YAZILIM GEÇERLEME VE SINAMA

Yazılım Güvenliği

Yazılım Güvenliği Nedir?

Yazılım güvenliği, yazılımın yetkisiz erişim, değişiklik veya yok olma gibi tehditlere karşı korunmasını sağlar. Bu, yazılımın tüm yaşam döngüsü boyunca, yani tasarım, geliştirme, dağıtım ve bakım aşamalarında güvenlik prensiplerinin uygulanmasını içerir.

Yazılım Güvenliği Nedir?

Neden Önemlidir?

- Yazılım güvenliği, siber güvenlik tehditlerinin arttığı günümüzde, kullanıcıların ve kurumların bilgilerini korumak için hayati öneme sahiptir. Özellikle finans, sağlık ve devlet sektörlerindeki yazılımlar için güvenlik, kritik bir öneme sahiptir.

Yazılım Güvenliği Nedir?

- Güvensiz yazılımlar, sadece kullanıcıların bilgilerini tehlikeye atmaz, aynı zamanda kurumların itibarını da zedeler. Bu, maddi zararların yanı sıra hukuki sonuçlara da yol açabilir.

Yazılım Güvenliği Nedir?

Örnek: Bir bankacılık uygulaması düşünün. Eğer bu uygulama güvenli değilse, kullanıcıların finansal bilgileri tehlikede olabilir. Bu nedenle, yazılımın güvenliği, kullanıcıların bilgilerini korumak için hayati öneme sahiptir.

Yazılım Güvenliği Nedir?

Yazılıma karşı yapılan saldırılarının üç ana amacı bulunmaktadır:

- Yazılımın çalışmasını tamamen durdurmak ya da doğru bir şekilde çalışmasını engellemek,
- İçine zararlı kod dahil ederek veya yazılımın kodunu değiştirmek yoluyla, yazılımın çalışma akışını değiştirerek istenmeyen ya da öngörülmeyen bir amaca hizmet etmesini sağlamak,
- Yazılımın çalışmasındaki işlemler ve çevresiyle ilgili bilgi edinerek yazılımın zayıf noktalarını tespit etmek, çalışma ortamına sızmak ve istenilmeyen amaçları gerçekleştirmek için aracı olarak kullanmaktır.

Yazılım Güvenliği Prensipleri

Yazılım güvenliği prensipleri, yazılımın güvenli bir şekilde tasarlanması, geliştirilmesi ve bakımının yapılması için izlenmesi gereken temel kurallardır.

Yazılım Güvenliği Prensipleri

En Az Ayrıcalık İlkesi: Programların sadece gerektiği kadar ayrıcalıkla çalışmasını sağlama. Bu, bir hata veya güvenlik açığı durumunda zararın sınırlı kalmasına yardımcı olur.

Örnek: Bir web sunucusu, dosya sistemine sınırsız erişime sahip olmamalıdır. Sadece gerekli dosyalara erişim sağlanmalıdır.

Yazılım Güvenliği Prensipleri

- Bu ilke, programların sadece gerektiği kadar ayrıcalıkla çalışmasını sağlar. Eğer bir programın belirli bir görevi yapması için belirli bir ayrıcalığa ihtiyacı yoksa, bu ayrıcalık verilmemelidir.
- Eğer bir programın aşırı ayrıcalıkları varsa ve bu programda bir güvenlik açığı bulunursa, saldırgan bu ayrıcalıkları kötüye kullanabilir. Bu, sistemde daha büyük güvenlik ihlallerine yol açabilir.

Yazılım Güvenliği Prensipleri

- Sistem kaynaklarına erişmesi gereken uygulamalara yalnızca erişmesi gereken kısıtlı bölgeler için yetki verilmeli ve yönetici haklarıyla çalıştırılmamalıdır.
- Yükseltilmiş hakları gerektiren yazılım süreçleri bu haklara sadece gereken en az süre boyunca sahip olmalıdır.

