

php.testsparker Sızma Testleri Sonuç Raporu

Yazar: Ruhat Özgün Özcan

Testi Yapan: Ruhat Özgün Özcan

Tarih: 26.08.2022

İÇİNDEKİLER

1.Giriş.....	3
2.Web Uygulama Güvenlik Testleri	5
2.1 Güvenlik karakteri Güvenlik Önlemini atlatma	5
2.2 Tahmin Edilebilir/ Ön tanımlı Hesap Bilgisi Kullanımı	5
2.3 Kodlamadaki (Programlamadaki) Hata Mesajı:	6
2.4 SQL Injection.....	6
2.5 Depolanan Siteler Arası Script Çalıştırma	7
2.6 İşletim sistemi komut enjeksiyonu testleri.....	8
2.7 Local File Inclusion Zafiyeti (Windows)	9
2.8 PHP Evaluation(Code Evaluation) zafiyeti:	10
EK-1 Kullanılan Araçlar	11

GÖRSELLER

Şekil-1:Host adres görüntüleme	3
Şekil-2: nmap ile port tarama	3
Şekil-3:gobuster URL sonuçları.....	4
Şekil-4: whatweb ile pluginler sonucu.....	5
Şekil-5:Login Alanı	5
Şekil-6: hello.php deki hata mesajı.....	6
Şekil-7: Injection sonucu oluşan çıktı	7
Şekil-8: XSS Payload	7
Şekil-9:Script çalışması sonrasında çıkan ekran görüntüsü	7
Şekil-10: Enjeksiyon yapılabilecek alan	8
Şekil-11: LFI Payload	9
Şekil-12:Inclusion sonucu oluşan çıktı	9
Şekil-13: PHP Evaluation Payload	10
Şekil-14: Evaluation sonucu oluşan çıktı.....	10

1.Giriş

Bu rapor, techcareer.net'in düzenlemiş olduğu Cyber Security Bootcamp'in bitirme projesi için vermiş oldukları bir sızma testi raporudur. Bu raporda "php.testsparker" sitesindeki zafiyetler tespit edilip bunu raporlamayarak teslim edilecektir.

Öncelikle olarak bu web sitenden toplanabilecek verileri topluyoruz. İlk olarak ise "php.testsparker.com" un ip adresini host komutuyla bulabiliriz.

```
(kali@kali)-[~]  
$ host php.testsparker.com  
php.testsparker.com has address 107.20.213.223
```

Şekil-1:Host adres görüntüleme

Ardından bu web sitesinin açık portlarını tarayarak devam edebiliriz. Bunun için ise bize nmap yardımcı olacaktır.

```
(kali@kali)-[~]  
$ nmap -sC -sV 107.20.213.223  
Starting Nmap 7.92 ( https://nmap.org ) at 2022-08-24 05:40 EDT  
Nmap scan report for ec2-107-20-213-223.compute-1.amazonaws.com (107.20.213.223)  
Host is up (0.16s latency).  
Not shown: 998 filtered tcp ports (no-response)  
PORT      STATE SERVICE VERSION  
80/tcp    open  http    Apache httpd 2.2.8 ((Win32) PHP/5.2.6)  
|_http-title: Site doesn't have a title (text/html).  
|_http-server-header: Apache/2.2.8 (Win32) PHP/5.2.6  
443/tcp   open  http    Apache httpd 2.2.8 ((Win32) PHP/5.2.6)  
|_http-title: Site doesn't have a title (text/html).  
|_http-server-header: Apache/2.2.8 (Win32) PHP/5.2.6  
Service detection performed. Please report any incorrect results at https://nmap.org/submit/ .  
Nmap done: 1 IP address (1 host up) scanned in 88.02 seconds
```

Şekil-2: nmap ile port tarama

Böylelikle şimdiden bile bazı verilere erişebiliyoruz bunlardan biri kullandıkları server , php ve sürümleri hem de sitenin titlenin oladığını görebiliyoruz. Ardından iste site içersindeki diğer sayfları görebilmemiz için gobuster aracı yardımıyla ise websitesinin dizinleri arayabiliriz sonuç olarak ise bu dizinlerini bulabildik .

