# **Tarayıcıya Adres Yazıp “Enter”a Basınca Neler Oluyor?**

Bu yazıda, bir web sayfasının tarayıcınıza ulaşma sürecini teknik bir gözle inceliyoruz. “[www.necoolbirsite.com”](http://www.necoolbirsite.xn--com-9o0a/) gibi bir URL yazıp Enter’a bastığınızda, perde arkasında neler olduğunu adım adım ele alacağız.

## **1. URL’nin Parçalanması: Tarayıcının İlk Adımı**

“[www.necoolbirsite.com”](http://www.necoolbirsite.xn--com-9o0a/) yazıp Enter’a bastığınızda, tarayıcı (örneğin, Chrome veya Firefox) bu girişi bir komut gibi işler:

* **Protokol**: http:// (port 80) veya https:// (port 443, TLS şifrelemeli).
* **Domain**: www.necoolbirsite.com. Alt alan adı (www), ana domain (necoolbirsite) ve TLD (.com).
* **Yol ve Parametreler**: /sayfa?detay=123 gibi, sunucuya spesifik bir kaynağı işaret eder.



Tarayıcı, URL’yi ayrıştırır ve işletim sisteminin ağ yığınına (TCP/IP stack) bir soket açma talimatı gönderir. Ama önce, bu domain’in hangi IP’ye karşılık geldiğini bulmamız gerekiyor.

## **2. DNS: Adres Çözümlemenin Teknik Labirenti**

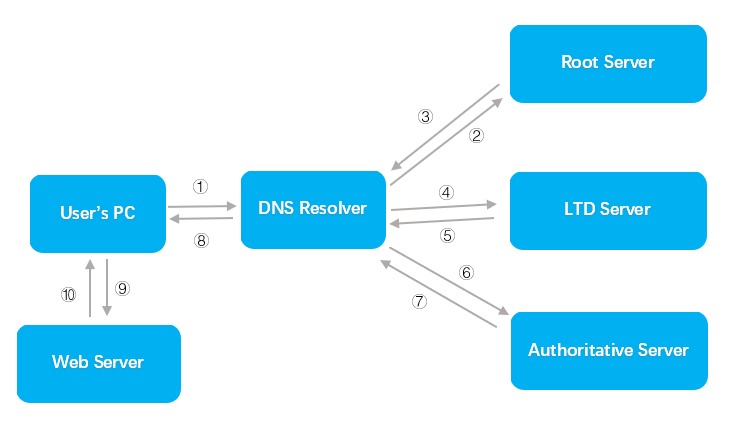
Domain adları insan dostudur, ancak internet IP adresleriyle çalışır (93.184.216.34 gibi). Bu dönüşümü **DNS (Domain Name System)** sağlar. DNS, basit bir “sorgu-cevap” mekanizması gibi görünse de, içinde birçok katman ve çeşitlilik barındırır. İşte adım adım detaylar:

### **2.1. Önbellek Kontrol Aşamaları**

* **Tarayıcı Önbelleği**: Tarayıcı, daha önce çözümlenmiş domain’leri tutar. Örneğin, Chrome’da chrome://net-internals/#dns ile bu önbelleği görebilirsiniz. TTL (Time to Live) süresi dolmadıysa, IP buradan alınır.
* **İşletim Sistemi Önbelleği**: Windows’ta ipconfig /displaydns veya Linux’ta nscd ile kontrol edilir.
* **Hosts Dosyası**: Yerel bir eşleştirme tablosu (/etc/hosts veya C:\Windows\System32\drivers\etc\hosts). Örneğin: 127.0.0.1 localhost  
  93.184.216.34 www.necoolbirsite.com  
   Bu dosya doluysa, DNS sorgusu bypass edilir.

### **2.2. DNS Sorgu Zinciri ve Çeşitleri**

Önbelleklerde sonuç yoksa, DNS resolver devreye girer. Bu süreç, hiyerarşik bir ağ üzerinden ilerler:

* **Yerel DNS Sunucusu (Recursive Resolver)**:
  + Genellikle ISS’niz tarafından sağlanır (192.168.1.1) veya manuel ayarlanır (Google: 8.8.8.8, Cloudflare: 1.1.1.1).
  + UDP port 53 üzerinden çalışır (büyük yanıtlar için TCP’ye geçer).
  + 
* **Root DNS Sunucuları**:
  + Dünyada 13 mantıksal root sunucu var (a.root-servers.net’ten m.root-servers.net’e). Fiziksel olarak Anycast ile yüzlerce kopyaya yayılır.
  + Örnek IP: 198.41.0.4 (Verisign).

