==========

==========

قبل خوندن این مطلب، سعی کنین اول کمی با برنامه‌نویسی آشنا باشین و بعدش رمزنگاری رو بخونین (می‌تونین از آموزش‌های من که به زودی در گیتهاب[1] قرار می‌گیره استفاده کنین) و بعدش این داک.

اول از همه باید با یه سری کلمات آشنا شیم که بهتر بتونیم حرف همو بفهمیم:

«داده» یا «دیتا» چیه؟

به طور کلی، داده یعنی اطلاعات. یعنی دانسته‌ها و همه چیز درواقع.
مثلاً اطلاعات مربوط به هواشناسی، داده هواشناسی هست.
ریز نمرات و اطلاعات مربوط به نمرات دانشجوها، داده‌های نمرات دانشجو هاست.
هرچیزی دراقع! حتی این کلمات من یه سری داده هست.
درواقع پشت هر چیز توی کامپیوتر یه سری متن وجود داره. کامپیوتر اون متن‌های خاص رو می‌بینه و با استفاده از اون، فایل رو میفهمه و براتون نمایش میده.
گنگ بود یکم؟ ایرادی نداره! بریم ببینیم اینکه پشت هر چیز یه سری متن هست یعنی چی؟
من یه عکس دارم. گفتم پشت هر چیز یه سری متن هست درسته؟ خب چجور اون متن رو ببینم؟ فکر کنین یکم!
+ عکس رو به جای اینکه با حالت معمول باز کنم، با یه چیزی که بتونه متن رو بخونه باز میکنم! که ببینم پشت این عکس چه متنی هست! خب چه چیزی متن رو میخوند؟ نرم‌افزارهای Text editor مثل Notepad! خب پس به جای double click روی یه عکس، روش کلیک راست می‌کنیم که گزینه‌های مختلف رو ببینیم. بعدش روی گزینه open with کلیک می‌کنیم. بعد میریم توی لیست برنامه‌ها و یکم میایم پایین و می‌زنیم show more و برنامه notepad رو انتخاب می‌کنیم. حالا عکس ما با نوت‌پد باز شد.
یه سری متن عجیب غریب نمایش داده شده؛ درسته؟!
خب کامپیوتر این متن‌ها رو میبینه و میفهمه باید یه تصویر رو فلان‌طور نشون بده. درواقع اینا همش داده هستن.

اجراشدن دستور:

1 https://github.com/kamal331/Books/
البته می‌تونین فعلاً قسمت رمزنگاری رو کنار بگذارین و بعداً بخونینش. چون جاهایی که نیاز بوده، توضیح دادم دربارش.

به طور کلی ما به کامپیوتر دستور می‌دیم که الآن اینکار کن. اینجا این کار کن. اونجا اون کار کن. وقتی کامپیوتر میاد دستورات ما رو اجرا می‌کنه، بهش می‌گیم که دستورات اجرا شد.

درواقع همه چیز توی کامپیوتر به وسیله اجرای دستور انجام میشه. مثلاً:

۱. یک رمز عبور بگیر.
۲. چک کن ببین رمز عبور درسته یا نه؟
۳. اگر درسته، اجازه بده وارد شه.
۴. وگرنه پیغام خطا بده.

همه اینا دستورن. پشت برنامه‌ها هم دقیقاً همیناس. همش اینطوریه. پس وقتی می‌گیم یه برنامه اجرا شده یا دستورات اجرا شدن، یعنی این چیزا.

اینترنت و سرور:

خب به طور کلی اینترنت یه فضاییه که همه با هم درارتباطن. یعنی من می‌تونم به شما پیام بدم. شما پیام منو می‌گیری. یا مثلاً وارد یه سایت میشی. یه ویدیو نگاه می‌کنی.

دیدین بالای صفحه مرورگرتون، یه آدرسی نوشته شده؟ مثلاً نوشته:

https://en.wikipedia.org/wiki/Ariobarzanes_of_Persis

این یعنی مثلاً فایرفاکس گرامی، برو به وبسایت «https://en.wikipedia.org» بگو که بره توی پوشه «wiki» و بعدش بره متن «Ariobarzanes_of_Persis» رو برام بیاره. درواقع به منی که درخواست می‌دم میگن «کاربر» یا «clinet» و به کامپیوترهایی که منتظر درخواستن که جواب پس بدن (مثل کامپیوترهای شرکت گوگل)، می‌گن «سرور».

باگ چیه؟

به صورت خیلی خلاصه، مشکلات یه برنامه رو می‌گن باگ. مثلاً روی یه دکمه کلیک می‌کنی، صفحه یهو سیاه میشه. بهش میگن اینجا باگ داره. یعنی مشکل داره.

حالا فرض کنین یه خونه دارین. کسی که خونه رو ساخته، هر چند وقت یکبار میاد یه نگاه میندازه بهش. مثلاً نگاه میکنه میبینه عه لوله‌ی آبش خرابه. (باگ داره) بعد به شما میگه میخوای رایگان تعمیرش کنم؟ شما میگی آره تعمیر کن یا میگی نه دستش نزن که خراب نشه؟ برنامه‌های گوشیتون هم مثل یه خونه هستن. کسی که اون برنامه رو نوشته، میاد چک میکنه که آیا اون برنامه مشکلی داره یا نه. اگر مشکل داره، مشکل رو با نسخه‌ای جدید از اون برنامه حل می‌کنه. حالا برای شما توی یه برنامه‌ای که تحت عنوان فروشگاه برنامه‌ها هست یا برنامه‌ی گوگل پلی استور، یه پیغام میاد. میگه نسخه‌ی جدید فلان برنامه اومده. میخوای بروزرسانی یا آپدیتش کنم؟ یعنی نسخه‌ی جدیدش رو نصب کنم؟

پس فلسفه‌ی آپدیت به طور کلی حل مشکلات گذشته هست. اما گاهی برنامه‌نویس ممکنه قابلیت جدیدی هم اضافه کنه. مثلاً رنگ برنامه رو عوض کنه. اینکه شما میگین چرا هی واتسپ رو آپدیت می‌کنیم چیز جدیدی نداره، به خاطر اینه که واتسپ چیز جدید اضافه نمی‌کنه. فقط printتوی آپدیتاش، مشکلات رو برطرف میکنه ولی برعکس تلگرم هم مشکلات رو برطرف میکنه و هم فکر میکنه چجور این برنامه رو تغییر بدم که بهتر شه؟

مثلاً خونه ممکنه قفلش شکسته باشه یا یه پنجره‌اش باز باشه و فرد بد وارد خونه شه. توی برنامه‌ها هم مشابه همین هست. ممکنه یه جای برنامه مشکل داشته باشه و بشه از اون طریق دستگاه شما رو هک کرد. پس بروز بودن میتونه جلوی این رو بگیره. وقتی شما برنامه‌هاتون همیشه آپدیت باشن، هر مشکلی وجود داشته باشه، زود توسط اون آپدیت و بروزرسانی برطرف میشه. مثل همون مثال پنجره‌ی باز. اگر برنامتون رو آپدیت کنین، اون پنجره رو میبندین و کسی نمیتونه بیاد. پس اینکه همیشه میگن برنامه‌هاتون آپدیت باشه، برای اینه که یه سری مشکلات امنیتی وجود داره که با آپدیت برطرف میشه. مثلاً قبلاً واتسپ یه مشکلی داشت که اگر یه نفر فقط به شما زنگ میزد، یعنی فقط تماس خالی، هک میشدین. اما این با آپدیت برطرف شد. اونایی که آپدیت نکرده بودن، در معرض خطر بودن. پس برای همین نیازه برنامه‌هاتون و سیستم‌عاملتون همیشه آپدیت باشه.

امنیت چیه؟

به طور کلی در امنیت، ما تلاش می‌کنیم که کسی دسترسی غیرمجاز نداشته باشه. یعنی کسی که مجاز نیست، نتونه به اطلاعات و داده‌های ما دسترسی داشته باشه. مثلاً توی دنیای فیزیکی، ما دَر رو قفل می‌کنیم که کسی غیرمجاز وارد خونه ما نشه که حالا بعداً بیاد خرابکاری کنه یا چیزی برداره.

توی دنیای کامپیوتر هم امنیت همینه. یعنی درواقع ما تلاش می‌کنیم اطلاعات از دید کسایی که نباید ببینن، پنهون، محرمانه و غیرقابل دسترسی و تغییر باشه.

رمزنگاری چیه؟

فرض کنین من می‌خوام یه نامه برای یکی بنویسم. خب باید با پست ارسال شه. من از کجا بدونم یه نفر وسط راهی که پست داره نامو می‌بره، نامم رو باز نمی‌کنه که بخونتش؟ هیچ تضمینی نیست که کسی نخونتش!

اینجا بحث رمزنگاری پیش میاد. یعنی من بیام یه متن رمزی بنویسم که فقط خودم و فرد مقابلم بفهمه یعنی چی.

مثلاً به جای «سلام» بنویسم «مالس» (درواقع اومدم از انتها به ابتدا نوشتم)

حالا چون صرفاً خودم و اون فرد مقابلم می‌دونیم چجور بخونیمش، می‌فهمیمش.

درواقع من تلاش کردم که از داده‌های نامم با رمزکردنش حفاظت کنم. این دقیقاً توی کامپیوتر استفاده میشه. یعنی همون HTTPS ای که شما استفاده می‌کنین، برای اینکه کسی وسط راه نتونه رمزتون رو بفهمه، به شیوه رمزشده متنا رو می‌فرسته. منتها رمزش خیلی خیلی قوی هست.

ما باید از کجا محافظت کنیم؟

ببینین امنیت مثل یه زنجیره.

- خب به نظرتون زنجیر از کجاش پاره میشه؟ از جایی که زنگ زده یا جای سالمش؟

+ مطمئناً از جایی که زنگ زده. یا درواقع از ضعیف‌ترین و آسیب‌پذیرترین حلقه‌اش.

پس درواقع آدم بدا همیشه سعی می‌کنن که از جایی که در دسترسه و آسیب‌پذیره به سیستم نفوذ کنن و هدف ما تقویت حلقه‌های این زنجیره.

به قسمتای آسیب‌پذیر، آسیب‌پذیری یا «vulnerability» و به بهره‌برداری و استفاده از اون برای نفوذ، اکسپلویت «exploit» گفته میشه.

درواقع همونطور که یه سری آدم توی دنیای عادی کارشون ساخت دزدگیر، قفل و... هست، یه سری افراد هم توی دنیای مجازی، کارشون امن‌کردن چیزاس.

درواقع از سخت‌افزار شروع کنیم، بیایم بالاتر برسیم به ویندوز، بیایم بالاتر برسیم به مرورگر بیایم بالاتر برسیم به مودم خونه و... همه و همه یه حلقه از زنجیر ما هستن. درواقع شما توی یکی از این‌ها تخصص کسب می‌کنی و میری اون حلقه رو امن می‌کنی. یا تلاش می‌کنی که از جاهای آسیب‌پذیر (vulnerability ها) به سیستم نفوذ کنی.

بیایم از سخت‌افزار شروع کنیم.

پایه‌ای‌ترین چیز یه کامپیوتر سخت‌افزاره. همه چیز روی سخت‌افزار اجرا میشه. اگر سخت‌افزار امن نباشه، عیناً مثل اینه که شالوده خونتون رو بد بنا نهادن. خب یه روزی فرو می‌ریزه؛ یه روزی اطلاعات و دیتای شما به واسطه ناامن‌بودن سخت‌افزار آسیب می‌بینه.

قبول دارین که سخت‌افزار یه قطعه الکترونیکیه؟ قبول دارین که قطعات الکترونیکی موقع کار و عبور جریان برق ازشون یه سری نویز رو دور و برشون منتشر می‌کنن؟

حالا من میتونم بیام با یه سری دستگاه خاص، اون نویزها رو ضبط کنم و الگوهاش رو پیدا کنم. مثلاً میگم اگر شکل موج نویز اینطوری بود، احتمالاً حرف A هست. اگر شکل موج اونطوری بود، حرف B و... . همینطوری تحلیل می‌کنم و می‌تونم کم‌کم با نویزهای تولیدی، اطلاعاتی که توی سیستم داره تولید میشه رو کشف کنم. مثلاً طرف که داره پسورد وارد می‌کنه، من پسوردش رو بفهمم![۲]

یا مثلاً با استفاده از مقدار انرژی مصرف CPU و **تفاوت** انرژی مصرفی در کارها و ورودی‌های مختلفی که به سیستم میدیم، بتونم بفهمم که توش چه خبره![۳]

درواقع به جای اینکه مثلاً مستقیم به الگوریتم رمزکردن داده‌ها حمله کنیم و تلاش کنیم رمز و پیدا کنیم، میایم از یه کانال جانبی سعی می‌کنیم اطلاعات و کلید رمزنگاری (شما بخونید پسورد) رو پیدا کنیم که با کلید و پسورد بتونیم رمز رو باز کنیم.[۴]

- ما به عنوان یه مهندس امنیت چیکار می‌کنیم؟

+ مثلاً شما سخت‌افزار رو جوری طراحی می‌کنین که امکان استخراج اطلاعات وجود نداشته باشه. (گفتیم از اطلاعاتی مثل نویز میشه به عملکرد داخلی پی برد) چیزای مهم رو توی سخت‌افزارهای خاص (مثل یه صندوقچه) قرار میدین که کسی بهش دسترسی پیدا نکنه.[۵] (چیزایی مثل کلید رمزنگاری[۶])

- خب نیاز این قسمت چیا هست؟

---

۲ این مثال خیلی ساده‌شده بود! درواقع مکانیزم پیچیده‌تره. صرفاً خواستم دید بدم! درواقع یکم تخصصی‌تر بگیم، کلید رمزنگاری رو می‌تونم پیدا کنم:
RSA Key Extraction via Low-Bandwidth Acoustic Cryptanalysis:
https://www.tau.ac.il/~tromer/papers/acoustic-20131218.pdf -Daniel Genkin, Adi Shamir, Eran Tromer

3 Differential Power Analysis: http://gauss.ececs.uc.edu/Courses/c6055/lectures/SideC/DPA.pdf

۴ به این نوع حملات، side-channel attack گفته میشه.

۵ یه دیدی شبیه TPM و چیپ Pluton.

