دانشگاه صنعتی اصفهان

دانشکده برق و کامپیوتر

مقطع: کارشناسی

گرایش: نرم‌افزار

حملات مبتنی بر کاربردهای وب - XSS

اردیبهشت ۱۳۹۹

فهرست مطالب

صفحه

بخش اول:

آشنایی با حملات XSS

حملات XSS[1] نوعی از حملات مبتنی بر تزریق کد هستند که در آن‌ها مهاجم اسکریپت‌های مخرب خود را به وب سایت هدف تزریق می‌کند. حملات XSS زمانی رخ می‌دهد که مهاجم از یک کاربرد وب برای ارسال کد مخرب (عموما به صورت ارسال یک کد مخرب در قالب اسکریپت مرورگر[2]) به یک کاربر نهایی[3] استفاده می‌کند. نواقص و باگ‌هایی که منجر به این حملات می‌شوند شایع هستند و در جاهایی رخ می‌دهد که، یک کاربرد وب[4] از کاربر ورودی می‌گیرد و یک خروجی را بدون ارزیابی کردن آن، به کاربر نشان می‌دهد.

یک مهاجم می‌تواند از XSS استفاده کند و یک اسکریپت مخرب را به یک کاربر معمولی ارسال کند. کاربران در حال استفاده از مرورگر نیز راهی برای فهمیدن مخرب بودن اسکریپت ندارند چراکه این اسکریپت از یک منبع معتبر[5] آمده است، لذا به اجرای آن می‌پردازند. هم چنین از آن جا که این اسکریپت از یک منبع معتبر آمده است می‌تواند به کوکی‌ها[6]، توکن‌ها[7] و دیگر اطلاعات حساس که توسط مرورگر نگهداری شده و برای آن وب سایت استفاده می‌شود، دسترسی پیدا کند. بنابراین نقطه قوت این دسته از حملات تشخیص دشوار آن‌ها توسط کاربر به دلیل ارسال کد مخرب از طریق یک وب سایت معتبر می‌باشد. هر وب‌سایتی که معیارهای امنیتی استاندارد را رعایت نکرده باشد، احتمال حضور آسیب‌پذیری در آن وجود دارد. این دسته از آسیب‌پذیری‌ها عموما در فیلد جستجوی وب سایت یافت می‌شوند.

---

[1] Cross-Site Scripting

[2] Browser Side-Script: این نوع از اسکریپت به کاربر اجازه می‌دهد که قابلیت‌های مرورگر را به کمک جاوااسکریپت افزایش دهد

[3] End-User | Client

[4] Web Application

[5] Trusted

[6] Cookies

[7] Tokens

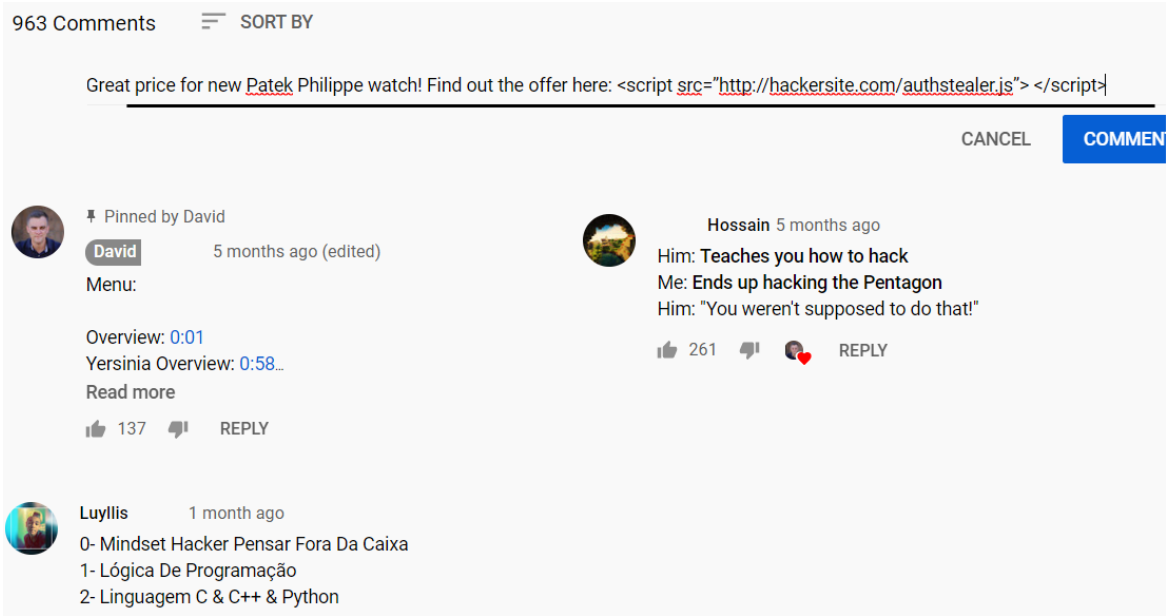
حملات XSS به طور کلی چندین دســـته‌بندی مختلف دارند، که دو مورد از مهم‌ترین آن‌ها به نام‌های زیر می‌باشد:

۱. Stored: AKA Presistent یا همان مدل اول (Type ۱)

۲. Related: AKA Non-Persistent یا همان مدل دوم (Type ۲)

۲-۱- Stored XSS

مدل اول به طور کلی زمانی رخ می‌دهد که ورودی کاربران (User Inputs) برروی ســرور هدف (همانند یک پایگاه‌داده) در قالب نظرات کاربر، پیام‌های او در انجمن و گزار شات بازدیدکنندگان از سایت ذخیره می شود. سپس قربانی می‌تواند این داده‌های ذخیره شده را بدون اینکه در مرورگر ایمن شوند، از طریق کاربرد وب دریافت کند. برای مثال فرض کنید در یک ســایت امکان ثبت نظرات در زیر پســت‌ها وجود داشته باشد. مهاجمی متوجه وجود باگ XSS در وب سایت شده و کامنت زیر را ثبت می‌کند:



از لحظه‌ای که این کامنت در سایت ثبت می‌شود تگ‌های HTML موجود درکامنت فایل جاوااسکریپتی را فعال می‌کنند که روی ســایت دیگری قرار دارد و توانایی ربودن Session ID کاربران را دارد. داشــتن

کوکی مربوط به Session کاربران، می‌تواند منجر به دسترسی مهاجم به حساب کاربری اعضای سایت و در نتیجه دسترسی مهاجم به اطلاعات شخصی افراد شود. اینها همه در حالی است که کاربر حتی ممکن است در صفحه تا بخش نظرات اسکرول نکرده باشد و از جمله خبر نداشته باشد. یعنی برخلاف حملات XSS نوع دوم که به آن‌ها reflected (اســـکریپت بعد از کلیک کردن برروی آن فعال می‌شـــود) گفته می‌شود، در حملات XSS نوع اول یعتی stored، کافیست کاربر صفحه وب دستکاری شده را باز کند.

Reflected XSS -۲-۲

این نوع از حملات XSS زمانی رخ می‌دهد که ورودی کاربر فورا در قالب:

- یک پیام خطا
- نتیجه جستجو
- یا هر پاسخ دیگری که همه یا بخشی از ورودی کاربر را در خود دارد

پاسخ داده می‌شود بدون اینکه قبل از رسیدن به مرورگر کاربر، این پاسخ از نظر امن بودن مورد بررسی قرار گرفته باشد.

بر خلاف نوع Stored از حملات XSS مه نیاز است مهاجم وبسایتی را پیدا کند که اجازه تزریق دائمی کد مخرب را می‌دهند (مثل قرارگیری کد مخرب در قسمت نظرات یک صفحه خرید) در این نوع از حملات XSS یعنی حملات Reflected، کافیست کد مخرب درون یک لینک تعبیه شود. به همین دلیل برای اینکه حمله موفقیت‌آمیز باشد، قربانی میبایست برروی لینک حاوی کد مخرب کلیک کند.

به طور کلی ۳ تفاوت اصلی میان حملات XSS نوع Reflected و Stored وجود دارد:

- حملات Reflected مرسوم‌تر هستند
- حملات Reflected کمتر از حملات Stored برای مهاجم سودبری دارند

- حملات Reflected توسط کاربرانی که هوشیار هستند (برروی لینک‌های نامربوط کلیک نمی‌کنند) مورد اجتناب قرار می‌گیرند

طبیعی است در حملات Reflected مهاجم سعی می‌کند با ارسال لینک مخرب به تعداد بیشتری از کاربران، شانس خود برای موفق شدن را افزایش دهد.

