Missing Content-Type Header

Content-Type hem http request'te hem de http response'da paketin body'sindeki dosyanın türünü belirtmek için kullanılan bir header'dır. Bir dosya upload'layacaksak upload'ladığımız dosyanın türü http request'teki Content-Type header'ı ile gider. Eğer bir tarayıcıdan bir dosya görüntüleyeceksek (talep edeceksek) sunucudan gelen dosyanın türü bilgisi http response'daki Content-Type header'ında gelir.

Missing Content-Type header zafiyeti http request'ten ziyade http response'daki Content-Type header'ının olmayışını ifade etmektedir. Bu zafiyeti anlamak için önce http request'teki Content-Type'ın nasıl istismar edilebileceğini, sonra ise http response'daki Content-Type eksikliğinin nelere sebep olabileceğini görelim.

Diyelim ki hedef web sunucusunda file upload zafiyeti mevcut. Hedef web sunucusuna çerez çalan kodların yer aldığı bir javascript dosyası yollamak istiyoruz. Fakat file upload mekanizması .js uzantılı dosyaları kabul etmiyor olsun ve sadece resim dosyalarını kabul ediyor olsun. Bu durumda javascript dosyamızın uzantısını .jpeg yapabiliriz.

xss-codes.js => xss-codes.jpeg

Ardından dosyayı upload'la deyip burp'le araya girerek http request'teki content-type header'ını sunucunun kabul ettiği resim uzantısı ile doldurabiliriz (örn; image/jpeg)

Http Request Sample:

…
…
Content-Type: image/jpeg

((Javascript dosyası içeriği))

Eğer hedef web uygulaması gelen dosyanın uzantısını http request'deki Content-Type header'ına bakarak karar veriyorsa bu durumda image/jpeg'i görüp dosya resim dosyasıdır diyecektir ve dosyanın upload'lanmasına izin verecektir. Diyelim ki öyle oldu ve javascript dosyamız hedef web sunucusuna upload'landı ve upload'ladığımız dosyayı tarayıcıdan görüntülemeye (talep etmeye) çalıştık. Tarayıcılar varsayılan olarak sunucudan gelen dosyaların uzantısını içindeki veriye bakarak belirleme methodu güttüklerinden dosyamız resim dosyası görünse bile içi javascript kodları ile dolu olduğundan tarayıcı dosyayı javascript olarak çalıştıracaktır. Dolayısıyla bu dosyayı tarayıcılarında görüntüleyen herkes xss saldırısına maruz kalacaktır. Halbuki dosyanın geldiği http response'da *Content-Type* header'ı ile dosyanın türünü belirtseydik ve beraberinde *X-Content-Type-Options: nosniff* ile tarayıcının uzantı analiz etme işlevini dosya içine bakarak değil de response'daki Content-Type header'ına bakarak yap deseydik tarayıcı dosya uzantısını http response'daki Content-Type header'ına bakarak belirleyecekti ve buna göre dosyayı çalıştıracağından xss-codes.jpeg dosyamız javascript olarak değil, resim olarak çalıştırılacaktı.

Böylece olası bir xss saldırısının önüne geçilmiş olacaktı.

Dolayısıyla diyebiliriz ki sunucun upload mekanizlarını atlatmak için http request'teki Content-Type header'ı manipule edildiğinden http request'teki content-type header'ı baz alınacak bir header değildir. Fakat tarayıcılar gelen dosyaları görüntülemeye çalıştıkları durumlarda dosya türünü belirleme ve ona göre çalıştırma işini sunucudan gelen http response'daki Content-Type header'ını baz alarak yapmalıdırlar. Diğer türlü sunucuda zararlı bir dosya varsa tarayıcı o dosyayı sunucunun zannetiği halde değil de zararlı haliyle çalıştıracaktır. Sonuç olarak geliştiriciler upload'lamada dosya türü kontrolünü http request'teki content-type'a göre kurgulamamalıdırlar. Ancak tarayıcıdan sunucudaki bir dosya talep edildiğinde tarayıcının dosya türünü belirlemesi ve ona göre çalıştırması olayı http response'daki content-type'a bakarak olmalıdır. Böylece sunucuda zararlı bir dosya olsa bile o dosya tarayıcıdan talep edildiğinde tarayıcı dosyayı content-type header'ında belirtilen uzantıya göre çalıştıracağından (örneğin javascript dosyamızı resim olarak çalıştıracağından) istemci zarar görmeyecektir.

Sonuç

Content-Type talep ve yanıt başlığı kaynağın türünü belirtmek için kullanılır. X-Content-Type-Options yanıt başlığı ise Content-Type başlığının önerdiği kaynak türünün değiştirilmemesini ve ona uyulmasını belirtmek için sunucu tarafından tarayıcı verilen bir uyarıdır. Böylece kaynak türü oynamaları ile tarayıcıda kaynakların farklı yorumlanmasının (çalıştırılmasının) önüne geçilmiş olur.

Missing Content-Type Header Zafiyeti Nasıl Kapatılır?

(-) Birebir denenmemiştir.

Bu zafiyeti kapatmak için birinci adım

Content-Type:

başlığını sunucu bir kaynak (.html gibi) dönerken yanıt paketinde ekli şekilde servis etmelidir. İkinci adım ise

X-Content-Type-Options:

başlığını sunucu aynı kaynağı dönerken yanıt pakette ekli şekilde servis edilmelidir. İkinc adım eski IE ve eski Chrome tarayıcılarda gelen yanıt paketinin body’sinde farklı türden content olduğunda content sniff’leme ile farklı türden content’e göre render yapması ve örneğin xss gibi kodların çalıştırılmasını önlemek içindir. Bu başlık ile bu türden tarayıcılara content’i sniff’leyerek tür belirleme, Content-Type ile belirtilen türe göre content’i render’la denmiş olur ve farklı türden content enjekte edilen durumlarda bu content’ler çalışmaz yapılır.

Http response'lara X-Content-Type header'ını eklemek için nginx, apache ve IIS konfigurasyon dosyalarına aşağıdaki satırlar eklenmelidir.

Not: Apache X-Content-Type-Options header'ını uygulamalı olarak eklemek için bkz. Ubuntu Masaüstü / Paketleme için Gözden Geçirilecekler / Sıkılaştırmalar / Apache'de Http Güvenlik Başlıklarını Ekleme.docx

```
# Apache için
Header always set X-Content-Type-Options "nosniff"


# Nginx için
add_header X-Content-Type-Options nosniff;


# IIS için
  <httpProtocol>
    <customHeaders>
      <add name="X-Content-Type-Options" value="nosniff" />
    </customHeaders>
  </httpProtocol>
```

Böylece http response'larda X-Content-Type-Options header'ı gelecektir ve tarayıcı gelen dosyaları sunucunun belirttiği uzantıyla çalıştıracaktır. Diğer bir ifadeyle böylece tarayıcı gelen dosyaları http response'daki content-type'ın belirttiği uzantıyla çalıştıracaktır.

Yararlanılan Kaynaklar

https://www.mehmetince.net/http-security-headerlari-neden-ve-nasil-kullanilmalidir/

https://stackoverflow.com/questions/23714383/what-are-all-the-possible-values-for-http-content-type-header

https://www.keycdn.com/blog/http-security-headers/

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Type

https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Content-Type-Options

https://netsparker.com/web-vulnerability-scanner/vulnerabilities/missing-content-type-header/