Review Webpage Content for Information Leakage (Bilgi için Web Sayfası İçeriğini İnceleme Sızıntısı)

Summary (Özet)

Programcıların kaynak kodlarına ayrıntılı yorumlar ve meta veriler içermeleri çok yaygındır ve hatta tavsiye edilir. Bununla birlikte, HTML koduna dahil edilen yorumlar ve meta veriler, potansiyel saldırganlar için mevcut olmaması gereken iç bilgileri ortaya çıkarabilir. Herhangi bir bilginin sızdırılıp sızdırılmadığını belirlemek için yorumlar ve meta veri incelemesi yapılmalıdır.

Modern web uygulamaları için, ön uç için istemci-Yollu JavaScript kullanımı daha popüler hale geliyor. Popüler ön uç yapım teknolojileri ReactJS, AngularJS veya Vue gibi istemci tarafı JavaScript kullanır. HTML kodundaki yorumlara ve meta verilere benzer şekilde, birçok programcı ön uçtaki JavaScript değişkenlerinde hassas bilgileri de sertleştirdi. Hassas bilgiler içerebilir (ancak bunlarla sınırlı değildir): Özel API Anahtarları *(örneğin* sınırsız bir Google Harita API Anahtarı), dahili IP adresleri, hassas rotalar (*örneğin* gizli yönetici sayfalarına veya işlevselliğe giden yol) ve hatta kimlik
bilgileri. Bu hassas bilgiler bu tür ön uç JavaScript kodundan sızdırılabilir. Saldırganlar tarafından istismar için sızdırılabilecek herhangi bir hassas bilginin sızdırılıp kullanılmadığını belirlemek için
bir inceleme yapılmalıdır.

Büyük web uygulamaları için, performans sorunları programcılar için büyük bir endişe kaynağıdır. Programcılar, Syntaktikally Müthiş Stil Levhalar (SASS), Sassy CSS (SCSS), web paketi vb. Dahil olmak üzere ön uç performansını optimize etmek için farklı yöntemler kullanmışlardır. Bu teknolojileri kullanarak, ön uç kodu bazen anlaşılması zorlaşır ve hata ayıklama zorlaşır ve bu nedenle programcılar genellikle hata ayıklama amacıyla kaynak harita dosyalarını dağıtır. Bir “kaynak harita”, bir varlığın (CSS veya JavaScript) minified / genişletilmiş bir sürümünü

orijinal yazarlı sürüme bağlayan özel bir dosyadır. Programcılar hala üretim ortamına kaynak harita dosyalarını getirip getirmemeyi tartışıyorlar. Bununla birlikte, üretim ortamına yayınlanırsa hata ayıklama için kaynak harita dosyalarının veya dosyalarının kaynaklarını daha insancıl hale getireceği inkar edilemez. Saldırganların ön uçtan itibaren güvenlik açıklarını bulmasını veya ondan hassas bilgi toplamasını kolaylaştırabilir. Herhangi bir hata ayı dosyasının ön uçtan itibaren ortaya çıkarılıp çıkarılmadığını belirlemek için JavaScript kodu incelemesi yapılmalıdır. Projenin bağlamına ve hassasiyetine bağlı olarak, dosyaların üretim ortamında var olup olmadığına bir güvenlik uzmanı karar vermelidir.

Test Objectives (Test Hedefleri)

- Herhangi bir bilgi sızıntısını bulmak için web sayfası yorumlarını ve meta verileri gözden geçirin.
- JavaScript dosyalarını toplayın ve uygulamayı daha iyi anlamak ve herhangi bir bilgi sızıntısını bulmak için JS kodunu inceleyin.
- Kaynak harita dosyaları veya diğer ön uç hata ayı dosyaları olup olmadığını belirleyin.

How to Test (Nasıl Test Edilir)

Review webpage comments and metadata (Web sayfası yorumlarını ve meta verileri gözden geçirin)

HTML yorumları genellikle geliştiriciler tarafından uygulama hakkında bilgi hata ayıklamayı içerecek şekilde kullanılır. Bazen yorumları unuturlar ve üretim ortamlarında bırakırlar. Testçiler, başlayan HTML yorumlarını aramalıdır `<!--`. .

Saldırganın uygulama hakkında daha fazla bilgi edinmesine yardımcı olabilecek hassas bilgileri içeren yorumlar için HTML kaynak kodunu kontrol edin. SQL kodu, kullanıcı adları ve şifreler, dahili IP adresleri veya hata ayıklama bilgileri olabilir.

```
...
<div class="table2">
  <div class="col1">1</div><div class="col2">Mary</div>
  <div class="col1">2</div><div class="col2">Peter</div>
  <div class="col1">3</div><div class="col2">Joe</div>
```

```
<!-- Query: SELECT id, name FROM app.users WHERE active='1' -->

</div>
...
```

Testçi böyle bir şey bile bulabilir:

```
<!-- Use the DB administrator password for testing:  f@keP@a$$w0rD -->
```

Geçerli sürüm numaraları ve Veri Türü Tanım (DTD) URL'ler için HTML sürüm bilgilerini kontrol edin

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
```

- `strict.dtd` – varsayılan katı DTD
- `loose.dtd` – gevşek DTD
- `frameset.dtd` – Çerçeve seti belgeleri için DTD

Bazıları `META` Etiketler aktif saldırı vektörleri sağlamaz, bunun yerine bir saldırganın bir uygulamanın profilini çıkarmasına izin verir:

```
<META name="Author" content="Andrew Muller">
```

Yaygın (ama WCAG uyumlu değil) `META` RefreshEtiketi yenileniyor.

