mod_antiCrawl: İnternet Robotlarından Korunma Modülü

M.Oğuzhan Topgül[1], M.Ufuk Çağlayan[2]

[1] TÜBİTAK-BİLGEM, UEKAE, Kocaeli

[2] Boğaziçi Üniversitesi, Bilgisayar Mühendisliği Bölümü, İstanbul

oguzhan.topgul@uekae.tubitak.gov.tr[1], caglayan@boun.edu.tr[2]

Özet:İngilizcede crawler diye adlandırılan internet robotu yazılımları bir web sayfasındaki tüm bağlantıları gezerek bu sitenin haritasını çıkartan otomatize yazılımlardır. Bir web sitesinin haritasının çıkartılması, o siteye yapılacak bir saldırı için temel teşkil edeceğinden otomatize saldırılar için büyük önem taşımaktadır. Bu yüzdendir ki otomatize web açıklık tarayıcılarının hepsi, taramaya başlamadan önce mutlaka sitenin haritasını çıkartmak için bağlantı keşfi işlemi (crawling) gerçekleştirir. İnternet robotları, otomatize web açıklık taramalarına temel olmak dışında, içerik hırsızlığı için de sıkça kullanılmaktadır. Otomatize bir şekilde bir sitenin tüm içeriğinin sayfa sayfa gezilerek başka bir web sitesine kopyalanması konusunda internet robotları büyük rol oynamaktadır. Bu çalışmada internet robotlarına karşı bağlantı keşfi önleyici yöntemlerden bahsedilmiş ve bu bağlamda web sunucular için internet robotu koruma modülü mod_antiCrawl geliştirilmiştir.

Anahtar Sözcükler: İnternet robotu, Apache, Modül. Web sitesi, İçerik hırsızlığı, Otomatize taramalar

mod_antiCrawl: A Module of Crawler Protection

Abstract: Web crawler can be defined as an automated software that extracts website maps by visit all the links in that website. Because of this website map extraction process can be used as a basis for a web attack, crawling plays an important role as a part of an automated attacks. That's why all the automated vulnerability scanners do crawling before vulnerability tests in order to see the overall map. Beside automated scanning issues, crawlers are also used for content theft frequently. By using a crawler, one can copy all the web pages and content into another website through visiting all pages respectively. In this work, it is aimed to explain anti-crawling methods that are ways of protecting websites from crawlers, and in this sense an anti-crawling web server module is developed.

Keywords: Crawler, Apache, Module, Website, Content theft, Automated scanning

1 Giriş

Bağlantı keşfi adı verilen işlem bir web sayfasında bulunan tüm bağlantıların otomatize bir şekilde ziyaret edilmesi olarak tanımlanmaktadır. Bu işlemi gerçekleştirmek için kullanılan otomatize yazılımlar da internet robotu (crawler) olarak adlandırılır. İnternet robotları teker teker sayfalardaki tüm bağlantıları ziyaret ederek web sitesinin haritasını çıkartmakta ve bu site haritası web odaklı otomazite saldırılara temel oluşturmak için kullanılmaktadır. Bir otomatize web saldırısının ilk adımını her zaman bağlantı keşfi işlemi oluşturmaktadır [1].

Bağlantı keşfi ve web indeksleme işlemleri genellikle arama motorları tarafından arama performansını ve kullanıcıya gösterilen sonuçların doğruluğunu arttırmak amacıyla kullanılmaktadır. Ancak internet robotlarının saldırılara temel oluşturmak ve otomatize yazılımlar ile saldırıların daha hızlı ve kolay bir şekilde gerçekleştirilmesini sağlamak gibi amaçlarla kullanılması, güvenlik açısından üzerinde durulması gereken hususlardır.

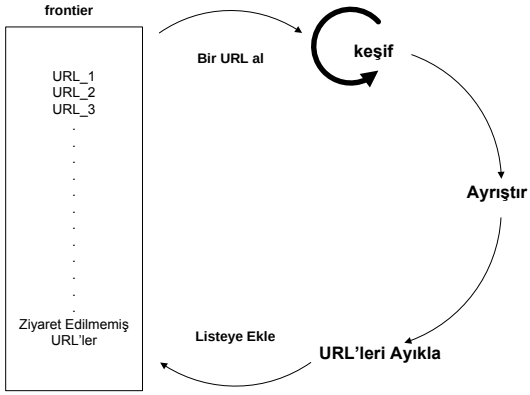
İnternet robotlarının kötüye kullanım amaçlarından biri de içerik hırsızlığıdır. Bir web sitesi, içerisindeki tüm bağlantılar ziyaret edilerek başka bir siteye, içerik sahibinin izni olmaksızın internet robotları kullanılarak kopyalanabilmektedir. Hatta inline-linking adı verilen bir yöntem ile içeriğin kopyalandığı sahte site üzerindeki resimler gerçek siteden çekilmekte ve resimlerin gösterilmesi sahte sitenin bant genişliği kullanılmadan sağlanmaktadır.

Yukarıda bahsi geçen otomatize açıklık taramaları ve otomatize saldırıları engellemek için en verimli yöntemlerden biri internet robotlarının engellenmesidir.

2 İnternet Robotu - Crawler

İnternet robotları tohum (seed) adı verilen bir sayfadan işlemlerine başlayarak bu tohum sayfası üzerindeki tüm bağlantılardan (URL) yeni sayfalar bulmaktadır. Bu işlem yeni keşfedilen sayfalar üzerinde de tekrarlanmalı bir şekilde gerçekleştirilerek web sitesi içerisindeki tüm sayfalar çıkartılmaktadır.

