اختبار اختراق
Web Storage (User Experience)

الفهرس والكاتب

الكاتب :

عبدالرحمن عبدالله
يمكنك متابعتي على حسابي في لينكد ان
https://www.linkedin.com/in/abduc/

المراجع والمدقق :

emyovfelipovsky@

ملاحظة : جميع الثغرات الواردة في الورقة تم ابلاغ جميع الشركات عنها وتم اصلاحها مسبقا
تم نشر هذه الورقة كبحث وتجربة ولا اتحمل اساءة استخدام هذه الورقة

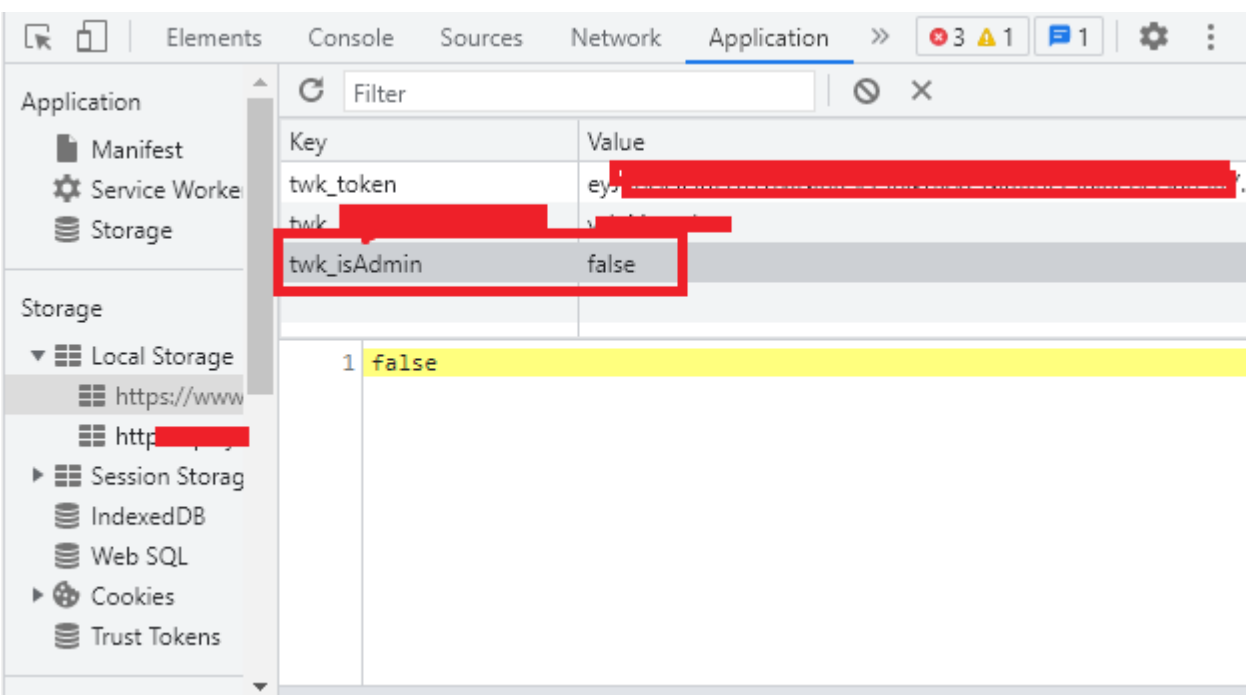
الفهرس :