/index	(Status: 200) [Size: 136]
/images	(Status: 301) [Size: 335] [→ http://php.testsparker.com/images/]
/products	(Status: 200) [Size: 2712]
/3	(Status: 200) [Size: 311]
/page	(Status: 200) [Size: 4383]
/Images	(Status: 301) [Size: 335] [→ http://php.testsparker.com/Images/]
/Products	(Status: 200) [Size: 2712]
/test	(Status: 301) [Size: 333] [→ http://php.testsparker.com/test/]
/Index	(Status: 200) [Size: 136]
/redir	(Status: 200) [Size: 451]
/style	(Status: 200) [Size: 8916]
/conf	(Status: 200) [Size: 1]
/vendor	(Status: 301) [Size: 335] [→ http://php.testsparker.com/vendor/]
/Page	(Status: 200) [Size: 4383]
/robots	(Status: 200) [Size: 25]
/process	(Status: 200) [Size: 1380]
/'	(Status: 200) [Size: 0]
/auth	(Status: 301) [Size: 333] [→ http://php.testsparker.com/auth/]
/artist	(Status: 200) [Size: 1279]
/IMAGES	(Status: 301) [Size: 335] [→ http://php.testsparker.com/IMAGES/]
/%20	(Status: 403) [Size: 296]
/INDEX	(Status: 200) [Size: 136]
/hello	(Status: 200) [Size: 2764]
/*checkout*	(Status: 403) [Size: 305]
/Test	(Status: 301) [Size: 333] [→ http://php.testsparker.com/Test/]
/Style	(Status: 200) [Size: 8916]
/nslookup	(Status: 200) [Size: 3826]
/*docroot*	(Status: 403) [Size: 304]
/*	(Status: 403) [Size: 296]
/con	(Status: 403) [Size: 298]
/Hello	(Status: 200) [Size: 2764]
/Process	(Status: 200) [Size: 1380]
/Robots	(Status: 200) [Size: 25]
/internals	(Status: 301) [Size: 338] [→ http://php.testsparker.com/internals/]
/http%3A	(Status: 403) [Size: 300]

Şekil-3:gobuster URL sonuçları

Aynı zamanda <http://php.testsparker.com/> sitesinin pluginlerine de whatweb aracı sayesinde tespit edip kayıtlarımıza geçiriyoruz.

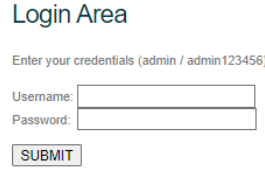
Detected Plugins:	
[Apache]	The Apache HTTP Server Project is an effort to develop and maintain an open-source HTTP server for modern operating systems including UNIX and Windows NT. The goal of this project is to provide a secure, efficient and extensible server that provides HTTP services in sync with the current HTTP standards.
Version	: 2.2.8 (from HTTP Server Header)
Google Dorks:	(3)
Website	: http://httpd.apache.org/
[HTTPServer]	HTTP server header string. This plugin also attempts to identify the operating system from the server header.
OS	: Windows (32 bit)
String	: Apache/2.2.8 (Win32) PHP/5.2.6 (from server string)
[PHP]	PHP is a widely-used general-purpose scripting language that is especially suited for Web development and can be embedded into HTML. This plugin identifies PHP errors, modules and versions and extracts the local file path and username if present.
Version	: 5.2.6
Version	: 5.2.6
Google Dorks:	(2)
Website	: http://www.php.net/
[Script]	This plugin detects instances of script HTML elements and returns the script language/type.
[X-Powered-By]	X-Powered-By HTTP header
String	: PHP/5.2.6 (from x-powered-by string)

