OWASP TOP 10

OWASP Top 10, geliştiriciler ve web uygulaması güvenliği için standart bir farkındalık belgesidir. Web uygulamalarına yönelik en kritik güvenlik riskleri hakkında geniş bir fikir birliğini temsil eder. Bu sayede uygulama geliştiren kişiler projelerinde bu hususlar başta olmak üzere dikkat edebilir ve önlemlerini önceden alabilirler. 10 maddeden oluşan bu listeyi incelemeye başlayalım.

A01:2021 - Broken Access Control (Bozuk Erişim Kontrolü)

Nedir?
Bir web uygulamasındaki yetki ve erişim kontrol mekanizmalarının düzgün bir şekilde uygulanmadığı durumları ifade eder. Bu güvenlik zafiyeti, kullanıcıların yetkisiz erişimde bulunmalarına olanak sağlar.

Neden Kaynaklanır?
Geliştiricilerden kaynaklı yapılandırma hatalarından kaynaklanabilir.

Türleri

- **IDOR (Insecure Direct Object References):** Yetkisiz bir kullanıcının, URL'deki veya formdaki bir parametreyi değiştirerek başka bir kaynağa erişmesi.
- **Missing Function Level Access Control:** Geliştiricilerin bazı işlevlere erişim kontrolü eklememesi veya yanlış yapılandırması sonucu.

Nasıl Önlenir?

- Herhangi bir erişim kontrolü uygulanmadan önce, kullanıcının yetkilerini kontrol etmeliyiz.
- Yetki kontrolü için çerezleri kullanmalıyız.
- Erişim kontrol mekanizmalarını sürekli test edin ve güncel tutmalıyız.

IDOR için örnek vermemiz gerekirse, bir web uygulamasında canlı destek hattı düşünelim. Yetkili kişi ile kullanıcı arasında geçen bir diyalog var ve bu www.example.com/support/234 adresi üzerinde gerçekleşsin. Başka bir kullanıcının da iletişimi /235 üzerinde gerçekleşsin. Kullanıcılar yetkileri olmadan başka kullanıcıların mesajlarına, diyaloglarına erişebiliyorsa buna IDOR diyebiliriz.

MFLAC için ise yönetici hesabının doğru yapılandırılmaması sonucu normal düzeydeki kullanıcının bu eksik yapılandırmayı istismar ederek yetki elde etmesi ve hassas bilgilere ulaşması ile sonuçlanır.

A02:2021 - Cryptographic Failures (Kriptografik Hatalar)

Nedir?
Verilerin veya iletişimlerin şifrelenmesinde kullanılan algoritmalarda veya uygulamalarda zayıflıkların olması anlamına gelir. Bu, saldırganların verileri ele geçirmesine veya iletişimi bozmasına izin verebilir.

Neden Kaynaklanır?
Güçlü şifreleme yöntemlerinin kullanılmaması, şifreleme anahtarlarının yanlış yönetimi veya şifrelenmiş verilerin doğru bir şekilde korunmaması bu zafiyete neden olabilir.

Nasıl Önlenir?

- Güçlü, güncel ve güvenilir şifreleme algoritmaları kullanılmalı.
- Şifreleme anahtarları güvenli bir şekilde depolanmalı.

Hash yöntemleri günümüzde çözümlenebildiği için parola ve şifrelerin bu yöntemle tutulması güvenliği çok fazla sağlamayacaktır. Basit parolalar da aynı şekilde güvensizliği getirecektir.

A03:2021 - Injection (Enjeksiyon)

Nedir?
Bir saldırganın bir web uygulamasına istenmeyen kod enjekte edebilmesi anlamına gelir. Bu, SQL enjeksiyonu, XSS ve komut enjeksiyonu gibi çeşitli biçimlerde olabilir.

Neden Kaynaklanır?
Girdi doğrulaması ve filtreleme eksikliği, kullanıcıdan gelen verilerin doğrudan işlenmesi bu zafiyetin başlıca sebepleridir.

