Anti Scanner

written by Mert SARICA | 3 February 2014

2011 yılının Şubat ayında, web sitemin, Acunetix, Netsparker ve Appscan web uygulaması güvenliği zafiyet tarama araçları ile sıkça taranmasından dolayı bu araçlar üzerinde ufak bir araştırma yapıp Script Kiddie Bezdirme Mekanizması adında bir yazı yazmıştım. Geçtiğimiz aylarda sitemin kayıtlarını incelerken yine çok sayıda Netsparker ile tarama kaydına rastladım. Ufak bir araştırma ve karşılaştırma sonucunda, geçtiğimiz 3 sene içinde sitemi taramak için kullanılan araçların başında yine Netsparker'ın (community edition) olduğunu, ikinci olarak ise Acunetix'in (ticari sürüm) olduğunu gördüm. Netsparker'ın hem ücretsiz olması hem de ticari sürümüne göre kısıtları olmasına rağmen, rakiplerine kıyasla daha tutarlı sonuçlar üretebilen etkili bir araç olması, güvenlik uzmanlarının yanı sıra niyeti bozuk arkadaşlar tarafından da tercih edilmesine neden olmaktadır. 3 sene önceye göre sitesi daha da sık taranan ve nezaketen de olsa tarayanlar tarafından ne idari ne de teknik zafiyet analiz raporu paylaşılmayan biri olarak (:)) tarayanların işini 3 sene önceye göre biraz daha zorlaştırmaya, yöntemi ve ilgili kodları sizlerle paylaşmaya karar verdim.

Sitem daha çok Netsparker ile tarandığı için ilk olarak Netsparker odaklı basit bir çözüm üretmeye karar versem de, özelleştirilebilen daha esnek bir çözümün daha fazla zafiyet tarayıcısını ve zafiyet arayan botları engellemede kullanılabileceğini düşünerek farklı çözümler üzerinde düşünmeye başladım.

İşe ilk olarak WordPress'in trafik kayıtlarını incelemekle başladım. Çoğu zafiyet tarayıcısı tarama esnasında, USER-AGENT alanları da dahil olmak üzere sunucuya gönderilen verilere imzalarını (Acunetix, Netsparker vs.) atarlar. Özellikle Netsparker gibi ücretsiz olarak dağıtılan araçlarda bu imzaların arayüz üzerinden değiştirilmesi çoğu zaman mümkün olmamaktadır dolayısıyla bu imzaya yönelik üretilebilecek basit bir çözüm, tara ve geçten öteye gidemeyen niyeti bozuk kişileri ve/veya botları bezdirmek için yeterli olacaktır. Örneğin aşağıdaki iki ekran görüntüsüne bakacak olursanız burada Netsparker'ın USER-AGENT alanında imzasına yer verdiğini görebilirsiniz.

17 December, 2013	12:51:08	85.102.160.100	English		http://\'--/style/scriptscriptnetsparker(0x0002B4)/script	[Page]: Home
17 December, 2013	12:51:05	85.102.160.100	English		http://\'+NSFTW+\'	[Page]: Home
17 December, 2013	12:50:46	85.102.160.100	English		http://\'+NSFTW+\'	[Page]: Home
17 December, 2013	12:50:45	85.102.160.100	English		http://ns:netsparker056650=vuln	[Page]: Home
17 December, 2013	12:50:40	85.102.160.100	English		http://ns:netsparker056650=vuln	[Page]: Home
17 December, 2013	12:50:38	85.102.160.100	English		http://netsparker.com/n	[Page]: Home
17 December, 2013	12:50:34	85.102.160.100	English		http://\'+NSFTW+\'	[Page]: Home
17 December, 2013	12:50:32	85.102.160.100	English		http://\'+NSFTW+\'	[Page]: Home
17 December, 2013	12:50:30	85.102.160.100	English		http://\'--/style/scriptscriptnetsparker(0x00028F)/script	[Page]: Home
17 December, 2013	12:50:30	85.102.160.100	English		http://netsparker.com/n	[Page]: Home
17 December, 2013	12:50:25	85.102.160.100	English		http://\'--/style/scriptscriptnetsparker(0x000284)/script	[Page]: Home
17 December, 2013	12:50:15	85.102.160.100	English		http://ns:netsparker056650=vuln	[Page]: Home
17 December, 2013	12:50:07	85.102.160.100	English		http://ns:netsparker056650=vuln	[Page]: Home

