Güvenli Web Uygulamaları Geliştirmek

netsparker

whoami

- Alperen YILMAZ
- QA & Security Researcher at Netsparker
- Work & Study Remotely
- Anime, Manga, Moba, Bisiklet
- alperen@netsparker.com
- @alperenyilmaz54

netsparker

netsparker

AÇIK KAYNAK WEB UYGULAMALARININ GÜVENLİK DURUMU 2016

5 Yıl içerisinde taradığımız açık kaynak web uygulamalarında bulduğumuz zafiyetlerin istatistiği.

396 UYGULAMA TARANDI

netsparker

TARANMIŞ WEB UYGULAMALARI İLE TESPİT EDİLMİŞ ZAFİYETLERİN KARŞILAŞTIRILMASI

netsparker

EN POPÜLER 3 ZAFİYET TİPİ

Açık kaynak web uygulamalarında tespit edilen en popüler 3 zafiyet tipi:

XSS: 180

Reflected, Stored, URI, RFI ile XSS ve DOM XSS'i içermektedir.

SQL Injection: 55

Blind ve Boolean SQL Injection'ı da içermektedir.

File Inclusion: 16

Hem local hem de remote tipi atakları içermektedir.

netsparker

EN POPÜLER GELİŞTİRME DİLLERİ

Taranmış web uygulamalarında bulunan en popüler teknolojiler.

PHP: 326

ASP / ASP.NET: 31

Diğer 39 Uygulama; Java, JQuery, Ruby on Rails, Python gibi 10'dan fazla değişik teknolojiyle oluşturulmuştur.

netsparker

EN POPÜLER VERİTABANI SUNUCULARI

Bu web uygulamalarında kullanılan en popüler veri tabanı teknolojileri.

MySQL: 337

MSSQL: 29

SQLite: 5

MySQL

Microsoft SQL Server

SQLite

netsparker

DAHA GÜVENLİ WEB UYGULAMALARI İÇİN

Eğer özgür bir yazılım geliştiriyorsanız, hiçbir koşul olmadan **ÜCRETSİZ Netsparker Cloud Web Uygulaması Güvenlik Taraması** için info@netsparker.com adresine e-posta gönderebilirsiniz.

Daha fazla bilgi almak için websitemizi ziyaret edin **https://www.netsparker.com**

netsparker

Yazılımcılar Neden Güvenliğe Gereken Önemi Vermiyor?

netsparker

Yaygın Güvenlik Yanılgıları

- **Bir web uygulama geliştirme çatısı (web framework) kullanıyoruz, güvenlik konusunda endişelenmemize gerek yok**
- **Uygulamam ya da ben yeterince ilginç değiliz, kim beni neden hacklemek istesin**
- **Uygulamamın yedeğini alıyorum, endişe etmeme gerek yok**
- **Uygulamam SSL kullanıyor, yani güvendeyim**
- **Güvenlik duvarı (firewall) kullanıyorum, benim yerime güvenliği sağlıyorlar**

http://devnot.com/2017/yaygin-yazilim-guvenligi-yanilgilari/

OWASP

The Open Web Application Security Project

netsparker

OWASP

OWASP, Open Web Application Security Project'nin kısaltılmış halidir. Açık web uygulama güvenliği projesi anlamına gelen OWASP, güvensiz yazılımların oluşturduğu problemlere karşı mücadele etmek için kurulmuş bir topluluktur. OWASP'ın tüm araçları, dokümanları, listeleri, ve bölümleri ücretsiz olarak her yazılım güvenliği çalışanı ve meraklısına sunulmuştur.

OWASP TOP 10

netsparker

OWASP TOP 10

OWASP Top Ten projesinin amacı; web uygulamalarında en sık rastlanan on adet güvenlik açığı vektörünü adreslemek ve bu bağlamda güçlü bir farkındalık sağlamaktır. Proje hazırlanırken güvenlik üzerine araştırma / çalışma yapan bir çok firma ve kişilerin görüşleri doğrultusunda hareket edilmektedir.