Türleri:

- **SQL Injection:** Veri tabanına zararlı SQL komutları enjekte edilmesi.
- **Cross-Site Scripting (XSS):** Zararlı JavaScript kodlarının web sayfasına enjekte edilmesi.
- **Command Injection:** Komut satırı argümanlarına zararlı komutların enjekte edilmesi.

Nasıl Önlenir?

- Kullanıcı girdileri filtrelenmeli.
- SQLi için direkt sorgu değil API tarzı yöntemler kullanılmalı.
- Web uygulamasında verilen girdi direkt olarak çalıştırılmamalı.

Web uygulama zafiyetlerinde en çok aradığımız en çok istismar etmek istediğimiz zafiyetler muhtemelen enjeksiyon zafiyetleridir. Çünkü büyük etki gösterebilirler. SQLi zafiyeti web uygulamasına bağlı veritabanını ele geçirme ve hedef sunucuda komut yürütmeye kadar etkileri olabilir. Bu zafiyete karşı iyi önlemler alınmalıdır. XSS zafiyeti de hedef sistem üzerinde

kod çalıştırma, kullanıcı verilerini elde etme gibi etkilere sebep olabilir. Aynı şekilde XSS içinde önlemler alınmalıdır. Komut enjeksiyonu direkt olarak sunucu tabanlı kod yürütmemize olanak tanır. Web uygulamasında girilen değerlerin sunucu tarafında direkt komut olarak işlenmesiyle bu zafiyet ortaya çıkar. Kritik sonuçları olabilir.

A04:2021 - Insecure Design (Güvensiz Tasarım)

Nedir?
Bir web uygulamasının güvenliğinin tasarım aşamasında göz ardı edilmesi anlamına gelir. Bu, zayıf kimlik doğrulama, veri doğrulama eksikliği ve hatalı hata işleme gibi çeşitli sorunlara yol açabilir.

Neden Kaynaklanır?
Güvenlik önlemlerinin başlangıçta planlanmaması, geliştiricilerin güvenliği göz ardı etmesi veya yetersiz bilgi.

Nasıl Önlenir?

- Tasarım aşamasından itibaren güvenlik kontrol edilmelidir.
- Tehdit modelleme ve risk analizi yapılmalıdır.

Buna en basit olarak ilk akla gelen örnek hata mesajlarındaki zafiyetleri verebiliriz. Mesela PHP kodlarındaki hataların direkt olarak ekrana basılması buna örnektir. Dosyasının hangi dizinde çalıştığını, kaçıncı satırında hata çıktığını ve PHP sürümüne kadar bazı bilgilere hata mesajından erişebiliriz. Geliştiricinin ilk aşamalarda bunları kapatması beklenir. Ekstra olarak veritabanına kaydedilen verilerin olduğu gibi gönderilip kaydedilmesi de buna dahil edilebilir. Hassas veriler mutlaka ama mutlaka şifrelenmelidir.

A05:2021 - Security Misconfiguration (Güvenlik Yanlış Yapılandırılması)

Nedir?
Bir web uygulamasının veya sunucusunun güvenli bir şekilde yapılandırılmamış olması anlamına gelir. Bu, varsayılan parolaların kullanımı, güvenli olmayan protokollerin etkinleştirilmesi ve güncellemelerin eksikliği gibi çeşitli sorunlara yol açabilir.

Neden Kaynaklanır?
Varsayılan ayarların kullanılması, hatalı yapılandırmalar veya güvenlik güncellemelerinin uygulanmaması bu tür zafiyetlere neden olabilir.

Nasıl Önlenir?

- Uygulama ve sunucu yapılandırılmaları sürekli kontrol edilmelidir.
- Varsayılan ayarlar kullanılmamalıdır. Özellikle şifreler, parolalar değişmelidir.