İnternet robotları ilk adım olarak keşif listesi (frontier) adı verilen ve içerisinde daha önce ziyaret edilmemiş bağlantı adreslerinin bulunduğu bir liste oluştururlar. Bu listeden daha önce ziyaret etmediği bir adresi alarak işleme başlarlar. Tarama işleminin başında keşif listesinde sadece hedef web sayfasının adresi bulunur. Listedeki ilk adresin alınıp sayfa içerisindeki diğer bağlantı adresleri çıkartıldıkça keşif listesi yeni bulunan bağlantılar ile güncellenir. Bu işlem döngüsel bir şekilde tekrarlanarak tüm web sitesinin haritası çıkartılana kadar ya da hedeflenen başka bir veri var ise, o veriye ulaşılana kadar devam ettirilir. Bu döngü Şekil 1'de gösterilmiştir.



Şekil 1 İnternet Robotu Çalışma Diagramı

İnternet robotları çalışmaları sırasında daha önce ziyaret ettikleri ve ayrıştırdıkları sayfaları tekrar tekrar işleme almamak için tarama geçmişini tutarlar. Böylece internet robotunun daha verimli çalışması sağlanmış olunur.

İnternet robotları, sayfaları ayrıştırıp içindeki bağlantı adreslerini (URL) ayıklarken öncelikli olarak sayfa içerisinde **<a>** HTML etiketini ararlar. Çünkü **<a>** etiketi içindeki **href** parametresinin değeri sayfadaki bağlantının adresinin yazılı olduğu kısımdır. href parametrelerinin değerleri ayıklanıp keşif listesine eklenmektedir.

3 Bağlantı Keşfi Önleyici Metodlar

Bağlantı keşfi önleme (Anti-Crawling) terimi, web sitelerini internet robotlarından koruma metodlarını ifade etmektedir. Bu metdolardan bazıları aşağıda açıklanmıştır.

3.1. Pozitif İstemci Ajanı Listesi

İstemci Ajanı (User-Agent), kullanıcıların kullandıkları istemci uygulamasının ve bu uygulamaya ait sürüm bilgisinin bulunduğu bir HTTP başlığını ifade etmektedir. HTTP taleplerinin içeriğinde istemci ajanı bilgisi bulunmaktadır. Şekil 2'de örnek bir HTTP talebi ve içerisindeki istemci ajanı başlığı gösterilmiştir.

```
GET / HTTP/1.1
Host: boun.edu.tr
User-Agent: Mozilla/5.0 (Windows; U; Windows
Accept: text/html,application/xhtml+xml,appl
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0
Keep-Alive: 115
Proxy-Connection: keep-alive
```

Şekil 2 HTTP Talebi İçerisindeki İstemci Ajanı Başlığı

İstemci ajanı başlığının kontrol edilmesi ve sadece geçerli bir istemci ajanı değerine sahip olan taleplere izin verilmesiyle bazı internet robotları engellenebilmektedir. Ancak bu metodun işlevsel olabilmesi için öncelikli olarak geçerli istemci ajanı değerlerinin belirlenmesi ve pozitif istemci ajanı listesinin oluşturulması gerekmektedir. Pozitif istemci ajanı listesi oluşturulduktan sonra sunucu sadece User-Agent başlığının değerini kontrol ederek bu talebe izin verilip verilmeyeceğini anlayabilmektedir.

3.2. Oturum Bilgisi Kontrolü

Bazı internet robotları oturum bilgisi tutma yeteneğine sahip değildir. Bu sebeple yapılan ardışık taleplerde gönderilmeyen bir oturum bilgisi bu istemcinin gerçek bir kullanıcı değil otomatize bir robot olduğu fikrinin verebilmektedir.

3.3. Sayfaların JavaScript ile Oluşturulması

Birçok temel ve gelişmiş internet robotunun javascript desteği bulunmamaktadır. Bu sebeple javascipt ile oluşturulan bir içeriğin robotlarla ayrıştırılması ve ayıklanması pek mümkün değildir. Sayfaların javascript ile oluşturulması internet robotlarına karşı bu teknolojinin yapısı gereği bir koruma sağlamaktadır.

3.4. Talep Sayısı Kontrolü

İnternet robotlarına karşı kullanılabilecek en verimli korunma yöntemlerinden birisi de belirli bir zaman dilimi içerisinde belirli bir kaynak IP'den belirli bir üst limitten daha yüksek sayıda talebin gelmesi durumunun kontrolüdür. Böylece internet robotları gibi çok hızlı ve devamlı olarak talep gönderen otomatize araçlar tespit edilebilmektedir.

3.5. Gizli Tuzak Bağlantı Kontrolü

İnternet robotlarına karşı kullanılabilecek en etkili ve en akıllı yöntemlerden birisi gizli tuzak bağlantı kontrolüdür. Bu yöntem, sayfanın içerisine yerleştirilen ve normal kullanıcılar tarafından görülemeyecek; ancak internet robotları gibi sayfanın kaynak kodu üzerinden işlem yapan araçlar tarafından görülebilecek gizli tuzak bağlantıların istenmesi durumunda bu istemcinin bir internet robotu olduğunun anlaşılmasına dayanmaktadır.

