Testing Browser Storage (Tarayıcı Depolamasının Test Edilmesi)

Summary (Özet)

Tarayıcılar, geliştiricilerin verileri depolamaları ve almaları için aşağıdaki istemci tarafı depolama mekanizmalarını sağlar:

- Yerel Depolama
- Oturum Depolama
- EndeksededDB
- Web SQL (Depresif)
- Çerezler

Bu depolama mekanizmaları, tarayıcının Google Chrome DevTools veya Firefox'un Depolama Müfettişi gibi geliştirici araçları kullanılarak görüntülenebilir ve düzenlenebilir.

Not: Önbellek aynı zamanda bir depolama şekli olsa da, kendi özelliklerini ve endişelerini kapsayan ayrı bir bölümde kaplanır.

Test Objectives (Test Hedefleri)

- Web sitesinin hassas verileri istemci tarafı depolamada saklayıp saklamadığını belirleyin.
- Depolama nesnelerinin kod işlemesi, geçersiz olmayan girdi veya savunmasız kütüphaneler kullanmak gibi enjeksiyon saldırılarının olasılıkları için incelenmelidir.

How To Test (Nasıl Test Edilir)

Local Storage (Yerel Depolama)

`window.localStorage` Web Depolama API'sini uygulayan ve tarayıcıda **kalıcı** anahtar değeri depolama alanı sağlayan küresel bir muktedirdir.

Hem anahtarlar hem de değerler sadece teller olabilir, bu nedenle herhangi bir ip olmayan değer, genellikle JSON.stringify aracılığıyla yapılan, bunları saklamadan önce önce dizelerlere dönüştürülmelidir.

Girişler `localStorage` Tarayıcı penceresi kapandığında bile, Özel / Gizli moddaki pencereler hariç.

Maksimum depolama kapasitesi `localStorage` Tarayıcılar arasında değişir.

List All Key-Value Entries (Tüm Anahtar Değer Girişlerini Listeleyin)

```
for (let i = 0; i < localStorage.length; i++) {
  const key = localStorage.key(i);
  const value = localStorage.getItem(key);
  console.log(`${key}: ${value}`);
}
```

Session Storage (Oturum Depolama)

`window.sessionStorage` Web Depolama API'sini uygulayan ve tarayıcıda **geçici** anahtar değer depolama alanı sağlayan küresel bir özelliktir.

Hem anahtarlar hem de değerler sadece teller olabilir, bu nedenle herhangi bir ip olmayan değer, genellikle JSON.stringify aracılığıyla yapılan, bunları saklamadan önce önce dizelerlere dönüştürülmelidir.

Girişler `sessionStorage` Tarayıcı sekmesi / penceresi kapalı olduğunda temizlendikleri için geçicidirler.

Maksimum depolama kapasitesi `sessionStorage` Tarayıcılar arasında değişir.

List All Key-Value Entries (Tüm Anahtar Değer Girişlerini Listeleyin)

```
for (let i = 0; i < sessionStorage.length; i++) {
  const key = sessionStorage.key(i);
  const value = sessionStorage.getItem(key);
  console.log(`${key}: ${value}`);
}
```

IndexedDB (EndeksededDB)

IndexedDB, yapılandırılmış veriler için tasarlanmış işlemsel, nesne yönelimli bir veritabanıdır. Bir IndexedDB veritabanı birden fazla nesne mağazasına sahip olabilir ve her bir nesne mağazası birden fazla nesneye sahip olabilir.

Yerel Depolama ve Oturum Depolamanın aksine, IndexedDB sadece iplerden daha fazlasını depolayabilir. Yapılandırılmış klon algoritması tarafından desteklenen herhangi bir nesne IndexedDB'de saklanabilir.

EndeksedDB'de saklanabilen karmaşık bir JavaScript nesnesi örneği, ancak Yerel / Oturum Depolama'da değil CryptoKey'lerdir.

