Fingerprint Web Application Framework (Parmak İzi Web Uygulama Çerçevesi)

Summary (Özet)

Güneşin altında yeni bir şey yok ve gelişmeyi düşünebileceğiniz neredeyse her web uygulaması zaten geliştirildi. Dünya çapında aktif olarak geliştirilen ve konuşlandırılan çok sayıda ücretsiz ve Açık Kaynaklı yazılım projesiyle, bir uygulama güvenliği testinin tamamen veya kısmen bu iyi bilinen uygulamalara veya çerçevelere bağlı bir hedefle karşılaşması çok muhtemeldir (örneğin. WordPress, phpBB, Mediawiki, vb. Test edilmekte olan web uygulama bileşenlerini bilmek, test sürecinde önemli ölçüde yardımcı olur ve ayrıca test sırasında gereken çabayı büyük ölçüde azaltacaktır. Bu iyi bilinen web uygulamaları, uygulamayı tanımlamak için numaralandırılabilecek HTML başlıkları, çerezler ve dizin yapılarına sahiptir. Web çerçevelerinin çoğu, bir saldırganın veya testçinin onları tanımasına yardımcı olan bu yerlerde birkaç belirteceğe sahiptir. Temel olarak tüm otomatik araçların yaptığı şeydir, önceden tanımlanmış bir yerden bir belirteç ararlar ve daha sonra bilinen imzaların veritabanıyla karşılaştırırlar. Daha iyi doğruluk için birkaç belirteç genellikle kullanılır.

Test Objectives (Test Hedefleri)

- Web uygulamaları tarafından kullanılan bileşenleri parmak izi koyun.

How to Test (Nasıl Test Edilir)

(Siyah-Box Testi)

Çerçeveleri veya bileşenleri belirlemek için göz önünde bulundurulması gereken birkaç ortak yer vardır:

- HTTP başlıkları
- Çerezler
- HTML kaynak kodu

- Belirli dosyalar ve klasörler
- Dosya uzantıları
- Hata mesajları

HTTP Headers (HTTP Başlıkları)

Bir web çerçevesini tanımlamanın en temel şekline bakmaktır. `X-Powered-By` HTTP yanıt başlığında saha. Bir hedefin parmak izini parmak için birçok araç kullanılabilir, en basit olanı net kedidir.

Aşağıdaki HTTP İstemini Düşünün:

```
$ nc 127.0.0.1 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Server: nginx/1.0.14
[...]
X-Powered-By: Mono
```

The'den `X-Powered-By` field, web uygulama çerçevesinin muhtemel olduğunu anlıyoruz `Mono` .
. Ancak, bu yaklaşım basit ve hızlı olsa da, bu metodoloji vakaların% 100'ünde çalışmaz. Kolayca devre dışı bırakmak mümkündür
`X-Powered-By` Doğru bir konfigürasyon ile başlık. Ayrıca, bir web sitesinin HTTP başlıklarını gizlemesine izin veren birkaç teknik de vardır (İdealat bölümünde bir örnek bölümüne bakın). Yukarıdaki örnekte, belirli bir versiyonunu da not edebiliriz. `nginx` İçeriğine hizmet etmek için kullanılıyor.

Yani aynı örnekte test edici ya onu kaçırabilir. `X-Powered-By` Aşağıdaki gibi bir cevap başlığı veya alın:

```
HTTP/1.1 200 OK
Server: nginx/1.0.14
Date: Sat, 07 Sep 2013 08:19:15 GMT
Content-Type: text/html;charset=ISO-8859-1
Connection: close
```

```
Vary: Accept-Encoding
X-Powered-By: Blood, sweat and tears
```

Bazen belirli bir çerçevede bu noktada daha fazla HTTP başlığı vardır. Aşağıdaki örnekte, HTTP talebindeki bilgilere göre, bunun `X-Powered-By` Başlık PHP sürümünü içerir. Ancak, `X-Generator` Başlık, kullanılan çerçevenin aslında olduğuna dikkat çekiyor `Swiftlet` Bu da bir penetrasyon test cihazının saldırı vektörlerini genişletmesine yardımcı olur. Parmak izi alırken, bu tür sızıntılar için her HTTP başlığını dikkatlice inceleyin.

```
HTTP/1.1 200 OK
Server: nginx/1.4.1
Date: Sat, 07 Sep 2013 09:22:52 GMT
Content-Type: text/html
Connection: keep-alive
Vary: Accept-Encoding
X-Powered-By: PHP/5.4.16-1~dotdeb.1
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0, pre-chec
k=0
Pragma: no-cache
X-Generator: Swiftlet
```

Cookies (Çerezler)

Mevcut web çerçevesini belirlemenin bir başka benzer ve biraz daha güvenilir yolu çerçeveye özel çerezlerdir.

