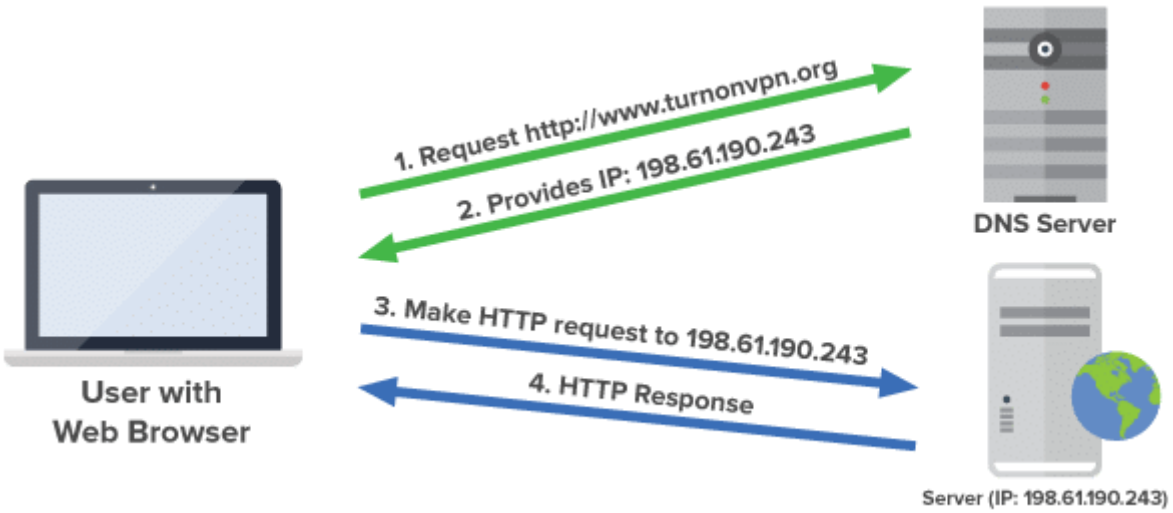
DNS – Domain Name System

Sistem koji prevodi Domain imena u IP adrese. Kada korisnik želi da komunicira sa drugim korisnikom ili serverom (npr. WEB pregledač) neophodno je da zna tačnu njegovu lokaciju u mreži, odnosno IP adresu. Da ne bi pamtili sve adrese u mreži i pravili svoj imenik adresa komunikacija otpočinje tako što znamo ime korisnika ili ime domena.

Kada u Web browser-u ukucamo neku URL adresu, računar ne ide direktno, odmah do tog web servera, ne traži sadržaj tog sajta već prvo kontaktira DNS server. Informacija gde se nalazi, na kojoj IP adresi DNS server uneta je ili ručno, statički, ili je IP adresa dobijena od DHCP servera. Računar se obraća DNS-u i šalje upit da mu DNS server da IP adresu na kojoj se nalazi sajt sa datim imenom. DNS na osnovu dobijenog imena pronađe u svojim registrima IP adresu koja je povezana sa datim imenom i pošalje računaru odgovor sa odgovarajućom IP adresom.

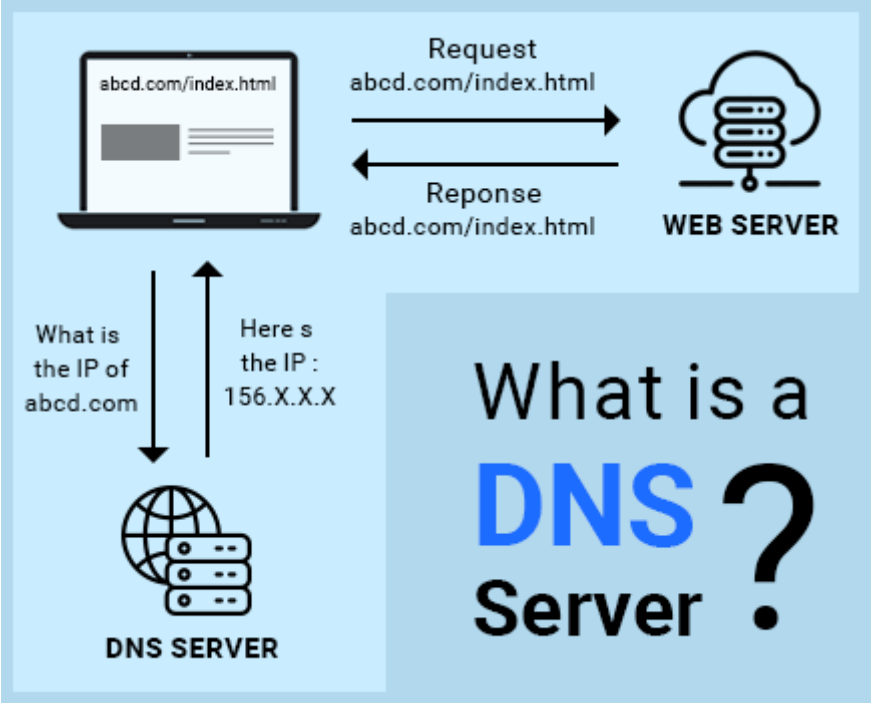
DNS server prvo pretražuje u svojim registrima, ukoliko ne pronađe odgovor šalje zahtev sledećem nadređenom DNS-u i sve tako dok se ne pronađe odgovor. Računar kada sazna pravu IP adresu nekog sajta direktno šalje zahtev tom web serveru da mu prosledi njegov sadržaj. Sve ovo dešava se u pozadini rada Web pregledača i računara tako da mi ne vidimo to.

