به نام خدا

درس برنامه‌سازی پیشرفته

امنیت شبکه

دانشکده مهندسی کامپیوتر

دانشگاه صنعتی شریف

نیم سال دوم ۹۹ـ۹۸

اساتید:

مهدی مصطفی‌زاده، ایمان عیسی‌زاده، امیر ملک‌زاده، علی چکاه

نگارش و تهیه محتوا:

محسن دهقان کار، صابر ظفرپور

تنظیم داک:

امیرمهدی نامجو، محسن دهقان کار

فهرست

امنیت یعنی چی؟ ۲

شاخه‌های مختلف امنیت ۲

تهدیدات و آسیب پذیری ها و کنترل ها ۲

آسیب پذیری ها و حملات رایج ۴

جزییات آسیب پذیری‌ها ۵

Replay Attacks ۵

Improper Inputs ۵

SQL-Injection ۶

Broken Authentication ۶

Sensitive Data Exposure ۶

Brute force Attack ۷

Broken Access Control ۷

Remote Code Execution ۸

Man In The Middle ۸

پس پروژه چی شد این وسط؟؟ ۱۰

سایر منابع ۱۰

امنیت یعنی چی؟

به طور کلی همه ی ما یک ذهنیت اولیه (احتمالا اشتباه) از امنیت داریم ، در این داک ما قصد داریم یک توضیح مختصر و البته پوشا از موضوعات مرتبط با امنیت در فاز ۳ پروژه این ترم AP به شما عزیزان ارائه دهیم. امنیت در کامپیوتر یا به عبارت دیگر همان امنیت سایبری یکی از شاخه‌های مهم حوزه کامپیوتر است که در عصر ارتباطات و با افزایش تصاعدی ضریب نفوذ کامپیوتر در عرصه‌های مختلف نیاز به آن نیز در حال افزایش است.در دانشکده ما نیز این درس جزو چارت اجباری (ترم ۸) است.

شاخه‌های مختلف امنیت

در این داک ما تنها قصد داریم که امنیت را در سطوح شبکه به صورت محدود بررسی کنیم و در این سطح نیز فقط به موضوعات امنیت نرم افزار در سطح شبکه کفایت می‌کنیم.از دیگر شاخه‌های مهم بحث سایبر سکیوریتی (cyber security) می‌توان به امنیت در حوزه اینترنت اشیا ، امنیت زیرساخت‌های سخت افزاری ، امنیت در چرخه تولید و حیات نرم افزار و ... اشاره کرد.

تهدیدات و آسیب پذیری ها و کنترل ها

در مبحث امنیت ما با تهدیداتی روبرو هستیم ، اما ابتدا باید تهدید را به طور دقیق تعریف کنیم:

تهدید(Threat): یک خطر بالقوه که می‌تواند از یک آسیب پذیری موجود درون سیستم سوء استفاده کند.

به طور مثال: درز اطلاعات بانکی حساب‌های درون فروشگاه شما ـ آسیب‌های فیزیکی به سرورهای فروشگاه شما نظیر سیل و زلزله و حریق

اما در تعریف تهدید به کلمه آسیب پذیری برخوردیم ، پس باید برای آن نیز یک تعریف دقیق ارائه کنیم:

آسیب پذیری(Vulnerability) : ضعفی که درون سیستم وجود دارد و می‌تواند توسط یک عامل (به طور مثال یک هکر) برای رسیدن به اهداف (معمولا خصمانه) استفاده شود.
به طور مثال: آسیب پذیری (Cross Site Scripting XSS) ـ آسیب پذیری عدم رمزگذاری ارتباط کلاینت و سرور و ...
برای مقابله با این آسیب پذیری‌ها می‌توان یک سری راهکار امنیتی پیشنهاد کرد که با اجرای آن می‌توان بسیاری از آسیب پذیری‌های شناخته شده را برطرف کرد.
به طور مثال :(Input Sanitization) یک کنترل بسیار قوی برای جلوگیری از حملات تزریق است.