۶ نمی‌دونین کلید رمزنگاری چیه؟ فکر کنین منظورم پسورد و رمز هست.

+ درک سخت‌افزار! تا وقتی شما ندونین سخت‌افزار کامپیوتر چجور کار می‌کنه، چه ارتباطاتی با هم دارن، چجور اطلاعات تبادل میشه و...، چه‌جوری می‌خواین امنیتش رو متوجه شین؟! پس قدم اول درک و کسب دانش هست. (درواقع شما باید بفهمین که CPU چطور کار می‌کنه، راه‌های انتقال داده بین CPU و بقیه جاها از چه طریق و چطوره؟ و...)

ببینین درواقع هیچ راه میونبری نیست که من فلان چیزو یاد بگیرم هکر شم! هرکی بهتون گفت بیا این یاد بگیر هکر شو، اون فرد داره سر شما رو شیره میماله! امنیت یعنی من یه درک دارم که سیستم چه‌جوری کار می‌کنه، حالا چیکار کنم که اونجور کار نکنه. چیکار کنم که به مشکل بر بخوره؟ چیکار کنم بتونم از سیستمی که داره کار می‌کنه، یه اطلاعاتی رو بکشم بیرون. حالا که مشکل رو یافتم، چیکار کنم که حلش کنم؟

درواقع شما باید دیدتون رو فراتر ببرین. هک وایفای و هک اینستا و اینا به هیچ دردتون نمی‌خوره. الآن جوونید سرتون گرمه و دنبال هیجانین و فکر می‌کنین وای فلان کار کردم چه خفنم! نه نیستین! اینا براتون نون و آب نمیشه! شما باید یه شغل واقعی پیدا کنین. یه شغلی که وقتی چهل سالتون شد، بتونین باهاش زندگیتون رو بچرخونین. اونجا دیگه کسی نیست خرجیتون رو بده. خودتون باید خرجیتون رو بدین! هک اینستا و هک وایفای و این چیزا یه سری کار چیپ و مسخره هست که هم غیرقانونیه و هم هیچ آینده‌ای نداره. به جاش دانش کسب کنین تا مهندس امنیت شین. کسی که کارش درسته، قانونی و انسانیه و پول خیلی خوبی هم در میاره.

پس درواقع ویدیوهای اینطوری که هک اینستا و اینا و یا ویدیوهایی با تیتر «بیا هکر شو»، «بیا بهت یاد بدم وبسایت هک کنی»، «می‌خوای هکر شی؟ بیا پایتون یادت بدم هکر شو» و... احتمالاً صرفاً چهارتا چیز هیجانی می‌گن که در نهایت شغل واقعی از توش در نمیاد و صرفاً در یه سطح کوتاهی می‌مونین. بلکه اصولی برین جلو و کتاب‌های درست بخونین. هر کتابی که یکی نوشته که امنیت فلان، قرار نیست خوب باشه. مخصوصاً کتاب‌های فارسی که ترجمه‌های به شدت بدی دارن یا بعضا صرفاً کاپی ناشیانه‌ای از کتاب‌های خارجی هستن و یا توضیحات درستی ندارن.

بله مسیر راه مهندس امنیت‌شدن طولانیه و دو سه ماهه نمیشه رسید بهش، همیشه هم باید بروز باشین. درواقع توی کامپیوتر همیشه باید بروز باشین. اگر بروز نباشین، اونایی که بروزن میان جاتون رو می‌گیرن. نیاز به هوش مصنوعی که بیکارتون کنه هم نیست! بلکه توسط افرادی که دانش بیشتری دارن، جایگزین می‌شین.

رشته کامپیوتر خیلی سریع عوض میشه. پس شما باید همیشه در حال مطالعه باشین، همیشه چیز یاد بگیرین. درواقع به قول جادی: ((بیشترین‌چیزی که باید یاد بگیرین، یادگرفتنه.))

اما خب وقتی به اونجا برسین، یه فردی هستین که نه نگرانین هوش‌مصنوعی بیکارتون کنه، نه نگرانین که کار گیرتون نیاد و درامدتونم واقعاً بالاست. میانگین درآمد یه متخصص امنیت، خیلی وقتا از برنامه‌نویسا بالاتره![7] (البته مسیرشم کمی سخت‌تر و طولانی‌تره)[8]

7 https://survey.stackoverflow.co/2023/#section-salary-salary-by-developer-type
8 https://survey.stackoverflow.co/2023/#section-salary-salary-and-experience-by-developer-type

مثال:

دیدین وقتی با مداد یه چیزی روی کاغذ نوشتین، بعدش که بخواین با پاک‌کن پاکش کنین، یکم جای نوشته‌ها روی کاغذ میمونه؟

توی دنیای کامپیوتر هم همینه. وقتی شما یه چیزی رو پاک می‌کنی، درواقع صرفاً از طریق سیستم‌عامل (مثل ویندوز) در دسترس نیست. ولی هنوز جاش روی هارد (HDD) شما هست![۹] پس میشه با یه سری برنامه خاص قسمتای مختلف هارد رو خوند و اطلاعاتی که فکر کردین پاک شده رو بازیابی کرد. این اطلاعات ممکنه اطلاعاتی حساس باشه.

حالا نحوه حل این مشکل چیه؟

دیدین بخواین جای یه چیز که پاک کردین نمونه، باز میاین با مداد روش هِی چیزای الکی می‌نویسین و پاک می‌کنین که معلوم نشه نوشته اولی چی بود؟

توی دنیای کامپیوتر هم همینه. شما یه درک از سخت‌افزار داری که اطلاعات چه‌جور روی هارد نوشته میشن. پس حالا می‌تونی یه سری الگوریتم بنویسی که بیاد چیز میز روی اون قسمت پاک‌شده فایل بنویسه و هی پاک کنه که اثر فایل اصلی مشخص نشه! بله یه سری افراد بودن که این الگوریتما رو نوشتن و نرم‌افزارهایی هم نوشتن که اون الگوریتم رو اجرا کنه.

دیدین؟ شما نیاز دارین بدونین هارد چه‌جور کار می‌کنه، که بعدش بتونین مشکل رو پیدا و بعد رفعش کنین. همه چیز توی این سه مرحله خلاصه میشه: ۱- دونستن نحوه کار ۲- پیداکردن مشکل با خلاقیت و یا یافتن اینکه چیکار کنم که سیستم درست کار نکنه ۳- رفع اون.

یه نکته دیگه!

شما در هر حوزه‌ای بخواین برین، برنامه‌نویسی رو باید به چشم یه ابزار ببینین. همینجا هم گفتم الگوریتم پاک‌کردن امن رو با برنامه‌نوشتن و اجراش کردن. پس مثل ضرب و تقسیم که بلدین، برنامه‌نویسی هم باید بلد باشین. درواقع برنامه‌نویسی ابزار شما برای به کار بردن دانش شماست. یعنی یه دانشی داری که با برنامه‌نویسی اون رو پیاده‌سازی می‌کنی.

مثال:

شرکتا همیشه سعی می‌کنن که سخت‌افزارها رو سریع‌تر کنن که کامپیوترهای شما سریع‌تر شه. یعنی سعی می‌کنن که راهکارهای مختلف رو پیدا کنن که سرعت رو افزایش بده. فرض کنین کد برنامه اینه[۱۰]:

۱. یه عدد از کاربر بگیر و بذار توی متغیر (که می‌دونیم متغیر توی رم هست).
۲. عدد رو بیار و چک کن ببین آیا عدد بزرگ‌تر از ۱۰ هست یا نه؟ اگر آره، حاصل ۱۰ * ۲ رو حساب کن.

---

۹ البته برای SSD هم قابلیت بازیابی هست ولی با تفاوت:
https://eraser.heidi.ie/do-solid-state-drives-ssds-really-destroy-data/

۱۰ این مثالی که می‌زنم، مثال خیلی خیلی ساده‌ای هست و دقیق نیست! دقیق‌ترش رو سرچ کنین speculative execution. درواقع من صرفاً برای سادگی یه مثال خیلی خیلی ساده‌ای رو زدم.

۳. حاصل رو چاپ کن.

یا به صورت کدی:

```
num = int(input())
if num > 10:
    result = 10 * 2
    print(result)
```

خب کامپیوتر میگه که من اول باید برم متغیر num رو از ممووری بیارم که ببینم آیا بزرگ‌تر از ۱۰ هست یا نه؟ اگر بود باید ۱۰ رو ضربدر ۲ کنم. به طور کلی، آوردن متغیر از ممووری زمان‌بره. یعنی طول می‌کشه. من تا وقتی که داره میاد، میام ۱۰ ضربدر ۲ رو حساب می‌کنم که حاصل رو آماده داشته باشم و اگر دیدم بزرگ‌تر از ۱۰ بود، حاصل آماده باشه که بتونم بذارم توی result. نخوام صبر کنم که اول متغیر بیاد. میتونم چیزای جلوتر رو زودتر آماده کنم که زودتر انجام شه. بیکار نشینم که صبر کنم هر وقت num اومد حساب کنم. تا زمانی که بیکارم حساب رو انجام میدم که جلو بیوفتم.

بذارین یه مثال بزنم! فرض کنین من منشی یه دکترم. می‌بینم که پنج دفعه قبلی که وارد مطب شدی، می‌خواستی پروندتو بهت بدم که ببری پیش دکتر. خب به نظرتون کدوم منطقی‌تره؟

۱- منتظر بمونم شما بِرِسی کنار میز من و من تازه بگردم دنبال پروندت.

۲- تا از دور دیدمت، بگردم دنبال پروندت که تا رسیدی، پروندتو بدم بهت و زیاد منتظر نمونی.

قاعدتاً حالت دوم بهتره! چون از تأخیر جلوگیری می‌کنه.

شاید شما این دفعه کار دیگه‌ای داشته باشی، اما شانس اینکه بازم پروندتو بخوای زیاده و اگر من پروندت رو آماده داشته باشم، خیلی زود بهت میدم و کارا خیلی خیلی سریع‌تر پیش میره.

درواقع CPU هم همیشه می‌خواد از تأخیر جلوگیری کنه. یعنی میگه آقا من الآن خط ۲ کد هستم باید چندتا چیز رو با هم جمع بزنم. حالا تا وقتی که متغیرا از حافظه میان، یکم طول میکشه. خب من بیکار نشینم! برم خط بعدی هم اجرا کنم که یکم جلو بیوفته کارا.

به این جلوجلو اجراکردن، «Out-of-order execution» گفته میشه. یعنی برخلاف خط اصلی برنامه جلوجلو و برخلاف صف و اولویت اجرا، چیزی اجرا شده. حالا این چیزا ممکنه باعث شه که یه سری دستورات اشتباهی جلوتر اجرا شن که اطلاعات حساس سیستم من رو لو بدن. درواقع خواستن سرعت زیاد شه ولی از اونور امنیت کم شده.[۱۱]

مثال پیشرفته:

فرض کنین من یه برنامه‌ای می‌خوام بنویسم که یه سری داده (ages) رو دونه‌دونه بهش بدم (مثلاً سن افراد)، بعد بیاد تعداد سن‌های بالای ۱۸ رو بهم بده.

```
for i in range(len(ages)):
    if ages[i] > 18:
```

---

۱۱ درواقع توی دنیای کامپیوتر همه چیز یه نوع بده و بستون (trade-off) هست. یعنی شما گاهی امنیت زیاد می‌کنی ولی از اونور سرعت کم میشه. سرعت زیاد می‌کنی، امنیت کم میشه. سرعت زیاد می‌کنی، انرژی مصرفی سیستم زیاد میشه. خلاصه همه چیز بده و بستونه. شما به عنوان یه مهندس باید تصمیم بگیرین که آیا سرعتی که زیاد میشه میرزه به امنیتی که کاهش پیدا کرده یا نه؟

```
        count = count + 1
```

توضیح: قاعدتاً اول تعداد برابر صفره. بعد سن‌ها رو میدیم بهش. (اینکه چه‌جوری بهش میدیم رو فعلاً کاری نداشته باشین!) بعدش اگر هر دونه بزرگ‌تر از ۱۸ بود، میگیم count جدید ما برابر count قبلی بعلاوه یک هست. (یکی رو اضافه می‌کنیم بهش)

یعنی درواقع هر دفعه یکی از سنا میاد و اگر بیشتر از ۱۸ بود، یکی به count اضافه میشه.

خب به نظرتون این کد در دوحالت زیر، چه‌زمانی سریع‌تره؟

```
1 - 3 - 4 - 6 - 8 - 10 - 20 - 21 - 24 - 25 - 29 - 30
21 - 4 - 29 - 3 - 30 - 8 - 10 - 21 - 1 - 6 - 25 - 20
```

حالت اول مرتب شده هست، حالت دوم هم نامرتب. (توجه داشته باشین که مرتب‌کردن اعداد، خودش مقداری زمان می‌بره.)

- خب اینکه خیلی سادس! مطمئناً حالت اول سریع‌تره! چون نیاز نیست من یه دور اول مرتبش کنم که زمان الکی ببره!

+ خب خب خب :) در نگاه اول آره به نظر میاد حالت اول سریع‌تر باشه، اما درواقع حالت دوم سریع‌تره!

درواقع اگر من از نحوه کار CPU (مغز) کامپیوتر آشنا باشم، می‌دونم که CPU ها یه سری قابلیت دارن که تأخیر رو کاهش بدن.

خب بیایم رو کد. کامپیوتر میرسه به if. خب پیش خودش میگه که نمیدونم که داخل if باید برم یا نه! خط چیکار کنم؟ حدس بزنم که اگر احتمالش زیاده وارد if بشم، خب برم توش. وگرنه دستورای بعد if رو جلو جلو اجرا کنم.

نگاه می‌کنه به قبل می‌گه عه! از ۳ بار قبلی که رسیدم به if، من هر ۳ بار رفتم توش! پس ایندفعه هم شانس بالایی هست که باز بخوام برم توش. برای همین میره دستورای توی if رو جلو جلو حساب می‌کنه.

حالا چرا مرتب‌شده سریع‌تره؟

چون کامپیوتر شروع می‌کنه از اول، دو سه تای اول می‌بینه وارد if نمیشه. اما از یه جایی به بعد، میبینه داره وارد if میشه. پس میگه بار بعدی که رسیدم به if، توی زمانی که شرط داره چک میشه، من بیکار نمیشینم! میرم توی if و چیزای داخلش رو حساب می‌کنم که یکم بیوفتیم جلو.