Aşağıdaki HTTP talebini göz önünde bulundurun:

```
GET /cake HTTP/1.1
Host: defcon-moscow.org
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.7; rv:22.0) Gecko/20100101 Firefox/22.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: ru-ru,ru;q=0.8,en-us;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
DNT: 1
Cookie: CAKEPHP=rm72kprivgmau5fmjdesbuqi71;
Connection: keep-alive
Cache-Control: max-age=0
```

Şekil 4.1.8-7: Kekpit HTTP İsteği

Kurabiye `CAKEPHP` Otomatik olarak ayarlanmıştır, bu da kullanılan çerçeve hakkında bilgi verir. Çerezler bölümünde ortak çerez adlarının listesi sunulur. Bu tanımlama mekanizmasına güvenmede sınırlamalar hala mevcuttur - çerezlerin adını değiştirmek mümkündür. Örneğin, seçilenler için `CakePHP` çerçeve aşağıdaki yapılandırma ile yapılabilir (ifadeden alıntı `core.php`) ::

```
/**
* The name of CakePHP's session cookie.
*
* Note the guidelines for Session names states: "The session name reference
s
* the session id in cookies and URLs. It should contain only alphanumeric
* characters."
* @link http://php.net/session_name
*/
Configure::write('Session.cookie', 'CAKEPHP');
```

Bununla birlikte, bu değişikliklerin yapılmasındaki değişikliklerden daha az olasıdır. `X-Powered-By` Başlık, bu nedenle tanımlamaya bu yaklaşım daha güvenilir olarak kabul edilebilir.

HTML Source Code (HTML Kaynak Kodu)

Bu teknik, HTML sayfası kaynak kodundaki belirli kalıpları bulmaya dayanır. Genellikle bir test cihazının belirli bir bileşeni tanımasına yardımcı olan çok fazla bilgi bulabilirsiniz. Ortak işaretlerden biri, doğrudan çerçeve ifşasına yol açan HTML yorumlarıdır. Daha sık belirli çerçeveye özel yollar bulunabilir, yani çerçeveye özel CSS veya JS klasörlerine bağlantılar bulunur. Son olarak, belirli komut dosyası değişkenleri de belirli bir çerçeveye işaret edebilir.

Aşağıdaki ekran görüntüsünden, belirtilen işaretçiler tarafından kullanılmış çerçeveyi ve sürümünü kolayca öğrenebilir. Yorum, belirli yollar ve komut dosyası değişkenleri, bir saldırganın ZK çerçevesinin bir örneğini hızlı bir şekilde belirlemesine yardımcı olabilir.

```
<script type="text/javascript" src="/zkau/web/bb9dff2f/js/zk.wpd" charset="UTF-8"></script>
<script type="text/javascript" src="/zkau/web/bb9dff2f/js/zul.lang.wpd" charset="UTF-8"></script>
<script type="text/javascript" src="/zkau/web/bb9dff2f/js/zuljsp.js" charset="UTF-8"></script>
<!-- ZK 6.5.1.1 EE 2012121311 -->
<script class="z-runonce" type="text/javascript">//<![CDATA[
zkopt({to:660});//]]>
</script><script type="text/javascript">
                zUtl.progressbox = function(id, msg, mask, icon, _opts) {
                    if (mask && zk.Page.contained.length) {
                        for (var c = zk.Page.contained.length, e = zk.Page.contained[--c]; e; e = zk.Page.contained[--c]) {
```

Şekil 4.1.8-2: ZK Framework HTML Kaynak Numune

Bu tür bilgiler sıklıkla yer almaktadır. `<head>` HTTP yanıtlarının bir bölümü, `<meta>` Etiketler,
ya da sayfanın sonunda. Bununla birlikte, tüm yanıtlar analiz edilmelidir, çünkü diğer yararlı yorumların ve gizli alanların incelenmesi gibi diğer amaçlar için yararlı olabilir. Bazen, web geliştiricileri kullanılan çerçeveler veya bileşenler hakkında bilgi saklamayı pek umursamaz. Sayfanın altında böyle bir şeye rastlamak hala mümkündür:

Şekil 4.1.8-3: Banshee Alt Sayfası

Specific Files and Folders (Belirli Dosyalar ve Klasörler)

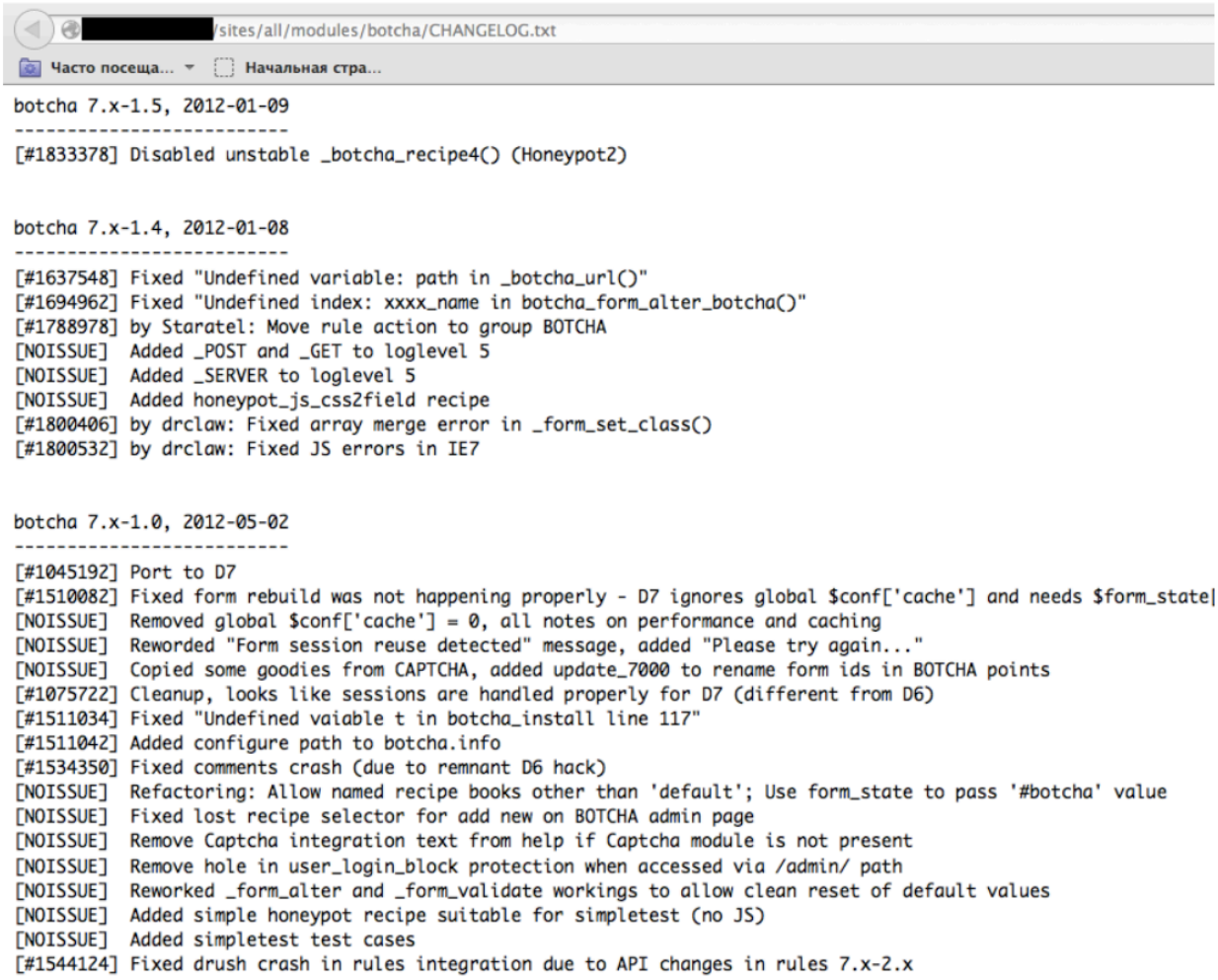
Bir saldırganın veya testçinin uygulamaları veya bileşenleri yüksek doğrulukla tanımlamasına büyük ölçüde yardımcı olan başka bir yaklaşım vardır. Her web bileşeninin sunucuda kendi özel dosyası ve klasör yapısı vardır. HTML sayfası kaynağından belirli yolu görebileceği belirtilmiştir, ancak bazen orada açıkça sunulmazlar ve hala sunucuda bulunurlar.

