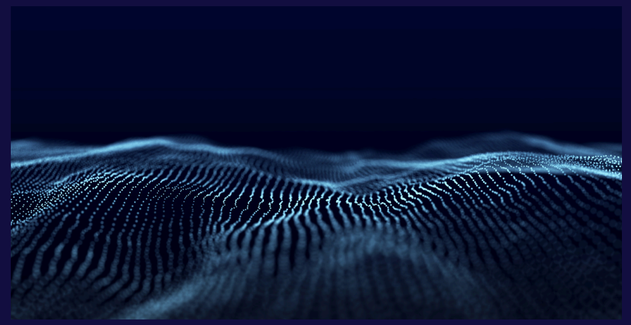
MODEL CONTEXT PROTOCOL (MCP)

Hazırlayan : Eda Nur Arslan

MCP Nedir?

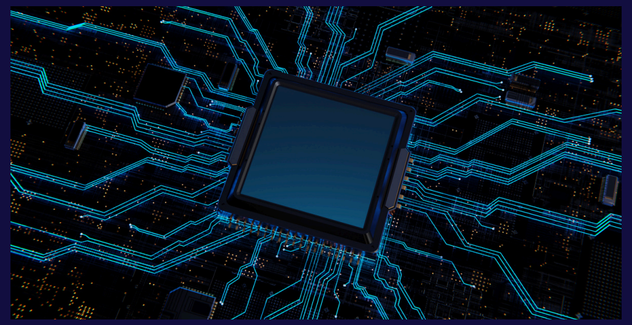
Model Context Protocol (Model Bağlam Protokolü), uygulamaların büyük dil modellerine (LLM'ler) nasıl bağlam sağladığını standartlaştıran açık bir protokoldür. MCP, AI modellerinin farklı veri kaynakları ve araçlarla standart bir şekilde entegre olmasını sağlar.

Neden MCP?



Entegrasyon Kolaylığı

LLM'lerin doğrudan bağlanabileceği, sürekli artan sayıda önceden oluşturulmuş entegrasyonlar sunar.