Web Crypto API'deki W3C önerisi, tarayıcıda devam etmesi gereken CryptoKey'lerin IndexedDB'de saklanmasını önermektedir.
Bir web sayfasını test ederken, IndexedDB'deki herhangi bir CryptoKey'i arayın ve olarak ayarlanıp ayarlanmadıklarını kontrol edin `extractable: true` Ayarlanmaları gerektiği zaman `extractable: false` (yani, kriptografik işlemler sırasında temel özel anahtar materyalin asla açığa çıkmamasını sağlayın.)

Print All the Contents of IndexedDB (IndexedDB'nin Tüm İçeriğini Yazdırın)

```
const dumpIndexedDB = dbName => {
  const DB_VERSION = 1;
  const req = indexedDB.open(dbName, DB_VERSION);
  req.onsuccess = function() {
    const db = req.result;
    const objectStoreNames = db.objectStoreNames || [];

    console.log(`[*] Database: ${dbName}`);

    Array.from(objectStoreNames).forEach(storeName => {
      const txn = db.transaction(storeName, 'readonly');
      const objectStore = txn.objectStore(storeName);

      console.log(`\t[+] ObjectStore: ${storeName}`);

      // Print all entries in objectStore with name `storeName`
      objectStore.getAll().onsuccess = event => {
```

```
      const items = event.target.result || [];
      items.forEach(item => console.log(`\t\t[-] `, item));
    };
  });
 };
};

indexedDB.databases().then(dbs => dbs.forEach(db => dumpIndexedDB(db.name
```

Web SQL

Web SQL, 18 Kasım 2010'dan beri dışlanır ve web geliştiricilerinin kullanmaması önerilir.

Cookies (Çerezler)

Çerezler, öncelikle oturum yönetimi için kullanılan önemli bir depolama mekanizmasıdır, ancak web geliştiricileri hala keyfi dize verilerini depolamak için kullanabilir.

Çerezler, Çerezlerin nitelikleri senaryosu için yapılan testlerde kapsamlı bir şekilde kaplanmıştır.

List All Cookies (Tüm Çerezleri Listeleyin)

`console.log(window.document.cookie);`

Global Window Object (Küresel Pencere Nesnesi)

Bazen web geliştiricileri, yalnızca sayfanın çalışma süresi ömrü boyunca mevcut olan küresel durumu küresel olarak özel atıflar atayarak başlatır ve korur. `window` Nesne. Örneğin:

```
window.MY_STATE = {
  counter: 0,
  flag: false,
};
```

Üzerinde eklenen herhangi bir veri `window` Sayfa yenilendiğinde veya kapatıldığında nesne kaybolacaktır.

List All Entries on the Window Object (Pencere Nesnesindeki Tüm Girişleri Listeleyin)

```
(() ⇒ {
  // create an iframe and append to body to load a clean window object
  const iframe = document.createElement('iframe');
  iframe.style.display = 'none';
  document.body.appendChild(iframe);

  // get the current list of properties on window
  const currentWindow = Object.getOwnPropertyNames(window);

  // filter the list against the properties that exist in the clean window
  const results = currentWindow.filter(
    prop ⇒ !iframe.contentWindow.hasOwnProperty(prop)
  );

  // remove iframe
  document.body.removeChild(iframe);

  // log key-value entries that are different
  results.forEach(key ⇒ console.log(`${key}: ${window[key]}`));
})();
```

(Bu snippet'in modifiye edilmiş versiyonu)

Attack Chain (Saldırı Zinciri)

Yukarıdaki saldırı vektörlerinden herhangi birinin tanımlanmasının ardından, DOM tabanlı XSS saldırıları gibi farklı türde istemci tarafı saldırılarıyla bir saldırı zinciri oluşturulabilir.

Remediation (Düzeltme)

Uygulamalar, hassas verileri sunucu tarafında değil, en iyi uygulamaları takiben güvenli bir şekilde saklamalıdır.

Referances (Referanslar)

- Local Storage
- Session Storage
- IndexedDB
- Web Crypto API: Key Storage
- Web SQL
- Cookies

HTML5 Web Depolama API'sinde daha fazla OWASP kaynağı için Oturum Yönetimi Hile Sayfasına bakın.