**3.1. Authorization Endpoint** bölümü, OAuth 2.0 protokolünün **yetkilendirme uç noktası** (authorization endpoint) ile ilgili gereksinimleri açıklar. Yetkilendirme uç noktası, istemcinin kullanıcıdan (kaynak sahibi) **yetkilendirme izni** almak amacıyla kullandığı bir kaynaktır. Bu uç nokta, OAuth 2.0 protokolünün temel taşlarından biridir ve aşağıdaki önemli unsurları içerir:

## 1. Yetkilendirme Uç Noktasının Amacı

- **Yetkilendirme grant'ı alma:** Yetkilendirme uç noktası, istemcinin **yetkilendirme grant'ı** almak için kullanılır. Bu grant, istemcinin kaynağa (örneğin, API'ye) erişim izni almasını sağlar. Kullanıcı, yetkilendirme işlemi sırasında istemcinin belirttiği yetkilendirme uç noktasına yönlendirilir ve burada onay vermesi beklenir.

## 2. Kaynak Sahibinin Kimlik Doğrulaması

- **Kimlik doğrulama gereksinimi:** Yetkilendirme sunucusu, kaynak sahibinin kimliğini doğrulamalıdır. Kimlik doğrulama işlemi (örneğin, kullanıcı adı ve şifre girişi veya oturum çerezleri) bu spesifikasyonun dışında olsa da, bu adım **gereklidir**.
- Kimlik doğrulama işlemi, kullanıcının doğru olduğuna ve kaynak sahibinin izin verme yetkisine sahip olduğuna emin olmak için yapılır.

## 3. Yetkilendirme Uç Noktasının Konumu

- **Uç nokta konumunun belirlenmesi:** İstemci, yetkilendirme uç noktasının yerini öğrenmek için belirli yöntemler kullanır. Bu spesifikasyon, bu yöntemi belirtmez ancak genellikle uç nokta konumu, hizmetin dokümantasyonunda sağlanır.

## 4. Authorization Endpoint URI Yapısı

- **Query Parametreleri ve Formülasyonu:** Yetkilendirme uç noktasının URI'si, **"application/x-www-form-urlencoded"** formatında sorgu parametreleri içerebilir. Bu parametreler, RFC3986'ya uygun şekilde eklenmelidir ve parametrelerin sırası değiştirilmemelidir. Ancak **fragment** (örn. # işareti ile başlayan kısımlar) kullanımı yasaktır.
- **Tekrar eden parametreler:** Aynı parametre, istekte birden fazla kez gönderilemez. Eğer bir parametre eksikse, sunucu bu durumu, parametrenin yok sayılması olarak kabul etmelidir.

## 5. Güvenli İletişim Gereksinimi

- **TLS Kullanımı:** Yetkilendirme uç noktasına yapılan talepler, **kullanıcı kimlik bilgilerini** ve **açık metin verileri** ilettiğinden, bu verilerin güvenli bir şekilde iletilmesi önemlidir. Bu nedenle, **TLS (Transport Layer Security)** protokolü kullanımı **zorunludur**. Yani, HTTPS üzerinden iletişim yapılmalıdır.
- **Veri Güvenliği:** Bu gereklilik, kullanıcının kimlik bilgilerini korumak ve olası kötü niyetli saldırıları engellemek için gereklidir.

## 6. HTTP Yöntemleri: GET ve POST

- **GET Yöntemi:** Yetkilendirme uç noktası, en azından **HTTP GET** yöntemini desteklemelidir. Bu yöntem, genellikle URL üzerinden sorgu parametreleri gönderilmesini sağlar.

- **POST Yöntemi:** Yetkilendirme uç noktası **POST** yöntemini de destekleyebilir, ancak bu zorunlu değildir. **POST**, genellikle daha büyük veri miktarlarının gönderilmesi gerektiğinde tercih edilen bir yöntemdir.

## 7. Tanımadık Parametreler ve Değerler

- **Tanımlanmayan parametreler:** İstemciden gelen tanınmayan parametreler, yetkilendirme sunucusu tarafından **yok sayılmalıdır**. Sunucu bu parametreleri dikkate almaz.
- **Parametre Değerlerinin Yok Sayılması:** Parametrelerin değerleri eksikse, yetkilendirme sunucusu bu parametreleri **omiş** olarak kabul etmelidir.

## 8. Tekrar Eden Parametreler

- **Bir parametre birden fazla kez gönderilmemelidir:** Herhangi bir parametre birden fazla kez istemci tarafından gönderilmemelidir. Eğer böyle bir durum söz konusuysa, parametrelerin her birini tek seferde işlemesi gereklidir.

---

## Özet:

- Yetkilendirme uç noktası, istemcinin kaynağa erişim izni almak için kullandığı ana kaynaktır.
- Sunucu, kullanıcı kimliğini doğrulamalıdır.
- Yetkilendirme uç noktasına yapılan talepler, **TLS** üzerinden güvenli bir şekilde iletilmelidir.
- İstemci, **GET** ve gerekirse **POST** yöntemlerini kullanabilir.
- **Query parametreleri**, belirli bir formata uygun şekilde iletilmeli, tekrar edilmemeli ve sunucu tanımadığı parametreleri göz ardı etmelidir.

Bu uç nokta, OAuth 2.0 protokolünde kullanıcının kimliğini doğrulamak ve ona erişim izni sağlamak amacıyla kritik bir rol oynar.

**3.1.1. Response Type** bölümü, **authorization endpoint** (yetkilendirme uç noktası) üzerinden gönderilen taleplerin nasıl yanıtlanacağına dair önemli bir parametreyi açıklar. Bu parametre, istemcinin hangi tür **yetkilendirme yanıtını** istediğini belirtir. İstemci, **authorization server** (yetkilendirme sunucusuna) bu parametreyi kullanarak hangi tür **grant** (izin) tipini talep ettiğini bildirir. Bu bölümde açıklanan "response_type" parametresi, OAuth 2.0 protokolündeki farklı yetkilendirme akışlarını belirler. Aşağıda **response_type** parametresinin nasıl çalıştığı ve olası değerleri açıklanmıştır:

## 1. response_type Parametresi

- **Zorunlu Parametre:** "response_type" parametresi, yetkilendirme isteği gönderilirken **zorunlu** bir parametredir. Bu parametre, istemcinin hangi tür **yetkilendirme yanıtını** beklediğini belirtir.
- İstemci, yetkilendirme sunucusuna hangi tür yanıt almak istediğini bildirir ve sunucu buna göre uygun bir işlem yapar.