Örnek vermek gerekirse;
<a style=”display:none” href=/dontfollow.php />

gibi bir bağlantı normal kullanıcılar tarafından görülemeyeceğinden böyle bir sayfanın talep edilmesi beklenmemektedir. Ancak internet robotları sayfanın kaynak kodu üzerinden işlem yaptıklarından bu bağlantının var olduğunu görecek ve bu bağlantıyı ziyaret etmek isteyeceklerdir. Böyle bir durumda internet robotları tespit edilmiş olunacaktır.

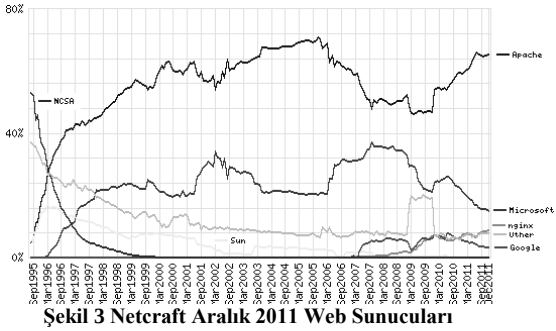
4 Önerilen Modeller ve Uygulanması

İnternet robotları arama motorlarının sonuç dönme performanslarının arttırılması gibi faaliyetleri sebebiyle faydalı araçlardır. Ancak bu araçların saldırı temeli oluşturma ve içerik hırsızlığı gibi amaçlara hizmet etmesi gibi kullanımları bu araçlara karşı da bazı önlemlerin alınması gerekliliğini gündeme getirmiştir. Bu yüzden yapılan bu çalışmada yukarıda bahsedilen bazı korunma metotları bir Apache sunucu modülü olarak gerçeklenmiştir. Bu modül mod_antiCrawl olarak isimlendirilmiştir.

mod_antiCrawl tasarlanırken etkili olduğu düşünülen Talep Sayısı Kontrolü ve Gizli Tuzak Bağlantı Kontrolü adımlarının gerçeklenmesi planlanmıştır. Günümüz internet robotları, istemci ajanı değerinin istenilen herhangi bir değerle değiştirilmesine olanak sağlayan yapılandırma ayarlarına sahiptirler. Ayrıca birçok temel ve gelişmiş seviyedeki internet robotu oturum tutma yeteneğini geliştirmişlerdir. Her iki yöntemin de teoride kullanışlı gibi görünmelerine rağmen internet robotlarının gelişimi ile birlikte uygulamada yeterli bir koruma sağlamayacağı düşünülmüş ve bu nedenle mod_antiCrawl çalışması bünyesine dahil edilmemişlerdir.

4.1. Apache Sunucusu

Apache, Apache Software Foundation tarafından desteklenen ve büyük ölçüde açık kaynak camiası tarafından geliştirilen ve sürdürülen bir web sunucusudur. Apache License lisansı altında geliştirilmektedir. Netcraft isimli internet hizmetleri şirketinin yapmış olduğu 2011 Aralık Web Sunucuları Araştırması'na [2] göre Apache %65'lık pazar payı ile dünya çapında en çok kullanılan web sunucusudur. Birden çok platforma uyumlu olması ve arkasındaki camianın de etkisiyle Apache'nin 500'den fazla resmi eklenti modülü vardır [3]. Yüksek kullanım oranı göz önüne alındığında herhangi bir modül geliştirmek için Apache çok uygun bir ortam olarak göze çarpmaktadır. Ayrıca çıkan ürünün açık kaynak kod camiasına sağlayacağı katkı da düşünülerek bu çalışmada geliştirilecek modülün Apache web sunucusuna yönelik olması uygun görülmüştür.



Şekil 3 Netcraft Aralık 2011 Web Sunucuları Araştırmasına Göre Web Sunucuların Pazar Payları

4.2. Uygulanan Metodun Detayları

Internet robotlarının tespit edilmesi aslında gerçek kullanıcılardan gelen talepler ile otomatize taleplerin ayırt edilmesi olarak açıklanabilir. Bu ayırt etme işlemi Turing Test adı verilen insan davranışları ile bilgisayarın davranışlarını birbirinden ayırmak için yapılan teste de benzemektedir. Bağlantı Keşfi Önleyici (Anti-crawler) yazılımları da tam olarak bu noktaya odaklanmaktadır. Çünkü kullanıcı temelli HTTP taleplerinin, otomatize taleplerden ayırt edilmesi, internet robotlarının da tespit edilmesi anlamına gelecektir. Bu sebeple otomatize HTTP taleplerin ve kullanıcı kaynaklı HTTP taleplerinin karakteristikleri incelenmelidir.

Otomatize taleplerin en önemli karakteristiği hız ve yoğunluklarıdır. Bu araçların en önemli amaçları yapılacak işlemlerin süresini kısaltmak olduğundan, otomatize yazılımlar bir insanın gerçekleştiremeyeceği kadar yüksek miktarlardaki talebi, çok kısa bir süre zarfında gerçekleştirmektedirler. Bu sebeple talep hızı ve yoğunluğu internet robotu tespit etme işlemi sırasında kullanılabilecek yöntemlerden biridir.