DNS - Registar u kome su Domain imena mapirana IP adresama, svako Domain ime, URL upareno je sa nekom IP adresom. DNS server predstavlja adresar imena i IP adresa.



Kada računar sazna IP adresu datog željenog sajta čije mu je ime bilo poznato, u svojoj memoriji privremeno čuva IP adresu, neki vremenski period, tako da ne mora svaki put kada pošalje zahtev tom web serveru za sadrzaj datog sajta da šalje i upit

opet DNS-u. Da ne postoji mogućnost čuvanja IP adrese nekog sajta u svojoj privremenoj memoriji, računar bi morao svaki put kada želimo da promenimo stranu datog sajta, kliknemo na neke delove sajta, otvorimo druge delove sajta, da šalje upit DNS-u da sazna IP adresu.



DHCP – Dynamic Host Configuration Protocol

Računar, bilo koji krajnji korisnik, može dobiti IP adresu na 2 dva načina, odnosno dodeljivanje IP adrese može biti:

- Statičko – Hostu (računaru) statički je dodeljena IP adresa od strane administratora (korisnika)
- Dinamičko – Host (računar) zahteva od DHCP da mu dodeli IP adresu, ili ukoliko DHCP server nije dostupan usvaja se mehanizam za samostalno automatsko usvajanje IP adrese iz unapred definisanog opsega

Dinamička dodela IP adrese od strane DHCP servera

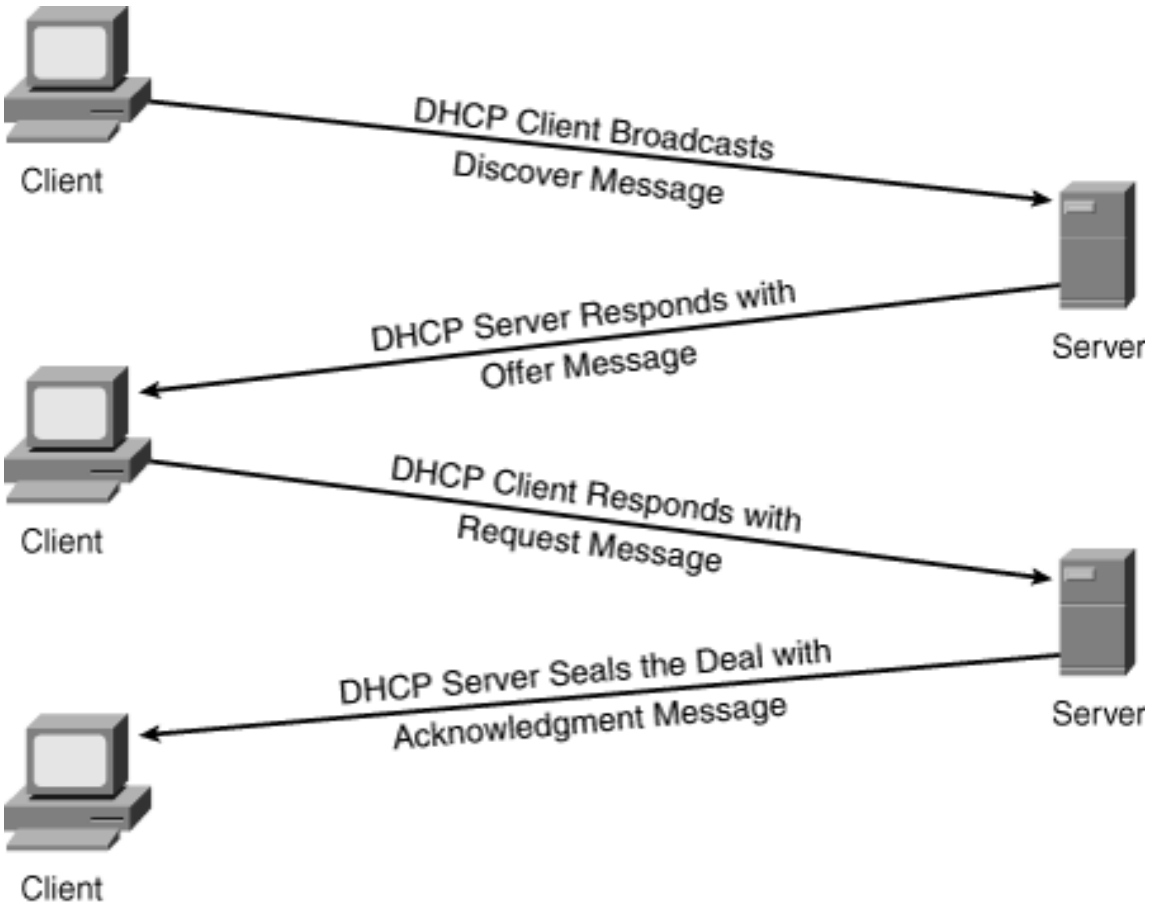
Osnovene informacije koje DHCP dodeljuje su: IP adresa, maska, Default Gateway, IP adresa DNS servera.