## 2. response_type Değerleri

İstemcinin talep ettiği yanıt türüne göre, **response_type** parametresi farklı değerler alabilir. Bu değerler, istemcinin yetkilendirme sunucusundan alacağı **yetkilendirme grant'ı** türünü belirler.

### a. code (Authorization Code Grant)

- **Açıklama:** Bu, **authorization code grant** türü için kullanılır. Bu, OAuth 2.0'ın en yaygın akışlarından biridir. İstemci, yetkilendirme sunucusundan bir **authorization code** (yetkilendirme kodu) almak ister. Bu kod, daha sonra istemci tarafından **token endpoint**'e gönderilir ve bir **access token** (erişim jetonu) alınır.
- **Ne Zaman Kullanılır?**: Bu yanıt türü, istemcinin güvenli bir şekilde yetkilendirme kodunu almasını isteyen istemciler için kullanılır. Bu tip genellikle web uygulamaları gibi **gizlilik** gereksinimi yüksek olan istemciler tarafından kullanılır.

### b. token (Implicit Grant)

- **Açıklama:** Bu, **implicit grant** türü için kullanılır. İstemci doğrudan bir **access token** (erişim jetonu) almak ister, ancak authorization code alınmaz. Bu tür genellikle daha az güvenlik gereksinimi olan istemciler için uygundur (örneğin, tarayıcı tabanlı uygulamalar veya tek sayfalık uygulamalar - SPA).
- **Ne Zaman Kullanılır?**: Bu, istemcinin daha hızlı bir şekilde erişim sağlamak istediği, ancak daha düşük güvenlik önlemleri gerektiren uygulamalarda kullanılır.

### c. Extension Values (Uzantı Değerleri)

- **Açıklama:** OAuth 2.0 protokolüne ek olarak bazı özel uzantılar (extension grant types) olabilir. Bu uzantı türleri, **response_type** parametresinde özel bir değer alabilirler. Bu uzantılar, OAuth 2.0'ın dışında olan ancak OAuth 2.0 uyumlu sistemlere özgü özellikler sunabilir.
- **Örnek:** "a b" gibi bir **response_type** değeri olabilir, burada **"a"** ve **"b"** değerleri uzantı yanıt türlerini temsil eder ve sıralama fark etmez. Örneğin, bir OAuth uzantısı, her iki türdeki yanıtları aynı anda almayı mümkün kılabilir.
- **Ne Zaman Kullanılır?**: Özel uygulama gereksinimleri veya ek OAuth 2.0 uzantıları kullanıldığında.

## 3. response_type Parametresinin Eksik Olması veya Anlaşılmaması Durumu

- **Eksik veya Tanınmayan response_type:** Eğer istemci, **response_type** parametresini göndermemişse veya gönderdiği değer yetkilendirme sunucusu tarafından anlaşılmıyorsa, yetkilendirme sunucusu bir **hata yanıtı** döndürmelidir. Bu hata yanıtı, **Section 4.1.2.1**'de açıklanan şekilde yapılandırılmalıdır.
    - **Hata Durumu:** Yetkilendirme sunucusu, beklenen "response_type" değerini alamazsa veya eksikse, bu durumda **400 Bad Request** gibi bir hata dönebilir ve istemciye uygun hata mesajını iletebilir.

## 4. Uzantı Yanıt Türlerinin (Extension Response Types) Kullanımı

- Uzantı yanıt türleri, **response_type** parametresine bir **boşluk ile ayrılmış bir dizi değer** ekleyebilir. Bu tür yanıtların anlamı, ilgili uzantı spesifikasyonları tarafından tanımlanır.

- **Örnek:** "a b" ve "b a" yanıt türleri eşdeğerdir; sıralama önemli değildir.

---

## Özet:

- **response_type** parametresi, istemcinin yetkilendirme uç noktasına yaptığı isteklerde hangi tür yanıt beklediğini belirtir.
- Bu parametre, **"code"** (authorization code grant) veya **"token"** (implicit grant) gibi değerler alabilir.
- Uzantı yanıt türleri, belirli OAuth 2.0 uzantıları tarafından tanımlanmış özel değerler olabilir.
- Eğer **response_type** parametresi eksik ya da geçersizse, yetkilendirme sunucusu hata mesajı döndürmelidir.

Bu parametre, istemci ile yetkilendirme sunucusu arasındaki yetkilendirme akışının doğru şekilde başlatılmasını sağlar.

**3.1.2. Redirection Endpoint** bölümü, OAuth 2.0 protokolündeki bir **redirection endpoint** (yönlendirme uç noktası) kullanımıyla ilgilidir. Yönlendirme uç noktası, yetkilendirme süreci sonunda kullanıcının (kaynak sahibi) onayı alındığında istemciye geri dönmesini sağlar. İstemci, yetkilendirme sunucusuyla etkileşim tamamlandığında, kullanıcıyı bu uç noktaya yönlendirir. Bu bölümdeki açıklamalar, yönlendirme uç noktasının nasıl yapılandırılacağına ve nasıl çalışması gerektiğine dair ayrıntıları sunar.

## 1. Redirection Endpoint'ın Rolü

- Yetkilendirme süreci tamamlandıktan sonra, **authorization server** (yetkilendirme sunucusu), kaynak sahibinin **user-agent** (kullanıcı aracısı, yani tarayıcı) aracılığıyla istemciye geri yönlendirme yapar.
- **Yönlendirme uç noktası**, istemcinin, yetkilendirme sunucusuyla yapılan etkileşimden sonra alacağı yanıtı alacağı yerdir. Bu uç nokta, istemcinin kaydolduğu ve yetkilendirme isteği sırasında belirttiği yerdir.
- Yönlendirme uç noktası, istemciye dönüş için belirlenen bir **URI**'dir (Uniform Resource Identifier), ve bu URI, yetkilendirme isteği sırasında önceden belirlenmiş olmalıdır.