Report for 85.102.160.100

Ban IP address

Records in database:	952
Latest hit:	17 December, 2013 12:52:27
First hit:	17 December, 2013 12:44:41
User agent(s):	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; Netsparker) Mozilla/5.0 (compatible; MSIE 9.0; Windows NT 6.1; Trident/7.0) Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/31.0.1650.63 Safari/537.36

URLs Requested

Date	Time	OS	Browser	Agent	Referrer	URL Requested
17 December, 2013	12:52:27	Windows XP	Internet Explorer 6	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; Netsparker)		/wp-content/doxygen.conf
17 December, 2013	12:52:18	Windows XP	Internet Explorer 6	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; Netsparker)		/wp-content/httpd-vhosts.conf
17 December, 2013	12:52:17	Windows XP	Internet Explorer 6	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; Netsparker)		/wp-content/local.conf
17 December, 2013	12:52:16	Windows XP	Internet Explorer 6	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; Netsparker)		/wp-content/httpd.conf
17 December, 2013	12:52:16	Windows XP	Internet Explorer 6	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; Netsparker)		Euroforensics 2013 Adli Bilimler, Siber Güvenlik ve Gözetim Teknolojileri Konferansı ve Sergisi
17 December, 2013	12:52:14	Windows XP	Internet Explorer 6	Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1; SV1; .NET CLR 1.1.4322; Netsparker)		/wp-content/test.conf

Bezdirme yöntemi olarak tarayıcı tarafından web sunucusuna gönderilen her istek (request) için, rastgele değerlerden oluşan bir form ve az sayıda sahte zafiyet (dizin bilgisi ifşası, veritabanı bilgisi ifşası) oluşturan kısa ve öz bir PHP uygulaması hazırlamaya karar verdim. Buradaki amacım, rastgele değerlerden oluşan bir form oluşturan bu PHP uygulaması sayesinde tarayıcı, her gönderdiği yeni istekte, yeni bir form ve bunun bağlı olduğu yeni bir sayfa ile karşılaştığını zannederek her sayfayı, bu sayfada bulunan formu ve ilgili alanları, test edilecek sayfalar kuyruğuna alarak kısır döngüye girmesini ve/veya sistem üzerinde performans sorununa yol açmasını sağlamaktı.