Yazılım Güvenliği Prensipleri

Tüm Erişimleri Denetle İlkesi:

- Her bir öğeye yapılan her bir erişimde yetki kontrolü yapılmalıdır. Bu ilkeye uyum, sistem genelinde erişim denetimi sağlanmasını gerektirir.
- Normal durumların dışında başlatma, geri kazanım, kapatma, bakım safhalarında da erişim denetlenmelidir.
- Bir istek alındığında yapılan yetki kontrolü aynı isteğin yanıtı verilirken de yapılmalıdır.

Yazılım Güvenliği Prensipleri

Yetkileri Ayır İlkesi:

- Bir sistem yüksek güvenlikli bir işleme yalnızca tek bir koşula bağlı olarak izin vermemelidir.
- Yüksek güvenlikli bir işleme izin verilmesi için birden fazla koşulun sağlandığı doğrulanmalıdır.

Yazılım Güvenliği Prensipleri

Varsayılan Değerleri Güvenli Hale Getir İlkesi:

Kullanılan tüm varsayılan değerlerin güvenliği artıracak şekilde seçilmesi gerekmektedir. Bu konuda sık görülen hatalar şunlardır:

- Çalıştırılabilir dosyaların varsayılan olarak herkesin yazabileceği şekilde kurulması
- Uygulama ana dizininin herkesin okuyabileceği şekilde kurulması
- Herkesin yazabileceği iz kaydı dosyaları

Yazılım Güvenliği Prensipleri

- Herkesin okuyabileceği dosyalarda varsayılan (geliştirme, test) parolaların bulunması
- Herkes tarafından okunabilen dizinler
- Cihazlarda IP sahtekarlığına izin veren varsayılan ayarlar kullanılması
- Paylaşılan anahtar dosya / veri tabanlarında güvensiz hakların verilmiş olması

Bu gibi açıklıklara önlem olarak tüm varsayılan değerlerin gözden geçirilmesi ve güvenli değerlerin atanması gereklidir.

Yazılım Güvenliği Prensipleri

Ortak Erişilen Kaynaklara Farklı Kanallardan Eriş İlkesi:

Kaynaklara erişen mekanizmalar ortak kullanılmamalıdır (örneğin paylaşılan dosyalar, ortak port vb). Bu durum, bu kanallardan bilgi akışından kaynaklanan bilgi sızması/bozulması ve yan kanal saldırıları gibi problemlere neden olmaktadır. Ayrıca sistem ve yazılım içerisinde farklı bileşenler arasında da yalıtım sağlanmalıdır.

Yazılım Güvenliği Prensipleri

En Zayıf Halkayı Tespit Et ve Güçlendir İlkesi:

Bir yazılımın ilk saldırıya uğrayacak noktası, yazılımın en zayıf noktasıdır. Yazılımın güvenliği en zayıf halkanın güvenliği kadardır. Güvenlik analizi adımında en zayıf halka tespit edilmelidir. Bu halka kendisine yönelik riski karşılayabilecek şekilde yeniden tasarlanıp gerçeklenmelidir.

Yazılım Güvenliği Prensipleri

Saldırı Yüzey Alanını Azalt İlkesi:

Saldırı yüzey alanı, bir yazılım için yetkisiz bir kullanıcının (saldırgan) yazılım ortamından veri alabileceği, veri girebileceği veya yetkisiz işlem yapabileceği noktaların tümüdür. Uygulamaya gereksiz özellikler eklenmemelidir. İletişim yapılacak noktalar sınırlanmalıdır. Saldırı yüzeyi tasarım aşamasında belirlenmeli ve sürekli olarak izlenmelidir.

Yazılım Güvenliği Prensipleri

Savunma Derinliği Oluştur İlkesi:

Savunma derinliği, savunma mekanizmalarının katmanlı olarak uygulanmasından oluşur. Bu şekilde bir savunma mekanizması aşıldığında, sistem tamamen savunmasız kalmaz.

