WEB OTURUMLARI NASIL ÇALIŞIR?

Betül Biteker, BTYÖN Danışmanlık

Web uygulamaları, HTTP protokolü üzerinden çalışır. HTTP Protokolünün durum bilgisi (state) tutmamasından ötürü, uygulamaya gelen HTTP isteklerinin aynı kullanıcıya ait olup olmadığının belirlenmesi için oturum bilgisi kullanılır.

Çerezler (cookie) kullanılarak yapılan, en genel ve en basit giriş yapma yöntemi esas alınırsa:

1. A kullanıcısı örnek bir web uygulamasına giriş yaptıktan sonra, uygulama A kullanıcısına özel bir giriş çerezi(authentication cookie) atar. Çerez ataması yapılan örnek bir HTTP cevabı şu şekildedir:

```
HTTP/1.0 200 OK
Content-type: text/html
Set-Cookie: name=SESSIONID; Expires=Wed, 09 Jun 2021 10:18:14 GMT
```

2. Web tarayıcısı (browser) bu cookie'yi kendi hafızasına kaydeder. Sonra o browser penceresinden yapılan her isteğe bu cookie otomatik şekilde eklenir:

```
GET /spec.html HTTP/1.1
Host: www.example.org
Cookie: name=SESSIONID
Accept: */*
```

3. Uygulama daha önce atadığı cookie'leri HTTP isteğinde görünce, bu isteğin hangi kullanıcıdan geldiğini belirler ve ona göre yanıt verebilir.

Güvenli Oturum Yönetimi

Web oturum sistemini güvenli hale getirmek için aşağıdaki hususlara dikkat etmek gerekir.

1. Oturum Tanımlayıcı (Session ID) Özellikleri

1.1. Oturum Tanımlayıcı (Session ID) İsmi

Oturum tanımlayıcısının ismi, kullanıcıya herhangi bir bilgi vermemelidir. Zira çerezler kullanıcıların kullanması için değil, tarayıcı (browser) ile web sunucu (web server) arasında paylaşılan, birbirlerini tanımalarını sağlayan bir bilgidir.

Bir çok klasik web frameworkleri (geliştirme çatıları) PHPSESSID (PHP), JSESSIONID (J2EE), CFID & CFTOKEN (ColdFusion), ASP.NET_SessionId (ASP .NET) gibi isimler kullanmakta ve bu isimlere bakarak arka planda çalışan teknolojiler hakkında(fingerprinting) bilgi sahibi olmak mümkün olmaktadır.

Oturum tanımlayıcı çerezler için, 'id' gibi genel ve ekstra bir bilgi sızdırmayan isimler tercih edilmelidir.

1.2. Oturum Tanımlayıcı (Session ID) Boyutu ve Entropisi

Kaba kuvvet saldırılarını ve dolayısıyla saldırganların bütün muhtemel oturum tanımlayıcıların varlığını kontrol etmelerini engellemek amacıyla, oturum tanımlayıcılarının uzunluğu tatminkar boyutta olmalıdır. Günümüz teknolojileri için, 128 bit uzunluğundaki oturum tanımlayıcıları yeterli güvenlik sağlamaktadır.

Oturum tanımlayıcıları tahmin edilebilir yapıda da olmamalıdır. Çok sayıda oturum tanımlayıcısı toplayan bir saldırgan, bu oturum tanımlayıcıları üzerinde istatiksel analizlerle, herhangi bir korelasyon çıkartamamalıdır. Bunu başarabilmek için ise; oturum tanımlayıcı üretilirken, işletim sistemi ve teknolojinin sağladığı rastgele sayı üreteci (PRNG) fonksiyonları kullanılmalıdır.

1.3. Oturum Tanımlayıcı Değeri

Oturum tanımlayıcının değeri (ya da içeriği) herhangi bir şey ifade etmemeli ve bir bilgi sızdırmamalıdır. Zira saldırganlar tarafından görülebilecek olan bu değer, saldırganın üzerinde analiz yapabileceği bir veridir ve uygulamanın çalışma mantığının açığa çıkmasına sebep olabilir.