2.Web Uygulama Güvenlik Testleri

2.1 Güvenlik karakteri Güvenlik Önlemini atlatma

Önem Derecesi : Orta

Yaptığım sızma testinde görebildiğimiz gibi kullanıcı giriş yapma esnasında bir CAPTCHA yani güvenlik testi aracı kullanılmamıştır. CAPTCHA bir kullanıcı oturum açmaya çalışırken rastgele karakterler çıkarılarak bunun kullanıcı tarafından girilmesine sebep olur. Bu araç saldırganların sistem üzerinde erişim elde edememeleri için uygulanan ek bir güvenlik önlemidir. Eğer bunun gibi test araçları kullanılmaz ise saldırganlar çeşitli otomatize araçlar kullanarak bu web formlarına sözlük ve kaba kuvvet saldırıları gerçekleştirebilirler, başkalarının hesaplarını ele geçirebilirler



Şekil-5:Login Alanı

Açıklığı barındıran Sistemler: <http://php.testsparker.com/auth/login.php>

Çözüm önerileri: Sisteme Captcha gibi güvenlik karakterleri eklenerek kontrol yapılarak oturum açılabilmesi.

Referanslar: <https://en.wikipedia.org/wiki/CAPTCHA>

2.2 Tahmin Edilebilir/ Ön tanımlı Hesap Bilgisi Kullanımı

Önem Derecesi: Kritik

Yapılan testlerde birçok kullanıcının adı ve şifresi varsayılan değerlerde bırakılmış ve birçok kullanıcı tahmin edilebilir kolay bir parola koymasına olanak sağlamıştır. Güvenli ve güçlü parolaların seçilmemesi durumunda, sistem ne kadar güncel olursa olsun güvenlik konusunda büyük bir zafiyetler meydana gelmiş olur. Sistemde yetki alan saldırganlar veya kötü niyetli çalışanlar servis dışı bırakmaya, gizli bilgilerin ifşasına ve benzeri kötü sonuçlara neden olabilir.

Kullanıcı Adı: admin

Parola: admin123456

Açıklığı barındıran Sistemler: <http://php.testsparker.com/auth/login.php>

Çözüm önerileri: Parola içerisinde büyük harf, küçük harf, rakam ve özel karakterlerin kullanılması önerilmektedir. Parolaların belirli periyodlarla değiştirilmesi önerilmektedir

Referanslar: <http://www.cyberciti.biz/tips/linux-security.html>

2.3 Kodlamadaki (Programlamadaki) Hata Mesajı:

Önem Derecesi: Orta

Uygulamada oluşabilecek hata kodları bazı bilgileri gösterebilir ve saldırganlar bu mesajlar doğrultusunda yeni saldırılar düzenleyebilir veya başlatmış oldukları saldırıyı daha kolay hale getirebilir, hızlandırabilir. Bundan dolayı ne olursa olsun bunlar açık bir şekilde kullanıcıya yansıtılmamalı bunun yerine hata mesajı için özel bir hata mesajı oluşturmalı. Sızma testinde de bu hata ile karşıladık ve raporluyoruz. Hata ise şu şekildedir:

Hello Service

Hello Visitor

Parse error: syntax error, unexpected T_STRING in C:\AppServ\www\hello.php(26) : eval()'d code on line

1

Şekil-6: hello.php deki hata mesajı

Açıklığı barındıran Sistemler: <http://php.testsparker.com/hello.php?name=Visitor>

Çözüm önerisi: Kodlama ortamında hata mesajlarını aynı şekilde web sitesine yansıtmak yerine hata mesajlarına referans verilerek kullanıcıya daha güzel bir görselle sunmak olabilir.

2.4 SQL Injection

Önem Derecesi: Yüksek

SQL Injection zafiyeti, uygulama parametreleri aracılığı ile yollanan bilgilerin düzgün kontrol edilmemesi sebebi ile arka planda çalışan veritabanına yollanan sorgulara, saldırganın sorgularını eklemesine imkan tanıyan bir güvenlik açığıdır. Hata Tabanlı SQL Injection saldırıları, uygulamanın veri tabanına gönderdiği sorgularda herhangi bir yazım hatası syntax error olması durumunda veya sorgunun veri tabanında çalışması sonucu dönen verilerin, ekrana çıktı olarak verilmesi temeline dayanır.