درواقع به دلیل اینکه یه سری محاسبات جلوجلو انجام میشه، حالت مرتب‌شده سریع‌تره!

ولی توی مرتب نشده، میبینه یه بار میره تو if، یه بار نمیره و اصلاً نمی‌فهمه باید چیکار کنه و کدوم دستورا رو جلو جلو اجرا کنه که سرعت زیاد شه![۱۲]

---

۱۲ به این میگن «branch predictor». مبحث سختیه و فعلاً نمی‌تونین بفهمیدش درست! ولی اگر دانش از رجیستر و کمی زبون C و یا اسمبلی و کمی نحوه ران‌شدن برنامه‌ها و timing attack (که در ادامه باهاش آشنا میشیم) دارین، لینکای زیر رو بخونین:

Why is processing a sorted array faster than processing an unsorted array? -Stackoverflow: https://stackoverflow.com/questions/11227809/why-is-processing-a-sorted-array-faster-than-processing-an-unsorted-array

Spectre Attack: https://spectreattack.com/

قسمت «Variant 1: Exploiting Conditional Branches» رو از مقاله بخونین.

رفع مشکلات سفت‌افزاری خیلی سخته!

نکته بعدی اینه که مثلاً اگر نرم‌افزار آسیب‌پذیر باشه، یه آپدیتی میدن که امن شه ولی اگر سخت‌افزار آسیب‌پذیر باشه، رفعش خیلی مشکله! نمی‌تونیم که تموم سخت‌افزارا و کامپیوترهای دنیا رو جمع کنیم و درستش کنیم! پس از این لحاظ، مبحث سخت‌افزار خیلی مهمه. چون درست‌کردن مشکلاتش خیلی خیلی سخته.

گاهی مجبورن به خاطر یه ایراد سخت‌افزاری، صدتا چک مختلف به صورت نرم‌افزاری انجام بدن که یه وقت چیز بدی اجرا نشه. یعنی به خاطر اینکه مثلاً پنجره خونه شکسته، ده تا لایه میلگرد و محافظ باید جلوی پنجره بگذاریم. یا ده‌تا دوربین مداربسته و دزدگیر بگذاریم. یا درواقع مثال کامپیوتریش این میشه که هی به صورت نرم‌افزاری محافظت خیلی خیلی بیشتر و چک‌های خیلی خیلی بیشتری رو انجام بدیم که چیز درستی بیاد روی سخت‌افزار که اجرا بشه همه این چک‌ها چون دارن کار و چک انجام میدن، سرعت رو خیلی پایین میارن.[۱۳]

بیایم یکم بالاتر و در لایه سیستم‌عامل قرار بگیریم. (درواقع امنیت لایه‌های مختلف داره که حالا شما باید ببینین به کدوم علاقه دارین)

ببینین ما زبون‌های برنامه‌نویسی مختلفی داریم. یکی از اونها مثلاً پایتونه. مدلای دیگه هم داریم. مثلاً زبون‌هایی مثل C/C++ یا Rust که برای نوشتن سیستم‌عامل استفاده میشن.

خب شما چطور می‌خواین در حوزه سیستم‌عامل کار کنین وقتی با این زبونا آشنا نباشین؟ چطور می‌خواین سیستم‌عامل و کدهاشو درک کنین وقتی این چیزا رو بلد نیستیم.

چطور می‌خواین در حوزه سیستم‌عامل کار کنین وقتی خود سیستم‌عامل رو درک نکردین!

پس اول باید این چیزا رو یاد بگیرین.

قسمت «امنیت و مهندسی معکوس» از وبسایت زیر رو بخونین:

https://memoryleaks.ir/how-to-become-a-hacker/

برای امنیت وابسته به زبون‌هایی مثل C/C++ (مثل این قسمت)، باید خیلی روی این زبونا و نحوه استفاده و خالی‌کردن مموری تخصصی‌تر دانش داشته باشین. بدونین چه توابعی[۱۴] مموری رو allocate می‌کنن؟ آیا نیازه خالیش کنین؟[۱۵]

چه مشکلات رایجی هنگام استفاده ازشون پیش میاد؟ چه مشکلات امنیتی این کدا ممکنه داشته باشن؟

Fuzzing[16]

خیلی خیلی ساده شروع می‌کنیم. فرض کنین کد زیر به پایتون رو داریم:

---

13 e.g.

i) https://arstechnica.com/information-technology/2022/06/researchers-exploit-new-intel-and-amd-cpu-flaw-to-steal-encryption-keys/

ii) https://www.zdnet.com/article/beyond-spectre-foreshadow-a-new-intel-security-problem/

۱۴ اگر نمی‌دونین یعنی چی؟ بخونین کد.

۱۵ لینک زیر می‌تونه ایده خوبی بده بهتون که منظورم چیه:

http://www.yolinux.com/TUTORIALS/C%2B%2BMemoryCorruptionAndMemoryLeaks.html

16 https://owasp.org/www-community/Fuzzing

```
num = int(input())
```

توضیح: از درونی‌ترین پرانتزی بخونین. این کد میاد یه عدد صحیح رو از کاربر ورودی می‌گیره. (اول یه چیز ورودی می‌گیره؛ ورودی به صورت string گرفته میشه. پس سعی می‌کنه با «int» تبدیل به int (عدد صحیح)اش کنه.

خب فرض کنین که یه کاربر به جای عدد صحیح، یه کلمه رو به ورودی بده. چه اتفاقی میوفته؟

```
ValueError: invalid literal for int() with base 10: 'hello'
```

بله ارور می‌خوریم و سیستم کِرَش می‌کنه.

درواقع ایده کلی تست‌های fuzzy اینه که ما بیایم داده‌های غیرقابل قبول و invalid رو به ورودی بدیم تا یه جایی سیستم کرش کنه یا دسترسی غیرمجاز بده به ما و مثلاً باعث ورودغیرمجاز یا دسترسی غیرمجاز به داده‌ها بشه.

درواقع من میگم که اینجا که برنامه از من تقاضای یه عدد داره، اگر من عدد منفی بدم چی میشه؟ اگر عدد صفر بدم چی؟ اگر عدد مثبت خیلی بزرگ بدم چی؟ اگر عدد منفی خیلی کوچیک بدم چی؟

اگر مثلاً سن رو بین ۱ تا ۱۰۰ قبول می‌کنه، من ۲۰۰ بدم چی؟ (ایده اینه خارج از محدوده مورد قبول یا مورد انتظار نویسنده یه برنامه یه چیزی وارد کنم.)

اگر میگه بین گزینه ۱ تا ۵ یکی رو انتخاب کن، من گزینه ۱- رو انتخاب کنم چی میشه؟

اگر اینجا آدرس وبسایت و URL رو می‌گیره، من به جای `https://example.com`، بیام بگم `https:/example.com` (یه دونه «/» به کار بردم به جای دوتا)، چه اتفاقی میوفته؟

اگر یه فایل ورودی میگیره، من فایل اشتباه و عجیب غریب با پسوند و فرمت عجیب و غریب بدم چی؟

درواقع همش ایده اینه که مخالف خط اصلی برنامه پیش برم. درواقع امنیت هم شما باید بدونین سیستم چجور کار می‌کنه و هم بدونین چجور پیش برم که مورد انتظار برنامه‌نویس نبوده تا به مشکل بر بخوره. چجور خرابش کنم؟ چطور برعکس چیزی که می‌خواد پیش برم. همش همین تفکره.

گاهی دیدین توی بازیا وقتی به ترتیب روی چند تا دکمه خاص کلیک کنین، مثلاً پولتون زیاد میشه یا مرحله رو رد می‌کنین؟

یا مثلاً میرید روی میز و به فلان قسمت خاص میز تیر می‌زنین و باعث میشه بتونین برید توی میز و قایم شین؟

دقیقاً همین ایده‌هاست.

حالا اگر بیایم این چیزای رندوم رو تست کنیم، بهش می‌گن تست‌های fuzzing.

اگر به زبون‌هایی مثل C که int و long int دارن آشنا هستین، مورد زیر رو بخونین:

اگر int ورودی میگیره و من به عددی خارج محدوده int بدم چی؟

```
int i;
scanf("%d", &i);
```

یا مثلاً یه جایی از کد اینطوریه:

```
int res, num1, num2;
num1 = input()
num2 = input()
res = num1 * num2;
```

خب مشکل کد چی می‌تونه باشه؟

ضرب num1 در num2 اگر خیلی بزرگ شه، توی محدوده int قرار نمی‌گیره و overflow می‌کنه. یا حتی ممکنه underflow کنه. یعنی منفی خیلی کوچیکی بشه که در رنج int جا نگیره.

راه حل:

+ هیچ‌وقت به داده‌ای که کلاینت میده اعتماد نکنید! نقاط حساس رو چک کنین.[۱۷]

همیشه قبل ذخیره داده‌های کاربر، داده رو چک کنین که معتبر باشه و اشتباه نباشه. مثلاً قرار بوده عدد باشه، کاربر اینطوری وارد کرده: 14212a1643. خب غلطه!

سعی کنین توی این موارد ارورها و داده‌های اشتباه رو هندل کنین و برنامتون کرش نکنه. (اکثر زبونا مکانیزم‌های ساده‌ای دارن که هندلشون کنین. مثلاً توی پایتون ساختاری به نام try و except هست.[۱۸] [۱۹]

+ fuzzing انجام بدین.

خب شاید بگین که برنامه من خیلی بزرگه. بازی من خیلی بزرگه. من هزار نفر رو هم استخدام کنم که بیان بازی منو تست کنن و خودشونو به دیوار بمالن، به فلان‌جا تیر بزنن، فلان جا حرکات رندوم انجام بدن که مشکلات رو پیدا کنن هم کمه. پس چیکار کنم.

موضوع اینه که بله هیچ‌وقت شما نمی‌تونین دستی این کار رو انجام بدین! بلکه باید تلاش کنین برنامه بنویسین که بیاد برای شما تست «fuzzing» انجام بده. یعنی بیاد مقادیر رندوم عجیب‌غریب، کرکترهای مثل «٪×،*)» بده تا ببینه کِی به مشکل بر می‌خورین.

درواقع یادتونه گفتم برنامه‌نویسی رو به چشم یه ابزار ببینین؟ درواقع به خاطر اینه که برنامه‌نویسی میاد کارهای خسته‌کننده رو به صورت خودکار و خیلی بهتر و سریع‌تر انجام میده. پس نیازه دانش برنامه‌نویسی داشته باشین.

پیشنهاد emoji تلگرم:

من یه روز داشتم توی secret chat تلگرم چت می‌کردم و دیدم برای چیزی که می‌نویسم، پیشنهاد ایموجی میده:

---

۱۷ حواستون باشه که برای حوزه وب، چک‌کردن client side (صرفاً سمت مرورگر کاربر)، کافی نیست! چک client side میشه به راحتی bypass اش کرد (حتماً باید سمت سرور هم چک شه):
CWE-602: Client-Side Enforcement of Server-Side Security →
https://cwe.mitre.org/data/definitions/602.html

۱۸ خیلی ساده بخوام بگم، مکانیزم try و except میگه که اول try رو امتحان کن، اگر به مشکلی یا چیزی خوردی که نشد try رو انجام بدی، برو به جاش قسمت except رو انجام بده.

۱۹ البته صرفاً try و except کافی نیست! بلکه باید حواسمون باشه که موارد دیگه‌ای مثل white list و black list کردن و فیلترکردن ورودی رو هم انجام بدیم. چون ممکنه یه چیزی لزوماً باعث نشه که ارور رخ بده که وارد except شه، ولی می‌تونه باعث مشکل امنیتی شه.)

خب من پیش خودم اول فکر کردم که چون به زبونای مختلف و کلمات مختلف پیشنهاد ایموجی میده، احتمالاً کلمات من رو می‌فرسته به سمت سرور و سرور تشخیص میده که چه ایموجی باید نمایش داده شه و اون لیست رو برای من می‌فرسته و برنامه تلگرم پیشنهادها رو نشون میده.

بعد گفتم خب اینکه نمیشه که! سیکرت‌چت تلگرم از چت عادی متفاوته. برای این ساخته شده که پیام فقط توسط من و طرف مقابلم قابل خوندن باشه. نباید پیام بره سمت سرور! پس این یه مشکل امنیتیه!

بعد برای اینکه مطمئن شم که آیا واقعاً پیام میره سمت سرور یا نه، اومدم اینترنتم رو قطع کردم و یه چیزی رو سرچ کردم که هیچ‌وقت سرچش نکرده بودم (که توی کَش یا چیزی نباشه). دیدم که بازم داره پیشنهاد میده.

اونجا فهمیدم که توی دستگاه من داره پردازش انجام میشه و به سمت سرور چیزی نمیره.

این یه نمونه از اون تفکر هکرمندی بود که بهتون گفتم. یعنی من اتفاقات رو درک کنم و ببینم چه مسائل و مشکلاتی مربوط به امنیت و حریم شخصی ممکنه توشون وجود داشته باشه.

Timing Attack

یادتونه علامت «==» توی پایتون (یا زبونای دیگه مثل جاوا) چیکار انجام میداد؟ میومد چک می‌کرد آیا دو تا چیزی که دادیم عیناً با هم برابرن؟

مثلاً:

```
'abcd' == 'bbbb'
```

درواقع پایتون میاد از اولین کرکتر چپ و راست شروع می‌کنه. می‌گه اولین کرکتر متن اول، آیا با اولین کرکتر متن دوم برابره یا نه؟ اگر برابر نبود، همون اول میگه خب پس لابد متنا یکسان نیستن! پس همون موقع میگه که دو متن برابر نیستن. (دیگه بقیش رو نگاهم نمی‌کنه. چون میگه برای چی نگاه کنم؟ وقتی یه کرکتر برابر نیست پس برابر نیستن دیگه! الکی چک کنم که سرعت برنامه کم شه؟!)