## 2. Redirection Endpoint URI

- **Mutlaka Tam URI Olmalı:** Yönlendirme uç noktası bir **absolute URI** olmalıdır. Bu, URI'nin başında **şema (örn. http://)**, **host** (sunucu adı) ve **path** (yol) gibi bileşenlerin yer alması gerektiği anlamına gelir.
    - **RFC 3986**'ya göre, bu bir tam URI olmalıdır ve bu RFC'de tanımlanan kurallara uymalıdır.
- **Query Bileşeni:** Yönlendirme uç noktası URI'si, bir **"application/x-www-form-urlencoded"** biçiminde query parametreleri içerebilir. Bu query parametreleri, istemcinin yetkilendirme sunucusundan dönen bilgileri taşır.
    - Bu query bileşeni, URI'ye ek parametreler eklenmeden önce tutulmalı, yani önceden var olan parametrelerin kaybolmaması sağlanmalıdır.
    - Parametreler **Appendix B**'de tanımlandığı şekilde şifrelenmiş formda olabilir.

- **Fragment İçermemeli:** Yönlendirme URI'si **fragment** bileşeni içeremez. **Fragment** kısmı, URI'nin parçası olarak sadece istemci tarafında işlenen bir bölümdür ve yetkilendirme sürecinde herhangi bir sunucu tarafı işlemi için kullanılmaz. Bu yüzden, OAuth 2.0'da yönlendirme URI'leri fragment içermemelidir.

## 3. Yönlendirme Endpoint'inin İşlevi ve Kısıtlamalar

- **İstemci Kaydı ve Yetkilendirme İsteği:** Yönlendirme uç noktası, istemcinin **client registration** (istemci kaydı) sırasında belirlediği bir URI'dir. Yetkilendirme isteği yapılırken, istemci bu URI'yi belirtir.
    - Örneğin, bir istemci kaydolurken, "http://client.example.com/callback" gibi bir yönlendirme URI'si belirlerse, yetkilendirme sunucusu bu URI'yi kullanarak kullanıcıyı geri gönderir.
- **Kullanıcı-Agent (Tarayıcı) ile Etkileşim:** Yönlendirme, kullanıcıyı tekrar istemciye yönlendirmek için kullanıcı aracısı (tarayıcı) üzerinden yapılır. Yetkilendirme sunucusu, istemciye gerekli erişim izinlerini verdikten sonra, kullanıcının tarayıcısını bu URI'ye yönlendirir.

---

## Özet:

- **Redirection endpoint** (yönlendirme uç noktası), kaynak sahibinin (kullanıcının) yetkilendirme işlemini tamamladıktan sonra istemciye geri gönderileceği yerdir.
- Yönlendirme URI'si **mutlaka tam bir URI** olmalıdır ve **fragment** içeremez.
- Yönlendirme URI'sine eklenen query parametreler **"application/x-www-form-urlencoded"** biçiminde olmalı ve daha sonra ek parametreler eklenmeden önce korunmalıdır.
- Bu uç nokta, istemcinin kaydolduğu ve yetkilendirme sırasında belirttiği yerdir.

Bu bölüm, istemcilerin ve yetkilendirme sunucularının **OAuth 2.0 akışları** sırasında güvenli ve uyumlu bir şekilde nasıl iletişim kurması gerektiğini tanımlar.

**3.1.2.1. Endpoint Request Confidentiality** bölümü, OAuth 2.0 protokolünde **yönlendirme uç noktasına** yapılan isteklerin güvenliği ile ilgilidir. Bu bölüm, özellikle kullanıcıya ait hassas bilgilerin iletildiği durumlarda güvenlik önlemleri alınmasını vurgular.

## 1. TLS Kullanımı ve Öneri

- **TLS (Transport Layer Security)**, verilerin internet üzerinden güvenli bir şekilde iletilmesini sağlayan bir şifreleme protokolüdür. Bu bölümde, **TLS**'nin kullanımı teşvik edilir.
    - **TLS Kullanımı Gereklidir:** Eğer istemci, **"code"** (yetkilendirme kodu) veya **"token"** (erişim belirteci) gibi yanıt türlerini talep ediyorsa veya yönlendirme isteği, **açık ağda hassas kimlik bilgileri** iletilecekse, **TLS** kullanılması **ŞARTTIR**.
    - **Hassas Bilgilerle İletişim:** Kullanıcı adı, şifre, yetkilendirme kodu veya erişim belirteci gibi hassas veriler açık ağlarda iletildiğinde, bu verilerin **gizliliği ve bütünlüğü** bozulabilir. TLS bu verileri şifreleyerek güvenliği artırır.

## 2. TLS Zorunluluğunun Uygulanmaması

- Bu spesifikasyon, **TLS'nin zorunlu hale getirilmesini** istememektedir çünkü o dönemde **TLS'i uygulamak** çoğu istemci geliştiricisi için büyük bir engel olabilir. Yani, bazı istemcilerin TLS desteği olmayabilir veya kullanımı zorlu olabilir.
    - Ancak, **TLS'nin sağlanamadığı durumlarda**, yetkilendirme sunucusu, kaynak sahibine (kullanıcıya) **güvensiz uç nokta** hakkında bir uyarı mesajı göstermelidir. Bu, kullanıcıyı güvenlik riski hakkında bilgilendirmek için yapılır.

      Örneğin, kullanıcı bir üçüncü parti giriş hizmeti (third-party sign-in) kullanıyorsa, yetkilendirme sunucusu kullanıcıya şu uyarıyı gösterebilir: "Bu bağlantı güvenli değil. Kimlik bilgilerinizi risk altında bırakabilirsiniz."

