فصل‌نامه تخصصی امنیت سایبری مرکز آپا دانشگاه کردستان
شماره هفتم/ پاییز ۹۹

CVE-2020-1206 | SMBleed

CVE-2020-0688 | Microsoft Exchange

CVE-2020-1472 | Zerologon

CVE-2020-0796 | SMBGhost

- معرفی ابزار Splunk
- معرفی دوره SANS SEC510
- کدنویسی امن در زبان اندروید
- امن‌سازی تجهیزات شرکت Cisco
- کدنویسی امن Upload File در PHP
- نشت اطلاعات و چگونگی جلوگیری از آن
- Session ها، خطرات و تأمین امنیت آن‌ها
- بررسی تخصصی آسیب‌پذیری‌های بحرانی در سال ۲۰۲۰

درباره

مرکز آپا دانشگاه کردستان

آپا مخفف عبارت آگاهی‌رسانی، پشتیبانی و امداد رخدادهای رایانه‌ای است و معادل بومی اصطلاح CSIRT می‌باشد. مرکز آپا دانشگاه کردستان، در راستای انجام فعالیت‌های خود در زمینه آگاهی و اطلاع رسانی، با بکارگیری نیروهای متخصص و پتانسیل‌های پژوهشی در استان کردستان اقدام به انتشار نشریه‌ای الکترونیکی در حوزه امنیت فضای سایبری نموده است.

مخاطبان اصلی نشریه کارشناسان و متخصصان فناوری اطلاعات و شبکه، دانشجویان و علاقمندان فضای سایبری است. مطالب این نشریه عموماً محورهای زیر را شامل می‌شود:

- اطلاع‌رسانی رخدادهای اخیر فضای سایبری
- آگاهی‌رسانی نسبت به آخرین تهدیدات و آسیب‌پذیری ابزارهای فضای مجازی
- آموزش‌های تخصصی و عمومی در جهت ارتقاء دانش امنیت

شایان ذکر است، ویرا، اسم نشریه، واژه‌ای در زبان کردی به معنی صاحب‌فکر و هوشمند است.

صاحب امتیاز: مرکز آپا دانشگاه کردستان
مدیر مسئول: محمد فتحی
سردبیر: هادی گلباغی
سردبیر فنی: محمد حبیبی
ویراستار: نازیلا خسروی
طراحی و صفحه‌آرایی: پرستو مجیدی
نویسندگان (به‌ترتیب مطالب):
محمد حبیبی/ سینا فقیری/ محمد ساروقی
نازیلا خسروی/ هادی گلباغی/ آرش بهرام‌زارعی

تلفن مرکز: ۰۸۷۳۳۶۱۱۴۱۵
نشاني مجله: کردستان، سنندج، بلوار پاسداران، دانشگاه کردستان، دانشکده مهندسی، ساختمان شماره ۳، طبقه همکف، مرکز آپا
وب‌سایت: www.cert.uok.ac.ir
ایمیل: apa@uok.ac.ir

راهنمایی:

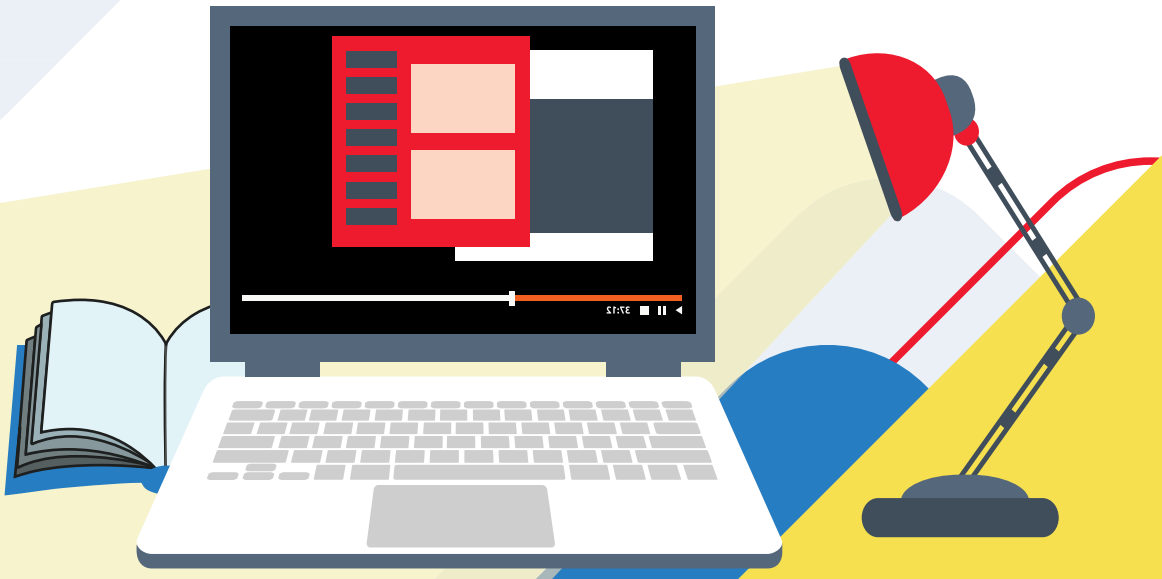
- در فهرست مطالب می‌توانید با کلیک بر روی هریک از بخش‌ها و مطالب به صفحه مورد نظر منتقل شوید.
- با کلیک بر روی QR کدها می‌توانید مستقیماً به لینک‌ها منتقل شوید.

فهرست

مطالب

Tutorials

مقاله‌های آموزشی



بررسی تخصصی سه آسیب‌پذیری بحرانی در سال ۲۰۲۰

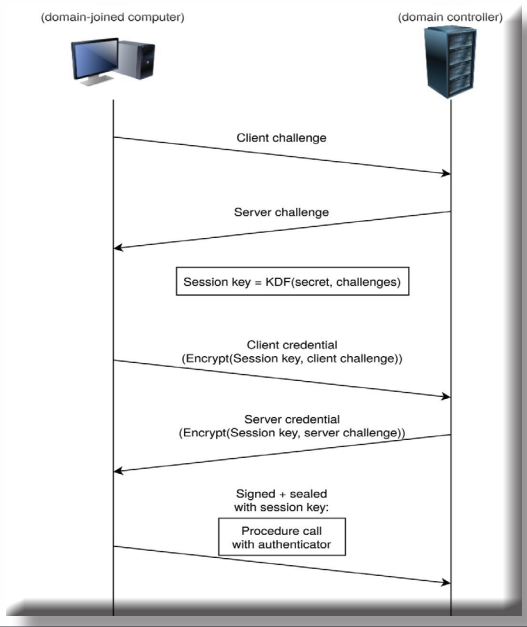
تهیه و تنظیم: محمد حبیبی

آسیب‌پذیری با شناسه CVE-2020-1472

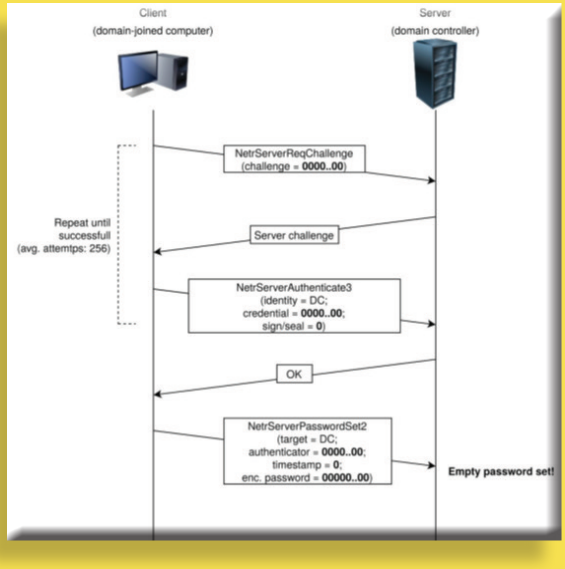
آسیب‌پذیری با شناسه CVE-2020-1472 که به‌عنوان Zerologon شناخته می‌شود، یک آسیب‌پذیری با درجه حساسیت بحرانی است که در کامپوننت Netlogon ویندوز کشف‌شده است. Netlogon یک سرویس برای احراز هویت کاربران در شبکه Domain می‌باشد که وظیفه ایجاد یک کانال ارتباطی امن بین کاربر و Domain Controller را دارد. آسیب‌پذیری Zerologon در آگوست ۲۰۲۰ معرفی شد. این آسیب‌پذیری از نوع Elevation of privilege می‌باشد و مهاجم صرفاً با داشتن یک ارتباط TCP با Domain Controller می‌تواند از آن بهره‌برداری کند و در شبکه به سطح دسترسی Domain admin برسد. برای رفع این آسیب‌پذیری نیاز است وصله‌های امنیتی منتشرشده برای تمام Domain Controller های مربوط به Active Directory اعمال شود. آسیب‌پذیری Zerologon از نقص موجود در رمزنگاری پروتکل احراز هویتی استفاده می‌کند که برای تشخیص اصالت یا احراز هویت میزبان‌های متصل به Domain Controller استفاده‌شده است. دلیل اصلی به وجود آمدن این آسیب‌پذیری استفاده ناامن از پروتکل AES-CFB8 می‌باشد.

پروتکل MS-NRPC یا Microsoft Netlogon Remote Protocol یک سرویس Remote Procedure Call (RPC) است که Domain Controller از آن برای احراز هویت و تصدیق هویت کاربران شبکه دامین استفاده می‌کند. رمزنگاری سفارشی که در این پروتکل استفاده‌شده است، موجب به‌وجود آمدن آسیب‌پذیری Zerologon شده است.

در پروتکل MS-NRPC میزبان عضو دامنه و Domain Controller، مقادیری تصادفی به طول هشت بایت را تولید و تبادل می‌کنند و با ترکیب این دو مقدار و کلمه عبور کاربر شبکه دامین کلید نشست (Session Key) را تولید می‌کنند، سپس مقدار تصادفی خود با کلید نشست شبکه را رمزگذاری کرده و به طرف مقابل می‌فرستد و خروجی این مرحله را با حاصل رمزگذاری کلید نشست و مقدار تصادفی طرف مقابل مقایسه می‌کند، در صورت مشابه بودن دو مقدار، دو طرف برای هم به‌عنوان کاربر و سرور احراز هویت می‌شوند. این فرایند در تصویر زیر آورده شده است.



در پروتکل‌های رمزنگاری به دلیل ارتقاء امنیت، همواره از مقدار تصادفی به‌عنوان Initialization Vector استفاده می‌شود. در پروتکل MS-NRPC نیز از الگوریتم AES-CFB8 و یک Initialization Vector به طول ۱۶ بایت استفاده می‌شود.
در آسیب‌پذیری Zerologon مهاجم هنگام احراز هویت با سرور، فقط بایت‌های صفر یا 0x00 را به‌عنوان Initialization Vector ارسال می‌کند که برابر با NULL است و ضعف موجود در پروتکل باعث می‌شود که مهاجم پس از ۲۵۶ بار ارسال این مقادیرِ صفر به سرور، خروجی تابع رمزنگاری تماماً صفر شود، بنابراین در یکی از این تلاش‌ها ممکن است مقدار ارسالی که همان بایت‌های صفر است با خروجی تابع برابر باشد و کاربر احراز هویت شود.
پس از احراز هویت نیاز است که مشخص شود پیام‌های بعدی رمزگذاری شوند یا خیر که کاربر شبکه دامین که در این سناریو مهاجم است، مشخص می‌کند که برای بقیه پیام‌ها از رمزگذاری استفاده نمی‌کند و بنابراین نیاز است برای ارسال پیام‌های مهم از مقدار Authenticator استفاده کند.
Authenticator با استفاده از رمزگذاری مقدار تصادفی کاربر عضو دامنه، زمان فعلی را به فرمت POSIX و کلید نشست تولید می‌کند. زمان فعلی توسط کاربر تعیین می‌شود، چون محدودیتی در تاریخ زمان فعلی توسط پروتکل تعیین نشده است، مجدداً مهاجم می‌تواند با ارسال بایت صفر به‌عنوان زمان فعلی به سرور، حمله خود را انجام دهد.



بایت صفر در فرمت POSIX برابر با تاریخ,1 January ۱۹۷۰ و ساعت ۱۲:۰۰ می‌باشد، در این مرحله نیز مهاجم می‌تواند مجدد با ارسال بایت‌های صفر به‌عنوان Authenticator مقدار صحیح آن را تولید کند.
لازم است کاربر پس از احراز هویت گذرواژه صحیح را وارد نماید، به این دلیل که برای رمزگذاریِ گذرواژه از الگوریتم AES-CFB8 استفاده‌شده است، مهاجم می‌تواند به ارسال بایت‌های صفر به سرور، به‌عنوان گذرواژه جدید، باعث تغییر گذرواژه کاربر به مقدار صفر یا NULL شود و درنهایت مهاجم با گذرواژه Blank یا Null به‌حساب کاربری که حتی می‌تواند یک Domain Administrator باشد، وارد شود.
در تصویر روبرو نحوه چگونگی حمله و دور زدن احراز هویت نمایش داده‌شده است.

لیست محصولات آسیب‌پذیر:

لیست محصولات آسیب‌پذیر (۲۰۲۰-۱۱-۱۷)
Windows Server 2012 R2 (Server Core installation)
Windows Server 2012 R2
Windows Server 2012 (Server Core installation)
Windows Server 2012
Windows Server 2016
Windows Server 2019
Windows Server 2008 R2 for x64-based Systems Service Pack 1 (Server Core installation)
Windows Server 2008 R2 for x64-based Systems Service Pack 1
Windows Server 2016 (Server Core installation)
Windows Server, version 1903 (Server Core installation)
Windows Server 2019 (Server Core installation)
Windows Server, version 2004 (Server Core installation)

وضعیت آسیب‌پذیری	
کلاس آسیب‌پذیری	Privilege escalation (CWE-269)
درجه حساسیت	بحرانی
نحوه اکسپلویت‌شدن	از راه دور
موجود بودن اکسپلویت	بله
وجود اکسپلویت به‌صورت عمومی	بله
نام نویسنده اثبات آسیب‌پذیری	djrevmoon
لینک اثبات برای تست	https://github.com/SecuraBV/CVE-2020-1472
تاریخ انتشار Advisory مایکروسافت	۲۰۲۰/۱۱/۰۸
تاریخ عمومی شدن اکسپلویت	۲۰۲۰/۱۰/۰۹

آسیب‌پذیری با شناسه CVE-2020-0688

این آسیب‌پذیری دارای درجه حساسیت بحرانی در کامپوننت Exchange Control Panel (ECP) از سرورهای Microsoft Exchange کشف شد که منجر به اجرای کد از راه دور (Remote Code Execution) می‌شود. این آسیب‌پذیری به دلیل عدم پیاده‌سازی صحیح ایجاد کلید رمزنگاری منحصربه‌فرد (Unique keys) در زمان نصب سرور به‌وجود آمده است. مهاجم برای بهره‌برداری از این آسیب‌پذیری به یک Credential (نام کاربری و گذرواژه) نیاز دارد که به پنل ECP دسترسی داشته باشد. درصورتی‌که بتواند به‌صورت موفقیت‌آمیز آسیب‌پذیری را اکسپلویت کند به سطح دسترسی کاربر System در سرور آسیب‌پذیر می‌رسد و بدون هیچ محدودیتی می‌تواند عملیات مخرب خود را انجام دهد.

ماهیت این آسیب‌پذیری ساده است، هنگام نصب سرور به‌جای استفاده از کلیدهای تصادفی برای هر نصب، به‌صورت عمومی Microsoft Exchange Server با یک ValidationKey و DecryptionKey یکسان نصب می‌شود و مقادیر این دو کلید در فایل web.config وجود دارد. این کلیدها برای تأمین امنیت ViewState استفاده می‌شوند. ViewState یک داده سمت سرور است که برنامه‌های کاربردی وب به زبان ASP.NET به شکل سریال‌سازی شده در سمت کاربر ذخیره می‌کنند. کاربر این مقدار را با استفاده از پارامتر __VIEWSTATE مجدداً به سمت سرور ارسال می‌کند. در تصویر زیر بخشی از فایل web.config که حاوی کلید ValidationKey است، نشان داده می‌شود.

```
<system.web>
  <machineKey validationKey="CB2721ABDAF8E9DC516D621D8B8BF13A2C9E8689A25303BF"
decryptionKey="E9D2490BD0075B51D1BA5288514514AF" validation="SHA1"
decryption="3DES" />
  <!--
```

به دلیل استفاده از کلیدهای ثابت و مشترک، مهاجمی که احراز هویت شده است و به ECP دسترسی دارد، می‌تواند یک ViewState که به‌صورت مخرب ساخته‌شده است را برای سرور ارسال کند و سرور آن را به اصطلاح Deserializing کند، چون همان‌طور که پیش‌تر گفته شد مقدار این پارامتر، سریال‌سازی شده است؛ بنابراین از طریق پارامتر ViewState مهاجم می‌تواند یک کد دلخواه .NET. برای سرور ارسال کند و سرور آن را اجرا کند. به دلیل اینکه ECP با سطح دسترسی SYSTEM اجرا می‌شود، مهاجم نیز با اکسپلویت موفق این آسیب‌پذیری به سطح دسترسی SYSTEM بر روی سرور آسیب‌پذیر می‌رسد.
برای اکسپلویت این آسیب‌پذیری ابتدا نیاز است که مقدار ViewStateUserKey و VIEWSTATEGENERATOR را از جلسه کاری یک کاربر احراز هویت شده به‌دست آورد. ViewStateUserKey را می‌توان از کوکی SessionID_ و همچنین مقدار را می‌توان در یکی از فیلدهای مخفی موجود در وب‌سایت، به‌دست آورد که برای این کار می‌توان از Developer tools موجود در مرورگرها استفاده کرد.
برای شروع نیاز است به مسیر ecp/default.aspx/ رفته و سپس وارد سیستم شد. برای انجام اکسپلویت اکانت کاربری که سطح دسترسی خاصی داشته باشد، نیاز نیست. در مرحله بعد مقدار Validationkey را ذخیره می‌کنیم که همان‌طور که گفته شد در فایل web.config موجود است. سپس مقادیر ViewStateUserKey و VIEWSTATEGENERATOR را به هرکدام از روش‌های گفته‌شده در پاراگراف قبلی به دست می‌آوریم. اکنون داده‌های موردنیاز جمع‌آوری‌شده است که به‌عنوان‌مثال می‌تواند مقادیر زیر باشد:

```
--validationkey = CB2721ABDAF8E9DC516D621D8B8BF13A2C9E8689A25303BF
 --validationalg = SHA1
  --generator = B97B4E27
  --viewstateuserkey = 05ae4b41-51e1-4c3a-9241-6b87b169d663
```

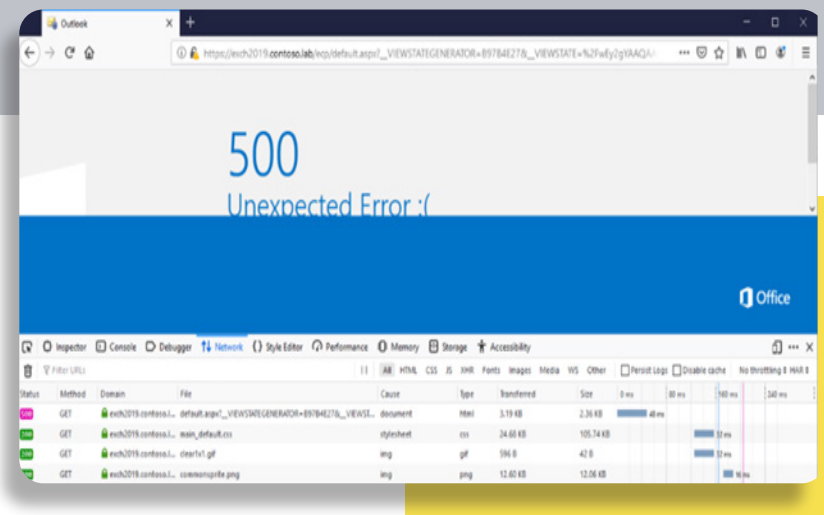
در مرحله بعد نیاز است Payload موردنظر برای قرار گرفتن در پارامتر ViewState را تولید کنیم، برای این امر می‌توان از اسکریپت موجود در Github استفاده کرد. یکی از این اسکریپت‌ها در مخزن زیر قابل‌دسترس است:
https://github.com/pwntester/ysoserial.net
سپس به شکل زیر اسکریپت را اجرا کرده و Payload مورد نظر را تولید می‌کنیم:

```
ysoserial.exe -p ViewState -g TextFormattingRunProperties -c «echo OOOPS!!! > c:/Vuln_Server.
txt» --validationalg=»SHA1» --validationkey=»CB2721ABDAF8E9DC516D621D8B8BF13A2C9E8689A-
25303BF» --generator=»B97B4E27» --viewstateuserkey=»05ae4b41-51e1-4c3a-9241-6b87b169d663»
--isdebug –islegacy
```

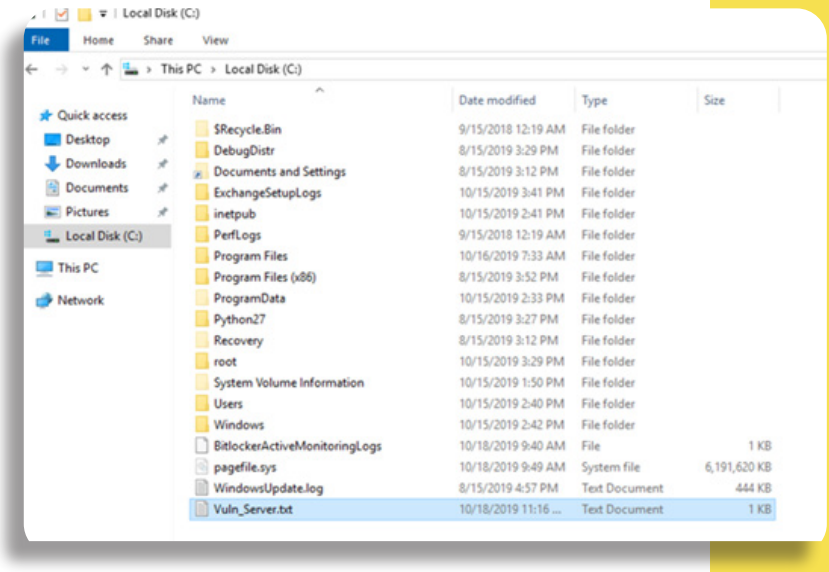
در مرحله بعد خروجی اسکریپت قبلی که همان Payload ما برای انجام اکسپلویت می‌باشد را به‌صورت URL کدگذاری می‌کنیم و خروجی آن را ذخیره می‌کنیم. درنهایت برای اکسپلویت، ما یک URL به شکل زیر داریم:

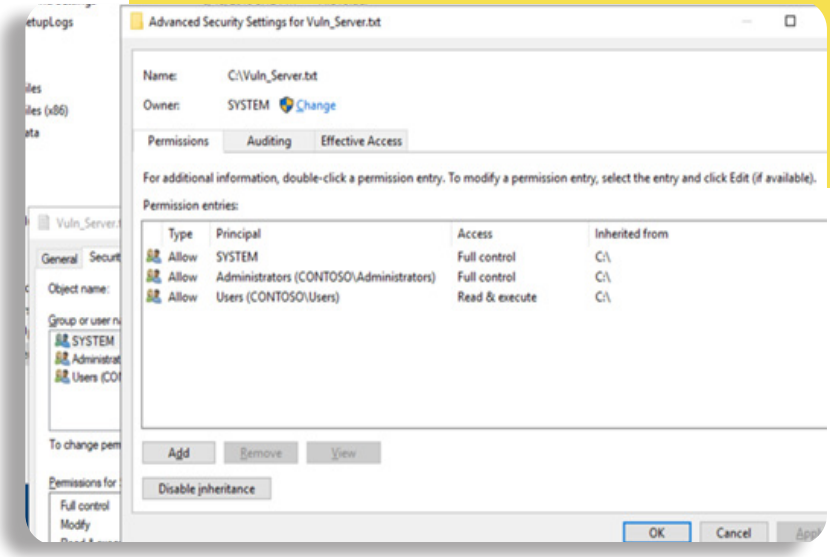
```
/ecp/default.aspx?__VIEWSTATEGENERATOR=<generator>&__VIEWSTATE=<ViewState>
```

سپس مقداری که به‌صورت URL کدگذاری شده است را در URL به دست آمده جایگذاری و به سرور ارسال می‌کنیم. پس از ارسال درخواست همان‌طور که در تصویر زیر مشاهده می‌کنید وب‌سرور به ما پاسخی با کد ۵۰۰ می‌دهد.