Yazılım Güvenliği Prensipleri

Bu bağlamda aşağıdaki hususlar dikkate alınmalıdır:

- Farklı üreticilerin/geliştiricilerin işletim sistemi, yazılım kütüphaneleri, güvenlik yazılımlarının kullanılması
- Yazılımdaki güvenlik mekanizmalarının, sistemdeki güvenlik mekanizmalarıyla entegrasyonu
- Siber güvenlik politikalarının tüm bileşenlerde dikkate alınması ve zorunlu olarak uygulanmasının sağlanması

Yazılım Güvenliği Prensipleri

Basit Güvenlik Mekanizması Tasarla İlkesi:

Güvenlik mekanizmaları mümkün olduğunca basit olmalıdır. Bu şekilde hata olasılığının daha az olması sağlanır, hatalar oluştuğunda anlaşılması ve düzeltilmesi kolaylaşır. Bu kapsamda özellikle ara yüzlerin mümkün olduğunca basit ve dokümante edilmiş olması gereklidir.

Yazılım Güvenliği Prensipleri

Anlaşılabilir ve Kolay Kullanılabilir Güvenlik Mekanizması Tasarla İlkesi:

Yazılımda kullanıcıyla etkileşim gerektiren güvenlik mekanizmaları anlaşılabilir ve kolay kullanılabilir olmalıdır. Aksi takdirde kullanıcı güvenlik mekanizmalarının etrafından dolaşmak için farklı yollar arayabilir.

Yazılım Güvenliği Prensipleri

Bu bağlamda aşağıdaki hususlar dikkate alınmalıdır:

- Kullanıcı işini en kolay şekilde ancak en az yetkiyle yapabilmelidir.
- Kullanıcı yapacağı işe ilişkin güvenlik politikalarını tanımlayabilmelidir.
- Kullanıcının izin verdiği eylemlere ilişkin yetkilendirme otomatik olarak verilebilmelidir.
- Kullanıcı daha önce verdiği yetkileri geri alabilmelidir.
- Kullanıcı verdiği yetkileri görebilmelidir.

Saldırı Türleri

Siber güvenlikte, saldırganların sistemlere veya uygulamalara zarar vermek veya yetkisiz erişim sağlamak için kullandığı çeşitli yöntemler vardır.

Saldırı Türleri

Enjeksiyon Saldırıları:

- Bu tür saldırılarda, saldırgan kötü amaçlı veriyi bir uygulamaya veya sisteme enjekte eder.
- SQL enjeksiyonu, bir saldırganın veritabanı sorgularını manipüle ederek veritabanına erişim sağlamasına olanak tanır. Bu, saldırganın veritabanındaki bilgilere erişmesine veya bu bilgileri değiştirmesine neden olabilir.

Saldırı Türleri

Enjeksiyon Saldırıları:

SQL enjeksiyon ile saldırgan tablo yaratabilir, değişiklikler yapabilir, veri tabanı üzerinde erişim sağlayabilir veya veri tabanı kullanıcısının hakları doğrultusunda sunucuda komut çalıştırabilir.

Saldırı Türleri

Kimlik Doğrulama ve Oturum Yönetimi Kusurları:

- Bu tür güvenlik açıkları, kötü amaçlı kullanıcıların yetkisiz erişim sağlamasına olanak tanır.
- Oturum kaçırma, bir saldırganın başka bir kullanıcının oturumunu ele geçirmesine olanak tanır. Bu, saldırganın o kullanıcının haklarına sahip olarak sisteme erişmesine neden olabilir.

Saldırı Türleri

Çapraz Site Scripting (XSS):

- Bu tür saldırılarda, saldırgan kötü amaçlı kodları bir web sayfasına enjekte eder.
- Yansıtıcı XSS saldırısında, saldırgan bir web sitesine kötü amaçlı bir JavaScript kodu ekler. Bu kod, bu siteyi ziyaret eden bir kullanıcının tarayıcısında çalıştırılır, bu da saldırganın o kullanıcının bilgilerine erişmesine neden olabilir.