## 3. TLS Kullanımının Güvenlik Üzerindeki Etkisi

- **TLS'in eksikliği, istemci ve korunan kaynaklar üzerinde ciddi güvenlik etkileri yaratabilir.** Yönlendirme uç noktasındaki isteklerin açık ağda iletilmesi, hassas verilerin saldırganlar tarafından ele geçirilmesine, değiştirilmesine veya kötüye kullanılmasına neden olabilir. Bu, OAuth 2.0'ın temel güvenlik garantilerinden biri olan **kimlik doğrulama** ve **yetkilendirme** süreçlerini zayıflatabilir.
- Özellikle OAuth 2.0 kullanılarak gerçekleştirilen **üçüncü taraf kimlik doğrulaması** (third-party sign-in) gibi durumlarda, **TLS kullanımı kritik** hale gelir. Çünkü bu senaryolarda, kullanıcı oturumu genellikle bir üçüncü parti uygulama tarafından devralınır ve kimlik bilgileri (örneğin, sosyal medya hesapları veya banka hesapları) çok daha hassas hale gelir.

## 4. Özet

- **TLS kullanımı önerilir** ve genellikle "code" veya "token" yanıt türleri talep edildiğinde, veya hassas bilgilerin iletilmesi bekleniyorsa zorunludur.
- Eğer **TLS sağlanamıyorsa**, yetkilendirme sunucusu kullanıcıyı **güvensiz uç nokta hakkında uyar** yapmalıdır.
- **Güvenlik** açısından, TLS, özellikle **üçüncü parti kimlik doğrulaması** gibi durumlarda büyük önem taşır. Bu güvenlik protokolü, **hassas bilgilerin korunması** ve **veri iletiminin gizliliği** için gereklidir.

Bu bölüm, güvenli veri iletiminin önemini vurgular ve yetkilendirme sürecinin güvenli bir şekilde yapılması için TLS kullanılması gerektiğini belirtir.

**3.1.2.2. Registration Requirements** bölümü, **OAuth 2.0**'da **yönlendirme uç noktası (redirection endpoint)** kayıt sürecini tanımlar. Bu kayıt, güvenliği sağlamak ve potansiyel saldırılara karşı korunmak için gereklidir. Yönlendirme uç noktası, kullanıcının yetkilendirme işlemi tamamlandığında OAuth sunucusu tarafından geri gönderileceği URI'dir. Bu bölümde, yönlendirme uç noktalarının nasıl kaydedileceği ve hangi tür istemcilerin kaydettirilmesi gerektiği açıklanır.

## 1. Kayıt Zorunluluğu Olan İstemciler

Yetkilendirme sunucusu, aşağıdaki istemcilerin **yönlendirme uç noktasını** kaydettirmelerini **zorunlu kılmalıdır**:

- **Public clients (Açık istemciler)**: Bu istemciler, istemci kimlik bilgilerini gizli tutamazlar, yani uygulama kodu kullanıcının cihazında çalışır ve kimlik bilgileri korunmaz. Bu nedenle, bu istemcilerin yönlendirme uç noktalarının kaydedilmesi gereklidir.
- **Confidential clients utilizing the implicit grant type (Yetkilendirme kodu kullanılmayan gizli istemciler)**: İstemci, güvenli kimlik doğrulama yapabilen, fakat açık istemciler gibi kimlik bilgilerini riske atan istemcilere kıyasla daha güvenlidir. Ancak, **implict grant type** kullanarak erişim belirteci elde etmeyi amaçlayan istemciler için de yönlendirme uç noktası kaydı gereklidir.

## 2. Tüm İstemcilerin Kayıt Zorunluluğu

- Yetkilendirme sunucusu, **tüm istemcilerin** yönlendirme uç noktasını, **yetkilendirme uç noktasını** kullanmadan önce kaydetmelerini **önerir**. Bu, yalnızca güvenli istemciler için değil, tüm istemciler için önemlidir.
- Yetkilendirme sunucusunun, istemcilerden kaydetmeleri gereken yönlendirme URI'yi **tam** olarak talep etmesi gerekir. Yani istemci sadece bir kısmı değil, URI'nin tamamını kaydetmelidir. Ancak, bu zorunlu değilse, sunucu, istemcinin yalnızca **URI şeması**, **yetki kısmı** ve **yol kısmı** (path) gibi temel bileşenleri kaydetmesini isteyebilir. Bu durumda istemci, sorgu bileşenini (query component) dinamik olarak değiştirebilir.

## 3. Birden Fazla Yönlendirme Uç Noktasının Kaydı

- Yetkilendirme sunucusu, istemcinin birden fazla yönlendirme uç noktası kaydetmesine **izin verebilir**. Bu, istemcinin birden fazla yetkilendirme uç noktası kullanabileceği anlamına gelir.

## 4. Yönlendirme Uç Noktası Kaydının Olmaması Riskleri

- Eğer yönlendirme URI'sinin kaydedilmesi zorunlu kılınmazsa, kötü niyetli bir saldırgan **açık yönlendirme** (open redirector) saldırısı yapabilir. Bu tür saldırılar, yetkilendirme uç noktasının **başka bir siteye yönlendirme yapacak şekilde manipüle edilmesi** ile gerçekleştirilir. Bu saldırılar, yetkilendirme sunucusunun istemcinin doğruluğunu doğru şekilde kontrol etmemesinden faydalanır.

  Örneğin:

  - **Açık yönlendirme** durumu şu şekilde gerçekleşebilir: Eğer istemci tarafından kayıtlı olmayan bir yönlendirme uç noktası, kötü niyetli bir siteye yönlendiriliyorsa, kullanıcı oturum açtıktan sonra, kimlik bilgileri yanlış bir siteye yönlendirilebilir ve saldırgan bu bilgileri ele geçirebilir.

## 5. Özetle

- **Public** ve **Confidential (implicit grant)** istemcilerinin yönlendirme uç noktalarını kaydetmeleri gereklidir.
- Tüm istemcilerin yönlendirme uç noktalarını kaydetmesi önerilir.
- Yönlendirme URI'sinin **tam** olarak kaydedilmesi önemlidir; yalnızca şema, yetki ve yol kısmı kaydedilerek, istemci sorgu parametrelerini dinamik olarak değiştirebilir.
- Yetkilendirme sunucusu, birden fazla yönlendirme uç noktasının kaydedilmesine izin verebilir.