antiscanner.php

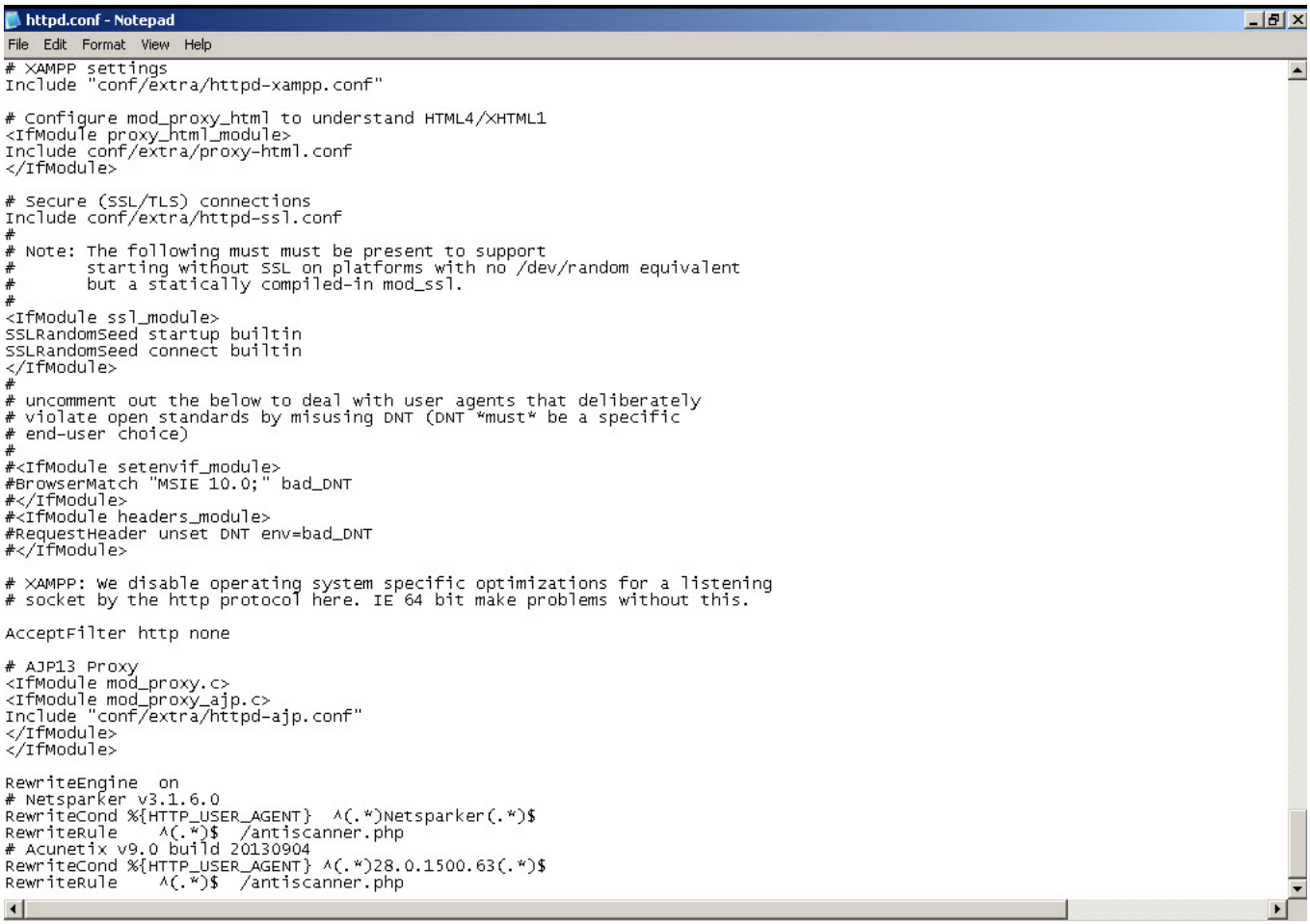
```php
<?php
function randString($length, $charset='ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789')
{
    $str = '';
    $count = strlen($charset);
    while ($length--) {
        $str .= $charset[mt_rand(0, $count-1)];
    }
    return $str;
}
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US">
<head profile="http://gmpg.org/xfn/11">
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>Hack 4 Career - http://www.mertsarica.com</title>
<link rel="stylesheet" href="http://www.<?php echo randString(50);?>.com/<?php echo randString(50);?>.css" type="text/css" media="screen" />
<link rel="stylesheet" href="http://www.<?php echo randString(50);?>.com/<?php echo randString(50);?>.css" type="text/css" media="screen" />
<link rel="alternate" type="application/rss+xml" title="Hack 4 Career RSS Feed" href="http://www.<?php echo randString(50);?>.com/feed/" />
<link rel="alternate" type="application/atom+xml" title="Hack 4 Career Atom Feed" href="http://www.<?php echo randString(50);?>.com/feed/atom/" />
<link rel="pingback" href="http://www.<?php echo randString(50);?>.com/xmlrpc.php" />

<form action="<?php echo randString(50);?>.php" method="post">
 <p><?php echo randString(50);?>: <input type="text" name="<?php echo randString(50); ?>" /></p>
 <p><?php echo randString(50);?>: <input type="text" name="<?php echo randString(50); ?>" /></p>
 <p><input type="submit" /></p>
</form>

function antiscanner($antiscanner)<br \>
{<br \>
    return $antiscanner;<br \>
}<br \>

"/usr/local/<?php echo randString(50);?>"<br \>

"c:/<?php echo randString(50);?>"<br \>

define( 'DB_NAME',     'database' );<br \>
define( 'DB_USER',     'www.mertsarica.com' );<br \>
define( 'DB_PASSWORD', 'antiscanner' );<br \>
define( 'DB_HOST',     'localhost' );<br \>
define( 'DB_CHARSET',  'utf8' );<br \>

<?php echo randString(50);?>@<?php echo randString(50);?>.com<br \>
</html>
```

Tabii tarayıcıyı kısır döngüye sokabilmek için web sunucusu üzerinde PHP uygulaması tarafından oluşturulan her sahte form sayfasının çağrıldığında, web sunucusunun tarayıcıya geçerli (200 OK) sağlamam gerekiyordu. Bunun için sunucu üzerinde olası binlerce sayfa oluşturamayacağım için Apache'nin mod_rewrite modülünden faydalanmaya karar verdim.

mod_rewrite gelen URL isteklerini düzenli ifade kurallarına dayanarak devingen olarak dönüştürmek için bir yöntem sağlar. Böylece keyfi URL'leri kendi URL yapınızla istediğiniz şekilde eşleştirmeniz mümkün olur. Gerçekten esnek ve güçlü bir URL kurgulama mekanizması oluşturmak için sınısız sayıda kural ve her kural için de sınırsız sayıda koşul destekler. URL değişiklikleri çeşitli sınamalara bağlı olabilir: sunucu değişkenleri, HTTP başlıkları, ortam değişkenleri, zaman damgaları, çeşitli biçimlerde harici veritabanı sorguları.

