1.1.1 Güvensiz X-XSS-Protection Başlığı Kullanılması (Security Misconfiguration) (CWE-16)

Açıklık Önem Derecesi: Düşük

Açıklığın Etkisi: Hassas bilgilere yetkisiz erişim, Uzaktan kod çalıştırma, Javascript Kod Pasifleştirme

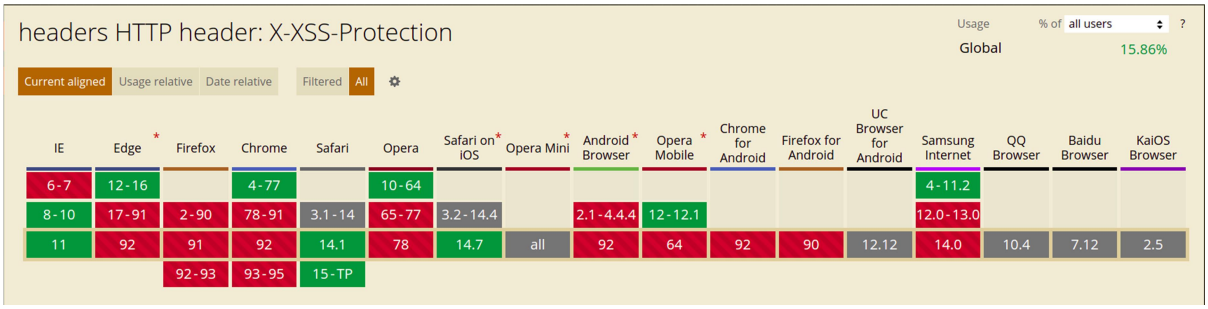
Açıklığın Barındıran Dosyalar/Satırlar:

Proje Dosyası/Dosya Adı	Satır Numarası

Açıklığın Açıklaması:

X-XSS-Protection yanıt başlığı eski web tarayıcılarda tarayıcının XSS Denetleyiciyi (XSS Audior) mekanizmasını aktifleştiren ve XSS saldırılarını önleyen bir http güvenlik başlığıdır. Bu http güvenlik başlığı web tarayıcılarının görüntülediği web uygulamalarda xss zararlısı gelirse bu xss zararlısının web tarayıcıda çalışmasını önler ve son kullanıcının güven içinde web uygulamada gezinmesini sağlar. Günümüzde daha kapsamlı olan Content-Security-Policy'ye yerini bırakmıştır. Fakat eski işletim sistemleri kullanan ve dolayısıyla eski web tarayıcılar kullanan kullanıcıları web uygulamalarda XSS saldırılarından korumak için bu http güvenlik başlığı kullanılmaktadır.

Bu başlık birçok web tarayıcının eski sürümlerinde tanımlıdır. Ancak örneğin Firefox web tarayıcıların hiçbir sürümünde tanımlı bir başlık olmamıştır.



Bu başlık ile eski web tarayıcılarda XSS önlenmektedir. Ancak XSS saldırılarından sadece Reflected XSS saldırıları önlenmektedir. Bu başlık ile örneğin Stored XSS saldırısı önlenememektedir.

Bir Düzeltme

Bu zamana kadar bir web uygulamaya eski web tarayıcıdan erişen kullanıcıları ve web uygulamanın kendisini Reflected XSS saldırılarından korumak için "bir http güvenlik başlığı olan X-XSS-Protection başlığı yanıt paketlerine eklenmelidir ve değeri 1; mode=block şeklinde doldurulmalıdır' denmekteydi. Fakat artık bir düzeltmeye gidilmesi gerekmektedir. Zaman içerisinde bu http güvenlik başlığının web uygulamaya önlem olarak eklenmesi sonrası ayrı bir açıklık doğduğu keşfedilmiştir. Yani güvenlik önlemi uygulandığında web uygulamaya ilave bir açıklık eklenmektedir. X-XSS-Protection http güvenlik başlığı kullanıldığında doğan / ortaya çıkan açıklığı anlamak için şöyle bir örnek verilebilir;

```
<script>guvenlikKontrolCagir()</script>
```