المقدمة

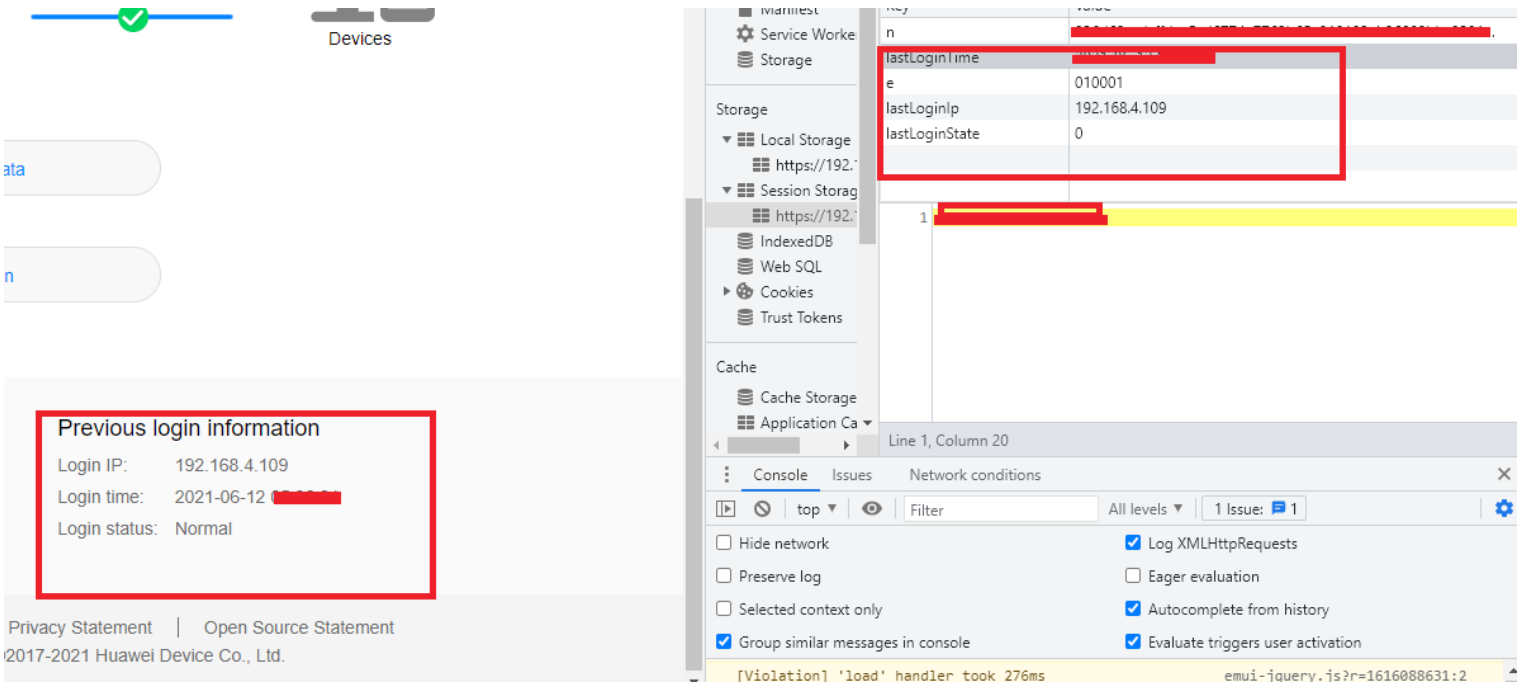
بسم الله الرحمن الرحيم

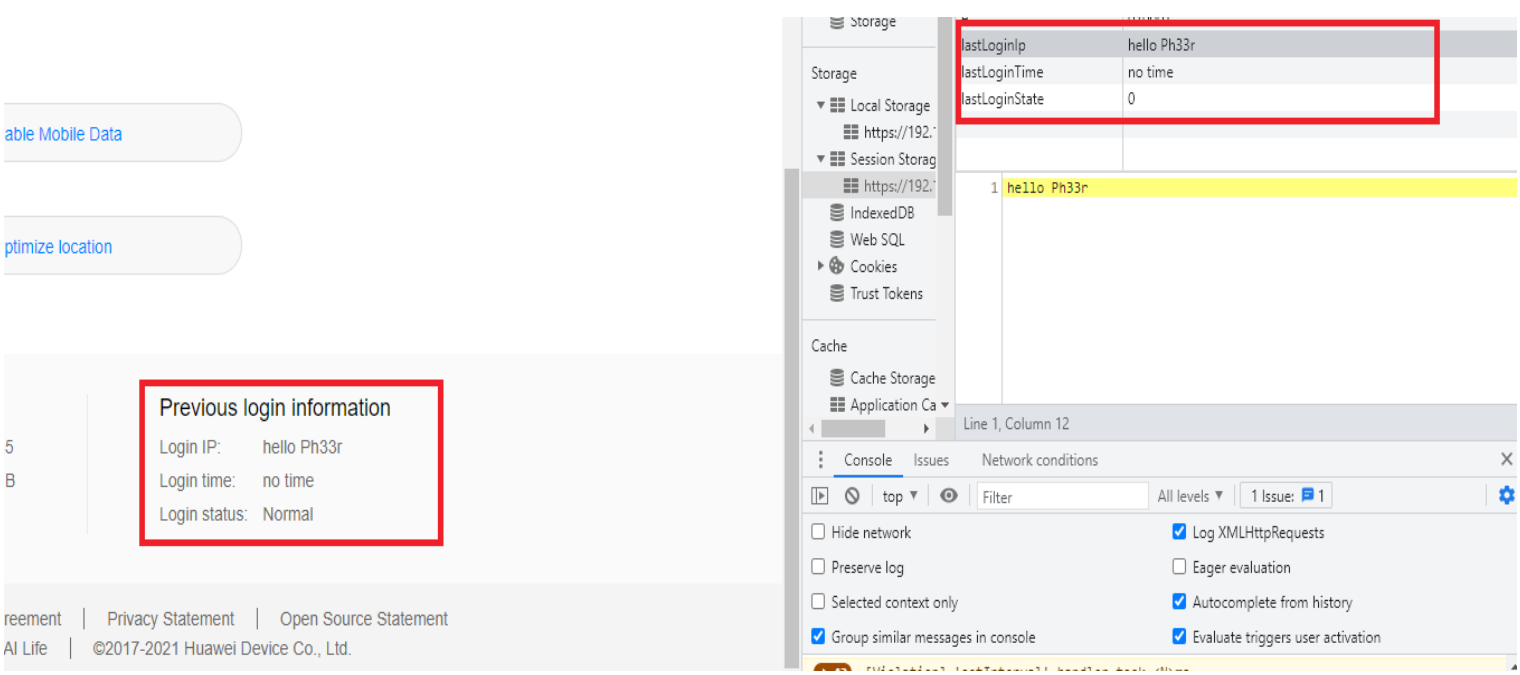
السلام عليكم ورحمة الله وبركاته

تعتبر الجافا سكربت الكنز المخفي لمختبري اختراق تطبيقات الويب ابلكيشن في هذه الورقة كتبت جزء بسيط عن الجافا سكربت حيث اني لم اجد اداة واحدة تفحص هذا النوع من الأخطاء مثل ZAP Proxy, BurpSuite لم اجد اي اداة تفحصها حسب علمي استخدمت owasp للقراءة عن طرق لاستغلال هذا النوع من الاخطاء وكانت النتيجة فقط نوعين وغير واضحة
Local Storage فقط (steal or load) ولكن وجدت عدة ثغرات مختلفة في Web Storage
امثلة بسيطة:



المقدمة


Devices
Previous login information
Login IP: 192.168.4.109
Login time: 2021-06-12
Login status: Normal
Privacy Statement | Open Source Statement
2017-2021 Huawei Device Co., Ltd.
Service Worke
Storage
lastLoginTime
e
010001
lastLoginIp
192.168.4.109
lastLoginState
0
Local Storage
Session Storag
IndexedDB
Web SQL
Cookies
Trust Tokens
Cache Storage
Application Ca
Line 1, Column 20
Console
Issues
Network conditions
top
Filter
All levels
1 Issue: 1
Hide network
Preserve log
Selected context only
Group similar messages in console
Log XMLHttpRequests
Eager evaluation
Autocomplete from history
Evaluate triggers user activation


able Mobile Data
ptimize location
Previous login information
Login IP: hello Ph33r
Login time: no time
Login status: Normal
reement | Privacy Statement | Open Source Statement
AI Life | ©2017-2021 Huawei Device Co., Ltd.
lastLoginIp
hello Ph33r
lastLoginTime
no time
lastLoginState
0
hello Ph33r
Line 1, Column 12
Console
Issues
Network conditions
Hide network
Preserve log
Selected context only
Group similar messages in console
Log XMLHttpRequests
Eager evaluation
Autocomplete from history
Evaluate triggers user activation

المقدمة

هل انت مهتم الان بطريقة اختبار اختراق لاتستطيع الادوات فحصها حاليا لانها تخزن في المتصفح
كقيمة نصية وعندما يطلبها الموقع لاستخدامها اذا اكتشف الموقع ان القيمة غير موجودة في المتصفح او في الطلب
يستخدم الموقع قيمة افتراضية مثال (لغتك العربية واللغة الافتراضية هيا الانجليزية)
لذلك لايمكن اختبارها الا عن طريق المتصفح لكن بطريقة المدرسة القديمة (يدوي)

لنتحدث الان بشكل مفصل ثم نتطرق الى طرق الاستغلال .

مـاهـي الـجافـا سـكربـت ؟

جافا سكريبت (بالإنجليزية: JavaScript) :

هي لغة برمجة عالية المستوى تستخدم أساسا في متصفحات الويب لإنشاء صفحات أكثر تفاعلية يتم تطويرها حاليا من طرف شركة نتسكيب وشركة موزيلا.

وكما تعرفها ويكيبيديا , فهي لغة برمجية عالية المستوى تستخدم أساساً في متصفحات الويب ولها استخدامات واسعة أخرى في المجالات الأخرى , تضيف اساساً تفاعلية عالية الى صفحة الويب الخاصة بك , ظهرت أساساً لغة جافا سكريبت للمبرمجين الهواة !
وتم ملاحظتها من عموم المبرمجين المحترفين وتطويرها وإصدار نسخ أعلى قوة منها لاحقاً لتصبح اللغة الأكثر اثارة للاهتمام من عام 2015 وحتى الآن.

