Testing for Credentials Transported over an Encrypted Channel (Şifreli bir Kanal Üzerinden Taşınan Kimlik Bilgileri Testi)

Summary (Özet)

Kimlik bilgileri taşıma için test, web uygulamalarının transit kimlik doğrulama verilerini şifrelediğini doğrular. Bu şifreleme, saldırganların ağ trafiğini koklayarak hesapları ele geçirmesini önler. Web uygulamaları, istemcinin sunucuya ve sunucuya istemci iletişimlerine istemci iletişimleri için transit bilgileri şifrelemek için HTTPS'yi kullanır. Bir istemci aşağıdaki etkileşimler sırasında kimlik doğrulama verilerini gönderebilir veya alabilir:

- Bir müşteri giriş talebinde bulunmak için bir kimlik bilgisi gönderir
- Sunucu, bir oturum belirteci ile başarılı bir oturuma yanıt verir
- Doğrulanmış bir istemci, web sitesinden hassas bilgileri talep etmek için bir oturum belirteci gönderir
- Bir müşteri şifrelerini unuttukları takdirde web sitesine bir jeton gönderir

Bu kimlik bilgilerinin transit olarak şifrelenmemesi, ağ koklama araçlarına sahip saldırganların kimlik bilgilerini görüntülemelerine ve muhtemelen bir kullanıcının hesabını çalmak için kullanmalarına izin verebilir. Saldırgan doğrudan Wireshark veya benzeri araçları kullanarak trafiği koklayabilir veya HTTP isteklerini yakalamak için bir vekil kurabilirler. Hassas veriler bunu önlemek için transit olarak şifrelenmelidir.

Trafiğin şifrelenmesi, mutlaka tamamen güvenli olduğu anlamına gelmez. Güvenlik ayrıca kullanılan şifreleme algoritmasına ve uygulamanın kullandığı anahtarların sağlamlığına da bağlıdır. Şifreleme

algoritmasını doğrulamak için Zayıf Taşımacı Katmanı Güvenliği için Testlere bakın yeterlidir.

Test Objectives (Test Hedefleri)

- Web sitesinin veya uygulamasının herhangi bir kullanım durumu,sunucunun veya istemcinin şifreleme olmadan kimlik bilgilerini paylaşmasına neden olup olmadığını değerlendirir.

How to Test (Nasıl Test Edilir)

Kimlik bilgileri taşımayı test etmek için, kimlik bilgilerine ihtiyaç duyan bir istemci ve web uygulama sunucusu arasındaki trafiği yakalayın. Başvuruyu geçerli bir oturumla kullanırken giriş sırasında ve uygulamayı kullanırken aktarılan kimlik bilgilerini kontrol edin. Test için ayarlama yapmak için:

1. Aşağıdakilerden biri gibi trafiği yakalamak için bir araç kurun ve başlatın:
    - Web tarayıcısının geliştirici araçları
    - OWASP ZAP dahil olmak üzere bir proxy
2. Web tarayıcısını HTTPS'yi tercih eden herhangi bir özelliği veya eklentiyi devre dışı bırakın. HTTP Everywhere gibi bazı tarayıcılar veya uzantılar, HTTP isteklerini HTTPS'ye yönlendirerek zorunlu tarama ile mücadele edecektir.

Yakalanan trafikte, aşağıdakiler de dahil olmak üzere hassas verileri arayın:

- Passhors veya şifreler, genellikle bir mesaj gövdesinin içinde
- Tokenler, genellikle kurabiyelerin içinde
- Hesap veya şifre sıfırlama kodları

Bu hassas verileri içeren herhangi bir mesaj için, değişimin HTTPS kullanılarak gerçekleştiğini doğrulayın (ve HTTP) kullanılarak gerçekleşmiştir. Aşağıdaki örnekler, web uygulamasının bir sunucuda olduğu, geçen veya başarısız testleri gösteren yakalanan verileri gösterir. `www.example.org` . .

Login (Giriş Yap)

Giriş sayfasının adresini bulun ve protokolü HTTP'ye geçirmeye çalışın. Örneğin, zorunlu tarama için URL aşağıdaki gibi görünebilir: `http://www.example.org/login` . .

Giriş sayfası normalde HTTPS ise, giriş sayfasının HTTP olarak yüklenip yüklemediğini görmek için “S” uzulamaya çalışın.