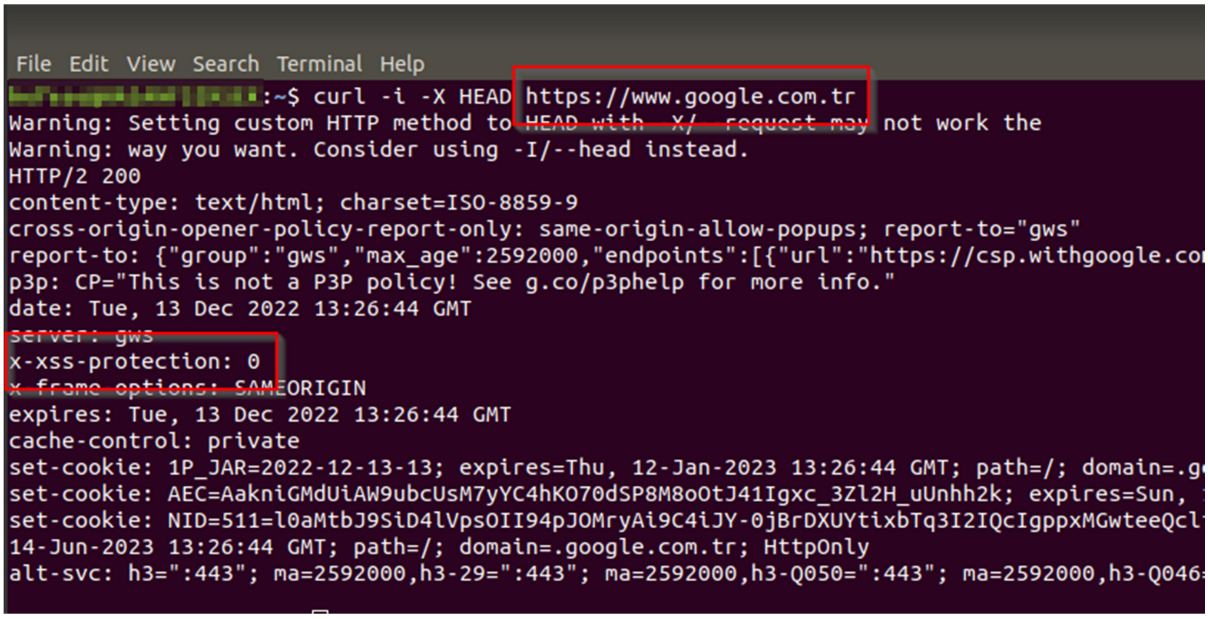
Şekil 2'de gösterilen javascript kod bloğu Frame Busting işlemi yapıyor olsun. Frame Busting web sayfanın bir iframe içerisinde yüklenip yüklenmediğini kontrol eden ve web sayfanın render'lanmasını buna göre belirleyen / önleyen bir javascript kod parçasıdır. Şekil 2'deki javascript kodu web uygulamanın geliştiricisi tarafından web uygulamaya konulmuş javascript kodudur. Güvenlik kontrolü işlemi uygulamaktadır. Saldırgan web uygulamanın güvenlik kontrolü uygulayan bu javascript fonksiyonunun çalışmaması / pasifleşmesi için şöyle bir özel URL hazırlayabilir ve kurbanlara erişmesi için bu url'i paylaşabilir.

```
Özel Hazırlanmış URL:

https://www.webuygulama.gov.tr/webSayfa?herhangiBirParametre=<script>guvenlikKontrolCagir()</script>
```

