ITBI Tutoriatele 4-5

Albert Simon, Spiridon Mihnea-Andrei

November 26, 2025

1 Rețele de calculatoare

1.1 Ce e o rețea

Suntem obișnuiți cu ideea de a lucra pe propriul calculator. Însă, putem să ne aflăm în situația în care avem nevoie să interacționăm la distanță cu alt calculator, poate un coleg de lucru sau un server. Rețeaua de calculatoare e conceptul care descrie legăturile dintre un set de dispozitive interconectate.

1.2 Rețeaua locală

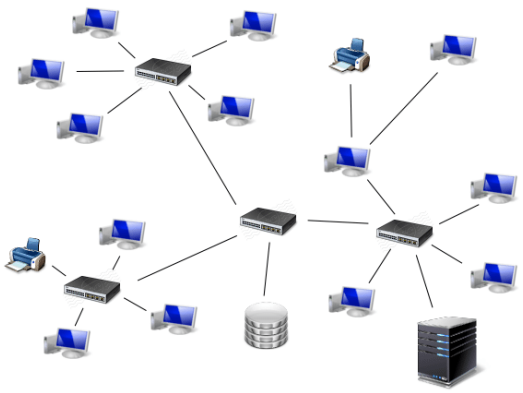

Figure 1: Dispozitive în rețea

O rețea de calculatoare poate fi modelată ca un graf, unde nodurile sunt dispozitive și muchiile sunt conexiunile stabilite între ele. Putem vedea graful ca fiind fără cost, însă în practică, costurile de pe muchii pot reprezenta diverse costuri din lumea reală e.g. timpi de așteptare pentru a trimite anumite date prin rețea. Aici își găsesc aplicabilitate algoritimii pe grafuri.

Rețeaua locală se stabilește între dispotivite apropiate. Gândiți-vă că acasă probabil aveți o rețea de care se ocupă **router-ul** vostru, poate aveți 2 calculatoare, un telefon, un smart TV, un aparat de aer condiționat care poate fi controlat printr-o aplicație web - aceste dispozitive se află în rețeaua locală.

În rețea fiecare dispozitiv este identificat printr-o adresă IP. O adresă IP este o secvență de 4 numere de la 0 la 255, precum: 192.168.1.200. Putem exprima adresa IP și în hexa: C0.A8.01.C8, iar "coincidența" face că avem un total de $8 \cdot 4 = 32$ biți aici, cât un întreg pe 32 de biți.

Pentru rețeaua locală sunt rezervate anumite range-uri de adrese IP. Cel mai comun veți întâlni. 192.168.XX.XX, însă puteți citi și lista tuturor domeniilor de adrese IP rezervate pentru rețele locale (LAN, **local access network**): https://en.wikipedia.org/wiki/Reserved_IP_addresses.

Observație: Întotdeauna 127.0.0.1, sau *localhost*, se referă la dispozitivul curent.

Ce comenzi putem încerca?

- **ifconfig** (s-ar putea să fie necesar să o instalați) pentru a afla diverse informații despre cine suntem în rețea.
- **ssh** user@ip ca să deschidem un terminal, logat ca user, pe dispozitivul de la adresa ip specificată.
- De exemplu, porniți o mașină virtuală, conectați-o la rețea, și aflați adresa cu ifconfig. Apoi, conectați-vă la ea din gazdă prin ssh și creeați un fișier. Îl veți putea citi din mașina virtuală!
- **scp** sursă:locație destinație:locație pentru a face operația de **cp** de pe un alt calculator

1.3 WAN

Cuvântul "internet" vine de la "interconnection of computer networks". Practic, din rețeaua locală vom putea accesa calculatoare din altă rețea locală, la distanță foarte mare (gândiți-vă că ne putem conecta, de ex., la un server din Australia).

O conexiune la un alt dispozitiv se face acum prin orice adresă IP valabilă, nu prin cele rezervate. Cele rezervate se referă în continuare la dispozitive din LAN.