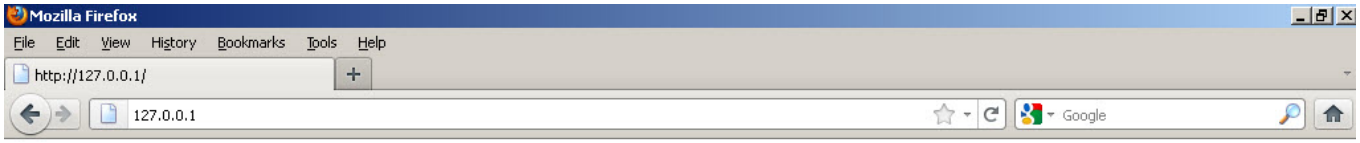
Tabii yanıtlanması gereken ufak bir soru daha vardı o da mod_rewrite ile tarayıcıyı kısır döngüye sokarken gerçek kullanıcının bundan nasıl etkilenmemesini sağlayabilirdim ? Bunun için yazının girişinde bahsettiğim ve tarayıcıların imzalarını kullandıkları USER-AGENT alanına yönelik bir mod_rewrite kuralı yazmaya karar verdim. Tabii Acunetix'in ticari sürümündeki (v9.0 build 20130904) varsayılan USER-AGENT imzası, Netsparker'ın (v3.1.6.0)

aksine kendi adı yerine Chrome'un USER-AGENT değerini kullanıyordu. Chrome internet tarayıcısı otomatik güncellemeye sahip olduğu ve Acunetix'in USER-AGENT alanında varsayılan olarak kullandığı bu değer, eski bir sürüme ait olduğu için dert etmeden, gönül rahatlığıyla Acunetix için de bir kural yazabileceğime karar verdim.



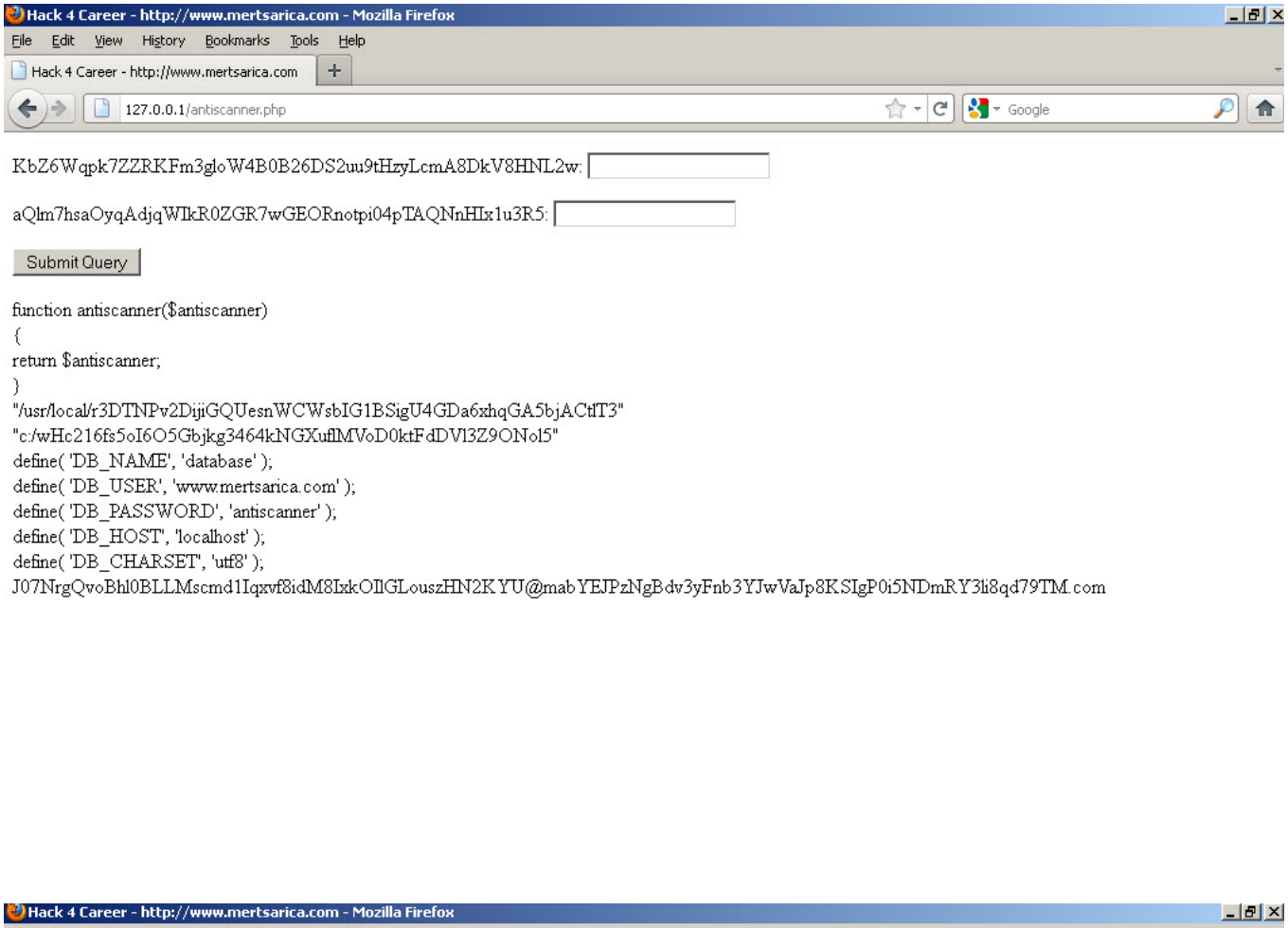
Yukardaki mod_rewrite kuralı ile USER-AGENT alanı, Netsparker veya Acunetix'in kullandığı değere eşit ise, istekleri otomatik olarak hazırladığım PHP uygulamasına (antiscanner.php) yönlendirdim.