يتم إنشاء ملفات جافا سكريبت بطريقة طبيعية , وباضافة امتداد .js الى الملف لتعريفه على أنه ملف جافا سكريبت .

استخدامات جافا سكريبت في برمجة الويب:

تؤثر لغة جافا سكريبت في برمجة الويب بشكل كبير جداً , وتعتبر من قواعد واساسيات اي مبرمج ويب فلن تجد اي مبرمج ويب الا ويستطيع التعامل مع لغة جافا سكريبت . تدخل جافا سكريبت بشكل مباشر في تصميم وبرمجة الويب على حد سواء فلها ميزاته ا التصميمية كمميزاتها البرمجية بنفس القدر وخصوصاً بظهور مكتبات جافا سكريبت الحديثة ك Vue.js و Angular و طبعاً jQuery

ماهي الجافا سكربت ؟

مميزات الجافا سكريبت : (مهم)

تستخدم الجافا سكريبت في البرمجة من طرف العميل أو ما تدعى client side لكنها تتميز بالعديد من التقنيات والميزات والتي منها:

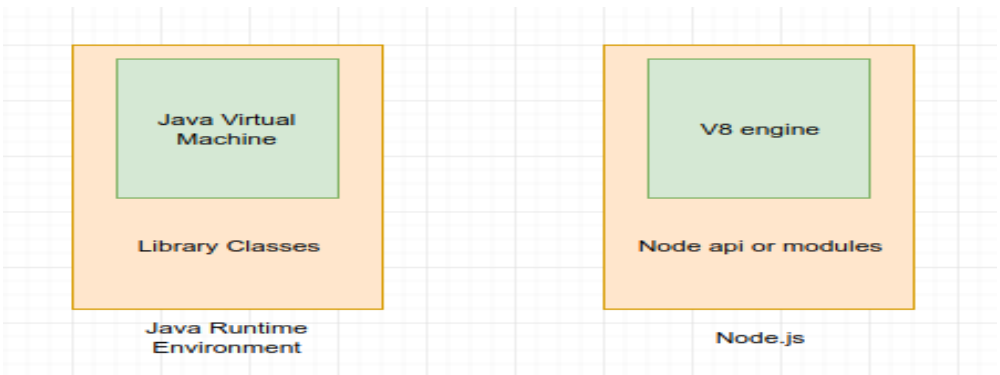
تنفذ من جهة العميل، فعلى سبيل المثال يمكنك التحقق من صحة أي مدخلاتٍ قبل إرسال الطلب إلى السيرفر .
تعتبر لغةً سهلة التعلم نسبيًا وقريبةً من اللغة الإنجليزية.
لغة برمجة مستقلة وليست كما يعتقد البعض أنها مرتبطةٌ بلغة جافا.
توفر خدمات تحكمية أكبر بالمتصفحات.
تفاعلية وسريعة.
تتميز بواجهات غنية ويمكنك سحب وإسقاط المكونات لتغني واجهتك بالعناصر المطلوبة.
لغة برمجة وظيفية.
يمكنها التعديل على ملفات html ,css
التعامل مع الكوكيز والطلبات ajax,comet
حفظ البيانات في التخزين المحلي للجهاز،(موضوع اليوم)

أشياء لا تستطيع جافا سكربت في المتصفح القيام بها :

هناك حدود للجافا سكريبت في المتصفح بحيث لا يستطيع تجاوزها، والهدف بكل تأكيد هو حماية بيانات وملفات المستخدمين من الوصول إليها من قبل بعض المواقع الخبيثة فعلى سبيل المثال

لا يمكن لأي موقع الوصول إلى الملفات في حاسوب المستخدم أو قراءتها إلا إذا قام المستخدم بنفسه برفع تلك الملفات إلى ذلك الموقع عن طريق عنصر File Input أو تقنية السحب والإفلات (Drag and Drop).
حتى في داخل المتصفح الواحد، لا يمكن لجافا سكربت في موقع مفتوح (Tab) أن يصل إلى محتويات موقع آخر في (Tab) مختلف.
يمكن لجافا سكريبت أن يتواصل ويتبادل البيانات بشكل عادي وسهل مع خادم الموقع الذي قام بإرسال تلك الصفحة التي يشتغل فيها ذلك الجافا سكريبت، بينما يصبح هذا الأمر مقيدا إذا كانا الإثنان من أصول مختلفة. فمثلا لا يمكن لجافا سكريبت من موقع google أن يرسل بيانات أو يحصل عليها من موقع exploit-db.com
إذا لم يسمح هذا الأخير بذلك. هذه السياسية تعرف Same-origin Policy والهدف منها هو عزل أكواد جافا سكريبت الخبيثة في نطاق محدد لا تستطيع تجاوزه.

محركات جافا سكربت



تعتبر Node.js بيئة تشغيل- runtime environment خاصة بلغة جافا سكريبت- JavaScript.

لكن ماذا يعني ذلك؟ وكيف تعمل؟ تتضمن بيئة التشغيل الخاصة بـ Node.js كل ما تحتاجه لتنفيذ برنامج مكتوب بلغة JavaScript.

ظهرت تقنية Node.js عندما وسع المطورون الأصليون نطاق لغة JavaScript من لغة لا يمكنك تنفيذها إلا من خلال المتصفح browser إلى شيء يمكنك تشغيله على جهازك مباشرة في شكل تطبيق مستقل، وبذلك اصبح بالإمكان فعل كثيرٍ من الأمور باستخدام لغة JavaScript أكثر من مجرد جعل مواقع الويب تفاعلية.

الآن؛ أصبح لدى لغة JavaScript القدرة على القيام بأشياء يمكن أن تفعلها اللغات النصية – scripting languages الأخرى مثل لغة الـ Python (وهي لغات مُعدّة لتنفيذ مهامٍ بغرض التحكم بالتطبيق المبني وتُعالَج باستخدام المفسر interpreter لا المترجم compiler).