بخش دوم:

انجام حمله XSS

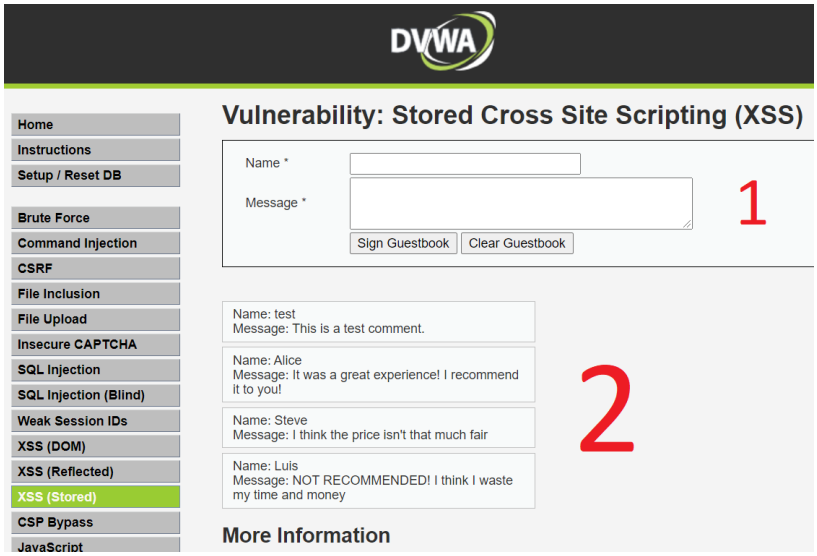
برای شروع به کار، از یک کاربرد وب با اکثر آسیب‌پذیری‌های مرسوم و موجود به نام DVWA[۸] تحت عنوان محیط آزمایشگاهی استفاده می‌کنیم. ابتدا DVWA را نصب کنید. برای نصب آن می‌توانید از ویدیو زیر نیز کمک بگیرید:

https://www.youtube.com/watch?v=UoS۰hysLADU

پس از نصب DVWA، و ورود کردن به وسیله نام کاربری و رمزعبور، از منوی سمت چپ می‌توانید نام تعداد زیادی از آسیب‌پذیری‌های مرسوم و شایع در زمینه کاربردهای وب را مشاهده کنید. ابتدا برای آزمایش اولیه، از منوی DVWA Security امنیت کاربرد وب را پایین بیاورید.

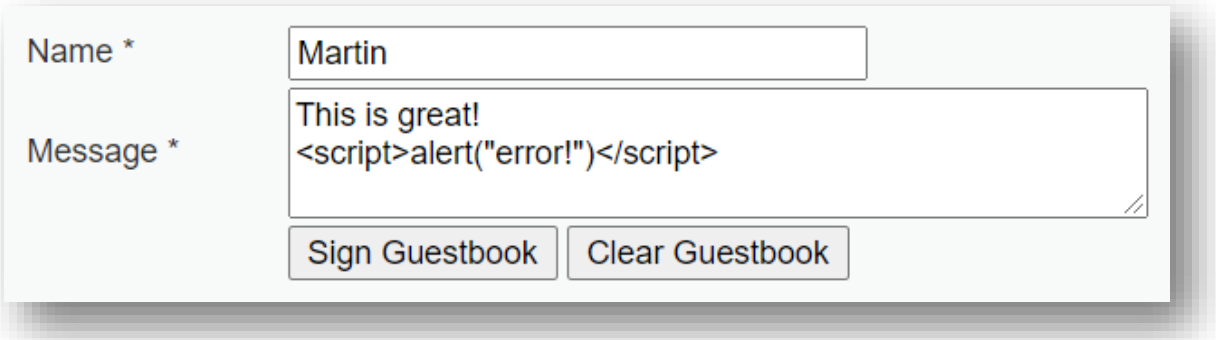
۲-۱- حمله XSS(Stored)

ابتدا وارد DVWA شده و به منوی XSS(Stored) بروید. در اینجا با یک صفحه مواجه هستیم که در آن نام و پیام از کاربر دریافت می‌شود، در پایگاه داده‌ای ذخیره می‌شود و بعد از آن هر کس به این صفحه مراجعه کند، پیام او در قسمت پایینی نمایش داده می‌شود:

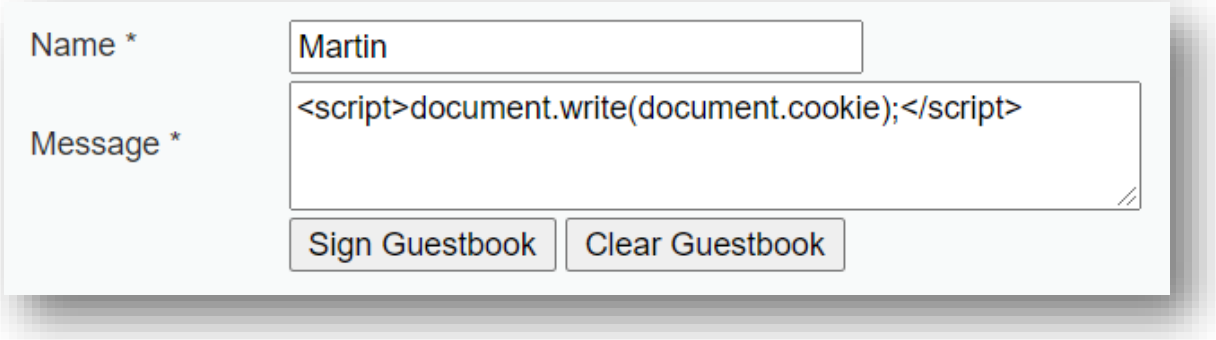


[۸] Damn Vulnerable Web App

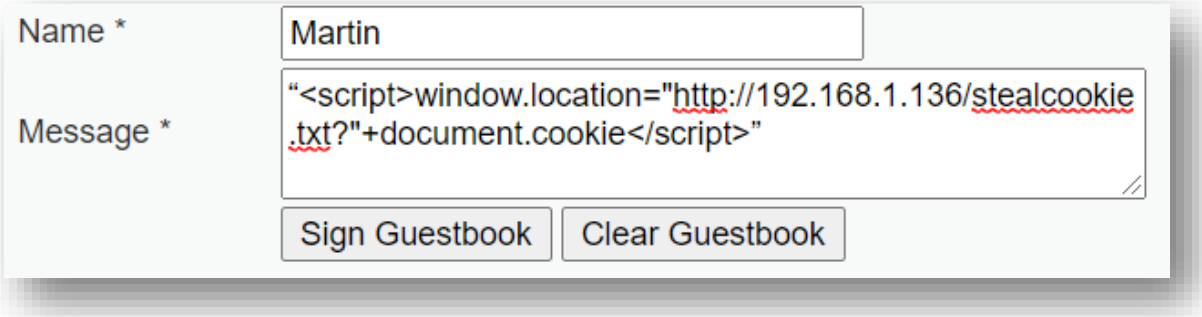
مطابق شکل بالا در قسمت ۱ کاربر می‌تواند نظر جدیدی را وارد کند و در قسمت ۲ می‌تواند نظرات ثبت شده را ببیند. در این جا به نظر می‌رسد که نظرات کاربران در پایگاه‌داده‌ای ذخیره می‌شود بنابراین اگر ورودی‌های کاربران به هنگام ذخیره‌سازی مورد بررسی قرار نگیرد، امکان وجود آسیب‌پذیری XSS از نوع Stored وجود دارد. برای آزمایش وجود و یا عدم وجود این آسیب‌پذیری در این بخش از وب سایت، نظر خود را به صورت زیرثبت می‌کنیم:



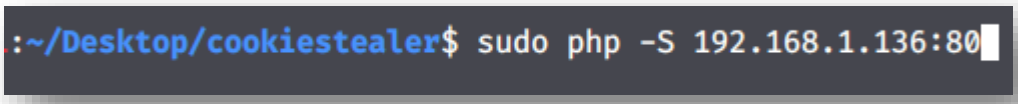
با کلیک کردن برروی Sign Guestbook صفحه بارگیری شده و پیامی با محتوای error! بر روی صفحه نمایش داده می‌شود. در اینجا مشخص می‌شود که کاربرد وب ما دارای آسیب‌پذیری XSS می‌باشد. چرا که ورودی کاربران چک نشده و کاربر می‌تواند قطعه کد جاوااسکریپت خود را در HTML سایت قرار دهد، سپس این که به کاربران نمایش داده شود. همانطور که مشاهده کردید، مهاجم با وارد کردن یک تکه اسکریپت ساده، پیامی را از طریق وب سایت به کاربران نمایش داد. اما مهاجم می‌تواند کارهایی فراتر از نمایش دادن یک پیام، همانند دسترسی به کوکی‌ها انجام دهد. بدین منظور مهاجم دستور زیر را به عنوان نظر وارد می‌کند و دکمه تایید را میزند:



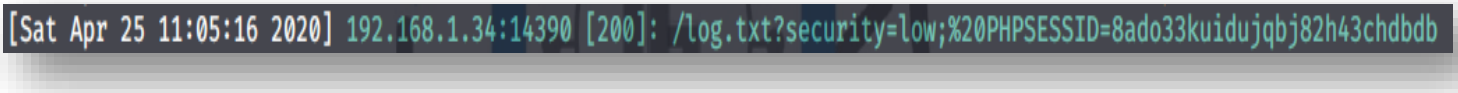
که در آن document.write عمل چاپ کردن را انجام می‌دهد. همچنین document.cookie نیز اطلاعات کوکی را در خود نگهداری می‌کند. بنابراین دستور بالا اطلاعات کوکی کاربر را چاپ خواهد کرد. از این به بعد هر شخصی که وارد این صفحه شود، اطلاعات کوکی او در آن کامنت نمایش داده می‌شود. تنها با کمی گسترش در تکه کد وارد شده، مهاجم می‌تواند این کوکی را برای خود ارسال کند و در ادامه به تلاش برای ربودن Session قربانی بپردازد. بدین منظور مهاجم بعد از راه‌اندازی یک سرور برروی سیستم خود، قطعه کد زیر را در نظرات وارد می‌کند:



که در اینجا آدرس ۱۹۲,۱۶۸,۱,۱۳۶ همان آدرسی است که مهاجم برروی آن یک سرور برای دریافت پیام راه‌اندازی کرده است. Document.cookie نیز کوکی همان لحظه قربانی می‌باشد. برای مثال می‌توان به وسیله دستور زیر یک سرور (این نوع از سرور عموما برای تست می‌باشد) راه‌اندازی کرد:



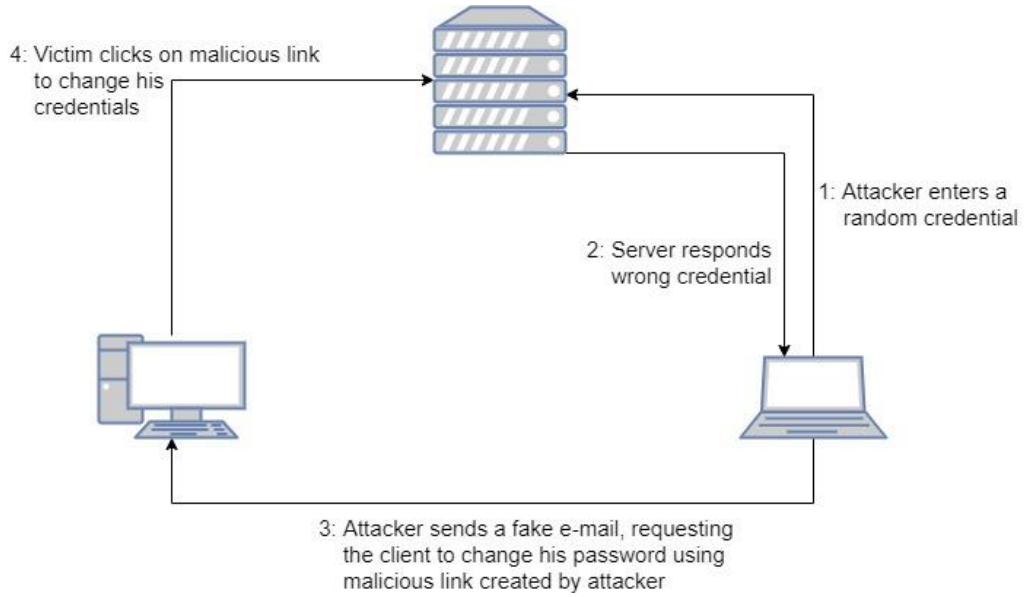
بنابراین برروی پورت ۸۰ یک سرور راه‌اندازی می‌شود. حال با وارد کردن پیام فوق‌الذکر در قسمت نظرات، کافیست قربانی صفحه نظرات را باز کند، در اینصورت، یک پیام حاوی document.cookie به سرور ارسال می‌شود.



