# OWASP TOP-10

1-Broken Access Control:

Broken Access Control web uygulaması içerisinde yetkisiz kullanıcıların erişmemesi gereken kaynaklara, dosyalara erişmelerini ifade eder.

Broken Access Control açıkları, uygulama tarafından yetki kontrolünün yetersiz yapılmasından dolayı kaynaklanmaktadır. Örneğin bir bilet uygulaması düşünelim. Bu uygulama kullanıcılara aldıkları bileti göstersin. Bu uygulama içerisinde bir kullanıcının biletini başka bir kullanıcının görmemesi gerekir. Eğer bu uygulamada yetersiz doğrulama varsa ve başka bir kullanıcı diğer biletlere erişebilirse burada bu zafiyet ortaya çıkar.

İDOR açığı bu kategoriye girmektedir. İDOR açığı kullanıcının erişmemesi gereken kaynaklara erişebilmesini ifade eder. Yukarıdaki örneğimizden yola çıkarsak bilet görüntüleme sayfasının URL kısmı <https://şubilet.com/1337/bilet.pdf> olsun. 1337 sayısı da kullanıcının id’sini temsil etsin. Eğer kullanıcı bu sayıyı değiştirerek başka kullanıcıların biletine ulaşabilirse İDOR açığının bulunduğunu anlarız.

CSRF( Siteler arası istek sahteciliği) açığı da bu kategoriye dahildir. CSRF açığı kullanıcıların session değerlerini elde eden saldırganın dışarıdan istek göndererek kullanıcı adına işlem gerçekleştirmesidir. Bu açık bulunan web uygulamasında, kullanıcının saldırıya uğraması için; sadece kötü amaçlı bir linke tıklaması yeterlidir.

Bu açığın önlenmesi için:

* Giriş yapan kullanıcıların talep ettiği dosya ve kaynakların gerçekten o kullanıcıya ait olup olmadığı doğrulanmalıdır.
* Token-based authentication ve JWT Token yöntemleri kullanılmalıdır. JWT Token sayesinde benzersiz cookie bilgileri oluşturularak, başka kullanıcıların oturum cookie’sinin tahmin edilmesi zorlaştırılır.

2-Cryptographic Failures:

Bu kategori web uygulaması tarafından, kullanıcıların hassas bilgilerinin açık metin şekilde tutulması veya şifrelenmişse; zayıf şifreleme algoritmaları kullanılması veya kullanılan algoritmada bir açığın olmasını ifade eder.

Bu açık, modası geçmiş eski algoritmaların kullanılması sonucu ortaya çıkar. Şifreyi çözecek olan anahtar (Key) değerinin kullanıcı tarafından ulaşabilecek bir konuma koyulması veya direkt olarak kod içerisinde bulunması sonucunda ortaya çıkar.

Bu açığın önlenmesi için:

* Doğru ve güçlü şifreleme algortimaları kullanılmalı. Örneğin MD5 ve SHA1 gibi algoritmalar kullanılmamalı.
* Şifreyi çözen Key değeri kolay ulaşılabilir bir konumda bulunmamalı ve düzenli aralıklara bu Key değeri değiştirilmelidir.

3-İnjection:

İnjection kategorisi, kullanıcıdan alınan girdilere çeşitli SQL dili sorguları veya işletim sistemleri komutları eklenerek; zararlı kod enjekte edilmesidir.

Bu açık kullanıcıdan alınan verinin filtrelenmeden direkt olarak arka planda çalıştırılması sonucunda ortaya çıkar.

Saldırgan bu açığı kullanarak, örneğin bir SQL sorgusuna müdahale ederek bütün kullanıcı bilgilerini alabilir veya bir işletim sistemi komutu enjekte ederek web sunucusunun kontrolünü ele geçirebilir.

SQL İnjection açığı bu kategoriye dahildir. Bu açık saldırganın SQL dilinde sorgular göndererek, SQL sorgusuna müdahale etmesidir. Kritik bir açıktır, saldırgan bu açık sayesinde sunucudan bütün veriyi çekebilir.

