OWASP TOP 10 VE LAB ÇÖZÜMÜ

Melik Özdemir
M. Tolga Akbaba
Ali Rıza Zengince
w the help of @gbilge

yapıştır bi' <script>alert(42)</script> hocam

MACSEC

OWASP NEDİR?

Open Web Application Security Project.

Web uygulamalarındaki güvenlik açıklarının kapatılması ve bu uygulamaların güvenli bir şekilde korunmasını sağlamak için çalışmalar yapan özgür bir topluluk.

OWASP TOP 10

1- Broken Access Control
2- Cryptographic Failures
3- Injection
4- Insecure Design
5- Security Misconfiguration
6- Vulnerable and Outdated Components
7- Identification and Authentication Failures
8- Software and Data Integrity Failures
9- Security Logging and Monitoring Failures
10- Server-Side Request Forgery (SSRF)

Kavram Karmaşası

Broken Authentication -> Hatalı Kimlik Doğrulama

Broken Access -> Hatalı Erişim

1 - Broken Access Control

Yetki ve yetkilendirme kontrolünün yeterli bir şekilde yapamadığı durumlarda ortaya çıkan güvenlik açıklarıdır. Bilgi ifşalama, değiştirme veya silmeye kadar giden bir saldırı yüzeyi ortaya çıkarır. Kullanıcı yetkisinin de ötesinde *erişilmemesi gerekilen* veri ve yerlere erişilmesidir.

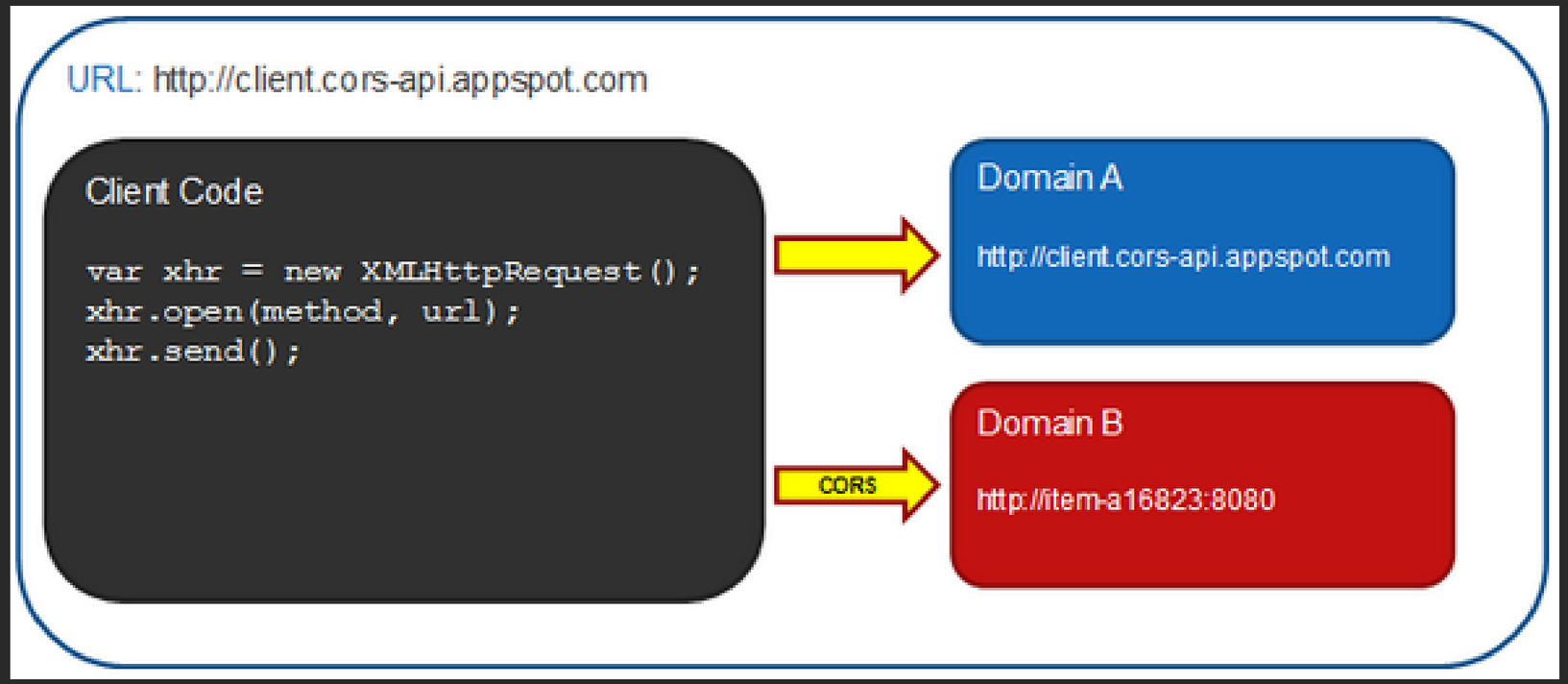
Nasıl/ne zaman sömürülür?