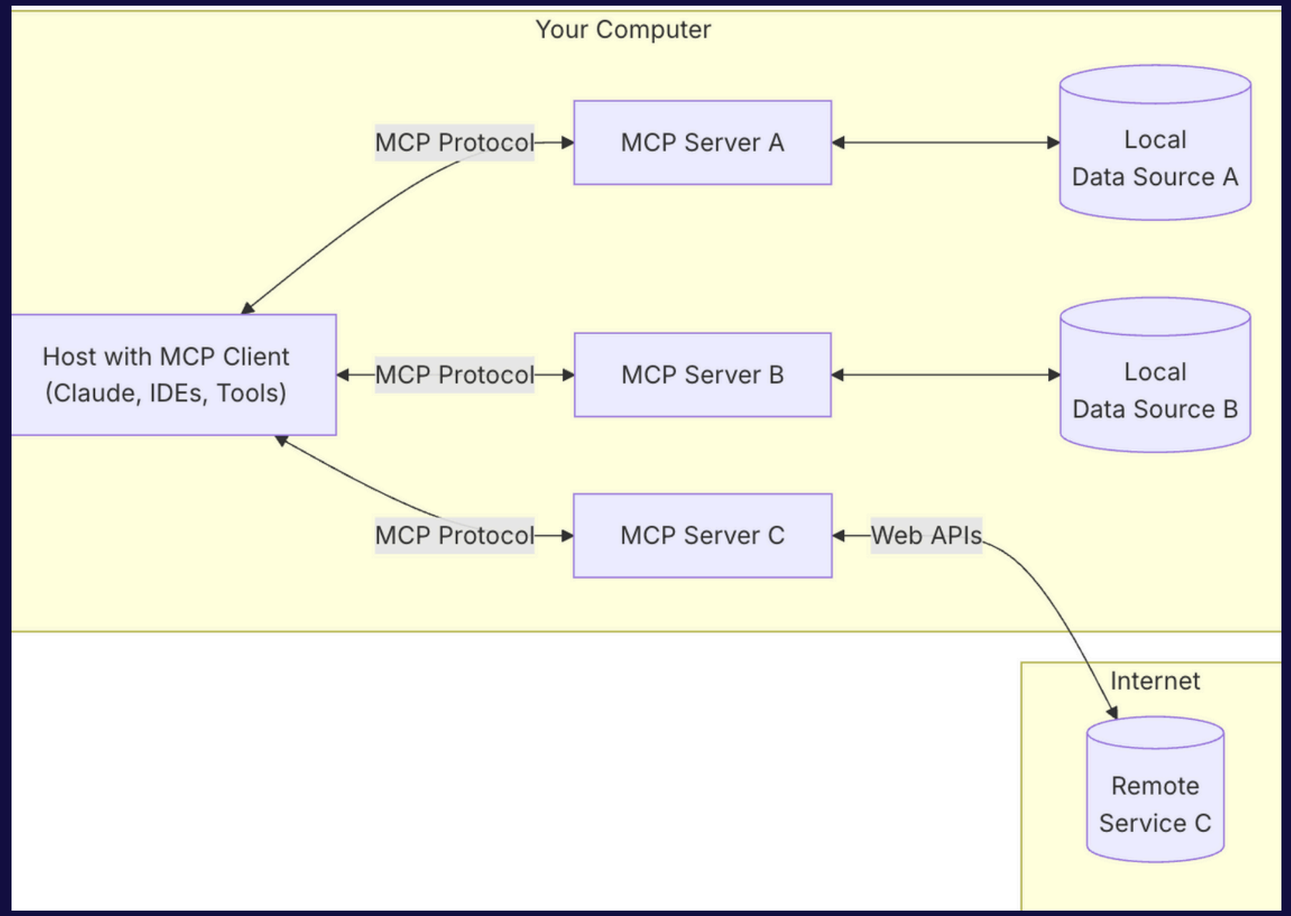
Esneklik

Farklı LLM sağlayıcıları ve satıcıları arasında geçiş yapma esnekliği sağlar.

Güvenlik

Verilerinizi kendi altyapınız içinde güvence altına almak için en iyi uygulamaları içerir.

Genel Mimari



MCP, temel olarak bir istemci-sunucu mimarisi izler ve bir ana uygulama birden fazla sunucuya bağlanabilir:

- **MCP Ana Bilgisayarları:** Claude Desktop, entegre geliştirme ortamları (IDE'ler) veya MCP aracılığıyla verilere erişmek isteyen AI araçları gibi programlar.
- **MCP İstemcileri:** Sunucularla birebir bağlantılar kuran protokol istemcileri.
- **MCP Sunucuları:** Standartlaştırılmış Model Context Protocol aracılığıyla belirli yetenekleri sunan hafif programlar.
- **Yerel Veri Kaynakları:** MCP sunucularının güvenli bir şekilde erişebileceği bilgisayarınızdaki dosyalar, veritabanları ve hizmetler.
- **Uzak Hizmetler:** MCP sunucularının bağlanabileceği, internet üzerinden erişilebilen harici sistemler (örneğin, API'ler).

Çekirdek Mimarisi

Model Context Protokolü (MCP), büyük dil modelleri (LLM'ler) ile harici veri kaynakları ve araçları arasında standart bir iletişim çerçevesi sunan açık bir protokoldür. Bu sayede, yapay zeka uygulamalarının farklı veri kaynaklarına güvenli ve verimli bir şekilde erişmesi mümkün hale gelir.