Oturum tanımlayıcı kesinlikle kişisel bilgi içermemelidir. Bunun bir örneğini vermek gerekirse:

```
Set-Cookie: kullanıcı=betul; Expires=Wed, 09 Jun 2021 10:18:14 GMT
```

Uygulamaya giriş yapıldıktan sonra, yukarıda bahsedilen gibi bir çerez atanıyor ve çerezdeki kullanıcı adı bilgisine göre tanımlama yapılıyorsa, kötü niyetli bir saldırgan bu değeri (Burp gibi bir proxy ya da cookie manager gibi bir tarayıcı eklentisi ile) manipüle edebilir ve aşağıdaki gibi bir istek ile uygulamaya 'admin' kullanıcısıyla giriş yapabilir.

```
GET /spec.html HTTP/1.1

Host: www.example.org

Cookie: kullanıcı=admin
```

```
Accept: */*
```

Oturum tanımlayıcıya verilecek değer, bir şey ifade etmeyen bir değer olmalıdır. Bunu da yapmanın en güzel yolu, kriptografik özet fonksiyonlarını kullanmaktır. Örneğin aşağıdaki örnekte, rastgele bir değer üretilmiş ve o değerin SHA-256 algoritmasi ile kriptografik özeti alınmış ve o kullanılmıştır.

```
Set-Cookie:  id=1FD27D7ED2AE348AD0325CAC43D65EE01DF3CBDC3C701093E4408A5CD0E54491;
Expires=Wed, 09 Jun 2021 10:18:14 GMT
```

2. Taşıma Katmanı Güvenliği (TLS)

Web uygulamalarında HTTPS implementasyonu başlı başına uzun bir konu, konu web uygulamalarında oturum güvenliği olduğu için kabaca kesişen noktaları özetlemek gerekirse:

- Oturum tanımlayıcısı çerez 'secure cookie' olmalıdır.
- Giriş (login) sayfası ve hatta onu linkleyen ana sayfa HTTPS üzerinden çalışmalıdır.
- Oturum tanımlayıcısını ve oturum sırasında web tarayıcı ile web server arasında gidip gelen diğer bilgileri de ağ izlemesinden (network monitoring) korumak için, oturum sırasında bütün bilgi akışı HTTPS(SSL/TLS) üzerinden yapılmalıdır.
- Kullanıcıların https://www.example.com yerine, el alışkanlığı ile www.example.com adresini yazıp girmeleri ve bu adresin de HTTPS üzerinden çalışmaması sebebiyle meydana gelebilecek MITM ataklarını engellemek için de 'HTTP Strict Transport Security (HSTS)' kullanılmalıdır. Bu protokolün kullanımında kullanıcı yanlışlıkla www.example.com yazıp bu websitesine erişmek istese dahi, web tarayıcısı buna izin vermeyecek ve kullanıcıyı https://www.example.com 'a yönlendirecektir.

3. Çerez Güvenliği

Oturum tanımlayıcıları, web tarayıcıları ve web sunucuları arasında gidip gelirken, çerezler üzerinden taşınır. Bundan ötürü kullanılan çerezi de güvenli hale getirmeliyiz.

3.1. Güvenli Çerez (Secure Cookie)

Çerezin 'secure' özelliği işaretlenirse, web tarayıcısı bu çerezi sadece HTTPS üzerinden yapılan bağlantılarda kullanacak, HTTP isteklerine eklemeyecektir. Aşağıda örnek bir 'secure' işaretlenmiş çerez ataması örneği bulunmaktadır.

```
Set-Cookie: kullanıcı=betul; secure
```

Aksi takdirde, https://www.youtube.com tarafından atanmış olan çerez, bir saldırganın araya girme saldırısı yapıp, SSL'i çözmeden, herhangi bir HTTP isteği tetikleyecek bir içerik eklemesiyle (örneğin) oluşacak HTTP isteğine eklenecek ve saldırgan tarafından kolaylıkla elde edilebilecektir.

3.2. HTTPOnly Çerezi (HTTPOnly Cookie)

HTTPOnly isimli çerez özelliği, çerezin JavaScript ya da benzeri istemci tarafında çalışan betik ortamlarından erişimlerini engeller. Örneğin bir çerez eğer 'httponly' olarak işaretlendiyse, JavaScript üzerinde 'document.cookie' nesnesiyle erişmek mümkün olmayacaktır.

```
Set-Cookie: kullanıcı=betul; httponly
```

Bu sayede XSS ataklarıyla oturum çerezinin çalınması dolayısıyla gerçekleşen oturum korsanlığı engellenmiş olur.

3.3. Kalıcı ve Geçici Çerezler

İki türlü çerez olabilir, kalıcı ve geçiçi şekilde. Eğer bir çerez Max-Age ya da Expires özellikleri ile atanırsa, bu kalıcı bir çerez olarak değerlendirilecek ve geçerli olduğu sürece web tarayıcı tarafından harddiskte saklanacak anlamına gelir. Örneğin Max-Age 1 yıl olarak atanırsa, o çerez o bilgisayarda 1 yıl tutulacak demektir.

Web oturumlarında genellikle geçiçi çerezler kullanılmalıdır. Bu sayede web tarayıcı kapatılınca çerezler de silinecek ve tekrar web tarayıcı açılıp aynı sayfa ziyaret edilirse, oturumun artık açık olmadığı görülecektir.