- URL değiştirerek erişim kontrollerini atlatma
- Birincil anahtar (Primary Key) değiştirme yetkisini elde etme
- Privilege elevation (Yetki yükseltme)
- Metadata değiştirme (JWT, cookie, gizli alanlar gibi)
-Hatalı CORS konfigürasyonu sonucu CSRF...

Korunmak için neler yapılabilir?

- Deny bu Default
- Erişim kontrol mekanizmaları, minimum CORS
- Gerçekleşen erişim kontrol hatalarını loglama (tekrarlı hata gözlemlenirse adminlerin uyarılması)
- API limiti koyma
- JWT, cookie gibi bilgilerin kısa ömürlü tutulması

Cross Origin Resource Sharing (CORS)



Same Origin Policy (SOP)

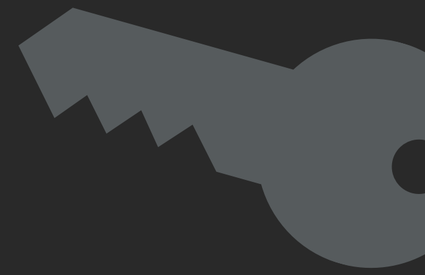
Cross Site Request Forgery (CSRF)

1. A hacker creates a request (in the form of a URL) for their own benefit from a website
2. Hacker embeds that request into a hyperlink and sends it to a visitor who they hope is logged in to the site
3. The website visitor clicks the link, unwittingly sending the request to the site
4. Assuming the request is legitimate, the website fulfills the request, sending data, funds, or access to the hacker

okta

2 - Cryptographic Failures (Kriptografik Hatalar)

Şifreleme hataları ya da yetersizliği sonucu ortaya çıkan güvenlik açıklarıdır. Hassas. verilerin açığa çıkması (sensitive data exposure) gibi kötü durumlara sebep olabilir.



Nasıl/ne zaman sömürülür?

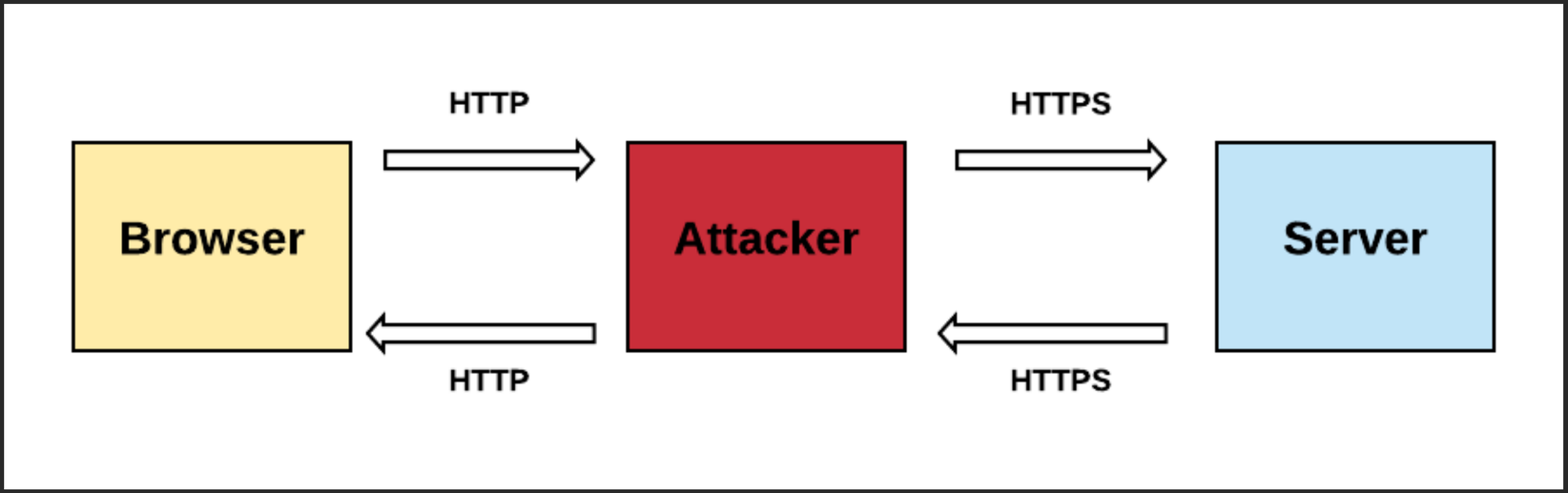
- Eski, zayıf kriptografik algoritmalar ya da padding metodları kullanıldığında
- Geliştirilmiş algoritma yeterince güvenli olmadığında
- Default şifreleme anahtarları kullanıldığında