Yaptığım test sonucu link üzerinden SQL sorgusu yazdığım taktirde tüm artistleri bilgilerini veri tabanından çekip görebiliyoruz. Bu Injection yöntemiyle kullanıcılar url üzerinden tüm verilere erişim sağlayabilirler. Ekran görüntüsü ise şu şekildedir.

Results: 1 OR 1			
ID	Name	SURNAME	CREATION DATE
2	NICK	WAHLBERG	2006-02-15 04:34:33
3	ED	CHASE	2006-02-15 04:34:33
4	JENNIFER	DAVIS	2006-02-15 04:34:33
5	JOHNNY	LOLLOBRIGIDA	2006-02-15 04:34:33
6	BETTE	NICHOLSON	2006-02-15 04:34:33
7	GRACE	MOSTEL	2006-02-15 04:34:33
8	MATTHEW	JOHANSSON	2006-02-15 04:34:33
9	JOE	SWANK	2006-02-15 04:34:33
10	CHRISTIAN	GABLE	2006-02-15 04:34:33
11	ZERO	CAGE	2006-02-15 04:34:33
12	KARL	BERRY	2006-02-15 04:34:33
13	UMA	WOOD	2006-02-15 04:34:33
14	VIVIEN	BERGEN	2006-02-15 04:34:33
15	CUBA	OLIVIER	2006-02-15 04:34:33
16	FRED	COSTNER	2012-03-13 12:14:54 22
17	HELEN	VOIGHT	2012-03-13 12:14:54 22
18	DAN	TORN	2012-03-13 12:14:54 22
19	BOB	FAWCETT	2012-03-13 12:14:54 22
20	LUCILLE	TRACY	2012-03-13 12:14:54 22
21	KIRSTEN	PALTROW	2012-03-13 12:14:54 22
22	ELVIS	MARX	2012-03-13 12:14:54 22
23	SANDRA	KILMER	2012-03-13 12:14:54 22
24	CAMERON	STREEP	2012-03-13 12:14:54 22
25	KEVIN	BLOOM	2012-03-13 12:14:54 22
26	RIP	CRAWFORD	2012-03-13 12:14:54 22
27	JULIA	MCQUEEN	2012-03-13 12:14:54 22
28	WOODY	HOFFMAN	2012-03-13 12:14:54 22
29	ALEC	WAYNE	2012-03-13 12:14:54 22
30	SANDRA	DECK	2012-03-13 12:14:54 22

Şekil-7: Injection sonucu oluşan çıktı

Açıklığı barındıran Sistemler: <http://php.testsparker.com/artist.php?id=1%20OR%201>

Çözüm önerisi: Kodlama üzerinden bir GET istediği oluşturulurken bazı kısıtlamalar eklenerek GET sorgusunu url üzerinden istedikleri gibi yapamaması veya bu id'lerin şifrelenerek GET aksiyonu yapılabilir.