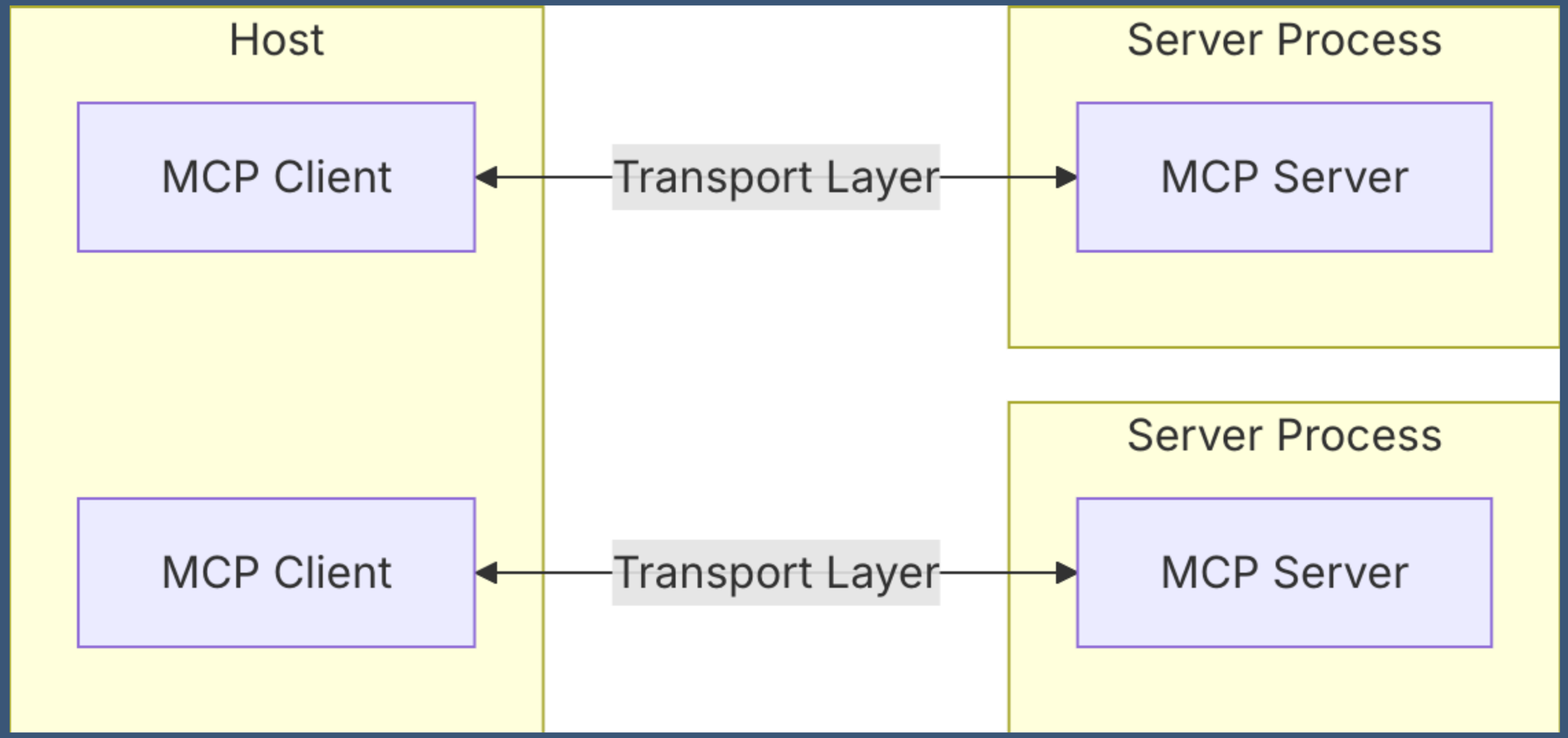
MCP, istemci-sunucu modelini takip eder ve bu yapıda üç ana bileşen bulunur:

1. **Hostlar:** LLM uygulamalarıdır ve bağlantıları başlatan taraflardır. Örneğin, Claude Desktop veya entegre geliştirme ortamları (IDE'ler) gibi uygulamalar host olarak görev yapar.