Node.js

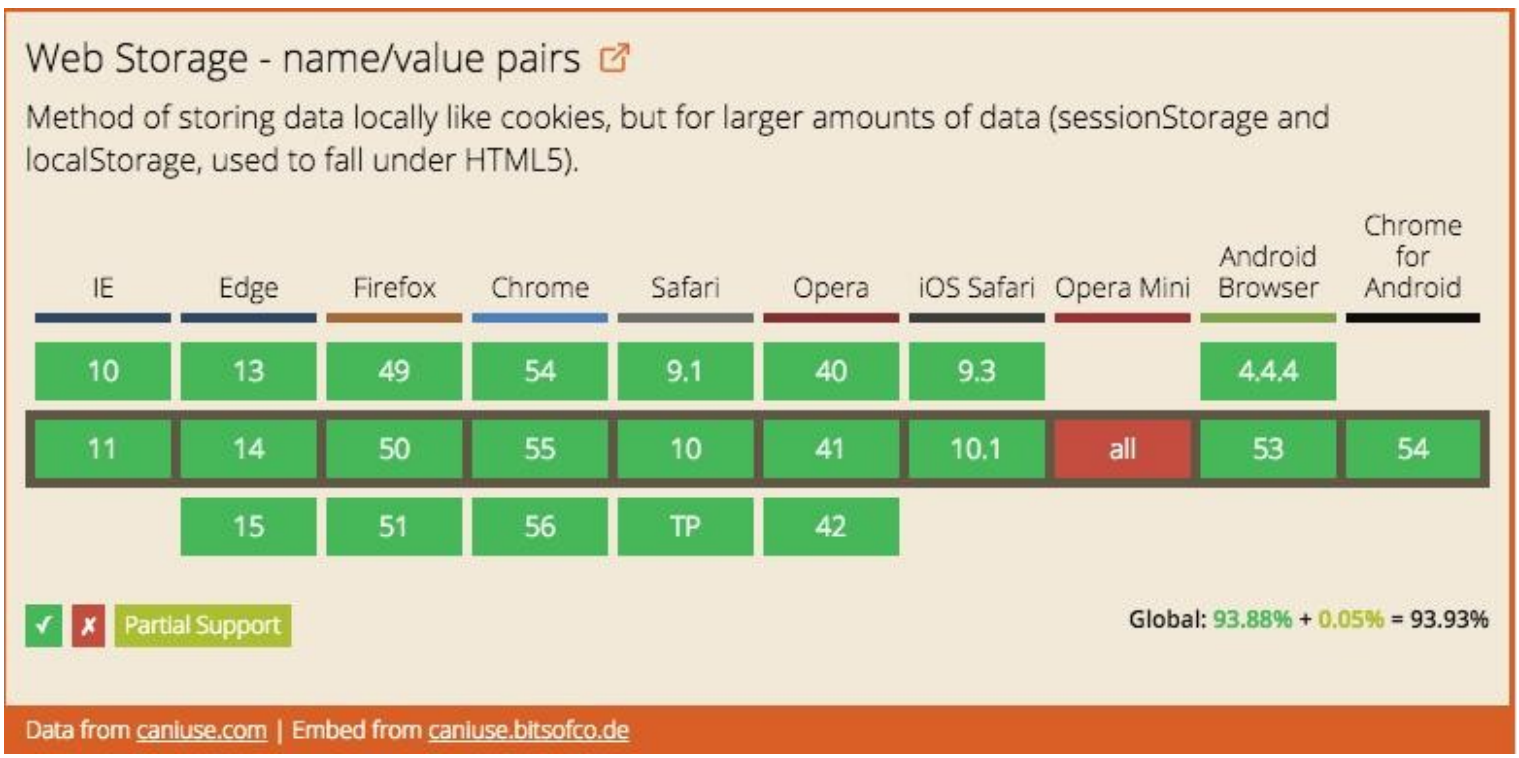
هي بيئة تشغيل خاصة بلغة JavaScript مبنية باستخدام محرك متصفح Chrome الذي يدعى V8.
تُنَفَذ لغة JavaScript في متصفح Chrome باستخدام محرك التشغيل V8 (وهو محرك مفتوح المصدر مكتوب بلغة C++)، إذ يوجد لكل متصفح المحرك الخاص به، وهو نفس المحرك الذي تستخدمه Node.js.
يمكن تشغيل محرك V8 بشكل مستقل، أو يمكن تضمينه في أي تطبيق مكتوب بلغة C++، كذلك يحتوي على إضافات تسمح لك بكتابة كود بلغة C++ وجعله متاحًا للاستخدام في JavaScript.

يأخذ هذا المحرك الكود المكتوب بلغة JavaScript ويحوله إلى لغة الآلة Machine code التي يمكن للحاسوب تنفيذها بشكل مباشر وسريع؛ حيث إن لغة الآلة هي كود منخفض المستوى أقرب إلى كونه مجموعة من 0 و1 يمكن للحاسوب تنفيذه مباشرة دون الحاجة إلى تفسيره وتحويله إلى لغة اخرى

for more

https://en.wikipedia.org/wiki/JavaScript_engine

التخزين في HTML5



هناك العديد من المزايا لتخزين البيانات مباشرة في المتصفح ، والأهم هو الوصول إلى "قاعدة البيانات" بسرعة وبشكل مستقل عن الشبكة. توجد حاليًا أربع طرق (بالإضافة إلى الإيقاف) لتخزين البيانات على العميل:

Cookies
Local Storage
Session Storage
IndexedDB
WebSQL (مهمل)
Cookies

التخزين في HTML5

Cookies

يتم إرسال البيانات إلى العميل ، ثم تخزينه وإعادتها إلى Cookies ،إنها طريقة كلاسيكية لتخزين البيانات ، وتخزين بيانات سلسلة بسيطة في ملف. عادة .السيرفر في الطلبات اللاحقة. يمكن استخدام الكوكيز لإدارة الحسابات وتتبع معلومات المستخدم

يمكننا إنشاء ملفات تعريف الارتباط وقراءتها وتحديثها وحذفها باستخدام بناء الجملة التالي:

```
// Read (All)
console.log( document.cookie );
// Update
document.cookie = "token_age=08;max-age=31536000;secure";
// Delete
document.cookie = "token=0x00;expires=Thu, 01 Mar 1337 00:00:01 GMT";
```

مزايا ملفات تعريف الارتباط

يمكن استخدامها للتواصل مع السيرفر

يمكن ضبطه على انتهاء صلاحيته تلقائيًا دون الحاجة إلى الحذف يدويًا

مساوئ ملفات تعريف الارتباط

عدد محدود من ملفات تعريف الارتباط وطولها

يمكن تخزين السلاسل فقط

مشاكل أمنية محتملة

مدعوم بشكل أساسي في جميع المتصفحات الرئيسية

التخزين في HTML5

Local Storage :

نعم **Web Storage API** واحد منهم هو **API** الذي يخزن بيانات القيمة الرئيسية في المتصفح. إنه يحل المشكلة من خلال توفير واجهة برمجة تطبيقات أكثر سهولة وأمانًا لتخزين البيانات البسيطة في المتصفح **Cookies** مشكلة.

على الرغم من أنه من الناحية الفنية ، يمكننا فقط تخزين السلاسل في التخزين المحلي ، إلا أنه يمكن تخزين **JSON** مع **Cookies** على سبيل المثال ، يسمح لنا هذا بتخزين بيانات أكثر تعقيدًا محليًا.