O conexiune este o pereche (ip, căsuță poștală), unde căsuța poștală, cunoscută și sub numele de **port**, este un identificator unic de la 0 la 65535 pentru o rută de comunicare. Această pereche poartă numele de **socket**.

O cerere la un anumit socket (un IP și un port) ajunge la router. Router-ul poate fi configurat să facă **port forwarding** în care trimite toate datele primite pe un anumit port la un anumit dispozitiv din LAN, și răspunsul e primit de la același dispozitiv și trimis înapoi la cel care a făcut cererea. De exemplu, în rețeaua locală putem avea două servere care primesc mesaje pe porturile 9001 și, respectiv, 9002. În WAN ieșim cu IP-ul 89.101.55.73. Dacă încercăm să ne conectăm la 89.101.55.73:9001, vorbim cu primul server, iar dacă încercăm 89.101.55.73:9002, vorbim cu al doilea server.

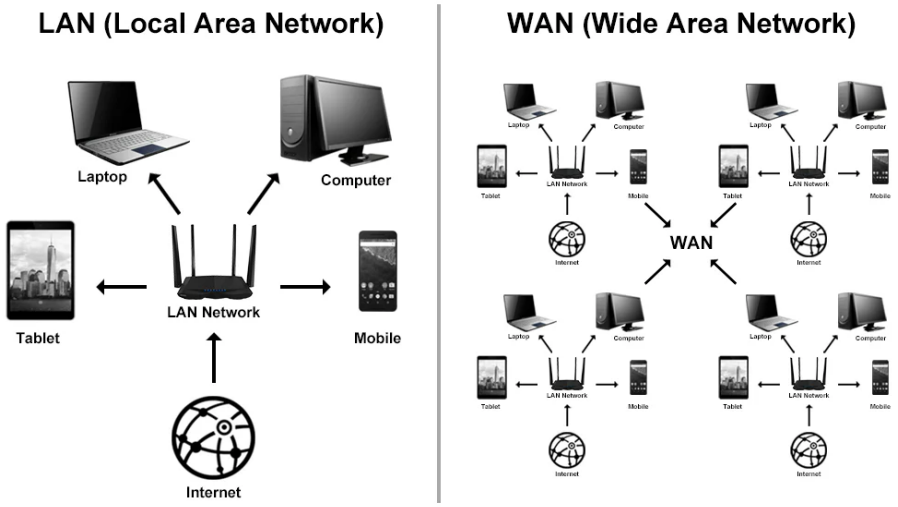

Figure 2: Acesta este internetul

1.4 DNS

Am vorbit anterior de conectare prin socket-uri, perechi adresă IP-căsuță poștală (port). Dar atunci când ne conectăm la un website de obicei avem un string de tipul "fmi.unibuc.ro" sau "google.com". Cum facem asocierea de la string la IP?

Soluția Unix tradițională este fișierul **/etc/hosts**. Acesta era folosit la început, în epoca în care existau 100-200 de rețele, în principal servere menținute de diverse universități, sau structuri militare. Obțineai cumva adresa IP și îi treceai în /etc/hosts un alias ca să nu mai trebuiască să reții IP-ul. Acest fișier este prima formă de **domain name resolution** (procesul de a trece de la nume la IP) inventată și încă ia prioritate, dacă sunt găsite intrări, însă acum e de obicei goală.

Soluția modernă: Nu memorăm IP-ul la cele peste 2 miliarde de website-uri, servere etc., ci memorăm 1-2 IP-uri ale unor servere specializate, numite DNS (domain name server). Un exemplu de DNS faimos este găsit la IP-ul 8.8.8.8, este un server creeat de Google care răspunde la cereri de forma "ce IP are asociat numele fmi.unibuc.ro?". Principial, un DNS răspunde la întrebarea "unde îl găsesc pe NUME", unde NUME este cel la care doresc să mă conectez.

Exemplu: Conectarea la fmi.unibuc.ro din perspectiva calculatorului.

