



Web Güvenliği

OWASP TOP 10

 **UITSEC**
More Secure Than Ever

Web uygulamaları sızma testleri



OWASP TOP 10

 **UITSEC**
More Secure Than Ever

Web Uygulama Güvenliği ve Hacking Yöntemleri

Web uygulamalarında gerçekleştirilen hacking yöntemlerden en yaygın olanları OWASP kuruluşunun da yayınladığı listedeki zayıflıkların tetiklenmesi sonucu;

- Oturum bilgileri çalınabilir,
- Uzaktan yetkisiz kod çalınabilir,
- Sisteme ait bilgiler ifşa edilebilir,
- Kullanıcı manipüle edilebilir,
- Web uygulamanın çalıştığı sunucunun kontrolü tamamen ele geçirilebilir.

OWASP Top 10 - 2017
A1:2017-Injection
A2:2017-Broken Authentication
A3:2017-Sensitive Data Exposure
A4:2017-XML External Entities (XXE)
A5:2017-Broken Access Control
A6:2017-Security Misconfiguration
A7:2017-Cross-Site Scripting (XSS)
A8:2017-Insecure Deserialization
A9:2017-Using Components with Known Vulnerabilities
A10:2017-Insufficient Logging & Monitoring

Enjeksiyon (Injection)

Enjeksiyon açıklıkları komutun ya da sorgunun bir kısmında güvenilmeyen verinin yorumlayıcıya gönderilmesiyle meydana gelmektedir. Enjeksiyon saldırıları SQL, OS ve LDAP komutları üzerinden yapılmaktadır. Saldırganın program üzerinde güvenilmeyen girdiler kullanarak bu program üzerinde komut veya sorgu şeklinde işlenmesini sağlayarak programın çalışma şeklini manipüle eden ve geniş saldırı vektörlerinden oluşan saldırı yöntemidir.



Enjeksiyon SQL Injection

- <?php
 \$id = \$_GET['username'];
 \$query = "SELECT * FROM USERS WHERE username='+\$id+'";
 mysqli_query(\$query);
 ?>
- <http://zafiyetlisite.com/login.php?username=%20OR'1='1>
- mysql> SELECT * FROM USERS WHERE username="" OR '1='1';



Etkileri: Bu saldırı verinin kaybına, bozulmasına, erişimine engellenmesine, güvenirlüğünün kaybedilmesine ya da erişim yetkisi olmayan kişilerin eline geçmesine yol açabilir. Bazı durumlarda sunucunun ele geçirilmesine de sebep olabilir.

Saldırganın program üzerinde güvenilmeyen girdiler kullanarak bu program üzerinde komut veya sorgu şeklinde işlenmesini sağlayarak programın çalışma şeklini manipüle eden ve geniş sınıflarda vektörlerinden oluşan saldırı yöntemidir.

Bozuk Kimlik Doğrulama (Broken Authentication)

Uygulama fonksiyonlarının kimliklendirme ve oturum yönetimiyle ilgili fonksiyonlarının uygulanmaması sonucu meydana gelmektedir. Bu açılık üzerinden saldırganlar şifreler, anahtarlar, oturum jetonu (token) ya da diğer kullanıcıların bilgilerini tahmin etme şeklinde açılığı kullanabilmektedir.

- Kullanıcı kimlik doğrulama türü
- Şifre politikası
- Giriş-Çıkış İşlevi/Cache yönetimi
- Kimlik doğrulamanın atlatılabilmesi
- Brute Force (Kaba Kuvvet Saldırısı)
- Dictionary Attack (Sözlük Saldırısı)
- CAPTCHA kullanımı



probethis
jsfinder

Bozuk Kimlik Doğrulama

Nasıl Gerçekleşir: Saldırganların erişimi olan milyonlarca kullanıcı adı-parola kombinasyonlarını, varsayılan kullanıcı veya yönetici bilgilerini yada brute-force saldırısını kullanarak erişim hakkı kazanabilirler. Çoktan geçerliliği geçmiş oturum jetonlarının (token) uygulama tarafından kabul edilmesi, oturumların (session) doğru şekilde entegre edilmemesi ile bu bilgilerin çalınması veya tekrar tekrar kullanılması ile de gerçekleştirilebilir.