Ek olarak 'beni hatırla' gibi mekanizmalarla, yukarıda bahsedilmiş olan Expires ya da Max-Age kullanarak, bir hafta ya da iki hafta gibi süreli kalıcı çerezler de kullanılabilir. Bu tercih uygulamanın sağladığı işlevlerin önemi (örneğin bankacılık uygulamalarında para transferi mümkün mü?) ve potansiyel sızacak bilgilerin hassaslığına bağlı olarak, risk analizi yapılarak, belirlenmelidir.

4. Oturum Tanımlayıcısı Yaşam Döngüsü

4.1. Oturum Tanımlayıcı Üretimi

Oturum tanımlayıcıları üretilirken, rastgele sayı üreteci fonksiyonlar yardımıyla (PRNG) rastgele değerler üretilmeli ve bu değerlerin kriptografik özeti alınmalı, bu özetler çerez olarak atanmalıdır.

Bunun yerine web framework 'lerinin otomatik oturum tanımlayıcı (session id) üretme mekanizmaları da kullanılabilir.

Bu noktada birkaç güvenlik notundan bahsetmek gerekirse:

- Oturum tanımlayıcıları her giriş (login) işleminde sunucu tarafından yeniden üretilmeli ve bu üretilen tanımlayıcı kullanıcıya atanmalıdır. Kullanıcıdan gelen oturum tanımlayıcılarına itibar edilmemelidir. Bu sayede oturum sabitleme (session fixation) saldırılarının önüne geçilmiş olur.
- Oturum tanımlayıcıları, diğer her türlü input (girdi) gibi değerlendirilmeli ve programcı tarafından uygulamanın içinde kullanılmadan önce onaylanmalı ve güvenlik kontrollerinden geçirilmelidir. Bu sayede oturum tanımlayıcılarının uygulama içerisinde işlenirken meydana gelebilecek SQL Injection (örneğin oturum tanımlayıcı veritabanina kaydedilir ya da arama yapılırken) ya da XSS (oturum tanımlayıcı bir şekilde ekrana bastırılıyorsa) gibi zafiyetlerden de kaçınılabilir.

5. Oturum Sonlandırılması ve Zaman Aşımı

Birkaç farklı şekilde oturum sonlandırması ve zaman aşımı kontrolleri uygulanmalıdır.

5.1. Otomatik oturum sonlandırılması

Oturumların çalındığı durumlarda, saldırganlara işlem yapma süresini kısıtlamak için çeşitli zaman aşımı mekanizmaları devreye sokulmalıdır.

5.2. Hareketsiz Kalma Zaman Aşımı

Kullanıcıdan belli bir süre yanıt alınamadığı zaman gerçekleşen oturum sonlandırılmasıdır. Örneğin bunu 5 dakika olarak ayarlarsak; eğer kullanıcıdan 5 dakika boyunca hiçbir HTTP isteği gelmez ise o oturum otomatik olarak kapatılmalıdır.

Burada 5 dakika örnek olarak verilmiştir, uygulamanın hassasiyetine göre bir süre belirlenmelidir.

5.3. Toplam Süre Zaman Aşımı

Kullanıcı uygulamaya giriş yaptıktan sonra, herhangi bir aktivite olup olmadığına bakmaksızın, belli bir toplam süreyi geçince yapılması gereken oturum sonlandırması işlemidir. Örneğin maksimum süreyi

