**THTML Injection** ve **CSRF (Cross-Site Request Forgery)**, web uygulamalarına karşı düzenlenen **istemci taraflı güvenlik açıklarıdır**. Sızma testlerinde hem kullanıcı hem de sistem güvenliğini tehlikeye atabilecek bu zafiyetler, test süreçlerinde dikkatle incelenmelidir.

Bu döküman, THTML Injection ve CSRF açıklarını ayrı ayrı tanımlar, nasıl tespit ve istismar edileceğini açıklar ve örnek kullanım senaryoları ile destekler.

**THTML Injection Nedir?**

**THTML Injection**, “Templated HTML” yani HTML şablon motorlarına özgü bir güvenlik açığıdır. Geliştiricilerin kullandığı {{ user\_input }} benzeri şablon yapılar, kullanıcıdan alınan verilerin filtrelenmeden doğrudan HTML çıktısına eklenmesi sonucu **kod enjeksiyonuna** neden olabilir.

**Bu Açığın Özellikleri Nelerdir?**

* **Şablon Motoru Temellidir**: Genellikle Django, Jinja2, Handlebars gibi şablon motorlarında görülür.
* **Kötü Amaçlı HTML veya JavaScript Enjeksiyonu** yapılmasına izin verebilir.
* **XSS’e yol açabilir**: Doğrudan tarayıcıda kod çalıştırılabilir.
* **Veri Sızdırma ve Oturum Ele Geçirme** gibi ciddi etkileri olabilir.

**Örnek THTML Injection Senaryosu**

Bir web uygulaması şu HTML’i render ediyor:

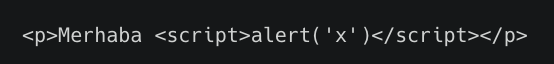


Kullanıcı şu girdiyi girerse:

metin, yazı tipi, grafik, siyah içeren bir resim

Açıklama otomatik olarak oluşturuldu

Çıktı şu hale gelir:



Bu durumda tarayıcıda JavaScript çalışır; bu da **Stored XSS** saldırısına eşdeğerdir.

**Tespit Adımları**

**Adım 1: Kullanıcı girdisinin şablona doğrudan yansıyıp yansımadığını kontrol et**

“{{7\*7}} → 49 çıktısı alınıyorsa şablon kodu çalıştırılabiliyor demektir.”

**Adım 2: JavaScript veya HTML etiketi enjeksiyonu deneyin**

yazı tipi, grafik, metin, tipografi içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Adım 3: Filtre bypass’ları deneyin**



**Adım 4: Payload’ları tarayıcıda gözlemleyin**

* Burp Suite kullanarak yanıtı inceleyin.
* Tarayıcıda “Inspect > Network > Response” kontrolü yapın.

**CSRF**, kimliği doğrulanmış bir kullanıcıdan, onun bilgisi dışında kötü niyetli istek gönderilmesini sağlayan saldırı türüdür. Genellikle bir kullanıcı oturum açmışken, saldırganın hazırladığı sahte bir form aracılığıyla hedef sisteme istek gönderilir.

**Bu Açığın Özellikleri Nelerdir?**

* **Oturum açıkken çalışır**: Kullanıcının oturumu aktif olmalı.
* **Hedef sistemde işlem yapılır**: Kullanıcıdan POST, GET veya DELETE isteği gönderilir.
* **Hedef: Admin Paneli / Profil Ayarları / E-posta Değişimi / Parola Sıfırlama**

**Örnek CSRF Saldırısı**

**Hedef URL:**

metin, yazı tipi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

**Saldırganın hazırladığı HTML dosyası:**

metin, yazı tipi, ekran görüntüsü içeren bir resim

Açıklama otomatik olarak oluşturuldu

Kurban bu sayfayı ziyaret ettiğinde, e-postası saldırganın belirlediği adres ile değiştirilir.

**CSRF Tespiti Adımları**

**Adım 1: Hedef formun CSRF token kullanıp kullanmadığını kontrol et**

* HTML kaynak kodda <input type="hidden" name="csrf\_token" ...> varsa kontrol vardır.
* Yoksa saldırı mümkün olabilir.

**Adım 2: İsteği Burp Suite ile yakalayın**

* Request detayına bakın, oturum çerezi (cookie) varsa, saldırı başarabilir.

**Adım 3: Aynı isteği HTML form içine gömüp test edin**

* Basit form ile aynı POST isteğini yapın.
* Sonuç alınıyorsa CSRF açığı vardır.

metin, yazı tipi, ekran görüntüsü, tasarım içeren bir resim

Açıklama otomatik olarak oluşturuldu

HTML form ile bu istek tekrarlandığında başarılı oluyorsa → **CSRF var**