Testing Web Messaging (Web Mesajlaşmasının Test Edilmesi)

Summary (Özet)

Web Mesajlaşma (Çarpma Belge Mesajlaşması olarak da bilinir), farklı alanlarda çalışan uygulamaların güvenli bir şekilde iletişim kurmasını sağlar. Web mesajlaşmasının yürürlüğe girmesinden önce, farklı kökenlerin (iframlar, sekmeler ve pencereler arasında) iletişimi aynı köken politikası ile kısıtlandı ve tarayıcı tarafından uygulandı. Geliştiriciler bu görevleri yerine getirmek için birden fazla hack kullandılar ve çoğu çoğunlukla güvensizdi.

Tarayıcıdaki bu kısıtlama, kötü amaçlı bir web sitesinin diğer ifadelerden, sekmelerden vb. gizli verileri okumasını önlemek için geçerlidir; Bununla birlikte, iki güvenilir web sitesinin birbirleriyle
veri alışverişinde bulunması gereken bazı meşru durumlar vardır. Bu ihtiyacı karşılamak için, Çapraz Belge Mesajlaşma WHATWG HTML5 taslak spesifikasyonunda tanıtıldı ve tüm büyük tarayıcılarda uygulandı. Ifamlar, sekmeler ve pencereler arasında birden fazla köken arasında güvenli iletişim sağlar.

Mesajlaşma API'si tanıttı `postMessage()` Yöntem, hangi düz metin mesajlarının çapraz sicil gönderilebileceği. İki parametreden oluşur: mesaj ve alan adı.

Kullanırken bazı güvenlik endişeleri vardır `*` Aşağıda tartıştığımız alan olarak. Mesaj almak için, alıcı web sitesinin aşağıdaki özelliklere sahip yeni bir etkinlik işleyicisi eklemesi gerekir:

- Veriler, gelen mesajın içeriği;
- gönderenin kökeni; ve
- Kaynak, kaynak penceresi.

İşte kullanımdaki mesajlaşma API'sinin bir örneği. Bir mesaj göndermek için:

```
iframe1.contentWindow.postMessage("Hello world","http://www.example.com");
```

Bir mesaj almak için:

```
window.addEventListener("message", handler, true);
function handler(event) {
    if(event.origin === 'chat.example.com') {
        /* process message (event.data) */
    } else {
        /* ignore messages from untrusted domains */
    }
}
```

(Köken Güvenliği)

Kökeni bir şemadan, ev sahibinden ve limandan oluşur. Etkinliği, mesajı gönderen veya alan edinen kişiyi benzersiz bir şekilde tanımlar ve URL'nin yolunu veya parça parçasını içermez. Örneğin, `https://example.com` Farklı olarak kabul edilecektir `http://example.com` Çünkü ilkinin entrikası `https`, ikincisi ise `http`. . Bu aynı etki alanında çalışan ancak farklı limanlarda çalışan web sunucuları için de geçerlidir.

Test Objectives (Test Hedefleri)

- Mesajın kaynağının güvenliğini değerlendirin.
- Güvenli yöntemler kullandığını ve girdisini doğruladığını doğrulayın.

How To Test (Nasıl Test Edilir)

(Menşe Güvenliğini İnceleyin)

Testçiler, başvuru kodunun güvenilir etki alanlarından gelen mesajları filtreleyip işlemediğini kontrol etmelidir. Gönderme alanında, alıcı alanın açıkça belirtilmesini ve bunun `*` İkinci argüman olarak kullanılmaz `postMessage()`. . Bu uygulama güvenlik endişelerini ortaya çıkarabilir ve bir yönlendirme durumunda veya kökenin başka yollarla değişmesi durumunda, web sitesinin bilinmeyen ana bilgisayarlara veri göndermesine ve bu nedenle gizli verileri kötü amaçlı sunuculara sızdırmasına yol açabilir.

Web sitesi, bir web sitesine mesaj göndermesine izin verilen alanları veya kökenleri kısıtlamak için güvenlik kontrolleri ekleyemezse, bir güvenlik riski getirmesi muhtemeldir. Testçiler mesaj etkinliği

dinleyicileri için kodu incelemeli ve geri arama işlevini almalı `addEventListener` Daha fazla analiz için yöntem. Etki alanları her zaman veri manipülasyonundan önce doğrulanmalıdır.

(Giriş Doğrulamasını Inceleyin)

Web sitesi teorik olarak yalnızca güvenilir alanlardan gelen mesajları kabul etse de, verilerin hala dışarıdan kaynaklı, güvenilmez veriler olarak ele alınması ve uygun güvenlik kontrolleriyle işlenmesi
gerekir. Testçiler kodu analiz etmeli ve özellikle verilerin değerlendirildiği yerlerde güvensiz yöntemler aramalıdır.
`eval()` veya DOM'a sokulmuş `innerHTML` DOM tabanlı XSS güvenlik açıkları oluşturabilecek mülkiyet.

(Statik Kod Analizi)

Web mesajlaşmanın nasıl uygulandığını belirlemek için JavaScript kodu analiz edilmelidir. Özellikle, testçiler web sitesinin güvenilmez etki alanlarından gelen mesajları nasıl kısıtladığı ve verilerin güvenilir alan adları için bile nasıl ele alındığıyla ilgilenmelidir.

Bu örnekte, owasp.org etki alanında her subdomain (www, sohbet, forumlar, ...) için erişim gereklidir. Kod, herhangi bir etki alanını kabul etmeye çalışıyor `.owasp.org` : :

```
window.addEventListener("message", callback, true);

function callback(e) {
    if(e.origin.indexOf(".owasp.org")!=-1) {
        /* process message (e.data) */
    }
}
```

Amaç, aşağıdakiler gibi domanalara izin vermektir:

- `www.owasp.org`
- `chat.owasp.org`
- `forums.owasp.org`

Ne yazık ki, bu güvenlik açıklarını getiriyor. Bir saldırgan, bir alan gibi bir alandan bu yana filtreyi kolayca atlayabilir `www.owasp.org.attacker.com` - Eşleşecek.

İşte bir köken kontrolü olmayan bir kod örneği. Bu çok güvensizdir, çünkü herhangi bir alandan gelen girdiyi kabul edecektir:

```
window.addEventListener("message", callback, true);

function callback(e) {
      /* process message (e.data) */
}
```

İşte XSS saldırısına yol açabilecek giriş doğrulama güvenlik açıkları ile ilgili bir örnek:

```
window.addEventListener("message", callback, true);

function callback(e) {
      if(e.origin === "trusted.domain.com") {
         element.innerHTML= e.data;
      }
}
```

Mülkü kullanmak için daha güvenli bir yaklaşım olacaktır `innerText` yerine `innerHTML` . .

Web mesajlaşma ile ilgili daha fazla OWASP kaynağı için, bkz. OWASP HTML5 Güvenlik Hile Sayfası