اکنون اکسپلویت به شکل صحیح اجراشده است. Paylod ما یک فایل متنی در درایو C می‌سازد، در تصاویر زیر خروجی اکسپلویت مشخص است و فایل با سطح دسترسی System ایجادشده است.





در سناریوهای واقعی ممکن است مهاجم یک شل کد/ فایل آپلودر و درنهایت یک بدافزار بر روی سیستم آپلود کند و به اطلاعات موجود در سرور دسترسی یابد یا آن‌ها را از بین ببرد.

سیستم‌های آسیب‌پذیر:

لیست محصولات آسیب‌پذیر (۲۰۲۰-۱۱-۱۸)
Microsoft Exchange Server 2019 Cumulative Update 3
Microsoft Exchange Server 2016 Cumulative Update 15
Microsoft Exchange Server 2010 Service Pack 3 Update Rollup 30
Microsoft Exchange Server 2019 Cumulative Update 4
Microsoft Exchange Server 2016 Cumulative Update 14
Microsoft Exchange Server 2013 Cumulative Update 23

وضعیت آسیب‌پذیری	
وضعیت آسیب‌پذیری	Deserialization of Untrusted Data (CWE-502)
درجه حساسیت	بحرانی
نحوه اکسپلویت‌شدن	از راه دور
موجود بودن اکسپلویت	بله
وجود اکسپلویت به‌صورت عمومی	بله
نام نویسنده اثبات آسیب‌پذیری	Photubias (Tijl Deneut)
لینک اکسپلویت	https://www.exploit-db.com/exploits/48153
تاریخ انتشار Advisory مایکروسافت	۲۰۲۰/۱۱/۰۲
تاریخ عمومی شدن اکسپلویت	۲۰۲۰/۲۸/۰۲

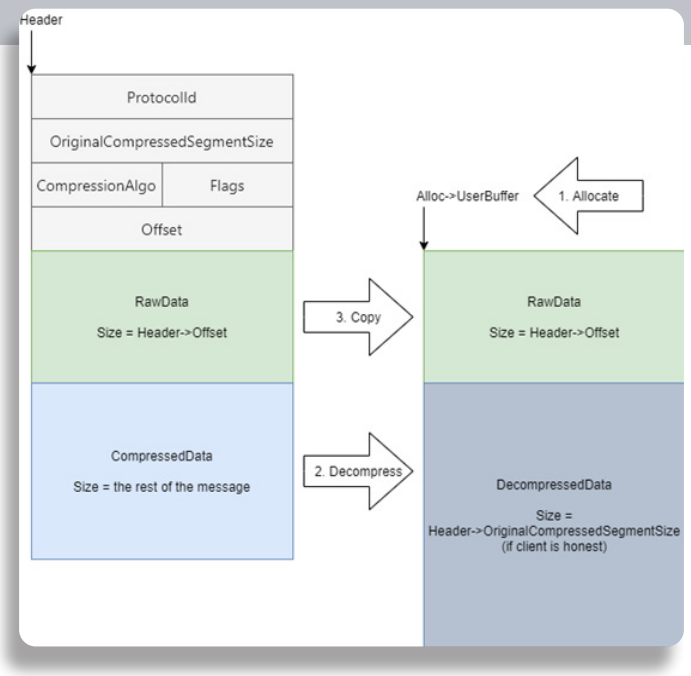
آسیب‌پذیری با شناسه CVE-2020-0796

این آسیب‌پذیری که با نام SMBGhost نیز شناخته می‌شود در سرویس SMBv3 برخی از نسخه‌های کلاینت و سرور ویندوز کشف‌شده است که مایکروسافت برای آن درجه حساسیت بسیار بحرانی تعیین کرده است و به‌صورت محلی و از راه دور از آن بهره‌برداری می‌شود. این آسیب‌پذیری به‌صورت محلی باعث ارتقاء سطح دسترسی کاربر و از راه دور منجر به Remote code execution یا اجرای کد از راه دور می‌شود.

این آسیب‌پذیری از نوع Integer overflow می‌باشد که در تابع Srv2DecompressData در درایور srv2.sys مربوط به SMB Server قرار دارد.
در صفحه بعد نمونه‌ی ساده‌شده‌ای از تابع آورده شده است.

```
typedef struct _COMPRESSION_TRANSFORM_HEADER
{
   ULONG ProtocolId;
   ULONG OriginalCompressedSegmentSize;
   USHORT CompressionAlgorithm;
   USHORT Flags;
   ULONG Offset;
} COMPRESSION_TRANSFORM_HEADER, *PCOMPRESSION_TRANSFORM_HEADER;

typedef struct _ALLOCATION_HEADER
{
   // ...
   PVOID UserBuffer;
   // ...
} ALLOCATION_HEADER, *PALLOCATION_HEADER;

NTSTATUS Srv2DecompressData(PCOMPRESSION_TRANSFORM_HEADER Header, SIZE_T TotalSize)
{
   PALLOCATION_HEADER Alloc = SrvNetAllocateBuffer(
      (ULONG)(Header->OriginalCompressedSegmentSize + Header->Offset),
      NULL);
   If (!Alloc) {
      return STATUS_INSUFFICIENT_RESOURCES;
   }

   ULONG FinalCompressedSize = 0;

   NTSTATUS Status = SmbCompressionDecompress(
      Header->CompressionAlgorithm,
      (PUCHAR)Header + sizeof(COMPRESSION_TRANSFORM_HEADER) + Header->Offset,
      (ULONG)(TotalSize - sizeof(COMPRESSION_TRANSFORM_HEADER) - Header->Offset),
      (PUCHAR)Alloc->UserBuffer + Header->Offset,
      Header->OriginalCompressedSegmentSize,
      &FinalCompressedSize);
   if (Status < 0 || FinalCompressedSize != Header->OriginalCompressedSegmentSize) {
      SrvNetFreeBuffer(Alloc);
      return STATUS_BAD_DATA;
   }

   if (Header->Offset > 0) {
      memcpy(
         Alloc->UserBuffer,
         (PUCHAR)Header + sizeof(COMPRESSION_TRANSFORM_HEADER),
         Header->Offset);
   }

   Srv2ReplaceReceiveBuffer(some_session_handle, Alloc);
   return STATUS_SUCCESS;
}
```

تابع Srv2DecompressData یک پیام فشرده‌شده را که توسط کاربر ارسال‌شده است دریافت می‌کند، سپس حافظه موردنیاز را تخصیص (Memory allocation) می‌دهد، سپس داده‌ها (پیام کاربر) را از حالت فشرده خارج می‌کند. اگر فیلد Offset برابر با صفر نباشد، داده‌هایی را که قبل از داده‌های فشرده قرار داده‌شده است، در ابتدای بافر اختصاص داده‌شده، کپی می‌کند. در تصویر زیر این فرایند را مشاهده می‌کنیم:



در کد روبرو، اگر با دقت به کدهای تابع نگاه کنیم، بخشی از کد که آبی رنگ است، می‌تواند برای ورودی‌های خاصی منجر به Integer overflow شود.

```
PALLOCATION_HEADER Alloc = SrvNetAllocateBuffer(
        (ULONG)(Header->OriginalCompressedSegmentSize +
Header->Offset),
    NULL);
```

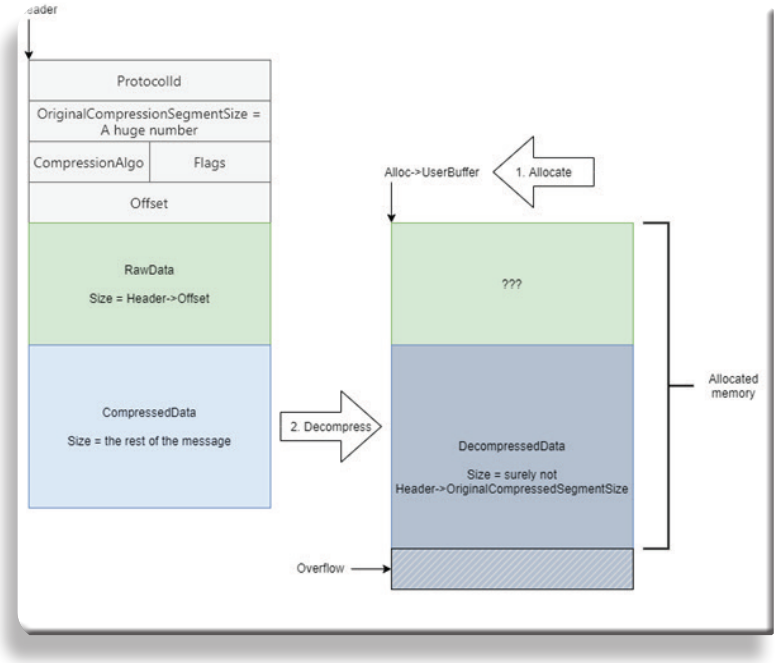
به‌عنوان‌مثال بیشتر اثبات (POC) هایی که کمی پس از انتشار آسیب‌پذیری نوشته‌شده‌اند و باعث کرش کردن سیستم می‌شوند، فقط مقدار xFFFFFFF0 را در فیلد Offset ارسال می‌کنند؛ یعنی در بخش اول کدها صرفاً ارسال مقدار xFFFFFFF0 موجب رخدادن Integer overflow و درنتیجه اختصاص یافتن حافظه کمتر، می‌شود.
بعدها یک Integer overflow دیگر در بخش دیگری از کدها (بخش آبی رنگ کدهای زیر) کشف شد. به تکه کد زیر دقت کنید:

```
NTSTATUS Status = SmbCompressionDecompress(
    Header->CompressionAlgorithm,
    (PUCHAR)Header + sizeof(COMPRESSION_TRANSFORM_HEADER) + Header->Offset,
    (ULONG)(TotalSize - sizeof(COMPRESSION_TRANSFORM_HEADER) - Header->Offset),
    (PUCHAR)Alloc->UserBuffer + Header->Offset,
    Header->OriginalCompressedSegmentSize,
    &FinalCompressedSize);
```

کرش‌کردن به دلیل دسترسی حافظه در آدرسی که در بخش سبز رنگ محاسبه‌شده است، بسیار دورتر از پیام دریافت‌شده رخ می‌دهد. برای ایجاد یک Integer overflow ما فقط دو بخش را برای کنترل کردن داریم، مقادیر OriginalCompressedSegmentSize و Offset، بنابراین پس از بررسی چند ترکیب متفاوت، ترکیب زیر موفقیت‌آمیز بوده‌است.

اگر یک مقدار مجاز برای فیلد Offset به همراه یک مقدار بسیار بزرگ برای OriginalCompressedSegmentSize ارسال کنیم، برنامه به شکل زیر اجرا می‌شود:

۱- مرحله تخصیص حافظه (Allocate): مقدار حافظه تخصیص داده‌شده از مجموع هردو فیلد کمتر است و باعث رخ دادن Integer overflow می‌شود.



۲- مرحله خارج کردن پیام از حالت فشرده (Decompress): در این بخش کد یک مقدار بسیار بزرگ برای فیلد OriginalCompressedSegmentSize دریافت می‌کند سپس با بافر با این فرض رفتار می‌کند که اندازه محدودی دارد. سایر پارامترها تغییری نمی‌کنند بنابراین انتظار می‌رود به شکلی که ما می‌خواهیم فرایند اجرا شود.

۳- مرحله Copy: در این مرحله نیز انتظار می‌رود داده‌ها به شکلی که انتظار داریم کپی شوند.

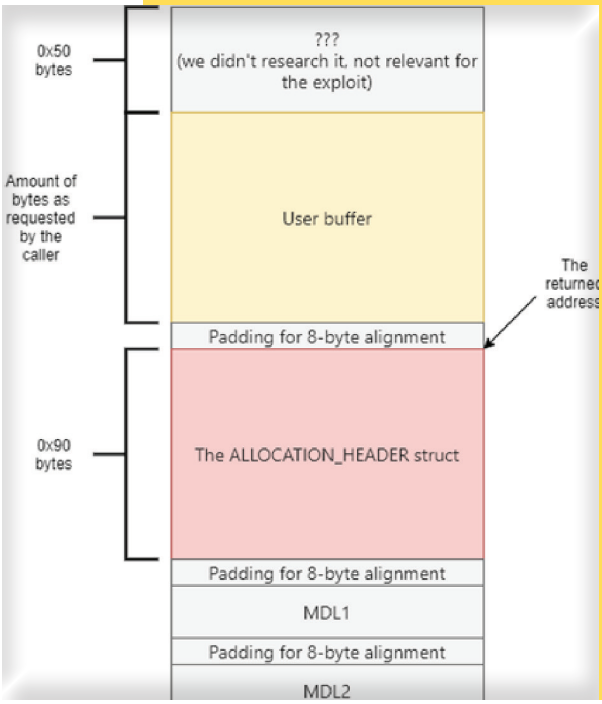
مرحله Copy اجرا شود یا خیر، خروجی کار خوب به نظر می‌رسد. ما می‌توانیم در مرحله Allocate با تخصیص بایت‌های کمتری نسبت به بایت‌هایی که نیاز است، باعث رخ دادن Out of bounds شویم. مراحل گفته‌شده در تصویر روبرو نمایش داده‌شده است.

همان‌طور که مشاهده کردیم استفاده از این روش می‌تواند باعث بروز سرریز حافظه (Overflow) شود. اکنون سؤالی که به‌وجود می‌آید این است که بعد از بافر، چه چیزی در حافظه قرارگرفته و ما در اینجا چه داده‌ای را بازنویسی خواهیم کرد؟! برای پاسخ دادن به این سؤال نیاز است تابع SrvNetAllocateBuffer را بررسی کنیم، این تابع برحسب مقدار بایت‌های موردنیاز عملکرد متفاوتی دارد. برای مثال برای تخصیص حافظه بیش از ۱۶ MB متوقف می‌شود و مقدار NULL را برمی‌گرداند. برای تخصیص حافظه با مقدار متوسط یعنی بیش از ۱ MB و کمتر از ۱۶ MB از تابع SrvNetAllocateBufferFromPool برای تخصیص حافظه استفاده می‌کند.

همچنین برای تخصیص حافظه با اندازه کوچک از لیست‌هایی کنار هم استفاده می‌کند. این لیست‌ها برای رزرو بافرهایی با قابلیت استفاده مجدد و اندازه ثابت در درایور استفاده می‌شوند. برای مطالعه بیشتر در این زمینه به منابع تابع SrvNetBufferLookasides مراجعه شود.

با بررسی این توابع به این نتیجه رسیدیم که درنهایت تمامی تخصیص‌های حافظه به تابع SrvNetAllocateBufferFromPool ختم می‌شوند بنابراین نیاز است که این تابع را بررسی کنیم.

تابع SrvNetAllocateBufferFromPool در NonPagedPoolNx pool با استفاده از تابع ExAllocatePoolWithTag، حافظه موردنظر را تخصیص می‌دهد. ساختار بافر تخصیص داده‌شده را در تصویر روبرو مشاهده می‌کنیم:



تنها بخش مرتبط از این ساختار که بررسی می‌کنیم ALLOCATION_HEADER است زیرا زمانی که ما باعث سرریز در User buffer می‌شویم، بخشی که بازنویسی می‌شود بخش ALLOCATION_HEADER می‌باشد.

نسخه‌هایی از ویندوز ۱۰ و ویندوز سرور که لیست آن‌ها در ادامه آورده شده است، نسبت به SMBGhost آسیب‌پذیرند. برای بهره‌برداری از این آسیب‌پذیری بر روی ویندوز سرور نیاز است پکت حاوی اکسپلویت برای سرور آسیب‌پذیر ارسال شود، اما برای بهره‌برداری از آسیب‌پذیری بر روی ویندوز ۱۰ نیاز است که مهاجم یک سرویس‌دهنده SMB که به‌صورت خاصی پیکربندی‌شده است را اجرا کند، سپس قربانی را ترغیب به اتصال به این سرویس‌دهنده کند، پس از اتصال قربانی، اکسپلویت بر روی سیستم او اجرا می‌شود.

اگر مکانیزمی که موتور جستجوی شودان برای تشخیص این آسیب‌پذیری استفاده می‌کند دقیق باشد، تا زمان نگارش این مطلب می‌توان گفت در حدود ۱۰۳ هزار ماشین آسیب‌پذیر به این نوع آسیب‌پذیری داریم که وصله امنیتی بر روی آن‌ها اعمال نشده است. این آسیب‌پذیری بسیار خطرناک است و می‌تواند توسط هکرهای کلاه‌سیاه، بدافزارهایی که رفتار کرم گونه دارند و خصوصاً باج افزارها، استفاده شود؛ بنابراین نیاز است درصورتی‌که سیستم‌عامل شما در لیست محصولات آسیب‌پذیر وجود دارد سریعاً به‌روزرسانی شود یا وصله امنیتی مرتبط بر روی آن نصب گردد.

لیست محصولات آسیب‌پذیر (۱۸-۱۱-۲۰۲۰)
Windows Server, version 1909 (Server Core installation)
Windows 10 Version 1909 for ARM64-based Systems
Windows 10 Version 1909 for x64-based Systems
Windows 10 Version 1909 for 32-bit Systems
Windows Server, version 1903 (Server Core installation)
Windows 10 Version 1903 for ARM64-based Systems
Windows 10 Version 1903 for x64-based Systems
Windows 10 Version 1903 for 32-bit Systems

وضعیت آسیب‌پذیری	
کلاس آسیب‌پذیری	Improper Input Validation (CWE-20)
درجه حساسیت	بسیار بحرانی
نحوه اکسپلویت‌شدن	به‌صورت Local و Remote
موجود بودن اکسپلویت	بله
وجود اکسپلویت به‌صورت عمومی	بله
نام نویسندگان اکسپلویت	Daniel García Gutiérrez/Manuel Blanco Parajón
لینک اکسپلویت	https://github.com/danigargu/CVE-2020-0796
تاریخ انتشار Advisory مایکروسافت	۲۰۲۰/۱۲/۰۳
تاریخ عمومی شدن اکسپلویت	۲۰۲۰/۳۰/۰۳

کدنویسی امن Upload File در PHP

تهیه و تدوین: سینا فقیری

آپلود فایل در PHP

امروزه آپلود فایل یکی از امکانات اصلی وب‌سایت‌ها به‌حساب می‌آید. گالری تصاویر، فایل ایمیل شده، ویدئوی آنلاین وغیره همه به خاطر وجود امکان آپلود فایل از طریق مرورگر کاربرهاست که تا این حد گسترش پیداکرده است. در بسیاری مواقع برای اهداف مختلفی نیازمند آپلود فایل توسط کاربران مانند ارسال رزومه یا تکمیل پروفایل کاربری وغیره هستیم، در این صورت مواردی هستند که باید محدود و رعایت شوند.

آپلود فایل در PHP کار پیچیده‌ای نیست اما چند مورد که نیازمند توجه هستند، وجود دارند. در غیر این صورت احتمالاً با مشکل مواجه می‌شوید. در این بخش آپلود تصویر در PHP و خطرات و آسیب‌پذیری‌های مربوط به آن را تشریح می‌کنیم. در مرحله اول باید از فعال بودن تنظیمات آپلود فایل در PHP مطمئن شد. برای اطمینان می‌بایست، فایل php.ini بررسی شود که در آن مؤلفه file_uploads برابر on باشد.

```
File_uploads = on
```

برای بخش UI، می‌بایست یک form به‌صورت HTML طراحی شود. باید توجه داشت که enctype برابر multipart/form-data قرار گیرد. این خصیصه، نوع داده ارسالی را مشخص می‌کند.

```
<form action="upload.php" method="post" enctype="multipart/form-data">
    <input type="file" name="fileToUpload" id="fileToUpload">
    <input type="submit" value="Upload Image" name="submit">
</form>
```

فرم بالا، داده‌ها را به سمت فایلی با نام upload.php ارسال می‌نماید. این فرم، دارای یک تگ input از نوع file است که کاربر با کلیک روی دکمه Browse تصویر موردنظر خود را انتخاب و توسط دکمه submit به سمت سرور ارسال می‌کند. فایل upload.php شامل کدهایی است که در صفحه بعد نشان داده خواهد شد.

```
$target_dir = "uploads/";
$extension = pathinfo($_FILES["fileToUpload"]["name"],PATHINFO_EXTENSION);
$target_file = $target_dir.  uniqid() . '-' . time() . '.' . $extension;
$uploadOk = 1;
$imageFileType = strtolower(pathinfo($target_file,PATHINFO_EXTENSION));
if(isset($_POST["submit"])) {
   $check = getimagesize($_FILES["fileToUpload"]["tmp_name"]);
   if($check !== false) {
      echo "File is an image - ".  $check["mime"].  ".";
         $uploadOk = 1;
  } else {
      echo "File is not an image.";
         $uploadOk = 0;
  }
```

• $upload_dir="uploads/": تعیین‌کننده پوشه یا directory محل ذخیره فایل است.
• $upload_file: تعیین‌کننده مسیر ذخیره فایل است.
• $uploadok: مقدار این متغیر، به‌صورت پیش‌فرض ۱ است و اگر در هنگام آپلود فایل خطایی صورت گرفت مقدار آن برابر ۰ می‌شود و سپس پیام اخطار، نمایش داده می‌شود. این متغیر تعیین‌کننده موفقیت‌آمیز بودن یا نبودن آپلود به هر دلیلی می‌باشد.
• $imageFileType: توسط تابع pathinfo پسوند فایل فرستاده‌شده را در خود ذخیره می‌کند.
در فایل upload.php می‌بایست محدودیت‌هایی برای امنیت خاطر بیشتر قرارداد که در بخش‌های جداگانه‌ایی به آن‌ها پرداخته‌شده است.