Örnek Senaryo: Uygulama oturum kapanma süresi doğru entegre edilmediği için halka açık bilgisayarlarda özellikle oturum kapatılmadığı zaman aradan saatler geçse bile aynı oturuma ulaşılabilir. Kimlik bilgisi doldurma (Credential stuffing) saldırısı ile eğer kullanılan uygulamanın bu tip saldırılara karşı koruması yok ise bu bilgiler ile manipüle edilebilir.



Bozuk Kimlik Doğrulama

Etkileri: Kullanılan uygulamanın kapsamına göre kimlik bilgileri hırsızlığı, para aklama ya da hassas bilgilerin paylaşılması gibi hasarlar verebilir.

Nasıl Önlenebilir: Broken Authentication açığından korunabilmek için brute-force, credential stuffing, stolen credentials, re-usable credentials saldırıları önleyici çok faktörlü doğrulama sistemlerinin entegre edilmesi gereklidir. Varsayılan bilgilerin mevcut olduğu hiçbir uygulamanın yayılmaması. Zayıf parolaların güçlendirilmesi. Başarısız giriş işlemlerini geciktirme ve kayıt altına alarak bosphorus saldırlardan korunmak.





Hassas Veriyi Açıkta Bırakma (Sensitive Data Exposure)

Web uygulamaların birçoğunda kredi kartları, vergi numaraları ve kimlik bilgilerini düzgün olarak korunmamaktadır. Saldırganlar da zayıf korunan verileri kredi kartı dolandırıcılığı, kimlik hırsızlığı ya da diğer suçlar için çalabilmekte ya da değiştirebilmektedir.

Index of /admin/backup

Name	Last modified	Size	Description
Parent Directory		-	
FTP_It.log	2012-10-25 08:20	63K	
database_connect.php	2012-10-25 08:22	298	
ib_dump.sql	2012-10-25 08:21	98K	
old_pass.txt	2012-10-25 08:22	6.3K	

Apache/2.4.2 (Win32) OpenSSL/1.0.1c PHP/5.4.4 Server at www.vubnweb.com
Port 80

Hassas Veriyi Açıkta Bırakma (Sensitive Data Exposure)

Nasıl Gerçekleşir: Saldırganın erişim anahtarı ele geçirmesi, man-in-the-middle saldıruları gerçekleştirerek veya verileri sunucudan, gönderme sırasında yada kullanıcılarından yazı formatında çalınması ile gerçekleşir. Veri tabanından alınan veriler daha sonra (GPU) grafiksel işlem birimi kullanılarak brute-force saldırısı ile kırılabilir.

Örnek Senaryo: Kredi kartı bilgileri kullanılan bir uygulama bu bilgileri şifreleyerek veri tabanında saklıyor. Ancak bu bilgiler kullanılacağı sırada otomatik olarak şifreleri çözülüyor bu da bilgilerin şifresiz biçimde ele geçirilebilmesine yol açıyor. Veri tabanı kullanıcı şifrelerini saklamak için basit şifreleme yöntemi ile saklaması ve başka bir açıktan dolayı bu bilgilerin ele geçmesi durumunda bu bilgilerin şifrelerinin kırılması durumu bu açıga örnek gösterilebilir.

Hassas Veriyi Açıkta Bırakma (Sensitive Data Exposure)

Etkileri: Bu açık korunması gereken tüm verilerin yüzeye çıkmasına sebep olur. Sağlık kayıtları, kimlik bilgileri, kredi kartı bilgileri gibi hassas verilerin açığa çıkmasına sebep olur.

Nasıl Önlenir: Uygulamada gerçekleştirilen veri işlemlerinin, saklanmasıının veya gönderilmesi işlemlerinin sınıflandırılması ile gizlilik yasalarına ve regülasyonları, iş gereksinimlerine göre düzenlenerek bu açıklardan kaçınılabilir. Hassas ve gereksiz bilgilerin saklanması yerine silinmesi. Tüm hassas bilgilerin şifrelenerek korunması.