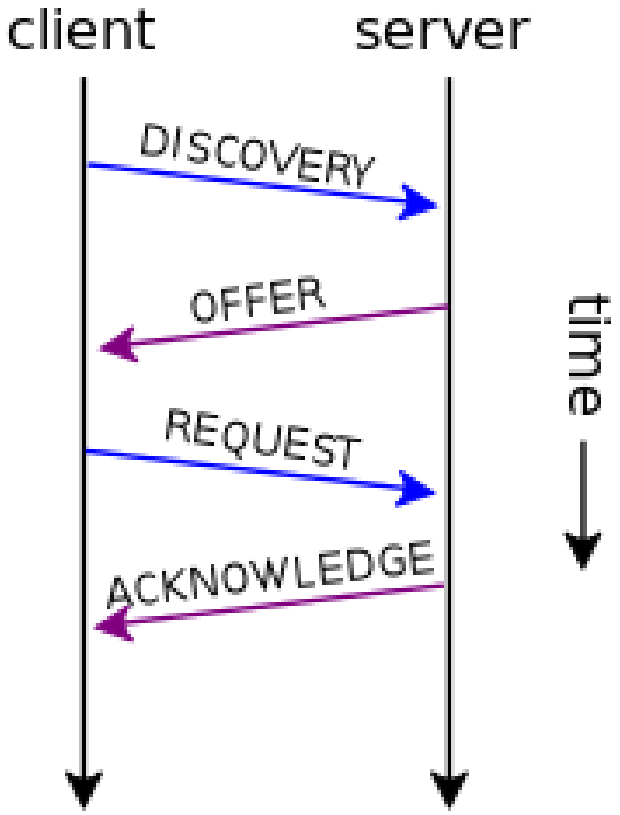
- IP adresa – sa njom se računar identifikuje u reži, logička adresa računara na određenom mestu u mreži. Da bi mogao da komunicira sa drugim korisnicima neophodno je da dobije IP adresu da bi ostali mogli da znaju gde se nalazi i koja mu je lokacija u mreži.
- Subnet mask – identifikuje mrežu kojoj pripada, na koju je vezan računar. Bez maske IP adresa ne znači ništa, zajedno čine jedinstvenu lokaciju korisnika u mreži. Na osnovu maske korisnika i IP adrese vidi se kojoj LAN mreži pripada računar.
- Default Gateway – identifikuje mrežni uređaj preko kojeg računar, mreža kojoj pripada, pristupa udaljenim lokacijama van mreže, komunicira sa računarima iz drugih mreža. Da bi računar komunicira sa drugim korisnicima koji su u drugim LAN mrežama, neophodno je da poseduje izlaz iz sopstvene mreže, odnosno Default Gateway, odnosno IP adresu rutera na koji je njegova mreža nakačena.
- Opciono daje informaciju na kojoj IP adresi se nalazi DNS server. Kada bi znali i pamtili IP adrese svih korisnika, servera, sajtova na mreži, DNS nam ne bi ni bio potrebam. Pošto je ljudskom mozgu mnogo lakše da zapamti neki alfabetski niz, rađa se potreba za DNS-om, a samim tim i da znamo IP adresu DNS-a.

DHCP je protocol koji je zadužen da korisnici dobiju automatski IP adresu i sve ove parameter čim izraze želju da se priključe na mrežu i aktivno učestvuju na njoj.

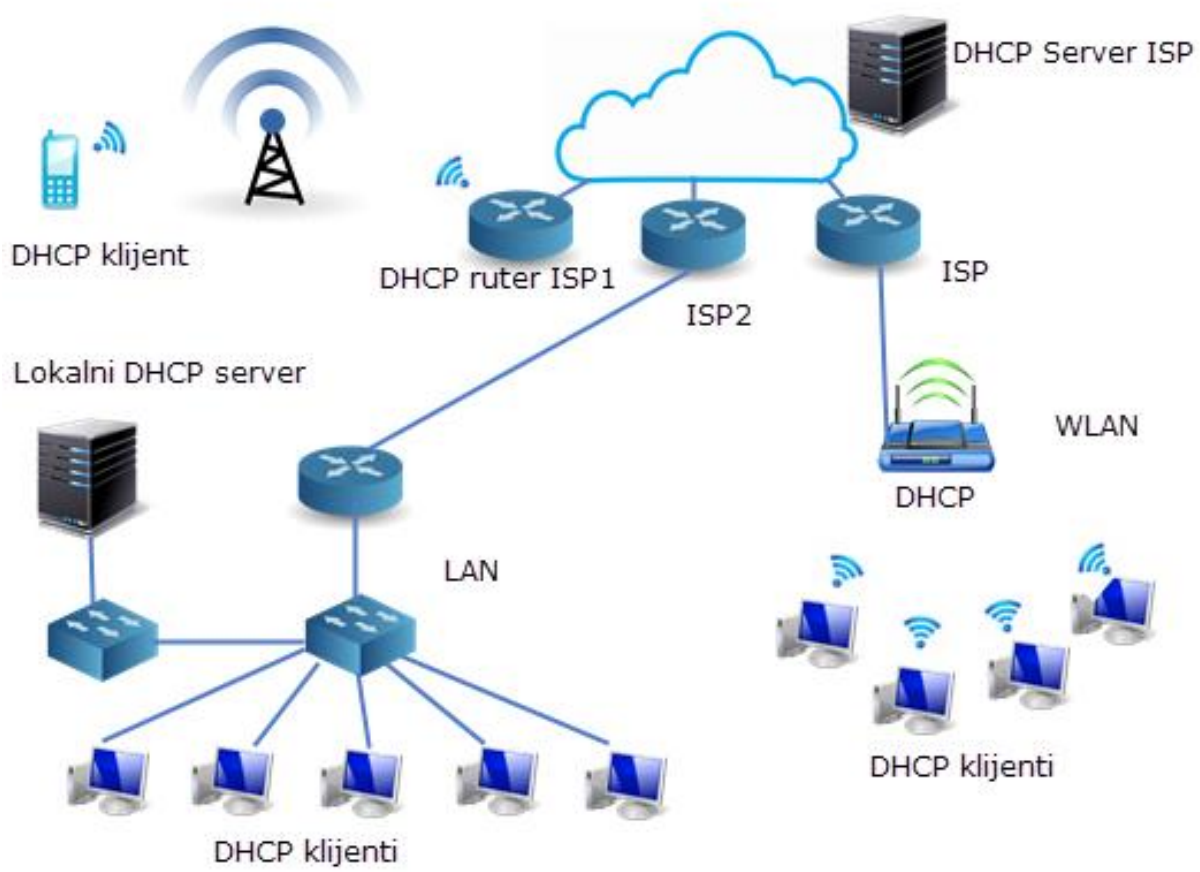
Postupak DHCP mehanizma:

- DHCP Discover – računar šalje broadcast poruku na mrežu u cilju otkrivanja DHCP servera i traži da mu se dodeli IP adresa
- DHCP Offer - poruka stiže do svih, pa i do DHCP servera koji prepoznaje da neki host želi da pristupi mreži, odgovara mu tako što mu ponudi IP adresu iz opsega koji je namenjen za dodeljivanje
- DHCP Request – računar obaveštava da želi da prisvoji tu adresu i šalje poruku DHCP serveru
- DHCP Acknowledgment – DHCP potvrđuje da računar može da prisvoji tu adresu


Client
DHCP Client Broadcasts
Discover Message
Server
DHCP Server Responds with
Offer Message
Client
DHCP Client Responds with
Request Message
Server
DHCP Server Seals the Deal with
Acknowledgment Message
Client

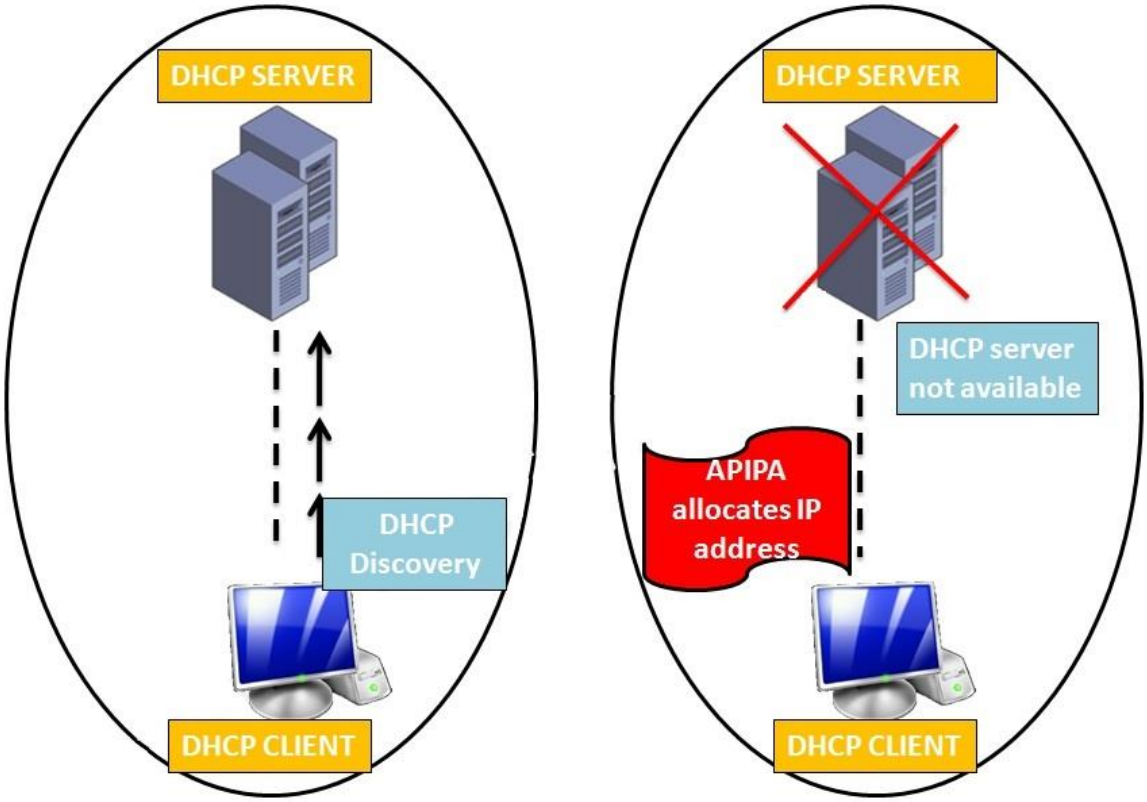

client
server
DISCOVERY
OFFER
REQUEST
ACKNOWLEDGE
time

DHCP može biti svuda na mreži:



APIPA – Automatic Private IP Address

Automatska privatna IP adresa. Ukoliko IP adresa računara nije statički podešena i ukoliko DHCP ne postoji u mreži ili ne radi kako treba računar sam sebi dodeljuje IP adresu iz sledećeg opsega: 169.254.0.0 – 169.254.255.255. To se dešava kako bi računar bio u mogućnosti da se poveže sa nekim drugim računarom bez potreba poznavanja računarskih mreža i načina adresiranja od strane korisnika. Računar sam dodeljuje sebi adresu iz datog opsega.