- Se trece numele fmi.unibuc.ro în browser sau alternativ
- Se verifică /etc/hosts; nu va exista intrare, se apelează la lista de DNS-uri și se încearcă unul până unul oferă un răspuns
- Apelez la un DNS (pe care îl cunosc în prealabil, am IP-ul lui, 8.8.8.8, memorat)
- DNS-ul îl cunoaște pe fmi.unibuc.ro și îmi spune că adresa lui fmi.unibuc.ro este 89.57.19.102
- Calculatorul începe conexiunea cu 89.57.19.102 (care este site-ul facultății)

Comenzi utile pentru lucrul cu DNS:

- **nslookup** întreabă un DNS care este IP-ul unui nume
- **dig** are comportament similar cu nslookup, dar e mai parametrizabil

2 Git

2.1 Ce este Git?

Git este un program foarte des-utilizat în industrie. Funcționalitatea lui este de **source control**, anume un sistem prin care putem avea un proiect la care lucrează mai mulți oameni în același timp, pe diferite versiuni, având opțiunea să revenim la o versiune mai veche a proiectului dacă ceva nu merge bine (e.g. schimbăm ceva și apare o eroare pe care nu mai știm să o rezolvăm).

Unitatea de lucru în Git este repository-ul, puteți să îl gândiți ca reprezentând întreg proiectul + istoricul proiectului vostru.

2.2 Diferența dintre Git și GitHub

Este foarte important să înțelegem diferența dintre Git și GitHub. Git e un program, care poate lucra complet local, independent de alții, dacă vrem noi pentru noi înșine să beneficiem de source control. GitHub este un hub de repository-uri la care ne putem conecta.

2.3 Procesul de lucru în Git

- Facem **git clone** pentru a obține o copie locală a unui repository din GitHub
- Facem modificările noastre la proiect
- Pregătim un commit: facem **git add** la fișierele modificare și apoi **git commit**
- Urcăm modificările pe GitHub cu **git push**
- Vedem dacă între timp au făcut alții vreo modificare cu **git fetch**
- Dacă da, obținem noile modificări local, cu **git pull**

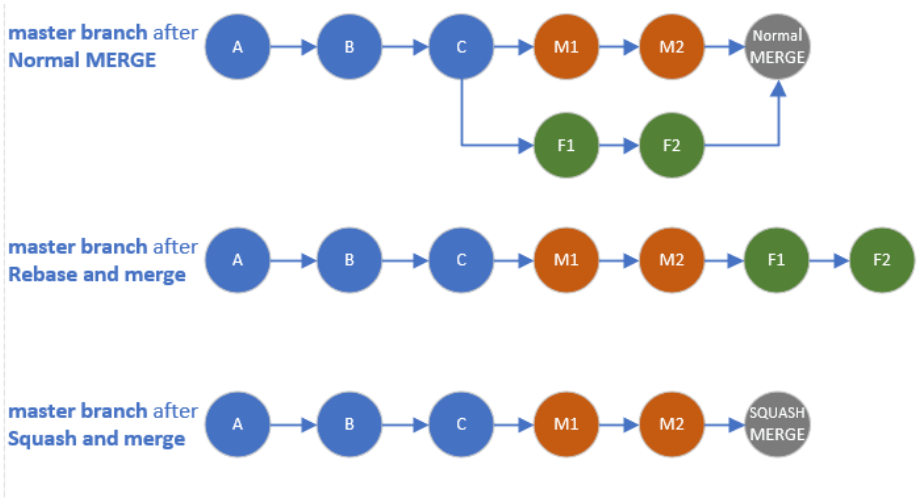

Figure 3: Un proiect în GitHub, evoluând prin versiuni în timp și pe diverse branch-uri

2.4 Recomandare de lucru

Lumea în general nu lucrează cu git din linia de comandă. Deși este util să știm despre comenzi și, conceptual, să învățăm despre principiul de funcționare al git, vom găsi că ne e mai util să lucrăm cu o interfață grafică, e.g. **GitKraken** sau **GitHub Desktop**. Unele IDE-uri precum