يمكننا إنشاء ملفات تعريف الارتباط وقراءتها وتحديثها وحذفها باستخدام بناء الجملة التالي:

```
// Create
const user = { name: 'Ph33r', age: 08 }
localStorage.setItem('user', JSON.stringify(user));

// Read (Single)
console.log( JSON.parse(localStorage.getItem('user')) )

// Update
const updatedUser = { name: 'Ph33r', age:  08 }
localStorage.setItem('user', JSON.stringify(updatedUser));

// Delete
localStorage.removeItem('user');
```

مزايا التخزين المحلي

يوفر واجهة أبسط وأكثر سهولة لتخزين البيانات (مقارنة بملفات تعريف الارتباط)

التخزين المحلي أكثر أمانًا (مقارنة بملفات تعريف الارتباط)

السماح بتخزين المزيد من البيانات (مقارنة بملفات تعريف الارتباط)

مساوئ التخزين المحلي

السماح فقط بتخزين السلاسل

التخزين في HTML5

Session Storage :

هذا هو النوع الثاني من واجهة برمجة تطبيقات تخزين الشبكة. مع Local Storage مشابه جدا ، الفرق هو Local Storage لا يوجد إعداد وقت انتهاء الصلاحية للبيانات المخزنة فيه Session Storage سيتم مسح البيانات الموجودة في الداخل في نهاية جلسة الصفحة.

يمكننا إنشاء ملفات تعريف الارتباط وقراءتها وتحديثها وحذفها باستخدام بناء الجملة التالي:

```
// Create
const user = { name: 'Ph33r', age: 08 }
sessionStorage.setItem('user', JSON.stringify(user));

// Read (Single)
console.log( JSON.parse(sessionStorage.getItem('user')) )

// Update
const updatedUser = { name: 'Ph33r', age: 08 }
sessionStorage.setItem('user', JSON.stringify(updatedUser));

// Delete
sessionStorage.removeItem('user');
```

التخزين في HTML5

IndexedDB

إنه حل أكثر تعقيدًا ولكنه أكثر شمولًا للمتصفحات لتخزين البيانات. يعد IndexedDB
"واجهة برمجة تطبيقات لتخزين كمية كبيرة من البيانات المنظمة على العميل واستخدام فهارس على هذه البيانات لاسترجاعها عالي الأداء" (Mozilla).
هذه قاعدة بيانات تعتمد على جافا سكريبت وموجهة للكائنات تسمح لنا بتخزين البيانات واسترجاعها بسهولة

كيفية استخدام IndexedDB لإنشاء تطبيق دون اتصال.

بالمقارنة مع طرق تخزين المتصفح الأخرى ، فإن استخدام IndexedDB أكثر تعقيدًا. قبل أن نتمكن من إنشاء / قراءة / تحديث / حذف أي بيانات ، نحتاج أولاً إلى فتح قاعدة البيانات وإنشاء مستودع بيانات.

```
function OpenIDB() {
 return idb.open('SampleDB', 1, function(upgradeDb) {
  const users = upgradeDb.createObjectStore('users', {
   keyPath: 'name'
  });
 });
}
```

لإنشاء (أو تحديث) البيانات ، نحتاج إلى اتباع الخطوات التالية:

```
// 1. Open up the database
OpenIDB().then((db) => {
 const dbStore = 'users';
// 2. Open a new read/write transaction with the store within the databas
 const transaction = db.transaction(dbStore, 'readwrite');
 const store = transaction.objectStore(dbStore);
// 3. Add the data to the store
 store.put({
  name: 'Ph33r',
  age: 08
 });
 // 4. Complete the transactio
 return transaction.complete;
});
```

لقراءة البيانات ، نحتاج إلى اتباع الخطوات التالية:

```
// 1. Open up the database
OpenIDB().then((db) => {
 const dbStore = 'users';

 // 2. Open a new read-only transaction with the store within the database
 const transaction = db.transaction(dbStore);
 const store = transaction.objectStore(dbStore);
 // 3. Return the data
 return store.get('Ph33r');
}).then((item) => {
 console.log(item);
})
```

لحذف البيانات ، نحتاج إلى اتباع الخطوات التالية :

```
// 1. Open up the database
OpenIDB().then((db) => {
 const dbStore = 'users';

 // 2. Open a new read/write transaction with the store within the database
 const transaction = db.transaction(dbStore, 'readwrite');
 const store = transaction.objectStore(dbStore);

 // 3. Delete the data corresponding to the passed key
 store.delete('Ph33r');

 // 4. Complete the transaction
 return transaction.complete;
})
```

التخزين في HTML5

مزايا DB :

يمكن التعامل مع البيانات الأكثر تعقيدًا وتنظيمًا

في كل "قاعدة بيانات" ، يمكن أن تكون هناك عدة "قواعد بيانات" و "جداول"

مساحة تخزين أكبر

تحكم في كيفية تفاعلنا معها

عيوب IndexedDB

معقدة للغاية مقارنة بواجهة برمجة تطبيقات تخزين الويب الأخرى

انتهينا من التفاصيل وطريقة عمل التخزين في HTML5 بالاضافة المزايا

لنتحدث الان ببعض الفروقات الجوهرية بين Local Storage ,sessionStorage,IndexedDB :

قبل البدء بالتعرف على طرق الاستغلال ذكرت في الاعلى ان الفرق بين Local Storage ,sessionStorage, هوا ان sessionStorage بمجرد اغلاق الصفحة او المتصفح الذي يخزن

البيانات يتم مسح جميع البيانات المخزنة فيه لكن Local Storage تحفظ بالمتصفح ولا تمسح الا بتدخل المستخدم نفسه

ويفضل اغلب المبرمجين استخدام هذه الطريقة في تخزين بعض البيانات الخاصة بتجربة المستخدم مثل تخزين لغة المستخدم,

