1. Giriş (Introduction)

Geleneksel Kimlik Doğrulama Modeli ve Problemleri

- **Geleneksel Model:**
  İstemci, sunucudaki erişim kısıtlanmış bir kaynağa (protected resource) ulaşmak için kaynak sahibinin kimlik bilgilerini (örneğin, kullanıcı adı ve şifre) kullanır.
- **Üçüncü Taraf Uygulamalar:**
  Kaynak sahibi, üçüncü taraf bir uygulamaya erişim izni vermek istediğinde, kimlik bilgilerini (örneğin şifre) bu uygulama ile paylaşmak zorunda kalır.
  Ancak bu yaklaşım aşağıdaki sorunları doğurur:
  1. **Şifre Saklama Problemi:**
     Üçüncü taraf uygulamaların, kaynak sahibinin kimlik bilgilerini (genelde düz metin olarak) saklaması gerekir. Bu, güvenlik riski yaratır.
  2. **Şifre Doğrulama Zorunluluğu:**
     Sunucular, güvenlik açıkları olan şifre doğrulama mekanizmasını desteklemek zorunda kalır.
  3. **Gereksiz Geniş Yetki:**
     Üçüncü taraf uygulamalar, kaynak sahibinin tüm korunan kaynaklarına geniş erişim yetkisi alır. Kaynak sahibi, erişimi süre veya belirli kaynaklarla sınırlayamaz.
  4. **Erişim Kısıtlama Eksikliği:**
     Kaynak sahibi, belirli bir üçüncü taraf uygulamanın erişimini iptal edemez. Erişim kaldırmak için şifresini değiştirmek zorunda kalır, bu da tüm uygulamaların erişimini kaldırır.
  5. **Güvenlik Riski:**
     Üçüncü taraf uygulamalardan birinin güvenliği ihlal edilirse, kullanıcının şifresi ve şifre ile korunan tüm veriler tehlikeye girer.

OAuth ile Gelen Çözüm

OAuth, yukarıdaki sorunları çözmek için bir **yetkilendirme katmanı** (authorization layer) sunar ve istemciyi kaynak sahibinden ayırır:

1. **Erişim Token'ları:**
   Kaynak sahibinin kimlik bilgileri yerine, istemciye bir **erişim token'ı** (access token) verilir. Bu token:
   - Belirli bir erişim kapsamı (scope),
   - Belirli bir ömür süresi (lifetime),
   - Diğer erişim özelliklerini belirtir.
2. **Yetkilendirme Sunucusu:**
   - Erişim token'ları, kaynak sahibinin onayı ile bir **yetkilendirme sunucusu (authorization server)** tarafından oluşturulur.
   - İstemci, bu token'ı kullanarak kaynak sunucusundaki (resource server) korunan kaynaklara erişir.

Bir Örnek Senaryo

- **Senaryonun Tarafları:**
    - **Kaynak Sahibi (End-user):** Fotoğrafların sahibi.
    - **İstemci (Client):** Baskı hizmeti sağlayıcısı.
    - **Kaynak Sunucusu (Resource Server):** Fotoğraf paylaşım hizmeti.
    - **Yetkilendirme Sunucusu:** Fotoğraf paylaşım hizmetinin güvendiği bir kimlik doğrulama sunucusu.
- **Nasıl Çalışır:**
    - Kullanıcı, baskı hizmetine fotoğraflarına erişim izni verir.
    - Kullanıcı, doğrudan fotoğraf paylaşım hizmetinin güvendiği yetkilendirme sunucusunda kimlik doğrulaması yapar.
    - Yetkilendirme sunucusu, baskı hizmetine belirli bir erişim yetkisi veren bir erişim token'ı (delegation-specific access token) sağlar.
    - Baskı hizmeti, bu token'ı kullanarak kullanıcı fotoğraflarına erişir. Kullanıcının şifresi hiçbir zaman baskı hizmetine verilmez.

OAuth 2.0 ve HTTP

- **Protokol Uyumlu:**
  OAuth 2.0, HTTP protokolü (RFC 2616) ile uyumlu olacak şekilde tasarlanmıştır.
  HTTP dışında başka bir protokolde OAuth kullanımı bu dokümanın kapsamı dışındadır.

OAuth 1.0 ve OAuth 2.0

1. **Geçmiş:**
   OAuth 1.0 protokolü, küçük bir topluluğun çabalarıyla oluşturulmuş bir bilgi dokümanıdır (RFC 5849).
2. **Deneyim ve Gelişim:**
   OAuth 2.0, OAuth 1.0'ın uygulama deneyimlerinden ve daha geniş bir IETF topluluğunun ek gereksinimlerinden yola çıkarak geliştirilmiştir.
3. **Geriye Dönük Uyum (Backward Compatibility):**
    - OAuth 2.0, OAuth 1.0 ile **geriye dönük uyumlu değildir.**
    - İki protokol aynı ağda birlikte çalışabilir, ancak OAuth 2.0 yeni uygulamalar için tavsiye edilir.
    - OAuth 1.0 yalnızca mevcut uygulamaları desteklemek için kullanılmalıdır.
4. **Farklı Yapılar:**
   OAuth 2.0, OAuth 1.0 ile çok az uygulama detayı paylaşır. OAuth 1.0'ı bilenlerin, bu dokümanı **ön yargısız** bir şekilde incelemeleri önerilir.

Öne Çıkan Detaylar

1. OAuth 2.0, istemcinin kaynak sahibinin kimlik bilgilerine erişmeden, güvenli ve sınırlı bir şekilde kaynaklara erişmesini sağlar.
2. Yetkilendirme ve erişim token’ları aracılığıyla erişim daha güvenli, esnek ve sınırlı hale getirilmiştir.
3. OAuth 2.0, önceki versiyonun eksiklerini kapatmak ve modern ihtiyaçlara cevap vermek için yeniden tasarlanmıştır.