Otomatize araçların bir diğer karakteristiği de web sayfasının kaynak kodu üzerinden çalışmalarıdır. Otomatik talep üreten araçlar, normal kullanıcılar gibi web sayfasının görünen yüzünden değil, kaynak kodundan işlem yaptıkları için, normal kullanıcıların göremeyecekleri birçok objeyi görebilmekte ve onlarla işlem yapabilmektedirler. Bu çalışma prensibi göz önüne alındığında sayfa kaynak koduna gömülecek bir gizli obje kullanılarak internet robotları tespit edilebilecektir.

Bu çalışmada yukarıda belirtilen iki farklı karakteristik kullanılarak internet robotu kaynaklı HTTP talepleri ayırt edilmeye çalışılmıştır.

4.2.1. Talep Yoğunluğu Modeli

Talep yoğunluğu modeli temel olarak belirli bir zaman dilimi içerisinde bir insanın yapamayacağı kadar sayıda talebin yapılması durumuna dayanmaktadır Böyle bir durumla karşılaşıldığı zaman sunucunun, taleplerin geldiği IP adresini engellemesi ve HTTP yanıtı olarak bir hata sayfası göstermesi beklenir.

Talep yoğunluğu modeli tasarlanırken bünyesinde iki adet liste barındırması planlanmıştır. Bu listelerden zaman tablosu olarak adlandırılan listede gelen taleplere ait IP adresi, geliş zamanı ve bu IP'den daha önce gelen talep sayısının tutulduğu sayaç bilgisi saklanmaktadır. Kara Liste adı verilen listede ise cezalı IP adresleri ve engelleme cezasının başladığı zaman damgası tutulmaktadır.

Sunucuya bir talep geldiğinde ilk olarak kara listeye bakılır. Eğer talebi yapan IP adresi bu kara listede yer alıyorsa ceza süresinin dolup dolmadığı kontrol edilir. Eğer ceza süresi dolmuşsa IP adresi kara listeden çıkartılıp zaman tablosuna eklenir ve sayaç değeri 1 arttırılır. Eğer ceza süresi dolmamışsa ceza başlangıç zamanı, geliş zamanı olarak güncellenir ve cevap olarak bir hata sayfası döndürülür.

Eğer talep yapan IP adresi kara listede bulunmuyorsa burada iki durum gerçekleşmiş olabilir.

- Bu IP adresinden ilk defa talep yapılıyor olabilir;
- IP adresi zaman tablosu listesinde olabilir.

Eğer talep yapan IP adresi zaman tablosu listesinde bulunuyorsa ilk yapılan talep ile şimdiki talep arasındaki zaman farkı hesaplanır. Bu zaman farkının modül konfigürasyonunda belirtilen maksimum zaman aralığı değerinden büyük olması durumunda sayaç sıfırlanır. İlk talep ile şindiki talep arasındaki zaman farkı, maksimum zaman aralığı değerinden küçük ise bu sefer sayaç değeri kontrol edilmelidir. Sayaç değeri maksimum zaman aralığı periyodunda yapılabilecek maksimum talep sayısından fazla ise bu taleplerin bir internet robotu kaynaklı olduğu kanaatine varılır, IP adresi zaman tabosu listesinden çıkartılıp kara listeye eklenir ve istemciye bir hata sayfası gösterilir. Geliş zamanı değeri de ceza başlangıç zamanı olarak atanır. Eğer sayaç değeri maksimum talep sayısından aşağıda ise bu durumda sayaç arttırılarak istemciye talepte bulunduğu sayfa gösterilir.

4.2.2. Gizli Tuzak Bağlantı Modeli

Talep sayısı temelli Talep yoğunluğu modeli, internet robotları dışında bazı saldırı türleri

için de koruma sağlamaktadır. Özellikle servis dışı bırakma (DoS) ve dağıtık servis dışı bırakma (DDoS) gibi saldırı türleri için koruma amaçlı kullanılmaktadır [4]. Literatür incelendiğinde bu tarz talep sayısı temelli bazı koruma metotlarının internet robotları için de belirli ölçülerde koruma sağladığı görülmüştür. Fakat daha yararlı ve daha verimli koruma yöntemleri geliştirmek için korunmak istenilen saldırıların genel özelliklerinin ve karakteristiklerinin analiz edilmesi gerekir.

DoS ve DDoS saldırılarında asıl amaç sunuculara karşılayamayacakları miktarda talep göndererek sunucunun cevap veremez veya erişilemez hale gelmesini sağlamaktır. Dolayısıyla gönderilen talebe ait yanıtların beklenmesi gerekmemektedir. Bu sebeple dağıtık servis dışı bırakma saldırılarında sahte kaynak IP'leri kullanılarak talepler sanki bu sahte IP adreslerinden geliyormuş gibi yapılarak saldırının etkisi arttırılmaktadır. Dolayısıyla servis dışı saldırılarında sahte kaynak IP adreslerine dönecek yanıtlar ve içeriklerinin ise hiçbir önemi ve anlamı yoktur.

Fakat internet robotlarının çalışma mantığı tamamen farklıdır. İnternet robotları gönderilen bir HTTP talebine sunucunun verdiği HTTP yanıtı ile ilgilenmekte, dönen yanıtın koduna göre sayfanın var olup olmadığını anlamaktadırlar. Aynı zamanda dönen yanıtın gövde kısmı robotlar tarafından ayrıştırılarak yeni bağlantı adresleri gövde kısmından ayıklanmaktadır.