Saldırı Türleri

Çapraz Site Scripting (XSS):

- Bu tür komutlar ile kullanıcıya ait site çerezleri alınabilir, kaydedilmiş şifreler çalınabilir veya tarayıcıda bulunabilecek güvenlik açıkları ile kullanıcı sistemi ele geçirilebilir.
- Ayrıca elektronik ticaret veya bankacılık uygulamaları için sahte giriş ekranları oluşturularak ziyaretçilerin yanıltılması ve sonucunda kullanıcıya ait önemli bilgilerin ele geçirilmesi mümkün olabilir.

Güvenli Kodlama Teknikleri

Güvenli kodlama teknikleri, yazılımın güvenli bir şekilde tasarlanması ve geliştirilmesi için izlenmesi gereken yönergelerdir.

Güvenli Kodlama Teknikleri

Girdi Doğrulama ve Sınırlama: Kullanıcıdan alınan verinin güvenli ve beklenen formatta olduğundan emin olma teknikleri.

- Uygulamanın tüm bileşenlerinde kullanılan değişkenler için kontroller oluşturulmalı ve değişkene atanması beklenen veri türü ile kullanıcı girdisi karşılaştırılmalıdır.

Güvenli Kodlama Teknikleri

- Beklenen girdi türünden farklı karakterler saptanması durumunda, karakterler SQL sorgularında anlam ifade etmeyecek biçimde değiştirilmeli, silinmeli veya kullanıcıya uyarı mesajı döndürülmelidir.
- Tercihen uygulamanın tamamı için geçerli olacak, değişken türü ve atanabilecek girdi türünü parametre olarak alan ve kontrolleri yaptıktan sonra girdi kabul sonucu üreten sabit bir fonksiyon tercih edilmelidir.

Güvenli Kodlama Teknikleri

Girdi Doğrulama ve Sınırlama

Örnek Durum: Bir web sitesinde kullanıcıların yorum bırakabildiği bir bölüm düşünün. Kullanıcılar bu bölüme her türlü metni girebilirler.

Kötü Kullanım: Kullanıcı, yorum olarak SQL sorgusu veya kötü amaçlı bir script girerse, bu sorgu veya script doğrudan veritabanında çalıştırılabilir.

Güvenli Kodlama Teknikleri

Hata Mesajlarını Sınırlama: Kötü amaçlı kullanıcılara fazla bilgi vermemek için hata mesajlarını genel ve belirsiz tutma teknikleri.

- Ayrıntılı hata mesajları, saldırganlara sistemin iç yapısı hakkında bilgi verebilir. Bu, saldırganların sisteme saldırmak için kullanabileceği bilgileri içerebilir.

Güvenli Kodlama Teknikleri

Hata Mesajlarını Sınırlama

Örnek Durum: Bir web sitesinde kullanıcıların giriş yapabildiği bir bölüm düşünün. Kullanıcılar bu bölüme kullanıcı adı ve şifrelerini girerler.

Kötü Kullanım: Eğer bir kullanıcı yanlış bir şifre girerse ve sistem "Yanlış şifre" şeklinde bir hata mesajı verirse, bu saldırganın doğru bir kullanıcı adı girdiğini anlamasına yardımcı olabilir.