Referans: <https://linuxhint.com/sql-injection-kali-linux/>

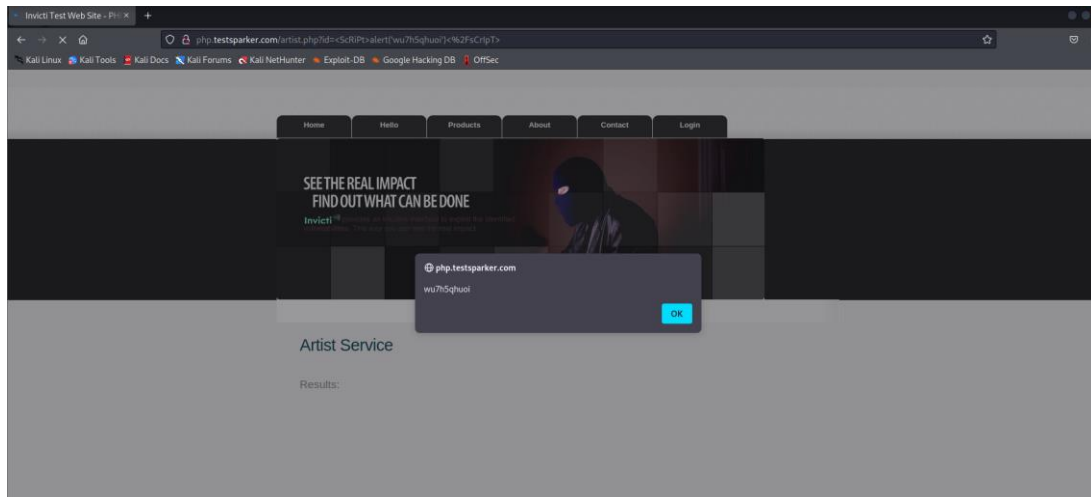
2.5 Depolanan Siteler Arası Script Çalıştırma

Önem Derecesi: Yüksek

Siteler arası script çalıştırma zafiyeti olarak bilinen XSS, kötü niyetli kişilerin bu site üzerinden diğer kullanıcılara istemci tarafında çalışmak üzere kod (genellikle JavaScript ve html) gönderip kötü amaçlarla çalışmalarına imkân tanımaktadır. XSS zafiyetleri uygulamalarda dışarıdan alınan bilgiler için yeterli girdi ve çıktı denetimi yapılmadığı durumlarda ortaya çıkar ve art niyetli bir kullanıcı istediği gibi javascript kodu çalıştırarak hedef aldığı kişilere ait oturum bilgilerini çalabilir, hedef aldığı kişilerin browserini istediği gibi yönlendirebilir. Testler sonucunda URL de yapılan değişiklikler sayesinde çıkan ekran görüntüsü şu şekildedir:

```
XSS vulnerability in http://php.testsparker.com/artist.php via injection in the parameter id
Evil request:
GET /artist.php?id=%3CScRiPt%3Ealert%28%27wu7h5qhuoi%27%29%3C%2FsCrIpT%3E HTTP/1.1
Host: php.testsparker.com
Referer: http://php.testsparker.com/process.php?file=Generics%2Findex.nsp
```

Şekil-8: XSS Payload



Şekil-9:Script çalışması sonrasında çıkan ekran görüntüsü

Açıklığı barındıran Sistemler:

<http://php.testsparker.com/artist.php?id=%3CScRiPt%3Ealert%28%27wu7h5qhuoi%27%29%3C%2FsCrIpT%3E>

Çözüm önerisi: Uygulama kodlarının gözden geçirilerek parametreler ve http başlığındaki diğer alanlar vasıtası ile yollanan her türlü bilginin kullanılmadan önce zararlı karakterlerden filtrelenebilir önerilmektedir. Uygulamalardaki bütün girdi ve çıktı noktalarından gelen değişkenler kontrole tabi

tutulmalı ve bu girdilerdeki bütün meta karakterler filtrelenebilir. Detaylı XSS önleme yöntemleri için aşağıda belirtilen referanslar incelenebilir.

Referans: <http://ha.ckers.org/xss.html> , <https://www.cgisecurity.com/xss-faq.html>

2.6 İşletim sistemi komut enjeksiyonu testleri

Önem Derecesi: Yüksek

OS Komut Enjeksiyonu, sistem üzerinde izinsiz komut çalıştırma yetkisini saldırganlara vermiş olur. Uygulamanın kurulumuna ve onu yürüten işlem yapılandırmasına bağlı olarak, işlemin ayrıcalık düzeyinin artmasına veya saldırganın tam etkileşimine izin veren uzak bir ters kabuk oluşturmaya neden olabilir. Aslında saldırgan sistemin izin verdiği ölçüde hemen hemen her şeyi yapabilir. Bu yüzden çok tehlikeli ve ciddi zararlara yol açan bir zafiyettir.