Gizli tuzak bağlantı (link) modeli, internet robotlarının çalışma mantığı olan yanıt odaklı çalışma stratejisine uygun olan bir yaklaşımdır. Bu model iki ana kısımdan oluşmaktadır. Birinci ana kısım istemciye döndürülecek olan sayfa içerisine gizli bir HTML objesi enjekte etme işlemidir. İkinci kısım ise bu enjekte edilen HTML objesinin çağırıldığı durumları yakalama işlemidir.

Birinci kısım olan HTML objesi enjekte etme adımı bu modelin en önemli adımıdır. Bu çalışmada enjekte edilecek HTML objesi olarak HTML bağlantıları kullanılmaktadır. Çünkü internet robotları, sayfalardaki HTML bağlantılarını ziyaret etmekte ve işlemlerini devam ettirmek için bağlantı adreslerini kullanmaktadırlar. Bu karakteristik sebebiyle internet robotları için hazırlanacak bir tuzak objenin, HTML bağlantısı olması iş uygun bir yaklaşım olarak göze çarpmaktadır.

İstemciye döndürülecek olan sayfa içerisinde HTML bağlantı adresi enjekte etmek için en etkili ve kolay çözüm yerine koyma (substitution) yöntemidir. Bu yöntem, tüm HTTP yanıt gövdesinin ayrıştırılması ve çok kullanılan bir anahtar kelimenin sonuna veya başına bir bağlantı adresi eklenmiş hali ile değiştirilmesine dayanır. Başarılı bir yerine koyma işlemi için anahtar kelimenin doğru seçilmesi gerekmektedir. Sayfa gövdesinden aranan anahtar kelimenin bulunamaması yer değiştirme işlemi gerçekleştirilemeyeceği yani HTML bağlantı adresi enjeksiyonunun başarısız olacağı anlamına gelir. Tüm model bu gizli bağlantı adresi enjeksiyonuna dayandığı için bu çalışmada, anahtar kelime olarak <html> </html>, <head> </head>, <title> </title> ve <body> </body> gibi HTML açılış ve kapanış etiketleri tercih edilmiştir.

Yerine koyma işlemi aşağıdaki gibi özetlenebilir.

- Bir HTML etiketi seçilir. Örneğin *</title>*
- *</title>* etiketi *</title><a href=hiddenurl.html style=display:none; />* ile değiştirilir.

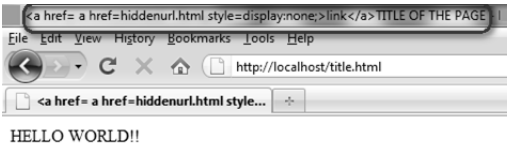
Bu işlemdeki en önemli parametrelerden biri *style=display:none;* parametresidir. Bu parametre bu bağlantı adresinin kullanıcılar tarafından görünmez olmasını sağlamaktadır. Örnek bir bağlantı adresi Şekil 4'de gösterilmiştir.

```
<html xmlns="http://www.w3.org/1999/xhtml"
xml:lang="en" lang="en">
<head>
<title>TITLE OF THE PAGE</title><a
href=hiddenurl.html style=display:none />
<meta http-equiv="Content-Type"
content="text/html; charset=utf-8">
</head>

<body>
   <p> HELLO WORLD!!</p>
   <img src="/images/banner2.gif">
   <br />
   <a href="index.php"></a>
   <a href="index.php"></a>
   <h1 align="center">HEADING (H1)</h1>
</body>
</html>
```

Şekil 4 Bağlantı Enjeksiyonu Sonrası Örnek Bir Kod

Bağlantı enjeksiyonunda önemli noktalardan biri de açılış ve kapanış etiketlerinden doğru ve uygun olanını seçmektir. Bu seçim sayfanın görünümünde bir bozukluğa sebep olmayacak şekilde yapılmalıdır. Örneğin yukarıda verilen örnekte </title> kapatma etiketi yerine <title> açma etiketi anahtar kelime olarak kullanılmış olsaydı web sayfasının tarayıcının üst kısmında görünen başlık kısmı Şekil 5'de görüldüğü gibi bozulmuş olarak görünecekti. Bağlantı enjeksiyonunda en önemli husus sayfa görünüşüne zarar verilmeden bu işlemin gerçekleştirilmesi olduğundan doğru anahtar kelime seçimi hayati önem arz etmektedir.



Şekil 5 Anahtar Kelime Olarak Yanlış Bir Etiket Seçilmesi Sonucu Sayfa Görünüşünün Bozulması

Gizli tuzak bağlantı modelinde ikinci kısım sunucuya gelen HTPT taleplerinde, enjekte edilen adresin istendiği talebin yakalanması işlemidir. Bu bağlantının çağırılması, bu talebi yapan istemcinin bir internet robotu olduğu kanaatini oluşturur. Çünkü bu bağlantı kullanıcılar tarafından kaynak kod incelenmediği taktirde görülemeyen bir bağlantıdır ve kaynak koddaki bağlantıların takip edilmesi genel bir kullanıcı davranışını yansıtmamaktadır. Bu yöntem ile gerçek kullanıcılar ile internet robotları birbirinden davranış farkı sebebiyle ayrılmış olur.

