Testing for HTTP Splitting Smuggling (HTTP Bölme Kaçakçılığı Testi)

Summary (Özet)

Bu bölüm, HTTP protokolünün belirli özelliklerinden yararlanan saldırı örneklerini, web uygulamasının zayıflıklarından veya farklı ajanların HTTP mesajlarını yorumlama biçimindeki özelliklerinden yararlanarak göstermektedir.
Bu bölüm, belirli HTTP başlıklarını hedef alan iki farklı saldırıyı analiz edecektir:

- HTTP bölme
- HTTP kaçakçılığı

İlk saldırı, bir davetsiz misafirin CR ve LF karakterlerini uygulama yanıtının başlıklarına sokmasını ve iki farklı HTTP mesajına cevap veren 'bölünmesine' izin veren girdi sanitizasyonu eksikliğinden yararlanır. Saldırının amacı bir önbellek zehirlenmesinden site komut dosyasına kadar değişebilir.

İkinci saldırıda saldırgan, özel olarak hazırlanmış bazı HTTP mesajlarının, onları alan ajana bağlı olarak farklı şekillerde ayrıştırılıp yorumlanabileceği gerçeğini istismar eder. HTTP kaçakçılığı, HTTP mesajlarını (web server, proxy, firewall) ele alan farklı ajanlar hakkında bir miktar bilgi gerektirir ve bu nedenle yalnızca gri kutu test bölümüne dahil edilecektir.

Test Objectives (Test Hedefleri)

- Uygulamanın bölünmeye karşı savunmasız olup olmadığını değerlendirin, hangi olası saldırıların mümkün olduğunu belirleyin.
- İletişim zincirinin kaçakçılığa karşı savunmasız olup olmadığını değerlendirin, hangi olası saldırıların mümkün olduğunu belirleyin.

How To Test (Nasıl Test Edilir)

Black-Box Testing (Siyah-Kutu Testi)

HTTP Splitting (HTTP Bölünme)

Bazı web uygulamaları, bazı başlıkların değerlerini oluşturmak için kullanıcı girişinin bir kısmını kullanır. En basit örnek, hedef URL'nin kullanıcı tarafından gönderilen bazı değerlere bağlı olduğu
yönlendirmelerle sağlanır. Örneğin, kullanıcının standart veya gelişmiş bir web arayüzünü tercih edip etmediklerini seçmeleri istendiğini söyleyelim. Seçim, karşılık gelen sayfaya yönlendirmeyi tetiklemek için yanıt başlığında kullanılacak bir parametre olarak kabul edilecektir.

Daha spesifik olarak, parametre 'ara yüz' değeri 'ilerlemiş'se, uygulama aşağıdakilerle cevap verecektir:

```
HTTP/1.1 302 Moved Temporarily
Date: Sun, 03 Dec 2005 16:22:19 GMT
Location: http://victim.com/main.jsp?interface=advanced
<snip>
```

Bu mesajı alırken tarayıcı, kullanıcıyı Konum başlığında belirtilen sayfaya getirir. Bununla birlikte, uygulama kullanıcı girişini filtrelemezse, farklı çizgileri ayırmak için kullanılan CRLF dizisini
temsil eden diziyi %0d%0d%0aa 'çayla yüzlü' parametreyi eklemek mümkün olacaktır. Bu noktada, testçiler, onu ayrıştıran herkes tarafından iki farklı yanıt olarak yorumlanacak bir yanıtı tetikleyebilecekler, örneğin uygulama ile aramızda oturan bir web önbelleği. Bu, bu web önbelleğini
zehirlemek için bir saldırgan tarafından kullanılabilir, böylece sonraki tüm isteklerde yanlış içerik sağlar. Diyelim ki önceki örnekte testçi arabirim parametresi olarak aşağıdaki verileri aktarıyor:

```
advanced%0d%0aContent-Length:%200%0d%0a%0d%0aHTTP/1.1%20200%20OK%0d%0aContent-Type:%20text/html%0d%0aContent-Length:%2035%0d%0a%0d%0a<html>Sorry,%20System%20Down</html>
```

Bu nedenle, savunmasız başvurunun ortaya çıkan cevabı şu olacaktır:

```
HTTP/1.1 302 Moved Temporarily
Date: Sun, 03 Dec 2005 16:22:19 GMT
Location: http://victim.com/main.jsp?interface=advanced
Content-Length: 0
```

```
HTTP/1.1 200 OK
Content-Type: text/html
Content-Length: 35

<html>Sorry,%20System%20Down</html>
<other data>
```

Web önbelleği iki farklı yanıt görecektir, bu nedenle saldırgan ilk istekten hemen sonra gönderirse, ikincisi bir tane ister. `/index.html` Web önbelleği bu isteği ikinci yanıtla eşleştirecek ve içeriğini önbelleğe alacak, böylece sonraki tüm istekleri yönlendirecektir. `victim.com/index.html` Bu
web önbelleğinden geçen "sistem aşağı" mesajını alacaktır. Bu şekilde, bir saldırgan, web önbelleği web uygulaması için ters bir proxy ise, bu web önbelleğinin tüm İnternet'i kullanarak siteyi etkili bir şekilde çözebilir.