Dikkat edilirse özel hazırlanmış url'de herhangi bir parametreye web uygulamanın "kendine ait" bir javascript kodu girilmiştir. Bu URL'e gidildiğinde web tarayıcıdaki XSS denetleyici (XSS Auditor) önce parametre üzerinden gönderilen javascript kodunu görecektir, sonra karşılığında gelen http yanıt paketinde aynı javascript kodunu (yani bu sefer web uygulamanın kendine ait olan aynı javascipt kodunu) görecektir. Bunun üzerine aynı javascript kodları gidip geldiğinden Reflected XSS açıklığı vardır diyecektir ve XSS Denetleyici (XSS Auditor) web uygulamanın kendine ait javascript kodunu zararlı zannedecektir. Bunun sonucunda web tarayıcıdaki XSS Denetleyici (XSS Auditor) web uygulamanın kendine ait güvenlik fonksiyonu javascript kodunun çalışmasını engelleyecektir. Böylece web geliştiricisinin uygulamak istediği javascript güvenlik kontrolü uygulanamamış olacaktır. Yani kurbanın web tarayıcısında web uygulamaya ait bir javascript kodu pasifize edilmiş halde kalacaktır. X-XSS-Protection güvenlik önleminin aktif olarak kullanılması bu şekilde bir açıklık doğurmaktadır. Yani saldırgan web uygulamadaki istediği bir javascript kodunu pasifleştirerek kurbanlara web sayfaları ziyaret ettirebilir. Bu açıklığa javascript kodunu pasifleştirme açıklığı denilebilir.

Web tarayıcılardaki XSS Denetleyicileri (XSS Auditor'ları) aktifleştirme yapan X-XSS-Protection güvenlik başlığı o halde kullanılmamalıdır mı denecek olursa cevap hayır olacaktır. Güvenlik başlığının halen kullanılması gerekmektedir. Çünkü eski web tarayıcılarda XSS Denetleyici (XSS Auditor) mekanizmaları varsayılan olarak açık gelmektedir. Yani X-XSS-Protection var ve web tarayıcıdaki XSS önlem mekanizmasını aktifleştiriyormuş gibi bu mekanizmalar açık gelmektedir. Bu ise yine aynı açıklığa götürmektedir. Dolayısıyla saldırgan web uygulamadaki istediği javascript bloğunun çalışmasını kurban ekranında pasifleştirebilecektir. Bu nedenle eski web tarayıcılardan web uygulamayı kullanan kullanıcıları ve web uygulamayı yeni keşfedilen javascript kodunu pasifleştirme açıklığından korumak için X-XSS-Protection http güvenlik başlığı kullanılmalıdır, fakat değeri 0 şeklinde bırakılarak kullanılmalıdır. Bu sayede eski web tarayıcılarda XSS denetleyici (XSS Auditor) mekanizmaları açıksa kapatılsın denmektedir. Bunun neticesinde web uygulamaya eski web tarayıcılardan erişen kullanıcılar için yeni keşfedilen javascript kod pasifleştirme açıklığı kapatılmış olacaktır ve eski web tarayıcılardan kullanıcılar web uygulamayı güvenle kullanabilecektir. Örneğin Google web uygulaması eski web tarayıcıdan kendisini kullanan kullanıcıları için bu şekilde bir uygulamada bulunmaktadır.



Web uygulama güvenliği dünyasında bir otorite olan OWASP kuruluşunun da tavsiyesi bu şekilde kullanılması yönündedir. Kurum web uygulamasında eski web tarayıcı kullanan kullanıcıların ve kurum web uygulamasının kendisinin eski web tarayıcılarda varsayılanda gelen xss filtreleme mekanizması nedeniyle doğan yeni açıklığa karşı korunmadığı tespit edilmiştir.