2. **İstemciler:** Host uygulama içinde, sunucularla birebir bağlantıları yöneten bileşenlerdir. İstemciler, sunucularla iletişimi sağlayarak gerekli veri ve araçların entegrasyonunu gerçekleştirir.

3. **Sunucular:** İstemcilere bağlam (context), araçlar ve komut şablonları (prompts) sağlayan harici hizmetlerdir. Sunucular, LLM uygulamalarının ihtiyaç duyduğu ek bilgileri ve fonksiyonları temin eder.

Çekirdek Mimarisi



MCP Katmanları

Protokol Katmanı

Mesaj çerçeveleme, istek/yanıt ilişkilendirmesi ve yüksek seviyeli iletişim desenlerini yönetir. Bu katman, istemci ve sunucu arasındaki iletişimin düzenli ve uyumlu olmasını sağlar.

Taşıma Katmanı

İstemci ve sunucu arasındaki gerçek veri iletimini gerçekleştirir. MCP, çeşitli taşıma mekanizmalarını destekler:

- **Stdio Taşıma:** Yerel süreçler için standart giriş/çıkış kullanarak iletişim kurar.
- **HTTP ve SSE Taşıma:** Sunucudan istemciye mesajlar için Server-Sent Events (SSE) ve istemciden sunucuya mesajlar için HTTP POST yöntemini kullanır.

Mesaj Türleri

MCP'de dört ana mesaj türü bulunmaktadır:

İstekler (Requests)

Karşı taraftan yanıt bekleyen mesajlardır.

```
interface Request {
  method: string;
  params?: { ... };
}
```

Sonuçlar (Results)

Başarılı isteklerin yanıtlarıdır.

```
interface Result {
  [key: string]: unknown;
}
```

Hatalar (Errors)

Başarısız isteklerin nedenlerini belirtir.

```
interface Error {
  code: number;
  message: string;
  data?: unknown;
}
```

Bildirimler (Notifications)

Yanıt beklemeyen tek yönlü mesajlardır.

```
interface Notification {
  method: string;
  params?: { ... };
}
```

Bağlantı Döngüleri

Başlatma (Initialization):
İstemci, protokol sürümü ve yeteneklerini belirten bir initialize isteği gönderir. Sunucu, kendi desteklediği sürüm ve yeteneklerle yanıt verir. İstemci, initialized bildirimi ile süreci tamamlar ve normal mesaj alışverişi başlar.

Mesaj Alışverişi (Message Exchange):
Başlatma sonrası, istemci ve sunucu arasında istek-yanıt ve bildirim mesajları değiş tokuş edilir.

Sonlandırma (Termination):
Taraflardan biri close() yöntemiyle bağlantıyı temiz bir şekilde sonlandırabilir, taşıma katmanında bir kopma yaşanabilir veya hata durumları nedeniyle bağlantı kesilebilir.

Hata Yönetimi

Hata yönetimi için MCP, standart JSON-RPC hata kodlarını kullanır ve ek uygulama spesifik hata kodları tanımlanabilir. Hatalar, istek yanıtları, taşıma katmanı olayları veya protokol seviyesindeki hata işleyicileri aracılığıyla iletilir.
MCP'nin bu mimarisi, LLM uygulamalarının harici sistemlerle etkili ve uyumlu bir şekilde çalışmasını sağlayarak, geniş bir uygulama yelpazesinde entegrasyonu kolaylaştırır.

Güvenlik ve Yetkilendirme

MCP, kullanıcıların veri erişimi ve işlemleri üzerinde tam kontrol sahibi olmasını sağlar. Kullanıcıların açık rızası olmadan veri paylaşımı veya işlem yapılması engellenir. Ayrıca, araçların güvenli kullanımı ve LLM örnekleme kontrolleri için katı güvenlik ilkeleri benimsenmiştir.

Dinlediğiniz İçin **Teşekkürler !**

Hazırlayan : Eda Nur Arslan