Alternatif olarak, saldırgan bu kullanıcılara çerezleri çalmak için bir çapraz site komut dosyası saldırısı düzenleyen bir JavaScript snippet'i geçebilir. Güvenlik açığı uygulamadayken buradaki hedefin kullanıcıları olduğunu unutmayın. Bu nedenle, bu güvenlik açığını aramak için, test cihazının yanıtta bir veya daha fazla kafayı etkileyen tüm kullanıcı kontrollü girdileri tanımlaması ve içine bir CR + LF dizisinin başarılı bir şekilde enjekte edip edemeyeceğini kontrol etmesi gerekir.

Bu saldırı için en olası adaylar olan başlıklar şunlardır:

- `Location`
- `Set-Cookie`

Gerçek bir dünya senaryosunda bu kırılganlığın başarılı bir şekilde sömürülmesinin oldukça karmaşık olabileceği unutulmamalıdır, çünkü birkaç faktör dikkate alınmalıdır:

1. Kalem test cihazı, başlıkları başarılı bir şekilde önbelleğe alınması için sahte yanıtta düzgün bir şekilde ayarlamalıdır (örneğin,gelecekte belirlenen bir tarihe sahip bir Son Modifiye başlık). Ayrıca,bir ön talep yayınlayarak, hedef sayfalayanların önceden önbelleğe alınmış sürümlerini yok etmek zorunda kalabilirler. `Pragma: no-cache` İstek başlıklarında
2. Uygulama, CR + LF dizisini filtrelemek için filtrelemezken, başarılı bir saldırı için gerekli olan diğer karakterleri filtreleyebilir (örneğin, `<` ve `>`). Bu durumda,

testçi diğer kodlamaları kullanmaya çalışabilir (örneğin, UTF-7)

3. Bazı hedefler (örneğin, ASP) Konum başlığının yol kısmını URL-takolar (örneğin, `www.victim.com/redirect.asp`), bir CRLF dizisi işe yaramaz hale getirmek. Ancak, sorgu bölümünü kodlamayı başaramazlar (örneğin,? arayüz = ileri), yani önde gelen bir soru işareti bu filtrelemeyi atlamak için yeterlidir

Bu saldırı ve olası senaryolar ve uygulamalarla ilgili diğer bilgiler hakkında daha ayrıntılı bir tartışma için, bu bölümün alt kısmında belirtilen kağıtları kontrol edin.

Gray-Box Testing (Gri-Kutu Testi)

HTTP Splitting (HTTP Bölünme)

HTTP Spliting'in başarılı bir şekilde sömürülmesi, web uygulamasının ve saldırı hedefinin bazı ayrıntılarını bilerek büyük ölçüde yardımcı olunmaktadır. Örneğin, farklı hedefler, ilk HTTP mesajının ne zaman biteceğine ve ikincisinin ne zaman başlayacağına karar vermek için farklı yöntemler kullanabilir. Bazıları önceki örnekte olduğu gibi mesaj sınırlarını kullanacaktır. Diğer hedefler, farklı mesajların farklı paketler tarafından taşınacağını varsayacaktır. Diğerleri her mesaj için önceden belirlenmiş uzunlukta bir dizi parça tahsis edecektir: bu durumda, ikinci mesajın tam olarak bir parçanın başında başlaması gerekecek ve bu, test cihazının iki mesaj arasında dolgu kullanmasını
gerektirecektir. Bu, savunmasız parametre URL'de gönderildiğinde bazı sorunlara neden olabilir, çünkü çok uzun bir URL'nin kesilmesi veya filtrelenmesi muhtemeldir. Gri kutu senaryosu saldırganın geçici bir bulmasına yardımcı olabilir: Örneğin, birkaç uygulama sunucusu, talebin GET yerine POST kullanılarak gönderilmesine izin verecektir.

HTTP Smuggling (HTTP Kaçakçılığı)

Girişte belirtildiği gibi, HTTP Kaçakçılığı, özellikle hazırlanmış bir HTTP mesajının farklı ajanlar (gözetlendiriciler, web önbellekleri, uygulama güvenlik duvarları) tarafından ayrılabileceği ve
yorumlanabileceği farklı şekillerde yararlanır. Bu nispeten yeni saldırı ilk olarak 2005 yılında Chaim Linhart, Amit Klein, Ronen Heled ve Steve Orrin tarafından keşfedildi. Birkaç olası uygulama var ve en muhteşem olanlardan birini analiz edeceğiz: bir uygulama güvenlik duvarının baypas. Daha ayrıntılı bilgi ve diğer senaryolar için orijinal whitepaper'a (bu sayfanın altındaki) bakın.