Command İnjection açığı da bu kategoriye girer. Bu açık saldırganın işletim sistemi komutlarını göndermesi ve bunun sunucuda çalışmasıdır. Örneğin bir web sitemiz olsun, bu web sitesi kullanıcıdan alınan İP adresine ping atsın. Saldırgan İP yerine ip’den sonra (&&) girdisini vererek başka komutlar çalıştırabillir.

Code İnjection bu kategoriye giren başka bir açıktır. Bu açık saldırganın girdiye kod girmesi ve bu kodun sunucu tarafından çalışmasıdır. Örneğin saldırgan bu açığın bulunduğu web uygulamasına php kodu göndererek php kodunu web uygulamasında çalıştırabilir.

Bu açığın engellenmesi için:

* Kullanıcıdan alınan verinin çeşitli filtreleme işlemlerinden geçirildikten sonra back-end işlemlerine dahil edilmelidir.
* WAF kullanılmalıdır.

4-İnsecure Design:

İnsecure Design, web uygulamasının geliştirilme aşamasında güvensiz olarak geliştirilmesini ifade etmektedir. Bu yapılan güvensiz geliştirme uygulamanın ömrü boyunca devam eder ve çeşitli güvenlik açıkları ortaya çıkarır.

Örneğin bir bankacılık uygulaması geliştirdiğimizi düşünelim. Bu uygulamanın şüpheli durumlar oluştuğunda, kullanıcıya uyarı vermesini bir özellik olarak eklemezsek güvensiz bir uygulama geliştirmiş oluruz. Saldırgan eğer kullanıcı bilgilerini ele geçirdikten sonra giriş yaptığında, uygulama “Farklı konumdan giriş algılandı” şeklinde uyarı vermelidir.

Bu açığın önlenmesi için:

* Uygulama geliştirici ekibi siber güvenlik hakkında bilgili ve bilinçli kişilerden oluşturulmalıdır.
* Uygulama geliştirilme aşamasında, kodu kontrol edecek ve açıkları test edecek kişiler ekipte bulunmalıdır.
* Uygulamanın korunması için güvenli ve sektörde bilinen 3. Parti yazılımlar kullanılmalıdır.

5-Security Misconfigration:

Bu kategori web uygulamasının yanlış şekilde ayarlanmış veya bazı ayarların hiç ayarlanmamış olmasından kaynaklanmaktadır.

Bu güvenlik açığı yanlış yapılandırma ve güvensiz 3.parti uygulamaların kullanılması sonucu ve yazılım güncellemelerinin yapılmaması sonucu ortaya çıkar.

Örneğin web uygulamasının admin giriş bilgilerinin default olarak kalması ve ayarlanmaması veya açığa çıkan hata mesajlarının çıktılarının detaylı olması. Hata detayları sayesinde saldırgan daha sofistike saldırılar gerçekleştirebilir.

Bu açıkların önlenmesi için:

* Yapılandırmalar doğru şekilde yapılmalı.
* Admin ve yönetici hesaplarının giriş bilgileri güvenli hale getirilmeli.
* Yazılım güncellemeleri, güvenlik yamaları takip edilmeli.
* Güvenli 3.parti uygulamalar kullanılmalı.

6. Vulnerable and Outdated Components

Web uygulamaları birçok 3. Parti yazılım kullanmaktadır. Bu uygulamalarda da zafiyetler çıkabilmektedir. Bu uygulamalarda çıkan zafiyetlerin kapatılmaması ve bu uygulamaların güncellenmemesi sonucu ortaya çıkan zafiyetler bu kategoriye girer.

Bu kategori 2017 yılında 9. Sırada iken 2021 yılında 6.sıraya yükselmiştir. Yani bu açığın görülme sıklığı artmıştır.

Bu açığın önlenmesi için:

* Kullanılan 3. Parti yazılımların güncellemelerini ve açıklarını sürekli takip etmek gerekir.
* Loglama işlemi düzgün yapılmalıdır. Bu sayede sömürülen açık erken keşfedilebilir.