آسیب‌پذیری‌ها و خطرات

آسیب‌پذیری Directory Traversal in File Upload

این آسیب‌پذیری به زبان ساده به این معناست که هکر می‌تواند فایل دلخواهی را در مسیری جز directory موردنظر برنامه‌نویس وب‌سایت، بارگذاری کند. این آسیب‌پذیری اگرچه در نگاه اول آسیب‌پذیری کم ریسکی به نظر می‌رسد اما تحت شرایطی می‌تواند باعث به وجود آمدن مشکلات بزرگی نظیر Deface یا حتی RCE در وب‌سایت نیز بشود. به‌عنوان‌مثال فرض کنید بتوان فایلی با نام دلخواه در directory دلخواهی قرار دهید، دور از ذهن نیست که با قراردادن فایلی مانند index.html می‌توان وب‌سایت را Deface کرد یا با بارگذاری یک فایل اجرایی روی وب سرور می‌توان امکان اجرای دستورات دلخواه مهاجم روی سرور را فراهم کرد.
این مشکل یکی از آسیب‌پذیری‌های بسیار مهمی است که معمولاً در آپلودرها وجود دارد.

این آسیب‌پذیری معمولاً در داخل پنل‌های مدیریت و کاربران پیدا می‌شود. معمولاً برنامه‌نویس دقتی در فیلترینگ فرآیند آپلودرها ندارد و همین می‌تواند باعث ایجاد این آسیب‌پذیری شود. بسته به نوع سرور، نوع برنامه و سیستم‌عامل می‌تواند نحوه بهره‌برداری از این آسیب‌پذیری متفاوت باشد، اما نتیجه همیشه نتیجه‌ای خطرناک است که امکان کنترل کل یک وب‌سایت را برای هکر فراهم می‌کند. این مشکل یکی از آسیب‌پذیری‌های بسیار مهمی است که معمولاً در آپلودرها وجود دارد.

علت اصلی بروز این آسیب‌پذیری، عدم سنجش یا سنجش ناقص صحت و اعتبار داده‌های ورودی کاربر در تابع بارگذاری فایل است. در اینجا منظور، عدم فیلتر درست نام فایل، نوع فایل و غیره است.

در ادامه به راهکارهایی برای رفع و جلوگیری مشکلات امنیتی آپلود فایل می‌پردازیم.

اعتبارسنجی کاربران و محدودیت آپلود برای اعضا

قبل از هر چیزی، محدود کردن امکان آپلود برای بازدیدکنندگان عادی و اعضا باید در نظر گرفته شود. در هر سایتی اگر فقط اعضای ثبت‌نام‌شده در سایت که قبلاً احراز هویت شده‌اند امکان آپلود فایل را داشته باشند، می‌توان اطمینان خاطر بیشتری ازنظر امنیت آپلود فایل‌ها داشت. همچنین می‌توان برای جلوگیری از خرابکاری و آپلود فایل توسط ربات‌ها از فرآیند Captcha استفاده کرد.

همان‌طور که می‌دانید برای ورود به پنل مدیریت وب‌سایت باید از صفحه لاگین، نام کاربری و کلمه عبور وارد کنید و سپس وارد پنل مدیریت شوید. برای آپلود فایل نیز باید فقط مدیران یا اعضایی که در وب‌سایت لاگین کرده‌اند این امکان را داشته باشند.
می‌توان با روش‌هایی ازجمله فعال بودن session ها یا قرار دادن بخش آپلودر در پنل کاربری این محدودیت را ایجاد کرد.

محدودکردن فایل بر اساس نوع و پسوند آن‌ها

در اسکریپت‌هایی که ارائه شد مقدار متغیر $imageFileType برابر با پسوند فایل ارسال‌شده تعیین شد. با استفاده از همین متغیر می‌توانیم تعیین کنیم که فقط تصاویر با پسوندهای jpg,png,jpeg می‌توانند آپلود شوند.

همچنین می‌توان در کنار محدودیت پسوندهای مجاز، پسوندهای مشکوک مانند فایل‌های اجرایی با پسوند exe. را در محدودیت آپلود قرار داد و با یک شرط و نمایش پیام مناسب از ورود آن‌ها جلوگیری کرد.

```
if($imageFileType != “jpg” && $imageFileType != “png” && $imageFileType != “jpeg” ) {
        echo “Sorry, only JPG, JPEG, PNG & GIF files are allowed.”;
        $uploadOk = 0;
}
```

محدودکردن حجم فایل

یکی دیگر از موارد مهمی که باید بررسی شود، حجم فایل‌های ارسالی است که باید یک سقف مشخص برای آن‌ها قرار داد. مثلاً در این بخش حداکثر حجم مجاز ۵۰۰kb در نظر گرفته‌شده است.
علاوه بر محدودیت حجم، حتماً محدودیت تعداد فایل در یک Request از طریق max_file_uploads نیز باید اعمال شود تا از حملات DOS در امان بود.

```
if ($_FILES[“fileToUpload”][“size”] > 500000) {
        echo “Sorry, your file is too large.”;
        $uploadOk = 0;
}
```

تغییر نام فایل آپلود شده

یکی دیگر از موارد مهم قابل‌بررسی این است که هرگز نباید فایل آپلود شده با نام اصلی خودش در سرور ذخیره شود. می‌توان با استفاده توابع تولید رشته یا عدد تصادفی نام فایل را عوض کرد سپس با نام تصادفی جدید ذخیره کرد. در این صورت هکر برای دسترسی و شناسایی فایل دچار مشکل می‌شود.

بررسی موجود بودن فایل

در این قسمت بررسی می‌شود که اگر فایل وجود دارد، مقداری به نام فایل افزوده شود تا از یکتا بودن نام فایل اطمینان حاصل شده و ذخیره فایل به درستی صورت گیرد. این کار توسط تابع file_exists() بررسی می‌شود و در جهت جلوگیری از جایگزینی با فایل‌های دیگر است.

```
if (file_exists($target_file)) {
    $append = '-' . time();
    $target_file = str_replace('.' . $extension,$append . '.' . $extension, $target_file);
    $uploadOk = 0;
}
```

استفاده از Imagick برای بررسی تصاویر

برای حصول اطمینان از آلوده نبودن تصاویر می‌توان از Imagick به‌صورت زیر استفاده کرد.

```
try {
        $uploadedImage = new Imagick($uploadedFile);
        $attributes = $uploadedImage->identifyImage();
        $format = $image->getImageFormat();
        var_dump($attributes, $format);
} catch (ImagickException $exception) {
}
```

درنهایت پس از کنترل موارد ذکرشده، مقدار متغیر $uploadOk بررسی می‌شود. اگر ۰ بود، یعنی آپلود به دلیلی دچار مشکل شده است و پیام خطای آپلود فایل به کاربر نمایش داده می‌شود.
اگر مقدار $uploadOk، برابر با ۱ بود، توسط تابع move_uploaded_file() فایل موردنظر آپلود و به مسیر در نظر گرفته‌شده منتقل می‌شود.
این تابع دارای دو پارامتر اجباری است. اولین پارامتر، گویای فایل آپلود شده در پوشه tmp سرور و دومین پارامتر نشان‌دهنده‌ی مسیر تعیین‌شده برای آپلود فایل روی سرور است.

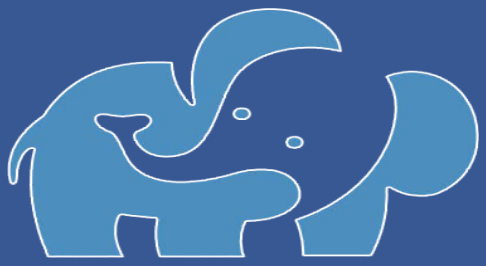
```
if ( $uploadOk == 0 ) {
    echo "Sorry, your file was not uploaded.";
} else {
    if ( move_uploaded_file( $_FILES["fileToUpload"]["tmp_name"], $target_file ) ) {
        echo "The file ".  basename( $_FILES["fileToUpload"]["name"] ).  " has been uploaded.";
    } else {
        echo "Sorry, there was an error uploading your file.";
    }
}
```

نکات تکمیلی و جمع‌بندی

اگر در سایت خود فرمی دارید که از طریق آن فایلی قرار است بر روی سایت آپلود شود، باید دقت ویژه‌ای در بررسی فایل‌های آپلود شده داشته باشید.
قبل از هر چیز باید پسوند فایل‌های مجاز را مشخص نمایید یعنی اگر کاربر قرار است عکسی آپلود نماید حتماً بایستی پسوندهای مجاز را مشخص نمایید. هم در سمت client و هم در سمت server مانند آنچه در قبل گفته‌شد، اعتبارسنجی فایل را قبل از آپلود انجام دهید.
گاهی پیش می‌آید که مهاجم با تغییر در پسوند فایل در صدد دور زدن کدهای اعتبارسنجی برمی‌آید. به‌طور مثال مهاجم فایلی به نام shell.php دارد و آن را به‌صورت shell.php.jpg تغییر نام می‌دهد، در این حالت درصورتی‌که کد اعتبارسنجی شما کامل و کاربردی نباشد، فایل به‌طور موفقیت‌آمیز بر روی سایت شما آپلود می‌شود. البته نسخه‌های جدید apache server نسبت به double extension امن‌سازی شده و این مشکل با دستورالعمل‌هایی رفع شده است.
همیشه سعی کنید فایل آپلود شده را با تغییر نام و پسوندی جدید بر روی سرور قرار دهید که دسترسی مستقیم به آن غیرممکن باشد.

همچنین در سرورهای لینوکس با تغییر سطح دسترسی پوشه‌ی آپلود به ۶۶۶ می‌توانید از اجرای مستقیم فایل‌ها جلوگیری کنید. بهترین راه برای امنیت در این حالت جلوگیری از دسترسی مستقیم کاربران به فایل‌ها می‌باشد که شما می‌توانید پوشه‌ی آپلود خود را در یک سطح بالاتر از ریشه سایت قرار دهید و با استفاده از توابع یا اسکریپت‌های خاص به فایل‌ها دسترسی داشته باشید تا نیازی به آدرس‌دهی مطلق به فایل‌ها نباشد.
برای عکس‌های حساس که صرفاً باید به کاربری خاص نمایش داده شود، فایل تصویر خود را به‌صورت زیر در سایت قرار دهید سپس با استفاده از متد get آن را از دیتابیس فراخوانی نمایید.

```
< img src="/imageload.PHP?id=133"/>
```



session

خطرات و تأمین امنیت آن

تهیه و تدوین: سینا فقیری

مفهوم Session

شما با یک نرم‌افزار در محیط ویندوز کار می‌کنید، به فرایندی که آن را بازکرده و تغییراتی درونش ایجاد می‌کنید و سپس آن را می‌بندید، یک Session می‌گویند. از ابتدای فرآیند تا لحظه‌ای که آن را به اتمام می‌رسانید، ویندوز شما را شناخته و می‌داند که چه کسی هستید اما در دنیای وب و اینترنت اوضاع فرق دارد. در وب‌سرورِ سایتی که در حال بازدید از آن با یک ارتباط HTTP هستید، برای شناخت شما نیاز به ایجاد یک قرار مدت‌دار می‌باشد که با آن توسط وب‌سرور شناسایی شوید. در غیر این صورت با جابجایی بین دو صفحه در یک وب‌سایت، HTTP شما را یک فرد جدید تلقی می‌کند و امکان دارد تغییراتی مانند خریدها و سفارشاتی که اعمال کرده‌اید ناپدید و بازنشانی شود.

وب سرور سایتی که وارد آن می شوید، اطلاعات هویتی شما را که می‌تواند یک توکن یا ID باشد، در متغییرهایی به نام Session ذخیره می‌کند. هر بار که صفحه‌ای از همان سایت را باز می‌کنید، سرور Cookie شما را با Session های موجود مطابقت می‌دهد و در صورت یافتن Session مرتبط با Cookie مرورگر، شما را خواهد شناخت.

تفاوت اصلی Session و Cookie

تفاوت عمده Session با Cookie در این است که Cookie اطلاعات موردنظر را بر روی سیستم کاربر ذخیره می‌کند اما در Session این اطلاعات بر روی سرور ذخیره می‌شود. باید توجه داشت که از Session و Cookie برای هویت‌سنجی کاربران می‌توان استفاده کرد به این شکل که اطلاعات کاربران بر روی Session ذخیره شده و Cookie متناظر آن بر روی سیستم کاربر ذخیره می شود، بنابراین هر Session دارای یک Cookie متناظر می‌باشد. البته Cookie ها به تنهایی نیز جهت ذخیره اطلاعات قابل استفاده هستند اما برای ذخیره اطلاعات حساس استفاده از آنها توصیه نمی‌شود.

ایجاد و استفاده از Session در PHP

یک روش برای در دسترس قرار دادن اطلاعات در صفحات مختلف یک وب‌سایت وجود دارد که آن بهره‌گیری از عملکرد session است. کاری که session انجام می‌دهد، به وجود آوردن یک فایل در دایرکتوری موقت بر روی سرور است، جایی که متغیرهای ثبت‌شده‌ی session و مقادیر آن‌ها نگهداری و ذخیره می‌شوند. این اطلاعات به هنگام بازدید کاربر، در تمام صفحات سایت آماده و در دسترس است.

مکان قرارگیری فایل موقتی، توسط تنظیمات واقع در فایل php.ini که session.save_path خوانده می‌شود، تعیین می‌گردد.

به محض این‌که session شروع به کار می‌کند، اتفاقات زیر رخ می‌دهد:

• ابتدا php یک شناسه‌گر (identifier) منحصربه‌فرد برای آن session معین ایجاد می‌کند که از رشته عددی تصادفی هگزا دسیمال تشکیل‌شده است.

• یک cookie به نام PHPSESSID، خودکار به سیستم کاربر فرستاده می‌شود تا رشته شناسایی منحصربه‌فرد session را ذخیره کند.

• فایلی به‌صورت خودکار در دایرکتوری موقت مشخص‌شده بر روی سرور به وجود می‌آید که اسم شناسه‌ی منحصربه‌فرد دارد و پیشوندش همان شناسه منحصربه‌فرد اختصاص‌یافته است.

هنگامی‌که اسکریپت php بخواهد مقدار را از session variable بازیابی کند، به‌صورت خودکار رشته شناسایی منحصربه‌فرد session را از PHPSESSID cookie دریافت می‌کند و سپس در دایرکتوری موقت به دنبال فایلی که آن نام و عنوان را دارد می‌گردد. فرآیند تایید اعتبار را می‌توان با مقایسه دو متغیر انجام داد و نتیجه‌ی آن را به دست آورد.

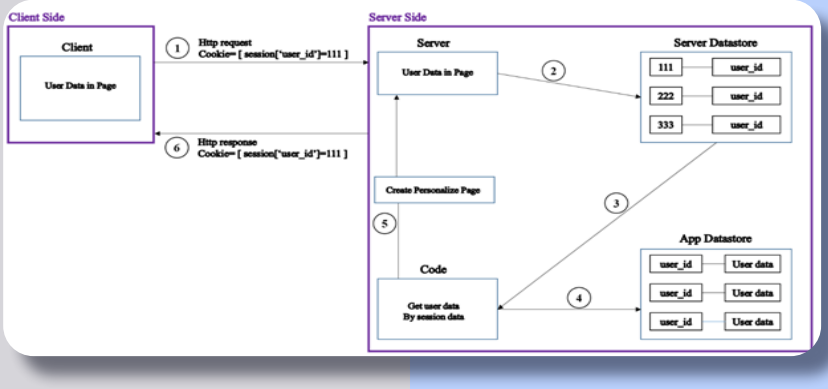
تعریف، حذف و فراخوانی یک session با استفاده از تابعی بنام session_start() صورت می‌گیرد. کلیه متغیرهای session توسط متغیر سراسری $_SESSION نگهداری می‌شوند.

```
<!--?php
        session_start();
        $_SESSION['sessionName']= "SessionValue";
?>
```

با استفاده از تابع session_destroy() می‌توان کل session ها را از بین برد. برای حذف تکی هم از unset() استفاده می‌شود.

نحوه کارکرد session ها

در یک web application که سرور چندین client دارد، user ها چگونه به sessionهای خود دسترسی پیدا می‌کنند؟ در این حالت سرور از کجا متوجه می‌شود که چه session ی متعلق به کدام user است؟ در اینجا نقش session id معلوم می‌شود. روند کلی به این شکل است که شما به‌عنوان یک client، session id خود را به سرور تحویل می‌دهید و در مقابل، سرور درصورتی‌که اطلاعات متناظر با session id شما را درون پایگاه داده sessionهایش پیدا کند، حق دسترسی به آن را به شما خواهد داد. درواقع session id کلیدی برای نمایش این کار است.



خطرات و تهدیدات session ها

حملات ربودن Session (Session hijacking)

در session hijacking attack، یک توکن معتبر از طریق پیش‌بینی و یا سرقت آن مورد تهدید قرار می‌گیرد که این کار به‌منظور به دست آوردن حق دسترسی غیرمجاز به وب‌سایت است. حمله Session Hijacking در دو سطح network-level و Application-level است. در حمله Network-level hijacking اطلاعات مفیدی به دست می‌آید که از آن می‌توان برای انجام یک حمله Application-level hijacking استفاده کرد، بنابراین Network-level و Application-level hijacking در اکثر حالات با یکدیگر رخ می‌دهند. Application-level hijacking شامل در اختیار گرفتن کنترل یک session موجود یا ایجاد یک session جدید بر اساس اطلاعات سرقت شده توسط Network-level hijacking است. Application-level session hijacking از راه‌های مختلفی می‌تواند یک session token را به خطر اندازد که در ادامه به آن‌ها اشاره می‌شود:

Session Sniffing

در این حمله، هکر با شنود ارتباط کاربران با وب‌سرور، http request های رمز نشده که می‌توانند حاوی session ID ها باشند را ضبط کرده و برای به دست آوردن session ID آنالیز می‌کند. پس از آنالیز موفق، خود را در نقش کاربر جعل و session ID را قبل از آنکه کاربر بتواند کاری انجام دهد، به وب‌سرور ارسال می‌کند و به‌راحتی کنترل session موجود را که شامل اطلاعات احراز هویت کاربر نیز هست، به دست می‌گیرد.

Predictable session token

پیش‌بینی session token یا همان session ID شیوه‌ای از ربودن یا جعل هویت یک کاربر وب‌سایت است که پایه و اساس آن بر حدس زدن و یا ایجاد یک مقدار منحصربه‌فرد مثل session ID ی که در احراز هویت یک کاربر، یا یک session خاص کاربرد دارد، می‌باشد. با استفاده از این روش، یک هکر می‌تواند با در دست داشتن اطلاعات احراز هویت و حقوق دسترسی یک کاربر، درخواست‌های ping برای وب‌سایت موردنظر ارسال کند. با مرور روند احراز هویت، هنگامی‌که کاربر درخواست برقراری ارتباط را برای وب‌سایت ارسال کرد، وب‌سایت در ابتدا سعی می‌کند احراز هویت را انجام دهد و هویت کاربر را بررسی کند. تا زمانی که کاربر نتواند هویت خود را برای وب‌سایت اثبات نماید، وب‌سایت اطلاعات درخواست شده را به کاربر تحویل نمی‌دهد. وب‌سایت‌ها معمولاً کاربران را بر اساس ترکیبی از شناسه کاربری و رمز عبور احراز هویت می‌کنند. پس‌ازآن که هویت کاربر برای وب‌سایت محرز شد، وب‌سایت یک session ID منحصربه‌فرد را برای کاربر ایجاد می‌کند. این session ID نشان می‌دهد که session ایجادشده برای کاربر، احراز هویت شده است. به‌منظور تداوم هویت تائید شده کاربر برای وب‌سایت، session ID به ارتباط بعدی بین آن دو نیز الصاق می‌شود. اگر هکر بتواند session ID را با حدس زدن و پیش‌بینی به دست آورد، توانسته session ایجادشده را به خطر بیندازد.

چگونه یک session token پیش‌بینی می‌شود؟

اکثر وب‌سرورها از الگوریتم‌های معمول یا از پیش تعریف‌شده برای ایجاد session ID های خود استفاده می‌کنند. این الگوریتم‌ها ممکن است session ID ها را بر اساس افزایش غیر خودکار اعداد و یا استفاده از روش‌هایی پیچیده‌تر مثل، دخیل نمودن فاکتورهای زمان و دیگر متغیرها، ایجاد نمایند. هنگامی‌که session ID محاسبه شد، در URL، در یک فرم مخفی یا در یک cookie ذخیره خواهد شد. در چنین مواردی، چنانچه هکر بتواند الگوریتم مورداستفاده را به دست آورد، به‌راحتی مدیریت session ID را بر عهده خواهد گرفت. راه‌های ممکن که هکر بتواند حمله را ترتیب دهد شامل:

- ایجاد ارتباط بانرم‌افزار وب و به‌دست آوردن session ID.
- Brute forcing و یا محاسبه session ID بعدی.
- تغییر مقدار فعلی session ID ذخیره‌شده در URL فرم مخفی کوکی با فرض هویت کاربر بعدی.

Man-in-the-middle

این حمله نوعی از حمله است که در آن هکرها در یک ارتباط موجود بین دو سیستم اختلال ایجاد کرده تا زنجیره‌ی پیغام‌های ردوبدل شده را قطع کنند و اطلاعات دلخواه خودشان را در آن تزریق کنند. در این حالت کاربر بر این گمان است که مستقیماً با طرف موردنظرش تبادل اطلاعات دارد اما واقعیت آن است که تمام اطلاعات ردوبدل شده توسط هکر کنترل می‌شود. عملکردهای مختلف این حمله شامل؛ جاسوسی در یک ارتباط، اخلال در یک ارتباط، قطع پیغام‌ها و ویرایش داده‌ها می‌باشد.

Man-in-the-Browser

این حمله شبیه حمله Man-in-the-middle است. تفاوت بین این دو روش آن است که حمله Man-in-the-Browser از Trojan نصب‌شده روی سیستم برای مداخله و دست‌کاری صحبت‌های ردوبدل شده بین مرورگر و مکانیزم امنیتی یا پایگاه داده آن استفاده می‌کند. این روش قابلیت ویرایش و شنود تراکنش‌ها را دارد. مهم‌ترین هدف این حمله سرقت مالی توسط دست‌کاری تراکنش‌ها در بانکداری اینترنتی است.

حملات Client-side

در این نوع حمله هکر سعی می‌کند تا از آسیب‌پذیری‌های موجود در برنامه client نهایت استفاده را برده و از طریق اجبار آن به برقراری ارتباط با سرور آلوده یا اجبار به پردازش اطلاعات آلوده به هدف خود برسد. در حالت اول یعنی هنگامی‌که کاربر با سرور تعامل و ارتباط برقرار می‌کند، شانس زیادی در اجرای این حمله وجود دارد. اگر client با سرور ارتباطی نداشته باشد، آنگاه اطلاعات آلوده نیز نمی‌توانند از طرف سرور به client فرستاده شوند بنابراین اپلیکیشن از حمله و آلودگی در امان می‌ماند. نمونه نرم‌افزاری که در برابر حمله Client-side آسیب‌پذیر است، نرم‌افزارهای پیام‌رسان هستند. هنگامی‌که این اپلیکیشن‌ها اجرا می‌شوند client ها معمولاً بر اساس ساختاربندی قبلی به یک سرورِ ریموت وصل می‌شوند. حملات client-side از سه راه اجرایی می‌شوند:

•حملات Cross-Site scripting (XSS): شیوه‌ای از حملات تزریق (injection) هستند که در آن‌ها اسکریپت‌های آلوده به وب‌سایت‌ها تزریق می‌شوند.

•کدهای آلوده جاوا اسکریپت: هکر ممکن است اسکریپت‌های آلوده‌ی جاوا را در یک webpage جاسازی نماید و شما را به بازدید از آن صفحه فریب دهد. هنگامی‌که شما آن صفحه را در مرورگر خود باز می‌کنید، اسکریپت‌های آلوده به‌صورت مخفیانه و بدون هیچ‌گونه پیغام هشداری اجرا می‌شوند.