4.3. mod_antiCrawl Uygulaması

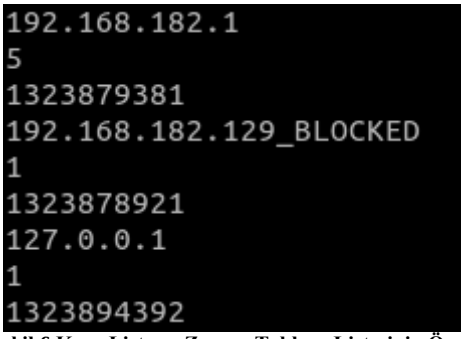
mod_antiCrawl, C dili ve Apache API'si [5] kullanılarak geliştirilmiş ve Gizli Tuzak Bağlantı Modeli ile Talep Yoğunluğu Modeli'ni kapsayan bir Apache modülüdür. Bu modüle ait akış diyagramı Ek-1'de bulunan Şekil 8'de gösterilmiştir.

4.3.1. Özet Tabloları

Bu çalışmada özet tablolarının (hash table) diğer veri yapılarına göre en büyük avantajı olan hız avantajından faydalanmak için, kullanılan kara liste ve zaman tablosu listeleri özet tablosu olarak tutulmuştur. Çünkü düzgün dağılımlı bir özet tablosu üzerinde arama, ekleme ve silme işlemleri, içerisindeki eleman sayısından bağımsız bir şekilde sabit bir zaman aralığı içerisinde gerçekleştirilebilmektedir. Ek olarak özet tablosunun daha düzgün dağılımlı olabilmesi için özet tablosu boyutu ve modül değeri asal sayı olarak seçilmiştir. Özet tablolarındaki cakışma problemi için ise, özet tablosunun her bir elemanı içerisinde bir bağlı liste (linked list) yapısı kullanılmıştır. Bu yapı sayesinde aynı indeks değerinin işaret ettiği her bölüm, kendi içerisinde verinin bağlı liste yapısında tutulmasını sağlayacak ve böylece çakışan değerler aynı bölüm içerisinde tutulabilecektir.

Uygulamanın geliştirme aşamasında kara liste ve zaman tabosu listelerininfarklı farklı özet tablolarında tutulması yerinde aynı özet tablosu içerisinde tutulması uygun görülmüştür. Bu sebeple kara listeye ait girdilerin zaman tablosu listesine ait

girdilerden ayrılabilmesi için kara liste girdileri özet tablosuna eklenirken <**IP_ADRESI**>**_BLOCKED** şeklinde IP adresinin sonuna _BLOCKED son eki kullanılarak eklenmektedir.



Şekil 6 Kara Liste ve Zaman Tablosu Listesinin Özet Tablosundaki Görünümü

İçerisinde hem kara listeye alınmış hem de zaman tablosunda yer alan verilerin bulunduğu bir özet tablosu ait örnek bir görüntü Şekil 6'de gösterilmiştir. Bu örnekte görülen 192.168.182.129 IP adresinin kara listeye alındığı son ek olarak aldığı BLOCKED ifadesinden anlaşılmaktadır. 192.168.182.1 IP adresinin ise daha önce 5 adet talepte bulunduğu ve ilk talep zamanının time_t tipinden 1323879281 saniye olduğu görülmektedir. time_t tipi, o verinin 1 Ocak 1970 saat 00:00 ile arasındaki zaman farkına işaret etmektedir.

5 Değerlendirme

Bu çalışma kapsamında içerisinde 2785 adet sayfa barındıran, gerçeğe eş bir web uygulaması üzerinde performans testleri gerçekleştirilmiştir.

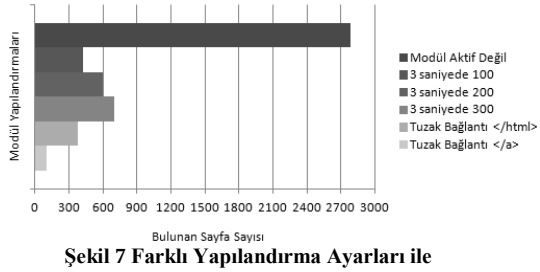
mod_antiCrawl modülünün devre dışı bırakıldığı ve farklı yapılandırma ayarları kullanılarak aktive edildiği durumlarda bir internet robotunun bu sayfalardan kaç tanesini bulabildiği değerlendirilmiştir.

Bu bağlamda mod_antiCrawl 5 ayrı ayarda çalıştırılmıştır. Bu ayarlar;

- talep yoğunluğu öncelikli
  - 3 saniye içerisinde maksimum 100 talep
  - 3 saniye içerisinde maksimum 200 talep
  - 3 saniye içerisinde maksimum 300 talep
- tuzak bağlantı öncelikli
  - Tuzak bağlantı </html> kapanış etiketinden önce
  - Tuzak bağlantı etiketinden sonra

olarak sınıflandırılabilmektedir. 3 saniye içerisinde 100, 200 ve 300 talepten fazla talepte bulunulması durumlarını ele alan testlerde internet robotu, 3 saniyede 100 yapılandırması ile 2785 sayfadan 419'unu; 3 saniyede 200 yapılandırması ile 602'sini; 3 saniyede 300 yapılandırması ile 698'ini tespit edebilmiştir.

Gizli tuzak bağlantı enjeksiyonu için ise iki farklı yapıda test yapılmıştır. Bu yapılardan birincisinde enjekte edilecek bağlantı </html> kapatma etikeninin önüne eklenmiştir ve test sonucunda internet robotunun 2785 sayfadan 374'ünü tespit edebildiği görülmüştür. Bu kapsamda yapılan ikinci testte ise enjekte edilecek bağlantı kapatma etiketinden sonra enjekte edilmiştir. Bu sayede sayfadaki her bağlantının ardına bir adet gizli tuzak bağlantı eklenmesi sağlanmıştır.



