OWASP TOP 10

Eren Ersoyluoğlu

Broken Access Control

Broken Access Control, yetkisiz bir kullanıcının erişmemesi gereken bir veriye erişmesi, değişiklik yapması, veya silmesidir.

Bu zafiyet session yönetiminin eksik veya yanlış yapılandırılmasından, rol tabanlı kullanıcı sistemlerinde gerekli kontrollerin bulunmamasından oluşur. Bu zafiyeti tetiklemenin yolları aşağıdaki gibidir;

- **IDOR:**
  Uygulama sistem içinde bulunan dosya yada veri tabanı gibi objelere kontrolsüz ve direkt bağlantı kurmasından kaynaklanır. Yetkisiz kullanıcı bu bağlantıları manipule ederek yetki yükseltir yada hassas bilgilere erişir.

- **URL parametrelerinin değiştirilmesi:**
  Geliştiriciler URL üzerinden parametre alırken gerekli kontrolleri ve kısıtlamaları koymaması durumunda zararlı kullanıcı bu zafiyeti kullanarak erişmemesi gereken verilere erişir.

- **API Sömürme:**
  Eğere geliştiriciler API' ler için gerekli güvenlik önlemlerini almazsa ve bunları ulaşılabilir şekilde bırakırsa zararlı kullanıcılar zafiyetli API' leri kullanarak yetkisi olmayan verilere erişebilir, değiştirebilir, yada silebilir.

Bu zafiyet kullanıcılara Principle of Least Privilege(En Az Ayrıcalık İlkesi) metodolojisi kullanarak sadece yapmaları gereken işleme yeticek yetkiye sahip olması ile engellenebilir.

Cryptographic Failures

Daha önce Hassas veri açığa çıkarılması olarak bilinen bu zafiyet kriptografik algoritmaların yanlış kullanılmasından gerçekleşir.

Şifreleme yöntemlerinin kullanılmaması, kriptografig fonksiyonların yanlış uygulanması, ve şifrelemede kullanılan anahtarların güvensiz bir şekilde saklanması bu zafiyete neden olur.

Bu zafiyeti engellemek için, hassas veriler gerekmedikçe sistem üzerinde saklamadan, güçlü şifreleme algoritmaları ve TLS gibi güvenlik politakaları kullanılabilir.

Injection

Injection saldırısı zararlı kodların sistem üzerinde etki yapıcak şekilde çalıştırılması ile oluşur, bu saldırının oluşmasının en büyük nedeni kullanıcı tarafından gelen girdilerin gerekli filtrelemeden geçmeyerek doğrudan sisteme gönderilmesinden kaynaklanmaktadır.

Injection kendi çinde birçok alt kategoriye ayrılır;

- **SQL Injection**
  Zararlı kullanıcının SQL kodunu sistemem göndererek database hakkında bilgi edinmesini sağlar.

- **Command Injection:**
  Zararlı kullanıcının sistem üzerinde shell, cmd, vb. Sistem komutlarının çalıştırması ile ortaya çıkan saldırı türü.

- **Cross-Site Scripting:**
  Zararlı kullanıcının web üzerinde JavaScript kodlarının çalıştırması ile ortaya çıkan saldırı türü

- **LDAP Injection:**
  Zararlı kullanıcıların LDAP sorgusunu manipule ederek o dizin üzerinde bilgi elde etmesi ile oluşan bir saldırı türüdür.

Injection saldırılarının önlemek için, veri doğrulamassı yapılmalıdır, kullanıcıdan alınan her input filtrelemeden geçirilerek zararlı kodların engellenmesi gerekir ve kullanıcılara gereğinden fazla yetki verilmeden en az etki ile erişim sağlanması gerekir.

Insecure Design

Bu zafiyet geliştiricinin uygulamayı tasarlama şeklinden kaynaklanır, eğer uygulama tasarlanırken gerekli güvenlik önlemleri göz önüne alınmazsa ve bunlar düzgün bir şekilde entegre edilmezse uygulama üzerinden hem kullanıcıları hemde sistemi tehlikeye atabilcek bir çok zafiyet ortaya çıkabilir.

Bu tarz zafiyetlerle karşılaşmamak için, gvenli uygulama geliştirme prosedürleri uygulanabilir, ve uygulamanın belirli versiyonlarında güvenlik testleri yapılabilir.