حالا اگر کرکتای اولی دو متن برابر بودن چی؟

```
'abcd' == 'aaaa'
```

میگه خب اولی برابر بود. میره کرکتر دوم رو چک می‌کنه ببینه آیا دومیا چی برابرن یا نه؟[۲۰]

---

۲۰ درواقع کدش اینه:

```
int isPassValid(char pass[], char inp_pass[])
{
    if (len(pass) != len(inp_pass))
        return 0;

    for (int i = 0; i < len(pass) && i < len(inp_pass); i++)
    {
        if (pass[i] != inp_pass[i])
            return 0;
```

حالا فرض کنین پسورد سایت **استرینگ** '۱۲۳۴۵۶۷۸۹' هست. خب من می‌خـوام پسـورد رو حـدس بزنم. چیکار می‌کنم؟

به ترتیب موارد زیر رو امتحان می‌کنم:

'000000000', '111111111', '222222222', ...

و همینطور ادامه میدم. خب شاید بگین برای چی؟

اگر کد رو نگاه کنین، می‌بینین که دونه‌دونه روی کرکترا حرکت می‌کنـه و اولین چـیزی mismatch و چیزی که یکسان نباشه رو ببینه، میگه یکسان نیستن..

حالا مشکل چیه؟ مشـکل اینـه کـه من مثلا می‌بینم ۰۰۰۰۰۰۰۰۰ مثلا ۱ میلی‌ثانیـه طـول کشـیده و بعدش می‌بینم که ۱۱۱۱۱۱۱۱۱، مثلاً ۲ میلی ثانیه طول کشیده و بقیه چیزا یعنی ۲۲۲۲۲۲۲۲۲ و... هم همگی ۱ میلی ثانیه طول کشیدن. پس من میگم عه چرا بیشتر طول کشید؟ بـذار فکـر کنم. هممم قاعـدتا کرکتر اول درست بوده و مجبور شده زمان بیشتری رو طرف کنـه کـه کرکـتر دوم رو هم چـک کنـه. خب درواقع کرکتر اول پسورد شما پیدا شد. خب با ۱۰ تا تست می‌تونم کرکـتر اول رو حـدس بـزنم. بـرای هـر کرکتر ۱۰ تا حدس نیازه. پس برای یه رمز ۱۰ کرکتری، ۱۰*۱۰ تا حدس نیازه. یعنی صرفا ۱۰۰ تا حدس. اما در حالت عادی که بخوایم همه حالات رو تست کنیم 10^10 حالت نیازه. یعنی به شدت کارمون ساده شد.

به این نـوع حملات میگن «Timing Attack». یعـنی از **تفاوت** زمـان مصـرفی، بتـونم اطلاعـاتی از سیستم بگیرم.

درواقع دقیق‌تر بگیم؛ وقتی صحبت از انجام کار و یا بررسی روی داده‌های حساس و پنهون میشه، تفاوت زمانیِ بین ورودی‌های مختلف، می‌تونه یه سری اطلاعات از اون داده حساسی که یه جای کد استفاده شـده (بیشتر برای بررسی و چک یا انجام یه سری کار بر حسب اون داده حساس)، پی برد.

- خب پس چیکار کنیم؟

+ یادتونه اسم این نوع حملات چی بود؟ حملات «Timing Attack» یا حملات مبتنی بر **تفاوت** زمان. خب پس برای رفع مشکل، ما نباید تفاوت زمانی داشته باشیم! یعنی مهم نباشه که کرکترای اول دوتاشـون یکسانن یا نه! فارغ از اون، کرکترا رو تا ته چـک کنـه و نگـه اولی یکسـان نبـود پس قطـع می‌کنم و می‌گم یکسان نیست. بلکه بره تا تهش و وقتی تا ته رفت و توی تعداد چک‌ها تفاوتی رخ نداد (هردو برنامه به خاطر اجراشدن تعداد دستور یکسان، یک زمان صرف کردن)، بعدش بگه یکسان نبودن.

درواقع شما به عنوان یه مهندس امنیت کدها رو مطالعه می‌کنی و میگی سازمان محـترمِ، اینجـا کـدت مشکل داره. بیا اینطور بنویس که امن باشه:

```
bool isPassValid(char pass[], char inp_pass[],
                 size_t lenPass, size_t lenInp)
{
    volatile bool isValid = true;
```

```
    }
    return 1;
}
```

```
    isValid &= (lenPass == lenInp);
    for (size_t i = 0; i < lenPass && i < lenInp; i++)
    {
        isValid &= (pass[i] == inp_pass[i]);
    }
    return isValid;
}
```

اینجا دیگه ما در هر صورت، چه طول یکسان باشه چه نباشه، چه اولین کرکتر mismatch بخوره چه آخرین، یه زمان طی میشه.[۲۱]

هر بار `isValid` رو `and` می‌کنم با چیزی که می‌خوام. پس اگر mismatch باشه، `isValid` مقدارش `false` میشه ولی بازم روند ادامه پیدا می‌کنه و روند قطع نمیشه. (که زمان یکسانی صرف شه)

- اگر به صورت رندوم تاخیر اضافه کنیم که متوجه نشن دقیق چه زمانی طی شده، مشکل حل میشه؟
+ سخت می‌کنه ولی ناممکن نمی‌کنه![۲۲]

درواقع فرض کنین شما تاخیرهای کاملاً رندوم اضافه کنین. قاعدتاً باید یه بازه‌ای انتخاب کنین. مثلاً بازه (۱۰۰ و ۲۰۰) نانوثانیه. خب قاعدتاً هر دفعه به صورت رندوم بین ۱۰۰ تا ۲۰۰ نانوثانیه یه تأخیر اضافه میشه.

این مشکل رو حل نمی‌کنه صرفاً مشکل‌تر می‌کنه. چون حمله‌کننده باید تعداد بار بیشتری رو تست کنه. مثلاً ۱۰۰۰ بار یه چیزی ثابت رو تست کنه و بعد نمودار بکشه و بفهمه که به طور میانگین ۱۵۰ نانوثانیه تأخیر اضافه شده.[۲۳] حالا با نمودار آنالیز رو پیش میبره. هردفعه این عملیات رو با کرکترای مختلف تست می‌کنه و در نهایت هی دونه‌دونه کرکترا رو پیدا می‌کنه.

سایت زیر توضیح خیلی خوبی داده و راه‌حل‌هایی هم پیشنهاد داده. حتماً بخونیدش:
https://www.chosenplaintext.ca/articles/beginners-guide-constant-time-cryptography.html

یکی از دلیلای استفاده از اسمبلی برای رمزنگاری اینه که ما می‌دونیم instruction ها هرکدوم آیا constant time هستن یا نه؟ چقدر clock cycle طول می‌کشه تا اون instruction اجرا شه؟
درواقع با اسمبلی، ما دسترسی خوبی روی سخت‌افزار داریم.

نمونه‌های دیگه‌ای از Timing Attack (پیشرفته):

---

۲۱ البته این صد درصد constant time نیست!

22 i) https://stackoverflow.com/questions/28395665/could-a-random-sleep-prevent-timing-attacks
ii) Remote Timing Attacks are Practical: https://crypto.stanford.edu/~dabo/papers/ssl-timing.pdf
iii) https://fahrplan.events.ccc.de/congress/2012/Fahrplan/attachments/2235_29c3-schinzel.pdf

23 Law of Large Numbers: https://en.wikipedia.org/wiki/Law_of_large_numbers

+ Timing Attacks on WhatsApp, Signal, and Threema can Reveal User Location[24]
→ Paper: Hope of Delivery: Extracting User Locations From Mobile Instant Messengers[25]

+ Security Best Practices for Side Channel Resistance (*Highly recommended!*)[26]
چندتا نکته اضافه کنم:

+ مثلاً نسخه ۲۳.۰۴ ابونتو، درسته ورژن بالاتری از 22.04LTS هست، اما مدت ساپورت امنیتیش کمتره! پس دقت کنین که از ورژنی استفاده می‌کنین که ساپورت امنیتی داشته باشه و جدیدتر به معنای ساپورت بیشتر ممکنه نباشه!

+ حواستون باشه که اروری که میدین، یه وقت به دلیل اورهای سیستمی، حاوی اطلاعات حساس نباشه! مثلاً پوینتر دست بخوره و تغییر کنه یا پوینتر به داده قبلی که اشاره‌گر به یه پسورده ریترن نشه! اینطوری ممکنه باعث شه اطلاعات حساس لو بره!

```
Return obj;
```

که obj یه object هست که ممکنه دستخوش تغییر شده باشه. ممکنه اسم ادمین باشه یا اسم یوزر! مطمئناً اسم ادمین نباید ریترن شه! (در شرایط خاص)

یا حتی ممکنه به خاطر کرش شدن، بخشی از مموری چاپ شه (که ممکنه حاوی اطلاعات حساس باشه)

+ همونطور که اینتل گفته، فکر نکنین که این نوع خاص side-channel attack امکان رخدادنش کمه و خب بگین حوصله حلشو ندارم پس ولش! نه! بلکه ممکنه الآن امکان رخدادنش کم باشه ولی بعداً محققانی بیان ثابت کنن که خیلی ساده میشه ازش سوءاستفاده کرد. (مراجعه شود به قسمت «Choosing Secure Open Source Package») پس حلش کنین. نگین مهم نیست!

+ حواستون باشه یه تغییری توی سیستم ممکنه side-channel ای که قبلاً رخ نمیداد رو عوض کنه و رخ بده. پس حواستون باشه!

+ Guidelines for Mitigating Timing Side Channels Against Cryptographic Implementations[27] (*Highly recommended*)

یه سری نکته اضافه کنم:

+ دسترسی به double word های زوج توی مموری سریع‌تر از فرده. درواقع اگر اعضای آرایه شما روی خونه‌های زوج باشه، دسترسی بهشون سریع‌تر از دسترسی به آرایه‌ای هست که اعضا روی خونه‌های فرد قرار دارن. پس همینم می‌تونه تفاوت زمانی ایجاد کنه. درواقع اینو گفتم که یه ایده‌ای بدم که وقتی میگیم دید system-level ای باید داشته باشیم یعنی چی؟ یعنی یه همچین مواردی وجود دارن که نیازه

24 https://restoreprivacy.com/timing-attacks-on-whatsapp-signal-threema-reveal-user-location/
25 https://arxiv.org/abs/2210.10523
26 https://www.intel.com/content/www/us/en/developer/articles/technical/software-security-guidance/secure-coding/security-best-practices-side-channel-resistance.html
27 https://www.intel.com/content/www/us/en/developer/articles/technical/software-security-guidance/secure-coding/mitigate-timing-side-channel-crypto-implementation.html

بدونیمشون و گاهی نیازه توی طراحی حواسمون بهش باشه. درواقع شما بدون درک سیستم لولی، نمی‌تونین رمزنگاریی طراحی کنین که در برابر حمله‌ها مقاوم باشه.

یا حتی سرعت دسترسی به یه داده ۴ بایتی با یه داده ۸ بایتی ممکنه متفاوت باشه.

+ Checking that functions are constant time with Valgrind[28]

image-editing tools

فرض کنین شما یه عکس رو crop (برش) می‌زنین و فکر می‌کنین که خب دیگه قسمت حساس رو جدا کردم و همه چی اوکیه. ولی نیست!

ببینین همه چیز توی کامپیوتر یه سری متنه. درواقع پشت هر عکس هم یه سری متن هست. کامپیوتر اون متن رو می‌خونه و میفهمه که باید فلان جور نمایشش بده. (یه عکس یا ویدیو رو به وسیله notepad باز کنین تا متنای عجیب و غریب رو ببینین.)

حالا وقتی شما عکس رو کراپ می‌کنین، ممکنه صرفاً بیاد یه flag پایان فایل بذاره که تا فلان جا نشون بده و بقیش نشون نده. یعنی وسطش بگه که اینجا فایل پایان یافته. یه همچین چیزی:

Original image: `sdklfjsadfjfdslkweruoiEND`

cropped image: `sdklfjsadfjENDfdslkweruoi`

درواقع بقیه اطلاعات عکس پاک نشدن، صرفاً از دید برنامه‌های عادی که متن رو تا پایان فلگ می‌خونن، مخفی شده. چون صرفاً یه فلگ مشخص کردیم که تا فلان‌جا نشونش بده.

اما با برداشتن اون، میشه اطلاعات رو ریکاوری کرد و به عکس اصلی دسترسی پیدا کرد![۲۹]

- چجور میشه فهمید که آیا برنامه‌ای این مشکل امنیتی رو داره؟

+ مثلاً نگاه به حجم فایل می‌کنیم و می‌بینیم که حجم فایل تغییر خاصی نکرده. خب اگر قرار بود قسمتای برش‌خورده پاک شن، درواقع متن پشت عکس پاک می‌شد و خب حجم هم باید کم میشد.

گاهی هم برای اینکه برنامه بتونه تغییرات رو undo کنه (برگردونه به عقب)، اطلاعاتی رو توی متادیتا نگه میداره. بازم میشه بازیابیش کرد.

یا مثلاً محوکردن (blur) عکسا، یه الگوریتم داره که فلان بیتا رو فلان کار کنه. خب اگر الگوریتم و ریاضیاتش رو برعکس کنیم، میشه عکس رو بازیابی کرد.

یه مثال دیگه!

فرض کنین شما یه عکس روی یه برگه دارین. حالا مثلاً روی خاص که می‌خواین چهرتون معلوم نشه، یه برگه سیاه می‌گیرین روی چهرتون. خب آیا کلاً چهرتون از بین رفت؟ نه! صرفاً یه لایه گذاشتین روش که زیرش معلوم نشه.

این دقیقاً اتفاقی هست که توی بعضی از نرم‌افزارهای ادیت عکس میوفته. یعنی یه نفر بیاد لایه رو برداره، تصویر شما نمایان میشه!

---

28 https://github.com/agl/ctgrind

۲۹ حالا نه اینقدرم ساده! من موضوع رو خیلی ساده‌سازی کردم که درکش راحت شه.

یا مثلاً برای اینکه تشخیص بدن چهره‌ای که محوشده، چه کسی بوده، میان عکسایی که حـدس می‌زنن شاید اون چهره باشه رو با همون نرم‌افزار بلور می‌کنن که ببینن کی عینـاً مشـابه عکس بلـوری میشـه کـه هویت یا حتی متن پشت عکس رو پیدا کنن.[۳۰]

راه حل:

عکسو که ادیت کردین، از روش screenshot بگیرین و اونو به اشتراک بگذارین کـه یـه عکس جدیـد بسازین که داده‌های اضافی عکس قبلی توش نباشن. (البته این لزوماً قرار نیست صد درصدی باشه!)
با یه گوشی دیگه از روی مانیتور و عکس ادیت‌شده، عکس بگیرین.