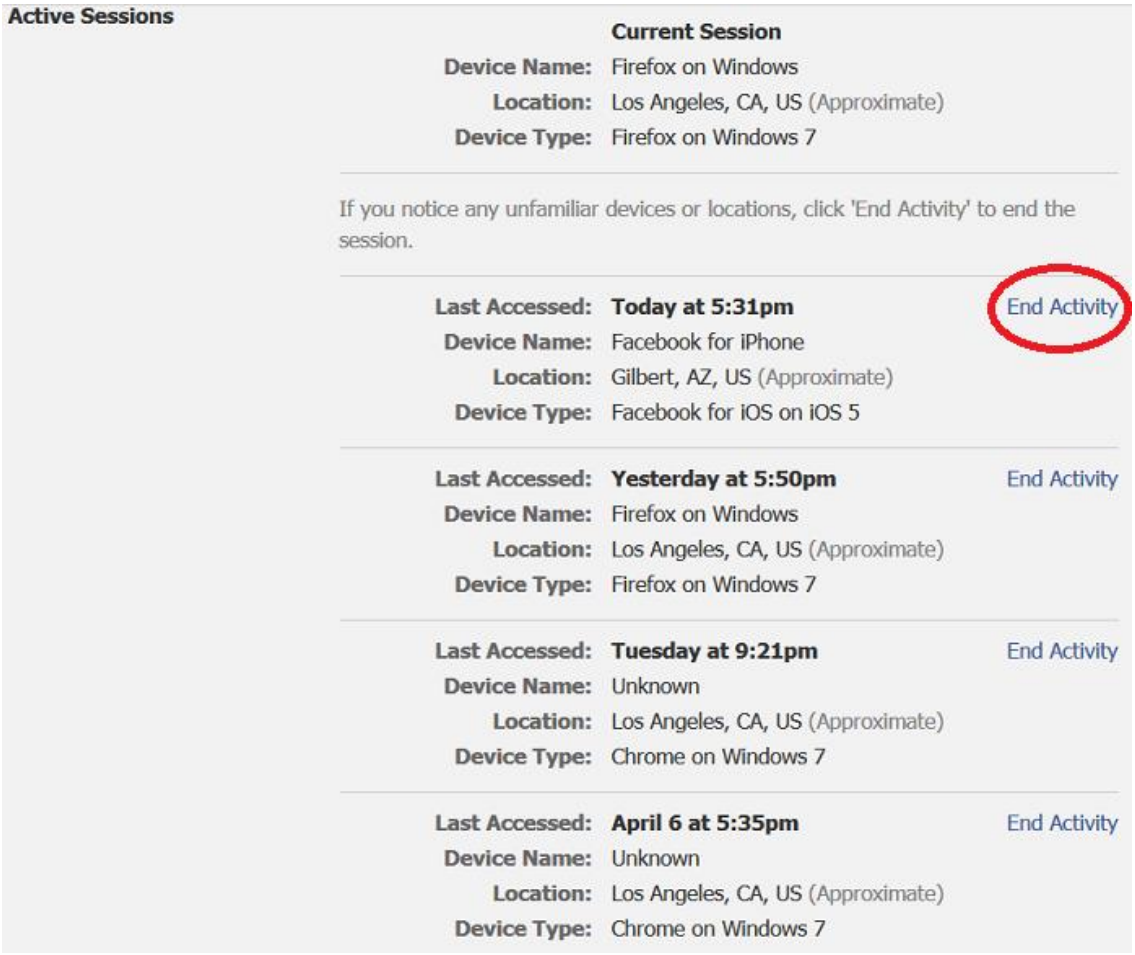
30 dakika olarak ayarlarsak, giriş yapıldıktan 30 dakika sonra o oturum tanımlayıcı geçersiz olarak işaretlenmeli ve kullanıcıyı giriş sayfasına yönlendirmek suretiyle tekrar giriş yaptırtılmalıdır.

5.4. Manuel Oturum Sonlandırılması

Kullanıcı istediği herhangi bir zamanda, uygulama üzerindeki bir tuşa basarak oturumu bitirebilmelidir. Yani web uygulaması, kullanıcıya bir 'çıkış' butonu sağlamalıdır. Kullanıcı çıkış butonuna bastığında, uygulama, oturum tanımlayıcısını geçersiz hale getirmelidir."

Ek Notlar

- Oturum çerezlerinin URL üzerinde taşınması engellenmelidir zira URL bilgileri bir çok yerde (web sunucu logları, proxy logları, ekran görüntüleri vs.) kayıt edilebilir. Her zaman HTTP paketinin Cookie bölümlerinde taşınmalıdır.
- Kullanıcıya IP adresi limitlemesi seçeneği verilebilir. Kullanıcı uygulamaya giriş yaptıktan sonra sadece belli bir IP listesinden gelen istekler ile uygulamaya giriş yapılması sağlanabilir. Bu yöntemde, kullanıcı adı ve parola çalınsa dahi saldırgan hesap sahibinin IP adresinden gelemeyeceği için, uygulamaya giriş (login) yapamayacaktır.
- Ülke limitlemesi de güzel bir koruma olabilir. Uygulamaya giriş yapacak kullanıcının hangi ülkede yaşadığı biliniyorsa, uygulamaya giriş (login) yapılırken, HTTP isteklerinin geldiği IP adresi üzerinden ülke tespiti yapılabillir ve istek yurtdışından geliyorsa, giriş yapılması engellenebilir.
- Kritik uygulamalar için 2 Factor Authentication dediğimiz, giriş yapılması işlemi sırasında kullanıcı adı ve parola dışında bir de SMS mesajı ile doğrulama yapılabilir.
- Facebook'taki gibi, şu anda uygulamaya yapılmış girişlerin görüntülenebileceği ve bunları sonlandırmanın mümkün olacağı bir ekran sağlanabilir. Örneğin:



Kaynaklar

1. https://www.owasp.org/index.php/Session_Management_Cheat_Sheet