HTTP – HyperText Transfor Protocol

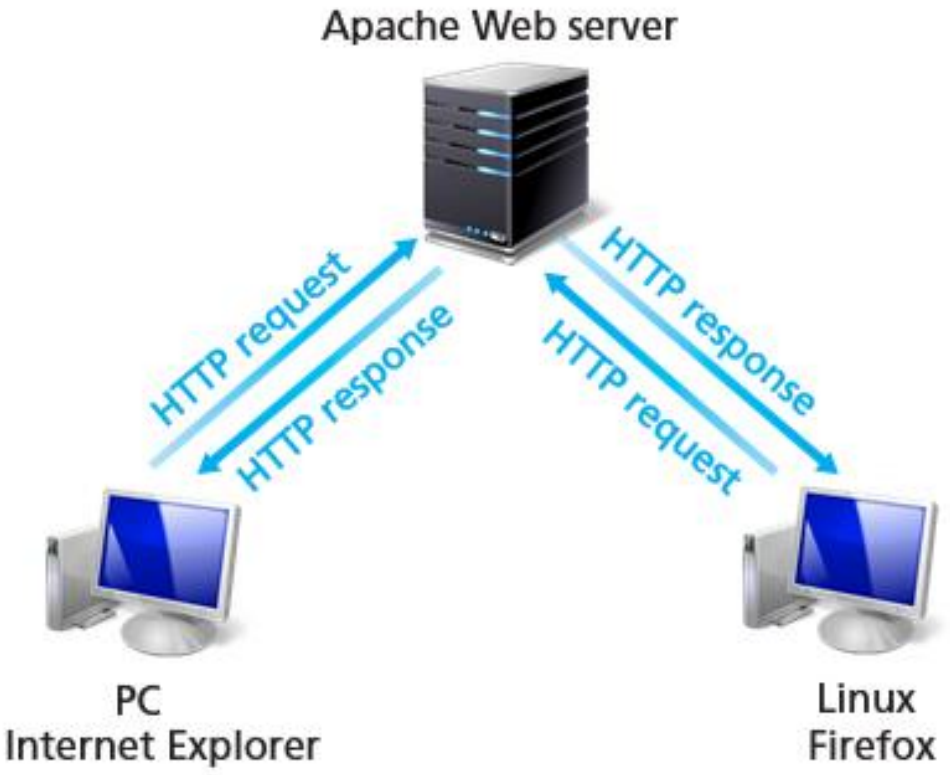
U samom srcu Weba nalazi se protokol HTTP, protokol sloja aplikacija. HTTP se implementira u dva programa - klijentskom i serverskom. Protokol HTTP definiše način na koji web klijenti (Čitači) traže web strane od web servera, kao i način na koji web serveri šalju tražene strane klijentima. Kada korisnik zatraži neku web stranicu (mišem izabere hipervezu), čitač šalje serveru HTTP poruke zahtevajući objekte sa date stranice, server prima ove zahteve i odgovara HTTP porukama u kojima se nalaze traženi objekti.

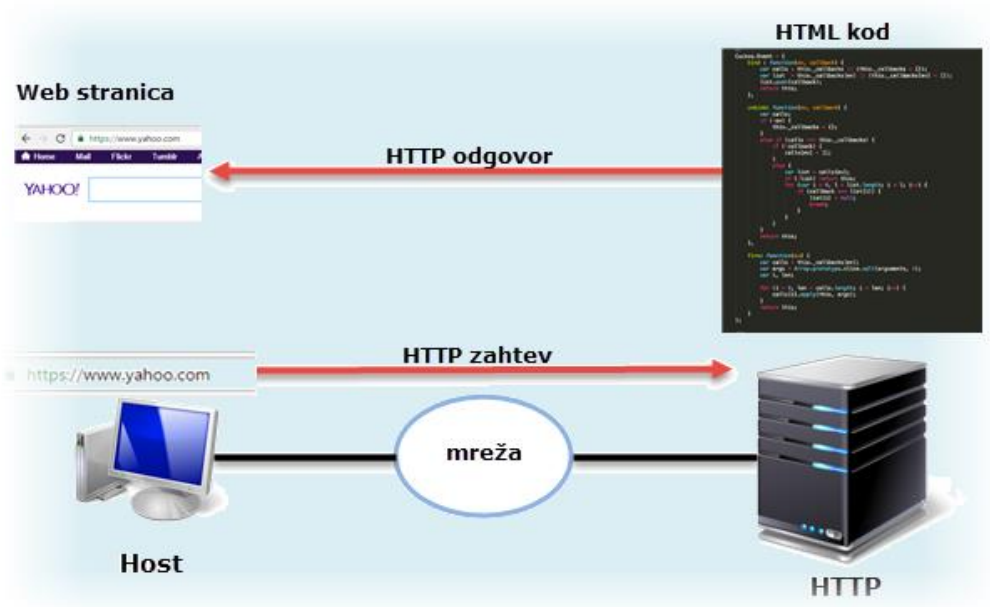
- Web resurse obezbeđuje Web server. Host pristupa resursima koristeći Hipertekt Transfer Protocol (HTTP) ili bezbedni HTTP (HTTPS)
- HTTPS dodaje šifrovanje i servis provere identiteta pomoću Secure Sockets Layer (SSL) protokola ili novijeg Transport Layer Securiti (TLS) protokola
- HTTP radi na portu 80. HTTPS radi na portu 443

Browser interpretira tri dela URL-a:

- 1. http (protokol ili šema)
- 2. www.yahoo.com (naziv servera)

- 3. index.html (zahtev specifične datoteke)
- Pretraživač zatim proverava sa Domain Name Server (DNS) za pretvaranje www.yahoo.com u numeričke adrese, koje se koristi za povezivanje na server. Koristeći HTTP zahtev, pretraživač šalje zahtev na server i traži index.html fajl. Server šalje HTML kod za ovu web stranicu nazad u browser-u klijenta, i na kraju, browser prikazuje formatiranu stranicu HTML koda.