- **Açık yönlendirme** (open redirector) saldırılarından korunmak için yönlendirme URI'sinin kaydedilmesi çok önemlidir.

Bu bölüm, istemci güvenliğini sağlamak ve kötüye kullanımı engellemek için doğru yönlendirme uç noktası kaydının yapılmasının önemini vurgular.

**3.1.2.3. Dynamic Configuration** bölümü, **OAuth 2.0** protokolünde istemcilerin yönlendirme URI'leri (redirection URIs) ile nasıl çalıştığını ve yetkilendirme sunucusunun bu URI'lerle nasıl etkileşimde bulunduğunu açıklamaktadır. Bu bölüm, istemcilerin yönlendirme URI'lerini dinamik bir şekilde kullanabilmelerini sağlar ve ayrıca, istemcinin kayıtlı URI'ler ile uyumsuzluk durumunda nasıl bir işlem yapılacağını belirtir.

## 1. Yönlendirme URI'lerinin Kayıt Durumu

- Eğer istemci birden fazla yönlendirme URI'si kaydettirmişse veya yalnızca URI'nin bir kısmı (örneğin, şema, yetki ve yol) kaydedildiyse, istemci yetkilendirme isteğinde bulunduğunda **yönlendirme URI'sini "redirect_uri"** parametresiyle sunucuya göndermelidir. Bu, istemcinin tam yönlendirme URI'sini belirttiği ve yetkilendirme sunucusunun bu URI'yi doğrulaması gerektiği anlamına gelir.
- Eğer istemci **hiçbir yönlendirme URI'si kaydettirmemişse**, yine de **"redirect_uri"** parametresi ile yönlendirme URI'sini belirtmek zorundadır.

## 2. Yönlendirme URI'sinin Doğrulama Süreci

- **Yönlendirme URI'si** yetkilendirme isteğiyle birlikte sunucuya gönderildiğinde, yetkilendirme sunucusu şu adımları izler:
    1. **Kaydedilen URI'lerle Karşılaştırma**: Yetkilendirme sunucusu, istemcinin göndermiş olduğu yönlendirme URI'sini, kaydedilen yönlendirme URI'leri (veya URI bileşenleri) ile karşılaştırmalıdır. Bu karşılaştırma, **RFC 3986** bölüm 6'ya uygun olarak yapılmalıdır.
    2. **Tam URI Karşılaştırması**: Eğer istemci kaydında **tam yönlendirme URI'si** verilmişse, sunucu bu URI'yi doğrudan karşılaştırmalıdır. Bu karşılaştırma, basit bir **string (dize) karşılaştırması** olarak yapılır. Yani, istemcinin belirttiği URI ile kaydedilen URI'nin tam olarak eşleşip eşleşmediği kontrol edilir.
        - Örneğin, istemcinin kaydettiği yönlendirme URI'si `https://example.com/callback` ise, yetkilendirme sunucusu, istemcinin gönderdiği `redirect_uri` parametresinin bu tam URI ile eşleşip eşleşmediğini kontrol eder.

## 3. Yönlendirme URI'si Eşleşmeme Durumu

- Eğer istemcinin gönderdiği yönlendirme URI'si, kaydedilen URI'ler ile eşleşmezse, yetkilendirme sunucusu bu durumu hata olarak değerlendirmeli ve uygun bir hata yanıtı döndürmelidir. Bu genellikle, istemcinin kaydetmediği bir URI'ye yönlendirme yapmaya çalışması durumunda gerçekleşir.

## 4. Esneklik ve Dinamik Yönlendirme

- **Dinamik yapılandırma** sayesinde istemciler, kaydedilen URI'leriyle uyumsuzluk durumunda bile, belirli URI bileşenlerini dinamik olarak değiştirebilirler. Örneğin, istemci sadece URI'nin temel kısmını (şema, yetki, yol) kaydedebilir ve sorgu parametrelerini (query components) dinamik olarak değiştirebilir. Bu esneklik, istemcinin yönlendirme URI'sini daha esnek bir şekilde yönetmesine olanak tanır.

## 5. Özetle

- Eğer istemci birden fazla veya eksik yönlendirme URI'si kaydettirmişse, istemci her yetkilendirme isteğiyle birlikte tam yönlendirme URI'sini **"redirect_uri"** parametresi olarak göndermelidir.
- Yetkilendirme sunucusu, gönderilen yönlendirme URI'sini kaydedilen URI ile karşılaştırmalıdır. Bu karşılaştırma basit bir string karşılaştırmasıdır ve URI'nin tam olarak eşleşip eşleşmediği kontrol edilir.
- Bu süreç, istemci tarafından kullanılan URI'yi doğrulamak için gereklidir ve güvenlik amacıyla doğru eşleşme yapılmalıdır.

Bu özellik, istemcilerin dinamik yapılandırmalarını destekler ve yalnızca güvenli ve kayıtlı URI'lerin yönlendirme için kullanılmasına olanak tanır.

**3.1.2.4. Invalid Endpoint** bölümü, **OAuth 2.0** protokolünde, istemciden gelen geçersiz yönlendirme URI'leri (redirection URIs) ile ilgili olarak yetkilendirme sunucusunun nasıl davranması gerektiğini açıklamaktadır.

## 1. Geçersiz Yönlendirme URI'si Durumu

Eğer bir yetkilendirme isteği şu nedenlerden dolayı başarısız olursa:

- **Eksik yönlendirme URI** (yönlendirme URI'si parametresi gönderilmemişse),
- **Geçersiz yönlendirme URI** (geçersiz bir URI formatı veya hatalı yapılandırılmış bir URI gönderilmişse),
- **Uyumsuz yönlendirme URI** (gönderilen URI, kayıtlı URI ile eşleşmiyorsa),

o zaman **yetkilendirme sunucusu** şu şekilde hareket etmelidir:

## 2. Bilgilendirme ve Kullanıcı Yönlendirmesi

- **Yetkilendirme sunucusu**, **kaynak sahibi** (resource owner) olarak bilinen kullanıcının, yanlış bir yönlendirme URI'sine yönlendirilmeden önce bu hatayı bilgilendirmelidir. Yani, kullanıcıya bir hata mesajı gösterilmeli ve hatanın ne olduğunu anlaması sağlanmalıdır.
    - Örneğin, bir kullanıcı bir yetkilendirme işlemi yaparken, sistem geçersiz bir URI'yi doğrulamaya çalıştığında, sistem kullanıcıyı "Geçersiz yönlendirme URI" hatası ile bilgilendirebilir.
- **Otomatik Yönlendirme Yasaktır**: Yetkilendirme sunucusu, geçersiz bir yönlendirme URI'sine otomatik olarak kullanıcıyı yönlendirmemelidir. Yani, eğer bir hata oluşursa, kullanıcı doğrudan hatalı URI'ye yönlendirilmemelidir. Bunun yerine, sunucu bu hatayı bildirmeli ve kullanıcının doğru işlemi yapmasına olanak sağlamalıdır.