Şifrelenmemiş HTTP kullanımını zorlamaya çalışırken geçerli bir hesap kullanarak giriş yapın. Geçen bir testte, giriş isteği HTTPS olmalıdır:

```
Request URL: https://www.example.org/j_acegi_security_check
Request method: POST
...
Response headers:
HTTP/1.1 302 Found
Server: nginx/1.19.2
Date: Tue, 29 Sep 2020 00:59:04 GMT
Transfer-Encoding: chunked
Connection: keep-alive
X-Content-Type-Options: nosniff
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Set-Cookie: JSESSIONID.a7731d09=node01ai3by8hip0g71kh3ced41pmqf4.no
de0; Path=/; Secure; HttpOnly
ACEGI_SECURITY_HASHED_REMEMBER_ME_COOKIE=dXNlcmFiYzoxNjAyNT
UwNzQ0NDU3OjFmNDlmYTZhOGI1YTZkYTYxNDIwYWVmNmM0OTI1OGFhO
DA3Y2ZmMjg4MDM3YjcwODdmN2I2NjMwOWIyMDU3NTc=; Path=/; Expires
=Tue, 13-Oct-2020 00:59:04 GMT; Max-Age=1209600; Secure; HttpOnly
Location: https://www.example.org/
...
POST data:
j_username=userabc
j_password=My-Protected-Password-452
from=/
Submit=Sign in
```

- Girişte, HTTPS istek URL'si nedeniyle kimlik bilgileri şifrelenir
- Sunucu bir oturum belirtec için çerez bilgilerini iade ederse, çerez ayrıca dahil olmalıdır. `Secure` Müşterinin çerezi daha sonra şifrelenmemiş kanallar üzerinde ifşa etmesini önlemek için atfeder. Ara bakalım `Secure` Cevap başlığında anahtar kelime.

Test, herhangi bir girişin HTTP üzerinden bir kimlik bilgisi aktarırsa, aşağıdakilere benzer şekilde başarısız olur:

```
Request URL: http://www.example.org/j_acegi_security_check
Request method: POST
...
POST data:
j_username=userabc
j_password=My-Protected-Password-452
from=/
Submit=Sign in
```

Bu başarısız test örneğinde:

- Çekme URL'si `http://` ve sade metni ortaya çıkarır `j_username` ve `j_password` Posta verileri aracılığıyla.
- Bu durumda, test zaten tüm kimlik bilgilerini ortaya çıkaran POST verilerini gösterdiğinden, yanıt başlıklarını kontrol eden nokta yoktur (bu da muhtemelen bir oturum tokenini veya çerezi ortaya çıkarır).

Account Creation (Hesap Oluşturma)

Şifrelenmemiş hesap oluşturmayı test etmek için, hesap oluşturmanın HTTP sürümüne göz atmaya ve örneğin bir hesap oluşturmaya çalışın: `http://www.example.org/securityRealm/createAccount`

Test, zorunlu taramadan sonra bile geçer, istemci hala yeni hesap isteğini HTTPS aracılığıyla gönderir:

```
Request URL: https://www.example.org/securityRealm/createAccount
Request method: POST
...
Response headers:
HTTP/1.1 200 OK
Server: nginx/1.19.2
Date: Tue, 29 Sep 2020 01:11:50 GMT
Content-Type: text/html;charset=utf-8
Content-Length: 3139
```

```
Connection: keep-alive
X-Content-Type-Options: nosniff
Set-Cookie: JSESSIONID.a7731d09=node011yew1ltrsh1×1k3m6g6b44tip8.node0;
Expires: 0
Cache-Control: no-cache,no-store,must-revalidate
X-Hudson-Theme: default
Referrer-Policy: same-origin
Cross-Origin-Opener-Policy: same-origin
X-Hudson: 1.395
X-Jenkins: 2.257
X-Jenkins-Session: 4551da08
X-Hudson-CLI-Port: 50000
X-Jenkins-CLI-Port: 50000
X-Jenkins-CLI2-Port: 50000
X-Frame-Options: sameorigin
Content-Encoding: gzip
X-Instance-Identity: MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA33
...
POST data:
username=user456
fullname=User 456
password1=My-Protected-Password-808
password2=My-Protected-Password-808
Submit=Create account
Jenkins-Crumb=58e6f084fd29ea4fe570c31f1d89436a0578ef4d282c1bbe03ffac
```

- Bir girişe benzer şekilde, çoğu web uygulaması otomatik olarak başarılı bir hesap oluşturmada bir oturum belirteci verir. Eğer bir şey varsa `Set-Cookie:`, bir tane olduğunu doğrulayın `Secure;` Bir de öznitelik.