Sızma testi sonucunda ise <http://php.testsparker.com/nslookup.php> uzantısından komut enjeksiyonu yapabildiğini gördük

Products

IP Adress:

Server: ip-172-30-0-2.ec2.internal
Address: 172.30.0.2

Şekil-10: Enjeksiyon yapılabilecek alan

Açıklığı barındıran Sistemler: <http://php.testsparker.com/nslookup.php>

Çözüm önerisi: Girilen yazının formatının kontrol edilmesi. Girilen text için white list oluşturulabilir ev sadece bu kelimeler geçebilir. Header fonksiyonunun kullanımından sonra Die fonksiyonunun kullanılması, böylece sonraki girdilerin geçersiz kalması ve girdinin yalnızca alfanumerik karakterlerden oluşması yani sadece rakam ve harf kullanılması

Referans: <https://www.infinitumit.com.tr/os-command-injection-zafiyeti-nedir-ve-nasil-giderilir/>
https://owasp.org/www-community/attacks/Command_Injection

2.7 Local File Inclusion Zafiyeti (Windows)

Önem Derecesi: Yüksek

Local File Inclusion (LFI), çoğunlukla web sunucularında bulunan bir güvenlik zafiyetidir. Bu güvenlik zafiyeti, kullanıcının input(girdi) değerinde bulunan payload değeri ile sunucuda bulunan dosyaların içeriklerinin çıktı olarak web sitesine yansıtılmasıdır. Bu güvenlik açığı, hassas ve gizli verileri içeren dosyaları zafiyetli veya savunmasız sistemlerden okumak için kullanılabilir. LFI çoğu zaman önemli ve sınıflandırılmış verilere erişime (uygun izinler olmadan) yol açabilir. Bir saldırgan, password değerlerine, SSH anahtarları gibi hassas bilgileri okumak için LFI kullanabilir.

Bizde burada windowsun bazı belgelerini göstermeye olanak sağladık ekran görüntüsü ise şu şekildedir:

```
Windows local file disclosure vulnerability in http://php.testsparker.com/process.php via injection in the parameter Evil request:
GET /process.php?file=c%3A%5CWindows%5CSystem32%5Cdrivers%5Cetc%5Cservices%00.nsp HTTP/1.1
Host: php.testsparker.com
```

Şekil-11: LFI Payload



Şekil-12: Inclusion sonucu oluşan çıktı

Açıklığı barındıran Sistemler:

<http://php.testsparker.com/process.php?file=c%3A%5CWindows%5CSystem32%5Cdrivers%5Cetc%5Cservices%00.nsp>

Çözüm önerisi: Kullanıcı input değerinin uygun bir şekilde temizlenmesi. “disable_function” değerinde bazı parametreleri eklemek.

Referans: <https://www.netsparker.com.tr/blog/web-guvenligi/lfi-rfi-guvenlik-zafiyetleri-baglaminda-php-stream-wrapperlari/>