* + Görevleri, TLD sunucularını işaret etmektir (.com, .org).
* **TLD Sunucuları**:
  + .com için Verisign, .org için PIR gibi kuruluşlar tarafından yönetilir.
  + necoolbirsite.com’un yetkili name server’larını (NS kayıtları) döndürür, örneğin: ns1.necoolbirsite.com  
    ns2.necoolbirsite.com
* **Yetkili DNS Sunucusu**:
  + Domain’in sahibi tarafından belirlenir (örneğin, hosting firmanızın DNS’i).

* + A kaydıyla IP’yi (93.184.216.34) veya CNAME ile başka bir domain’i döndürür.
* **Kaç Sunucuya Soruluyor?**:

Önbellek boşsa, tipik bir sorguda 3-5 sunucu devreye girer: yerel resolver, root, TLD ve yetkili sunucu. Önbellek doluysa, sadece yerel resolver yeter.

* **DNS Çeşitleri**:
  + **A**: IPv4 adresi.
  + **AAAA**: IPv6 adresi.
  + **MX**: Mail sunucusu.
  + **NS**: Name server.
  + **SOA**: Alan adı otorite bilgisi.
* **Örnek Sorgu**:

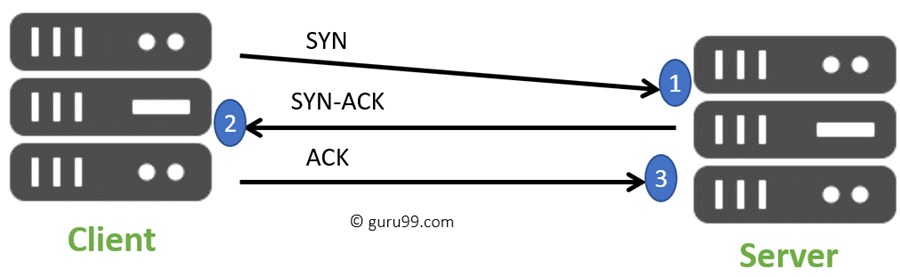
dig www.necoolbirsite.com

Çıktı:

ANSWER SECTION:  
www.necoolbirsite.com. 3600 IN A 93.184.216.34

## **3. TCP/IP ile Bağlantı: Protokollerin Dansı**

IP adresi elimizde. Şimdi sunucuya bağlanacağız:

* **IP (Internet Protocol)**:
  + Paketlerin adresleme ve yönlendirme katmanı.
  + IPv4 başlık örneği: Versiyon: 4 | Başlık Uzunluğu: 20 byte | Kaynak IP: 192.168.1.100 | Hedef IP: 93.184.216.34
* **IPv6**: **Adresleme**: 128 bit, 8 hextet
* (örn. 2001:0db8:85a3:0000:0000:8a2e:0370:7334).
  + Kısaltma: 2001:db8::8a2e:370:7334 (sıfırlar atılır).
  + 
* **TCP (Transmission Control Protocol)**:
  + Güvenilir veri aktarımı için:
    - **SYN**: İstemci, rastgele bir Sequence Number (örn. 1000) gönderir.
    - **SYN-ACK**: Sunucu, kendi Sequence Number’ını (örn. 5000) ve ACK (1001) döner.
    - **ACK**: İstemci, ACK 5001 ile bağlantıyı onaylar.
  + Window Size, akış kontrolünü sağlar (örneğin, 64 KB).
* **Wireshark Örneği**:

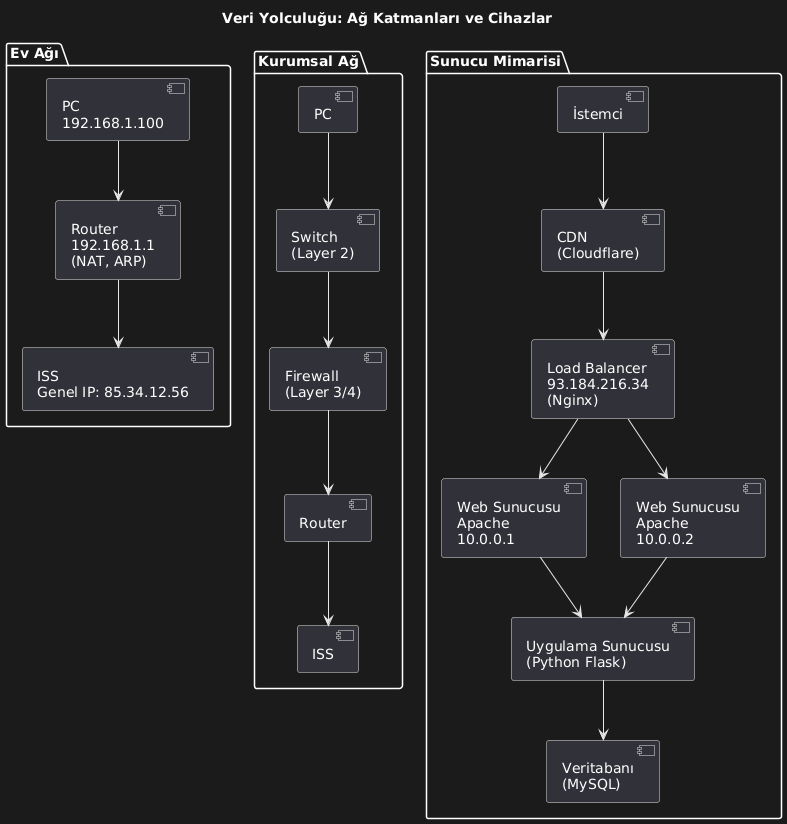
TCP 192.168.1.100:49152 > 93.184.216.34:443 [SYN] Seq=0 Win=64240