Şekil 7 Farklı Yapılandırma Ayarları ile mod_antiCrawl Performans Grafiği

Yapılan bu son testte internet robotunun 2785 sayfanın sadece 104 tanesini tespit edebildiği görülmüştür. Bu testlerde elde edilen verilere ait grafik Şekil 7'de gösterilmiştir.

Bu sonuçlara göre;

1. Tuzak bağlantının </html> kapatma etiketinden önce konulduğu durumda, eklenen tuzak bağlantının sayfanın en son kısmında bulunmasına karşın; internet robotlarının ayrıştırma işlemine sayfanın üstünden başlaması sebebiyle tuzak bağlantıya ulaşana kadar birçok bağlantıyı keşfettiği görülmüştür. Bu problemin tuzak bağlantının etiketlerinden sonra yerleştirilerek sayfanın birçok yerinde dağılması ile olumlu bir biçimde çözüldüğü gözlemlenmiştir. Sonuç olarak tuzak bağlantının sayfada bulunma sayısı ve sayfa içerisindeki konumunun modülün performansında büyük bir etkiye sahip olduğu görülmüştür.
2. Talep sayısı odaklı yaklaşımın, tuzak bağlantı temelli yaklaşıma kıyasla daha kötü bir performans sergilemesi neticesinde, tuzak bağlantı odaklı yaklaşımın internet robotlarının çalışma karakteristiğine daha uygun olduğu sonucuna varılmıştır.
3. Yapılan testlerde mod_antiCrawl modülünün en kötü performans sergilediği yapılandırma ayarlarında bile, deaktif olduğu duruma oranla bulunan sayfa sayısında yaklaşık %75'lik bir azalma sağladığı görülmüştür. Bu oranın tuzak bağlantı özelliğinin aktif olduğu durumda %95 seviyesine kadar çıktığı gözlenmiştir.

Ayrıca mod_antiCrawl modülü için yapılan performans testlerinde tek bir kaynaktan yapılan 1000 adet sıralı talep ile 5 ayrı kaynaktan yapılan toplamda eşzamanlı 5000 adet talebe test sunucusunun yanıt verme süreleri karşılaştırılmıştır.

Bu testlerden çıkan sonuçlara göre

1. Geliştirilen mod_antiCrawl modülü deaktif konumdayken test sunucusu tek bir kaynaktan gelen 1000 adet talebi ortalama olarak talep başına 880 mikrosaniye sürede yanıtlayabilmekte iken; mod_antiCrawl aktif konuma getirildiğinde bu değer yaklaşık 3400 mikrosaniye seviyesine ulaşmaktadır. Bu farkın, eş zamanlı 5 istemcinin simule edildiği testlerde daha da arttığı görülmüştür. Bu test, mod_antiCrawl modülünü talep yapan istemci sayısı arttığında daha yavaş yanıt vermeye başladığını göstermektedir. Birden fazla talebin aynı anda işlenmesi sırasında paylaşılan kaynakların kilitlenmesi ve kilidin kaldırılmasının beklenmesi yanıt verme süresini olumsuz olarak etkilemektedir.
2. mod_antiCrawl, çalışması için gerekli olan özet tablolarını dosya sistemi üzerindeki bir dosyada tutmaktadır. Dosyalar üzerinde yazma okuma işlemi, bellekten yapılan yazma okuma işlemine göre çok daha düşük performanslı bir işlem olduğundan modül aktive edildiğinde sunucuda mikrosaniyeler mertebesinde bir performans düşüşü gözlemlenmiştir.

6 Sonuç ve Öneriler

İnternet robotları yanlış amaçlar için kullanılmadıkları sürece faydalı ve bilginin kolay toplanmasını ve etkin kullanılmasını sağlayan araçlardır. Ancak bu araçların kötü amaçlar için kullanılması durumunda ortaya çıkabilecek olumsuz durumlardan hem bireysel anlamda hem de kurumsal ve sistemsel anlamda korunmak gerekmeketdir. mod_antiCrawl modülü Apache sunucuları için bu amaçla geliştirilmiş bir eklentidir. Geliştirme aşamasında internet robotlarının çalışma karakteristikleri incelenmiş ve bu bağlamda çözümler geliştirilmiştir.

mod_antiCrawl içerisinde gizli tuzak bağlantı yakalama ve yoğun talep kaynaklarını yakalama gibi iki temel özellik hayata geçirilmiştir. Yapılan performans testlerine bu modülün internet robotlarını tespit etme ve engelleme yeteneğinin yüksek olduğu görülmüştür.

Modülün performans ve kabiliyetlerinin arttırılması amacıyla, gelecekte yapılması planlanan geliştirlmeler aşağıda listelenmiştir:

- Sabit gizli tuzak bağlantılar yerine daha akıllı ve gelişmiş internet robotlarının bu tuzak bağlantıları öğrenerek bunları istememe gibi bir eğilim göstermelerinin önüne geçmek için her sayfaya tamamen rastgele üretilmiş tuzak bağlantıların enjekte edilmesi.
- Modülde gerçeklenmiş olan engelleme amaçlı tekniklere ilaveten, internet robotlarını kısır döngü içerisine sokup taramalarının güvenilirlikten uzak ve verimsiz ve hale gelmesini sağlayacak oyalama tekniklerinin geliştirilmesi.
- Kaynak ve bellek yönetiminde iyileştirmeler yapılması.
- Özet tablolarının bir dosyada tutulması yerine, paylaşımlı bellekte (shared memory) tutulması.

mod_antiCrawl çalışması, yukarıda açıklanan geliştirmeler ile birlikte hem açık kaynak kod camiası açısından hem de akademik açıdan yararlı bir araç olacaktır. mod_antiCrawl, çok fazla dikkate alınmayan internet robotlarının zararlı faaliyetlerinin önüne geçmek için önemli bir adım niteliği taşımakta olup, planlanan çalışmalarla daha da etkin hale gelecektir.