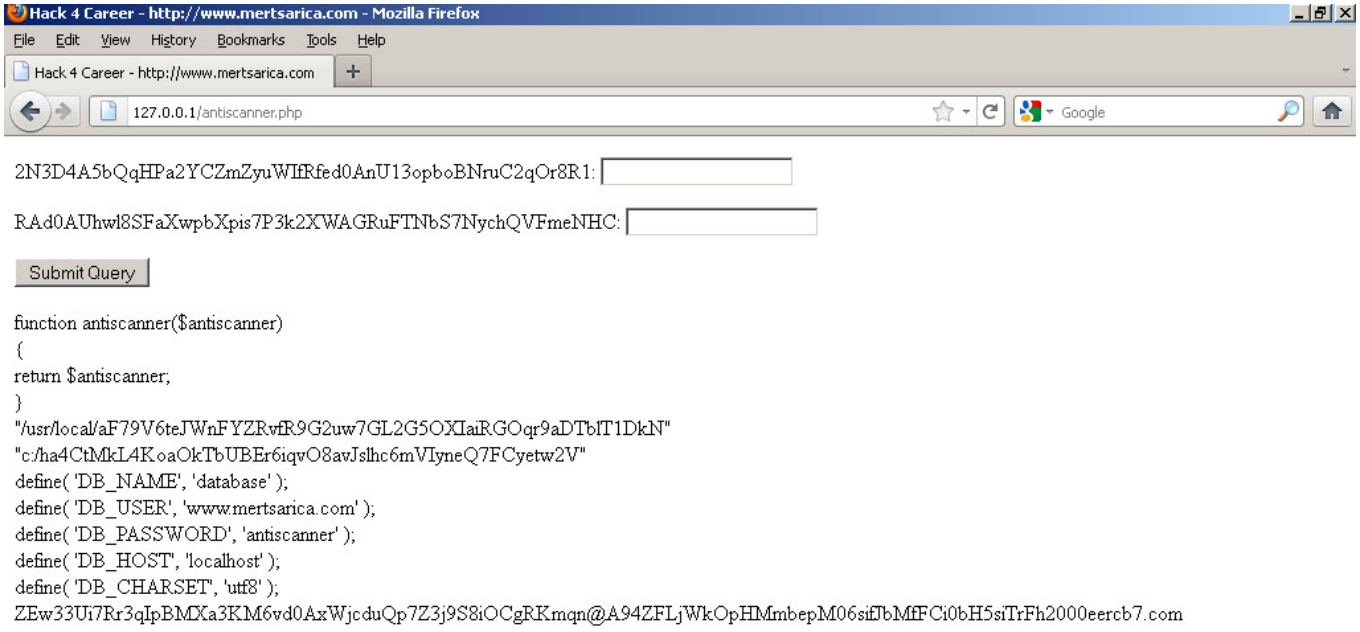
Öncelikle normal kullanıcıların bu PHP uygulamasından etkilenmediğini teyit etmek için sayfayı internet tarayıcısının varsayılan USER-AGENT'ı ile çağırdığımda sayfanın normal halini görüntüleyebildim.