لها عدة طرق في الاستخدام تختلف باختلاف الويب ابكليشن والمبرمج والان لنتطرق لها

كيف يمكنني اكتشافها ؟

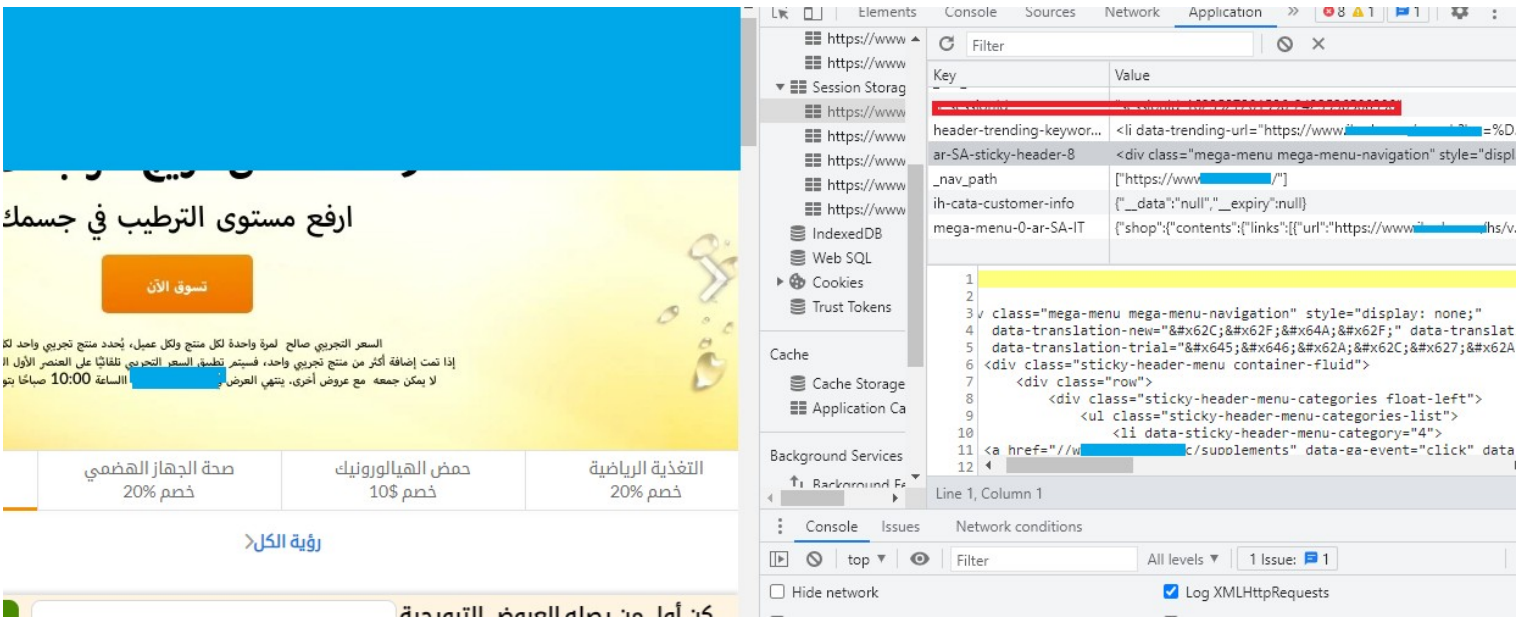
طرق اكتشافها:

سهلة وغير سهلة حيث ان الغير سهلة تحتاج منك الى معرفة مسبقة في قراءة اكواد الجافا سكربت لمحاولة تتبع

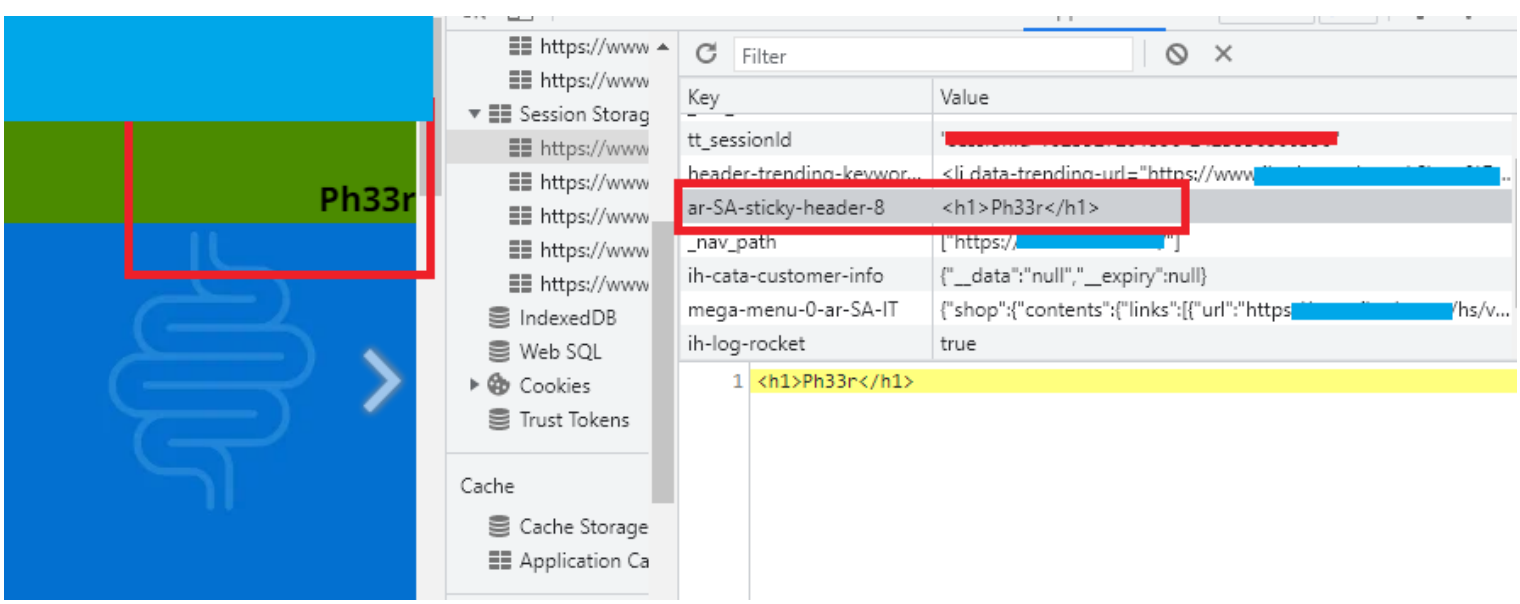
سبب التخزين وهل يتم عرضها للمستخدم او يتم تخزينها في ملفات لوق لمحللي البيانات او محللي تجربة المستخدم
او لايتم عرضها من الاساس نقسم اكتشاف هذا النوع من الثغرات الى قسمين (سهل وصعب)

القسم الاول السهل :
نستخدم التجربة واضافة اكواد html داخل Web Storage وهل يتم عرضها داخل المتصفح للمستخدم او لا ايضا لاننسى ان بعض
المواقع تخزن التوكن داخل Web Storage بحيث عند عمل عملية اختبار اختراق لموقع تحاول تغيير قيمة الكوكيز
بقيمة مختلفة كتغيير سعر المنتج او تغيير الصلاحيات والقيمة الموجودة داخل الكوكيز لا يتم التحقق منها لكن القيمة المخزنة بالمتصفح
يتحقق منها توجد عدة سيناريوهات وجدتها سيتم شرحها في الجزء الثاني باذن الله
مثال بسيط هذا موقع يخزن لغة المستخدم داخل sessionStorage اذا تستخدم اللغة العربية او لغة اخرى مختلفة غير الانجليزية
سيتم تخزين اللغة في sessionStorage اذا استخدمت مثال لBurpSuite او ZapProxy الموقع سيعرض لك فقط اللغة الافتراضية
وهي الانجليزية
لذلك نستخدم المتصفح في اكتشاف هذا النوع من الثغرات لكي لايتم عرض القيمة الافتراضية وانما يتم عرض القيمة من تجربة المستخدم
واختياره لنتابع الامثلة المصورة