Hypertext Transfer Protocol (HTTP) je mrežni protokol koji pripada aplikativnom sloju OSI referentnog modela mreže. HTTP predstavlja glavni i najkorišćeniji metod prenosa informacija preko Interneta. Osnovna namena ovog protokola je isporučivanje HTML stranica (Web prezentacija). Razvoj i standardizaciju HTTP protokola nadgledaju W3C (World Wide Web Consortium) i IETF (Internet Engineering Task Force).

HTTP je protokol za komunikaciju između servera i klijenta, koji funkcioniše po principu zahtev/odgovor. HTTP klijent, koji je najčešće veb pregledač, inicira prenos podataka nakon što uspostavi TCP/IP vezu sa udaljenim veb serverom na određenom mrežnom komunikacionom portu (uglavnom je to port 80, a ređe, može biti i 8080). Klijent šalje HTTP zahtev serveru. Server, na kojem su smešteni web sadržaji, kao što su HTML fajlovi i slike, vraća odgovor klijentu. Odgovor sadrži statusne informacije zahteva i može sadržati sadržaj koji je zahtevan.

Server konstantno osluškuje zahteve na već pomenutom portu, čekajući da se klijent poveže i pošalje svoj zahtev. Zahtev se sastoji iz osnovne HTTP komande (čija je sintaksa propisana standardom i koja se sastoji iz naziva komande, imena traženom veb resursa i verzije podrđanoh HTTP-a) i zaglavlja koje se sastoji od određenog broja redova teksta koji bliže opisuju aspekte zahteva.

Zahtev klijenta se obrađuje na serveru i, u zavisnosti od ispravnosti zahteva i mogućnosti zadovoljavanja istog, kli-jentu se šalje odgovarajući odgovor. Odgovor se sastoji od izveštaja o sta-tusu zahteva (koji se sastoji od tro-cifrenog koda i kratkog deskri-ptivnog teksta statusa, npr. 200 OK) i od konkretnog odgovora, ukoliko je zahtev moguće zadovoljiti. Odgovor se sastoji od zaglavlja koje je iste sintakse kao i zaglavlje zahteva, i daje osnovne podatke o prirodi odgovora.

Struktura HTTP poruka

HTTP koristi klijent/server mrežni model: HTTP klijent otvara konekciju i šalje zahtev HTTP serveru; server zatim vraća odgovor klijentu, koji obično sadrži resurse koji su traženi. Nakon slanja odgovora, server zatvara konekciju. Format zahteva i odgovora su slični i imaju sledeću strukturu:

- Inicijalna linija,
- Nula ili više linija u zaglavlju,
- Prazna linija,
- Opcionalno - telo poruke like koje može biti fajl, tekst itd.

Inicijalna linija : Request (Zahtev)

Inicijalna linija je različita kod zahteva i odgovora. Inicijalna linija u zahtevu ima tri dela koji su odvojeni razmacima (spaces):

- HTTP Ime metod,
- Lokalna putanja traženog resursa,

- Verzija HTTP-a koji se koristi.

Evo primera inicijalne linije kod zahtevne poruke.

GET /path/to/file/index.html HTTP/1.0

- GET je najkorišćenija HTTP metoda.
- Putanja je deo URL-a nakon imena hosta. Ova putanja se takođe zove i URI (request Uniform Resource Identifier). URI je sličan URL-u, stim što je URI opštiji.
- Verzija HTTP-a dolazi na kraju zahteva u formatu "HTTP/x.x.

Inicijalna linija: Response (Odgovor)

Inicijalna linija odgovora, koja se naziva statusna linija, se takođe sastoji od tri dela razdvojenih razmacima:

- Verzija HTTP-a koja je korišćena
- Statusni kod
- Opis statusnog koda na engleskom jeziku

Evo primera inicijalne linije kod odgovora:

HTTP/1.0 200 OK

Telo poruke

HTTP odgovor može sadržati telo poruke u kojem se nalaze zahtevani resursi od strane klijenta, a ponekad se u telu poruke može naći i detaljnije objašnjenje o nastaloj grešci. U HTTP zahtevu, telo poruke sadrži resurse koje treba poslati na server.

Ako HTTP poruka sadrži telo, onda moraju postojati i dodatne linije u zaglavlju koje će opisati tekst iz tela poruke. Najčešće su to:

- Content-Type: zaglavlje opisuje tip podataka koje se nalaze u telu poruke, kao što su npr. text/html ili image/gif.
- Content-Length: zaglavlje daje broj bajtova iz tela poruke.