•تروجان‌ها: تروجان یک اپلیکیشن ناخواسته است که به ظاهر خود را موجه نشان می‌دهد اما هدف اصلی آن ایجاد دسترسی‌های بی‌مجوز برای هکرها است.

حملات Cross-site Script

حمله تزریق اسکریپت از طریق وبگاه (Cross Site Scripting) که به صورت مخفف XSS نیز نامیده می‌شود، در بین رایج ترین حملات تحت وب قرار دارد. هکر Client-side Script را به صفحات وب تزریق کرده و آن‌ها را برای حملات cross-side script به‌سوی کاربران ارسال می‌کند. درواقع این نوع حمله همان حمله Client-inside است که در آن هکر با ایجاد کد یا برنامه‌های آلوده، Session ID را به خطر می‌اندازد.

راه‌های مقابله با خطرات و امن کردن sessionها

امن کردن در سطح network

• استفاده ازNetwork Level

در این سطح، تمام صحبت‌ها و اقدامات انجام‌شده درباره محافظت از packet ها می‌باشد. مهم‌ترین استراتژی برای رسیدن به هدف این است که تفسیر و تحلیل داده توسط هکر را سخت‌تر کنیم. درواقع اگر هکر نتواند header های داده‌ها را به‌طور صحیح بخواند، قطعاً نخواهد توانست که packet های جعلی را ایجاد و در جریان داده، تزریق کند.

•استفاده از پروتکل‌های انتقال رمزشده

مهم‌ترین اقدام پیشگیرانه که خواندن header ها را برای هکر سخت می‌کند، پیاده‌سازی پروتکل‌های انتقال رمزشده مثل TLS، SSL، IPSec و SSH می‌باشد. در صورت پیاده‌سازی چنین پروتکل‌هایی، هکر برای ربودن session باید از طریق tunnel زدن به یکی از پروتکل‌های بالا کار خود را پیش گیرد که در این صورت حداقل باید session key که از tunnel محافظت می‌کند را بداند و همان‌طور که می‌دانیم، حدس زدن و یا سرقت session key ناشدنی و یا بسیار سخت است.

•استفاده از(IPSec (IP Security

این پروتکل خود مجموعه‌ای از پروتکل‌های زیرمجموعه‌ای است که برای اطمینان از امنیت تبادل packet ها در لایه IP توسعه داده‌شده‌اند. IPSec دو روش رمزنگاری Transport و Tunnel را دارد. در حالت Transport، فقط بخش دیتا در هر packet رمز می‌شود و header آن packet دست‌نخورده و بی هیچ اقدام امنیتی باقی می‌ماند اما در حالت Tunnel هر دو بخش دیتا و هدر رمز می‌شوند. با این اوصاف روش Tunnel ایده‌آل‌تر است چراکه هر دو بخش دیتا و header را باهم رمز می‌کند.

•استفاده از SSL (Secure Sockets Layer)

این پروتکل از فایل‌های مربوط به وب که به صورت محرمانه و از طریق SSL connection ارسال می‌شوند، محافظت می‌کند. البته SSL در لایه Application بیشتر نقش محافظتی خود را ایفا می‌کند چراکه دیتای ارسال‌شده از طریق http session را رمز می‌کند.

•استفاده از (SSH(Secure Shell

این پروتکل شبکه را در برابر IP Spoofing و IP Source Routing که تکنیک‌های رایج در session hijacking هستند، محافظت می‌کند. با پیاده‌سازی این پروتکل، هکر بیشترین کاری که از دستش برمی‌آید، disconnect کردن SSH session است و نمی‌تواند ترافیک را منحرف و یا کانکشن رمز شده را سرقت کند.

امن کردن در سطح Application

نکته کلیدی در محافظت از وب اپلیکیشن در برابر session hijacking، پیاده‌سازی یک ساختار و الگوریتم قدرتمند در ایجاد session id است. مدیریت session چه به‌صورت client side باشد و چه به‌صورت server side، این موضوع هیچ فرقی نمی‌کند و در هر دو حالت صادق است. در ادامه روش‌هایی برای هر چه قوی‌تر شدن session id در وب اپلیکیشن‌ها بیان می‌شود:

•افزایش طول cookie یا session id

طول session id باید به اندازه‌ای باشد که آن را در برابر حملات brute-force که صرفاً به‌منظور حدس زدن و به دست آوردن session id معتبر است، محافظت کند. این افزایش طول باید طوری باشد که هکر تا زمانی که session منقضی نشده، نتواند حتی محدوده‌ای که session id ها بر اساس آن تولید می‌شوند را حدس بزند. بر اساس نوع پردازنده و میزان پهنای باندی که وجود دارد طول session id تعیین می‌شود اما توصیه‌شده است که طول آن کمتر از ۵۰ کاراکتر نباشد.

•تخصیص Session id ها را تصادفی‌تر کنید.

ممکن است که ایجاد session id ها به‌صورت ترتیبی باشد و یا آنکه عدد شاخصی برای یک user به‌عنوان session id در نظر گرفته شود که با تغییر کاربر، آن نیز تغییر کند. در این صورت به‌راحتی می‌توان session id های معتبر را حدس زد بنابراین توصیه می‌شود که ایجاد و تخصیص session id معتبر کاملاً به‌صورت تصادفی و random باشد.

•از یکپارچگی Session id محافظت کنید.

می‌توان به انتهای session id یک کد اصالت هویتی اضافه کرد تا به‌وسیله آن مطمئن شد که session id دست‌کاری نشده است. در این روش درستی کد هویتی در هر درخواستی که سمت سرور می‌رود، بررسی خواهد شد. با انجام این کار دست‌کاری در cookie ها و session id توسط هکر سخت‌تر خواهد شد.

•استفاده از Session idهایی که توسط سرور تامین می‌شوند.

به‌جای آنکه وب اپلیکیشن را مجبور به ایجاد session idهای مخصوص به خودش کرد، می‌توان به‌جای این کار از یک application server ی استفاده کرد که کارش به‌طور خاص تولید session id است. نمونه‌های چنین session id هایی، ASPSESSIONID و JSESSIONID هستند. این session idها نحوه تولیدشان طوری است که کمترین آسیب‌پذیری را دارند.

•از Session id رمز شده استفاده کنید.

با رمز کردن session id ها می‌توان امنیت بیشتری را در قبال آن‌ها فراهم کرد. می‌توان کدی نوشت که هر session token ی را رمز کند و یا آنکه کل session را با استفاده از SSL رمزنگاری کرد.

امنیت وب اپلیکیشن

در این بخش روش‌هایی برای بالا بردن امنیت وب اپلیکیشن را بیان‌شده است:

اعتبارسنجی پارامترهای ورودی

سرور باید اعتبارسنجی نسبتاً دقیقی را از تمام پارامترهای ورودی‌اش که از طرف client ها می‌آیند، انجام دهد. تمام داده‌های موجود در درخواست‌های GET و POST، باید به‌طور دقیق رصد شوند تا به این وسیله شانس‌های موفق حملات HTML Injection و XSS کاهش یابد. باید تمام ورودی‌های کاربر کنترل شوند.

بازه زمانی منقضی شدن Session

Userها معمولاً از سیستم‌های client مشترک استفاده می‌کنند بنابراین این نکته مهم است که بعد از بازه زمانی مشخصی از آخرین فعالیت user، session آن بسته و نامعتبر شود. اگر چنین بازه زمانی برای session در نظر گرفته نشود، هکر برای اجرای حمله Brute force و یا ربودن session زمان نامحدود خواهد داشت.

تولید مجدد توکن

راه دیگری که برای محدود کردن بازه زمانی که هکر حمله brute force را قبل از منقضی شدن session انجام می‌دهد، وجود دارد، تولید مجدد session token در بازه‌های کوتاهی از زمان است. HTTP Server می‌تواند به‌طور یکپارچه، توکن‌ها را منقضی و مجدداً تولید کند تا بدین‌وسیله هکر بازه زمانی بسیار کمی را برای سوءاستفاده کردن هر توکن معتبر داشته باشد.

احراز هویت مجدد

با ایجاد تمایز بین بخش‌های امن و ناامن سایت، می‌توان برای ورود به قسمت‌های حساس سایت، احراز هویت مجدد انجام داد. درنتیجه اگر هکر کنترل session user را به دست آورده باشد، نمی‌تواند فعالیت خاصی انجام دهد و یا به بخش‌های حساس سایت وارد شود. برای مثال در یک اپلیکیشن آنلاین بانکی و یا انتقال پول، این کار باعث می‌شود تا لایه امنیتی بیشتری در مقابل هکر ایجاد شود.

تشخیص حملات Brute Force

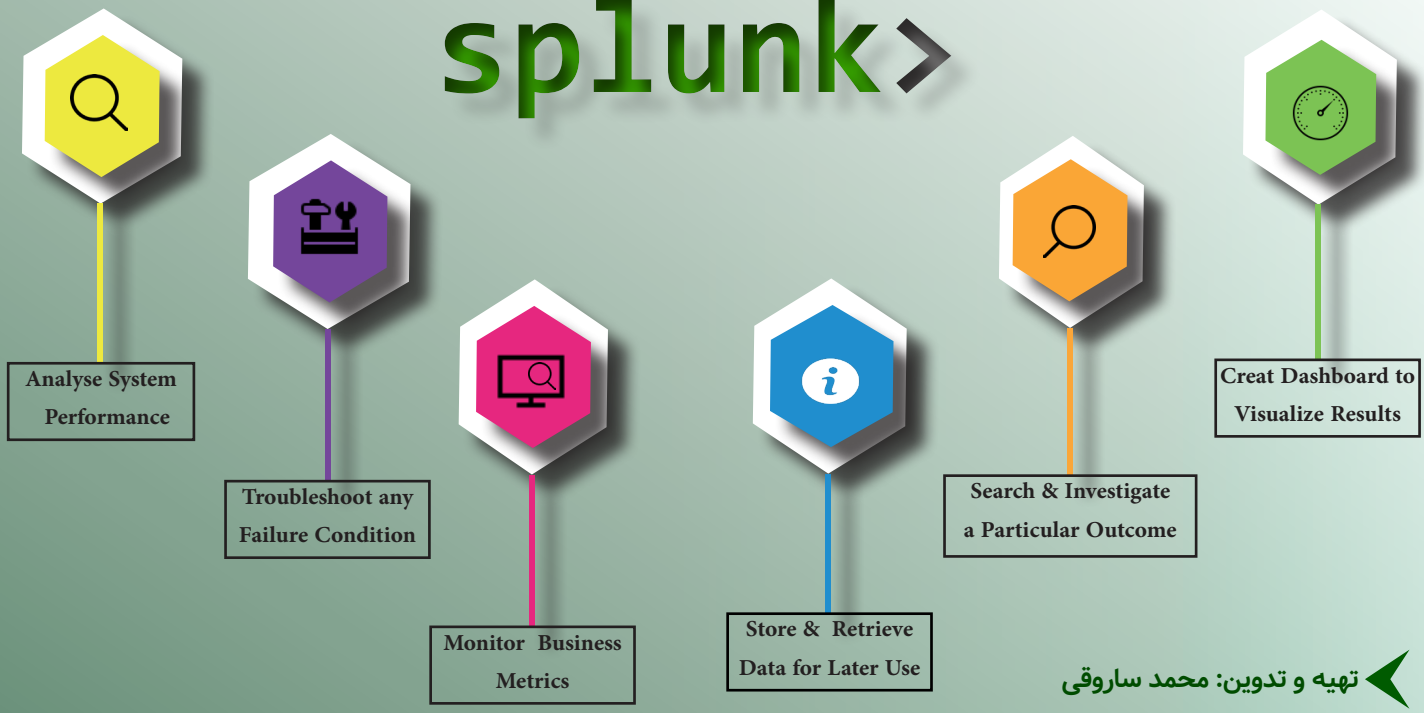
تلاش برای شناسایی brute force session hijacking، قطعاً راه خوبی برای محافظت از sessionها می‌باشد. OWASP پیشنهاد می‌کند از session token هایی که مانند تله‌های انفجاری هستند، استفاده شود، این توکن‌ها هرگز تولید نشده‌اند اما اگر هکری سعی در brute force بازه‌ای از توکن‌ها را داشته باشد، به‌عنوان توکنی معتبر خود را به هکر نشان خواهند داد. وقتی‌که با کمک این توکن‌های قلابی، تلاش برای ربودن session شناسایی شود، می‌توان IP مبدأ را بلاک کرده و یا اکانت موردنظر را lock کرد.

ذخیره در پایگاه داده

به‌صورت پیش‌فرض session ها در فایل ذخیره می‌شوند. تعداد زیادی برنامه وب روی میزبان‌های اشتراکی میزبانی می‌شوند که فایل‌های session های آن‌ها در شاخه /tmp ذخیره می‌شود. اگر این فایل‌ها رمزنگاری نشده باشند، احتمالاً برای کاربران دیگر قابل‌مشاهده خواهد بود بنابراین جایگزینی ذخیره در پایگاه داده می‌تواند روشی قابل‌اطمینان باشد.

Tool Review

معرفی ابزار



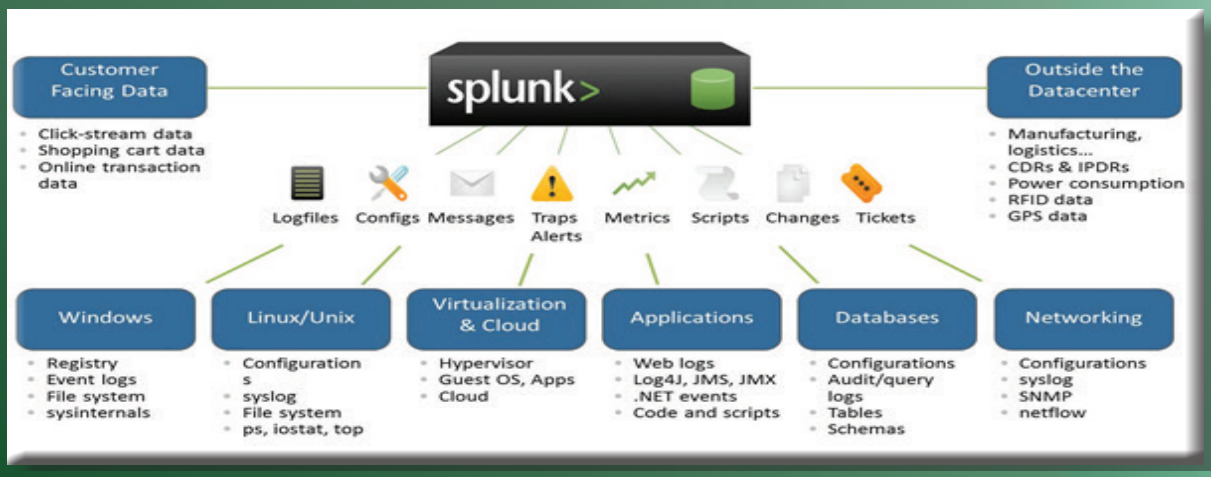
تهیه و تدوین: محمد ساروقی

مقدمه

Splunk یک Security Information and Event Management یا به اختصار SIEM است که به صورت پلتفرمی قدرتمند به‌منظور جمع‌آوری لاگ‌ها، جستجو، مشاهده، آنالیز و تحلیل داده‌ها در سازمان‌ها نصب می‌شود و فعالیت می‌کند. تحلیل اطلاعات از طریق پردازش هزاران داده و بررسی لاگ‌ها انجام می‌شود. به بیانی دیگر، Splunk داده‌های خام را جمع‌آوری و فهرست‌بندی می‌کند و به شما این امکان را می‌دهد که بتوانید بر روی تمام داده‌ها عملیات جستجو را انجام دهید و نتایج را به‌صورت قالب دلخواه مشاهده کنید.

همان‌طور که در تصویر ۱ نشان داده شده است، این پلتفرم به نوع و فرمت لاگ‌ها وابستگی ندارد و برای وارد کردن آن‌ها به Splunk فقط متنی بودن لاگ‌ها کافی است. لاگ‌های وارد شده به Splunk شامل موارد زیر می‌باشند:

۱. لاگ‌های ایجاد شده توسط تجهیزات امنیتی از قبیل IPS,Firewall ,Anti-Virus و غیره
۲. لاگ‌های ایجاد شده توسط تجیزات زیرساخت از قبیل Switch ,Router ,Modem و غیره
۳. لاگ‌های ایجاد شده توسط نرم افزارهای داخلی از قبیل بانکداری، اتوماسیون، مالی، انبار
٤. لاگ‌های ایجاد شده توسط سرویس‌های داخلی از قبیل AD ,DNS ,IIS ,Apache ,DHCP و غیره
۵. لاگ‌های ایجاد شده توسط سیستم عامل‌های مختلف از قبیل Linux، Windows و Mac OS
٦. لاگ‌های ایجاد شده توسط تجهیزات هوشمند و همراه از قبیل گوشی و تبلت
۷. لاگ‌های ایجاد شده توسط تجهیزات الکترونیکی از قبیل درب‌های برقی، آسانسور، حسگرها و کنترل تردد



تصویر ۱. ساختا ر Splunk

SPL یا Search Processing Language زبانی است که با استفاده از آن و قواعد مربوطه می‌توان در میان حجم زیادی از لاگ‌های ذخیره شده در Splunk به جستجو پرداخت و نتایج دلخواه را استخراج کرد.

ضرورت استفاده از Splunk به‌عنوان یک SIEM

بدون بررسی و تحلیل لاگ‌های ثبت شده، امکان تشخیص وقوع حمله بسیار سخت است. لاگ‌ها یکی از مهمترین راه‌ها برای تشخیص هویت مهاجمین می‌باشند، حتی در صورت اطلاع سازمان از وقوع حمله، بدون داشتن لاگ‌هایی جامع و دست‌کاری نشده توسط مهاجمین، سازمان نسبت به جزییات حمله آنچنان دید وسیعی نخواهد داشت.
امروزه بهره‌گیری از یک سامانه متمرکز به‌منظور دریافت، جمع‌آوری و ساختاردادن به این وقایع، می‌تواند در تحلیل سریع و برقراری ارتباط‌های منطقی بین این وقایع گزارش شده، کمک بسیاری نماید.
Splunk Enterprise Security یا به اختصار Splunk ES می‌تواند به عنوان یک راهکار امنیتی مطلوب، به‌صورت On-Premise، Public Cloud، Private Cloud، SaaS یا هر ترکیبی از آن‌ها باشد. همچنین این تکنولوژی را می‌توان به صورت نرم‌افزار همراه با Splunk Enterprise یا به‌عنوان سرویس Cloud همراه با Splunk Cloud استفاده نمود. در ادامه قابلیت‌های Splunk ES به اختصار ذکر می‌گردد:

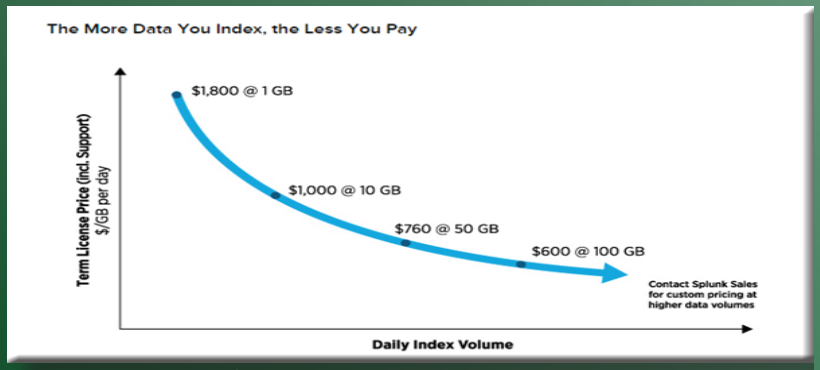
۱. مانیتورینگ مستمر در جهت واکنش نسبت به رویدادها
۲. ایجاد یک مرکز عملیات امنیت SOC
۳. شناسایی سریع حملات داخلی و خارجی و واکنش مقتضی
٤. تسهیل مدیریت تهدید
۵. کاهش ریسک و افزایش محافظت از کسب‌وکار
٦. امکان استفاده از تمام داده‌ها برای تیم امنیتی در تمام سطوح سازمان
۷. ارائه قابلیت‌های جدید برای مدیریت Alertها، قدرت کشف و شناسایی پویا، جستجوهای زمینه‌ای و همچنین شناسایی و تحلیل سریع تهدیدهای پیشرفته
۸. ارائه انعطاف‌پذیری در سفارشی‌سازی جهت بهبود عملکردهایی شامل جستجوها، هشدارها، گزارشات و داشبوردها مطابق با نیازهای خاص؛ برای انجام مانیتورینگ مستمر، پاسخ‌گویی به رویدادها و مرکز عملیات امنیتی SOC
۹. ایجاد امنیت مبتنی بر تجزیه‌وتحلیل و مانیتورینگ مستمر برای تهدیدها
۱۰. بهینه‌سازی عملیات‌های امنیتی با زمان پاسخ‌گویی کوتاه‌تر
۱۱. بهبود وضعیت امنیت با ایجاد دید End-to-End نسبت به تمامی اطلاعات دستگاه‌ها
۱۲. افزایش قابلیت‌های بررسی و شناسایی با هدف شناسایی ناهنجاری‌ها و تهدیدها
۱۳. اتخاذ تصمیمات آگاهانه‌تر نسبت به تهدیدات

برخی از پرکاربردترین افزونه‌های Splunk Enterprise به صورت زیر است:

۱. Cisco Security Suite App
۲. Cisco Networks App
۳. Microsoft Windows App
٤. Stream App F5 Networks App
۵. Web Analytics App
٦. Windows Security Operation Center App

لایسنس Splunk Enterprise

برای استفاده بدون محدودیت از تمام ویژگی‌های Splunk، بایستی لایسنس معتبر تهیه‌گردد که به دو صورت مدت‌دار و نامحدود، ارائه می‌شود. هزینه لایسنس بر اساس حجم داده‌های ایندکس شده، روزانه محاسبه می‌گردد. تصویر ۲ هزینه لایسنس برای داده‌های ایندکس شده روزانه را نشان می‌دهد. در حجم کمتر از ۵۰۰ مگابایت داده ایندکس شده در روز، به‌صورت رایگان می‌توان از Splunk با محدودیت در ویژگی‌ها استفاده کرد.



تصویر ۲. هزینه لایسنس برای داده‌های ایندکس شده روزانه

Cheat Sheet

دفترچه تقلب

دفترچه تقلب SNORT

گردآوری: نازیلا خسروی

Sniffer Mode

شنود بسته‌ها و ارسال خروجی استاندارد به عنوان فایل dump	
-v(verbose)	نمایش خروجی بر روی صفحه
-e	نمایش هدرهای لایه Link
-d	نمایش بسته‌ی کامل با هدرها در فرمت HEX
-x	نمایش data payload بسته‌ها

Packet Logger Mode

ورودی و خروجی برای یک فایل Log	
-R	بازخوانی محتویات یک فایل Log با استفاده از snort
-l (directory name)	ایجاد یک فایل Log با فرمت tcpdump در یک مسیر
-K (ASCII)	نمایش خروجی در قالب ASCII

NIDS Mode

استفاده از فایل مشخص شده به عنوان فایل پیکربندی و اعمال قوانینی برای پردازش بسته‌های ذخیره شده	
-c	تعیین مسیر فایل پیکربندی
-R	تست این مورد که آیا فایل پیکربندی شامل Rule می‌شود یا خیر.