Onları ortaya çıkarmak için, zorla fırlatma veya “sürüklenme” olarak bilinen bir teknik kullanılır. Dirbusting, bilinen klasör ve dosya adları olan bir hedefi zorlayan ve sunucu içeriğini numaralandırmak için HTTP yanıtlarını izlemek için kabadır. Bu bilgiler hem varsayılan dosyaları bulmak hem de onlara saldırmak ve web uygulamasının parmak izi için kullanılabilir. Dirbusting çeşitli şekillerde yapılabilir, aşağıdaki örnek, Burp Suite'in tanımlanmış listesi ve davetsiz misafir işlevselliğinin yardımıyla WordPress destekli bir hedefe karşı başarılı bir yanlış saldırı olduğunu göstermektedir.

Request ▲	Payload	Status	Error	Timeout	Length
1	wp-includes/	403	☐	☐	383
2	wp-admin/	302	☐	☐	396
3	wp-content/	200	☐	☐	181

Şekil 4.1.8-4: Burp ile Dirbusting

Bunu bazı WordPress'e özgü klasörler için görebiliriz (örneğin, `/wp-includes/`, `/wp-admin/` ve `/wp-content/`) HTTP yanıtları 403 (Forbidden), 302 (Bulunma, yönlendirmeye yönlendirme `wp-login.php`) ve sırasıyla 200 (Tamam). Bu, hedefin WordPress'e güç sağladığının iyi
bir göstergesidir. Aynı şekilde, farklı uygulama eklenti klasörlerini ve sürümlerini yönlendirmek mümkündür. Aşağıdaki ekran görüntüsünde, kullanılan uygulama hakkında bilgi sağlayan ve savunmasız bir eklenti sürümünü açıklayan bir Drupal eklentisinin tipik bir CHANGLOG dosyasını
görebilirsiniz.



Şekil 4.1.8-5: Drupal Botcha Açıklama

İpucu: dirbusting ile başlamadan önce, kontrol edin `robots.txt` Önce dosya. Bazen uygulamaya özel klasörler ve diğer hassas bilgiler de orada bulunabilir. Böyle bir örnek `robots.txt` Dosya aşağıdaki ekran görüntüsünde sunulur.

Şekil 4.1.8-6: Robotlar Bilgi Açıklaması

Belirli dosyalar ve klasörler her belirli uygulama için farklıdır. Belirlenen uygulama veya bileşen Açık Kaynak ise, hangi altyapının veya işlevselliğin sunulduğunu ve hangi dosyaların sunucuda
bırakılabileceğini daha iyi anlamak için penetrasyon testleri sırasında geçici bir kurulum kurmada değer olabilir. Bununla birlikte, birkaç iyi dosya listesi zaten var; iyi bir örnek, öngörülebilir dosyaların / klasörlerin FuzzDB kelime listeleridir.

File Extensions (Dosya Uzantıları)

URL'ler, web platformunu veya teknolojisini tanımlamaya da yardımcı olabilecek dosya uzantılarını içerebilir.

Örneğin, OWASP wiki PHP kullandı:

`https://wiki.owasp.org/index.php?title=Fingerprint_Web_Application_Framework&action=edit&section=4`

İşte bazı yaygın web dosyası uzantıları ve ilgili teknolojiler:

- `.php` – PHP
- `.aspx` – Microsoft ASP.NET
- `.jsp` – Java Sunucu Sayfaları

Error Messages (Hata Mesajları)

Aşağıdaki ekran görüntüsünde görülebileceği gibi, WordPress'in kullanımına yönelik listelenen dosya sistemi yolu noktaları (`wp-content`). Ayrıca testçiler WordPress'in PHP tabanlı olduğunun farkında olmalıdır (`functions.php`).