Korunmak için neler yapılabilir?

- Güncel algoritmalar kullanılmalı
- Kriptografik algoritmalar dünyası yakından takip edilmeli
- Tahmin edilebilir parolalar kullanılmamalı
- Veriler gizlilik adına kategorize edilmeli, iletimi buna göre sağlanmalı
- Gerekmedikçe hassas bilgi depolamamalı
- Bilgi aktarımı için eski protokoller (FTP, SMTP gibi) kullanılmamalı
- Salted hashing fonksiyonları kullanılmalı

HTTP vs HTTPS

İndirgeme Saldırısı (Downgrade Attack)



3 - Injection

Injection metodolojilerinin oldukça fazla çeşidi vardır. Saldırı çeşitleri benzerdir. Sisteme girilen herhangi bir veri saldırı vektörü haline getirilebilir.

En sık karşılaşılan injection türleri SQL, NoSQL, OS command, Object Relational Mapping (ORM), LDAP, ve Expression Language (EL) ya da Object Graph Navigation Library (OGNL) injectiondır.

XSS, XXE, SQLi, dışarıdan dosya adı veya yolu değiştirmek gibi birçok şeye sebep olabilir.

Nasıl/ne zaman sömürülür?

- Kullanıcı tarafından gelen veri kontrol edilmediğinde (validate, filter, sanitize)
- Parametreleştirilmemiş çağrılar ve değişken (dinamik) istekler kullanılabildiğinde
- Kötü niyetli veri direkt ya da birleştirilerek kullanıldığında

Korunmak için neler yapılabilir?

- Sunucu taraflı girdi kontrolü politikaları uygulanmalı ve sıkılaştırılmalıdır
- API çözümü olarak güvenilir çözümlere gidilmeli, örnek vermek gerekirse direkt olarak kodu çalıştıran değil, kontrollü mekanizmalar kullanılmalıdır.
- Özel karakterleri göz önünde bulundurarak, varsa ignorelayarak yeni komut çalıştırılması ya da veri çekilmesi engellenmeli

SQL INJECTION

SQL injection veri odaklı uygulamalarda arka planda çalışan SQL dilinden faydalanılarak standart sorgulara ek SQL ifadeleri ekleyerek yapılan bir saldırı çeşididir.

SQL injection 3 ana kategoriye ayrılabilir:

1-In-Band SQLI (Classic)

a.Error-Based SQLI

b.Union-Based SQLI

2-Inferential SQLI (Blind)

a.Boolean-Based SQLI

b.Time-Based SQLI

3-Out-Of-Band SQLI

IN-BAND SQLI

1-Error-Based SQLI

Veri tabanının yapısı hakkında bilgi edinmek için
veri tabanı sunucusu tarafından atılan hata mesajlarına dayanan bir SQL Injection tekniğidir.

2-Unıon-Based SQLI

İki veya daha fazla “SELECT” ifadesinin sonuçlarını tek bir sonuçta birleştirmek için UNION SQL operatörünü kullanan ve daha sonra HTTP yanıtının bir parçası olarak döndürülen bir SQL Injection tekniğidir.