XML External Entities (XXE)

Hedef sistem üzerine harici bir XML dosyası gönderebildiği ve çalıştırabildiği durumlarda oluşur. Sunucunun XML verisini parse ederken özel olarak tanımlanmış bir XML varlığının (entity) çağırılması sonucu tetiklenir.



```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [ <!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd" >]>
<details>
    <subnet_mask>&xxe;</subnet_mask>
</details>
```

XML External Entries

Nasıl Gerçekleşir: Varsayılan olarak birçok eski versiyon XML işleyicisi dışarıdan XML girdisini kabul eder ve kabul etme işleminden sonra ayırtılmaya başlar. Enjeksiyon saldırısına benzerlik göstererek gerekli önlemler alınmadığı sürece XML uzantılı veya formatında uygulamayı veya sistemi varsayılan kullanma şekli dışında kullanabilir.

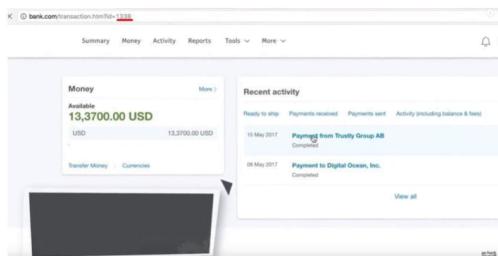
XML External Entries

Etkileri: Bu açık veri çalınması, sunucudan uzaktan form gönderme işlemleri, iç sistem hakkında bilgi toplama, DoS saldırısı veya açığın boyutuna bağlı olarak farklı saldırılarla gerçekleşebilir. Şirket boyutu uygulamanın ve etki ettiği verilerin durumuna, boyutuna göre değişkenlik gösterir.

Nasıl Önlenir: XML girişini kısıtlamak ve White listing ile kontrol altına almak. Girilen XML dosyaların içeriğini header kısımlarını inceleyerek filtrelemesini sağlamak. Web Application Firewall (WAF) sisteminin konfigürasyonunun ve izlenmesinin, gerektiği durumlarda de engelleyici önlemlerin alınması ile bu açığın engellenmesi sağlanabilir.

Çalışmayan Erişim Kontrol Sistemi (Broken Access Control)

Oturum açmış bir kişinin kendisi ile ilgili olmayan bir veriye ulaşması durumudur. Ortaya çıkan durumlar diğer kullanıcı hesaplarına erişim, hassas dosyalara erişim, diğer kullanıcıların verilerini değiştirmek, erişim haklarını değiştirmek gibi olaylardır.



Çalışmayan Erişim Kontrol Sistemi (Broken Access Control)

Nasıl Gerçekleşir: Genellikle erişim kontrolünün otomasyonu fark etmemesinden veya uygulama geliştiricilerin fonksiyonları etkili biçimde test etmemesinden kaynaklanır. Erişim kontrolü sistemleri genel olarak genellikle otomatik test etme ile uyumlu değildir, elle test etmek eksik veya hatalı özelliklerini belirleyemek için en ideal yöntemdir.

Örnek Senaryo: Ortak kullanılan bir bilgisayardaki oturumun özellikle kapatılması gereklidir. Eğer kapatılmazsa saatlerce bir işlem olmamasına ve hatta o sayfanın açılmasına rağmen kapanmaz ve bir sonraki kişi sizin oturumunuza erişebilir.

Kullanıcı oturumlarının cookie veya id gibi yöntem ile kontrol edilen bir uygulamada bu değişkenler değiştirilerek yollanırsa başka bir oturuma erişme hakkımız olur.