Hack4Career
http://www.mertsarica.com

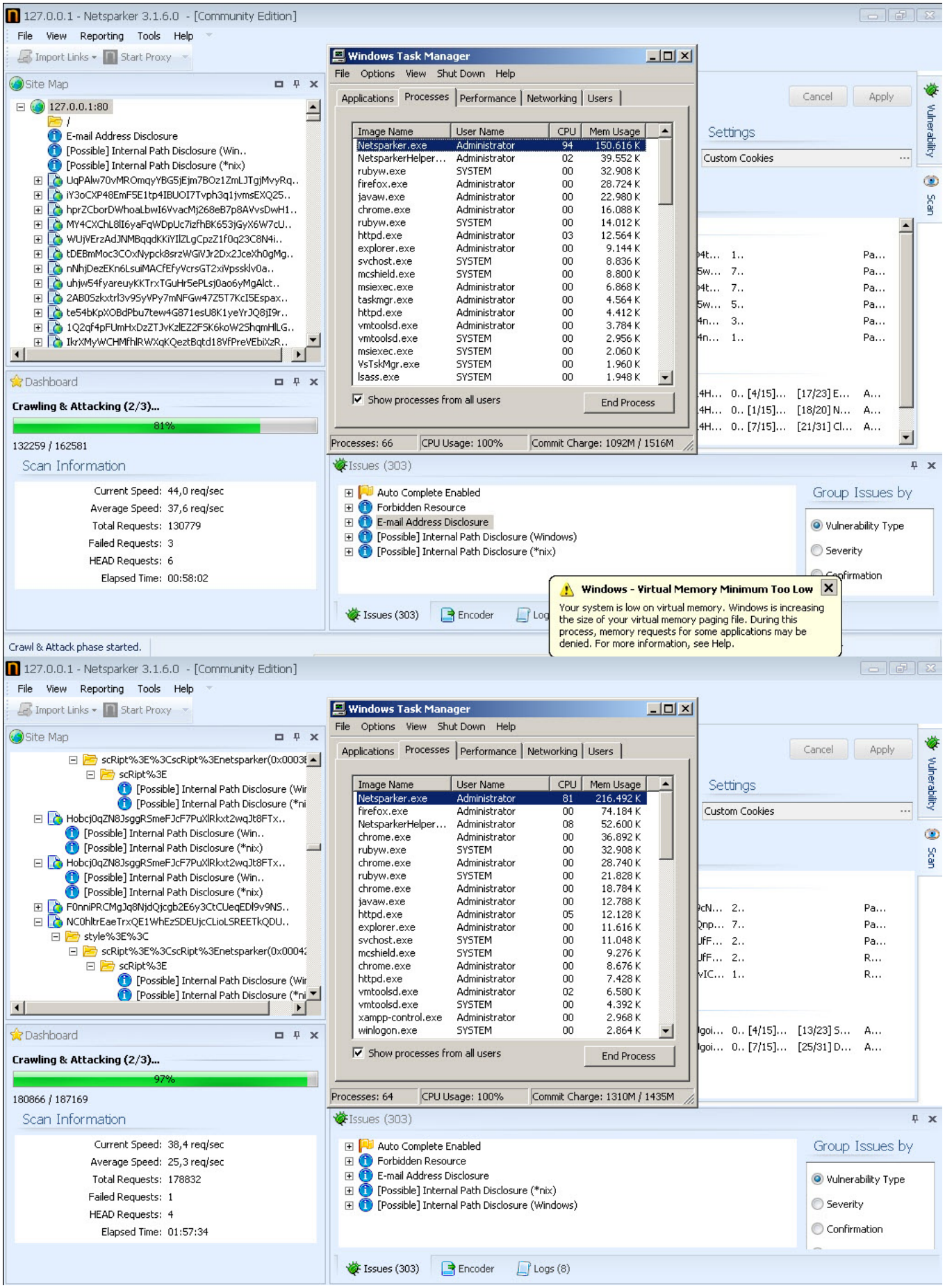
Ardından Firefox'un User Agent Switcher eklentisi ile USER-AGENT'ımı Netsparker olarak değiştirdikten sonra sayfanın her istekte farkı yanıt (form ve formun bulunduğu adres) döndüğünü doğruladım.