INFERENTIAL SQLI

1-Boolean-Based SQLI

Veri tabanına bir SQL sorgusu göndermeye dayanan ve sorgunun DOĞRU veya YANLIŞ sonucu döndürmesine bağlı olarak uygulamayı farklı bir sonuç döndürmeye zorlayan bir SQL Enjeksiyon tekniğidir.

2-Time-Based SQLI

Saldırgan, veri tabanına bir SQL sorgusu göndererek veri tabanının tepki verebilmesi için beklemesini sağlar. Saldırgan, veri tabanının yanıt vermesi için geçen süreyi, bir sorgunun doğru mu yanlış mı olduğunu görebilir.

OUT-OF-BAND SQLI

Bu teknik çok yaygın değildir, çünkü çoğunlukla web uygulaması tarafından kullanılan veri tabanı sunucusunda etkinleştirilen bazı özelliklere bağlıdır. Bu saldırıda saldırgan, saldırıyı başlatmak ve sonuçları toplamak için aynı kanalı kullanamadığında oluşur.

Out-of-band teknikleri, bir saldırgana özellikle de sunucu yanıtları çok kararlı olmadığı zamanlarda, inferential time-based tekniklere bir alternatif sunar,

Out-of-band SQLi teknikleri, veri tabanı sunucusunun bir saldırgana veri sağlamak için DNS veya HTTP talepleri yapma becerisine dayanır.

4 - Insecure Design (Yeterince Güvenilir Olmayan Tasarım)

İlgili sistem kendi yapısından, mimarisinden sömürülebilir özelliklere sahip olabilir. Bu bağlamda sistemin farkında olup tehdit modellemesi yapılmalıdır. Genel bir başlıktır.

!! Yeterince güvenilir olmayan bir tasarım "mükemmel implementasyon" ile düzeltilebilecek bir sıkıntı değildir, spesifik saldırılar güvenli sanılan sistem dizaynlarının bile düşünmediği noktalardan vuruyor olabilir.

Korunmak için neler yapılabilir?

- Güvenli tasarım kalıpları kullanılabilir.
- Secure Coding
- Tehdit Modellemesi
- Server side'dan client side'a güvenlik direktifi gönderilmesi
- Uygulama güvenliği ekibi ve geliştirici ekibin iletişiminin aktif ve verimli olması

Gereksinim ve Kaynak Yönetimi, Güvenli Tasarım (kültür ve metodolojisi), Güvenli Geliştirme Hayat Döngüsü (Lifecycle),

5 - Security Misconfiguration (Konfigürasyon Hataları)

Başlıkta da belirtildiği gibi yanlış konfigürasyon sonucu ortaya çıkan güvenlik açıklarını kapsayan bir maddedir. Bahsedilen açıkların sömürülmesi sonucu çok daha ciddi vakalarla karşılaşılabilinir.

Araştırmalara göre test edilen uygulamaların yüzde doksanında (90%) hatalı konfigürasyon bulunmakta.

Nasıl/ne zaman sömürülür?

- Bulut servisleri yanlış/eksik konfigüre edildiyse
- Gereksiz featurelar açıksa (port, servis, hesap, yetki vb.)
- Default kullanıcı ve parolalar varsa
- Hata işlenme süreçlerinde hata hakkında fazla bilgi veren yanıt mesajları varsa
- Kullanılan servislerin kendisi özünde güvenli değilse (ASP.NET, Spring gibi)
- Sunucu tarafında güvenlik header'ı vb yoksa

Korunmak için neler yapılabilir?

- KISS prensibi
- Yama yönetimi
- Tekrar eden, sürekli sıkılaştırma süreçleri
- Server side'dan client side'a güvenlik direktifi gönderilmesi

6 - Vulnerable and Outdated Components (Zafiyetli ve Tarihi Geçmiş Bileşenler)

Zafiyetli ve tarihi geçmiş bileşenlerin kullanılmasının neden yanlış olduğu kolayca tahmin edilebilir. Saldırganlar keşif ve bilgi toplama sırasında kullanılan sistem, ürün ve ara komponentlerin sürümünü de tespit edebilmektedir. İlgili durumlarda eski ve sömürülebilir zafiyet tespit edildiğinde metasploit gibi exploit frameworklerinden çekerek ya da kısa bir araştırma sonucu ilgili sistemdeki açığı araştırıp bularak spesifik saldırı durumuna geçilebilir veya zararlı yazılım saldırılarının radarlarına takılabilir.