## **4. Veri Yolculuğu: Ağ Katmanları ve Cihazlar**

İstek, yerel ağdan globale uzanır:

### **4.1. Ev Ağı**

* **İlk Karşılayan: Router**: [PC: 192.168.1.100] --> [Router: 192.168.1.1] --> [ISS: Genel IP 85.34.12.56]  
  + Router, NAT ile yerel IP’yi genel IP’ye çevirir.
  + ARP, yerel MAC adresini bulur.



### **4.2. Kurumsal Ağ**

* **İlk Karşılayan: Switch** : [PC] --> [Switch] --> [Firewall] --> [Router] --> [ISS]  
  + **Switch**: Layer 2, MAC adresleriyle çalışır.
  + **Firewall**: Layer 3/4, IP ve port filtrelemesi yapar (örn. Cisco ASA).
  + Router, trafiği dışarı yönlendirir.

**4.3. Internet Omurgası**

* **BGP**: AS (Autonomous System) numaralarıyla çalışır (örn. AS15169 - Google) en kısa yolu seçer.
* Fiziksel katman: Fiber optik (10 Gbps), bakır kablo veya uydu.
* **OSPF**: İç ağlarda en kısa yolu hesaplar (Link State Database).
* 

## **5. Sunucu Mimarisi: İsteğin Karşılandığı Yer**

* Bir HTTP isteği sunucuya ulaştığında, modern bir web uygulamasında birden fazla bileşen devreye girer. Tipik bir mimari ve her bir bileşenin teknik detayları:
* [İstemci] --> [CDN] --> [Load Balancer] --> [Web Sunucusu] --> [Uygulama Sunucusu] --> [Veritabanı]

### **5.1. CDN: Cloudflare**

* **Content Delivery Network (CDN)**, statik içeriği (CSS, JS, resimler) önbelleğe alarak istemciye en yakın coğrafi konumdan sunar. Cloudflare, dünya çapında 300+ veri merkeziyle bu hizmeti sağlar.
* **Teknik Detaylar**:
  + **Edge Sunucular**: İstemciye yakın bir "edge node" (örneğin, İstanbul’da bir sunucu) içeriği önbelleğe alır. RTT (Round-Trip Time) genelde 10 ms civarındadır.
  + **Anycast Yönlendirme**: Cloudflare’in IP’si (örn. 104.16.0.1) tek bir sunucuyu değil, en yakın node’u işaret eder. BGP ile otomatik rota seçimi yapılır.
  + **Önbellekleme**:
    - Cache-Control başlığıyla çalışır: Cache-Control
    - Statik dosya (örn. style.css) 1 saat önbellekte tutulur.
  + **Protokoller**: HTTP/2 ve QUIC (UDP tabanlı) ile hızlı teslimat. TLS 1.3 ile şifreleme.
  + **Performans**:
    - 10 ms gecikme, yerel ISS’den 50-100 ms’ye kıyasla büyük avantaj.
    - Bandwidth tasarrufu: Ana sunucuya trafik %70 azalabilir.
  + **Örnek**: GET <https://cdn.necoolbirsite.com/style.css>Response: 200 OK, 10 ms, Content-Length: 15 KB

### **5.2. Load Balancer: Nginx, Round-Robin**

* **Load Balancer**, gelen trafiği birden fazla sunucuya dağıtır, böylece tek bir sunucunun aşırı yüklenmesini önler. Nginx, hem web sunucusu hem de load balancer olarak kullanılabilir.
* **Teknik Detaylar**:
  + **Algoritmalar**:
    - **Round-Robin**: İstekleri sırayla sunuculara gönderir.