OWASP Top Ten projesi; Microsoft, NSA, PCI Council, Citrix, Oracle, Imperva, Cenzic ve daha bir çok sektörün önde gelen firmaları tarafından aktif olarak kullanılmakta / referans alınmaktadır.

OWASP TOP 10 - 2013

- A1 Injection
- A2 Broken Authentication and Session Management
- A3 Cross-Site Scripting (XSS)
- A4 Insecure Direct Object References
- A5 Security Misconfiguration
- A6 Sensitive Data Exposure
- A7 Missing Function Level Access Control
- A8 Cross-Site Request Forgery (CSRF)
- A9 Using Components with Known Vulnerabilities
- A10 Unvalidated Redirects and Forwards

OWASP TOP 10 - 2013

A1 Injection:

Uygulamanın arka planında çalışan bir veri tabanı sorgusu ya da işletim sistemi komutuna, kullanıcıdan alınan ve ‘güvenilmeyen veri’ olarak tabir ettiğimiz veriler, herhangi bir doğrulama ve filtreleme işleminden geçmeden geliyorsa SQL Injection, Command Injection ya da LDAP Injection gibi zafiyetler doğabilir. Saldırganlar da uygulamada ki bu zafiyetleri kullanarak uygulama sunucusu ya da veritabanı sunucusunda komutlar çalıştırarak yetkilendirilmedikleri halde verilere erişim sağlayabilirler.

OWASP TOP 10 - 2013

A2-Broken Authentication and Session Management:

Kimlik doğrulama ve oturum yönetimi ile ilgili süreç ve fonksiyonların yanlış implementasyonu sonucu saldırganların, kullanıcıların kimlik bilgileri, şifreleri, oturum anahtarları vs gibi değerlerine ulaşması mümkündür.

- Session Fixation
- Session Prediction

OWASP TOP 10 - 2013

A3-Cross Site Scripting (XSS):

Web uygulamasında, kullanıcıdan girdi beklenen noktalarda, kullanıcıdan gelen girdiyi kontrol etmesi gereken fonksiyonların var olmaması ya da doğru implemente edilmemesi nedeniyle XSS açıklıkları doğabilir. Saldırganlar da bu zafiyeti sömürüp hedef kullanıcıların internet tarayıcıları üzerinde JavaScript kodları çalıştırarak oturum bilgilerini alabilir ya da zararlı yazılım içeren başka web sitelerine yönlendirebilirler.

OWASP TOP 10 - 2013

A4-Insecure Direct Object References:

Özellikle query string ile URL üzerinden taşınan değerler için tip, doğruluk ve benzeri gibi kontrollerin yapılmaması nedeniyle saldırganlar bu değerleri manipüle ederek uygulamaya zarar verebilir.

OWASP TOP 10 - 2013

A5-Security Misconfiguration:

Uygulama sunucusu, uygulamanın geliştirildiği uygulama çatısı (framework), web sunucusu, veri tabanı sunucusu ve benzeri platformların güvenlik gerekliliklerinin yerine getirilmemesi nedeniyle ortaya çıkan zafiyetlerdir.

OWASP TOP 10 - 2013

A6-Sensitive Data Exposure:

Hassas bilgilerin açığa çıkması

- Kredi kartı bilgilerinin ifşası
- TC Kimlik no gibi hassas bilgilerin ifşası
- Yorum satırlarında önemli bilgilerin unutulması

OWASP TOP 10 - 2013

A7-Missing Function Level Access Control:

Web uygulamasında yetkilendirmenin eksik veya yanlış yapıldığı alanlara yetkisiz erişebilme

OWASP TOP 10 - 2013

A8-Cross-Site Request Forgery (CSRF):

CSRF atakları sayesinde saldırganlar tarafından hazırlanan HTTP istekleri, hedef kullanıcının internet tarayıcısı aracılığıyla uygulamaya gönderilecek ve bu isteklere otomatik olarak ilgili kullanıcının oturum bilgileri dahil edileceğinden uygulama istekleri meşru olarak değerlendirecek ve saldırgan istediği sonuca ulaşabilecektir.