Yukarıdaki açıklamalara ilave etmek istersek, çoğu yazılımın default olarak tanımlanan şifre ve parolaları aynıdır. En basitinden bir modem modelinin fabrika çıkışında tüm admin

panellerindeki kullanıcı adı ve şifreleri aynıdır. Bu da internetteki basit araştırmalar ile istismarlar gerçekleştirilebilir. Örneği buradan çoğaltabiliriz. Bizzat kendi tecrübemden örnek vermem gerekirse yabancı menşeili bir web sitesindeki yönetim panelinin giriş bilgileri "admin/admin" olarak bırakılmıştı. İlk denememde yönetici paneline erişim sağladım ve şirket içi AD yapısına benzer bir yapıya hükmetme yetkisine sahip oldum. Bu gibi kritik sistemlerde bu daha da önemlidir.

A06:2021 - Vulnerable and Outdated Components (Güvenlik Açığı Barındıran ve Güncel Olmayan Bileşenler)

Nedir?
Bir web uygulamasında kullanılan üçüncü taraf kütüphaneleri veya çerçeveleri gibi bileşenlerde güvenlik açıkları olması anlamına gelir. Bu, saldırganların bu açıkları kullanarak web uygulamasına sızmasına izin verebilir.

Neden Kaynaklanır?
Kullanılan bileşenlerin güncellenmemesi veya bilinmeyen zafiyetlere sahip olması bu tür risklere yol açar.

Nasıl Önlenir?

- Kullanılan tüm bileşenler güncel olmalıdır.
- Güvenlik yamaları düzenli olarak yapılmalıdır.

Burada da aklımıza ilk gelen şey zafiyetli PHP sürümleri. RCE barındıran PHP sürümleri ile çokça karşılaştık. Zafiyet içeren yazılım kullanmak sunucu güvenliğini tehlikeye atar. RCE tespit edilmiş PHP sürümünü kullanmak kötü niyetli kişilerin sunucuyu ele geçirmesine sebep olabilir. Benzer şekilde XSS'in çıktığı JS dilinde de zafiyetli JS kütüphaneleri ve frameworkleri mevcut olabilir. Saldırılardan korunmak için güncel yazılımlar ve sürümler kullanmalı, gelişmeleri takip etmeliyiz.

A07:2021 - Identification and Authentication Failures (Kimlik Doğrulama Hataları)

Nedir?
Bir web uygulamasının kimlik doğrulama mekanizmalarında zayıflıkların olması anlamına gelir. Bu, zayıf parolalar, çok faktörlü kimlik doğrulamanın eksikliği ve kimlik doğrulama bypass'ları gibi çeşitli sorunlara yol açabilir.

Neden Kaynaklanır?
Zayıf veya eksik kimlik doğrulama yöntemleri, bu tür zafiyetlerin ana sebebidir.

Nasıl Önlenir?

- Güçlü ve karmaşık parolalar zorunlu tutulmalı.
- Çok faktörlü kimlik doğrulama (MFA) kullanılmalı.

Bu zafiyetin istismarında kaba kuvvet saldırıları kullanılabilir. Bunu önlemek için rate limit tarzı ayarlar getirilmelidir. Çok faktörlü doğrulamalar gereklidir. CSRF yanı siteler arasında istek sahteciliği ile oturum yönetilebilir. Bunun gibi yöntemleri bu zafiyete örnek verebiliriz.

A08:2021 - Software and Data Integrity Failures (Yazılım ve Veri Bütünlüğü Hataları)

Nedir?
Bir web uygulamasının veya verilerin bütünlüğünü sağlama süreçlerinde zayıflıkların olması durumudur. Bu, zararlı yazılımların uygulamaya eklenmesi veya verilerin manipüle edilmesi gibi saldırılara yol açabilir.

Neden Kaynaklanır?
Yetersiz yazılım güncellemeleri, güvenli olmayan veri transferi yöntemleri veya kod imzalama süreçlerindeki hatalar bu zafiyete sebep olabilir.

Nasıl Önlenir?