Test, istemci şifrelenmemiş HTTP ile yeni bir hesap talebi gönderirse başarısız olur:

```
Request URL: http://www.example.org/securityRealm/createAccount
Request method: POST
...
```

```
POST data:
username=user456
fullname=User 456
password1=My-Protected-Password-808
password2=My-Protected-Password-808
Submit=Create account
Jenkins-Crumb=8c96276321420cdbe032c6de141ef556cab03d91b25ba60be
8fd3d034549cdd3
```

- Bu Jenkins kullanıcı oluşturma formu, POST verilerindeki tüm yeni kullanıcı ayrıntılarını (isim, tam ad ve şifre) HTTP oluşturma hesap sayfasına maruz bıraktı.

Password Reset, Change Password or Other Account Manipulation (Şifre Sıfırlama, Şifreyi Değiştirin veya Diğer Hesap Manipülasyonu)

Giriş ve hesap oluşturma gibi, web uygulaması bir kullanıcının bir hesabı değiştirmesine veya kimlik bilgileriyle farklı bir hizmeti aramasına izin veren özelliklere sahipse, tüm bu etkileşimlerin HTTPS olduğunu doğrulayın. Test etmek için etkileşimler aşağıdakileri içerir:

- Kullanıcıların unutulmuş bir şifreyi veya diğer kimlik bilgilerini kullanmalarını sağlayan formlar
- Kullanıcıların kimlik bilgilerini düzenlemesine izin veren formlar
- Kullanıcının başka bir sağlayıcıyla kimlik doğrulamasını gerektiren formlar (örneğin, ödeme işleme)

Accessing Resources While Logged In (Girişte Bulunurken Kaynaklara Erişim)

Giriş yaptıktan sonra, erişim için mutlaka bir giriş gerektirmeyen kamuya açık özellikler de dahil olmak üzere uygulamanın tüm özelliklerine erişin. Müşterinin kimlik bilgilerini sızdırıp sızdırmadığını görmek için web sitesinin HTTP sürümüne göz atmaya zorlandı.

Test, tüm etkileşimler oturum tokenini HTTPS üzerinden aşağıdaki örneğe benzer şekilde gönderirse geçer:

```
Request URL:http://www.example.org/
Request method:GET
...
Request headers:
GET / HTTP/1.1
Host: www.example.org
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:78.0) Gecko/20100101 Firefox/78.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
DNT: 1
Connection: keep-alive
Cookie: JSESSIONID.a7731d09=node01ai3by8hip0g71kh3ced41pmqf4.node0; AC
Upgrade-Insecure-Requests: 1
```

- Çerezdeki oturum token şifrelenir, çünkü istek URL HTTPS'dir

Tarayıcı, web sitesinin herhangi bir bölümünde HTTP üzerinden bir oturum tokeni gönderirse, bu davayı tetiklemek için zorunlu tarama gerekli olsa bile:

```
Request URL:http://www.example.org/
Request method:GET
...
Request headers:
GET / HTTP/1.1
Host: www.example.org
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:68.0) Gecko/20100101 Firefox/68.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Cookie: language=en; welcomebanner_status=dismiss; cookieconsent_status=di
Upgrade-Insecure-Requests: 1
```

- GET talebi oturum tokenini ortaya çıkardı `JSESSIONID` (tarayıcıdan sunucuya) istek URL'si `http://www.example.org/`

Remediation (Düzeltme)

Tüm web sitesi için HTTPS kullanın. HSTS'yi uygulayın ve herhangi bir HTTP'yi HTTP'ye yönlendirin. Site, tüm özellikleri için HTTPS'yi kullanmaktan aşağıdaki avantajları kazanır:

- Saldırganların web sunucusuyla etkileşimleri değiştirmesini önler (acelevir kötü amaçlı yazılımları tehlikeye atılmış bir yönlendirici aracılığıyla yerleştirmek de dahil).
- Güvensiz site uyarılarına müşteri kaybetmekten kaçınır. Yeni tarayıcılar HTTP tabanlı web sitelerini güvensiz olarak işaretler.
- Bazı uygulamaların yazılmasını kolaylaştırır. Örneğin, Android API'lerin HTTP aracılığıyla herhangi bir şeye bağlanmak için geçersiz kılmalara ihtiyacı vardır.

HTTPS'ye geçmek hantalsa, önce hassas operasyonlar için HTTPS'ye öncelik verin. Orta vadede, müşterileri tehlikeye atmak veya HTTP'nin güvensiz olduğu uyarılarını kaçırmamak için tüm uygulamayı HTTPS'ye dönüştürmeyi planlayın. Kuruluş zaten HTTPS için sertifika satın
almıyorsa, sunucudaki Let's Encrypt veya diğer ücretsiz sertifika yetkililerine bakın.