Snort Rules Format

Rule Header + (Rule Options)	
Rule Header + (Rule Options) Action - Protocol - Source/Destination IP -Source/Destination Ports - Direction of the flow	
alert udp !10.1.1.0/24 any ->10.2.0.0/24 any	یک نمونه هشدار
alert, log, pass, activate, dynamic, drop, reject, sdrop	انواع Actions
TCP, UDP, ICMP, IP	انواع پروتکل‌ها

گزینه‌های خط فرمان Logger Mode

-l logdir	ذخیره بسته‌ها در قالب tcpdump
-K ASCII	ذخیره بسته‌ها در قالب ASCII

NIDS Mode Options	
-C (Configuration file name)	تعریف یک فایل پیکربندی
-T –C (Configuration file name)	بررسی صحت ترکیب و فرمت Ruleهای موجود در فایل پیکربندی
-A (Mode : Full, Fast, None ,Console)	حالت‌های هشدار جایگزین
-s	هشدار به SYSLOG
-V	چاپ اطلاعات هشدار
-M (PC name or IP address)	ارسال هشدار SMB به سیستم
-K	Mode ASCII log
-N	بدون ورود به سیستم
-D	اجرا در پس زمینه
-I	شنود یک رابط شبکه خاص

یادآوری

در فصل‌نامه تخصصی امنیت سایبری ویرا شماره چهارم که در زمستان ۹۸ منتشر شده است در یک مطلب ابزار Snort معرفی شد و درخصوص ویژگی‌ها و کارکردهای آن توضیحات کاملی ارائه شده است که توصیه می‌شود این مطلب مطالعه شود.

لینک فصل‌نامه شماره چهارم

Course Description

معرفی دوره

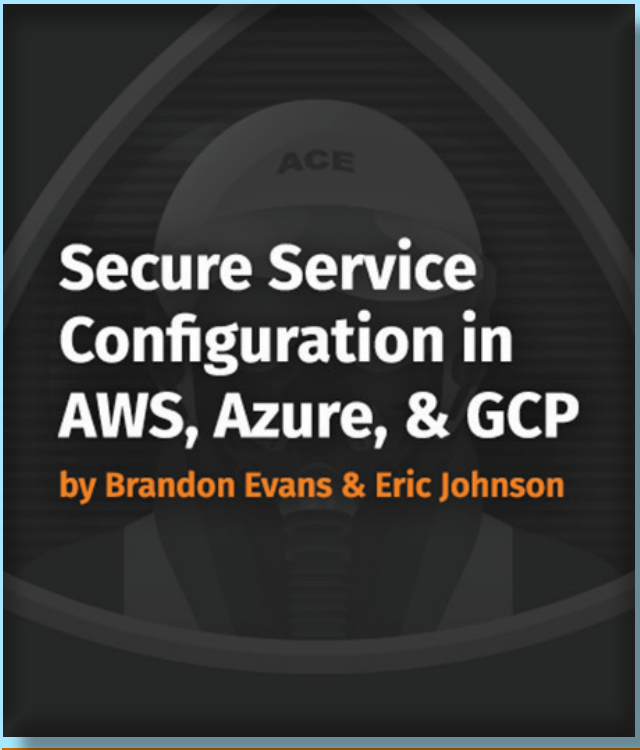
معرفی دوره
SANS SEC510: Public Cloud Security

تهیه و تدوین: محمد حبیبی

این دوره آموزشی به فراگیران و دانشجویان آموزش می‌دهد سیستم‌های ابری امنیتی عمومی شامل Amazon Web Services AWS، Azure و Google Cloud Platform (GCP) به چه شکل عمل می‌کنند و به تحلیل دقیق امنیت در آن‌ها می‌پردازد. دانشجویان و فراگیران پس از این دوره می‌توانند دانش و مهارت کافی برای استفاده از سرویس‌ها یا پلتفرم به عنوان سرویس (Platform as a Service) ارائه شده در هر ساختار ابری را خواهند داشت. در این دوره آموزشی دانشجو یک سیستم ابری ناامن را اجرا می‌کند، سپس پیکربندی‌های امن آن را بررسی کرده و در نهایت مشکلات موجود را رفع و سیستم ابری را امن‌سازی می‌کند. کسانی که تمایل به آشنایی با تست‌نفوذ سیستم‌های ابری را دارند، می‌توانند از دوره SEC588 استفاده کنند.

دوره شامل چه مواردی می‌شود؟

- برنامه‌های آموزشی الکترونیکی
- یک ماشین مجازی شامل تمامی تمرینات مرتبط با درس که می‌تواند خارج از کلاس انجام شود.
- هزاران خط کد زیرساختی برای هر پلتفرم ابری که می‌توان از آن‌ها در هر سازمانی استفاده کرد.
- فایل‌های صوتی ضبط شده برای دوره
- نسخه چاپی کتاب‌های مربوط به موضوع



سرفصل‌های دوره:

فصل اول: مدیریت اعتبارنامه‌ها در سیستم‌های ابری
فصل دوم: شبکه‌های مجازی در سیستم‌های ابری
فصل سوم: رمزگذاری، ذخیره‌سازی و ورود به سیستم
فصل چهارم: پلتفرم‌های بدون سرور (Serverless)
فصل پنجم: Cross-Account و Cross-Cloud Assessment

لینک دوره

Book Suggestion

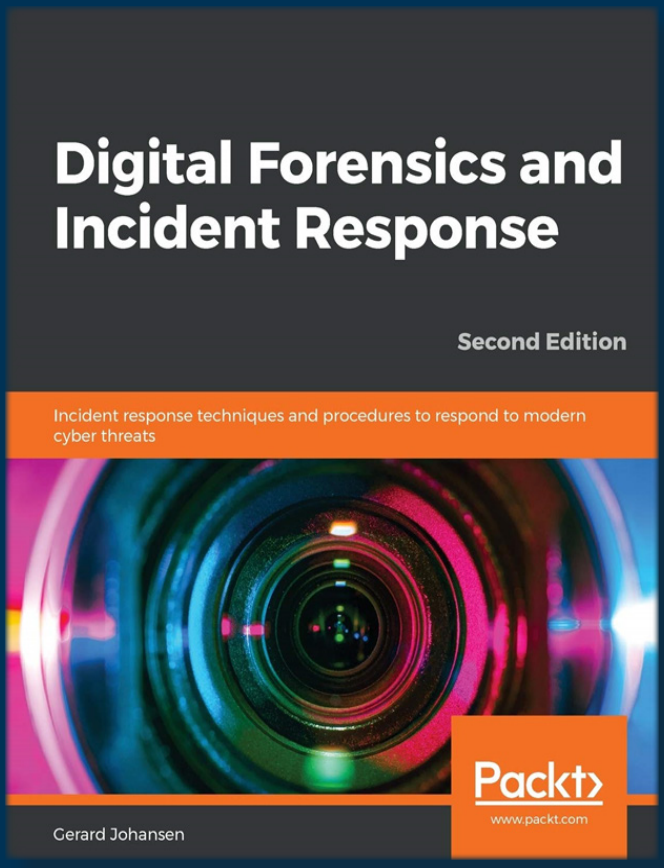
معرفی کتاب

معرفی کتاب

تهیه و تدوین: هادی گلباغی

مشخصات کتاب

نام کتاب:	Digital Forensics and Incident Response
نویسنده :	Gerard Johansen
زبان:	English
تعداد صفحات:	448
ناشر و سال انتشار:	Packt Publishing (January 2020)



ایجاد سیستم دفاع سایبری برای سازمان، با پیاده‌سازی و اجرای کارآمد تکنیک‌های جرم‌شناسی دیجیتال و مدیریت تهدیدها ممکن خواهد شد. در این کتاب، به این نکته پرداخته می‌شود که چه میزان جرم‌شناسی دیجیتال می‌تواند با کارکردهای پاسخگویی به تهدیدات سایبری، به‌منظور امن‌سازی بسترهای سایبری سازمان در مقابل حملات، یکپارچه شود. پس از تمرکز بر اصول پاسخ‌گویی به تهدیدات بحرانی برای تیم امنیت اطلاعات، به سراغ ایجاد چارچوب‌های پاسخگویی به تهدیدات خواهد رفت. در کتاب، راهنمایی‌ها و آموزش‌هایی همراه با مثال‌هایی واقعی و مفید درخصوص پاسخ‌های سریع و موثر به حوادث امنیت سایبری وجود دارد. همچنین تکنیک‌هایی برای افزایش سرعت در جرم‌شناسی دیجیتال در جمع‌آوری اطلاعات اولیه، تحلیل لاگ سیستم، بررسی حافظه، چک‌کردن هارد دیسک و بررسی دقیق بستر شبکه، آورده شده‌است.

در ادامه، روش‌هایی مبتنی بر هوش مصنوعی برای بررسی و پاسخگویی به تهدیدات، توضیح داده می‌شود. همچنین آموزش‌هایی درخصوص ایجاد مستندات و گزارش‌گیری درخصوص بررسی‌ها و تحلیل‌ها در کتاب توضیح داده شده‌است. در پایان علاوه‌بر بررسی فعالیت‌های مختلف پاسخگویی به تهدیدات، درخصوص تحلیل بدافزار و چگونگی بهره‌گیری از مهارت‌های جرم‌شناسی دیجیتال برای کشف فعال تهدیدات، توضیحاتی ارائه گردیده است. همچنین خواهید آموخت که چگونه به صورت کارآمد تهدیدات و خطرات ناخواسته‌ی اتفاق افتاده در سازمان را به خوبی گزارش و مستند کنید.

در این کتاب چه مواردی آموخته می‌شوند؟

- ایجاد و توسعه یک چارچوب برای پاسخگویی به تهدیدات سایبری در سازمان
- بررسی دقیق شواهد و مدارک در حوادث سایبری
- تجزیه و تحلیل اطلاعات جمع‌آوری شده و شناسایی علت‌های اصلی حادثه امنیتی
- بررسی و تحلیل بر روی حافظه و لاگ‌های سیستم
- ادغام و یکپارچه‌سازی تکنیک‌های جرم‌شناسی و فرآیندهای پاسخگویی به تهدیدات
- تکنیک‌های مختلف برای شناسایی تهدیدات
- مستندسازی و گزارش‌نویسی یافته‌ها درخصوص تهدیدات و حوادث سایبری

این کتاب مناسب چه کسانی است؟

- متخصصین حوزه جرم‌شناسی و پاسخگویی به تهدیدات
- متخصصین حوزه قضایی تهدیدات
- متخصصین حوزه امداد حوادث سایبری
- متخصصین تیم‌های آبی و قرمز در سازمان
- محققین حوزه امنیت سایبری علاقه‌مند به یادگیری اصول جرم‌شناسی

سرفصل‌های کتاب

لینک کتاب

Research Papers

مقاله‌های تحقیقاتی

امن سازی تجهیزات شرکت
C I S C o

تهیه و تدوین: محمد ساروقی

مقدمه

در یک شبکه تجهیزات گوناگونی از شرکت‌های مختلف با ویژگی‌های متفاوت وجود دارد که ایمن‌سازی آن‌ها از اهمیت بسیاری برخوردار است. همان‌طوری که می‌دانید شرکت سیسکو جزو شرکت‌های برتر در زمینه تولید محصولات مربوط به شبکه است که جایگاه مناسبی در بازار برای خود کسب کرده است.
در این گفتار سعی داریم در مورد نحوه ایمن‌سازی سیستم‌عامل IOS سیسکو مطالب و دستورات پیکربندی را ارائه نماییم که پیاده‌سازی آن‌ها در IOS می‌تواند تا حد بالایی مفید واقع گردد. پیکربندی‌هایی که در ادامه خواهید دید برای مدیران شبکه، متخصصان امنیتی و افرادی که قصد امن‌سازی شبکه خود را دارند، مفید خواهد بود. کدهایی که برای هرکدام از موارد ذکرشده است به‌عنوان‌مثال بوده و در صورت اعمال بر روی دستگاه خود باید اطلاعات را تصحیح نمایید.

Proxy-ARP

یک تکنیک در روترها می‌باشد که درخواست‌های پروتکل ARP را از خود عبور داده و به این درخواست‌ها پاسخ می‌دهد. با استفاده از Proxy ARP، روتر مسئولیت ارسال بسته به مقصد را عهده‌دار می‌شود، درواقع روتر اجازه عبور پیغام‌های Broadcast از خود را نمی‌دهد و در چنین شرایطی Physical address خود را جایگزین Physical address فرستنده کرده و بسته را ارسال می‌کند.

```
enable
configuration terminal
interface fastEthernet 0/0
no ip proxy-arp
exit
```

Port Filtering

برای محدودکردن دسترسی به پورت‌ها می‌توان برای پورت‌های موردنظر یک Access list تعریف کرد و سپس بر روی Interface موردنظر اعمال شود تا کاربرانی که از این Interface امکان دسترسی به سرویس موردنظر دارند با محدودیت مواجه شوند.

```
enable
configure terminal
ip access-list extended Port-Filtering
deny icmp host 192.168.1.10 host 192.168.2.10
deny tcp any any eq 445
deny tcp any any eq 593
deny tcp any any eq 137
deny udp any any eq 135
exit
interface fastEthernet 0/0
ip access-group Port-Filtering out
exit
```

Port Security

Port Security کنترل می‌کند که چه MAC آدرس‌هایی مجوز استفاده از یک پورت سوئیچ را دارند. این ویژگی به‌صورت اختصاصی بر روی هر پورت اجرا می‌شود. هر کاربر از یک آدرس MAC استفاده می‌کند مگر اینکه از ماشین مجازی استفاده کند در این صورت نیاز به بیشتر از یک آدرس MAC خواهد داشت. درهرصورت، برای جلوگیری از دسترسی دستگاه‌های مختلف به پورت سوئیچ، می‌توان از Port Security استفاده کرد و بر اساس آدرس MAC این محدودیت را برای هر پورت اعمال کرد.

```
enable
configure terminal
interface range fastEthernet 0/1-2,fa0/3
switchport mode access
switchport port-security
switchport port-security mac-address sticky
switchport port-security violation restrict
switchport port-security maximum 1
```

IP Source Guard

ویژگی امنیتی است که می‌توان از جعل IP در شبکه جلوگیری کند. ترافیک لایه دو، پورت‌ها را بر اساس بانک اطلاعاتی که ایجاد کرده‌ایم کنترل و فیلتر می‌کند. این قابلیت برای جلوگیری از حمله‌های IP Spoofing بسیار مناسب است به این صورت که میزبان متصل به آن پورت، تنها امکان استفاده از IP مشخص‌شده را خواهد داشت و هر ترافیکی که با Source IP مشخص نشده، وارد Interface شود Drop خواهد شد.

```
enable

configure terminal

ip source binding 1111.1111.1111 vlan 1 192.168.1.1 interface
fastEthernet 0/1

ip dhcp snooping

ip dhcp snooping vlan 1

interface fastEthernet 0/1

ip verify source port-security

exit
```

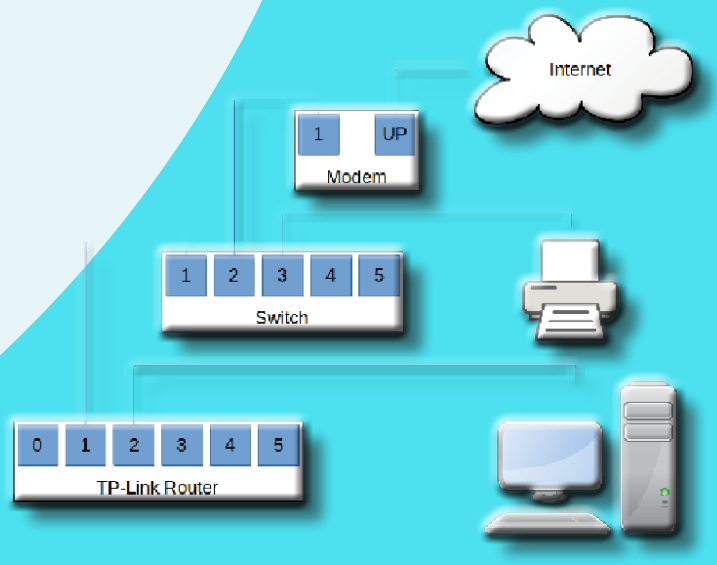
Unicast Reverse Path Forwarding

URPF یا Unicast Reverse Path Forwarding در شبکه‌های Routing برای افزایش امنیت شبکه و جلوگیری از IP Spoofing و بهره‌برداری‌های غیرمجاز است و در روتر هر بسته دریافتی را بدون توجه به Source IP به سمت Destination IP هدایت می‌کند. درواقع با دریافت هر بسته، روتر از خود سؤال می‌کند که مقصد این بسته کجاست؟ Destination IP و آیا می‌توانم آن را Forward کنم. اگر بنا به مسیریابی‌کردن آن بسته باشد، روتر به Routing Table خود نگاه می‌کند و بسته را با توجه به جدول مسیریابی به سمت آدرس مقصد ارسال می‌کند.URPF یک تکنیک امنیتی است برای مقابله با حملات IP Spoofing و همچنین برای افزایش امنیت و جلوگیری از استفاده غیرمجاز از منابع، بدین ترتیب که در حالت کلی با استفاده از URPF هر بسته قبل از مسیریابی شدن به سمت مقصد، ابتدا ازلحاظ Source IP موردبررسی قرار می‌گیرد، اگر به ازای Source IP موردنظر هیچ موردی در Routing Table وجود نداشته باشد، آن بسته Drop می‌شود.

```
enable

configure terminal

ip cef

interface fastEthernet 0/0

ip verify unicast source reachable-via rx

exit
```

VLAN که مخفف Virtual Lan است یک شبکه مجازی بوده که بر روی سوئیچ ایجاد می‌شود. بدین ترتیب که یک شبکه مجازی ایجاد می‌شود که پورت‌های عضو آن دارای یک Broadcast Domain هستند.
برای مثال دو Vlan3 و Vlan2 بر روی یک سوئیچ ایجاد می‌کنیم. تعدادی از پورت‌ها را به Vlan2 و تعدادی را هم به Vlan3 اختصاص می‌دهیم. هرکدام از Vlan ها یک Broadcast Domain جداگانه دارند.

```
enable
configure terminal
vlan 2
exit
vlan 3
exit
interface fastEthernet 0/1
switchport mode access
switchport access vlan 2
exit
interface fastEthernet 0/2
switchport mode access
switchport access vlan 3
exit
```



BPDU Guard

پروتکل Spanning Tree از به وجود آمدن Loop در شبکه جلوگیری می‌کند. این پروتکل برای عملکرد خود از بسته‌های تحت عنوان BPDU استفاده می‌کند تا بتواند یک شبکه بدون Loop ایجاد کند و یک ساختار درختی برای شبکه ما ایجاد کند. برای محافظت از این ساختار و درختی که به وجود آمده است باید مکانیزم‌های امنیتی را برای آن در نظر بگیریم که یکی از آن‌ها BPDU Guard است. زمانی که BPDU Guard را فعال می‌کنیم اگر بر روی پورت سوئیچ BPDU دریافت شود، ارسال روی این پورت متوقف می‌شود و پورت غیرفعال می‌شود. کاربرهای نهایی که به پورت‌های سوئیچ متصل هستند نباید بر روی این پورت BPDU ارسال کنند. این تنظیمات باید بر روی پورت‌های Access که به دستگاه‌های نهایی متصل هستند اعمال شود و از اتصال سوئیچ غیرمجاز به شبکه جلوگیری کند. با این کار می‌توان ایجاد تغییرات در توپولوژی STP را کنترل کرد.

```
enable
configure terminal
interface fastEthernet 0/5
spanning-tree bpduguard enable
exit
```

Root Guard

مکانیزم دیگری که برای حفاظت از پروتکل Spanning tree وجود دارد، Root Guard است که در آن از تغییر Root switch جلوگیری می‌شود. ممکن است که سوئیچ شما به سوئیچ‌های دیگری متصل باشد و شما آن‌ها را تنظیم نکرده باشید. اگر بخواهید این سوئیچ از طریق این پورت‌ها Root Switch جدیدی را شناسایی نکند می‌توانید از Root Guard برای این پورت استفاده کنید. نحوه‌ی عملکرد Root Guard به این صورت است که اگر بر روی پورت Superior BPDU دریافت کند، پورت بلاک شده و در وضعیت Root-inconsistent درمی‌آید و تا زمانی که Superior BPDU دریافت کند در این وضعیت خواهد ماند.

```
enable
configure terminal
spanning-tree vlan 1 priority 8
interface fastEthernet 0/4
spanning-tree guard root
exit
```

DHCP Snooping

Rogue DHCP سرورها DHCP غیرمجازی هستند که باعث مختل شدن فعالیت شبکه می‌شوند. برای جلوگیری از عملکرد DHCP سرورهای غیرمجاز در شبکه، در سوئیچ‌ها قابلیتی به‌عنوان DHCP Snooping ارائه‌شده است. ویژگی DHCP Snooping درواقع یک فایروال Logical در بین کلاینت‌های غیرقابل‌اعتماد و DHCP سرورها ایجاد می‌کند. سوئیچ بعد از فعال شدن قابلیت DHCP Snooping یک جدول یا Table ایجاد و نگهداری می‌کند که به DHCP Snooping Table معروف است. سوئیچ از این جدول برای شناسایی و فیلترکردن پیام‌های غیرقابل‌اعتماد در شبکه استفاده می‌کند، سوئیچ از این جدول برای شناسایی DHCP سرورهای مجاز و قابل‌اعتماد و همچنین DHCP سرورهای غیرمجاز استفاده می‌کند، زمانی که در این جدول یک DHCP سرور به‌عنوان مورد اعتماد یا Trusted معرفی

شد آدرس MAC آن به همراه مبدأ ارسالی آن در این جدول ثبت می‌شود، اگر پیام DHCP سروری از سایر پورت‌های شبکه دریافت شود که برابر آدرس MAC و مبدأ ثبت‌شده در این جدول نباشد، سوئیچ آن پیام و سرویس را مسدود می‌کند، یا بهتر بگوییم اگر Packet ای از Untrusted Port ها وارد سوئیچ شود و فرآیند DHCP را داشته باشد Drop خواهد شد.

```
enable
configure terminal
ip dhcp snooping
ip dhcp snooping vlan1
interface fa 0/1
ip dhcp snooping trust
ip dhcp snooping limit rate 10
```

قطع ارتباط به‌صورت خودکار

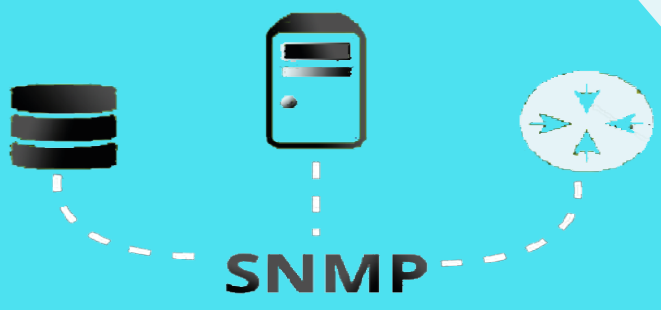
به‌منظور جلوگیري از سوءاستفاده از اتصالاتی که توسط کاربر رها شده‌اند، این امکان وجود دارد که پس از طی مدت‌زمان مشخصی که هیچ ورودي از طرف کاربر به دستگاه ارسال نشود اتصال او قطع گردد. به‌عنوان‌مثال اگر کاربري با سطح دسترسی مدیریتی سیستم خود را پس از اتصال به دستگاه رها کند، امکان سوءاستفاده از این بستر وجود دارد. انتخاب مدت‌زمان براي این امر باید تعادل مابین مسائل امنیتی و امور کاری و پیکربندي که زمان بیشتري لازم دارد را برقرار کند.

```
enable
configure terminal
line vty 0 4
exec-timeout 30 0
```