Nasıl/ne zaman sömürülür?

- Kullanılan teknolojilerin sürümü bilinmiyorsa
- Yazılım üzerinde açık, desteklenmemiş veya tarihi geçmiş bileşenler varsa (OS, DBMS, API vb.)
- Sürekli bir zafiyet taraması gerçekleştirilmiyor veya ilgili konu yakından takip edilmiyorsa
- Zafiyet tespiti sonucu güncelleme geçilmiyorsa

Korunmak için neler yapılabilir?

- Gereksiz dependencyler, featurelar, bileşenler, dosyaların ve dokümantasyonlar kaldırılması
- Sistemler güncel tutulması (no shit sherlock)
- Sunucu ve istemci tarafındaki bileşenlerin sürümünün yakından takip edilmesi
- Yalnızca resmi ürünlerin indirilmesi ve ; kullanılması

7 - Identification and Authentication Failures

Eski adıyla Broken Authentication. Kullanıcı kimliğinin onaylanması, kimlik onayının gerçekleştirilmesi ve oturum yönetimi ile doğrudan ilgili olan saldırılardan korunmak adına kritik önem taşıyan konulardandır.

En yaygın kullanılan 5 şifre:
1- 123456
2- 123456789
3- 12345
4- qwerty
5- password

Nasıl/ne zaman sömürülür?

- Credential Stuffing varsa
- Kaba kuvvet saldırısı veya diğer otomatize saldırılar engellenmiyorsa
- Varsayılan, zayıf veya çok bilinen parolaların kullanımı babında herhangi bir aksiyon alınmıyorsa
- Parola sıfırlama mekanizmaları yeterince güvenli değilse
- Parolalar düz metin veya kolayca kırılabilir bir hashlenmiş formatta bulunuyorsa
- Çoklu doğrulama yoksa ya da yeterince verimli değilse
- Session bilgileri saldırganların kolayca erişebileceği bir şekilde tutuluyorsa (direkt olarak URL'de gibi)
- Aynı session IDsi çok kez kullanılabiliyorsa veya session ID kontrolünde sıkıntılar varsa

Korunmak için neler yapılabilir?

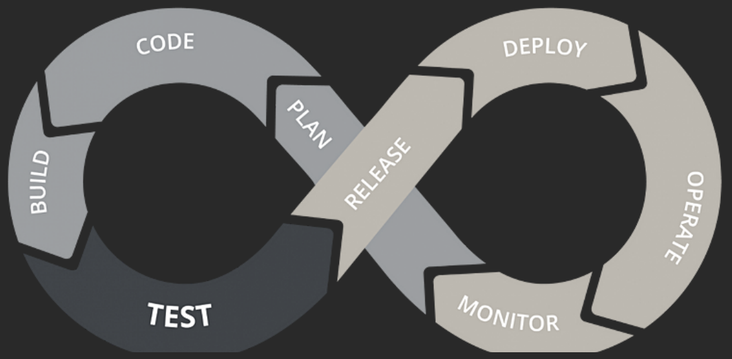
- Çoklu doğrulama sistemi yaygınlaştırılmalı
- Varsayılan kullanıcı hesapları kaldırılmalı veya değiştirilmeli
- Parola kontrol mekanizması getirilmeli
- Kayıt, hesap bilgisi yenileme ve API mekanizmalarının enumeration saldırılarına karşı sıkılaştırıldığından emin olunmalı
- Hatalı giriş sonucu verilen bilgiler detaylı olmamalı
- İlgili mekanizmaların loglama ve monitörlemesi yapılmalı.

8 - Software and Data Integrity Failures (Yazılım ve Veri Bütünlük Hataları)

Yazılım ve veri bütünlüğü hataları/eksiklikleri kapsamında bir sistemin kodu veya mimarisi o sistemi saldırılabilir hale getirebilmektedir.

Kaynağı bilinmeyen pluginler, kütüphaneler, modüller; repolar ve CDNler kullanılması bu riski artırır. CI/CD pipelineının güvenliği önemlidir. Autoupdate güzel bir feature olsa da can yakabilir.