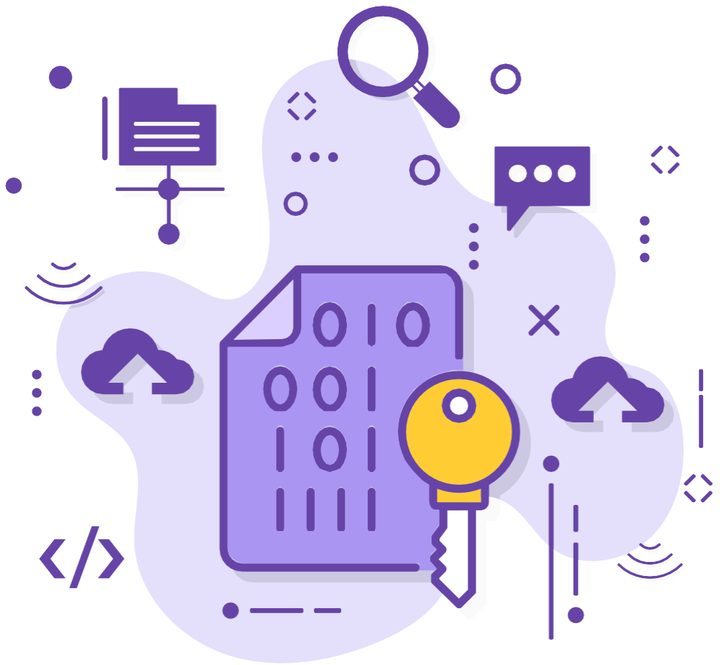
```python
# Flask kütüphanesinden Flask ve request modüllerini içe aktarırız.
from flask import Flask, request

# Flask uygulaması oluşturulur.
app = Flask(__name__)

# '/login' URL yoluna POST isteği geldiğinde çalışacak fonksiyonu tanımlarız.
@app.route('/login', methods=['POST'])
def login():
    # Kullanıcının formdan gönderdiği 'username' isimli veriyi alırız.
    username = request.form.get('username')
    # Kullanıcının formdan gönderdiği 'password' isimli veriyi alırız.
    password = request.form.get('password')
    # Kullanıcı adı 'admin' ve şifre 'password123' ise giriş başarılı mesajı döndürürüz.
    if username == 'admin' and password == 'password123':
        return "Logged in!"
    # Kullanıcı adı 'admin' ise fakat şifre yanlışsa yanlış şifre mesajı döndürürüz.
    elif username == 'admin':
        return "Wrong password!"
    # Kullanıcı adı 'admin' değilse kullanıcı bulunamadı mesajı döndürürüz.
    else:
        return "User not found!"
```

Bu kodda, kullanıcı adı doğru fakat şifre yanlışsa "Wrong password!" mesajı veriliyor. Bu da saldırganın doğru bir kullanıcı adı girdiğini anlamasına yardımcı olabilir.

Çözüm: Hata mesajlarını genel ve belirsiz tutarak bu tür bilgi sızıntılarını engelleyebiliriz. Örneğin, "Yanlış kullanıcı adı veya şifre" şeklinde bir mesaj kullanabiliriz.

Güvenlik Araçları ve Uygulamaları

Güvenlik araçları, yazılımın güvenliğini artırmak ve potansiyel güvenlik açıklarını tespit etmek için kullanılır.



Güvenlik Araçları ve Uygulamaları

- **Güvenlik Tarayıcıları:**
  - Web uygulamalarının güvenlik açıklarını tespit etmek için kullanılan araçlar.
  - **OWASP ZAP** gibi güvenlik tarayıcıları, bir web uygulamasının potansiyel güvenlik açıklarını tespit etmek için kullanılır. Bu araçlar, uygulamanın güvenliğini artırmak için kritik bir öneme sahiptir.

ZAP

ÖZET

- Çapraz Site Scripting (XSS), saldırganların kötü amaçlı kodları bir web sayfasına enjekte etmesidir.
- Kimlik Doğrulama Kusurları, kötü amaçlı kullanıcıların yetkisiz erişim sağlamasına olanak tanır.
- SQL Enjeksiyonu, saldırganların veritabanı sorgularını manipüle ederek veritabanına erişim sağlamasına olanak tanır.

ÖZET

- Hata Mesajlarını Sınırlama, kötü amaçlı kullanıcılara fazla bilgi vermemek için hata mesajlarını genel ve belirsiz tutma tekniklerdir.
- Güvenlik Tarayıcıları, web uygulamalarının güvenlik açıklarını tespit etmek için kullanılır.
- Parametreli sorgular, SQL enjeksiyon saldırılarına karşı koruma sağlar.
- Yazılım güvenliği, tehditlere karşı koruma sağlamak için kritik bir öneme sahiptir.