“be careful and if you want to hide something just paint over it with a solid color (and check it's also hidden in thumbnails, preview images, undo histories and such).”[31]

More reading:

+ aCropalypse[32]
+ Can you recover original data from a screenshot that has been 'blacked out'?[33]
+ How secure is 'blacking out' sensitive information using MS Paint?[34]
+ How to Recover Numbers from Blurred Images[35]
+ Why You Should Never Use Pixelation To Hide Sensitive Text[36]

---

۳۰ یه حالتی brute-force.

31 https://photo.stackexchange.com/a/35115, CC BY-SA 3.0
نمی‌دونین مثلاً thumbnail چیه؟ خب سرچ کنین!

32 https://en.wikipedia.org/wiki/ACropalypse
33 https://security.stackexchange.com/a/67298
34 https://security.stackexchange.com/questions/126932/how-secure-is-blacking-out-sensitive-information-using-ms-paint
35 https://www.schneier.com/blog/archives/2007/01/how_to_recover.html
36 https://dheera.net/posts/20140725-why-you-should-never-use-pixelation/
عملاً توصیه‌ای که کرده که به جای بلورکردن بیاین روش خط بکشین، حواستون باشه که اولاً **کاملا** سیاه سیاه باشه (نه خاکستری پررنگ) و همچنین توی نرم‌افزارهایی باشه که لایه اضافه نکنن. بلکه دیتا رو کامل پاک کنن و رنگ سیاه رو **جایگزین** کنن.

DNS (Domain Name Server)

- درخواست DNS چیه؟

+ اینترنت یه فضاییه که همه با هم در ارتباطن؛ همچنین می‌دونیم که من اگر بخوام به کسی توی دنیای فیزیکی پیام بدم، باید برم اداره پست، نامه بنویسم و آدرس مبدأ و مقصد رو مشخص کنم. اداره پست میبره تحویل میده.

همه چیز توی اینترنت یه سری آدرس داره. بهش میگن IP Address. و همه چیز عین نامه انتقال پیدا می‌کنه.

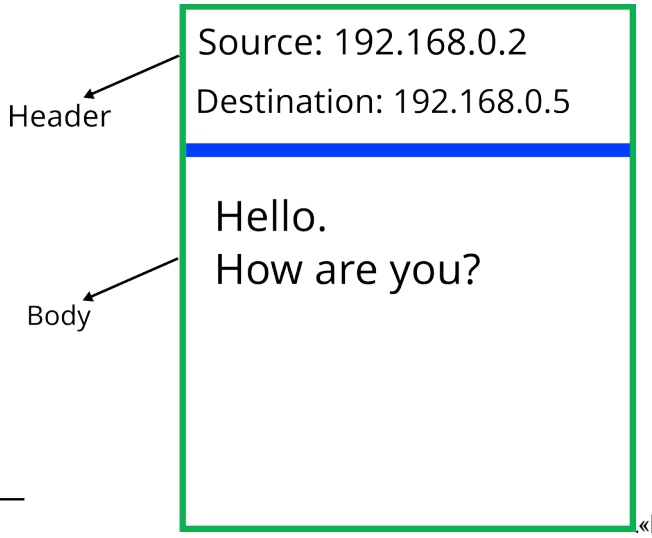
توی دنیای اینترنت، هرکسی یه آدرسی داره. من اگر بخوام به گوگل وصل شم، آدرس گوگل رو پیدا می‌کنم و به اون آدرس پیام می‌فرستم! این آدرس هم اسمش IP هست. آدرس آیپی یا همون Internet Protocol Address.

برای همین هم میگن آیپیتون رو کسی بهتره نفهمه. چون انگار آدرس خونتون رو فهمیده![37] [38]

آدرس‌ها توی اینترنت یه چیزی مثل اینن:

`61.156.22.11`

درواقع شکلش اینطوریه:

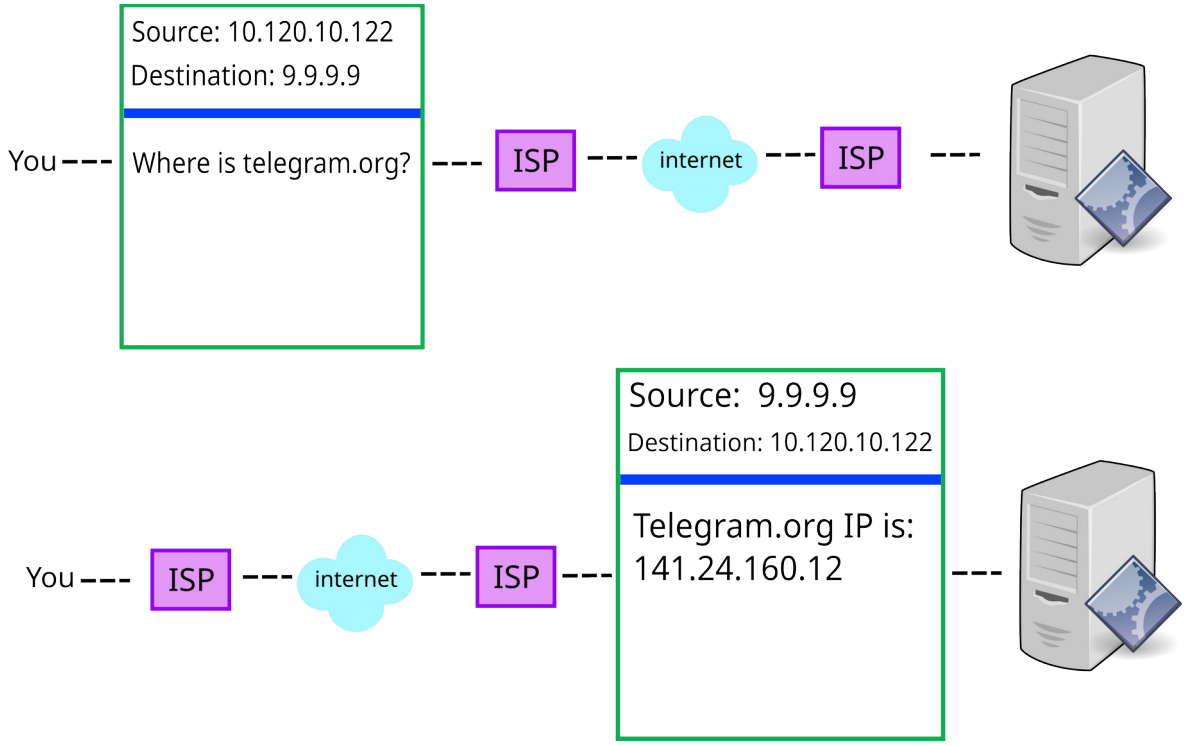


۳۷ با استفاده از «IP lookup»

۳۸ البته آیپی دقیقاً مساوی لوکیشن فیزیکی دقیق نیست!

بالاش می‌نویسیم گیرنده و فرستنده و پایینش متنایی که می‌خوایم بفرستیم رو می‌نویسیم. پشت عکس هم یه متنه درواقع! برای همینم میشه به وسیله پکت و اینترنت فرستاد. (یه عکسو بـه صـورت متـنی و بـا نوت‌پد باز کنین و متن پشتشو ببینین!)

اما یه چیزی! حفظ کردن آدرس همه خیلی سخته نه؟! خیلی سخته این همـه عـدد حفـظ کـنیم. خب اینجا به ما میگن شما نمی‌خواد عدد حفظ کنی! بـه جـاش تـو اسـم انگلیسـی حفـظ کن. تـو بنـویس مثلاً signal.org، من میرم خودم از یه سری جای خـاص می‌پرسـم کـه آدرس پشـت این signal.org چیـه؟ یعنی به جاش یه سری اسم بهش اختصاص میدیم و بعداً خودمون میریم پیداش می‌کنیم.
درواقع یه سری سرور (یه سری کامپیوتر خاص) یه جای خاص هستن (مثل ۹.۹.۹.۹، ۱.۱.۱.۱، ۸.۸.۸.۸ و...) که مرورگر آدرسشون رو بلده و میره ازشون می‌پرسه که فلان جا می‌خوام برم آدرسـش چیـه؟ یعـنی صرفاً مرورگر یه آدرس حفظ می‌کنه. هرجا بخواد بره میره از اون آدرسه می‌پرسه:



من از جای مخصوص که ۹.۹.۹.۹ هست پرسیدم که می‌خوام برم وبسایت تلگرام. آدرسش کجاست؟ اونم بهم جواب پس داد که آدرسش فلانه. بعد من از این به بعد عین همین میرم به سمت تلگرام.
– اون چجور پیدا میکنه؟
+ مثل دفترچه تلفن که این آدرسای حرفی رو به آدرس عددی وصل میکنه. یه چیزی مثل این[۳۹]:

Domain Name	IP
archive.org	198.50.177.216

۳۹ این سادشه. مطمئناً ما هدفمون بررسی دقیق DNS lookup نیست! صرفاً می‌خوایم با مفهوم کلیش آشنا شیم.

eff.org	173.239.79.196
mozilla.org	44.236.48.31

اینجا بهم جواب پس میده که آقا توی می‌خوای بری فلان‌جا، آدرسش ۴۴.۲۳۶.۴۸.۳۱ هست.
کل اینترنت عملاً همینطوریه! هی چیزا به وسیله نامه و عملاً زبون انگلیسی ارسال میشن! صرفاً نیازه آیپی مقصد رو داشته باشم تا بهش پیام بدم.

اما یه مشکل! این درخواست DNS به صورت Plain-text یا clear-text ارسال میشه! (یعنی به صورت کاملاً عادی و حروف انگلیسی! یعنی بدون هیچ رمزی!) یعنی هرکی تو مسیره، می‌تونه ببینه من میخواستم وارد چه سایتی شم و یا با چه سروری ارتباط بگیرم (حواستون هست که همیشه صرفاً بحث وارد سایت شدن نیست! بلکه ممکنه درون یه برنامه، نیاز باشه با یه سرور ارتباط بگیریم.)
خب این به حریم خصوصی افراد ضربه میزنه. چون ISP و دیگر افراد میتونن ببینن من میخواستم وارد چه سایتی بشم و حتی اون درخواست رو تغییر بدن. (یکی از راه‌هایی که برای فیلترینگ به کار میره همینه که درخواست DNS رو درستکاری می‌کنن و آیپی اشتباهی بهمون می‌رسه! درواقع آیپی صفحه فیلترینگ یعنی ۱۰.۱۰.۳۴.۳۵ به ما بر می‌گرده و ما الکی هدایت میشیم به صفحه فیلترینگ!)[۴۰]
- خب برای این چیکار کنیم؟!
+ عملاً قسمت body میاد رمز میشه که نفهمن چی داریم میگیم. یعنی وسط راه، صرفاً یه پیام رمزی دیده میشه.

درواقع یه هکر می‌دونه DNS به صورت متن عادی انگلیسی هست و میشه خوند و تغییرش داد. پس می‌تونه ابزار بنویسه که مثلاً وسط راه درخواستای DNS رو عوض کنه. درواقع یه دانشی از پروتکل‌های[۴۱] مختلف داره و حالا می‌دونه چه مشکلات امنیتی ممکنه به وجود بیاد. چجور ازش استفاده کنم که ترافیک رو بدزدم؟!

یا برای چی می‌گن از HTTP استفاده نکنین و از HTTPS استفاده کنین؟
چون HTTP متن عادیه و رمز نشده و وسط راه یکی می‌تونه بدزدتش و تغییرش بده و عملاً شما چیزی روی صفحه مرورگر می‌بینی که وجود نداره!

---

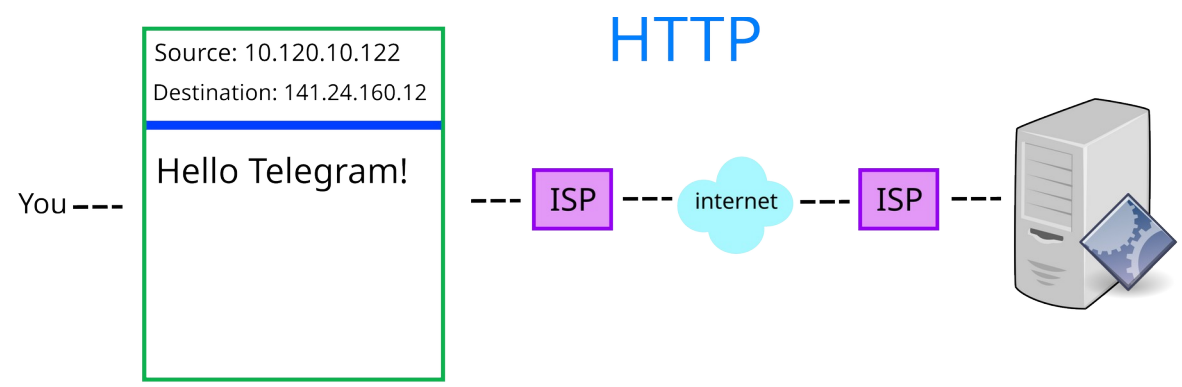
۴۰ این تنها راه فیلترینگ نیست. آیپی مقصد ← دراپ
۴۱ - عه پروتکل چیه؟
+ فرض کنین شما وارد ایران میشین، معمولاً گفت‌وگوها با سلام و دست‌دادن شروع میشه.
توی ژاپن مثلاً با سر تکون دادن صحبت رو شروع می‌کنن.
درواقع پروتکل (قرارداد) گفت‌وگو در ایران سلام و دست‌دادنه. یعنی یه قرارداد برای انجام کار.
کامپیوترا باید بینشون یه قراردادی تعریف شه که بتونن با هم ارتباط برقرار کنن. مثلاً قرارداد ارتباط VPN. اینکه چه‌جور اتصال VPN شروع شه و ادامه پیدا کنه، همش با یه سری قراداد از پیش تعریف شدس.
مثلاً قسمت امن‌شدن HTTP و تبدیلش به HTTPS اینطوریه که شما اول به وبسایت پیام میدی میگی سلام. من امن‌کننده‌های ورژن ۱.۲ و ۱.۳ با فلان رمزنگاری‌ها رو ساپورت می‌کنم.
بعدش سرور سلام می‌کنه و می‌گه اوکی از اونایی که گفتی من فلان چیز رو انتخاب می‌کنم. بیا از این به بعد سر این قراردادی که کردیم، با هم صحبت کنیم!
به اینا میگن پروتکل. یعنی یه قراردادی برای نحوه ارتباط. یعنی اینکه من چی بفرستم و در قبال چیزای مختلفی که می‌گیرم و دریافت می‌کنم، چی جواب بدم؟