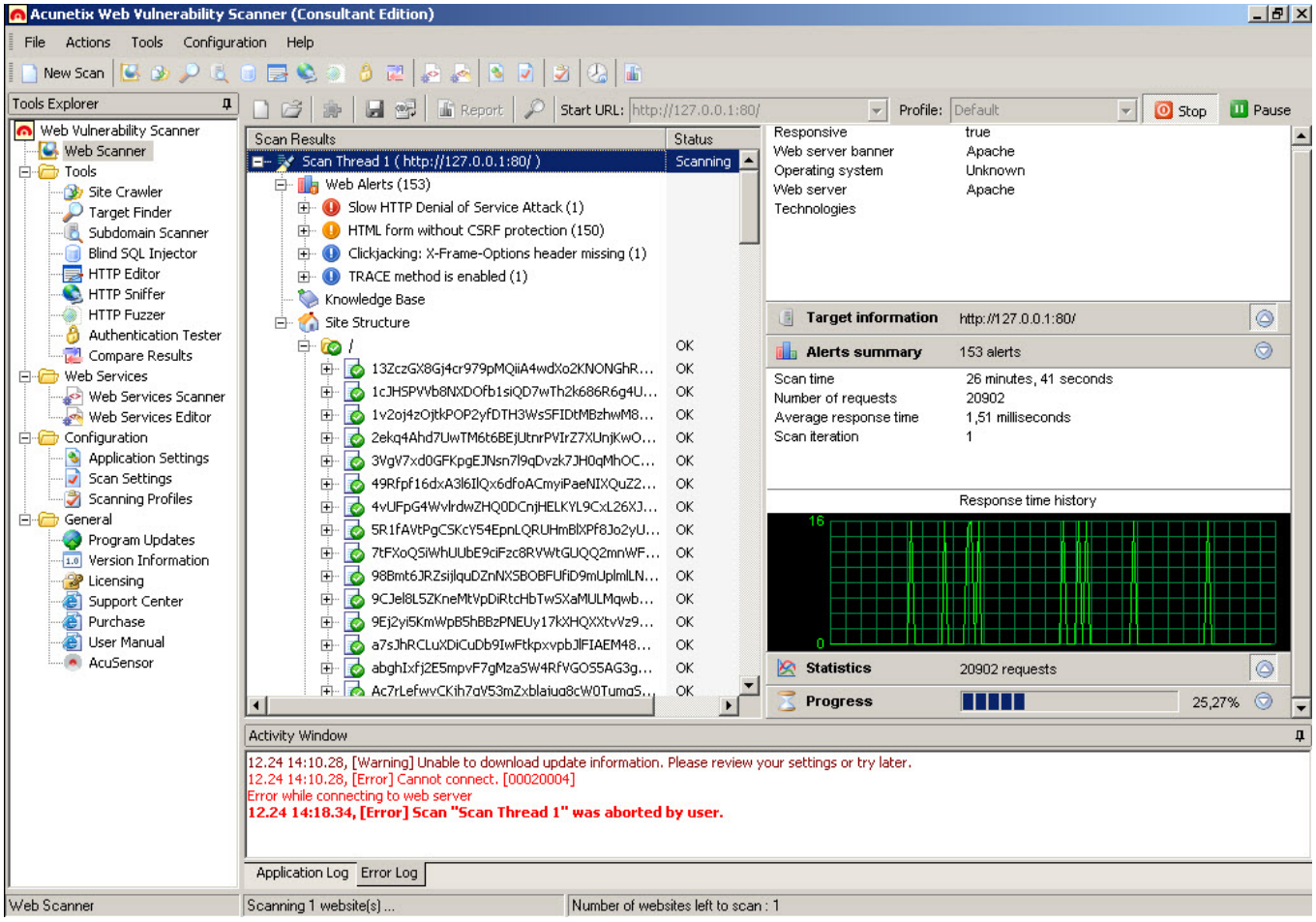


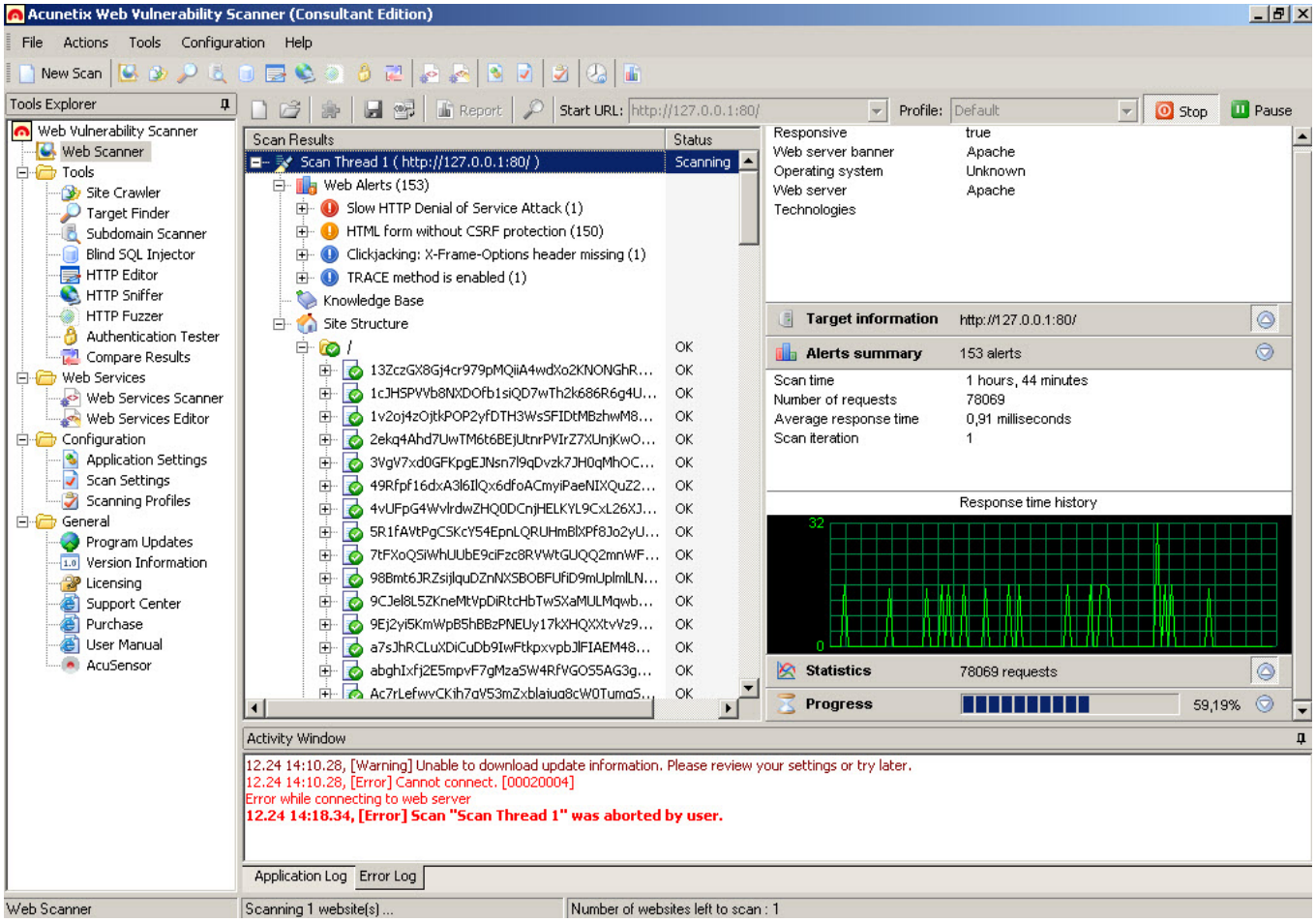
Sıra Netsparker ve Acunetix ile test yapmaya geldiğinde, Netsparker Community Edition sürümünün, başladığı taramayı 4 saat sonunda hala bitiremediğini ve

artan bellek kullanımı nedeniyle işletim sistemi üzerinde bellek sorununa yol açtığını gördüm.



Acunetix ile yapmış olduğum taramada ise bellek sorunu ile karşılaşmamış olsam da taramanın 2 saat sonunda hala bitemediğini gördüm.





Kısadan hisse, mod_rewrite ve ufak bir PHP uygulaması ile script kiddieler'in taramalarını yavaşlatacak bir yöntem geliştirmiş oldum. Evet baktığınız zaman bu yöntemin atlatılması çok zor değil ancak ilave kontroller uygulayarak kedi fare oyunundaki yerinizi alabilirsiniz :)

Örnek PHP uygulamasına ve mod_rewrite kuralı içeren httpd.conf dosyasını buradan indirebilirsiniz.

Bir sonraki yazıda görüşmek dileğiyle herkese güvenli günler dilerim.