OWASP TOP 10 - 2013

A9-Using Components with Known Vulnerabilities:

Güvenlik açığı bulunduran güncel olmayan bileşenler kullanma

OWASP TOP 10 - 2013

A10-Unvalidated Redirects and Forwards:

Web uygulamaları sıklıkla kullanıcıları başka sitelere ya da mevcut uygulama üzerinde başka sayfalara yönlendirmektedirler. Bu yönlendirme işlemleri, herhangi bir doğrulama sürecinden geçmiyorsa, saldırganlar bu yönlendirme süreçlerini manipüle ederek hedef kullanıcıları zararlı yazılım içeren web sitelerine yönlendirebilmektedirler.

OWASP PROACTIVE CONTROLS

netsparker

OWASP PROACTIVE CONTROLS

Proaktif olmak, bir davranış özelliğidir. Olaylarda edilgen olup sonuçlardan ve başkalarından etkilenmek yerine olaylar olmadan önce olasılıkları düşünüp planlı bir şekilde harekete geçerek sonucu etkilemektir.

C1: Verify for Security Early and Often

C1: Verify for Security Early and Often

Genellikle güvenlik testleri, development sürecinin çok dışında ele alınıp, tespit edilen sorunlar fixlenmesi gereken issuelar olarak geliştiricilere görev olarak atanıyor.

Yazılım geliştirme süreçlerine dahil edilmeyen güvenlik prensipleri, maalesef scan-then-fix olarak özetleyeceğimiz kısır bir döngüye giriyor.

“Security is a process, not a product” - Bruce Schneier

C1: Verify for Security Early and Often

Vulnerabilities Prevented:

All OWASP TOP 10

C2: Parameterize Queries

C2: Parameterize Queries

Enjeksiyon zafiyetleri özetle, kullanıcılardan alınan girdilerin, herhangi bir kontrole tabi olmadan, sorgulara, komut bloklarına eklenmesi ile oluşan zafiyetlerdir.

DO NOT TRUST USER INPUT!11!!!!

C2: Parameterize Queries

"select * from users where id=".$_GET["id"];

http://www.example.com?getUser?id=1

select * from users where id=1

http://www.example.com?getUser?id=1 or 1=1 --

select * from users where id=1 or 1=1 --

C2: Parameterize Queries

Java Examples:

```
String newName = request.getParameter("newName");
int id = Integer.parseInt(request.getParameter("id"));
PreparedStatement pstmt = con.prepareStatement("UPDATE EMPLOYEES
SET NAME = ? WHERE ID = ?");
pstmt.setString(1, newName);
pstmt.setInt(2, id);
```

C2: Parameterize Queries

PHP Examples:

```
$stmt = $dbh->prepare(”update users set email=:new_email where id=:user_id”);

$stmt->bindParam(':new_email', $email);

$stmt->bindParam(':user_id', $id);
```

C2: Parameterize Queries

Python Examples:

email = REQUEST[‘email’]

id = REQUEST[‘id’]

cur.execute("update users set email=:new_email where id=:user_id",
{"new_email": email, "user_id": id})

C2: Parameterize Queries

.NET Examples:

```
string sql = "SELECT * FROM Customers WHERE CustomerId = @CustomerId";

SqlCommand command = new SqlCommand(sql);

command.Parameters.Add(new SqlParameter("@CustomerId",
System.Data.SqlDbType.Int));

command.Parameters["@CustomerId"].Value = 1;
```

C2: Parameterize Queries

Keyword : “Query Parameterization Cheat Sheet”

Risks Addressed:

OWASP Top 10 2013-A1-Injection

C3: Encode Data

C3: Encode Data

Bir sistemde anlam ifade eden karakterlere meta karakterler denilmektedir. Örneğin HTML sayfalarında <a> karakterleri bir link yaratır, <script></script> tagları bir script başlangıç ve bitişini ifade eder. Bu tarz anlam ifade eden karakterler, kelimeler bazen inputlar içerisine enjekte edilerek, sistemin farklı bir biçimde çalışmasına yol açabilir, daha da kötüsü bu yolla bir saldırı gerçekleştirilebilir.