SNMP

پروتکل SNMP یا Simple Network Management Protocol پروتکلی است که برای مانیتورینگ و پایش دستگاه‌های متصل به شبکه به‌صورت از راه دور استفاده می‌شود. نیاز است که پروتکل SNMP در کلاینت نیز وجود داشته باشد تا SNMP Server یا SNMP Manager بتوانند آن‌ها را مدیریت و مانیتور کند. با استفاده از SNMP مدیر شبکه می‌تواند از وضعیت کلاینت‌ها باخبر شود و تقریباً تمام فعالیت‌های شبکه‌ای مربوط به آن را رصد و مدیریت کند. همانند سایر پروتکل‌ها، پروتکل SNMP نیز دارای نسخه‌های مختلفی است، نکته‌ای که باید موردتوجه قرار گیرد این است که SNMP v1 ارتباطش را رمزنگاری نمی‌کند اما در SNMP v2 ارتباطات SNMP بین کلاینت و سرور رمزنگاری می‌شود. برای استفاده از پروتکل SNMP برای مانیتورینگ کلاینت‌ها شما باید SNMP Agent را بر روی سرور نصب کنید، این Agent که یک نرم‌افزار است به تمام SNMP Client ها متصل می‌شود و سپس شما می‌توانید در سرورتان تمام کلاینت‌ها را مورد رصد قرار دهید. نرم‌افزاری که برای مانیتورینگ کلاینت‌های شبکه توسط پروتکل SNMP استفاده می‌شود، می‌تواند اطلاعاتی نظیر نوع دستگاهی که مانیتور می‌کنید، آدرس IP آن دستگاه، مقدار فضای خالی دیسک آن کلاینت، فایل‌های بازشده توسط کلاینت، آمار مربوط به شبکه‌ای که در آن قرار دارد، ARP Table کلاینت و سایر اطلاعات که مدیر SNMP می‌تواند از آن‌ها اطلاع یابد را در اختیار قرار دهد.

```
enable
configure terminal
snmp-server community apa ro NAME
access-list NAME permit 192.168.1.1
snmp-server group apa-group v3 priv
snmp-server user apa-user apa-group v3 authentication sha 1234 priv aes 256 1234
snmpOserver host 192.168.1.1 version 3 priv apa-user
```



Secure Programming in Android

تهیه و تدوین: آرش بهرام زارعی

مقدمه

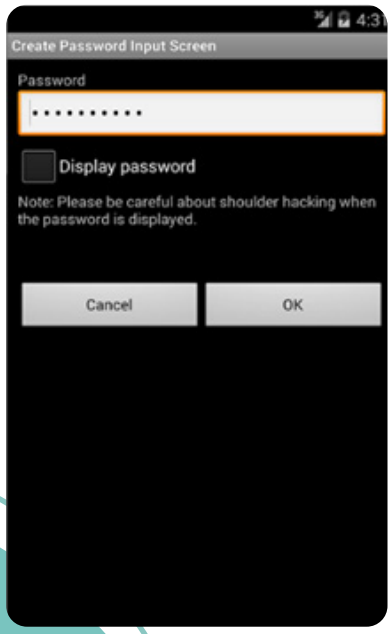
وقتی در مورد امنیت صحبت می‌شود، فقط منظور صحبت برای کاربران نرم‌افزارها و اپلیکیشن‌ها نیست. برنامه‌نویسان و مدیران سیستم نیز باید موارد امنیتی حوزه خودشان را بشناسند و آن‌ها را رعایت کنند. همین امر باعث تبلیغ و توسعه اپلیکیشن و برنامه‌ها برای برنامه‌نویسان می‌شود. به‌عبارتی برنامه‌های ایمن رابطه متقابلی با جذب کاربر دارند.
زمانی‌که نرم افزاری طراحی می‌شود، بهتر است قبل از تحلیل و طراحی برنامه، در برابر داده‌ها یا ورودی‌های غیرمجاز، اقدامات امنیتی را نیز در نظر گرفت و یا تحلیل امنیتی اپلیکیشن حتما مدنظر قرار گیرد. اشکال‌زدایی اپلیکیشن قدمی مهم برای کدنویسی امن می‌باشد.
کشف و شناسایی نقص در برنامه توسط تیم پیاده‌ساز هرچند ناخوشایند به نظر می‌رسد اما نباید از نعمت باگی که خود تیم پیدا کرده ناشکر شد! چرا که آن وقت مجبورند نقص اپلیکیشن را برطرف کنند و همین امر موجب بستن دری به روی خرابکاران می‌شود.
هرچند که طراحی برنامه را در جهت مثبت و برای کمک و راحت‌کردن کار کاربران خوب در نظر گرفته‌اند، اما همیشه خرابکارانی با داده‌ها یا ورودی‌های بد وجود دارند که بخواهند به برنامه حمله‌کنند. حتی ممکن‌است کاربری به اشتباه و ناخواسته داده‌ای را وارد کند که در برنامه خللی ایجاد شود و همین امر باعث به‌وجود آمدن نقص و حتی لو رفتن داده‌ها و اطلاعات اپلیکیشن شود؛ بنابراین اگر همیشه داده‌های ورودی بررسی شوند، قدمی برای کدنویسی امن برداشته شده‌است و یا حتی می‌توان با خیال راحت یک برنامه امن ساخت.
سیستم عامل اندروید بسیاری از ویژگی‌های امنیتی داخلی مانند تکنیک‌های جعبه شنی برنامه، محافظت در برابر حملات سرریز بافر و غیره را دارا می‌باشد. در نتیجه برنامه‌های ساده اندرویدی که هیچ سیستم فایل یا عملیات شبکه‌ای انجام نمی‌دهند، به طور پیش‌فرض می‌توانند امن در نظر گرفته شوند.
اگر برای ایجاد و توسعه یک برنامه پیچیده و حرفه‌ای طرحی کشیده شود، این وظیفه وجود داردکه آن برنامه ایمن شود و توسعه‌دهندگان و برنامه‌نویسان مسئول هستند که از حریم شخصی کاربران برنامه خود محافظت کنند.
در ادامه نکات مهم و قدم‌های مثبتی که هر توسعه‌دهنده یا برنامه‌نویس برای ایجاد و توسعه یک برنامه‌ی امن برای کاربر، ملزم به استفاده از آن‌ها می‌باشد، پیشنهاد شده است.

ایجاد صفحه نمایش امن برای رمز عبور

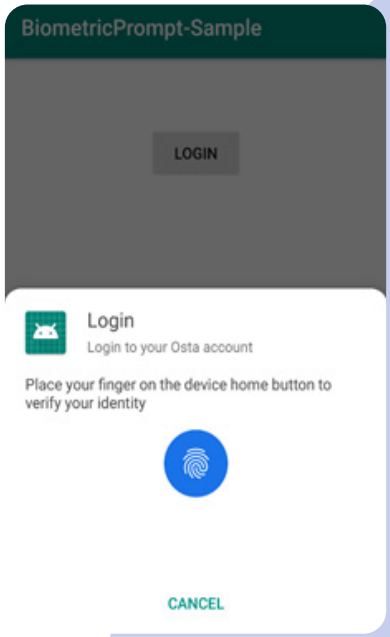
استفاده از توابع امنیتی

در هنگام ایجاد صفحه ورود با رمز عبور، از دیدگاه امنیتی برخی نکات وجود دارند که باید درنظر گرفته‌شوند:

۱. همانند تصویر ۱، رمز ورود باید به صورت ماسک‌دار نمایش داده‌شود (نمایش رمز عبور با * هنگام نوشتن).
۲. گزینه نمایش رمز عبور با یک متن ساده نوشته شود (بدون پیچیدگی و در حد امکان زیر کادر رمز عبور).
۳. به کاربر هشدار دهید که نمایش رمز عبور، در رمز عبور ساده دارای ریسک است.
٤. در صورت امکان از قابلیت اثر انگشت، همانند تصویر ۲، استفاده شود.



تصویر ۱. صفحه‌ی ورود امن

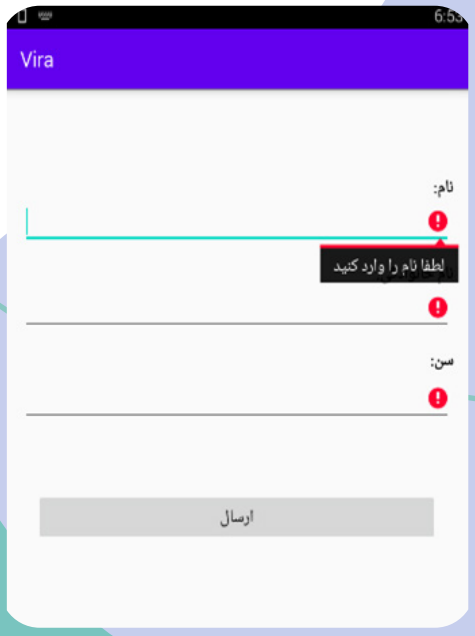


تصویر ۲. استفاده از قابلیت اثر انگشت

اعتبارسنجی داده‌ها

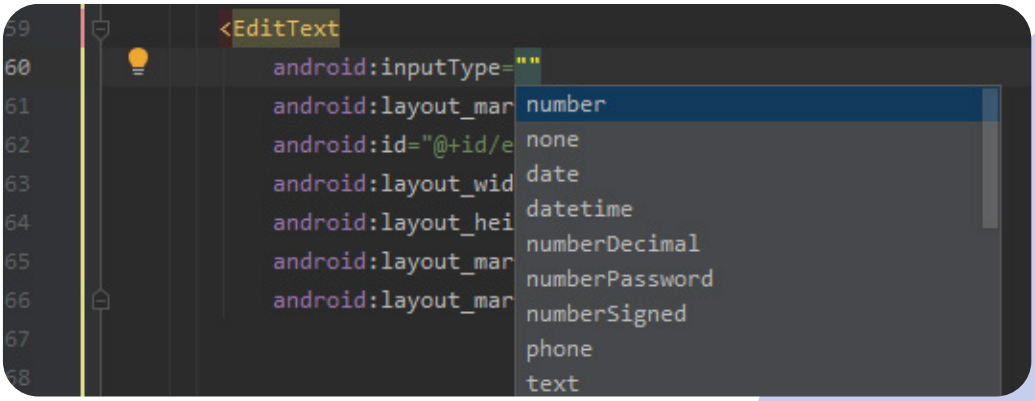
برای اعتبارسنجی داده‌ها در اندروید، روش‌های گوناگون و مختلفی وجود دارند که یکی از ساده‌ترین آن‌ها دستورات شرطی برای بررسی مقدار می‌باشد و همچنین می‌توان از خود صفت های موجود در ویوهایی که مقداری را دریافت و پردازش می‌کنند (EditText ها) استفاده کرد. انجام این عمل باعث می‌شود داده‌هایی را که ما به صورت ورودی دریافت می‌کنیم، به صورت صحیح‌ترین داده‌ی ممکن به‌دست بیاوریم.

در تصویر ۳ مثالی از گرفتن صحیح‌ترین نوع داده مورد نظر، گرفتن نام و نام خانوادگی و سن کاربر اجرا شده است.



تصویر ۳. دستور شرطی برای خالی نبودن EditText

همان‌طور که در تصویر ۳ مشاهده می‌کنید با استفاده از یک دستور شرطی می‌توان از خالی فرستادن داده‌ها جلوگیری کرد اما در این مثال فقط یک دستور شرطی ساده برای مقایسه‌کردن خالی بودن یا نبودن کادر EditText به‌کار رفته است.
همان‌گونه که گفته شد برای جلوگیری از نوشتن داده‌ی اشتباه (نوشتن حروف به جای عدد در کادر سن) می‌توانیم از صفتی به نام input Type در EditText استفاده کنیم (تصویر شماره ٤) و مقدار ورودی را به گرفتن نوع داده محدود کنیم، همین امر باعث می‌شود جز input Type مشخصی که قرار داده‌ایم، داده‌ی دیگری را نپذیرد.

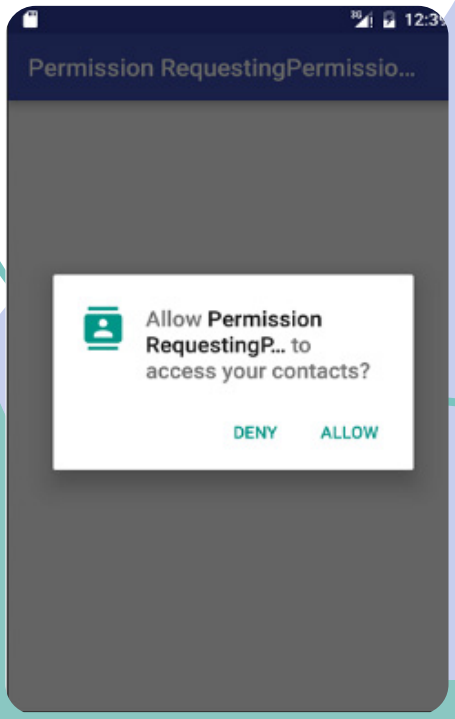


تصویر ٤. استفاده از صفت input Type در EditText

روش‌های استفاده از مجوزهای دارای ریسک

برای مجوز های دارای ریسک، برنامه در مواقع مناسب درخواست مجوز می‌کند. هنگامی که برنامه درخواست مجوز می‌کند، یک تأییدنامه مانند آنچه در تصویر ۵ نشان داده‌شده است، در سیستم عامل اندروید، به کاربر نمایش داده می‌شود و از کاربران اجازه میخواهد که آیا مجوز آن را صادر کند یا خیر. اگر کاربر اجازه استفاده از مجوز را بدهد، برنامه ممکن است هر عملیاتی که به آن مجوز احتیاج دارد را انجام دهد.
برای امنیت برنامه و همچنین اطمینان کاربر به فعالیت‌های داخل آن برنامه، لازم است موارد زیر جدی گرفته‌شود:

۱) در برنامه‌ای اگر که از مجوزهایی استفاده می‌شود، لازم است به صورت کامل، کاربر درجریان نحوه‌ی اجازه‌دادن به مجوز قرار داده شود.
۲) هشدار به کاربران (قبل از اجرای برنامه در صفحه نصب برنامه بررسی شود که آیا مجوزهایی به برنامه داده شده است یا خیر).
۳) هر فعالیتی که به نظر مشکوک می‌رسد را به کاربر توضیح داده و از مجوزهای مربوطه استفاده شود.
٤) در حد امکان از مجوزهای دارای ریسک استفاده نشود (بدون استفاده از مجوز)



تصویر ۵. استفاده از مجوز

حافظه‌ی پنهان

کاربرد حافظه‌ی پنهان برای زمانی است که داده‌هایی از سرور دریافت شده و می‌خواهند به صورت موقت و سریع از آن‌ها استفاده کنند. این داده‌ها را می‌توان در حافظه‌ی پنهان ذخیره کرد. گاهی اوقات لازم است تصویری پردازش شود و یا می‌خواهند قسمت‌هایی که به صورت کامل پردازش شده است را جای دیگری استفاده کنند، در این موارد نیز حافظه پنهان مورد استفاده خواهد بود. حافظه‌ی پنهان کاربردهای دیگری مانند ذخیره‌سازی داده‌های کاربر، ID و غیره را نیز دارد.
تصویر ٦ مثالی ساده از ساخت یک فایل موقت داخل حافظه پنهان را نشان می‌دهد:
قطعاً استخراج این فایل‌ها از حافظه‌ی پنهان و پیداکردن رمز عبور سخت‌تر از زمانی است که فایل‌ها را به صورت فیزیکی از حافظه استخراج کنیم.

```
File outputDir = this.getCacheDir();
File outputFile = File.createTempFile(“prefix”, “extension”, outputDir);
```

تصویر ٦. ساخت فایل موقت داخل حافظه پنهان

Cipher

معمولاً قبل از ذخیره‌کردن داده‌ها در حافظه نهان، Shared Preference یا دیتابیس، آن‌ها را Encode می‌کنند که برای این منظور استفاده از Cipher توصیه می‌شود. Cipher ابزاری مفید جهت Encode/Decode داده‌هاست.
در تصویر ۷ مثالی از کاربرد Cipherمشاهده می‌شود:

```
private static byte[] encrypt(byte[] key, byte[] input) throws Exception {
SecretKeySpec skeySpec = new SecretKeySpec(key, "AES");
Cipher cipher = Cipher.getInstance("AES");
cipher.init(Cipher.ENCRYPT_MODE, skeySpec);
byte[] encrypted = cipher.doFinal(input);
return encrypted;
}

private static byte[] decrypt(byte[] key, byte[] encrypted) throws Exception {
SecretKeySpec skeySpec = new SecretKeySpec(key, "AES");
Cipher cipher = Cipher.getInstance("AES");
cipher.init(Cipher.DECRYPT_MODE, skeySpec);
byte[] decrypted = cipher.doFinal(encrypted);
return decrypted;
}
```

تصویر ۷. مثالی از کاربرد Cipher

برای استفاده از برخی از الگوریتم‌های رمزنگاری مانند AES، نیازمند کلیدی با طول ثابت (۱۲۸, ۱۹۲, ۲۵۶ و ...) هستند.

Shared Preference

برنامه‌نویسانی که می‌خواهند برنامه امنی داشته باشند هیچ‌گاه حتی متن معمولی را داخل Shared Preference قرار نمی‌دهند و همه‌ی داده‌ها را رمزنگاری می‌کنند. هر داده‌ای که می‌خواهند ثبت کنند در حالت MODE_PRIVATE ذخیره می‌نمایند. این حالت اطمینان می‌دهد که تنها فقط این برنامه قادراست به داده‌های Shared Preference دسترسی داشته باشد و البته بدون دسترسی روت هیچ کسی نمی‌تواند به این داده‌ها دسترسی غیرمجاز داشته باشد. در همین رابطه ابزار دیگری تحت عنوان Shared Secure Preference وجود دارد که کار را برای شما آسان‌تر می‌کند.
این ابزار بر مبنای همان Shared Preference اما همراه با تعداد زیادی الگوریتم‌های مفید رمزنگاری بنا شده است. باید توجه داشت که نیازی نیست که برای استفاده از آن موارد جدیدی را بیاموزید، دقیقاً به همان صورت سابق از Shared Preference استفاده شده و همانند تصویر ۸ این ساختار امن‌تر خواهد بود.

```
SharedPreferences prefs = new SecurePreferences(context, "userpassword", "my_user_prefs.xml");
```

تصویر ۸. استفاده از Shared Preference با ابزار Secure Shared Preference

بعد از فرایند رمزنگاری، نتیجه به‌صورت تصویر ۹ خواهد بود.

```
<map>
<string name="TuwbBU0IrAyL9znGBJ87uEi7pW0FwYwX8SZiiKnD2VZ7">
pD2UhS2K2MNjWm8KzpFrag==:MWm7NgaEhvaxAvA9wASUl0HUHCVBWkn3c2T1WoSAE/g=rroijgeWEGRDFSS/hg
</string>
<string name="ʌlqCQqn73Uo84Rj">k73tlfVNYsPshll19ztma7U>
</map>
```

تصویر ۹. نتیجه فرایند رمزنگاری SharedPreference

Root

باید این مورد را درنظر گرفت که همیشه از قبل باید بررسی شود که آیا دستگاه کاربر روت شده است یا خیر؟ زیرا در صورت روت بودن گوشی، هکر می‌تواند به تمامی قسمت‌های برنامه دسترسی داشته باشد. برنامه‌نویسان قبل از شروع کدنویسی با یک کد ساده می‌توانند در صورت روت‌بودن دستگاه اجازه اجرای برنامه را ندهند. برای تشخیص روت‌بودن گوشی همانند تصویر ۱۰ می‌توان از این کد استفاده کرد. لازم به ذکر است که کد مورد استفاده در تصویر ۱۰ مربوط به کتابخانه RootTools است. دستور su معمولاً برای تغییر مالکیت از کاربر اورجینال به کاربر روت استفاده می‌شود. در صورت True بودن نتیجه‌ی این متد، می‌توان سرور را مطلع کرد که وقفه‌ای را صادر کند یا پیغام مناسب در این خصوص به کاربر داده شود. باید دقت کرد که زیاده‌روی و سخت‌گیری بیش از حد باعث تجربه کاربری بد و روگردانی کاربران از برنامه می‌شود.

```
private static boolean isRooted() {
return findBinary("su");
}

public static boolean findBinary(String binaryName) {
boolean found = false;
if (!found) {
String[] places = {"/sbin/", "/system/bin/",
"/system/xbin/", "/data/local/xbin/",
"/data/local/bin/", "/system/sd/xbin/",
"/system/bin/failsafe/", "/data/local/"};
for (String where : places) {
if (new File(where + binaryName).exists()) {
found = true;
break;
}
}
}
return found;
}
```

تصویر ۱۰. بررسی روت بودن دستگاه کاربر

مبهم‌سازی

در فرایند تولید و توسعه اپلیکیشن‌های تلفن همراه استفاده از توابع مبهم‌ساز پیشنهاد می‌شود. چند نمونه از این توابع شامل ProGuard، DexGuard و DexProtector است. ProGuard مبهم‌ساز پیش‌فرض اندروید استودیو است. این ابزار ضمن مبهم‌کردن کدها، حجم آن را نیز کاهش می‌دهد. در هنگام ساخت پروژه‌های اندروید، فایل کانفیگ ProGuard نیز به صورت خودکار ساخته می‌شود و اسم این فایل proguard-rules.pro است. در این توابع می‌توان قواعدی تعریف کرد که در هنگام ایجاد کردن فایل اپلیکیشن اجرا شوند.
برای فعال کردن ProGuard همانند تصویر ۱۱ عمل می‌شود.
کدهای بهم‌ریخته و مبهم، زمان زیادی را از هکرها به منظور تحلیل و سواستفاده از اپلیکیشن می‌گیرد و در اغلب مواقع به خاطر وقت و زحمت زیادی که رمزگشایی این کدها لازم دارد آنها را از ادامه‌ی کار منصرف می‌کند مگراینکه برنامه‌ی شما واقعاً ارزش این همه وقت‌گذاشتن را داشته باشد. البته به خاطر همه‌گیرشدن استفاده از ProGuard، سعی کنید از مبهم‌سازی‌های مطرح و ناشناخته‌تر استفاده کنید که هنوز DEobfuscatorهایی برایشان عرضه نشده است.
بهترین توابع مبهم‌ساز معمولاً دارای هزینه بالایی هستند ولی می‌توانند کل برنامه را غیرقابل درک کرده و تحلیل آن را بسیار سخت کنند که این نکته خوبی است. باید توجه داشت که این پایان راه نیست، بررسی روت بودن دستگاه، اشکال‌زدایی، شبیه‌سازی و اصلاح کدها از دیگر قابلیت‌های این ابزارها هستند که قیمت آنها را بیشتر کرده است

```
android {
buildTypes {
dev {
minifyEnabled true // enables ProGuard
proguardFiles getDefaultProguardFile('proguard-android.txt'), 'proguard-rules.pro'
}
}
```

تصویر ۱۱. فعال کردن ProGuard

کدهای بهم‌ریخته و مبهم، زمان زیادی را از هکرها به منظور تحلیل و سواستفاده از اپلیکیشن می‌گیرد و در اغلب مواقع به خاطر وقت و زحمت زیادی که رمزگشایی این کدها لازم دارد آنها را از ادامه‌ی کار منصرف می‌کند مگراینکه برنامه‌ی شما واقعاً ارزش این همه وقت‌گذاشتن را داشته باشد. البته به خاطر همه‌گیرشدن استفاده از ProGuard، سعی شود از مبهم‌سازی‌های مطرح و ناشناخته‌تر استفاده گردد که هنوز DEobfuscatorهایی برایشان عرضه نشده است.
بهترین توابع مبهم‌ساز معمولاً دارای هزینه بالایی هستند ولی می‌توانند کل برنامه را غیرقابل درک کرده و تحلیل آن را بسیار سخت کنند که این نکته خوبی است. باید توجه داشت که این پایان راه نیست، بررسی روت بودن دستگاه، اشکال‌زدایی، شبیه‌سازی و اصلاح کدها از دیگر قابلیت‌های این ابزارها هستند که قیمت آنها را بیشتر کرده است.