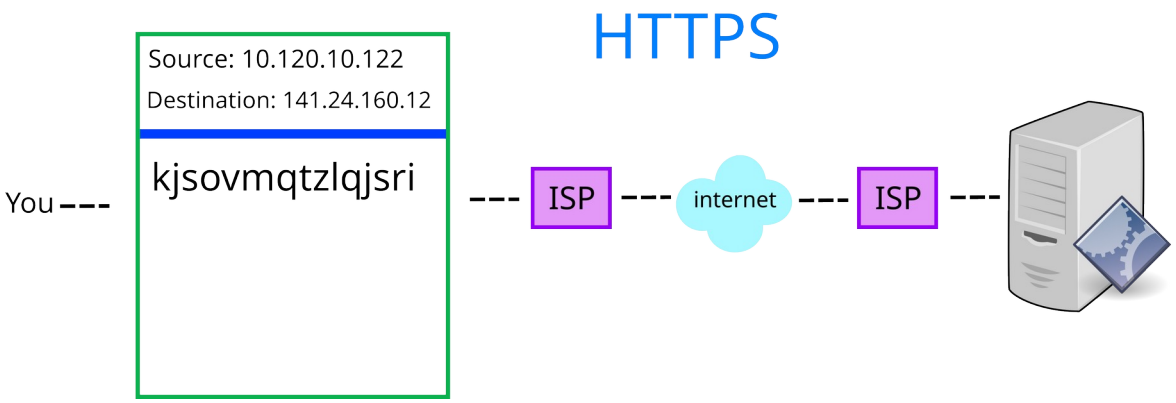
حالا مثلاً متخصص امنیت میره تست انجام میده که یه وقت داده حساسی به صورت رمزنشده عبور نکنه. یا یه وقت مشکلی توی رمزشدن داده‌ها پیش نیاد. داده‌ها به صورت امنی ذخیره و ارسال بشن. قبل ارسال درست رمز شن. نشه مشکلاتی ایجاد کرد که سیستم کرش کنه و عملکرد رمزکردن مختل شه.
همه اینا کارهای یه متخصص امنیته.

ببینین در حالت کلی وقتی وارد یه سایتی شین، دولت و ISP (شرکت ارائه دهنده اینترنت) و کسایی که توی شبکه شما هستن (مثل شبکه Wi-Fi)، چیزای زیر رو به صورت پیشفرض متوجه میشن: (البته چیزای دیگه هم با روش‌های مختلف متوجه میشن ولی اینا بدیهی‌ترینشونه!)

+ در صورت HTTP بودن اون وبسایت، دولت و ISP ما، چون بین حرکت پکت‌ها هستن، عیناً چیزایی که می‌فرستیم رو می‌بینن. به معنای واقعی کلمه، همه چیز! انگار کنارتون نشسته و داره میبینه.
حتی میتونه اطلاعاتی که شما دارین روی صفحه میبینین رو هم تغییر بده و عوض کنه! (که این اطلاعات ممکنه حاوی اطلاعات حساسی مثل رمز و آدرس خونتون و... باشه

+ در صورت HTTPS سایت:
قسمت Body رمز میشه. یعنی عملاً تبدیل به یه چیز غیرقابل خوندن میشه. (همونطور که می‌بینین، تبدیل به یه چیز عجیب‌غریب شده)
پس جاهای مثل ISP و دولت، آدرس وبسایتی که می‌خواین برین و مقدار مصرف اینترنتی که انجام دادین (حجم چیزایی که رد و بدل کردین)

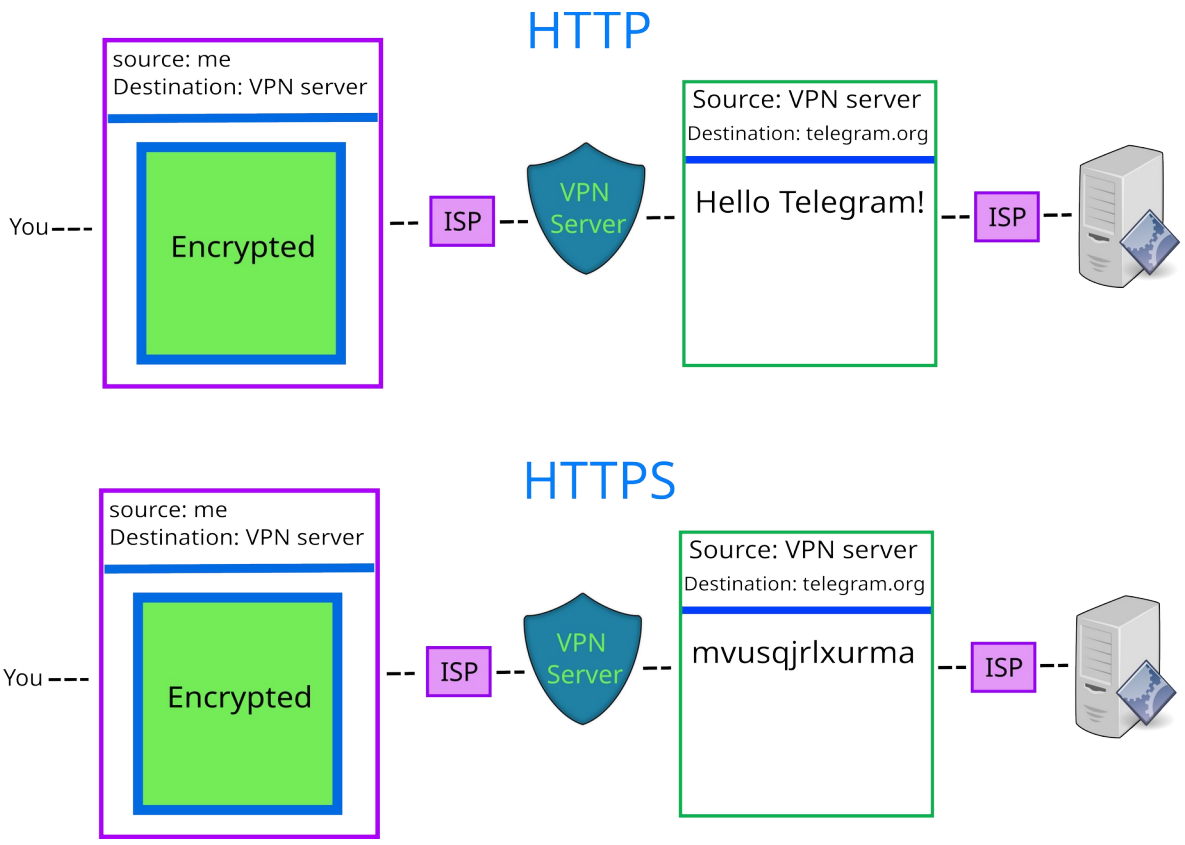




VPN ها چجور کار می‌کنن؟

+ ببینین VPN مثل یک تونل عمل میکنه. پیام‌های شما توی دستگاه شما رمز میشه. بعدش میره سمت VPN سرور (یعنی فرستنده شما و گیرنده VPN server) اونجا رمز باز میشه و پیام ادامه راهش رو میره.

درواقع این اتفاق میوفته:



پکت شما عیناً یه لایه مستطیل رمز میاد روش. گیرنده و فرستنده عوض میشه. برای همین هم عملاً فیلترینگ و تحریم رو دور میزنه. چون گیرنده شما میشه VPN سرور. نگاه می‌کنن نمیفهمن می‌خوای بری تلگرام. میبینن میخوای بری یه سرور عادی. بعدش از اونجا به بعد از سرور VPN میره به سمت وبسایت. وبسایت اگر ما رو تحریم کرده باشه، نمی‌دونه پیام از ایران اومده. می‌بینه از یه سرور عادی از مثلاً هلند اومده. برای همین اجازه میده.

- حالا VPN بخوایم وصل نشه چیکار می‌کنیم؟

+ آدرس آیپی سروری VPN رو در میاریم و فیلتر می‌کنیم و هر وقت دیدیم یه پکتی می‌خواد به اون گیرنده بره، اجازه عبور نمیدیم.

VPN Kill-switch

خب حالا فرض کنین بنا بر یه مشکلی، VPN تون قطع میشه. خب قاعدتاً داده‌هاتون عادی و خارج از تونل امن VPN منتقل میشه و IP تون لو میره.

اینجا یه قابلیتی گذاشتن به نام kill-switch. که درواقع برنامه میگه اگر من دیدم اتصال VPN قطع شد، همون لحظه میام اینترنت رو کلاً قطع می‌کنم که داده‌ای خارج از تونل VPN منتقل نشه. تا وقتی هم که VPN متصل نشه، امکان انتقال داده رو نمیدم.

اما فرض کنین من دستگاه رو خاموش می‌کنم. اگر برنامه درست نوشته نشده باشه، ممکنه اول پراسس (قابلیت) مربوط به kill-switch خاموش بشه و بعد سیستم خاموش بشه. یعنی در حد فاصله بین بسته‌شدن VPN و حذف پراسسش تا خاموش‌شدن سیستم، داده‌های خارج از تونل VPN و با اینترنت روشن انتقال پیدا کنن.

یا حتی برعکسش. زمانی که شما گزینه «auto-connect» و «launch app on start-up» رو روشن کردی که تا سیستم روشن شد، VPN هم روشن شه و داده‌ای خارج VPN عبور نکنه. اما حالا بنا بر یه سری چیزا، ممکنه برای مدت خیلی کمی تا VPN شروع به وصل‌شدن کنه، داده‌های خارج از VPN عبور کنه و IP واقعی ما رو فاش کنه.[۴۲]

درواقع توی امنیت ما دنبال نقاط حساس و لبه (edge) می‌گردیم. میگیم اگر یه وقت این حالت حساس و خاص و توی لبه پیش اومد چی؟

Heartbleed

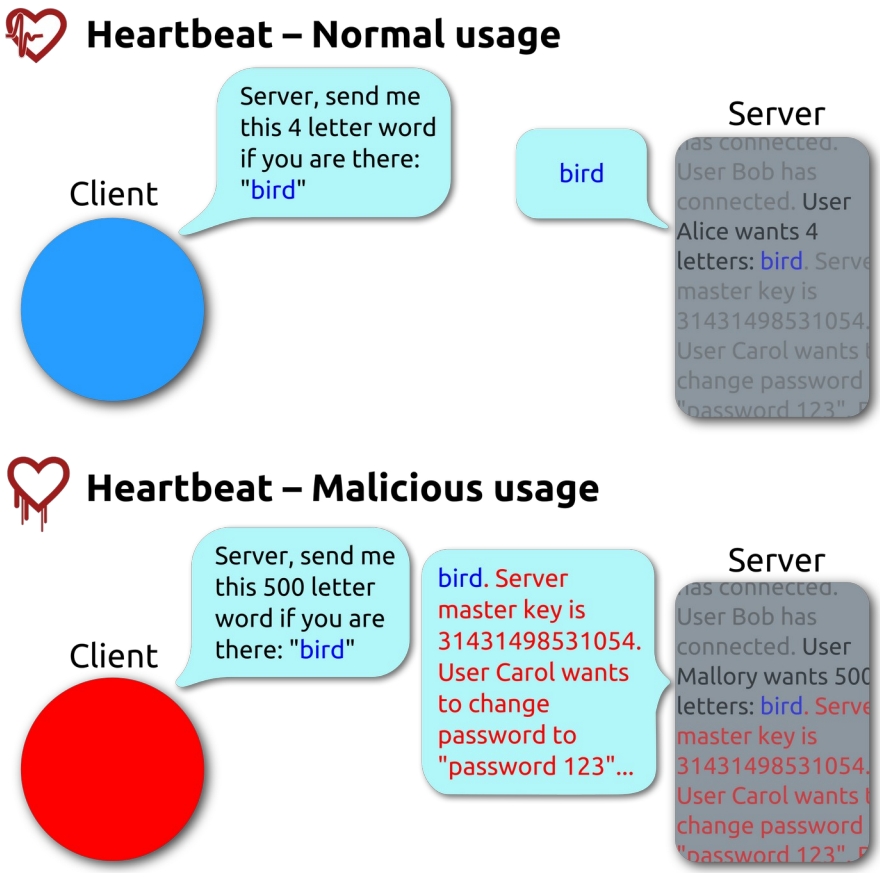


image1: FenixFeather, Simplified Heartbleed explanation, CC BY-SA 3.0

42 MUL22-02 (Medium): Leak of Traffic During System Shutdown: https://mullvad.net/en/blog/2022/10/21/security-audit-report-for-our-app-available/

image2[43]

خیلی خلاصه فارسیش (نادقیق): ایجاد و بازکردن HTTPS زمان‌بر و هزینه‌بره. برای همین هر از چندگاهی یه پیام ارسال میشه که آیا طرف مقابل هست و یا نیست؟ اگر هست، هنوز کانکشن رو نگه دارم که نخوام دوباره بازش کنم که زمان‌بر باشه. من برای اینکه بگم سرور من هنوز هستم و کانکشن رو نبند، به سرور میگم، سرور من هستم، اگر تو هم هستی، برام کلمه مثلاً bird رو بفرست. همچنین من توی هدر بهش میگم bird تعداد حروفش ۴ تاست. سرور میاد ۴ حرف از مموریِ اون قسمت رو میفرسته. اما چک نمیکنه که آیا من درست گفتم یا نه!

ممکنه من بگم سرور اگر هستی، bird رو برام بفرست. تعداد حروف = ۵۰۰ تا. اینطوری ۴ حرف bird رو میفرسته و در ادامه، ۴۹۶ تا حرف دیگه از مموریش میفرسته! توی اون مموری ممکنه اطلاعات حساس و کلیدهای رمز و... باشه!