7 Kaynaklar

[1] A. Doup´e, M. Cova ve G. Vign, ***Why Johnny Can't Pentest:An Analysis of Black-Box Wen Vulnerabilty Scanners,*** University of California, Santa Barbara (2010)

[2] Netcraft, ***December 2011 Web Server Survey*** (2011)

[3] Apache Modules, ***https://modules.apache.org/search.php?query=true***, Apache

[4] Jonathan Zdziarski, ***mod_evasive, http://www.zdziarski.com/blog/?page_id=442*** (2010)

[5] Apache API, ***http://httpd.apache.org/docs/2.0/developer/***, Apache

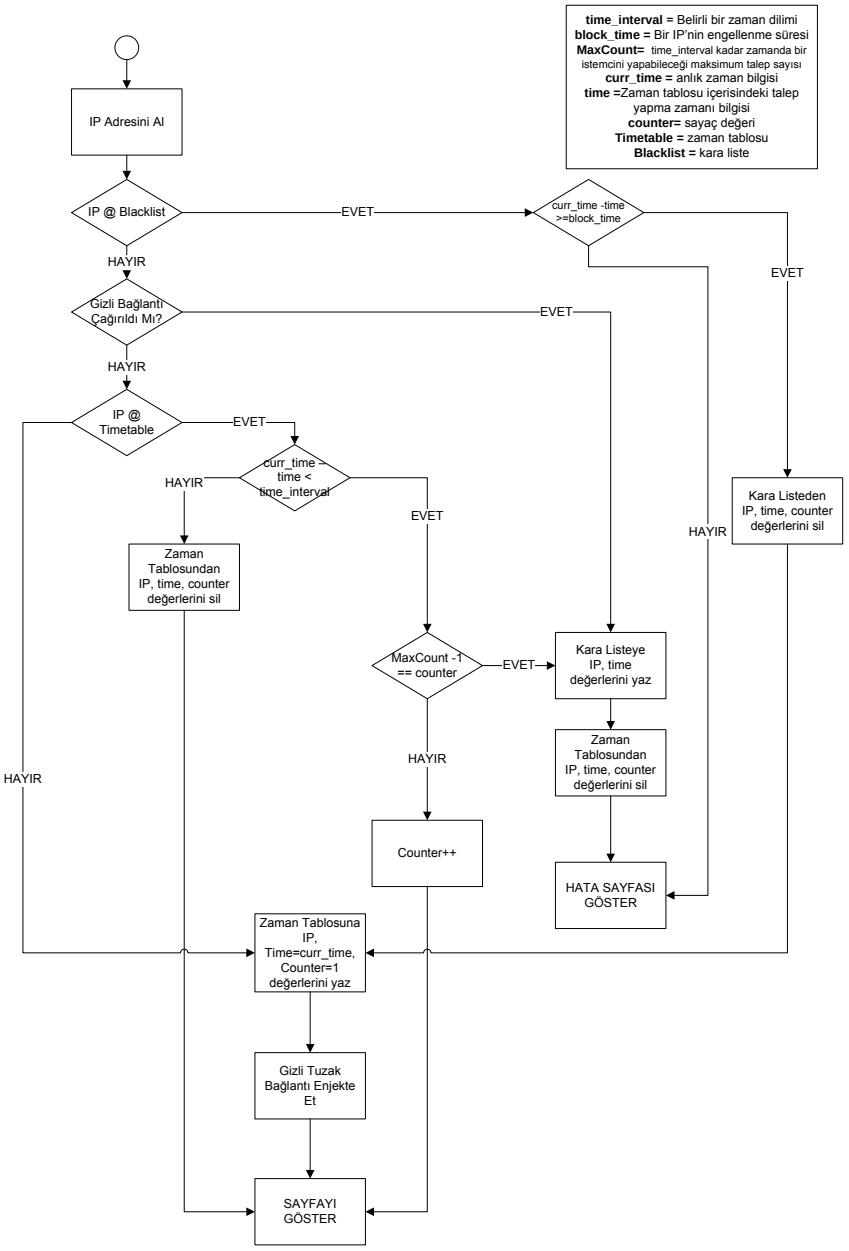
[6] J. Bau, E. Bursztein, D. Gupta, J. Mitchell***, State of the Art: Automated Black-Box Web Application Vulnerability Testing,*** Standford University (2010)

[7] B.Laurie, P.Laurie, ***Apache Definitive Guide***, O'REILLY (2002)

[8] N.Kew, ***The Apache Modules Book, Application Development with Apache***, PRENTICE HALL (2007)

[9] G.Pant, P.Srinivasan, F.Menczer, ***Crawling the Web***, Springer-Verlag (2003)

Ek-1



Şekil 8 mod_antiCrawl Akış Diyagramı