Pimer zahteva i odgovora:

Request

GET /index.html HTTP/1.1

Host: www.example.com

Response

HTTP/1.1 200 OK

Date: Mon, 23 May 2005 22:38:34 GMT

Content-Type: text/html; charset=UTF-8

Content-Encoding: UTF-8

Content-Length: 138

Last-Modified: Wed, 08 Jan 2003 23:11:55 GMT

Server: Apache/1.3.3.7 (Unix) (Red-Hat/Linux)

ETag: "3f80f-1b6-3e1cb03b"

Accept-Ranges: bytes

Connection: close

HTTP metode

Slede kratka objašnjenja HTTP metoda:

GET metoda zahteva nalaže serveru da klijentu dostavi tražene resurse koje se nalaze na izabranoj URI adresi. Po standardu, zahtevi koji koriste **GET** metodu ne bi trebalo da vrše bilo koju drugu operaciju osim preuzimanja podataka odnosno resursa.

Metoda **POST** šalje podatke serveru na procesuiranje, najčešće preko HTML formi. Podaci se smeštaju u telo poruke zahteva. Ovim se mogu kreirati novi ili pak ažurirati stari resursi na serveru koji se kasnije mogu preuzeti metodom **GET**.

Metoda **HEAD** traži odgovor koji je isti odgovoru dobijenom **GET** metodom ali bez tela poruke. Ovo može biti korisno za dobijanje maeta podataka koji se čuvaju u zaglavlju odgovora a da se pri tome ne učitava kompletan odgovor.

DELETE metoda briše izabrani resurs na serveru bez bilo kakvog upozorenja.

Metoda **OPTIONS** će vratiti spisak metoda koje server podržava za izabranu URL adresu.

Metode **HEAD**, **OPTIONS** i **GET** su označene kao sigurne metode jer ne vrše nikakve promene na serveru, tj dopuštaju samo pregled resursa. Za razliku od njih, metode **POST**, **PUT** i **DELETE** mogu dovesti do promena stanja servera te se zbog toga smatraju nesigurnim.

HTTP linije zaglavlja

Linije zaglavlja pružaju informacije o zahtevu i odgovoru kao i objektima poslatim kroz telo poruke. Prikazaćemo listu raspoloživih polja zaglavlja koja su dostupna u verziji HTTP/1.0.

Allow

Allow polje zaglavlja prikazuje listu HTTP metoda koje podržavaju resursi na traženoj URI lokaciji. Osnovna svrha je, dakle, da klijentu jasno prikaže koje metode moće da koristi.

Primer: *Allow: GET, HEAD*

Authorization

Ovo polje sadrži podatke o autetinfikaciji klijenta.

Primer: *Authorization : credentials*

Content-Encoding

The Content-Encoding entity-header field is used as a modifier to the media-type. Kada je ovo polje prisutno, njegova vrednost pokazuje na koji način je resurs kodiran, kao i način na koji je potrebno izvršiti dekodiranje. Content-Encoding se najviše koristi za obezbeđivanje bezbedne kompresije dokumenta, pri čemu se zadržava podešeni kodni sistem.

Primer: *Content-Encoding: x-gzip*

Content-Length

Content-Length prikazuje veličinu tela poruke, i to u decimalnom broju okteta, to jest u bajtovima

Primer: *Content-Length: 3495*

Content-Type

Content-Type polje zaglavlja prikazuje tip resursa koji se šalje klijentu kroz telo poruke odgovora.

Primer: *Content-Type: text/html*

From

Ovo polje bi trebalo da sadrži e-mail korisnika koji koristi klijent.

Primer: *From: webmaster@w3.org*

Last-Modified

Prikazuje kada je resurs poslednji put modifikovan odnosno ažuriran.

Primer: *Last-Modified: Tue, 15 Nov 1994 12:45:26 GMT*

Location

Prikazuje lokaciju resursa koristeći zahtevani-URI. Za određene odgovore moguće je da ovo polje prikaže URL adresu. Maksimalno jedna URL adresa se može dobiti kroz ovo polje HTTP zaglavlja.

Primer: *Location: http://www.w3.org/NewLocation.html*

Server

Ovo polje zaglavlja sadrži informacije o softveru koji server koristi da bi obradio zahtev.

Primer: *Server: CERN/3.0 libwww/2.17*

User-Agent

Ovo polje prikazuje informacije o klijentu, to jest veb pregledaču korisnika. Ovo polje se uglavnom koristi u statističke svrhe, za praćenje grešaka u protokolu na različitim klijentima, i slično.

Primer: *User-Agent: CERN-LineMode/2.15 libwww/2.17b3*

HTTP statusni kodovi

Već smo pominjali da se u zaglavlju svakog odgovora u inicijalnoj liniji nalazi statusni kod odgovora. HTTP statusne kodove odgovora možemo podeliti u pet grupa:

- **1XX** – Informacije
- **2XX** – Uspeh
- **3XX** – Redirekcija
- **4XX** – Greška na klijentskoj strani
- **5XX** – Greška na serveru

Slede bliža objašnjenja nekih od statusnih kodova koji se često susreću:

200 OK

Standardan odgovor za svaki uspešan HTTP zahtev. Odgovor zavisi od korišćene HTTP metode zahteva. U slučaju GET metode, ovaj statusni kod obaveštava klijenta

da je resurs pronađen i prikazan. Ukoliko je korišćena POST metoda, status će obavestiti da su preduzete akcije uspešno izvršene.

301 Moved Permanently

Ovaj i svi naredni zahtevi se redirektuju na datu novu URL adresu.

400 Bad Request

Zahtev ne može biti procesuiran zbog greške u sintaksi.