این باگ اینقدر بزرگ و بد بود (و همچنین ابعادش کاملِ کامل مشخصی نبود) که پروژه تور همچین چیزی رو گفته بود:

“Note that this bug affects way more programs than just Tor — expect everybody who runs an https webserver to be scrambling today. If you need strong anonymity or privacy on the Internet, you might want to stay away from the Internet **entirely** for the next few days while things settle.”[44]

Race condition

من برای بار اول Race Condition رو از یاشار شاهین‌زاده[۴۵] یاد گرفتم.

ببینین فرض کنین من کد تخفیف دارم و میخوام اونو وارد کنم؛ وقتی اونو وارد کنم، توی کامپیوتر این اتفاق میوفته:

۱- اول کامپیوتر چک میکنه آیا کد تخفیف معتبر هست یا نه؟

۲- بعدش میاد کد تخفیف رو اعمال می‌کنه و قیمت کالا کم میشه.

۳- بعدش که کد تخفیف اعمال شد، میاد کد تخفیف رو باطل می‌کنه که کسی دیگه نتونه استفاده کنه.

```
Get the code from user
Check validity
Apply code
Validity = False
```

یه چیزی شبیه زیر:

```
if coupon_validation == True:
```

43 See at: https://xkcd.com/1354/

44 https://blog.torproject.org/openssl-bug-cve-2014-0160/ , Archived: https://web.archive.org/web/20170710101031/https://blog.torproject.org/blog/openssl-bug-cve-2014-0160

۴۵ راه ارتباطی باهاش رو بعداً براتون می‌ذارم. حتماً دنبالش کنین.

```
    enable_discount() # (I)
    coupon_validation = False (II)
```

اما یکم فکر کنین که این چه مشکلی ممکنه به وجود بیاره.

راهنمایی: به این فکر کنین که تابع enable_discount بالاخره یکم طول میکشه.

جواب: ببینین هر کامپیوتری، هرچند سریع باشه، انجام مراحل یه زمان خیلی کمی براش زمـان میـبره. برای اینکه یکم واضح شه، یه عدد بزرگ مثال می‌زنیم. مثلاً فرض کنین که بین زمان شـروع مرحلـه (I) و باطل کردن کد که مرحله (II) باشه، مثلاً ۵ ثانیه زمان ببره.

فرض کنین من کد رو وارد کردم. چک کـرد کـد معتـبره و رفت خـط دوم. (پس از validation عبـور کردم).

حالا اگر توی این زمان ۵ ثانیه که برسه به خط سوم که کد رو باطل کنه، یه نفـر دیگـه هم درخواسـت اعمال کد تخفیف، بده. خب میاد توی خط اول، میبینه عه! کد معتـبره! پس از validation عبـور میکنـه! پس اون هم تونست از validation عبور کنه! چرا؟ چون هنـوز کـد باطـل نشـده! پس اون هم میتونـه بـه خاطر اینکه کد دیر باطل شده، از کد استفاده کنه! پس با این تکنیک میشـه چنـد بـار از یـه کـد تخفیـف استفاده کرد. به این میگن Race Condition! این آسیب‌پذیری توی web application هـا خیلی رخ میده.

- به نظرتون ممکنه دیگه کجاها رخ بده؟

+ هرجایی که نیازه یه validation ای چک بشه و شرط validation توی body اش تغییر پیدا کنه!

بیایم یه مثال دیگه بزنیم:

```
# race condition
my_balance = 100 # Get the balance from the bank API (I)
transfer_amount = 100
his_balance = 0
if my_balance >= transfer_amount:
    his_balance = his_balance + transfer_amount
    my_balance = my_balance - transfer_amount (II)
```

یعنی مثلاً من اگر دو درخواست انتقال پول با هم و دقیقاً همزمان بدم، هـر دو میـان تـوی if و می‌بینن که عه! شرط برقراره! هردو باز میرن پایین. بعدش توی خط آخر هم چون دو بار کم‌شدن پول اتفاق میوفته، موجودیم منفی میشه!

این کد یه آسیب‌پذیری دیگه هم داره! به نظرتون چیه؟!

راهنمایی: به transfer_amount فکر کنین!

خب کد چک نمیکنه که من آیا دارم عدد درست وارد می‌کنم یا نـه! ممکنـه عـدد منفی وارد کنم! اگـر منفی وارد کنم چی؟ فرض کنین موجودیم ۱۰۰ دلاره و من مقـدار انتقـال رو وارد می‌کنم ۵۰- دلار. حـالا پول میخواد بره به حساب طرف. اما منفی ۵۰ دلار میره! یعنی از پول طرف کم میشه!!!

```
his_balance = his_balance + transfer_amount
his_balance = his_balance + (-50)
```

حالا باید پول از حساب من منها بشه. اما خب چون پول منفیه، عملاً میشه منهایِ منفی ۵۰! یعنی به من پول اضافه میشه!

```
my_balance = my_balance - (-transfer_amount)
my_balance = 100 - (-50)
```

یعنی من می‌تونم با این تکنیک، به خودم هی پول اضافه کنم و از بقیه کم کنم!

برداشت مهم:

دیدین؟ تصور هالیوودی از هکر رو کنار بگذارین! هکر کسی هست که دانش خوبی از یه موضوع داره و میتونه روش‌های خلاقانه و غیرمعمول رو به کار بگیره که به هدفش برسه! مثلاً پیش خودش بگه، اگر اون کار کنم چی؟ چه اتفاقی میوفته؟ اگر از راه اصلی نَرَم و فلان کار کنم چی میشه؟ مثل اینجا که من پیش خودم گفتم که به طور معمول همه عدد مثبت وارد میکنن. اما اگر من منفی وارد کنم چی؟ اگر اعشاری وارد کنم چی؟ اگر فلان نوع کار غیرمعمول کنم چی؟ درواقع این، تفکر هکریی هست که باید داشته باشین. نیاز به هزار خط کدنویسی هم نیست! شما باید دانش عمیق از یه موضوع داشته باشین و بتونین از اون دانش عمیق استفاده کنین و بگین که کجاها یه سری چیزها رعایت نشده، کجاها یه سری چک‌ها صورت نگرفته و من میتونم ازش استفاده کنم! در کنارش کدنویسی هم نیاز میشه که شما نیازه یه سری ابزار بنویسین که از اون آسیب‌پذیری‌ها بتونین استفاده کنین. یا بتونین کشفشون کنین. ابزار بنویسین که بگرده و نشانه‌های وجود آسیب‌پذیری‌های مختلف رو به شما گزارش بده.

مطمئناً دستی هم میشه گشت و پیدا کرد، اما با ابزار خیلی سریع‌تر این روند اتفاق میوفته و توی دنیای امنیت و باگ‌بانتی، باید هرچه سریع‌تر تلاش کنین مشکلات رو پیدا کنین. پس با automation (خودکارسازی)، کارهاتون رو پیش میبرین.

درواقع به قول وبسایت «serveracademy» میگه قبل اینکه بخواین یه چیزی رو هک کنین، یادش بگیرین:

"Want to hack websites? Start designing and hosting websites. Want to hack MySQL databases? Install, configure and administer a SQL database!"[46]

درواقع قبل هک، باید با اون چیز کار کنی. قبل هک وبسایت، یاد بگیر وبسایت چطور طراحی میشه. وقتی شما یه وبسایتی رو طراحی می‌کنی و طراحیتو بررسی می‌کنی، میگی آهاا اینجاش مشکل داشت! اینجاش ممکنه اشتباه طراحی شه! اینجاش ممکنه یه برنامه‌نویس اشتباه کنه. اینجاش نقطه حساسه و... . حالا می‌تونی بهش حمله کنی حالا! حالا نقاط رو می‌شناسی و می‌تونی نفوذ کنه.

به نظرتون دیگه ممکنه کجاها این نوع آسیب‌پذیری رخ بده؟

+ مثلاً توی ساختن اکانت. موقعی که اکانت ساخته میشه این اتفاقاً میوفته:

```
1- Check if the email is found in database or not?
2- Create account
```

46 https://www.serveracademy.com/blog/how-to-become-a-hacker/

3- Add the email to the database

خب پس اگر چند درخواست همزمان ارسال شه، چون هنوز ایمیل وارد دیتابیس نشـده، قبـول میشـه و چند اکانت ساخته میشه!

خب حالا بریم یه کد بررسی کنیم که فرض کنین دو ترد همزمان به یه متغیر دسترسی پیدا میکنن:

```c
#include <stdio.h>
#include <stdlib.h>
#include <pthread.h>

// gcc -pthread fileName

int x = 0;

void *raceCondition(void *vargb);

int main()
{
    pthread_t t1, t2; // thread id

    pthread_create(&t1, NULL, raceCondition, NULL);
    pthread_create(&t2, NULL, raceCondition, NULL);

    pthread_join(t1, NULL);
    pthread_join(t2, NULL);

    printf("%d", x);
    return 0;
}

void *raceCondition(void *vargb)
{
    for (int i = 0; i < 1000000; i++)
          x++;

    return NULL;
}
```

ما یه متغیر X داریم. دو ترد (فرض کـنین دو برنامـه) میخـوان همزمـان این X رو بـردارن و یکی بهش اضافه کنن. هر ترد هم میاد ۱ میلیون بار این کار رو انجام میده. خب احتمالاً شاید بگین خب ۱ میلیون بار

ترد اول و ۱ میلیون بار ترد دوم. پس در نهایت X ما برابر ۲ میلیونه. اما بیایم یه چیزی رو در نظر داشته باشیم! گفتیم همزمان میخوان دسترسی پیدا کنن! یعنی اینطوری میشه:

ترد اول	ترد دوم
مقدار x رو نگاه میکنه	مقدار x رو نگاه میکنه
میگه خب یکی بالاتر از x چی میشه؟ همون رو بهش میده. مثلاً میگه x الآن دو هست، پس بعدش x باید ۳ بشه.	ترد دوم هم همون کار رو میکنه.

اما یه مشکل! در لحظه اول، هردو نگاه میکنن و هردو همزمان میبینن که مقدار X برابر صفر هست و باید یه دونه اضافه شه. پس هردو مقدار بعدی X رو میکنن ۱! خب اولی میکنه ۱، دومی هم پشت سرش همزمان میکنه ۱!

بعدش دوباره همزمان به متغیر نگاه میکنن میبینن X یک هست. خب باید دو بشه. هردو چون دارن موازی حرکت میکنن، هردو ۲ میکنن! یعنی درواقع مثل دو خط موازی همینطور با هم پیش میرن! پس در نهایت مقدار X همون ۱ میلیون میشه اگر هر دو دقیقاً با هم موازی و همزمان باشن! درسته دوتا تابع دارن یک میلیون بار کار انجام میدن، اما چون هردو روندشون همزمانه، همون ۱ میلیون میمونه![۴۷]

اما خب چون هیچ چیز توی کامپیوتر دقیقاً همزمان نیست، و بالاخره یه خورده تفاوت سرعتی بین تردهای مختلف و کندی و کاستی و... هست. پس یه خورده چون همزمان نیستن، یکم از ۱ میلیون بیشتر میشه. میشه ۱ میلیون و خوردی.

نمونه‌های دیگه از Race Condition:

+ https://stackoverflow.com/a/34550
+ Towards Systematic Black-Box Testing for Exploitable Race Conditions in Web Apps[48]

بکدور در لینوکس

به نظرتون مشکل این کد کجاست؟[۴۹]

```
if ((options == (__WCLONE | __WALL)) && (current->uid = 0))
    retval = -EINVAL;
```

---

۴۷ کد اسمبلیشو می‌تونین با وبسایت godbolt.org چک کنین که میره از مموری می‌خونه میاره توی رجیستر و بعد که اضافش کرد، دوباره می‌ره می‌ذارتش توی مموری. (بدون optimization)

48 https://essay.utwente.nl/78020/1/Emous_van_MA_EEMCS.pdf

49 More details about this topic:
https://freedom-to-tinker.com/2013/10/09/the-linux-backdoor-attempt-of-2003/
https://lwn.net/Articles/57135/

- خب مشکل کجاس؟

+ توی `if` اگر نگاه کنیم، به جای `==` اشتباهی `=` گذاشته! اما خب این چه مشکلی ممکنه به وجود بیاره؟!

بیایم برگردیم به سال ۲۰۰۳. زمانی که کد اصلی و برنچ اصلی لینوکس توی دوجا وجود داشت. هردو با هم سینک بودن ولی.

هر کد که قراره تایید بشه و وارد کرنل بشه، یه کد بهش اختصاص میدادن و یه متن که فلانی اینو تایید کرده.

اما دیدن که توی یکیش یه کد هست که کد نه تایید شده نه توضیحات داره. یکمی واردش شدن و دیدن که عه! این کد یه مشکل داره! `==` به کار نبرده و assignment (یکدونه مساوی) به کار برده.

اما مشکل کجاست؟

یکمی دقت کنیم. `uid` یعنی user id. و توی لینوکس کاربرها آیدی دارن. آیدی کاربر روت، ۰ هست. خب حدس زدیم پس که این کد درواقع id کاربر رو ۰ میکنه یعنی دسترسی روت میده.

درواقع اگر برنامه‌ای از این سرویس استفاده میکرد، خود به خود توی `if` روت میشد و دسترسی روت پیدا می‌کرد. دسترسی روت بالاترین سطح دسترسی هست. یعنی درواقع تقریبا هر کاری میتونه با سیستم کنه!

بعدا مشخص شد که سرور رو هک کردن و این کد رو وارد کردن. این درواقع تلاشی بود برای واردکردن یه backdoor[۵۰] توی لینوکس! که البته ناموفق بود و معلوم نشد که کی این کار رو کرده.

البته این کد هیچوقت وارد هسته اصلی لینوکس نشد و تاییدکنندگان کدها پیداش کردن!

Out-of-bounds memory access

مثال ۱:

بیایم برگردیم به کد چک‌کردن برابر بودن دو استرینگ. به نظرتون اگر کد رو اینطور می‌نوشتم، چه مشکلی داشت؟ (جز تایمینگ که بحث کردیم)

```
int isPassValid(char pass[], char inp_pass[])
{
    for (int i = 0; i < len(pass); i++)
    {
        if (pass[i] != inp_pass[i])
            return 0;
    }
    return 1;
}
```

۵۰ توضیح ساده بکدور: یه در پشتی که بتونن از طریق اون بدون اعتبارسنجی، به سیستم نفوذ کنن!