7**. Identification and Authentication Failures**

Bu kategorideki açıklar kullanıcı bilgilerinin doğruluğunu kontrol ederken veya kullanıcıya yetkilendirme yaparken ortaya çıkan sorunlardan dolayı oluşan açıklardır.

Web uygulamasının zayıf parolalara izin vermesi veya 2 aşamalı doğrulama gibi yöntemleri kullanmaması sonucu bu açıklar ortaya çıkar.

Bu açığın önlenmesi için:

* Kullanıcıların zayıf parolalar oluşturması engellenmeli.
* Kullanıcı adı ve parolalar yanında İki faktörlü doğrulama kullanılmalı.

**8. Software and Data Integrity Failures**

Bir web uygulamasının yazılımında veya veri tabanında meydana gelen beklenmedik hatalar sonucunda ortaya çıkan açıklardır. Ortaya çıkan bu açıklar yazılım ve veri bütünlüğünü bozar.

Bu açığın ortaya çıkma nedeni, web uygulamasında kullanılan uygulamalar güncellenirken; güvensiz repolardan indirme yapılmasıdır.

Saldırganlar sahte uygulama sunucuları oluşturarak, web uygulaması güncellemesine zararlı yazılım koyabilir ve web uygulamasını ele geçirebilir.

Bu açığın önlenmesi için:

* Uygulamanın doğru kaynaktan geldiğine emin olmak için, dijital imza yöntemleri kullanmak.
* Uygulama içerisindeki verilerin düzenli olarak yedeklenmesini sağlamak.
* Güvenli kodlama yöntemlerini kullanmak.

**9. Security Logging and Monitoring Failures**

Bu kategori uygulama içerisindeki loglama ve izleme faliyetlerinin eksikliğinden dolayı çıkan açıkları ifade etmektedir.

Bu açığın ortaya çıkma nedeni uygulama içerisindeki meydana gelen ihlalleri ve şüpheli olayların loglanmaması, bu ihlallerin bildirilme eksikliğinden kaynaklanır.

Örneğin bir e-ticaret platformumuz olsun. Bu platformda müşterilere ait kredi kartı bilgileri dahil olmak üzere kritik bilgiler bulunur. Saldırgan sunucuya sızarsa bu olayın loglanması ve monitoring yazılımında uyarı oluşturulması gerekir. Eğer yapılmazsa saldırgan sistemden istediği bilgiyi çalar, şirketin haberi bile olmaz. Haberi olmadığı için de verilerin çalınmasını engelleyemez.

Bu açığın önlenmesi için:

* Düzgün şekilde loglanma yapılmalı.
* Güvenlik logları düzenli olarak incelenmeli.
* Kritik olayların gerçekleşmesi durumunda uyarı ve alarm sistemleri kullanılmalı.
* Güvenilir ve sektöre kendini kanıtlamış loglama yazılımlarının kullanılması.

**10.SSRF**

Server Side Request Forgery (Sunucu tabanlı istek sahteciliği) anlamına gelir.

Genelde saldırganın URL’deki kontrol edilmeyen veya filtrelenmeyen bir parametreyi kullanması sonucu ortaya çıkar.

Örneğin bir uygulamamız olsun. Bu uygulama çeşitli haber sitelerinden haberleri çekerek yansıtsın. Bunu işlemi yapan php uygulamasının URL kısmı: <https://yavuzhaber.org/feed.php?url=https://hurriyet.com> olsun. Saldırgan url parametresini değiştirdiğinde verdiği URL’e istek atıyorsa burada SSRF vardır. Saldırgan bu sayede sunucu adına istek atmış olur. Bu sayede direkt olarak WAF arkasındaki sunucunun IP adresini öğrenebilir. Veya url kısmını localhost olarak değiştirirse sunucu içerisindeki dosyaları okuyabilir.

Bu açığın önlenmesi için:

* Başka sitelerden veri çeken uygulamalar için, sadece belirli sitelerden veri çekmesini ve sadece o sitelere istek atmasını sağlamak için whitelist yöntemi kullanılmalı.
* İsteklerde kullanılan http protokolleri için whitelist uygulanmalı.