* + - **Least Connections**: En az aktif bağlantıya sahip sunucuya yönlendirir.
    - **IP Hash**: İstemci IP’sine göre sabit sunucu seçimi (oturum sürekliliği için).
  + **SSL Sonlandırma**: HTTPS trafiğini çözer, HTTP olarak backend’e iletir.

* + **Sağlık Kontrolü**:
    - Her 5 saniyede bir sunuculara ping: health\_check interval=5s fails=3;
    - Sunucu yanıt vermezse devre dışı bırakılır.
  + **Performans**:
    - 10.000 eşzamanlı bağlantıyı 1 CPU çekirdeğiyle yönetebilir.
    - Latency: 1-2 ms ek gecikme.
  + **Örnek**: [LB: 93.184.216.34]  
     |--> [Web1: 10.0.0.1] (50% trafik)  
     |--> [Web2: 10.0.0.2] (50% trafik)

### **5.3. Uygulama Sunucusu: Python Flask**

* Flask, Python ile dinamik içerik (örneğin, API veya sayfa) üretir.
* **Ağdaki Yeri**:
  + IP: 10.0.0.4 (backend LAN).
  + Port: 8000 (üretimde).
  + Web sunucusundan istek alır:
* GET /post/123 -> 10.0.0.4:8000
  + Yanıt döner:
* {"id": 123, "title": "Merhaba"}
* Gunicorn ile çalışır:

Gunicorn (Green Unicorn), Python tabanlı web uygulamalarını (örneğin, Flask) üretimde çalıştırmak için kullanılan bir **WSGI** (Web Server Gateway Interface), Python’da web sunucuları ile web uygulamaları (örneğin, Flask) arasında standart bir iletişim arayüzüdür.

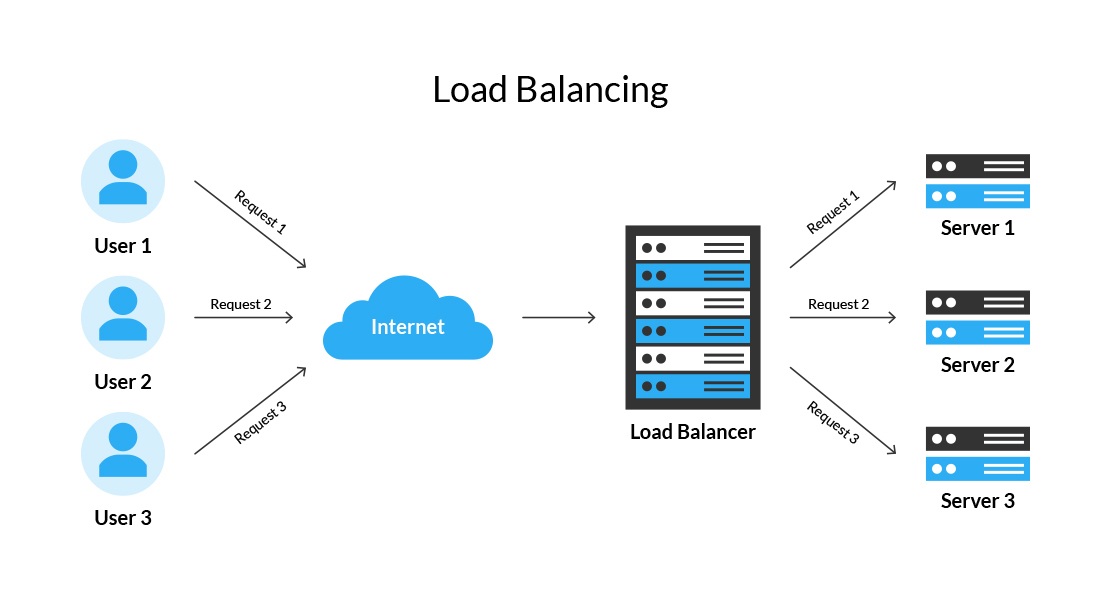
* **Ne İşe Yarar**: Flask gibi framework’lerin dahili sunucuları test içindir ve yavaş kalır. Gunicorn, daha hızlı ve çoklu istekleri aynı anda işler.
* gunicorn --workers 4 app:app
* **Hız**:
  + Basit yanıt: 5-15 ms.
  + Ağ: LAN’da 1-2 ms gecikme.
* **Örnek**:
  + Kod:
* from flask import Flask app = Flask(\_\_name\_\_) @app.route('/post/<id>') def post(id): return {"id": id, "title": "Post " + id}
  + İstek: /post/123 -> {"id": "123", "title": "Post 123"}.