كيف يمكنني اكتشافها ؟



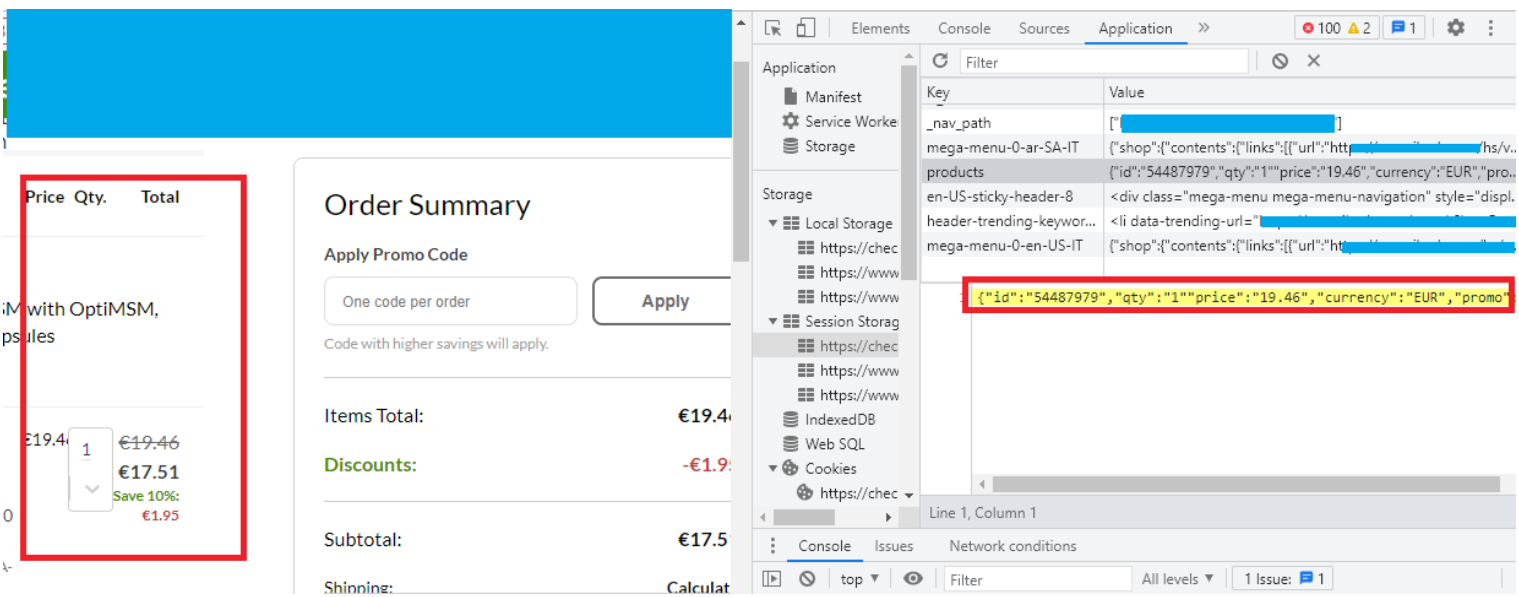
نحقن الكود الان ونحدث الصفحة هل حدث تغيير في الصفحة ام لا

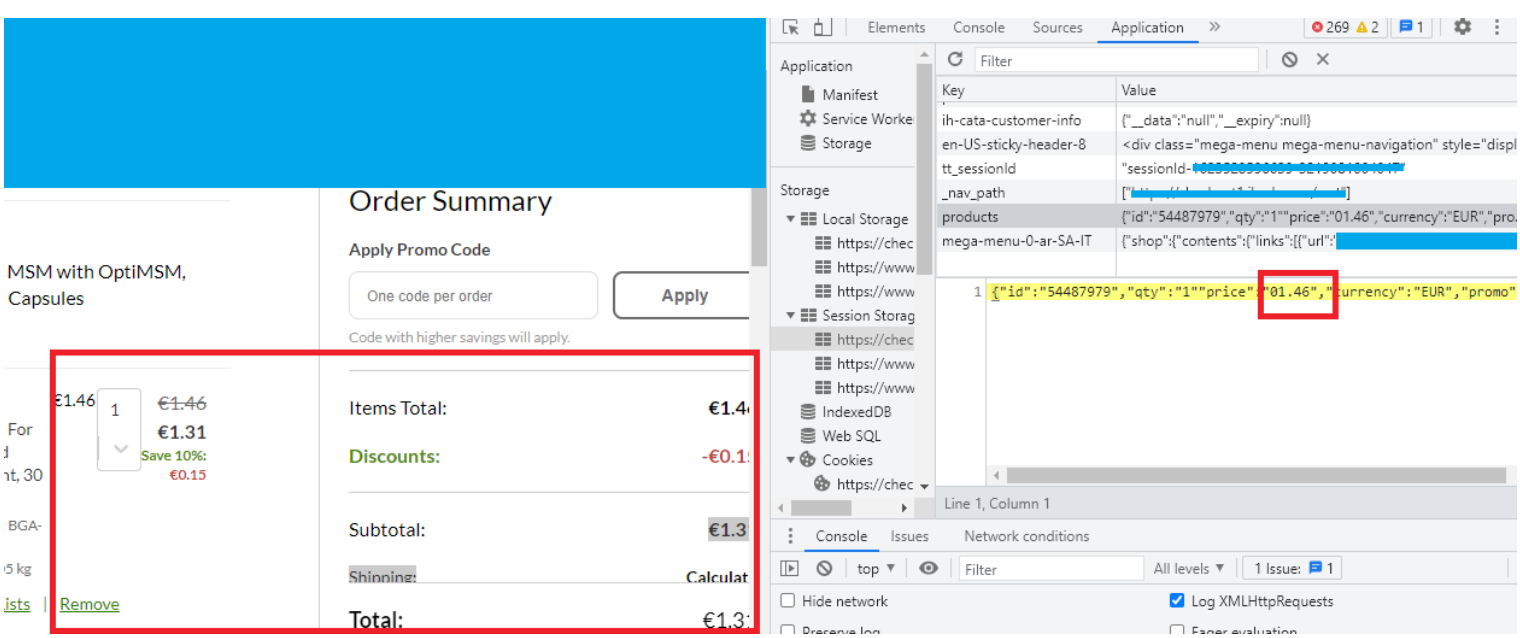


لقد تمت العملية بنجاح اذا الان هذه ثغرة من نوع self xss

كيف يمكنني اكتشافها ؟

ماذا عن ثغرات business logic





كيف يمكنني اكتشافها ؟

القسم الثاني الصعب :

سيتم شرح هذا في ورقة اخرى بشكل متقدم حيث ان بعض الشركات الى الان لم تنتهي من إصلاح بعض الثغرات بعد الانتهاء من الإصلاح سيتم نشر هذا القسم
في ورقة مختلفة مع كتابة كيف تم إيجاد هذه الثغرات

part2
00x01 = blhblh
00x02 = code review
00x03 = code review 2
00x04 = Blind XSS for (UX Analysts)
00x05 = 2 Token in (cookies,localstorage) .. !!
00x06 = Key Pollution (storage)
00x07 = tips

