP modell (4 réteg):

- Hálózati hozzáférési réteg: Kombinálja az OSI modell Fizikai és Adatkapcsolati rétegeit. Itt történik a bit szintű adatátvitel és a keretezés.

- Hálózati réteg: Felelős a csomagok hálózati útvonalválasztásáért és IP címzésért.

- Szállítási réteg: Biztosítja az adatátvitelt az alkalmazások között. Protokollok: TCP (megbízható, kapcsolat-orientált), UDP (gyors, kapcsolat nélküli).

- Applikációs réteg: Alkalmazási réteg: Hasonló az OSI modell legfelső három rétegéhez (Applikációs, Prezentációs, Session). Tartalmazza az alkalmazásokhoz szükséges protokollokat, mint a HTTP, FTP, DNS.

forgalomirányítási és IP címzési alapok

- Forgalomirányítás (Routing)

- Routerek feladata: A hálózati csomópontok (routerek) feladata a csomagok továbbítása a célállomás felé a leghatékonyabb útvonalon.

- Routing tábla: A routerek belső táblázatokban (routing table) tartják nyilván az útvonalakat. Ez tartalmazza a célhálózatokat, a subnet mask-okat, a kimenő interfészeket, a következő hop routert és a metrikákat (útvonalak költsége).

- Forgalomirányítási protokollok:

- Routing protokollok: A routerek közötti információcseréért felelős protokollok, például RIP (Routing Information Protocol), OSPF (Open Shortest Path First), BGP (Border Gateway Protocol).

- Egyéb protokollok: ICMP (Internet Control Message Protocol) diagnosztikai és hibaüzenetek küldésére.

- IP címzés:

- Az IP címzés lehetővé teszi az egyedi azonosítást és az eszközök közötti kommunikációt a hálózaton belül.

- Az IP címek három fő osztályba sorolhatók: A, B és C osztály. Mindegyik osztály különböző hálózati és hosztcímzési struktúrát kínál.

- Felépítése:

- Hálózat és csomópont azonosító: Az IP cím eleje a hálózat azonosítója, a vége a csomópont azonosítója.

- IPv4 Címzés: 32 bites, négy oktettből áll (pl. 192.168.1.1)

- IPv6 Címzés: 28 bites, hexadecimális formátum (pl. 2001:0db8:85a3:0000:0000:8a2e:0370:7334).

- Osztályok:

- A osztály: 0 - 127, hálózati maszk: 255.0.0.0, prefix hossz: 8.

- B osztály: 128 - 191, hálózati maszk: 255.255.0.0, prefix hossz: 16.

- C osztály: 192 - 223, hálózati maszk: 255.255.255.0, prefix hossz: 24.

- Speciális Címek:

- Privát IP címek: Nem kerülnek közvetlenül az internetre, hanem helyi hálózatokban használatosak. Példák: C osztály: 192.168.0.0 - 192.168.255.255

- Broadcast cím: Minden hoszt számára szól a hálózaton belül. Például, egy 192.168.1.0/24 hálózat broadcast címe 192.168.1.255.

- Loopback cím: 127.0.0.1, amit a hoszt saját magával való kommunikációra használ.

Kapcsolódó protokollok

- TCP (Transmission Control Protocol):

- Funkció: Megbízható, kapcsolat-orientált adatátvitel. Biztosítja, hogy az adatok helyes sorrendben és hibamentesen érkezzenek meg.

- Használat: Fájlátvitel (FTP), web (HTTP), email (SMTP).

- UDP (User Datagram Protocol):

- Funkció: Kapcsolat nélküli, gyors adatátvitel. Nem garantálja az adatok sorrendjét és hibamentességét, ezért gyorsabb.

- Használat: Streaming szolgáltatások (pl. videó), online játékok.

- ICMP (Internet Control Message Protocol):

- Funkció: Hibaüzenetek és diagnosztikai információk küldése. Például, ha egy csomag nem ér el a céljához, ICMP üzenetet küld vissza a forrásnak.

- Használat: Ping parancs

b) Szolgáltatás orientált programozás

Az RPC architektúrája, működési elve, jellemzői

- Remote Procedure Call: egy program egy másik programot hív meg egy távoli számítógépen anélkül, hogy a hálózati kommunikáció részleteivel kellene foglalkoznia. Leegyszerűsíti a távoli hívások végrehajtását úgy mintha helyi függvényhívások lennének

- Architektúrája: 2 fő komponensből áll

- Kliens:

- klienscsonk (stub): kliensoldali fgv. hívásokat átalakít távolira

- Kommunikációs modul: továbbít a hálózaton keresztül

- Szerver:

- szervercsonk (stub): ugyanezt csinálja csak magának

- Kommunikációs modul: fogadja a kliens által küldött üzeneteket és továbbítja a szerver oldali stubhoz

- Működése elve:

- Függvényhívás (kliens oldalról)

- Kliens stub összegyűjti a paramétereket és továbbít a hálózaton (marshalling)

- Hálózati kommunikáció (a szerver felé)

- Szerver stub (fogadás és kicsomagolás unmarshalling)

- Függvény végrehajtás (végrehajtódik és a szerver stubnak küldi vissza)

- Hálózati kommunikáció (vissza a kliensnek)

- Kliens fogadás (stub fogadja, kicsomagolja és a kliens alkalmazza)

- pl. egy összeadás funkció

- RPC jellemzői:

- átláthatóság, nyelvi függetlenség, protokoll függetlenség, szinkronizálás, hibakezelés

A Google RPC működési elve, jellemzői

- nyílt forráskódú távoli eljáráshívás (RPC) keretrendszer, amelyet a Google fejlesztett ki

- gRPC protokollon keresztül működik

- Protokoll Buffers (protobuf)

- .proto fájlban definiáljuk a szolgáltatást

- Jellemzői:

- magas teljesítményű (HTTP/2 protokoll miatt)

- nagy sebességű adatátvitel és alacsony késleltetés

- nyelv és platformfüggetlen

- streaming támogatás (RPC, szerver oldali, kliens oldali és kétirányú)

- erős típusellenőrzése (protobuf használata biztosítja)

- biztonságos (támogatja az SSL/TLS alapú hitelesítést)

A protocol buffers szerepe a gRPC-ben

- protobuf: nyelvfüggetlen, platformfüggetlen mechanizmus az adatok sorosítására és deszerializálására

- a gRPC-ben ebben a fájlban definiálod a szolgáltatási interfészeket

- maga gRPC kódgenerátor a protobuf fájl alapján hozza létre a kliens és szerver stubokat (ezek tartalmazzák a szükséges logikát és utasításokat a hívásokhoz)

- adatátvitelnél bináris formátumot használ, és ez sokkal gyorsabb mint a szöveges formátumok

- típusellenőrzés és szerződés betartása

A WEB, egyrétegű, kliens-szerver modell, többrétegű alkalmazások, vastag és vékony kliensek, elosztott rendszerek jellemzői

- különböző komponensek együttműködése és kommunikálása

- Egyrétegű alkalmazások:

- monolitikus

- az összes komponens egyetlen alkalmazásban fut

- könnyen fejleszthető, tesztelhető, kis projekthez, nehezen skálázható

- Kliens-szerver Modell:

- kliensre és szerverre oszlik

- fentebb volt már róla szó

- Többrétegű alkalmazások

- N-tier architecture

- több különálló rétegre van bontva

- prezentációs réteg (UI réteg), üzleti logika, adatkezelési réteg, esetleg köztes rétegek

- jobb karbantarthatóság, jó skálázhatóság, de nagy komplexitású

- Vastag kliens:

- rich/thick client, jelentős mennyiségű logikát futtat a kliens oldalon

- jobb felhasználói élmény, kevesebb hálózati forgalom, de bonyolult telepítés és karbantartás

- Vékony kliens:

- minimális logikát futtat a kliens oldalon

- a legtöbb logikát a szerver végzi, ezért egyszerűbb karbantartás, nagy hálózati és adatforgalom

- Elosztott rendszerek:

- a komponensek különböző hálózatba kapcsolt képeken futnak

- jó a hibatűrésük

A REST tulajdonságai

- webszolgáltatásokhoz

- az objektumokat/adatokat egyedi azonosítókkal érjük el (URI) pl. http://api.example.com/books/1

- gyakran API-val használjuk

- az adatokat általában JSON XML HTML-ben kezeljük

- CRUD műveletek támogatása

- állapotmentes, azaz minden kérés független az előző kérésektől

A web service jellemzői

- szabványos webprotokollokat használ a kommunikációra (XML, JSON, HTTP)

- HTTP/HTTPS, SOAP, REST

- itt is fontosok azok a dolgok amik pl a resztnél vannak

- biztonságos, állapotmentes, aszinkron komm.

A WCF architektúrája

- Micorosft keretrendszer (Windows Communication Foundation)

- webszolgáltatásokhoz

- endpointon keresztül a kliensek hozzáférnek a szolgáltatásokhoz

- a végpontok három fő elemből épülnek fel:

- cím: http://localhost:8000/Service

- kötés: pl. BasicHttpBinding

- szerződés: szolgáltatás művelete (interfész határozza meg)

- hosztolhatók, self-hosztolhatók

- biztonságos

- támogatja a SOAP-ot is

az ASMX és a WCF közötti eltérések

- ASMX (Active Server Methods)

- ez is támogatja HTTP/HTTPS protokollokat, de ez kizárólag SOAP-alapú webszolgáltatásokra korlátozódik

- nem self hosztolható

- összességében korlátozottabb