Güvenliğin Yanlış Yapılandırılması (Security Misconfiguration)

Uygulamanın güvenliğinin sağlanabilmesi için uygulamalar, çerçeveler, uygulama sunucusu, web sunucusu, veri tabanı sunucusu ve ortamlar için güvenlik ihtiyaçları bulunmalıdır, güvenli yapılandırma tanımlanmalıdır ve uygulanmış olması gerekmektedir



Güvenliğin Yanlış Yapılandırması (Security Misconfiguration)

Nasıl Gerçekleşir: Kullanılan yazılımın var olan güvenlik sistemlerinin veya özelliklerinin aktif edilmemesi, hata işleme (error handling) sisteminin fazla bilgilendirici mesajlar vermesi, gereksiz özelliklerin, portların, servislerin açık bırakılması. Sayfaların ve oturumlarının erişim izinlerindeki hatalar, gelen son güvenlik yükseltmelerinin yapılmaması.

Örnek Senaryo: Uygulamanın çalıştığı sunucu konfigürasyon sürecinde detaylı hata mesajları vermesi için ayarlanmış ise (varsayılan olarak çoğu sunucu bu şekildedir) ve uygulama kullanıma açıldıktan sonra bu ayar yapılmaz ise versiyon bilgisi ile o versiyonun bilinen açıkları kullanılarak saldırılara maruz kalınabilir veya bu çeşitli enjeksiyon saldırıları ile birleştirilerek hassas veriler çalınabilir.



Güvenliğin Yanlış Yapılandırması (Security Misconfiguration)

Etkileri: Bu tip açıklar genellikle saldırılara sistemin bazı özelliklerine ve fonksiyonlara erişim izinsiz erişim hakkı verir, bu fonksiyonları kullanarak sistemde yönetici kullanıcı olabilmesi ile sisteme erişimim ve sistem içerisindeki tüm verilerin ele geçirilmesi gerçekleşebilir.

Nasıl Önlenir: Sistemin veya uygulamanın kurulumu bittikten sonra güvenlik özelliklerinin aktif edilmesi, gereksiz belgelerin ve kullanıcıların kaldırılması, son güvenlik yükseltmelerinin yapılması, kullanılmayan ve gereksiz fonksiyonlar kapatılması gibi önlemler ile bu açık önlenebilir.



Cross-Site Scripting

Nasıl Gerçekleşir: HTML kodu dinamik olarak oluşturulduğunda, herhangi bir girdi temizlenmemiş girdi olduğunda ve bu girdi sayfaya yansığı durumlarda saldırın kendi HTML veya javascript kodunu girebilir. Üç farklı tipi vardır, bular;



Cross-Site Scripting

Yansıyan (Reflected) XSS: Uygulamanın kontrolsüz girdisini HTML sayfasında çıktı olarak gösterildiği durumlarda gerçekleşir. İl Başarılı bir saldırı kullanıcıların tarayıcıları üzerinde zararlı HTML, javascript kodları çalıştırmasına yol açabilir. Genellikle kullanıcının zararlı veya saldırganın kontrolünde olan bir sayfaya yönlendirildiği bir linke aracılığı ile yönlendirilmesiyle gerçekleşir.

Saklanan (Stored) XSS: Bu türün farkı saldırganın girdiği zararlı kodun uygulamada kaydedilerek uygulamanın yönetici dahil herkesin görmesi ve etkilenmesidir. Bu tür bir XSS yüksek veya kritik risk içerir.

DOM XSS: Bu XSS çeşidi HTML üzerinden değil de DOM (Document Object Model) objeleri üzerinden uygulanır. Reflected örneğindeki gibi kullanıcıya bir link bırakılır fakat kullanıcı linke girdiğinde bilgiler site üzerinden gönderilmez. Kullanıcının oturum bilgileri, saldırganın istediği farklı bir web sitesine gönderilir.

Cross-Site Scripting

Etkileri:

Reflected XSS: Kullanıcıların etkilenmiş bir sayfaya erişimi durumunda watering hole saldırısına yol açar.

Stored XSS: Bu açık kurbanın tarayıcısında komut çalışma gibi kritik saldırırlara sebebiyet verir.

DOM XSS: Kullanıcının oturumunun çalınması veya güvenli olmayan javascript kodlarının çakıştırılmasına yol açabilir.