- Uygulama bileşenlerini ve verilerin bütünlüğünü doğrulayan mekanizmalar kullanılmalı.
- Güvenilir kaynaklardan yazılım bileşenleri kullanın ve güncellemeleri düzenli olarak uygulayın.
- Kod imzalama ve doğrulama süreçleri iyileştirilmeli.

A09:2021 - Security Logging and Monitoring Failures (Güvenlik Günlüğü ve İzleme Hataları)

Nedir?
Bir web uygulamasının güvenlik olaylarını kaydetme ve izleme yeteneklerindeki eksikliklerdir. Bu, saldırıların tespit edilememesine ve olaylara geç müdahale edilmesine neden olabilir.

Neden Kaynaklanır?
Yetersiz loglama ve izleme mekanizmaları, logların düzgün bir şekilde depolanmaması veya analiz edilmemesi bu tür sorunlara yol açabilir.

Nasıl Önlenir?

- Güvenlik olaylarını detaylı bir şekilde loglanmalı.
- İzleme ve analiz araçlarını kullanarak loglar düzenli olarak incelenmeli.
- Anormal aktiviteleri tespit eden otomatik sistemler kurulmalı (MDR-EDR gibi).

Şüpheli davranışların tespiti için loglama çok önemlidir. Geliştiricinin veya sistem sahibinin gözünden kaçan şeyler saldırganların gözünden kaçmayabilir. Saldırganların hareketlerini takip ederek hangi kısımlara yoğunlaşmamız ve geliştirmemiz gerektiğini tespit edebiliriz. Bu açıdan loglama ve logların incelenmesi, analiz edilmesi önem arz eder.

A10:2021 - Server-Side Request Forgery (SSRF - Sunucu Taraflı İstek Sahteciliği)

Nedir?
Bir saldırganın, hedef sunucuya istek göndererek, sunucunun iç ağındaki diğer sistemlere erişmesine olanak tanıyan bir güvenlik zafiyetidir.

Neden Kaynaklanır?
Kullanıcının girdiği URL'leri doğrudan sunucudan istek olarak gönderen uygulamalar, bu tür zafiyetlere açıktır. Genellikle, sunucunun iç ağdaki kaynaklara erişimi sağlanmışsa bu zafiyet daha tehlikeli hale gelir.

Türleri:

- **Blind SSRF:** Kullanıcının girdiği verilerin doğrulanmaması nedeniyle sunucunun iç ağdaki kaynaklara erişmesi.
- **Non-Blind SSRF:** Sunucunun bir dış kaynağa gönderdiği isteğin yanıtını manipüle ederek iç ağda işlem yapılması.

Nasıl Önlenir?

- Kullanıcıdan gelen URL'ler veya diğer istek parametreleri sıkı bir şekilde doğrulanmalı.
- Sunucunun yalnızca izin verilen dış kaynaklarla iletişim kurmasına izin verilmeli.
- Güvenlik duvarı veya diğer ağ segmentasyon yöntemleri ile sunucu erişimi sınırlandırılmalı.

SSRF sunucu güvenliği açısından önemli bir yer tutar. Ana sunucu ile bağlantılı diğer sunucuların güvenliği için bazı önlemler alınmalıdır. Bunlara yukarıda zaten değindim. Örnek senaryo olarak iç ağda bulunan bir veritabanı sunucusuna erişim izinleri ve ksııtlamaları bulunabilir. Bunları istismar ederek yetkisiz erişim elde edebiliriz. SSRF zafiyeti için HTTP header kısmındaki Referrer başlığı önemlidir. Burada yapılacak olan değişimler istismarı sağlayabilir. Başka bir örnek olarak bir stok sorgulama api'si içeren sunucuya ana sunucudan giden isteği manipüle ederek başka hedeflere erişebiliriz. Bunun örneğini Port Swigger Web Academy sayfasındaki SSRF laboratuvarlarında çözmüştüm geçmiş zamanda.