در ادامه سعی می‌کنیم بسیاری از آسیب پذیری‌هایی که ممکن است در نرم افزار شما وجود داشته باشد را به شما اطلاع داده و راهکار و ابزارهای شناسایی آن و راهکار مناسب برطرف کردن آن و کنترل‌های مناسب برای آن آسیب پذیری را در زبان جاوا در اختیارتان قرار دهیم.

آسیب پذیری ها و حملات رایج

1. Replay attacks*
2. Improper Inputs*
3. Sql Injection* (When using SQL-Based database)
4. Broken Authentication*
5. Sensitive Data Exposure
6. Brute force attack*
7. Broken Access Control
8. Remote Code Execution (RCE)
9. Man In The Middle (MITM)
10. Denial of service* (DOS)

جزییات این آسیب پذیری‌ها در ادامه قرار داده شده است.

خب من کجا می‌تونم این حملاتو تست کنم و قلقشونو بگیرم؟؟

بسیار سوال خوبیه، شما هر چقدر هم در مورد این آسیب پذیری‌ها مطالعه کنید ، در صورتی که این آسیب پذیری‌ها توسط خودتون تست نشه و به نوعی خودتون در جایگاه فرد هکر نباشید، نخواهید توانست که به خوبی نسبت به امنیت نرم افزار خودتون دید خوبی داشته باشید. اما تست کردن این حملات بر روی سایت‌های واقعی میتونه براتون عواقب بدی داشته باشه:)

خب برای این موضوع بهترین راهکار تست کردن این باگ‌ها درون یک محیط ایزوله آزمایشگاهی هست، پس هر وقت که حوصله داشتید برید توی این سایت و نرم افزار

webgoat رو نصب کنید و یک تجربه جذابی از هکر بودن (حداقل توی آزمایشگاه) رو داشته باشید.

جزییات آسیب پذیری‌ها

Replay Attacks

همان طور که اسمش پیداست، این حمله زمانی رخ می‌دهد که یک مهاجم (attacker) پیام‌های رد و بدل شده بین سرور و کلاینت شما را برای خود ذخیره می‌کند(intercepting and capturing packets)، سپس پس از مدتی آن‌ها را دوباره به مقصد می‌فرستد یا حجم زیادی از این پیام‌ها را از طرف خودش به مقصد می‌فرستد. حال اگر دریافت کننده پیام، با این پیام‌های تکراری کاری انجام دهد که مطلوب نیست، replay attack رخ داده است.
برای توضیحات بیش‌تر به این لینک مراجعه کنید.

Improper Inputs

ممکن است به سمت سرور شما هر نوع ورودی فرستاده شود. اگر در سمت سرور (گیرنده پیام) هیچ اعتبار سنجی (validation) و یا اعتبار سنجی ضعیفی انجام شود، ممکن است سبب متوقف شدن سرور مثلا پرتاب شدن exception و یا تغییر در وضعیت سرور شود که مطلوب نیست. هدف این حمله این است که با دادن ورودی‌های نامعتبر به سرور دچار خلل در کار سرور شویم. یا به عبارتی دسترسی پذیری (availability) سرور را از بین ببرد.

همچنین حالت‌هایی را درنظر بگیرید که کلاینت و سرور پیامی را با هم رد و بدل کردند، شخصی در این بین این پیام را شنود کرد (کاری که در شبکه انجام دادنش اصلا سخت نیست برای مثال با استفاده از برنامه ای مثل wireshark). سپس این فرد شنودکننده، پیام را تغییر می‌دهد و برای سرور مجددا می فرستد. نباید با این کار اثر نامطلوبی بر روی عملکرد سرور ایجاد شود.

حالتی دیگر این است که سرور به کلاینت این اجازه را بدهد که نامی با هر طولی وارد کند، بنابراین کلاینت (فرستنده) می‌تواند با نوشتن یک رشته بسیار طولانی، حجم بزرگی از حافظه سرور را اشغال کند و با تکرار این عمل، سرور را کُند کرده و یا از کار بیندازد.

این یک آسیب پذیری کلی، ابتدایی و در عین حال خطرناک است، برای مطالعه بیشتر در مورد آن به این لینک مراجعه کنید.