معرفی ابزار DexGuard

DexGuard به‌عنوان یکی از ابزارهای قدرتمند امنیتی برای حفاظت از اپلیکیشن‌های اندرویدی در برابر مهندسی معکوس، دستکاری کد، حملات زمان اجرا و غیره است که به‌عنوان نسخه تجاری ProGuard با ویژگی‌ها و امکانات به مراتب پیشرفته‌تر، از سوی شرکت GuardSquare عرضه شده است. راه‌اندازی و اعمال تنظیمات DexGuard در محیط‌های مختلف برنامه‌نویسی بسیار آسان است. در این بخش به چگونگی راه‌اندازی DexGuard در Android Studio به‌عنوان محبوب‌ترین محیط برنامه‌نویسی اندروید در میان توسعه دهندگان و برنامه‌نویسان اپلیکیشن‌های موبایل و معرفی افزونه خاص DexGuard برای این محیط، می‌پردازیم.

اعمال DexGuard در یک پروژه اندرویدی

برای اعمال DexGuard تنها باید چهار بخش که در ادامه توضیح داده می‌شوند را به فایل build.gradle پروژه اضافه نمایید.
این چهار بخش شامل:

- باید مسیر قرار گرفتن فایل jar را که در پکیج DexGuard وجود دارد، همانند تصویر ۱۲ مشخص نمایید.

```
buildscript {
  ...
  dependencies {
    classpath ':dexguard:'
  }
}
```

تصویر ۱۲. مسیر قرار گرفتن فایل jar

- باید dependency ابزار DexGuard را برای ساخت برنامه از طریق classpath مطابق با تصویر ۱۳ تعریف نمایید.

```
buildscript {
  ...
  dependencies {
    classpath ':dexguard:'
  }
}
```

تصویر ۱۳. اضافه کردن DexGuard به classpath

- حال باید افزونه مانند تصویر ١٤ تعریف و اعمال گردد.

```
apply-plugin: 'dexguard'
```

تصویر ١٤. ایجاد افزونه

- در مرحله آخر همانند تصویر ۱۵ باید فایل تنظیمات DexGuard را مشخص نمایید. لازم به ذکر است که DexGuard به‌عنوان یک ابزار حرفه‌ای امنیتی، قابلیت اعمال تنظیمات مختلف را از طریق یک فایل تنظیمات، مهیا می‌سازد. برای شروع، یک فایل تنظیمات اولیه در پکیج DexGuard موجود است.

```
android {
  buildTypes {
    debug {
      proguardFile getDefaultDexGuardFile('dexguard-debug.pro')
    }
    release {
      proguardFile getDefaultDexGuardFile('dexguard-release.pro')
    }
  }
}
```

تصویر ۱۵. تنظیم فایل تنظیمات DexGuard

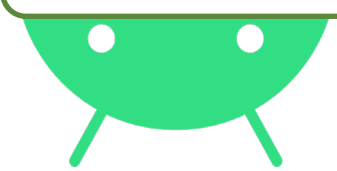
بعد از اضافه‌کردن چهار بخش معرفی‌شده فوق به فایل build.gradle، باید کلیه امکانات و تنظیمات مربوط به Proguard را از تنظیمات Buildtype، شامل مواردی همچون multidexenable ، minifyenabled و shrinkresources حذف نمایید. لازم به ذکر است DexGuard کلیه این امکانات را به‌همراه خود دارد.

تنظیم DexGuard

همان‌طور که اشاره شد، می‌توان باتوجه به نیازمندی امنیتی پروژه و قابلیت‌ها و ویژگی‌های مورد نیاز، اقدام به تنظیم DexGuard کرد. این ابزار با ارائه قابلیت‌های زیر، در اعمال مناسب‌ترین تنظیمات برای اپلیکیشن کمک می‌نماید:

- DexGuard به نوعی نسخه تجاری Proguard است که هر دو توسط یک شرکت ارائه شده‌اند. به این ترتیب این دو محصول کاملا با یکدیگر سازگار بوده و می‌توان برای شروع از تنظیمات فعلی Proguard در پروژه خود استفاده نمود و تنظیمات DexGuard را برای بکارگیری قابلیت‌های پیشرفته آن به تنظیمات فعلی اضافه کرد.
- DexGuard دارای یک افزونه (Plugin) برای محیط Android Studio است که قابلیت تکمیل خودکار را برای انجام تنظیمات بر روی فایل تنظیمات، در اختیار توسعه‌دهنده قرار می‌دهد. برای نصب این افزونه، به منوی Preferences > Plugins در اندروید استودیو رفته و بر روی گزینه Install plugin from disk... کلیک شود. از آنجا پوشه DexGuard را انتخاب و می‌توان عملیات نصب را انجام داد.
- به‌همراه پکیج DexGuard تنظیمات بهینه‌شده‌ای برای فریم‌ورک‌های معروف و متداول نظیر Firebase, Crashlytics, okHttp, Dagger وغیره موجود است. پیشنهاد می‌شود هرگونه تنظیمات مربوط به Proguard در هر یک از این فریم‌ورک‌ها، قبل از اعمال تنظیمات جدید، حذف گردند.

Intents



راه ارتباطی برنامه‌های اندرویدی Intentها هستند. دو نوع Intent وجود دارد: ضمنی و صریح .

Intent های صریح

- مزایا: شنودشان امکان‌پذیر نیست.
- معایب: تنها داخل خود برنامه قابل استفاده است.

Intent های ضمنی

- مزایا: در هر بخشی در داخل سیستم‌عامل اندروید قابل استفاده است.
- معایب: به‌سادگی شنود می‌شوند.

هکر به راحتی می‌تواند با تعریف یک intent-filter مشابه، Intent را شنود و همه‌ی داده‌هایش را سرقت کند. برای جلوگیری از این کار، باید تا می‌توان Intent را صریح تعریف کرد. همانند تصویر ۱۶ به طور مثال می‌توان یک package مشخص را تعیین کرد.

```
Intent intent = new Intent(Intent.ACTION_SEND);
intent.setPackage("com.test.package");
sendBroadcast(intent);
```

تصویر ۱۶. تعریف یک package به صورت صریح

همچنین باید دقت داشت که سرویس‌ها یا Broadcast Receiver هایی که با کامپونت‌های خارجی ارتباطی برقرار نمی‌کنند نباید Export شوند و میبایستی همانند تصویر ۱۷ از آنها استفاده کرد

```
<service android:name=".service.SomeService" android:enabled="true" android:exported="false">

<intent-filter>
<action android:name="android.intent.action.MAIN" />
<category android:name="android.intent.category.LAUNCHER" />
</intent-filter>
</service>
```

تصویر ۱۷. سرویس‌ها یا Broadcast Receiver

همان‌طور که گفته‌شد سعی شود در برنامه تا جاییکه که ممکن است از Intentهای صریح استفاده شود. همچنین برای حفظ امنیت بیشتر، گیرنده ای که می‌خواهید Intent شما را دریافت کند به صورت صریح مشخص نمایید.

شبکه و برقراری ارتباط

این روزها تقریباً همه‌ی اپلیکیشن‌ها برای تبادل اطلاعات کاربران، توکن ها و احراز هویت با سرورها و اینترنت در ارتبا ط هستند. برنامه‌ی امن باید به‌گونه‌ای باشد که اجازه سرقت اطلاعات کاربران در این فضا را ندهد.
اولین قدم در برقراری امنیت ارتباطات اینترنتی استفاده از پروتکل ایمن HTTPS به جای HTTP است. باید دقت داشت که این کار لازم است اما کافی نیست. یکی از مشهورترین حملات شبکه، حمله‌ی مرد میانی (MITM: Man-In-The-Middle) است که می‌تواند به دو صورت فعال (Active) و غیرفعال (Passive) صورت گیرد. برای مقابله با حملات غیرفعال MITM به طور مثال می‌توان از الگوریتم تبادل کلید دیفی-هلمن استفاده شود.
حملات فعال کمی قوی‌تر بوده که برای مقابله با آن از SSL Pinning استفاده می‌شود. برخی ابزارها از HTTPS و SSL pinning حمایت می‌کنند که Retrofit و OkHttp دو نمونه از آن‌هاست. کتابخانه‌ی Retrofit، استفاده‌ی آسانی دارد؛ از RxJava پشتیبانی می‌کند و پیکربندی آن وقت زیادی را نمی‌گیرد. به کمک OkHttp می‌توان گواهینامه‌ی SSL معتبر اضافه کرد.
باید توجه داشت که از نظر امنیتی، ارتباطات HTTPS ارجح‌تر است. اخیراً، سرویس‌های وب مطرح مانند Google یا Facebook از HTTPS به‌عنوان پیش فرض استفاده می‌کنند.

پایگاه داده‌ی امن و حفاظت از اطلاعات مهم

در برخی از پروژه‌های اندرویدی، اصول امنیتی به‌کار برده شده از اهمیت بالایی برخوردار است. به گونه‌ای که برنامه‌نویس تمایل دارد این اطلاعات برنامه توسط افراد دیگر و به‌وسیله‌ی مهندسی معکوس قابل استفاده نباشد، توصیه اکیدی که در این مواقع انجام می‌شود این است که این اطلاعات، درون اپلیکیشن ذخیره نشود و حتی‌الامکان در داخل سرور ذخیره شده و هر زمانی که به آنها نیاز بود، از طریق اپلیکیشن با اتصال به سرور، این اطلاعات در دسترس قرار بگیرند و در نهایت نیز از حافظه‌ی اپلیکیشن پاک شوند اما اگر به هر دلیلی دسترسی به سرور مقدور نبود در ادامه چندین روش برای حفاظت اطلاعات بیان می‌شود که هر کدام در نوع خود کاربرد دارند.
معمولا برای ذخیره اطلاعات در برنامه‌های اندروید چهار گزینه وجود دارد:
۱) ذخیره داده در پایگاه داده که در اندروید معمولا از Sqlite (با وجود نا امن بودن) استفاده می‌شود.
۲) ذخیره اطلاعات در Shared Preferences
۳) ذخیره اطلاعات در فایل‌های جانبی
٤) ذخیره اطلاعات در خود برنامه
لازم به یادآوری است که دسترسی به اطلاعاتی که این‌گونه ذخیره می‌شوند امکان‌پذیر و بسیار راحت است اما می‌توانیم با رعایت کدنویسی امن در اندروید، امنیت داده‌ها را به اندازه‌ی مطلوبی بالا ببریم.
در ادامه روش‌هایی برای ذخیره‌سازی اطلاعات معرفی می‌شوند که بتوان از داده‌های ذخیره شده بهتر محافظت کرد.

ذخیره داده‌ها و اطلاعات حساس

مهم‌ترین بحث در اپلیکیشن‌های اندروید، بحث پیرامون داده‌ها یا اطلاعات کاربر که آماده‌ی ذخیره شدن هستند، می‌باشد. نکته‌ی مهم این است که این اطلاعات کجا ذخیره شوند؟ نکته‌ی بسیار مهم برای ذخیره‌ی سازی داده‌ها مخصوصا داده‌های حساس، این است که حتما از حافظه داخلی استفاده شود.
حافظه‌ی داخلی محل ذخیره‌سازی داده‌های شخصی بر روی حافظه‌ی دستگاه می‌باشد. داده‌هایی که کاربر از طریق اپلیکیشن ثبت می‌کند، در یک فایل مشخص شده در حافظه داخلی اندروید دستگاه ذخیره می‌شوند اما کاربر نمی‌تواند به آن فایل‌ها دسترسی داشته باشد و این فایل فقط توسط خود برنامه قابل دسترسی است. هر برنامه‌ی اندرویدی دارای یک فهرست داخلی است که مسیر آن بر اساس نام پکیج اپلیکیشن می‌باشد. پوشه‌ها و پرونده‌های داخل این فهرست بسیار امن هستند زیرا به‌طور پیش‌فرض از حالت ایجاد پرونده MODE_PRIVATE استفاده می‌کند. یعنی این فایل‌ها در هیچ برنامه دیگری در تلفن همراه و یا دستگاه‌های دیگر، قابل خواندن نبوده و قابل دسترسی نیستند. بنابراین بهترین مکان برای ذخیره تمام داده‌های حساس برنامه در فهرست حافظه داخلی است.

حافظه داخلی اندروید دارای دو حالت است:
- MODE_PRIVATE: در حالت خصوصی داده‌های ذخیره شده قبلی با داده‌های فعلی جایگزین می‌شوند. یعنی هر بار که سعی می‌شود نوشته جدیدی به فایل اضافه گردد، نوشته قبلی در حافظه داخلی پاک می‌شود.
- MODE_APPEND: در این حالت داده‌ها به محتوای موجود اضافه می‌شوند و داده‌های قبلی حذف نمی‌شوند.

نوشتن فایل در حافظه داخلی اندروید چگونه است؟

برای نوشتن فایل در حافظه داخلی اندروید، نام فایل به‌عنوان یک رشته تعریف شود و همچنین داده‌هایی که لازم است نوشته شود، به‌عنوان رشته یا به هر فرمت ایجاد شده از برنامه یا هر منبع دیگر، تعریف میگردد. در این موارد باید از روش FileOutputStream با ایجاد شیء تعریف شده، استفاده شود. داده‌ها را قبل از نوشتن، به فرمت بایت تبدیل شوند، زیرا فایل، فقط فرمت بایت را می‌پذیرد.

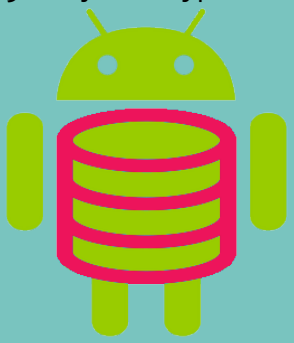
به‌عنوان مثال برای تعیین مسیر و ذخیره در حافظه داخلی برنامه، معمولا توصیه می‌شود که از روش getFilesDir استفاده شود. برای نمونه و باتوجه به تصویر ۱۸، فایلی به نام user_information.dat در فهرست حافظه داخلی برنامه ذکر شده است اما متاسفانه همیشه ظرفیت حافظه‌ی داخلی گوشی‌ها محدود هستند که برای ذخیره داده‌هایی با حجم زیاد ممکن است راه بن‌بستی برای کارکردن اپلیکیشن باشد، بنابراین راهی به‌جز ذخیره‌سازی داده‌ها در فضای ذخیره‌سازی خارجی وجود نخواهد داشت.

```
String File_Name= "Demo.txt"; //gives file name
String Data="Hello!!"; //define data

FileOutputStream fileobj = openFileOutput( File_Name, Context.MODE_PRIVATE);
byte[] ByteArray = Data.getBytes(); //Converts into bytes stream
fileobj.write(ByteArray); //writing to file
fileobj.close(); //File closed
```

تصویر ۱۸. ذخیره فایل user_information.dat در حافظه داخلی

رمزگذاری داده‌ها در فضای ذخیره‌سازی خارجی

همان‌طور که گفته شد ظرفیت ذخیره‌سازی داخلی یک دستگاه اندرویدی محدود است بنابراین، در برخی از موارد، ممکن است چاره‌ای جز ذخیره داده‌های حساس در حافظه‌ی ذخیره‌سازی خارجی مانند کارت SD که قابلیت جابجایی دارند، وجود نداشته باشد. از آنجا که داده‌ها در حافظه‌های ذخیره‌سازی خارجی توسط کاربران و سایر برنامه‌های موجود در دستگاه به‌طور مستقیم قابل دسترسی هستند، پس مهم است که در قالب رمزنگاری‌شده ذخیره شوند تا دیگر نگرانی بابت استفاده از فایل‌ها و داده‌ها و یا حمله به آنها وجود نداشته باشد. یکی از محبوب‌ترین الگوریتم‌های رمزنگاری که امروزه توسط توسعه‌دهندگان استفاده می‌شود AES می‌باشد. نوشتن کد برای رمزنگاری و رمزگشایی داده‌های برنامه با استفاده از بسته javax.crypto، که در SDK اندروید موجود است قابل استفاده می‌باشد، اما معمولا کار با این نوع بسته‌ی بسیار پیچیده، زمانبر و طولانی و یا می‌توان گفت که بسیار گیج‌کننده است. اکثر توسعه‌دهندگان ترجیح می‌دهند از کتابخانه‌هایی که در اندروید توسط توسعه‌دهندگان دیگر نوشته شده است، استفاده کنند مانند کتابخانه Facebook Conceal و JealousSky و غیره که کار با آنها نسبت به بسته‌ی خود SDK بسیار آسان‌تر است. در ادامه بحث رمزنگاری مثالی نیز برای درک کامل موضوع ارائه شده است.



رمزنگاری

منظور از رمزنگاری این است که اطلاعات به گونه‌ای تبدیل شوند که قابل فهم برای طرف مقابل نباشد. برای رمزنگاری روش‌ها و الگوریتم‌های مختلفی وجود دارد اما یک سری از الگوریتم‌ها وجود دارند که استاندارد و از امنیت بالایی برخوردار هستند، می‌توان به RSA ، AES و DES اشاره کرد.

یکی از کارایی‌های مهم رمزنگاری در برنامه‌نویسی اندروید، جلوگیری از بدست آوردن اطلاعات درون برنامه به وسیله‌ی کاربران است. به‌عنوان مثال اگر اپلیکیشنی در مورد کتاب را نوشته باشد و نخواهیم دیگران به داده‌های آن دسترسی پیدا کنند، می‌توان ابتدا داده‌ها را به صورت رمزنگاری‌شده داخل فایل یا پایگاه داده ذخیره شوند و سپس در داخل برنامه آن داده‌ها را رمزگشایی کرد، سپس به کاربر نشان داد.

برای استفاده از این الگوریتم‌ها می‌توان از کتابخانه‌های آماده استفاده کرد. به‌عنوان مثال کتابخانه JealousSky یک کتابخانه‌ی بسیار قدرتمند برای رمزنگاری و رمزگشایی است. با استفاده از این کتابخانه می‌توان علاوه بر متن، هر عکسی را نیز به صورت bitmap رمز کرد. این کتابخانه از الگوریتم AES ۱۲۸ بیت استفاده می‌کند.

پایگاه داده اندروید SQLite

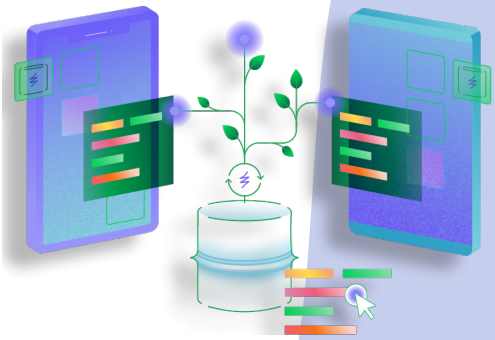
SQLite یک پایگاه داده بسیار کوچک با حجمی کمتر از یک مگابایت می‌باشد که در قالب یک کتابخانه نوشته شده و به صورت منبع‌باز و رایگان منتشر شده است. بنابراین گوگل یا توسعه دهنده نیازی نیست برای استفاده از این پایگاه داده مبلغی را به سازنده بپردازند. همچنین بر خلاف دیتابیسی مانند MySQL که نیاز به سرور دارد، SQLite بی‌نیاز از سرور بوده و به صورت مستقل بر روی هر دستگاه مستقر شده که اصطلاحا ServerLess نامیده می‌شود. از این رو معمولا اپلیکیشن‌هایی با پایگاه داده‌ی sqlite امن نبوده و باعث نفوذ و حمله‌ی هکرها می‌شوند. در ادامه چندین پایگاه داده برای جایگزینی sqlite معرفی می‌گردد.

Realm Database

پایگاه داده Realm Database یک سیستم مدیریت پایگاه داده منبع باز است. Realm یک جایگزین برای پایگاه داده SQLite می‌باشد که کارآمدتر و سریع‌تر بوده و جایگزین بسیار خوب و مناسبی برای سایر پایگاه داده‌های موبایل محسوب می‌شود. Realm فضای بسیار کمتری اشغال می‌کند، از این رو حجم اپلیکیشن را کاهش می‌دهد. این پایگاه داده در نوشتن و خواندن داده‌ها عملکرد بسیار سریعی را از خود به نمایش می‌گذارد، cross-platform بوده و از ios و android پشتیبانی می‌کند.

• قابلیت‌ها

پایگاه داده Realm Database هماهنگ سازی‌های دو طرفه بین Realm Object Server و پایگاه داده‌های جانبی مشتری که متعلق به کاربر وارد شده هستند را انجام می‌دهد. در این سیستم، هر دوی توسعه‌دهنده و نسخه تجاری همراه با مجوز



کسب‌وکار برای ادغام با دیگر سیستم‌های مدیریت پایگاه داده مانند PostgreSQL منتشر می‌شوند.

• ویژگی‌ها

پایگاه دادهRealm دارای ویژگی‌های زیاد و متنوعی است که در ادامه به بیان مهم‌ترین آن‌ها می‌پردازیم:

۱) ارتباط بین اشیاء از طریق لینک‌ها مجاز است. هر لینک یک Backlink به‌عنوان یک رابطه معکوس ایجاد می‌کند که با هر کدام از اشیاء فعلی پیوند دارد.

۲) نتایج پرس‌وجو توسط Realm Database، نشان دهنده نمایه‌های موضوعی محلی هستند.

۳) از Realm Database به‌عنوان جایگزین SQLite و CoreData یاد می‌شود، به‌طوری که در سال ۲۰۱۶ به‌عنوان بهترین دیتابیس اندرویدی انتخاب شد.

٤) همان طور که هر نتیجه پرس‌وجو و هر شی پروکسی یک دید به داده‌های اساسی است، هر تغییری که در پایگاه داده انجام شود نیز در تمام اشیاءهایی که به داده‌های مشابه اشاره می‌کنند، منعکس می‌شود.

• پشتیبانی از زبان‌های برنامه‌نویسی

پایگاه داده Realm Database از زبان‌های برنامه‌نویسی مختلفی پشتیبانی می‌کند. این سیستم مدیریت پایگاه داده در سیستم‌عامل IOS، زبان برنامه‌نویسی Swift و در سیستم عامل اندروید، زبان برنامه‌نویسی Java را پشتیبانی می‌کند و می‌توان از آن استفاده کرد.