- Verilerin değiştirilmediğinden emin olmak için imza mekanizmaları kullanılmalıdır.
- Kod ve konfigürasyon değişimleri sırasında inceleme mekanizmaları bulundurulmalıdır.



CI/CD Pipeline

9 - Security Logging and Monitoring Failures

Bir saldırı öncesi, eğer bilinmeyen bir metod değilse sistem üzerinde yüksek ihtimalle anomali (normal dışı) olarak tespit edilecek aktiviteler gözlemlenir. Bu yüzden sistemler üzerinde yeterince loglama (kayıt) veya izleme mekanizması bulunmaması sisteme gerçekleşen saldırının tespit edilememesine sebep olur.

Veri ihlallerini tespit etmek ve tespiti sonucu aksiyon almak bir kurum, kuruluş, kişi için oldukça önem taşımaktadır.

Nasıl/ne zaman sömürülür?

- Giriş denemeleri, hatalı girişler, yüksek değerli işlemler (alışveriş babında) gibi aksiyonlar loglanmadığında
- Loglar şüpheli aktivite adına incelenmediğinde
- Yalnızca lokalde log depolaması ve kontrolü yapılamadığında
- Alert mekanizmaları ve yanıt eskalasyon süreçleri yoksa veya yeterince iyi çalışmıyorsa
- Sistem real timeda response veremeyecek bir durumdaysa

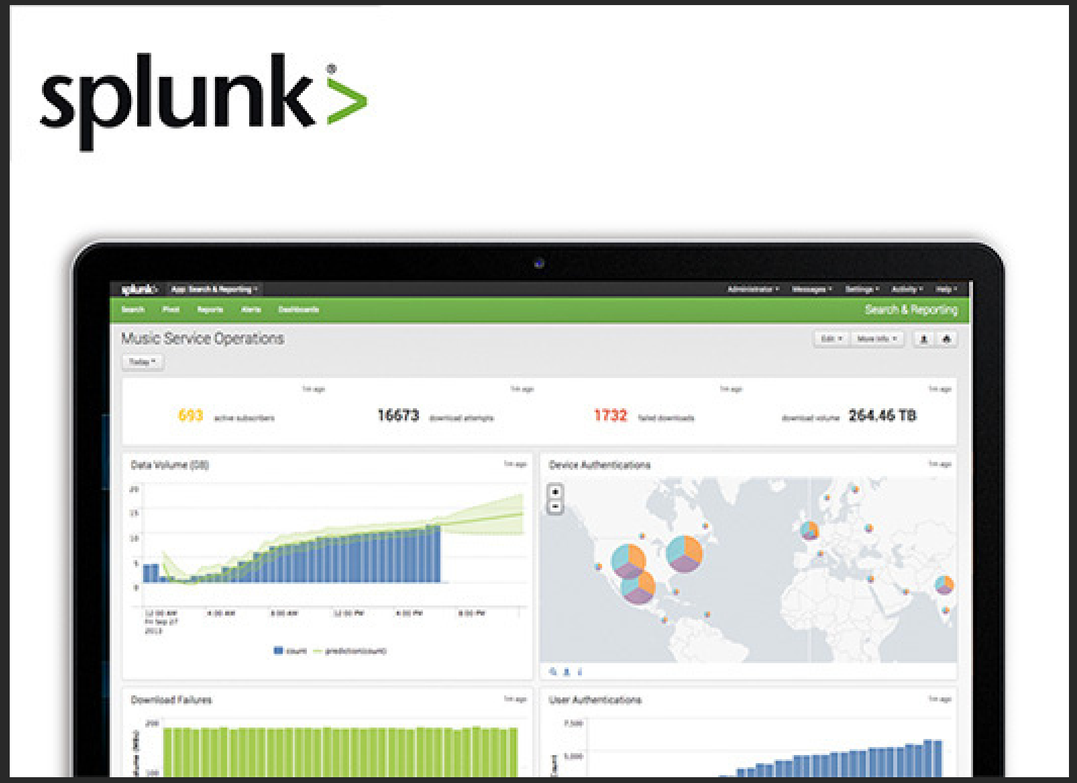
Korunmak için neler yapılabilir?