Şekil 4.1.8-7: WordPress Parse Hatası

Common Identifiers (Yaygın Tanımlayıcılar)

Cookies (Çerezler)

Framework	Cookie name
Zope	zope3
CakePHP	cakephp
Kohana	kohanasession
Laravel	laravel_session
phpBB	phpbb3_
WordPress	wp-settings

1C-Bitrix	BITRIX_
AMPcms	AMP
Django CMS	django
DotNetNuke	DotNetNukeAnonymous
e107	e107_tz
EPiServer	EPiTrace, EPiServer
Graffiti CMS	graffitibot
Hotaru CMS	hotaru_mobile
ImpressCMS	ICMSession
Indico	MAKACSESSION
InstantCMS	InstantCMS[logdate]
Kentico CMS	CMSPreferredCulture
MODx	SN4[12symb]
TYPO3	fe_typo_user
Dynamicweb	Dynamicweb
LEPTON	lep[some_numeric_value]+sessionid
Wix	Domain=.wix.com
VIVVO	VivvoSessionId

HTML Source Code (HTML Kaynak Kodu)

Application	Keyword
WordPress	`<meta name="generator" content="WordPress 3.9.2" />`
phpBB	`<body id="phpbb"`
Mediawiki	`<meta name="generator" content="MediaWiki 1.21.9" />`
Joomla	`<meta name="generator" content="Joomla! - Open Source Content Management" />`
Drupal	`<meta name="Generator" content="Drupal 7 (http://drupal.org)" />`
DotNetNuke	`DNN Platform - [http://www.dnnsoftware.com](http://www.dnnsoftware.com)`

General Markers (Genel İşaretleyiciler)

Specific Markers (Spesifik İşaretleyiciler)

Framework	Keyword
Adobe ColdFusion	`<!-- START headerTags.cfm`
Microsoft ASP.NET	`__VIEWSTATE`
ZK	`<!-- ZK`
Business Catalyst	`<!-- BC_OBNW -->`
Indexhibit	`ndxz-studio`

Remediation (İyileştirme)

Farklı çerez adlarını (konfigleri değiştirme yoluyla), dosya / direktifi yollarını gizlemek veya değiştirmek (yeniden yazma veya kaynak kodu değişiklikleri yoluyla) kullanmak için çaba gösterilebilirken, bilinen başlıkları kaldırmak vb. Bu tür çabalar "belirsizlik yoluyla güvenlik" olarak kaynar. Sistem sahipleri / yönetimler, bu çabaların yalnızca en temel hasımları yavaşlattığını kabul etmelidir. Zaman / çaba, paydaş farkındalığı ve çözüm bakım faaliyetlerinde daha iyi kullanılabilir.

Tools (Araçlar)

Genel ve tanınmış araçların bir listesi aşağıda sunulmuştur. Ayrıca birçok başka kamu hizmetinin yanı sıra çerçeve tabanlı parmak izi araçları da vardır.

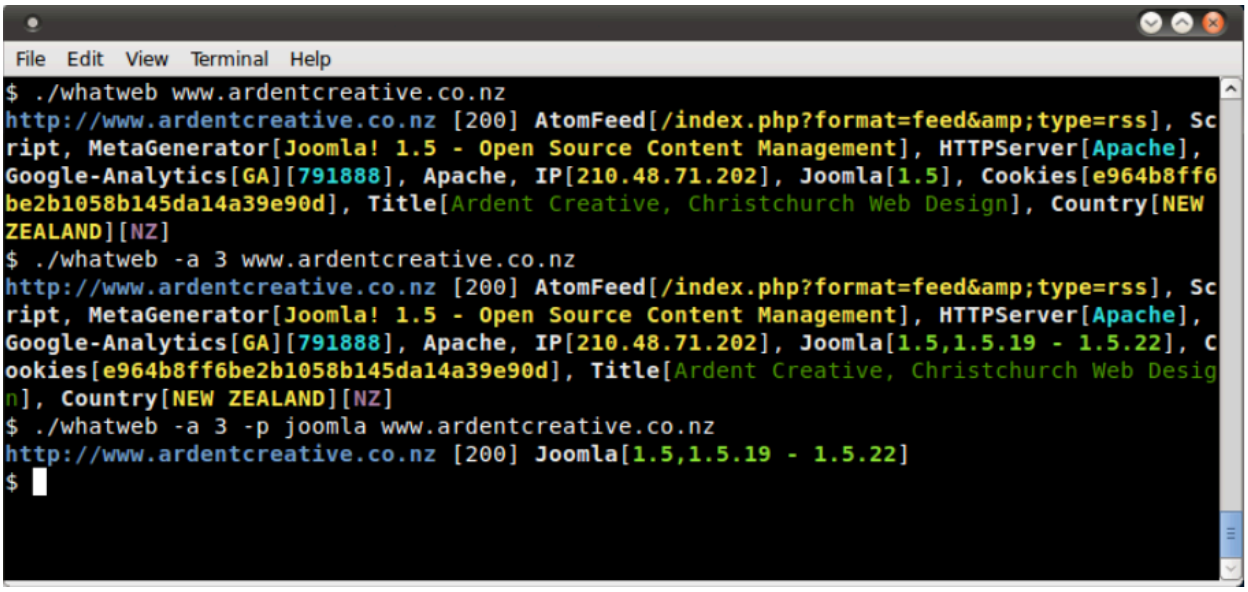
WhatWeb (Ne Web)