SQL-Injection

در صورت استفاده از دیتابیس‌های SQL ، این حملات می تواند کاملا دیتابیس شما را پاک کند یا اینکه در آن تغییرات موردنیاز نفوذگر را پیاده کند، از این حملات برای دور زدن صفحات احراز هویت نیز استفاده می‌شود. اما در زبان جاوا راهکارها و کنترل‌های بسیار کارایی برای مقابله با این حملات وجود دارد که شما باید در هنگام نوشتن نرم افزار خود از آن استفاده کنید.

این لینک راهکار عملی رفع این آسیب پذیری را در زبان جاوا نشان می‌دهد. اما پیش از آن بهتر است در مورد این آسیب پذیری مطالعه داشته باشد.

Broken Authentication

هدف اولیه و اصلی حمله کننده (attacker) به سرور، این است که بدون داشتن اطلاعات صحیح (مثلا یوزر و پسورد)، خودش را به جای شخص دیگری جا بزند یا به عبارتی وانمود کند که شخص دیگری است. برای این دسته از کارها او باید احراز هویت سرور را(authentication) دور بزند. برای این کار می‌تواند از روش‌های مختلفی استفاده کند یا در واقع احراز هویت شما ممکن است به اشکال مختلفی فریب بخورد.

اشکال مختلف این حمله و همچنین راه‌های جلوگیری از آن‌ها در این لینک قابل مشاهده است.

Sensitive Data Exposure

برنامه شما اطلاعاتی از کاربران مختلف دارد. این اطلاعات همواره باید محرمانه باقی بمانند.(confidentiality) مثلا در این پروژه، یک خریدار عادی نباید بتواند اطلاعات کاربری مدیر فروشگاه را بفهمد. یا بعنوان مثال دیگر وقتی خریدار وارد اکانت خود می‌شود، نباید اطلاعات مربوط به یوزر و پسورد و یا هرنوع اطلاعات دیگری از او، توسط شخصی که صرفا پیام‌های رد و بدل شده را شنود می‌کند، قابل دیدن و شناسایی باشد.

مثلا به‌عنوان مثال برای اینکه پیام‌های رد و بدل شده شنود نشوند می‌توان از رمزنگاری استفاده کرد. رمزنگاری از رایج‌ترین کنترل‌ها برای حفظ محرمانگی پیام‌های در حال انتقال و همچنین پیام‌های ذخیره شده در حافظه است.

Brute force attack

در صورتی که کاربری قصد داشته باشد برای دور زدن یک فرم در نرم افزار شما (به طور مثال یک فرم ورود به حساب کاربری) تعداد زیادی تلاش ناموفق می‌کند و سعی می‌کند با حدس زدن، ورودی درست را پیدا کند. این کار می‌تواند هم به مانند آسیب پذیری DoS عمل کند و هم می‌تواند باعث به مخاطره افتادن اطلاعات حیاتی نظیر رمز کاربران شود.

ساده‌ترین روش کنترل قرار دادن یک شمارنده برای تعداد تلاش‌های ناموفق هر IP است ، به طور مثال اگر یک IP در کمتر از ۱۰ ثانیه بیش از ۵ (با توجه به نوع برنامه و سرویس شما این عدد متفاوت است) درخواست اشتباه برای سرور فرستاد، باید آن IP توسط سرور در لیست سیاه موقت قرار گیرد و اجازه اتصال به آن به صورت موفق داده نشود.

Broken Access Control

کنترل دسترسی یا access control بخشی از سیستم است که مشخص می‌کند چه کسی چه اجازه‌ای دارد. مثلا در این پروژه، مدیر فروشگاه یکسری دسترسی‌هایی دارد که خریدار عادی ندارد. این دسترسی‌ها باید کنترل شوند. مثلا در سیستم عامل شما، ممکن است

کاربران (user) متفاوتی وجود داشته باشند و یک کاربر به فایل‌های کاربر دیگر دسترسی نداشته باشه و یا مثلا فقط اجازه خواندن از آن‌ها را داشته باشد (و نه نوشتن). کنترل کردن و هندل کردن این دسترسی‌ها و اجازه دادن به یک شخص (subject) که از یک منبع (object) استفاده کند، توسط access control انجام‌می شود.