2.8 PHP Evaluation(Code Evaluation) zafiyeti:

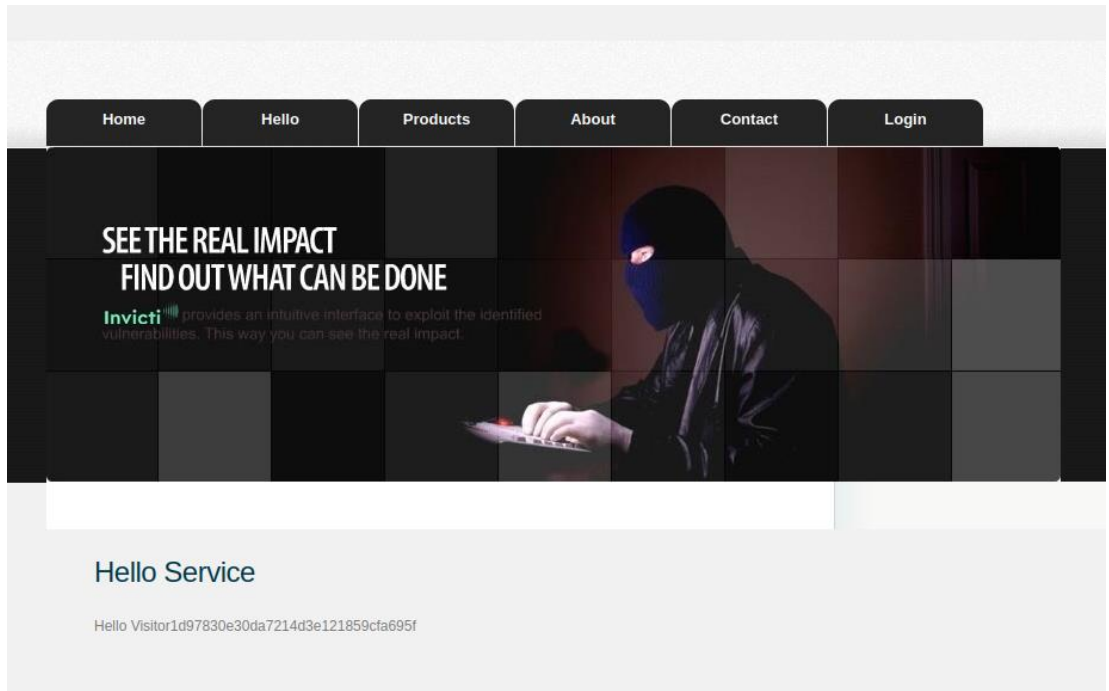
Önem Derecesi: Yüksek

“eval()” içine string olarak yazdığınız php kodunun çalıştırılmasını sağlıyor ve bu stringler geçerli bir PGP codu formatında olmalı ve sonu virgülle bitmelidir. PHP Code Evaluation is ebir saldırganın sunucu tarafı komut dosyası motoruna özel kod eklemesine izin veren bir güvenlik açığıdır. Bu güvenlik açığı, bir saldırgan bir eval() işlev çağrısına beslenen bir giriş dizesinin tamamını veya bir kısmını denetleyebildiğinde ortaya çıkar. Eval, argümanı kod olarak yürütür.

Ortaya çıkan ekran görüntüsü ise şu şekildedir. Md5 oluşturan bir PHP fonksiyonu yardımı ile.

```
PHP evaluation in http://php.testsparker.com/hello.php via injection in the parameter name
Evil request:
GET /hello.php?name=%3Bexit%28md5%28%27w4p1t1_md5%27%29%29%3B%2F%2F HTTP/1.1
Host: php.testsparker.com
```

Şekil-13: PHP Evaluation Payload



Şekil-14: Evaluation sonucu oluşan çıktı

Açıklığı barındıran Sistemler:

http://php.testsparker.com/hello.php?name=%3Bexit%28md5%28%27w4p1t1_md5%27%29%29%3B%2F%2F

Çözüm önerisi: Evaluate edilmiş kodun içinde kullanıcı girdilerini kullanmaktan kaçınmak iyi bir tercih olabilir. En iyi seçenek eval gibi fonksiyonları hiçbir zaman kullanmamak olurdu. Ayrıca ilgili dil tarafından ayrıştırılabilen dosya içeriklerinin kullanıcılar tarafından düzenlenmesine engel olmak gerekir.

Referans: <https://www.netsparker.com.tr/blog/web-guvenligi/remote-code-evaluation-zafiyeti/> , [https://wiki.owasp.org/index.php/Direct_Dynamic_Code_Evaluation_\('Eval_Injection'\)](https://wiki.owasp.org/index.php/Direct_Dynamic_Code_Evaluation_('Eval_Injection'))

EK-1 Kullanılan Araçlar

-gobuster: <https://github.com/OJ/gobuster>

-wapiti: <https://wapiti-scanner.github.io/>

-nmap: <https://nmap.org/>

-owasp: <https://owasp.org/>

-Zap : <https://www.zaproxy.org/>

-sqlmap : <https://sqlmap.org/>

-whatweb: <https://www.kali.org/tools/whatweb/>