```
<META http-equiv="Refresh" content="15;URL=https://www.owasp.org/index.html">
```

Ortak bir kullanım için `META` Etiket, bir arama motorunun arama sonuçlarının kalitesini artırmak için kullanabileceği anahtar kelimeleri belirtmektir.

```
<META name="keywords" lang="en-us" content="OWASP, security, sunshine, lollipops">
```

Her ne kadar çoğu web sunucusu arama motoru indekslemeyi yönetse de `robots.txt` dosya, aynı zamanda tarafından da yönetilebilir `META` Etiketler. Aşağıdaki etiket, robotlara etiket içeren HTML sayfasındaki bağlantıları dizine eklememelerini ve takip etmemelerini tavsiye edecektir.

```
<META name="robots" content="none">
```

İnternet İçerik Seçimi Platformu (PICS) ve Web Açıklaması için Protokol Kaynakları (POWDER), Meta verileri İnternet içeriği ile ilişkilendirmek için altyapı sağlar.

Identifying JavaScript Code and Gathering JavaScript Files (JavaScript Kodunu Tanımlamak ve JavaScript Dosyaları Toplama)

Programcılar genellikle ön uçta JavaScript değişkenleri ile hassasbilgileri sertleştirir. Testçiler HTML kaynak kodunu kontrol etmeli ve aralarında JavaScript kodunu aramalıdır `<script>` ve `</script>` Etiketler. Testçiler ayrıca kodu incelemek için harici JavaScript dosyalarını tanımlamalıdır (JavaScript dosyaları dosya uzantısına sahiptir `.js` ve JavaScript dosyasının adı genellikle içine konur `src` (kaynak) bir özniteliği `<script>` etiket).

Sistemi daha fazla kötüye kullanmak veya manipüle etmek için saldırganlar tarafından kullanılabilecek hassas bilgi sızıntıları için JavaScript kodunu kontrol edin. Örneğin aşağıdaki değerleri arayın: API tuşları, dahili IP adresleri, hassas rotalar veya kimlik bilgileri. Örneğin:

```
const myS3Credentials = {
  accessKeyId: config('AWSS3AccessKeyID'),
  secretAcccessKey: config('AWSS3SecretAccessKey'),
};
```

Testçi böyle bir şey bile bulabilir:

`var conString = "tcp://postgres:1234@localhost/postgres";`

Bir API Anahtarı bulunduğunda, testçiler API Anahtar kısıtlamalarının hizmet başına veya IP, HTTP yönlendirmesi, uygulama, SDK vb. ile seçilip belirlenmediğini kontrol edebilir.

Örneğin, testçiler bir Google Harita API Anahtarı bulduysa, bu API Anahtarının IP ile kısıtlanıp kısıtlanmadığını veya yalnızca Google Harita API'leri başına kısıtlanıp kısıtlanmadığını kontrol edebilirler. Google API Anahtarı yalnızca Google Harita API'leri uyarınca kısıtlanırsa, saldırganlar sınırsız Google Harita API'lerini sorgulamak için bu API Anahtarı kullanabilir ve uygulama sahibinin bunun için ödeme yapması gerekir.

```
<script type="application/json">
...
```

```
{"GOOGLE_MAP_API_KEY":"AIzaSyDUEBnKgwiqMNpDplT6ozE4Z0XxuAbqDi4",
 "RECAPTCHA_KEY":"6LcPscEUiAAAAHOwwM3fGvIx9rsPYUq62uRhGjJ0"}
...
</script>
```

Bazı durumlarda, testçiler JavaScript kodundan dahili veya gizli yönetici sayfalarına bağlantılar gibi hassas rotalar bulabilirler.

```
<script type="application/json">
...
"runtimeConfig":{"BASE_URL_VOUCHER_API":"https://staging-voucher.victim.ne
,"BASE_BACKOFFICE_API":"https://10.10.10.2/api", "ADMIN_PAGE":"/hidden_adm
inistrator"}
...
</script>
```

Identifying Source Map Files (Kaynak Harita Dosyalarını Belirlemek)

DevTools açıldığında kaynak harita dosyaları genellikle yüklenir. Testçiler ayrıca her harici JavaScript dosyasının uzantısından sonra ".map" uzantısını ekleyerek kaynak harita dosyalarını da bulabilirler. Örneğin, bir test cihazı bir görürse `/static/js/main.chunk.js` dosya, daha sonra kaynak harita dosyasını ziyaret ederek kontrol edebilirler `/static/js/main.chunk.js.map` . .

Black-Box Testing (Siyah-Box Testi)

Saldırganın uygulama hakkında daha fazla bilgi edinmesine yardımcı olabilecek herhangi bir hassas bilgi için kaynak harita dosyalarını kontrol edin. Örneğin:

```
{
  "version": 3,
  "file": "static/js/main.chunk.js",
  "sources": [
    "/home/sysadmin/cashsystem/src/actions/index.js",
    "/home/sysadmin/cashsystem/src/actions/reportAction.js",
    "/home/sysadmin/cashsystem/src/actions/cashoutAction.js",
```

```
    "/home/sysadmin/cashsystem/src/actions/userAction.js",
    "..."
  ],
  "..."
}
```

Web siteleri kaynak harita dosyalarını yüklediğinde, ön uç kaynak kodu okunabilir ve daha kolay açılır.

Tools (Araçlar)

- Kategori:
- Tarayıcı "gücünü görüntüle" fonksiyonu
- Gözbebekleri
- Kör
- Burp Suite'in
- Waybackurls'ın
- Google Haritalar API Tarayıcı

References (Referanslar)

- Anahtar Hackler

Whitepapers (Beyaz kağıtlar)

- HTML sürümü 4.01
- XHTML'NIN
- HTML sürümü 5