- Bu, güvenlik açısından kritik bir adımdır çünkü kötü niyetli bir kişi, geçersiz bir URI kullanarak kullanıcıyı zararlı bir siteye yönlendirmeyi amaçlayabilir. Yetkilendirme sunucusunun otomatik olarak hatalı URI'lere yönlendirme yapmaması, bu tür saldırılara karşı bir koruma sağlar.

## 3. Özetle

- **Geçersiz bir yönlendirme URI** geldiğinde, yetkilendirme sunucusu kullanıcılara bu hatayı bildirmelidir.
- Kullanıcı, hatalı URI'ye yönlendirilmemeli, doğru yönlendirme işlemi için hata mesajı gösterilmelidir.
- Bu davranış, güvenlik önlemleriyle uyumlu olup kötü niyetli saldırılara karşı bir koruma sağlar.

Bu bölüm, OAuth protokolünde güvenlik sağlamak ve kullanıcıyı olası hatalı yönlendirmelerden korumak için önemlidir.

**3.1.2.5. Endpoint Content** bölümü, OAuth 2.0 protokolünde, istemciye yönlendirme yapılan endpoint'in içerik güvenliği ile ilgili önlemleri tartışmaktadır. Bu bölüm, istemcilerin güvenli bir şekilde yönlendirme URI'lerinden alınan kimlik bilgilerini işlemeleri ve kullanıcı verilerini korumaları gerektiğini belirtir.

## 1. Yönlendirme Sonucu HTML Belgesi

Yönlendirme URI'sine yapılan bir istek genellikle bir **HTML belgesi** yanıtı döndürür. Bu HTML yanıtı, kullanıcının tarayıcısı (user-agent) tarafından işlenir ve gösterilir. Ancak, bu işlem sırasında **HTML belgesine eklenen scriptler**, URI'den alınan kimlik bilgilerine tam erişim sağlar. Bu da, kötü niyetli scriptlerin bu verilere müdahale edebilmesine veya onları dışarıya sızdırmasına olanak tanıyabilir.

Örneğin, bir istemci yönlendirme URI'sini içerik olarak gönderirse ve bu URI kimlik bilgileri içeriyorsa, içeriği işleyen tarayıcıdaki bir JavaScript kodu, bu kimlik bilgilerine erişebilir. Bu, özellikle kötü amaçlı scriptler için güvenlik açığı yaratabilir.

## 2. Üçüncü Taraf Scriptlerinin Kullanılmaması

- **İstemci, yönlendirme endpoint'inin yanıtında üçüncü taraf scriptlerini (örneğin, analitik araçları, sosyal medya eklentileri, reklam ağları) kullanmamalıdır.**
    - Üçüncü taraf scriptleri, kötü amaçlı yazılımlar, izleme scriptleri veya güvenlik açıkları taşıyabilir. Bu nedenle, istemci bu tür scriptleri kendi redirection endpoint'lerinde kullanmamalıdır.
- **Kimlik bilgilerini URI'den çıkartarak başka bir endpoint'e yeniden yönlendirme yapılmalıdır.**
    - Kimlik bilgileri URI'den çıkarılmalı ve güvenli bir şekilde başka bir endpoint'e yönlendirilmelidir. Bu, URI içinde hassas bilgilerin görünür olmamasını sağlar ve güvenliği artırır. Bu sayede, credential'lar (kimlik bilgileri) istemci tarafında herhangi bir şekilde sızmaz.

## 3. Scriptlerin Çalıştırılma Sırası

- Eğer istemci **üçüncü taraf scriptlerini kullanmak zorundaysa**, istemci, **kendi scriptlerinin** (kimlik bilgilerini URI'den çıkartıp güvenli bir şekilde işleyen scriptlerin) **öncelikli olarak çalışmasını sağlamalıdır.**
    - Bu, istemcinin kendi güvenlik önlemlerini alabilmesi için önemlidir. Kimlik bilgilerini URI'den çıkarıp başka bir endpoint'e yönlendirmeden önce, üçüncü taraf scriptlerinin bu verilere erişmesini engellemek gereklidir.

## 4. Özetle

- **Yönlendirme URI'si ile birlikte gelen HTML yanıtında** kimlik bilgileri yer alıyorsa, **üçüncü taraf scriptlerinin** bulunmaması gerekir.
- İstemci, **kimlik bilgilerini URI'den çıkartarak başka bir endpoint'e güvenli bir şekilde yönlendirme yapmalıdır.**
- **Üçüncü taraf scriptleri** kullanılacaksa, istemci, **kendi güvenlik scriptlerinin** öncelikli olarak çalışmasını sağlamalıdır.

Bu bölüm, güvenli olmayan uygulamalara karşı bir koruma sağlar ve istemcilerin kullanıcı kimlik bilgilerini üçüncü taraflardan gizli tutmalarını ve doğru şekilde işlemesini sağlar. Bu, OAuth protokolünün güvenli bir şekilde uygulanması açısından kritik bir güvenlik önlemidir.

**3.2. Token Endpoint** bölümü, OAuth 2.0 protokolündeki token endpoint'inin kullanımını ve güvenlik gereksinimlerini açıklar. Token endpoint, istemcinin **yetkilendirme belgesi** veya **yenileme token'ı** ile erişim token'ı almak için kullandığı bir kaynaktır. Bu süreç, kullanıcıya verilen yetkilendirme bilgileriyle istemcinin bir erişim token'ı almasını sağlar.