:::::::::::::: BULGU ::::::::::::::::

Açıklığın Önlemi:

a) IIS Sunucular

IIS sunucularda konfigürasyon dosyası Web.config açılmalıdır ve httpprotocol etiketi içerisindeki customheaders etiketi içerisine Şekil 5'te gösterilen satır eklenmelidir.

```
<httpprotocol>
<customheaders>
<add name="X-XSS-Protection" value="0">
</add>
</customheaders>
</httpprotocol>
```

b) Apache Sunucular

Debian / Ubuntu tabanlı linux işletim sistemlerinde yer alan apache web sunucularda apache2.conf, RedHat / Centos tabanlı linux işletim sistemlerinde yer alan apache web sunucularda httpd.conf dosyası açılmalıdır ve dosya içeriğinin en altına belirtilen satır eklenmelidir.

```
Header set X-XSS-Protection "0"
```

c) Nginx Sunucular

Nginx web sunucularda nginx.conf konfigürasyon dosyası açılmalıdır ve dosya içeriğindeki http { ... } bloğu içerisine belirtilen satır eklenmelidir.

```
add_header X-XSS-Protection "0";
```

Sonuç

En nihayetinde yapılandırma dosyasında yapılan değişiklik sonrası web sunucusu yazılımı yeniden başlatılmalıdır. Böylelikle kullanıcıların gönderdiği http / https taleplerine karşılık web sunucudan dönen http / https yanıtlarında eski tarayıcılardaki xss filtreleme mekanizması kapatılsın direktifi yer alır duruma gelecektir.

Referanslar:

1. http://www.insiderattack.net/2014/04/configuring-secure-iis-response-headers.html
2. https://www.ryadel.com/en/iis-web-config-secure-http-response-headers-pass-securityheaders-io-scan/
3. https://stackoverflow.com/questions/16385541/content-security-policy-invalid-command
4. https://www.keycdn.com/blog/http-security-headers/
5. https://stackoverflow.com/questions/39502968/apache-difference-between-header-always-set-and-header-set
6. https://scotthelme.co.uk/a-new-security-header-referrer-policy/

7. https://blog.qualys.com/securitylabs/2016/01/07/open-redirection-a-simple-vulnerability-threatens-your-web-applications
8. https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-3175
9. https://blog.appcanary.com/2017/http-security-headers.html#x-content-type-options
10. https://www.cyberciti.biz/faq/nginx-send-custom-http-headers/
11. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options
12. https://geekflare.com/tomcat-http-security-header/
13. https://docs.spring.io/spring-security/site/docs/current/reference/html/headers.html
14. https://spring.io/guides/gs/securing-web/
15. https://spring.io/blog/2013/08/23/spring-security-3-2-0-rc1-highlights-security-headers
16. https://www.dailyrazor.com/blog/glassfish-vs-tomcat/
17. http://www.edu4java.com/en/servlet/servlet1.html
18. https://stackoverflow.com/questions/24182367/how-to-add-x-content-type-options-to-tomcat-configuration
19. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Security-Policy/script-src#Unsafe_inline_script
20. https://caniuse.com/mdn-http_headers_x-xss-protection
21. https://jemurai.com/2018/11/28/dont-rely-on-x-xss-protection-to-protect-you-from-xss/
22. https://support.apple.com/en-us/HT204416
23. https://security.stackexchange.com/questions/253924/is-it-better-to-disable-x-xss-protection-header-or-set-the-header-as-x-xss-prote
24. https://crashtest-security.com/x-xss-protection-retired/
25. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-XSS-Protection
26. https://github.com/OWASP/CheatSheetSeries/issues/376
27. https://www.invicti.com/blog/web-security/goodbye-xss-auditor/
28. https://dergipark.org.tr/tr/download/article-file/2160227
29. https://stackoverflow.com/questions/9090577/what-is-the-http-header-x-xss-protection#:~:text=It%20is%20recommended%20to%20have,%2DSecurity%2DPolicy%20header%20instead.
30. https://hackademix.net/2009/11/21/ies-xss-filter-creates-xss-vulnerabilities/
31. https://github.com/github/secure_headers/issues/439