Kullanıcı girdileri içerisine enjekte edilen bu tarz karakterleri zararsız formlarına dönüştürmek için Encoding mekanizması kullanılmaktadır.

C3: Encode Data

http://www.example.com?link=main.html

```
$link = $_GET["link"];

echo '<a href="$link">Click me!</a>";
```

C3: Encode Data

http://www.example.com?link=main.html" onclick="alert(1);

Click me!

C3: Encode Data

```
$link = htmlspecialchars($_GET["link"]); // Encode the input;
```

```
<a href="main.html&amp;quot; onclick=&amp;quot;alert(1);">click me!</a>
```

Encoding'i hem girdi işlemlerinde hem de çıktılarda kullanabiliriz. Girdi işlemlerinde encoding kullanarak, Command Injection, LDAP Injection gibi zafiyetlere karşı korunabilir; çıktıları encode ederek ise, Cross Site Scripting (XSS) zafiyetlerine karşı önlem almış oluruz.

C3: Encode Data

Vulnerabilities Prevented:

OWASP Top 10 2013-A1-Injection

OWASP Top 10 2013-A3-Cross-Site_Scripting_(XSS)

C4: Validate All Inputs

C4: Validate All Inputs

DO NOT TRUST USER INPUT!111!!!

Sisteme dışarıdan gönderilen ya da dışarıdan bir etkiye maruz kalan tüm veriler "güvensiz" olarak addedilmeli ve validasyon işlemine tabii tutulmalıdır.

Validasyon işlemi iki farklı şekilde yapılmalı. Hem biçimsel (syntax) hem de anlamsal (semantic) olarak.

C4: Validate All Inputs

Biçimsel (syntax) olarak validasyona tabii tutma:

Kullanıcıdan, bir kullanıcı id'si beklediğimizi varsayalım. Biçimsel olarak bu girdiyi denetlemek istersek, girdinin gerçekten de integer, yani tamsayı tipinde, ve örneğin 4 basamaklı olup olmadığını kontrol ederiz. Buna biçimsel (syntax) validasyon denilmektedir.

C4: Validate All Inputs

Semantik, yani anlamsal, olarak geçerliliği denetletmek ise, örneğin yukarıdaki işlemde kullanıcı id'sini, şifre sıfırlamak için istediğimizi varsayalım. Kullanıcının sağladığı kullanıcı id'si gerçekten de bu kullanıcıya ait olup olmadığını kontrol ederiz.

C4: Validate All Inputs

Blacklist (Kara Liste) Oluşturmak:

netsparker

C4: Validate All Inputs

Whitelist (Beyaz Liste) Oluşturmak:

- Sadece kabul edilecek değerlerin belirtilmesi
- A-Z aralığı gibi belirli aralıklarda olabilir

C4: Validate All Inputs

Vulnerabilities Prevented:

OWASP Top 10 2013-A1-Injection (in part)

OWASP Top 10 2013-A3-Cross-Site_Scripting_(XSS) (in part)

OWASP Top 10 2013-A10-Unvalidated_Redirects_and_Forwards

C5: Implement Identity and Authentication Controls

netsparker

C5: Implement Identity and Authentication Controls

Authentication kısaca, bir kişi ya da girdinin kendisi olduğunu iddia ettiği kimliğin doğruluk değerinin sınandığı bir işlemdir. Authentication bu imkanı sunduktan sonra, yani kişinin kimliğinden emin olduktan sonra bu kimliğin uygulamalarımızdaki yetkilerini tespit etme işlemi ise, Authorization olarak yani yetkilendirme olarak tanımlanmaktadır.