Security Misconfiguration

Bu zafiyet uygulama için gerekli olan güvenlik işlemlerinin tam olarak yapılmamasından yada uygulama canlıya alındıktan sonra default' ta gelen ayarları kullanılmasından, gerek duyulmayan özelliklerin sistemde açık olması, vb. Durumlarda kaynaklanır.

Bu zafiyetin önüne geçmek için düzenli olarak güvenlik kontrolleri yapılmalı ve gerek olmayan tüm özellikler kapatılmalı, aynı zamanda sisteme default olarak gelen parola, kullanıcı adı, vb. Bilgiler değiştirilmeli.

Vulnerable and Outdated Components

Bu zafiyetin sebebi artık güncelleme almayan yada eski sürüm kütüphane, framework yada yazılım dili kullanmaktır. Geliştiricin gerekli güvenlik altyapısına sahip olmaması zararlı kullanıcılar için birçok saldırı yöntemi oluşturmakta.

Bu zafiyetin oluşmaması için kullanılan teknolojilerde belli aralıklarla güncelleme kontrolü yapılarak, web uygulamaları için belli aralıklara otomatize edilmiş zafiyet testleri yapılarak önünde geçilebilir.

Identification and Authentication Failures

Bu zafiyet sistem yada uygulamanın doğru bir şekilde kullanıcı doğrulaması yapamamsı yada dışarıdan gelen brute force saldırılarına karşı yeterli güvenliğin sağlanmamasından oluşur.

Bu zafiyeti engellemek için 2FA gibi birden fazla doğrulama sistemi kullanılabilir, brute force saldırılarını engellemek için rate limit ve time out gibi kısıtlamalar konulabilir, ve her bir kullanıcı için tek kullanımlık session' lar oluşturulabilir.

Software and Data Integrity Failures

Bu hata çoğunlukla yazılımlarda kullanılan 3. Parti güvenilmeyen pluginler, kütüphaneler, veya modüllerden, yazılım süreci sırasında yapılan hatalardan dolayı yazılım bütünlüğünün sağlanamamsından kaynaklanır.

Bu hatanın önüne geçmek için kullanılan 3.parti pluginlerde güvenlik güncellemeleri takip edilebilir, güncel olan kaynaklar kullanılabilir ve yazılım süreci boyunca belli aralıklarla güvenlik açıkları kontrol edilebilir.

Security Logging and Monitoring Failures

Security Logging and Monitoring Failures zafiyetten daha çok sistemsel bi hatadır, sistemin gerekli log kayıtlarını üretememesi, yada bu logların görüntelenememesinden kaynaklanmaktadır.

Bu tarz hataların yaşanmasından dolayıda bi zararlı kullanıcının yaptığı eylemler izlenemez ve kullanıcı gizliliğini koruyarak eylemlerine devam eder.

Bu hatanın önüne geçmek için log kayıdı yapan cihazların ayarlamalarının yapılması ve sistemde bulunan log kayıtlarının SIEM gibi log inceleme uygulamalarında görünebilcek formatta kayıt edildiğine dikkat edilmesi gerekmektedir.

Server-Side Request Forgery (SSRF)

Bu saldırı kullanıcının girdiği bir URL' yi kullanarak sunucu üzerinde başka bir kaynağa istek gönderilmesi ile oluşur. Bu saldırı uygulama kullancı veriyi almadan önce gönderdiği URL üzerinde bir kontrol, sanitizasyon yada doğrulama yapmadığı için kaynaklanır. SSRF kendi için şu türlere ayrılır;

- **Standart SSRF Saldırısı:**
  Kullanıcının zararlı URL göndererek sistemden kaynak çekmesi ile oluşur.

- **Kör SSRF Saldırısı:**
  Bu saldırıda kullanıcı kaynağı doğrudan elde etmez fakat uygulamanın verdiği dönüş tepkiye göre bu kaynağın orda olup olmadığını keşif edebilir.

- **Zaman Tabanlı SSRF Saldırısı**
  Bu saldırıda Serverın yanıt süresine göre elde edilmeye çalışan kaynak hakkında bilgi edinilebilir.

SSRF saldırısını engellemek için, kullanıcıların yaptığı girdiler üzerinde bir kontrol kullanılabilir, iç kaynaklara erişimde whitelist, blacklist gibi önlemler alınabilir ve rol tabanlı erişim engeli getirilebilir.