حال اگر بتوان به راحتی کنترل دسترسی را دور زد و مثلا یک خریدار بتواند خودش را به جای مدیر فروشگاه جا بزند، این یک آسیب پذیری محسوب می‌شود و امنیت را نقض می‌کند.

Remote Code Execution

از جمله آسیب پذیری‌های یک سیستم (بطور خاص یک سرور)، این است که کلاینت (هر کسی که به‌عنوان کاربر برای آن سرور محسوب می‌شود) بتواند کدی مخرب را بر روی آن بفرستد و اجرا کند. نام دیگر آن code injection نیز هست.

به این کار exploit کردن نیز گفته می‌شود و روش‌های بسیار متنوعی دارد. برای مطالعه بیشتر به این لینک مراجعه کنید. البته این حمله و فهمیدن مثال‌های آن نیاز به یکسری پیش زمینه‌هایی دارد که بهترین کار برای افراد علاقه مند جست و جو کردن و رفتن به سمت این مباحث است.

Man In The Middle

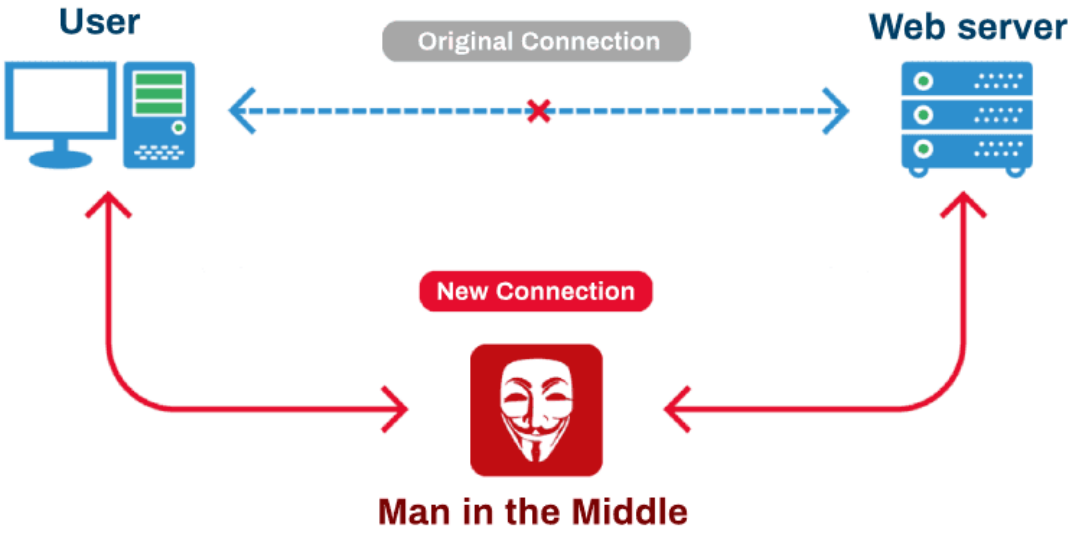
این حمله، حمله بسیار معروفی است که متوجه بسیاری از ارتباطات (communication) است. مثل فرض کنید محسن۱ و محسن۲ می‌خواهند از طریق شبکه با هم صحبت کنند (شبکه را صرفا یک سری کابل در نظر بگیرید که پیام را از یک فرد گرفته و به دیگری می‌رساند). همچنین فرض کنید در این شبکه (یه مشت کابل) یک نفر سومی به اسم صابر، یک کابل را قطع می‌کند و خودش بین این دو سرِ کابل می‌نشیند.

حال محسن۱ می‌خواهد به محسن۲ سلام بگوید. محسن۱ پیام را روی کابل می‌فرستد (نزدیکترین کابل به خودش) و این پیام به راهش ادامه می‌دهد تا این که به کابل قطع شده (صابر) می‌رسد، صابر که آدم ناجوری است(!!) این پیام را از یک سرِ کابل قطع شده گرفته، آن را به یک فحش تبدیل می‌کند و آن پیام را برروی آن یکی سرِ کابل قطع شده به سمت محسن۲ می‌فرستد.