### **5.4. Web Sunucusu: Apache**

* **Web Sunucusu**, HTTP isteklerini işler ve statik/dinamik içeriği istemciye döner. Apache, yaygın kullanılan bir açık kaynaklı sunucudur.
* **Teknik Detaylar**:
  + **MPM (Multi-Processing Module)**:
    - **Prefork**: Her bağlantı için ayrı bir süreç (process).
      * 1000 eşzamanlı bağlantı için 1 GB RAM gerekebilir.
    - **Worker**: Thread’lerle çalışır, daha az bellek kullanır.
  + **İstek İşleme**:
    - GET /index.html -> Dosya sistemi: /var/www/html/index.html.
    - MIME türü: text/html.
  + **Modüller**:
    - mod\_rewrite: URL yeniden yazma.
    - mod\_ssl: HTTPS desteği.
  + **Performans**:
    - 1000 bağlantı için CPU kullanımı: %20-30 (2.4 GHz CPU).
    - Ortalama yanıt süresi: 5-10 ms (SSD disk ile).

### **5.5. Veritabanı: MySQL**

* **Veritabanı**, dinamik verileri depolar ve sorgular. MySQL, ilişkisel bir veritabanı yönetim sistemidir.
* **Teknik Detaylar**:
  + **Yapı**:
    - Tablo: posts id (INT, PK), title (VARCHAR), content (TEXT)
  + **Sorgu**: SELECT \* FROM posts WHERE id = 123;  
    - Index: id üzerinde B-Tree indeks, 10 ms içinde sonuç.
  + **Bağlantı Havuzu**:
    - Maksimum 200 eşzamanlı bağlantı.
    - Connection pooling ile uygulama sunucusundan gelen istekler optimize edilir.
  + **Performans**:
    - 10 ms sorgu süresi: 1 milyon satırlı tabloda, indeksle.
    - Disk: NVMe SSD, 500K IOPS.
    - Bellek: InnoDB Buffer Pool (örn. 4 GB) ile önbellekleme.
  + **Replication**:
    - Master-Slave yapısı: Yazma master’da, okuma slave’de.
  + **Örnek**:
    - Uygulama sunucusu: “Post 123’ü getir.”
    - MySQL: 10 ms’de JSON döner: {"id": 123, "title": "Merhaba Dünya"}



## **6. Tarayıcı Render: Son Dokunuş**

* Sunucu yanıtı tarayıcıya ulaştığında, bu veriler ekrana çizilir. İşte bu sürecin temel adımları, ağla ilişkisi :

### **6.1. Genel Süreç**

* **Ağ Rolü**:
  + Sunucudan HTML, CSS ve JS gelir (IPv4/IPv6, TCP 443).
  + Hız: CDN ile 10-20 ms, normalde 50 ms.
* **Veri**:
  + Örnek:
* HTTP/1.1 200 OK <html><body><div>Merhaba</div></body></html>

### **6.2. Adım Adım Render**

#### **1. HTML Parsing: DOM Oluşumu**

* HTML, **DOM** (Document Object Model) ağacına çevrilir.
* HTML hızlı gelmezse süreç gecikir.
* **Örnek**: <div>Merhaba</div> -> DOM’da bir düğüm.
* **Hız**: 1-2 ms.

#### **2. CSSOM: Stil Ekleme**

* CSS, **CSSOM** (CSS Object Model) oluşturur.
* <link href="/style.css"> için ek istek (10 ms).
* **Örnek**:
* div { color: blue; }
* **Hız**: 1 ms.

#### **3. Render Tree: Görsel Plan**

* DOM ve CSSOM birleşir, görünür öğeler seçilir.
* Resimler gibi kaynaklar için istekler gider.
* **Hız**: 2 ms.

#### **4. Layout: Yerleşim**

* Öğelerin ekran konumu hesaplanır.
* **Örnek**: div { width: 100px; } -> 100x50 piksel kutu.
* **Hız**: 5 ms.

#### **5. Painting: Çizim**

* Pikseller ekrana çizilir (GPU ile).
* Resimler gecikirse boş alan kalır.
* **Hız**: 10 ms (60 FPS).

#### **6. JavaScript: Dinamiklik**

* JS çalışır, DOM’u günceller.
* <script src="/script.js"> için istek (15 ms).
* **Örnek**:
* document.querySelector("div").style.color = "red";
* **Hız**: 5-10 ms.

### **6.3. Ağ ve Performans**

* **Hızlandırma**:
  + CDN: Kaynaklar 10 ms’de gelir.
  + HTTP/2: Paralel indirme.
  + Önbellek: Tekrar istek azalır.