C5: Implement Identity and Authentication Controls

- 2FA veya MFA (Multifactor Authentication)
- Parolaları Şifrelemek (Encryption)
- Güvenli Şifre Sıfırlama Mekanizmaları
- Oturum Etkinleştirme ve Sonlandırma
- Uygulamadaki Önemli Özellikler İçin Tekrar Kimlik Doğrulaması Yapmak

Vulnerabilities Prevented:

OWASP Top 10 2013 A2- Broken Authentication and Session Management

C6: Implement Access Controls

C6: Implement Access Controls

- Tüm İstekleri Erişim Kontrol Noktalarına Yönlendirmek
- Ters Mizaçlı Olmak ya da Varsayılan Olarak Deny
- En Az Yetki Prensibi
- Kullanıcıya Güvenme!

Vulnerabilities Prevented:

OWASP Top 10 2013-A4-Insecure Direct Object References

OWASP Top 10 2013-A7-Missing Function Level Access Control

C7: Protect Data

C7: Protect Data

Datayı hem dolaşımda, hem de sunucu tarafında güvenli bir biçimde muhafaza etmeliyiz. Datanın aktarımındaki güvenliğini SSL gibi güvenli bir protokol kullanarak sağlayabiliriz. Datayı saklamak için ise güçlü kriptografik anahtarlar kullanılmalıdır. Daha da önemlisi, gerekmedikçe hiçbir data saklanmamalıdır. Hırsızlar, bizde olmayan bir şeyi çalamazlar

Vulnerabilities Prevented:

OWASP Top 10 2013-A6-Sensitive_Data_Exposure

C8: Implement Logging and Intrusion Detection

C8: Implement Logging and Intrusion Detection

Optimum bir loglama için, loglanacak veriler ne çok az ne de haddinden fazla olmalı. Zaman damgası, istemci kimliği (örneğin, IP) gibi olmazsa olmaz bilgiler tutulmalı; hassas bilgilerin saklanmamasına dikkat edilmelidir.

Loglamalarda iki husus özellikle güvenlik perspektifinden dikkate alınmalıdır. Bunlardan ilki log flooding ile servis dışı bırakma saldırılarına karşı gerekli denetim ve bakım yapılmalı; ikinci olarak da Log Injection saldırılarına karşı loglamadan önce girdilere mutlaka encoding işlemi yapılmalıdır.

C9: Leverage Security Frameworks and Libraries

C9: Leverage Security Frameworks and Libraries

Hep söylenegeldiği üzere güvenlik bir ürün değil bir süreç. Fakat bu süreçte başarısı kanıtlanmış ürünleri kullanmanın avantajlarını da yok sayamayız. Bu sebeple geliştirme süreçlerinde kullanılan frameworklerdeki güvenlik özelliklerini kullanmak bize zaman kazandıracaktır.

Netsparker'da SameSite Cookie attribute'ının implementasyonunu ya da tüm modern browserlar tarafından desteklenen Content-Security-Policy (CSP) önerirken, bu bilgilendirmeyi OWASP Proactive Controls 9. Madde ile sınıflandırıyoruz.

C10: Error and Exception Handling

C10: Error and Exception Handling

Hata yakalama ve istisna yönetimi güvenli kod geliştirmenin önemli bir parçası ve bu adımdaki hatalar, önemli güvenlik risklerine yol açabilir.

- Hassas bilgi ifşası
- Saldırganın sisteminiz hakkında bilgi edinmesini ve buna göre saldırılarını özelleştirmesini sağlar (işletim sistemi, versiyon bilgisi vb.)

Teşekkürler

- Ferruh Mavituna
- Onur Yılmaz
- Ziyahan Albeniz
- Emre İyidoğan
- Anıl Kurt
- SakaryaCoders ve tüm dinleyenler :)

netsparker