همان طور که دیدید، در واقع صابر پیام خودش را به جای پیام محسن۱ به محسن۲ می‌رساند و بالعکس.

این حمله در تصویر زیر نیز نشان داده شده است و مسلما خطرات زیادی دارد.

برای مطالعه بیش‌تر و دقیق‌تر در مورد این نوع حمله به این لینک و این لینک مراجعه کنید.



Denial of Service (DoS)

طبیعی است که هر نرم افزاری با کاربر خود در ارتباط است و تعدادی پیام با سرور جابجا می‌شود، اما گاهی فرد خرابکار سعی می‌کند با استفاده از یک کد تعداد بسیار زیادی درخواست به سرور شما در مدت زمان کم ارسال کند و با توجه به محدود بودن نرخ پذیرش پیام توسط سرور، باعث مختل شدن کل سرور و از کار افتادن سرویس شما می‌شود، پس شما باید با روش‌های هوشمندانه جلوی این رفتارهای خرابکارانه را بگیرید.

ساده ترین روش کنترل قرار دادن یک شمارنده برای هر IP است ، به طور مثال اگر یک IP در کمتر از ۱۰ ثانیه بیش از ۱۰۰ (با توجه به نوع برنامه و سرویس شما این عدد متفاوت است) درخواست برای سرور فرستاد، باید آن IP توسط سرور در لیست سیاه قرار گیرد و اجازه اتصال به آن داده نشود. (در صورتی که فرد خرابکار از تعداد زیادی IP برای حمله به سرویس شما استفاده کند این راهکار شکست می‌خورد، به آن نوع از حملات انکار، Distributed Denial of Service یا DDoS می‌گویند.)

پس پروژه چی شد این وسط؟؟

آسیب پذیری‌هایی که می‌توانید در پروژه‌تان جلویش را بگیرید و برای نمره‌ی امتیازی چک می‌شوند (در واقع چک کردنشون سخت نیست!)، فقط موارد ستاره دار در لیست بالا هستند. (البته ممکن است موارد دیگری هم در داک مربوط به امتیازات اضافه شود.)

- برای بررسی کردن و نمره دادن به این بخش، از شما یک مستند (یا در کل یک گزارش مکتوب) از کارهایی که برای رفع این آسیب پذیری‌ها و حملات انجام دادید، خواسته می‌شود. این که مستند چطور نوشته شود خیلی اهمیت ندارد اما باید هر کاری که برای جلوگیری از حمله مد نظر انجام دادید را قید کنید و مواردی که نوشته باشید، چک خواهند شد.
- همچنین نیازی نیست جلوی همه‌ی حملات ستاره دار را بگیرید، هر حمله نمره خاص خود را خواهد داشت.

سایر منابع

برای مشاهده آخرین آسیب پذیری‌های ثبت شده، هرگونه اطلاعات آماری و آسیب پذیری‌های مربوط به یک محصول یا شرکت خاص دیتابیس‌هایی موجود هستند و این آسیب پذیری‌ها را ثبت می‌کنند و به آن‌ها شناسنامه می‌دهند. از جمله معروف‌ترین‌ها CVE است.

`https://www.cvedetails.com`

همچنین بنیادی تحت عنوان Open Web Application Security Project یا OWASP با هدف بهبود امنیت نرم افزار وجود دارد و مثلا پروژه‌ای تحت عنوان OWASP Top 10 Web Application Security Risks وجود دارد که هر ساله لیستی از رایج‌ترین ریسک‌ها در زمینه برنامه‌های تحت وب ارائه می‌دهد. (و بطور مشابه پروژه‌ای برای موبایل و ...)

`https://owasp.org/www-project-top-ten`

هم چنین مسابقاتی تحت عنوان CTF یا Capture The Flag برگزار می‌شوند که چالش‌هایی

برای شرکت کنندگان دارند. خوب است در این زمینه هم جست و جویی داشته باشید. این رویداد امسال از طرف دانشکده خودمان هم انجام شد:

`https://susec.tf`