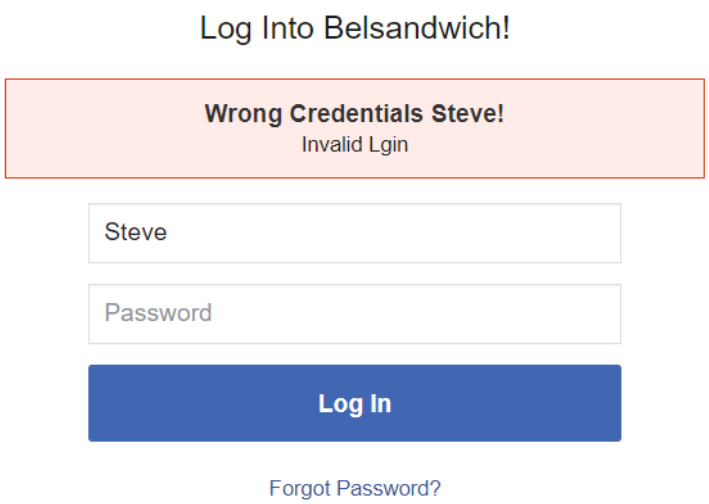
که در اینجا PHPSESSID شامل Session ID قربانی می‌باشد.

۲-۲- حمله XSS(Reflected):

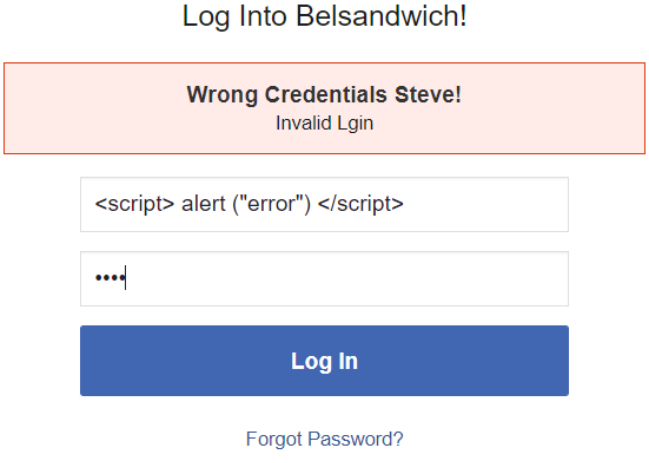
تا به اینجا نوع Stored از حملات XSS را گفتیم که در آن آسیب‌پذیری مربوطه اجازه ذخیره کد جاوااسکریپت در پایگاه‌داده را به مهاجم می‌داد که به هنگام بازدید کاربران از آن بخش سایت، اجرا می‌شد. روش دیگری از حملات XSS برای تزریق کد مخرب جاوااسکریپت وجود دارد که Reflected نام دارد. اگر یک وب سایت قسمتی از درخواست HTTP که کاربر صادر کرده است را دوباره به او نمایش دهد، که این اتفاق می‌تواند باعث بوجود آوردن امکان تزریق کد جاوااسکریپت شود بنابراین احتمال حضور این دسته از XSS در سایت مذکور وجود دارد. برای مثال فرض کنید وب سایت فرضی vuln.com در قسمت جستجوی خود، قسمتی از عبارات دریافتی از کاربر، برای جستجوی کالاها را در نتایج جستجو نمایش می‌دهد. بنابراین این امکان وجود دارد که مهاجم تکه کد مخربی را جستجو کند. در نگاه اول ممکن است این سوال بوجود بیاید که: "اگر کد مخربی اجرا شود، برروی سیستم خود کاربر بوده و برای خود او نیز اعمال خواهد شد چرا که او با سیستم خود یک کد مخرب را جستجو کرده است و نتیجه آن را در سیستم خود نیز می‌بیند."، اما این اتفاق پیش زمینه یک حمله بزرگ‌تر می‌باشد. برای فهم بهتر به مثال زیر توجه کنید:



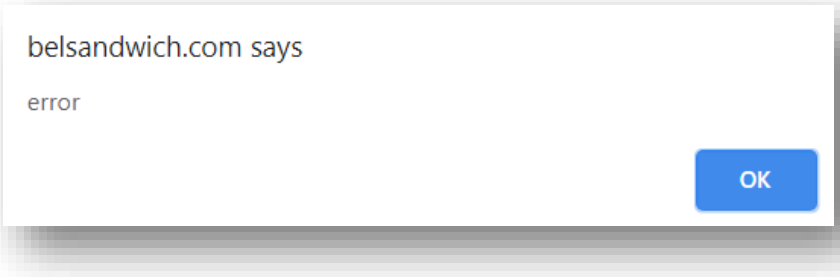
مطابق شکل، ابتدا در مرحله ۱، مهاجم وارد صفحه Login سایت موردنظر می‌شود. در فرم پیش رو، یک نام کاربری و رمزعبور تصـــادفی (همانند: Username: Steve | Password: stevE.A) وارد می‌کند ولی از آن‌جایی که به احتمال خیلی زیاد چنین Credential ای روی سایت تعریف نشده است با صفحه زیر روبرو می‌شود:



اما نکته‌ای که در این لحظه توجه مهاجم را به خود جلب می‌کند این ا ست که سایت ق سمتی از ورودی کاربر را در پاسخ به درخواست او نمایش داد و همانطور که مشخص است Steve که نام کاربری وارد شده توسـط مهاجم بود، در پیام خطا آورده شـده اسـت. حال نفوذگر با مشـاهده این مسـاله و به این امید که ورودی‌های وارد شـده توسـط کاربران به درسـتی فیلتر نمی‌شـود (یعنی وب سـایت هر ورودی را قبول می‌کند. مثل: یک قطعه کد HTML و با یک قطعه کد JS) شـروع به تزریق کد در این قسـمت می‌کند. بدین منظور کد زیر را در قسمت نام کاربری وارد می‌کند:

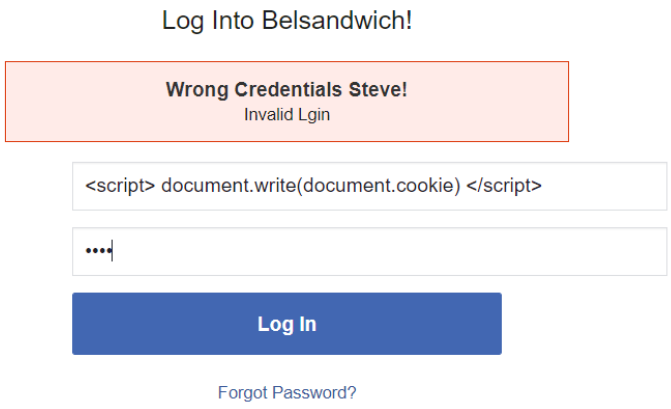


در اینصورت اگر وب‌سایت دارای آسیب‌پذیری باشد، یک پیغام خطا همانند زیر برای مهاجم نمایش داده می‌شود:



توجه کنید که وارد کردن تکه کد اخیر که تنها یک alert می‌باشد فقط برای این منظور است که مهاجم متوجه شود آیا ورودی/خروجی‌ها در این وب‌سایت فیلتر می‌شوند یا هر ورودی از سمت کاربر قابل قبول است. همانطور که مشاهده می‌شود قطعه کد جاوا اسکریپت مهاجم، بدون اینکه چک شود توسط وب‌سایت پردازش و خروجی آن نیز نشان داده شد.

تا به اینجا دیدیم که کدهای جاا سکریپت تو سط سایت پردازش شده و بدون اینکه چک شوند خروجی را به کاربر نمایش می‌دهند. بنابراین به کمک کدهای پیشرفته‌تری به نسبت یک alert ساده، می‌توانیم به اطلاعات مهمی دست پیدا کنیم. برای مثال کد زیر:



کوکی مربوط به کاربر فعلی را نمایش می‌دهد. در نتیجه در مرحله ۲، مهاجم تلاش می‌کند که به طریقی این کد برروی سیستم قربانیان اجرا شده، نتیجه برای مهاجم ارسال گردد. او می‌تواند اینکار را بو سیله

یک لینک انجام دهد. یعنی یک لینک را به دســت کاربران ســایت می‌رســاند و ســپس آن‌ها را متقاعد می‌کند که برروی آن کلیک کنند، نام کاربری و رمز عبور خود را وارد کرده ســپس دکمه تایید را کلیک کنند. پس از کلیک برروی دکمه تایید، اطلاعات وارد شــده برای مهاجم ارســال می‌شــود. البته مطابق با کدی که مهاجم ضمیمه لینک کرده ا ست، اطلاعاتی که به اون می‌ر سد می‌تواند متفاوت با شد. (کوکی کاربر، اطلاعات وارد شده در فرم ورود و ...)