401 Unauthorized

Slično kao i kod "403 Forbidden" statusnog koda, međutim ovaj kod nam poručuje da bismo mogli pristupiti resursima ukoliko izvršimo autentifikaciju.

403 Forbidden

Zahtev je ispravan ali server odbija da odgovori na. Ovaj statusni kod se pojavljuje kada nisu dozvoljena prava pristupa resursima na serveru.

404 Not Found

Traženi resurs nije pronađen na datoj URL adresi, međutim moguće je da će uskori biti dostupan ponovo.

500 Internal Server Error

Generiška poruka o grešci koja se prikazuje kada server nema drugo "objašnjenje" zašto je greška nastala.

HTTP server

Kada kažemo HTTP server u stvari mislimo na Web server, jer Web server upravo koristi HTTP protokol za rad. Osnovna funkcija Web servera je da dostavlja web prezentacije klijentima, dakle, slanje HTML dokumenata i svega ono što takav dokument može sadržati: teks, slike, video materijal itd.

U današnje vreme, najkorišćeniji Web server na Internetu je Apache server.

GIT Bash Terminal komande

ls – dobijamo prikaz svih fajlova i foldera koji postoje u trenutnom folderu (putanji) gde se nalazimo, takođe možemo da izlistamo neki drugi folder, ali je potreno da posle komande navedemo i putanju.

cd <naziv foldera> - ulazimo u taj folder koji smo naveli.

cd .. – izlazimo iz tekućeg foldera (putanje) korak nazad.

git --version – daje nam verziju GIT-a koja je instalirana.

git help <neka komanda> - ovim nizom komandi dobijamo objasnjenje o nekoj komandi koju hoćemo da koristimo, njeno značenje i pored toga spisak komandi koje mogu da se koriste sa njom sa objašnjenjem.

git <neka komanda> --help – isto kao i prethodna komanda, samo možemo da je napišemo na drugi način.

git clone <url> - na ovaj način kopiramo repozitorijum sa servera na lokalni računar, a za url postavljamo putanju do tog željenog repozitorijuma, putanju možemo iskopirati direktno sa servera kada uđemo i otvorimo repozitorijum, dugme *Clone or download*. Na taj način smo prebacili, iskopirali sve fajlove i foldere koji se nalaze u tom repozitorijumu. Njih možemo menjati, brisati, dodavati nove na lokalnom računaru. Sve promene ostaju samo na lokalnom računaru dok ih ne snimimo na git i ažuriramo sa serverom, ako to ne uradimo na serveru se neće videti promene i sve promene ostaju samo na lokalnom računaru.

touch <ime fajla sa ekstenzijom> - pravi novi fajl sa datim imenom i ekstenzijom koju navedemo. Može se takođe dati fajl otvoriti direktno iz terminala tako što se navede ime programa kojim želimo da editujemo fajl pa onda ime fajla.

git status – ovom komando proveravamo da li je bilo nekih promena. Fajlovi koji su obojeni crvenom bojom su novi ili promenjeni, fajlovi koji su obojeni zelenom bojom označavaju fajlove koji su snimljeni ali ne još odobreni i zavedeni sa komentarom, da bi bili ubačeni u git neophodno je potvrditi.

git add <naziv fajla> - kad proverimo status, i uočima da su neki fajlovi promenjeni ili novi (označeni su crvenom bojom) ako želimo da snimimo dati fajl na git neophodno je prvo da upotrebimo ovu komandu. Na ovaj način promene su snimljene ali ne i odobrene, zašta se koristi druga komanda potvrde. Ako proverimo sada sa komandom *git status* fajlovi će biti obojeni zelenom bojom.

git add . – ovom komandom mi snima sve fajlove koji su crvenom bojom označeni, odjednom (zajedno, grupno).

git commit -m'Neki komentar koji hoćemo da ostavimo' – ovom komanda potvrđujemo (odobravamo) ono što smo prethodno snimili, ova komanda se odnosi na fajlove označene zelenom bojom. Kada bi sada proverili komandom *git status* ne bi

postajali fajlovi koji su bili obojeni zelenom bojom, odnosno ne bi imalo ništa na čekanju za potvrdu.

git config --list – proveravamo konfiguracioni fajl, nama je bitno da budemo ulogovani sa našim username-om i mail-om, jer ako još neko koristi git bash na našem računaru može doći do problema prilikom identifikacije korisnika i sinhronizacije sa serverom.

git config --global user.name '<vaš username na git-u>' – ovim potvrđujemo da git koristi naš username, identifikujemo se da smo to baš mi.

git config --global user.email <vaš email koji koristite na serveru> - potvrda da ste vi dati korisnik sa datim registrovanim mail-om.

git push origin master – kada smo snilmili sve promene i nove fajlove, zatim i potvrdili da budu deo git-a na vašem lokalnom računaru, sinhronizacija sa serverom nije izvršine dok ne otkucamo ove komande. Posle toga sv epromene će biti zabeležene i naserveru i to na master grani.

git pull – kada napravimo promene da samom serveru, u smislu novog fajla ili promene starog fajla, ako hoćemo da povučemo te promene sa servera, odnosno sinhronizujemo promene na serveru sa git-om na lokalnom računaru neophodno je izvršiti ove komande u terminalu.

git remote –v – prikazuje nam origin putanju.

git branch –a – prikazuje koja je grana.

git log – pregled log fajla, sve nastale promene zabelezene su u ovom fajlu sa preciznim vremenima i opisom nastalih promena.