Application Firewall Bypass (Uygulama Güvenlik Duvarı Bypass)

Bir sistem yönetiminin, talebe gömülü olan bilinen bazı kötü amaçlı kalıplara bağlı olarak düşmanca bir web talebini tespit etmesini ve engellemesini sağlayan birkaç ürün vardır. Örneğin, bir saldırganın bir istekte bulunarak bir kişinin www root'u kırabileceği IIS sunucusuna karşı rezil, eski Unicode dizin fragman saldırısını düşünün:

`http://target/scripts/..%c1%1c../winnt/system32/cmd.exe?/c+<command_to_execute>`

Tabii ki, bu saldırıyı URL'deki ".." ve "cmd.exe" gibi iplerin varlığına göre tespit etmek ve filtrelemek oldukça kolaydır. Bununla birlikte, IIS 5.0, vücudu 48K baytlara kadar olan ve İçerik-Type başlığının uygulama / x-www-form-rülen kodludan farklı olduğu bu sınırın ötesinde olan tüm içeriği kesen POST istekleri hakkında oldukça seçicidir. Pen-tester, aşağıdaki gibi yapılandırılmış çok büyük bir istek oluşturarak bundan yararlanabilir:

```
POST /target.asp HTTP/1.1       ← Request #1
Host: target
Connection: Keep-Alive
Content-Length: 49225
<CRLF>
<49152 bytes of garbage>
```

```
POST /target.asp HTTP/1.0       ← Request #2
Connection: Keep-Alive
Content-Length: 33
<CRLF>
```

```
POST /target.asp HTTP/1.0       ← Request #3
xxxx: POST /scripts/..%c1%1c../winnt/system32/cmd.exe?/c+dir HTTP/1.0   ← R
quest #4
Connection: Keep-Alive
<CRLF>
```

Burada olan şey, `Request #1` 49223 bayttan yapılır, bu da çizgileri de içerir `Request #2` . . Bu nedenle, bir güvenlik duvarı (veya IIS 5.0 dışındaki herhangi bir ajan) İstek #1'i görecektir, `Request #2` (Verileri #1'in sadece bir parçası olacak), görecek `Request #3` ve ıskalama `Request #4` (Çünkü POST sahte başlığın sadece bir parçası olacak xxxx).

Şimdi, IIS 5.0'a ne olacak? Oymayı durduracak `Request #1` 4152 bayt çöpten hemen sonra (48K = 49152 bayt sınırına ulaşmış olacağından) ve bu nedenle ayrıştırılacaktır. `Request #2` Yeni, ayrı bir istek olarak. `Request #2` İçeriğinin 33 bayt olduğunu iddia ediyor, bu da her şeyi içeren "xxxx: "", IIS'i özleyen `Request #3` (bir parçası olarak yorumlandı `Request #2`) ama nokta `Request #4` , POST'u 33. byte'den hemen sonra başlarken veya `Request #2` . . Bu biraz karmaşıktır, ancak mesele, saldırı URL'sinin güvenlik duvarı tarafından algılanmayacağı (önceki bir isteğin gövdesi olarak yorumlanacaktır) ancak IIIS tarafından doğru bir şekilde ayrıştırılacağı (ve yürütülecektir).

Yukarıda bahsedilen durumda teknik bir web sunucusunun bir hatasından yararlanırken, farklı HTTP özellikli cihazların 1005 RFC uyumlu olmayan mesajları ayrıştırmasının farklı yollarından yararlanabileceğimiz başka senaryolar da vardır. Örneğin, HTTP protokolü yalnızca bir İçerik Boyu
Başlıka izin verir, ancak bu başlığın iki örneğine sahip bir mesajın nasıl ele alınacağını belirtmez. Bazı uygulamalar ilkini kullanacak, diğerleri ikincisini tercih edecek ve HTTP Kaçakçılığı saldırılarının önünü açacak. Başka bir örnek, İçerik Boyu Başlık başlığının bir GET mesajında kullanılmasıdır.

HTTP Kaçakçılığının Yaptış Yaptığını Unutmayın `*not*` Hedef web uygulamasında herhangi bir güvenlik açığından yararlanın. Bu nedenle, bir kalem testinde, müşteriyi yine de bir karşı önlemin
aranması gerektiğine ikna etmek biraz zor olabilir.

References (Referanslar)

Whitepapers (Beyaz Kağıtlar)

- Amit Klein, "Divide and Conquer: HTTP Response Splitting, Web Cache Poisoning Attacks, and Related Topics"
- Amit Klein: "HTTP Message Splitting, Smuggling and Other Animals"

- Amit Klein: “HTTP Request Smuggling - ERRATA (the IIS 48K buffer phenomenon)”
- Amit Klein: “HTTP Response Smuggling”
- Chaim Linhart, Amit Klein, Ronen Heled, Steve Orrin: “HTTP Request Smuggling”