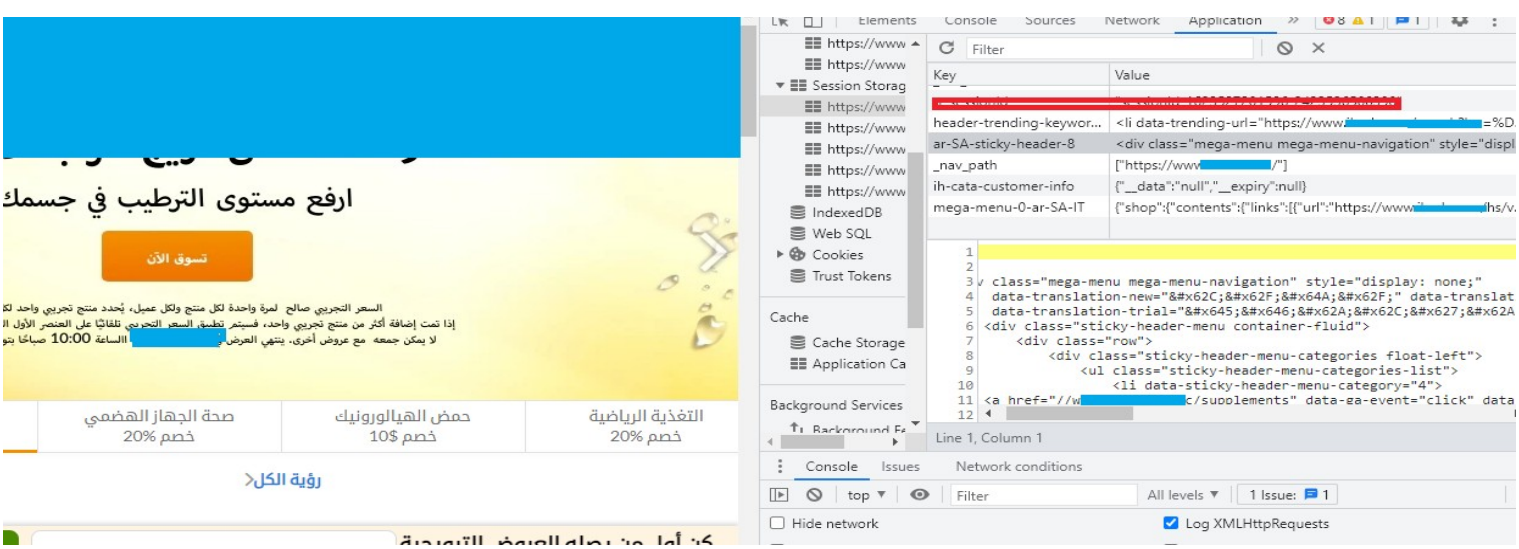
XSS Reflected to XSS Stored with lab تحويل ثغرة

كيف يمكنني الاستفادة من ثغرة self XSS لتصعيد الخطورة يجب في البداية ايجاد ثغرة من نوع XSS Reflected
عن طريقها نستطيع تغيير اي قيمة مخزنة في المتصفح تخص نفس الموقع وهو ان نحقن بايلود داخل اي من Web Storage
عن طريق عمل تحديث للقيم الموجودة داخلها مثال :

```
const updatedUser = { name: '<h1>Ph33r</h1>', age:  08 }
localStorage.setItem('user', JSON.stringify(updatedUser));
```

الان لدينا ثغرة من نوع XSS Reflected وايضا يقوم الموقع بتخزين جميع القيم التي تخص تجربة المستخدم داخل IndexedDB او localStorage
اول خطوة علينا القيام بها هيا الوصول لمفتاح القيمة التي نحتاج الى تغييرها
مثال اذا كان داخل:

```
localStorage.getItem('مفتاح القيمة المطلوب') OR  sessionStorage.getItem('مفتاح القيمة المطلوب')
```




```
sessionStorage.getItem('ar-SA-sticky-header-8')
```

بعد وصولنا للمفتاح والتاكد انه المفتاح الصحيح نحدث القيمة الموجودة داخل المفتاح :

```
sessionStorage.setItem('ar-SA-sticky-header-8', “code html”)
```

لنفترض اننا الان وجدنا ثغرة من نوع Reflected كل ما علينا القيام به الان حقن الكود الذي يمكننا من تحديث قيمة المفتاح بطريقة مخفية بالطريقة التالية :

```
http://exploit-db.com/index?xssReflected=<script>sessionStorage.setItem('ar-SA-sticky-header-8', “code html”)</script>
```

ويفضل اخفاء البايلود داخل نفس القيمة مع عدم تغيير القيمة الاصلية مثال عندما يتصفح الضحية الموقع يتم تشغيل البايلود في جهازه في كل مره يستخدم فيه الموقع خصوصا اذا كان البايلود تم حقنه داخل localStorage او IndexedDB بطريقة مخفية البايلود لن يتغير حتى يتم حذفه من قبل المستخدم

برمجة لاب صغير ولا يحتاج الى سيرفر محلي كل ما عليك هوا تحميل الملفات واستعراضها داخل اي متصفح ما عدا متصفح اكسبلور لاب مستوى سهل فيه ثغرتين من نوع XSS تقدرون تطبقون عليها كل الكلام الا تم ذكره

https://github.com/Ph33rr/webstorage

0x08 = اسرار (tips)

تعتمد تخزين البيانات داخل Web Storage على تجربة المستخدم في معظم المواقع
(لغة مفضلة , ستايل مفضل , عملة مفضلة ..الخ)

قبل فحص الموقع عليك تغيير جميع الاعدادات الافتراضية ثم اين يتم تخزينها داخل المتصفح بعدها تختبر القيم

في بعض مواقع التسوق يتم تخزين المنتجات التي تم اضافتها الى Web Storage اذا كنت زائر ولكن اذا كنت مستخدم يتم تخزين بياناتك داخل اسم قاعدة البيانات عليك التحقق من هذا قبل عملية اختبار الاختراق

يمكنك ايضا ايجاد التوكن مخزنة في Web Storage اذا لما تجدها جرب ان تستخدم تذكرني. . !!

0x09 = مراجع

https://www.researchgate.net/publication/259081595_An_Investigation_into_Possible_Attacks_on_HTML5_IndexedDB_and_their_Prevention
https://en.wikipedia.org/wiki/JavaScript_engine
https://en.wikipedia.org/wiki/JavaScript
https://www.w3schools.com/html/html5_webstorage.asp
https://academy.hsoub.com/programming/html/html5/%D8%A7%D9%84%D8%AA%D8%AE%D8%B2%D9%8A/%D9%86-%D8%A7%D9%84%D9%85%D8%AD%D9%84%D9%8A-local-storage-%D9%81%D9%8A-html5-r362/
https://arabicprogrammer.com/article/3636627125/
https://www.tutomena.com/what-is-javascript/