Web Sitesi: https://github.com/urbanadventurer/WhatWeb

Şu anda piyasadaki en iyi parmak izi araçlarından biri. Varsayılan Kali Linux yapısına dahil edilmiştir. Dil: Parmak izi için Ruby Matches:

- Metin telleri (vaka hassas)
- Düzenli ifadeler
- Google Hack Veritabanı sorguları (sınırlı anahtar kelimeler kümesi)
- MD5 hashs
- URL tanıma
- HTML etiket kalıpları
- Pasif ve agresif operasyonlar için özel yakut kodu

Örnek çıktı aşağıdaki ekran görüntüsünde sunulur:



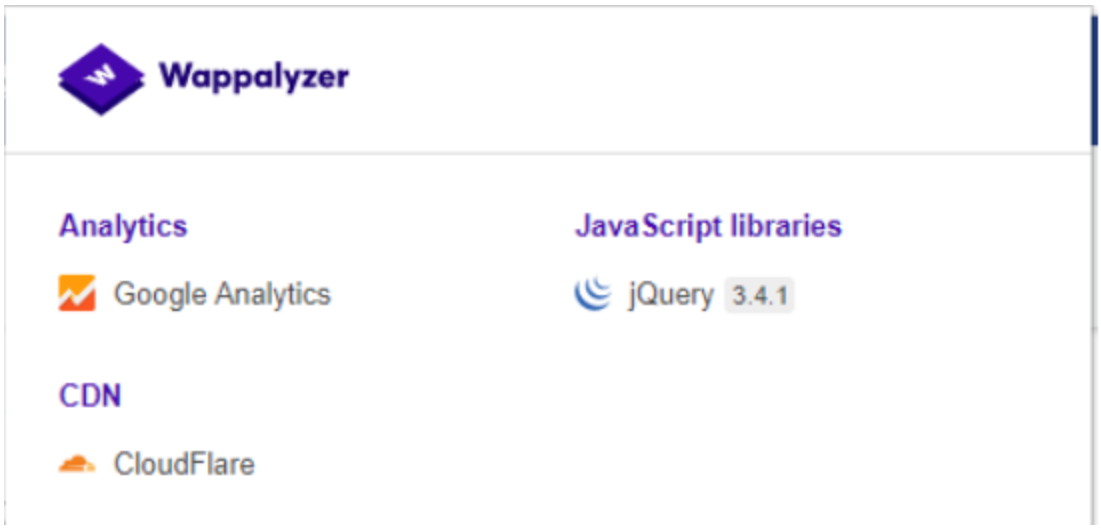
Şekil 4.1.8-8: Whatweb Output örnek

Wappalyzer (Wappalizer)

Web Sitesi: https://www.wappalizer.com/

Wapplyzer, en popüler olanı muhtemelen Firefox / Chrome uzantıları olan birden fazla kullanım modelinde mevcuttur. Sadece düzenli ifade eşleştirme üzerinde çalışırlar ve tarayıcıya yüklenmek üzere sayfadan başka bir şeye ihtiyaç duymaz. Tamamen tarayıcı seviyesinde çalışır ve simgeler şeklinde sonuçlar verir. Bazen yanlış pozitiflere sahip olsa da, bu, bir sayfaya göz attıktan hemen sonra bir hedef web sitesi oluşturmak için hangi teknolojilerin kullanıldığına dair fikir sahibi olmak için çok kullanışlıdır.

Bir eklentinin örnek çıktısı aşağıdaki ekran görüntüsünde sunulur.



Şekil 4.1.8-9: OWASP Web Sitesi için Wappalizer Çıkışı

References (Referanslar)

Whitepapers (Beyaz kağıtlar)

- Saumil Shah: " HTTP parmak izine giriş"
- Anant Shrivastava : "Web Uygulama Parmak Baskısı"