## 1. Token Endpoint Kullanımı

Token endpoint, istemcinin **yetkilendirme belgesini** veya **yenileme token'ını** sunarak bir **erişim token'ı** almasına olanak tanır. Bu endpoint, genellikle her yetkilendirme akışında kullanılır, ancak **implicit grant** akışında doğrudan erişim token'ı verildiği için bu endpoint kullanılmaz.

İstemcinin token endpoint'in yerini nasıl alacağı, bu spesifikasyonun kapsamı dışındadır, ancak genellikle **hizmetin belgelerinde** bu bilgi sağlanır.

## 2. Token Endpoint URI

Token endpoint URI'si, **application/x-www-form-urlencoded** biçiminde bir **sorgu parametresi** içerebilir. Bu parametreler, ek parametreler eklenirken **korunmalı** ve **URI'ye** eklenmelidir. Ancak, endpoint URI'si **fragment (parça) bileşeni** içermemelidir.

## 3. TLS Zorunluluğu

Token endpoint'e yapılan istekler, **kimlik bilgilerini** açık metin (clear-text) olarak taşıdığından, bu endpoint'e yapılan isteklerde **TLS (Transport Layer Security)** kullanılmalıdır. Bu, tüm erişim token'larının güvenli bir şekilde iletilmesini sağlamak için gereklidir. Bu, **SSL/TLS** şifrelemesi ile verilerin güvenli bir şekilde taşınmasını garanti eder.

## 4. HTTP Yöntemi

İstemci, token endpoint'e **erişim token'ı isteği** yaparken **POST** yöntemini kullanmalıdır. **GET** yöntemi burada kullanılmaz çünkü bu tür istekler genellikle kimlik doğrulama bilgilerini taşımaz ve daha az güvenlidir.

## 5. Parametreler ve Geçerlilik

- **Değeri olmayan parametreler**, istekten **ihmal edilmiş** gibi muamele görmelidir.
- **Tanımadığı parametreleri** yetkilendirme sunucusu **yok saymalıdır**.
- **Tekrar eden parametreler**, hem istek hem de yanıtlar için **tekrar edilmeyecek** şekilde işlenmelidir.

## 6. Özetle

- Token endpoint, istemcinin **erişim token'ı** almak için kullandığı **POST** tabanlı bir endpoint'tir.
- Bu endpoint'e yapılan isteklerin, kimlik bilgilerini güvenli bir şekilde taşıması için **TLS** kullanılmalıdır.
- İstek parametreleri, geçersiz veya eksik olduklarında doğru şekilde işlenmeli, ve tanınmayan parametreler görmezden gelinmelidir.

Bu bölümü anlamak, istemcinin güvenli bir şekilde erişim token'ları almasını ve istemci ile yetkilendirme sunucusu arasındaki iletişimin güvenliğini sağlamak açısından kritik öneme sahiptir.

**3.2.1. Client Authentication** bölümü, OAuth 2.0 protokolü çerçevesinde istemcilerin **token endpoint**'ine istek yaparken nasıl kimlik doğrulaması yapması gerektiğini açıklar. Özellikle, **confidential client'lar** (gizli istemciler) ve **client credentials** verilen diğer istemciler için kimlik doğrulama sürecinin nasıl işlemesi gerektiğini ele alır.

## 1. Kimlik Doğrulamanın Amacı

İstemci kimlik doğrulaması, aşağıdaki kritik amaçlar için gereklidir:

- **Yenileme token'ları ve yetkilendirme kodlarının istemciye bağlanmasını sağlamak**: Özellikle yetkilendirme kodları, güvenli olmayan bir kanal üzerinden **redirection endpoint**'ine gönderildiğinde veya redirection URI tam olarak kaydedilmediğinde kimlik doğrulaması kritik olur. Bu sayede, yalnızca belirli bir istemci, ona ait olan yetkilendirme kodlarını kullanabilir.
- **Tehlikeye düşmüş istemcilerin kurtarılması**: Eğer bir istemci tehlikeye düşerse, kimlik doğrulama sistemi kullanılarak istemcinin **kimlik bilgileri** değiştirilebilir ve **yeniden yapılandırılabilir**. Bu, saldırganların çalınan yenileme token'larını kullanarak erişim sağlamalarını engeller. Bu işlem, tüm yenileme token'larının iptal edilmesinden çok daha hızlı bir çözüm sağlar.
- **Kimlik doğrulama yönetimi ve güvenlik uygulamalarına uygunluk**: Kimlik doğrulama bilgilerini düzenli olarak güncellemek, bir güvenlik en iyi pratiğidir. İstemci kimlik bilgilerini döndürmek ve değiştirmek, yenileme token'larının tüm setlerini döndürmekten daha kolaydır. Bu da güvenlik açısından önemli bir avantaj sağlar.

## 2. Client ID Kullanımı

İstemci, token endpoint'ine istek gönderirken, kimliğini belirlemek için **client_id** parametresini kullanabilir. Bu parametre, istemcinin kimliğini tanımlamak için gereklidir ve isteğin doğru istemciye ait olduğunun doğrulanmasına yardımcı olur. Özellikle **authorization_code grant_type** kullanıldığında, istemci kimliği **client_id** parametresi olarak gönderilmelidir. Bu, istemcinin yanlışlıkla başka bir istemci için gönderilmiş olan yetkilendirme kodunu kabul etmesini engeller.

## 3. Kimlik Doğrulamanın Sağladığı Güvenlik

- **Yetkilendirme kodunun değiştirilmesi**: Client ID kullanmak, istemcinin yalnızca kendisi için verilmiş olan yetkilendirme kodunu kabul etmesini sağlar. Bu, **yetkilendirme kodu değiştirildiğinde** ya da **başka bir istemci** kodu kullanmaya çalıştığında, istemcinin bu durumu fark etmesine ve işlemi reddetmesine olanak tanır.
- **Korunan kaynaklar için ek güvenlik sağlamaz**: Ancak bu kimlik doğrulama, **korunan kaynaklara** ek güvenlik sağlamaz. Yalnızca istemcinin kendine ait olmayan bir yetkilendirme kodunu kullanmasını engeller.