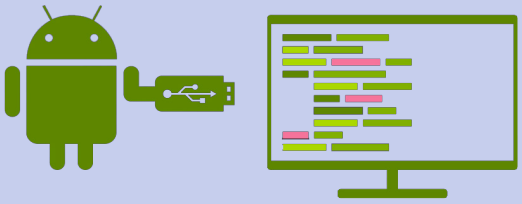
Neo4j Database

در عصر کنونی، Big Data (کلان داده) بخش قابل‌توجهی از دارایی‌های کسب‌وکارها را تشکیل داده اما باید توجه داشت که مدیریت کردن این حجم زیاد از داده‌ها نیاز به زیرساخت‌های مناسبی دارد و دیتابیس‌هایی که در گذشته معمول بوده‌اند دیگر پاسخگوی نیاز این دسته از کسب‌وکارها نخواهند بود. در همین راستا دیتابیسی تحت عنوان Neo4j طراحی و توسعه داده شده که در دسته‌بندی دیتابیس‌ها، جزء دیتابیس‌های مبتنی بر گراف محسوب می‌شود.
به طور کلی، دیتابیس گرافی نوعی از پایگاه‌های داده است که برای پرس‌وجو از ساختار گراف‌های مختلف استفاده می‌کنند. اکثر این نوع دیتابیس‌ها ماهیت NoSQL دارند و داده را برای ذخیره‌سازی به صورت یک جفت Key-Value ذخیره می‌کنند. این دیتابیس زبان مخصوص به خود، به اسم Cypher را داراست که برای ذخیره و بازیابی اطلاعات می‌توان از آن استفاده کرد.
Neo4j در سال ۲۰۰۰ توسعه یافت به‌طوری که پس از ده سال نسخه اول آن به بازار عرضه شد و با توجه به ماهیت منبع‌باز بودن آن، در این سال‌ها به‌عنوان معروف‌ترین دیتابیس گرافی دنیا قلمداد می‌شود.
برخی از ویژگی‌های بارز این دیتابیس عبارتند از:

- Neo4j جزء یکی از بهترین دیتابیس‌های گرافی دنیا است.
- این دیتابیس ماهانه چندین هزار بار دانلود می‌شود.
- یادگیری آسان
- سهولت در استفاده
- کاهش استفاده از حافظه
- مناسب برای رایانش ابری (cloud)
- پشتیبان‌گیری حرفه‌ای
- شمای انعطاف‌پذیر
- سازگاری با زبان پرس‌وجونویسی Cypher
- درایور برای زبان‌های جاوا، سی‌شارپ، پایتون، جاوا اسکریپت، روبی، پی‌اچ‌پی، آر، گو و غیره
- پشتیبانی از فریم‌ورک‌های اِسپرینگ، جَنگو، لاراول و غیره

نکاتی درباره سیاست حفظ حریم خصوصی کاربران در اپلیکیشن

۱) روش‌هایی را ارائه دهید که توسط آنها کاربر بتواند سیاست حفظ حریم خصوصی برنامه را مرور کند.
۲) روش‌هایی را ارائه دهید که با استفاده از آن کاربر بتواند داده‌های منتقل شده را حذف کند.
۳) روش‌هایی را فراهم کنید که با استفاده از آنها بتوان انتقال داده را متوقف کرد.
٤) از UUID ها یا کوکی‌ها برای پیگیری اطلاعات کاربر استفاده کنید.
۵) حتما نسخه خلاصه‌ای از خط‌مشی رازداری و سیاست حفظ حریم خصوصی برنامه را در پوشه دارایی‌ها قرار دهید.



Information Security

امنیت اطلاعات

نشت اطلاعات و راهکارهای جلوگیری از آن

تهیه و تدوین: هادی گلباغی

نشت اطلاعات چیست؟

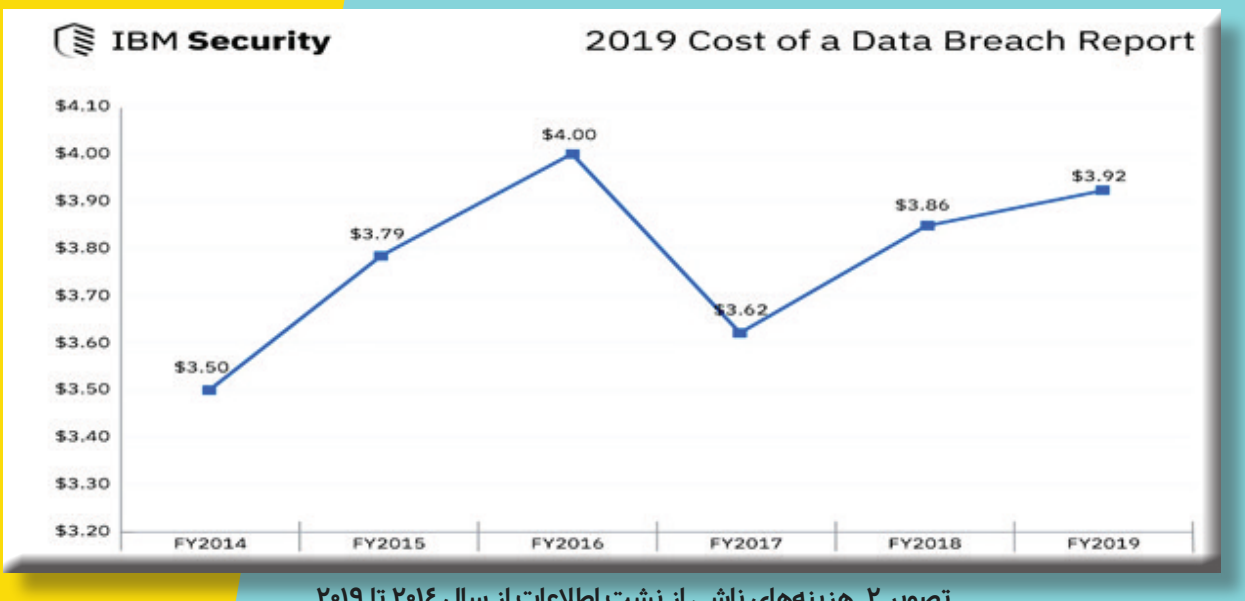
انتشار عمد یا غیر عمد اطلاعات حساس کاربران را می‌توان نشت اطلاعات نامید. در نشت اطلاعات داده‌های شخصی کاربران مانند نام، کد ملی، آدرس ایمیل، کلمات عبور، شماره تلفن‌ها، تاریخ تولد، شماره کارت بانکی و غیره، بدون مجوز صاحب آن نشر داده می شود که می‌تواند اطلاعات مهمی برای هکرها باشد. همچنین در نشست اطلاعات دسترسی به منابع داده و اطلاعات حساس بدون مجوز مالک اطلاعات صورت می‌گیرد. معمولاً اصطلاحات مترادف مختلفی در فارسی و انگلیسی برای نشت اطلاعات مانند درز اطلاعات، افشای اطلاعات، لو رفتن اطلاعات، Data leakage، Data Breach و Data loss استفاده می‌شود. باید توجه داشت که شما تا زمانی مالک اطلاعات هستید که اطلاعات نشت نکرده باشد، پس از آن دیگر شما مالک اطلاعات خود نیستید. بنابراین نکته حائز اهمیت این است که پیش از بروز حادثه از منابع اطلاعاتی خود محافظت کنید.

هزینه‌های ناشی از نشت اطلاعات

در سالهای اخیر شاهد افزایش چشمگیر جرایم سایبری بوده‌ایم و در برخی از کشورها مقررات بسیار سخت و دقیقی جهت حفاظت از منابع و داده‌های کاربران برای شرکت‌ها و سازمان‌ها وضع شده است. در اروپا و آمریکا با تصویب قوانینی درخصوص نشت اطلاعات حساس در GDPR گام بلندی در این خصوص برداشته شده است و سازمان‌ها و شرکت‌هایی که دچار نشت اطلاعات شوند، محکوم به پرداخت جریمه‌ای تا ٤ درصد از درآمد سالانه خود هستند.

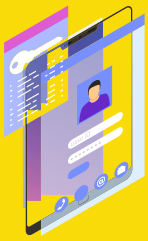
تصویر ۱. قوانین GDPR درخصوص نشت اطلاعات

بر طبق آماری که توسط IBM Security انتشار گردیده است هزینه‌های ناشی از نشت اطلاعات از سال ۲۰۱٤ تا ۲۰۱۹ در حدود ۱۲ درصد رشد داشته است که این آمار در تصویر ۲ نمایانگر این مسئله است. با بررسی آمارها در سال ۲۰۲۰ نیز مشاهده می‌شود که نرخ نشت اطلاعات افزایش پیدا کرده است. یکی از عوامل تاثیرگذار در سال ۲۰۲۰ درخصوص نشت اطلاعات، مسئله همه‌گیری کرونا و دورکاری کارمندان بوده است. به دلیل همه‌گیری کرونا اغلب سازمان‌ها و شرکت‌ها سیاست دورکاری را انتخاب کرده و کارها را در بستر مجازی انجام داده‌اند که انتقال داده‌ها به صورت ناامن در این بستر یکی از چالش‌ها درخصوص نشت اطلاعات بوده است. همچنین جلسات و کنفرانس‌ها در اغلب موارد در بستر مجازی و با استفاده از نرم‌افزارهایی مانند Skype، Zoom و غیره برگزار شده است که در همین دوره، تمرکز نفوذگران و هکرها بر این نرم‌افزارها نیز بسیار بیشتر شد و شاهد رشد چشمگیر آسیب‌پذیری‌های این نرم‌افزارها بوده‌ایم که این خود گاهاً به دسترسی به سیستم کاربران و نشت اطلاعات می‌انجامید. نکته دیگر در بحث همه‌گیری کرونا، عدم حضور کافی کارشناسان و پشتیبان‌های فنی برای انجام به‌روزرسانی‌ها، رفع نقص‌های سیستم‌ها و اعمال وصله‌های امنیتی بوده که بعضاً به بهره‌برداری موفق از آسیب‌پذیری‌ها و نقص‌های امنیتی سازمان‌ها و شرکت‌ها منجر شده و بدیهی است که در مواردی اطلاعات آنها نیز نشت پیدا کرده‌اند.



تصویر ۲. هزینه‌های ناشی از نشت اطلاعات از سال ۲۰۱٤ تا ۲۰۱۹

باید به این نکته توجه داشت که نشت اطلاعات هزینه‌هایی جبران ناپذیر برای یک سازمان یا شرکت خواهد داشت که در ذیل به چند مورد اشاره می‌شود:



- تخریب شهرت یک شرکت یا سازمان
- خسارت مالی به شرکت یا سازمان با پرداخت جریمه
- مشکل در کسب و کار و عدم اعتماد مشتریان و از دست دادن سهم بازار
- از دست دادن اعتماد سرمایه‌گذاران
- از دست دادن گواهینامه‌ها، لایسنس‌ها، رتبه بندی‌ها و غیره
- کاهش درآمد و حتی ورشکستگی

راهکارهای جلوگیری از نشت اطلاعات

در چند سال اخیر بر اساس آمارها، نشت‌های اطلاعات بسیاری رخ داده است که در فصل‌نامه تخصصی امنیت سایبری ویرا شماره ششم که در تابستان ۱۳۹۹ منتشر شده است در یک مطلب با نام نشت اطلاعات، آمار دقیق و توضیحات کاملی در خصوص بزرگ‌ترین نشت‌های اطلاعات تاریخ آورده شده است که توصیه می‌شود این مطلب مطالعه شود. در این بخش چند مورد از این آمار به صورت خلاصه نشان داده می‌شود:

- نشت اطلاعات از Yahoo در سال ۲۰۱۳ با سه میلیارد رکورد نشت شده
- نشت اطلاعات بزرگ‌ترین پایگاه‌داده بیومتریک جهان در سال ۲۰۱۸ با یک میلیارد رکورد نشت شده
- نشت اطلاعات شرکت مالی First American در سال ۲۰۱۹ با ۸۸۵ میلیون رکورد نشت شده
- نشت اطلاعات Verfications.io در سال ۲۰۱۹ با ۷٦۳ میلیون رکورد نشت شده
- نشت اطلاعات Facebook در سال ۲۰۱۹ با ۵٤۰ میلیون رکورد نشت شده

همچنین در سال‌های اخیر چند موارد نشت اطلاعات در داخل کشور رخ داده است که رسانه‌ها نیز در اغلب موارد اخبار آن را منتشر کرده‌اند که آخرین مورد آن در مهر ماه ۱۳۹۹ نشت اطلاعات مربوط به سازمان بنادر و دریانوردی بوده است که اطلاعات دقیقی از میزان اطلاعات نشت شده (همانند دیگر موارد نشت اطلاعات داخلی) در دست نیست!!!

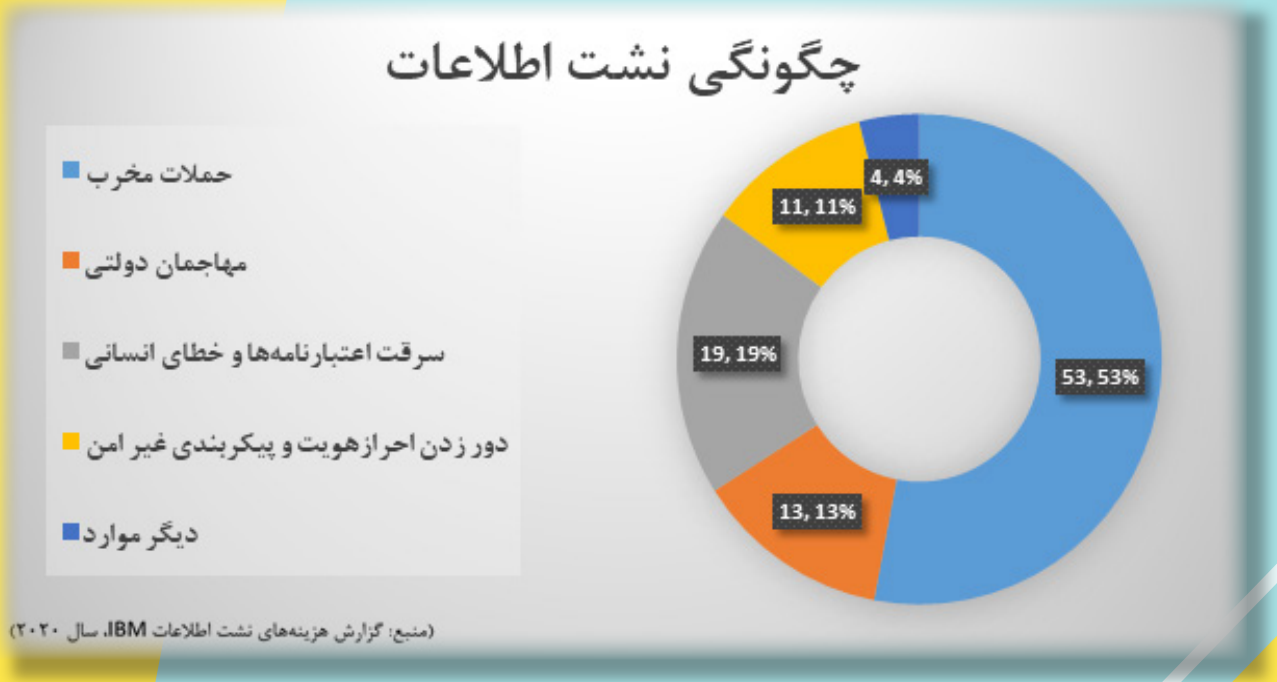
بر طبق آمار Forbes که در گزارش سال ۲۰۲۰ خود منتشر کرده است در ۱۰ سال اخیر در حدود ۳۰۰ مورد نشت اطلاعات با بیش از ۱۰۰ هزار رکورد داده نشت شده، وجود داشته است. همچنین بر طبق این گزارش ٤/١ میلیارد رکورد داده فقط در ٦ ماه اول سال ۲۰۱۹ نشت شده است. بر اساس بررسی‌ها و تحلیل‌هایی که در این گزارش آمده است، کشف نشت اطلاعات در سازمان‌ها بطور متوسط ۲۰۵ روز بعد از وقوع صورت گرفته و بیشترین زمان طی شده برای کشف نشت اطلاعات ۲۹۸۲ روز بوده است و نکته فاجعه‌بار این است که برخی از نشت‌های اطلاعات هرگز کشف نمی‌شوند.

نشت اطلاعات چگونه انجام می‌گیرد؟

با بررسی دلایل نشت اطلاعات می‌توان پی برد که دلایل و بسترهای مختلفی برای نشت اطلاعات وجود دارد. در ادامه به صورت موردی دلایل مختلف چگونگی رخ دادن نشت اطلاعات ذکر شده‌اند:

- حملات هدفمند مانند حملات باج‌افزاری و فیشینگ
- عدم طبقه‌بندی اطلاعات و بی‌نظمی در ساختار آنها
- پیکربندی غیراصولی و ناامن تجهیزات، نرم‌افزارها و سرویس‌ها
- عدم استفاده از تجهیزات سخت‌افزاری و محصولات نرم‌افزاری امنیتی
- مشکلات مدیریتی درخصوص عدم درک صحیح در ارتباط با نشت اطلاعات
- نبود سیاست درخصوص عدم قرارگیری اطلاعات حساس در شبکه‌های اجتماعی
- سیستم‌های احراز هویت و کنترل دسترسی ضعیف
- سوءاستفاده از نقاط ضعف سیستم‌های نرم‌افزاری
- دانلود ناخواسته برنامه‌های مخرب
- عدم رمزنگاری صحیح اطلاعات
- استفاده از کلمات عبور ضعیف
- سیستم‌های وصله نشده
- کاربران و کارمندان ناراضی

البته باید توجه داشت که طبق تحلیل نشت‌های اطلاعات صورت گرفته، در برخی از موارد چندین مورد از موارد ذکر شده در یک سازمان به صورت همزمان وجود داشته است که هر کدام به صورت جداگانه می‌توانست باعث نشت اطلاعات شود!!! در تصویر ۳ آماری از چگونگی نشت اطلاعات بر طبق گزارش هزینه‌های نشت اطلاعات IBM در سال ۲۰۲۰ نشان داده شده است.



تصویر ۳. چگونگی نشت اطلاعات

راهکارهای جلوگیری از نشت اطلاعات

در بخش قبل به چگونگی نشت اطلاعات و دلایل آن اشاره شد. بدیهی است که هرگونه راهکار برای جلوگیری از نشت اطلاعات کاملاً مرتبط با دلایل ذکر شده است. در ادامه راهکارهایی که ممکن است برای جلوگیری از نشت اطلاعات مفید باشند، ذکر شده‌اند:

- استفاده از ایمیل‌های سازمانی برای ارسال نامه‌ها و اسناد
- آموزش پرسنل و کارشناسان در حوزه امنیت اطلاعات
- کنترل و مانیتور کردن ترافیک شبکه توسط UTM
- شناسایی دارایی‌های اطلاعاتی موجود درسازمان
- رمزنگاری صحیح اطلاعات
- طبقه‌بندی و کلاس‌بندی اطلاعات
- مدیریت رسانه‌های انتقال فایل
- استفاده از کلمات عبور مطمئن
- اعمال وصله‌ها و به‌روزرسانی‌های امنیتی
- استفاده از ابزار امنیتی مناسب با پیکربندی امن
- تعریف سیاست‌هایی برای هرگونه استفاده و انتقال اطلاعات
- انجام مداوم ارزیابی‌های امنیتی برای شناسایی نقاط ضعف در سازمان

طبقه‌بندی اطلاعات یکی از مهمترین اصول برای جلوگیری از نشت اطلاعات است. برای طبقه‌بندی به صورت کلی می‌توان اطلاعات را به دسته‌های دسترسی عمومی، فقط دسترسی داخل سازمان و محرمانه طبقه‌بندی کرد. همچنین باید توجه داشت که ایمیل یکی از بسترهای اصلی سرقت و نشت اطلاعات است. پیکربندی امن سرور ایمیل، یکی از پازل‌های مهم جلوگیری از نشت اطلاعات است. باید دقت داشت که مدیریت اطلاعات صرفاً نصب و استفاده از چندین تکنولوژی امنیتی نیست. آگاهی‌رسانی و آموزش کاربران و کارمندان کلید اصلی حفاظت از اطلاعات است. توجه به این نکته با اهمیت است که رمزنگاری اطلاعات نیز می‌تواند راهکار مهمی برای جلوگیری از نشت اطلاعات باشد و این رمزنگاری هم بر روی درایوها، فایل‌ها و رسانه‌های انتقال میبایستی صورت گیرد.

برای رمزنگاری در سیستم‌عامل ویندوز از Axcrypt، Eset endpoint encryption pro، Bitlocker و Veracrypt می‌توان بهره برد. مانیتور کردن Realtime ترافیک شبکه توسط UTM در سازمان دارای اهمیت بسیاری است که می‌توان از UTMهای Juniper ، Cisco Meraki، Fortigate، Sophos و Checkpoint برای این منظور استفاده کرد. یکی دیگر از راهکارهای جلوگیری از نشت اطلاعات، مدیریت وصله‌ها در سیستم‌عامل است که در سیستم‌عامل ویندوز می‌توان از Comodo Patch Manager، SolarWinds Patch Manager و Symantec Patch Management Solution استفاده کرد. برای مدیریت و کنترل دسترسی‌ها AWS Directory Service، LDAP Account Manager، Apache Directory و SolarWinds Access Rights Manager می‌تواند مورد استفاده قرار گیرد.

توضیحاتی درخصوص DLP

با بررسی تخصصی راهکارهای مختلف جلوگیری از نشت اطلاعات می‌توان به این نکته رسید که مهمترین راهکار سیستم DLP یا Data Loss Prevention است. DLP ایجاد یک سیستم، سیاست و راهکار مناسب برای محافظت اطلاعات از هرگونه نشت غیرمجاز اعم از عمد و غیرعمد است. DLP صرفاً یک ابزار امنیتی خاص نیست، بلکه مجموعه‌ای از تکنولوژی‌ها و روش‌ها برای شناسایی محتوا، ردیابی فعالیت‌ها و مسدود نمودن انتقال بدون مجوز اطلاعات حساس می‌باشد که به جلوگیری از نشت اطلاعات می‌انجامد. سیستم DLP برای کشف، مانیتور، حفاظت و مدیریت اطلاعات سازمانی مورد استفاده خواهد بود. انواع فناوری‌های DLP به صورت زیر است:

- DLP برای داده‌های در حال استفاده
- DLP برای داده‌های در حال انتقال
- DLP برای داده‌های ذخیره شده

شرکت‌های مختلفی دارای محصول DLP هستند اما مطرح‌ترین آنها Forcepoint، Symantec و Digital Guardian هستند که پیشنهاد می‌شود یکی از این سیستم‌ها مورد استفاده قرار گیرد. همچنین در فصل‌نامه تخصصی امنیت سایبری ویرا شماره سوم که در پاییز ۱۳۹۸ منتشر شده است در یک مطلب سیستم DLP به طور کامل بررسی و توضیح داده شده است که توصیه می‌شود این مطلب مطالعه شود.

لینک فصل‌نامه شماره سوم

مرکز آپا دانشگاه کردستان
www.cert.uok.ac.ir