- Tüm giriş, erişim kontrolü, sunucu tarafı doğrulama hataları yeterli bilgi içerecek şekilde depolanmalı, gerekliyse ilgili kullanıcı adına analiz (forensic) yapılmalı
- Logların formatları log yönetim mekanizmaları tarafından kolayca işlenebilir olmalı
- IR & Recovery mekanizmaları olmalı
- DevSecOps

MACSECURITY |

9 - Security Logging and Monitoring Failures

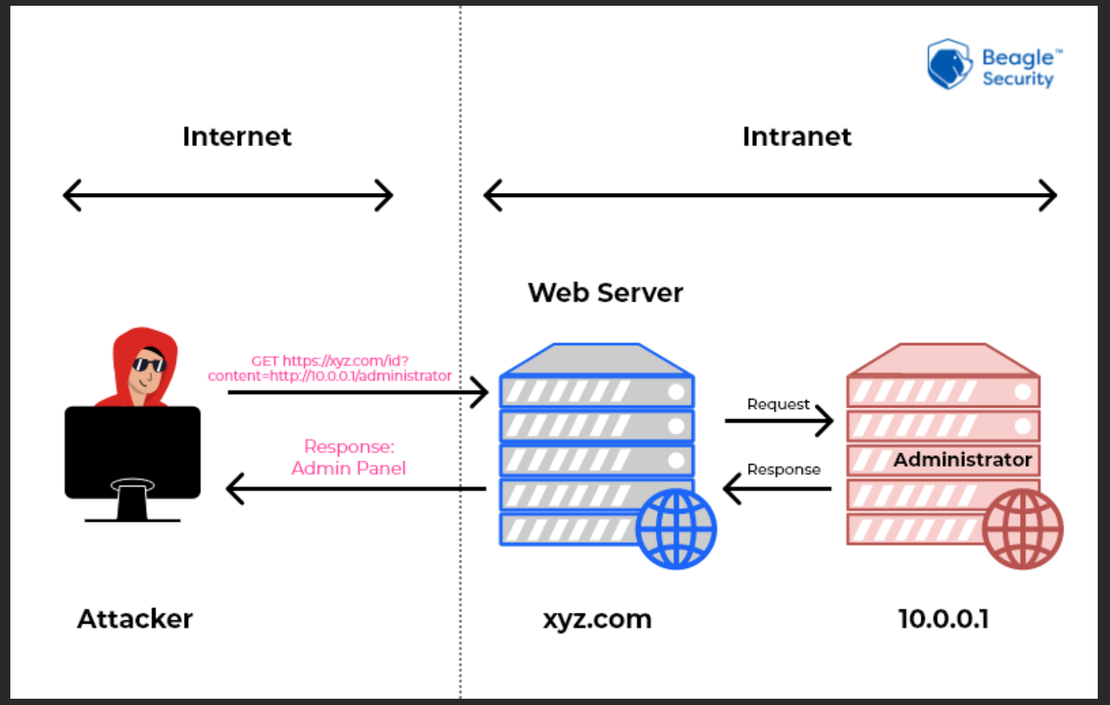
Splunk gibi log yönetim araçlarına yazılan kurallar, anomali tespiti sonucu ilgili yöneticileri pingleme ve tespit edilen bu durumun analiz aşamasına geçilmesi için aksiyon alınması adına kullanılmaktadır. SOC ekipleri genellikle benzeri bir mekanizma ile çalışır.



10 -SSRF (Server-Side Request Forgery)

SSRF açıkları, bir web uygulamasının kullanıcı URL girdisini kontrol etmeden uzak kaynaktan veri çekmesi sonucu ortaya çıkar. Saldırgan elle hazırladığı URL ile erişilmemesi gereken şeylere erişebilir.

Engellenmesi için ağ tarafında yapılabilecek önlemler arasında kaynak çekmek için kullanılan ağ ile farklı fonksiyonalitelerin olduğu ağların ayrıştırılması, iç ağda gerekliğ ağ trafiği hariç her şeyin varsayılan olarak bloklanması; web uygulamaları bağlamında gelen verinin kontrolü, HTTP redirectionunun engellenmesi, URL, port ve destination tanımlamalarının zorunlu bir formatta olması ve white list oluşturulması, TOCTOU farkındalığı gibi çözümler üretilebilir.



Sorularınızı alabiliriz :)

Bizi dinlediğiniz için teşekkür ederiz!
Şimdi lab çözümüne geçiyoruz :)