## 4. Kimlik Bilgisi Değişiklikleri ve Güvenlik

İstemci kimlik bilgilerini düzenli olarak değiştirmek, güvenlik açısından önemlidir. Bu, **yenileme token'larını** kullanarak yapılan saldırıların önlenmesinde önemli bir adım olabilir. İstemci kimlik doğrulamasının bu şekilde yapılandırılması, uygulamanın uzun vadede güvenliğini artırır ve dış saldırılara karşı daha dayanıklı olmasını sağlar.

## Özetle

- **Confidential clients** ve **client credentials** verilen diğer istemciler, **token endpoint**'ine yapılan isteklerde kimlik doğrulaması yapmalıdır.
- Kimlik doğrulamanın amacı, istemciye ait olan yetkilendirme kodlarının doğru istemci tarafından kullanılmasını sağlamak, tehlikeye düşmüş istemcileri kurtarmak ve kimlik doğrulama bilgilerini düzenli olarak güncellemektir.
- **client_id** parametresi, istemcinin doğru yetkilendirme kodunu almasını sağlamalıdır.
- Kimlik doğrulama, **korunan kaynaklara ek güvenlik sağlamaz**, ancak istemcinin doğru kaynaklara erişmesini garanti eder.

Bu süreç, istemcinin güvenli bir şekilde token almasını ve token'ların yanlış istemciler tarafından kullanılmasının önüne geçilmesini sağlar.

**3.3. Access Token Scope** bölümü, OAuth 2.0 protokolünde istemcilerin erişim taleplerini belirli bir **scope** (kapsam) ile sınırlandırmalarını ve bu kapsamın nasıl iletildiğini açıklar. Kapsam, istemcinin erişebileceği kaynaklar ve bu kaynaklar üzerinde hangi işlemleri gerçekleştirebileceği konusunda bir kısıtlama sağlar. Bu bölümde belirtilenler, istemci ile yetkilendirme sunucusu arasındaki etkileşimde **scope** parametresinin nasıl kullanılacağını anlatır.

## 1. Scope Parametresi

- **Scope** parametresi, istemcinin erişmek istediği kaynakları ve izinleri belirtmek için kullanılır. Bu parametre, istemcinin authorization (yetkilendirme) ve token (token alma) endpoint'lerine yaptığı isteklerde yer alabilir.
- **Scope** parametresi, bir dizi **boşlukla ayrılmış (space-delimited)** ve **büyük/küçük harfe duyarlı** kelimelerden oluşur. Bu kelimeler, yetkilendirme sunucusu tarafından tanımlanır.
- Birden fazla **scope-token** (kapsam belirleyicisi) içeren bir scope, her bir kelimenin erişim yetkisini artırdığı anlamına gelir. Yani birden fazla kelime (örneğin, "read write") belirtildiğinde, istemci her iki erişim hakkını talep eder.

  **Örnek**: `scope=read write` Bu örnekte, istemci hem okuma (read) hem de yazma (write) yetkisi talep etmektedir.

  **Kapsamın Tanımı:**

  *scope = scope-token *( SP scope-token )*
  *scope-token = 1*( %x21 / %x23-5B / %x5D-7E )*

Bu tanım, `scope-token`'ın hangi karakterleri içerebileceğini belirtir. Örneğin, `%x21` ASCII kodu ile "!" karakterini ifade eder.

## 2. Yetkilendirme Sunucusunun Cevabı

- Yetkilendirme sunucusu, istemciden gelen **scope** parametresini işleyerek, istemcinin talep ettiği **scope**'u doğrular. Ancak, yetkilendirme sunucusu bu **scope**'u tamamen veya kısmen göz ardı edebilir. Bunun nedeni, yetkilendirme sunucusunun güvenlik politikaları veya kaynak sahibinin (resource owner) verdiği talimatlar olabilir.
- Eğer yetkilendirme sunucusu, istemcinin talep ettiği **scope**'a uymayan bir **access token** verir ve verilen **scope** ile istemcinin talep ettiği **scope** farklıysa, **scope** yanıt parametresi kullanılarak istemciye verilen kapsam bildirilmelidir. Bu, istemcinin hangi erişim alanlarının sağlandığını anlamasını sağlar.

  **Örnek**: Eğer istemci `scope=read write` talep ettiyse ve sunucu yalnızca `read` izni veriyorsa, sunucu yanıtında şu şekilde bir **scope** parametresi dönebilir: `scope=read`.

## 3. Scope Parametresi Olmazsa Ne Olur?

- Eğer istemci, **scope** parametresini yetkilendirme talebinde belirtmezse, yetkilendirme sunucusu, bu durumda **önceden tanımlanmış bir varsayılan değer** ile isteği işleme alabilir. Eğer varsayılan bir değer yoksa, **invalid_scope** hatası dönebilir. Bu durumda istemci, hatanın nedeni olarak geçersiz bir **scope** olduğunu anlamalıdır.

- Yetkilendirme sunucusu, **scope** parametresinin ne olduğunu ve varsa varsayılan değeri **belgelerinde** açıklamalıdır.

## 4. Özet ve Güvenlik

- **Scope** parametresi, istemcinin erişmek istediği kaynakları sınırlamak için kullanılır. Bu, istemcinin sadece gerekli kaynaklara erişmesini ve gereksiz yetkileri talep etmemesini sağlar.
- Yetkilendirme sunucusu, istemcinin talep ettiği kapsamı tamamen ya da kısmen değiştirebilir, ancak istemciye bu durum bildirilmelidir.
- Eğer istemci **scope** parametresi göndermezse, yetkilendirme sunucusu varsayılan bir değer kullanabilir veya hatayı bildirebilir.

Bu mekanizma, OAuth 2.0 protokolünün daha esnek ve güvenli bir şekilde çalışmasını sağlar ve istemcilerin sadece gerekli erişim haklarını talep etmelerini teşvik eder.