Cross-Site Scripting

Nasıl Önlenir: XSS'in önlenmesi güvensiz bilginin tarayıcıdan ayrılmamasını gerektirir. XSS saldırısına karşı geliştirilen frameworklerin kullanılması. Güvenilmeyen girdilerin ayırtılması.



```
x-xss-protection 1; mode=block
```

Güveniksiz Seri Bozma İşlemi (Insecure Deserialization)

Bu zafiyet kullanıcıdan gelen güvenilmeyen zararlı girdilerin deserialization işlemi sonucunda oluşmaktadır. Bu zafiyet, denial of service saldırılara yada uzaktan kod çalıştırma saldırılara yol açabilmektedir.

Örneğin, bir PHP forumu; kullanıcı kimliğini, rolünü, şifresini içeren cerezi kaydetmek için PHP object serialization kullanıyor olsun;

- `a:4:{i:0;i:132;i:1;s:7:"Mallory";i:2;s:4:"user";i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}`
- Saldırısan, serileştirilmiş nesneyi kendisine admin ayrıcalığını vermek için değiştirebilir.
- `a:4:{i:0;i:1;i:1;s:5:"Alice";i:2;s:5:"admin";i:3;s:32:"b6a8b3bea87fe0e05022f8f3c88bc960";}`



Bilinen Açıklık Bileşenlerini Kullanma (Using Components with Known Vulnerabilities)

Yazılım modülleri, çerçeveler (frameworks) ve kütüphaneler gibi bileşenler, genellikle tam yetkiyle çalışmaktadır. Eğer bu bileşenlerin açıklıkları istismar edilebilir ise, saldırganların açıklıklar üzerinden sistem üzerinde, veri kaybına yol açmasına ya da sunucuya ele geçirmesini kolaylaştırabilir.

Uygulamalarda bilindik açıklıkları olan bileşenlerin kullanılması uygulamanın güvenlik seviyesini düşürebilir ve mümkün saldırı alanları ve etkilerine olanak sağlamaktadır.



Yetersiz kayıtlama ve izleme (Insufficient Logging & Monitoring)

Web uygulamaları, kullanıcı giriş, aksiyon ve aktivitelerini yeterli bir şekilde loglanmaması ve monitör etmemesi web uygulama güvenliği açısından risk oluşturmaktadır.



Bu açıktaki başarılı saldırılar zayıflığın keşfedilmesi sırasında kayıt altında tutulmama veya saldırı anında keşfedilmemeye dayanır.

Yetersiz kayıtlama ve izleme (Insufficient Logging & Monitoring)

Etkileri: Bir saldırı anındaörneğin Dos, DDoS gibi erişimin engellendiği saldırı cinslerinde herhangi bir alarm sistemi kurulu değil ise fark edilme süresi ve verdiği zarar artar. Herhangi bir saldırının gerçekleşmesinden sonra ise saldırganın kullandığı yöntemin veya hala içinde bir arka kapı (back door) bırakıp bırakmadığı gibi durumlarda da anlaşılmamasına yardımcı olur.



Örneğin, saldırganın kaba kuvvet saldırısı (brute force) sırasında log dosyalarının monitör edilmesi ve belirli bir yanlış giriş sonucunda aksiyonların alınması gereklidir.

Yetersiz kayıtlama ve izleme (Insufficient Logging & Monitoring)

Nasıl Önlenir: Bu açığın varlığını anlayabilmek için bir sızma testi (penetration testing) sonrasında test edilen uygulamanın veya sistemin tutulan kayıtlarına bakarak anlaşılabılır.



Bilgi Toplama



Reconnaissance

Bilgi Toplama aşaması bir test için en önemli aşamadır. Aktif ve pasif bilgi toplama aşamaları mevcuttur. Aktif bilgi toplama sistem ile bir iletişim gerektirip iz bırakır, pasif bilgi toplama ise hedef sistemler ile hiç iletişime geçmeden yalnızca internet ortamında bulunan sistemler yardımcı ile iz bırakmadan toplanır. Sistemi yakından tanımak, domainin alt domainlerini keşfetmek ve bu subdomainlerin hangi teknolojileri kullanıldığı, bu teknolojilerin sürüm bilgileri, sitenin alt dizinleri ve sitenin Google gibi arama motorları tarafından indexlenen sayfaları kayıt edilir.