+ بله! «out-of-bounds read». درواقع ممکن بود سایز ورودی من مثلاً ۲ باشه و سایز پسورد ۱۰. خب i ما تا ۹ پیش می‌رفت. یعنی من خونه‌های جلوتر inp_pass که اصلاً مال خودش نیست رو هم دارم می‌خونم!

مثال ۲:

+ برای اینکه هنگام کار با آرایه (لیست)، یهو به ایندکس اشتباهی دسترسی نداشته باشین، به شرط while یا if دقت کنین. مثلاً:

```
size = 5
arr = [1, 2, 3, 4, 5]
while size >= 0:
    print(arr[size-1])
    size -= 1
```

این کد مشکل داره. توی پایتون ۵ رو دوبار چاپ می‌کنه و توی زبونای دیگه out-of-bounds memory access هست. پس حواستون باشه که ببینین شرط رو درست تعیین کنین که اشتباهی به قسمتی اشتباه دسترسی پیدا نکنین. ببینین با توجه به استفاده از arr و ایندکس مورد نظرش، شرط رو باید چطور تعیین کنم که تا آخرش مثلاً بره ولی نزنه جلو یا عقب.

مثال ۳:

حالا به نظرتون کد زیر چه مشکلی داره؟ (منطق کد اینه که یه عدد از کاربر ورودی می‌گیره و بعدش به مقدار اون عدد، کرکتر از ورودی دریافت می‌کنه و در آرایه str ذخیره می‌کنه. همین!)

پ.ن: کاری به اینکه scanf ناامنه نداشته باشین. فرض کنین هم scanf و هم getchar کار خودشونو درست انجام میدن.

```
int main()
{
    int length;
    scanf("%d", &length);
      getchar(); // free buffer (Because we have an \0 after getting integer)
    char *str = (char *)malloc(sizeof(char) * length);
    int i;
    for (i = 0; i < length; i++)
    {
        str[i] = getchar();
    }
    str[i] = '\0';
    free(str);

    return 0;
```

```
}
```

پاسخ:

مشکل اینجاست که من دارم به تعداد اعضا فضا اختیار می‌کنم و جایی عملاً برای «\0» باقی نگذاشتم و «\0» عملاً باید در خونه‌ای خارج از جایی که اختیار کردم قرار بگیره. باید وقتی فضا اختیار می‌کردم، یکی بیشتر از تعداد اعضا اختیار می‌کردم که در نهایت بتونم «\0» هم تهش بذارم.

نکات دیگر:

+ تغییر یافتن سایز یه آرایه به اشتباه وسط کد و سپس استفاده ازش[51] (متغیری که سایز رو نگه میداره، نباید وسط راه تغییر پیدا کنه! یهو در آخر کد نیازش میشه و یادمون نیست که تغییرش دادیم! حواستون باشه که این متغیر لزوماً پس از تغییر بزرگ‌تر از متغیر اولی نیست! گاهی ممکنه اشتباهی منفی شده باشه. پس شرط باید اینطوری باشه:

```
if (index < size) and (index >= 0):
```

که هر دو حالت رو چک کنه. همیشه ته ذهنتون بگین که حالت بالاتر یا پایین‌تر هردو ممکنه رخ بده. نه لزوماً بالاتر!)

Buffer Overflow

دو ویدیو زیر از جادی هست که دربارش به شکل خیلی خوب و قابل فهمی توضیح داده. همچنین مثال دنیای واقعی و کدی که آسیب‌پذیره رو هم زده. به ترتیب ببینینشون:

جادی تی وی – حمله های استک و بافراورفلو، قسمت یک از دو:

https://www.youtube.com/watch?v=RLlQFfZoEB8

جادی تی وی – حمله های استک و بافراورفلو، قسمت دو از دو:

https://www.youtube.com/watch?v=xjRFubg5Ghs

Some interesting methods to prevent buffer overflow:

https://www.schneier.com/blog/archives/2006/08/security_and_mo.html

یکی از چیزای مهم توی برنامه‌نویسی‌های زبونایی مثل C/C++، رعایت درست «\0» هست. اگر درست رعایت نشه، باعث مشکلات مموری میشه.

خب حالا که با امنیت آشنا شدیم، شاید براتون سؤال پیش اومده باشه که چجور ازش پول در بیاریم. چه شغلایی هست؟

کار شرکتی

51 https://stackoverflow.com/questions/74611037/how-to-fix-out-of-bound-memory-accessaccess-exceeds-upper-limit-of-memory-block

تو شرکت استخدام شین و حواستون باشه که مثلاً داده‌ها به صورت درستی ذخیره شن. امن ذخیره شن. آیا یه وقت نسخه‌ای به صورت غیررمزشده جایی ارسال یا ذخیره نشده. درواقع محافظت کنین از داده‌ها.

آیا برنامه‌ها همگی بروز هستن؟ مثلاً یه برنامه‌ای می‌نویسین که بره نسخه‌های برنامه رو چک کنه و ببینه آیا آپدیتن؟ آیا درست کار می‌کنن؟ آیا سیستمی کرش نکرده؟

بررسی لاگ:

ببینین سرورها مثل کامپیوترهای شرکت هستن. خیلی چیزا مثل اطلاعات مشتریان، زمان انجام هر کار و... توش ذخیره میشه. یکی از مهمترین کارهای یه مدیر سرور اینه که بیاد لاگ‌ها رو بررسی کنه.

– لاگ چیه؟

+ لاگ‌ها یه سری توضیحات درباره یه چیز هستن. مثل یه سری توضیحات درباره یه برنامه، یه کاربر، یه چیز. درواقع بهتره بگیم رخدادها رو نشون میدن.

باز واضح نبود درسته؟ بیایم با مثال ببینیمش. مثلاً متن‌های زیر، لاگ وبگردی یه فرد هست:

```
28/05/22 13:20 duckduckgo.com
28/05/22 13:24 eff.org
28/05/22 13:32 mozilla.org
28/05/22 13:38 brave.com
28/05/22 13:46 proton.me
28/05/22 13:53 gimp.org
28/05/22 14:03 wikipedia.org
28/05/22 14:09 torproject.org
```

میگه روز ۲۸ ماه ۵ سال ۲۰۲۲، ساعت ۱۳:۲۰ از وبسایت duckduckgo.com بازدید کرده. بعدش ساعت ۱۳:۲۴ رفته به وبسایت eff.org بعدش ۱۳:۳۲ رفته به mozilla.org و... .

خب حالا مدیر سرور شاید بخواد دنبال یه چیز خاص بگرده. مثلاً دنبال این باشه که ببینه که فلان کاربر، در کدوم روز هفته بیشتر از روزای دیگه وبگردی می‌کنه که مثلاً بخواد اطلاعاتی از الگو وبگردی اون فرد به دست بیاره. (همون آنالیز و بررسی وبگردی شما توسط ISP ها و دولت‌ها)

یا مثلاً لاگ‌هایی هست که نشون میده هر دقیقه چه فایلی تغییر کرده. چه فایلی پاک شده. چه فایلی عوض شده و... . درواقع شما مثل یه نگهبان نظارت می‌کنی و دنبال چیزای مشکوک می‌گردی. مثلاً یهو می‌بینی که توی یه دقیقه یهو ۲۰۰ تا فایل عوض شدن. خب شاید مشکلی هست. میری چکش می‌کنی.

خب آیا شما می‌تونی توی ویندوز سرچ کنی دنبال یه فایلی می‌گردم که دیروز ساعت ۱۰ بروز شده و اول فایل حرف D هست و آخرش حرف کوچیک k هست؟

نه نمی‌تونین! اینجاست که سیستم‌عامل‌ها یه سری ابزاری که با کدنوشتن میشه باهاشون کار کرد در اختیارتون می‌گذارن که برید پیداشون کنین.

این ابزارها در یه محیطی هستن که اسمش ترمیناله. ابزارهای کوچیک و ریز ولی وقتی در کنار هم قرار می‌گیرن، به شدت قدرتمند. من می‌تونم بگم که:

۱. بیا قسمت روز رو جدا کن.

۲. حالا که قسمت روز رو جدا کردی، ببین از هر روز چندتا سایت هست. یعنی مثلاً توی روز ۲۸، این کاربر توی چندتا سایت رفته.

۳. حالا بیا بر اساس تعداد سایت‌ها، مرتبشون کن. حالا با نمودار بهم نشون بده. که ترتیب چه‌جوری هستن.

اینطوری راحت میشه دید مثلاً توی کدوم روز، بیشترین تعداد سایت رو رفته و قشنگ به صورت نموداری دیدش! اما هرکدوم از مراحل بالا، استفاده از یه ابزار ریز که توی ترمینال موجوده هستن. برای درک بهتر حتما ویدیو زیر رو از «جادی» ببینین(۱۲ دقیقس):

«چرا گنو/لینوکس رو دوست دارم: ترکیب ابزارها و کشیدن نمودار لاگ‌ها در کامندلاین»[۵۲]

https://youtu.be/hCkll2FEOLU

دیدین؟ چقدر قشنگ میشه با کنار هم قرار دادن ابزارها به نتیجه دلخواه رسید و وقتی به ابزارها مسلط باشین، به راحتی توی یه دقیقه به چیزی که توی ویدیو «جادی» نشون داد می‌رسین! حالا فهمیدین چرا میگیم لینوکس قدرتمنده؟! به خاطر ابزارهاشه! اما اگر می‌خواستین همینو با برنامه‌نویسی پیش برین، خیلی خیلی سخت‌تر میشد و زمان زیادی میبرد! علاوه بر اون ممکن بود ابزارهای اجرای اون برنامه‌ای که نوشتین توی سرور نباشه!

درواقع شما نیازه با برنامه‌ها لاگ‌ها رو چک کنین. دنبال چیزای عجیب باشین. درواقع ببینین کجا سیستم ممکنه به مشکل برخورده باشه؟ کجا ممکنه اونطوری که می‌خوایم عمل نکرده باشه.

- از برنامه‌نویسی میشه کمک گرفت؟

+ یه مشکل! شاید من به زبان C تسلط داشته باشم و ابزاهای نوشتن و اجرای اون برنامه توی سرور نباشن! همچنین من وقت این رو نداشته باشم که ده ساعت توی سرور آنلاین باشم و بخوام برنامه رو بنویسم! از قبل هم نمی‌تونم برنامه رو بنویسم و بفرستم توی سرور! چون انتقال فایل به سرور بنا بر دلایلی بسته شده.

یا کارهای دیگه:

+ اگر یه وقت مشکل امنیتی برای یه برنامه به وجود اومد، قبل اومدن آپدیت و رفع اون، چه کارهایی نیازه انجام بدین؟

مثلاً یه مشکل امنیتی توی یه قابلیت برنامه‌ای که شرکت داره استفاده می‌کنه پیش میاد. شما نیازه تا قبل اومدن آپدیت، فلان قابلیت برنامه رو غیرفعال کنین که یه وقت سیستماتون از اون قسمت هک نشه.

---

۵۲ گنو/لینوکس یه سیستم‌عامل شبیه ویندوزه که پر از این ابزارهای تحت ترمیناله. درواقع بخش قابل‌توجهی از سرورها لینوکسین.

+ حتی افرادی داریم که سعی می‌کنن همیشه مکانیزم‌های ورودی خروجی سازمان رو چک کنن و ببینن که یه وقت مشکلی نباشه و کسی نتونه بی‌اجازه وارد شه.

+ اگر یکی وارد شد، لاگ‌ها رو چک می‌کنن ببینن چی شده و چه تغییراتی توی سیستم ایجاد شده؟ چیا دزدیده شده؟ کی بوده؟ دنبالش بگردیم. یا چیکار کنیم که دفعه بعد اینطوری ضربه نخوریم.

Penetration Testing

خیلی ساده بخوام بگم، سعی می‌کنن توی این قسمت برن با یه سری شرکت قرارداد ببندن که فلان جا رو مورد تست و بررسی قرار بدن. در نهایت هم یه گزارش می‌دن که چه مشکلاتی پیدا شده و راه حلشون چی هست.

Bug Bounty

معمولاً شرکتا برای اینکه شما باگ‌هاشونو و مخصوصاً باگ‌های امنیتی و مشکلات امنیتی رو پیدا کنین (حالا بیشتر منظور باگ‌های امنیتی هست)، بهتون جایزه میدن. یعنی شرکت میگه من امنیت کاربرا برام مهمه. یه مسابقه برگزار میکنم که اگر کسی مشکل امنیتی توی سرویسای من پیدا کرد، بهم گزارش بده تا برطرفش کنم و منم به جاش به اونی که این مشکل رو پیدا کرده جایزه میدم. معادله‌ی برد - برد. هم شرکت امن میشه و هم متخصص امنیت پول گیرش میاد!

- خب شاید سؤال پیش بیاد که برای چی شرکتا بابت پیدا کردن باگ پول میدن؟

+ اولاً شرکت ممکنه خودش به امنیت کاربرانش توجه داشته باشه و بگه خب اطلاعات خصوصی کاربران و حق کاربران برای من مهمه. پس براش پول خرج می‌کنم.

دوم اینکه در کشورایی که حقوق کاربران خیلی مهمه مثل کشورای اروپایی، اگر اطلاعات کاربران به خطر بیوفته، اون شرکت جریمه میشه و این جریمه خیلی خیلی سنگینه. حتی ممکنه چند ده میلیون دلار باشه! برای همین شرکت میگه خب چه کاریه! من مثلاً دو میلیون دلار پول میدم به هانترها و شکارچی‌های باگ که بیان باگ پیدا کنن که نخوام مثلاً ۵۰ میلیون دلار جریمه شم. پس اگر شرکت نخواد هم مجبوره.

- خب شاید سؤال پیش بیاد که مثلاً فلان شرکت مگه تیم امنیتی نداره که برنامه رو چک کنن؟ چرا پس خودشون رو در معرض افراد ناشناس قرار میدن که تستش کنن؟ آیا اون فرد ناشناس بعد پیداکردن مشکلات امنیتی، نمی‌تونه مشکل رو پنهون کنه و گزارش نکنه و بره مثلاً خودش به آدمای بد بفروشتش؟

ببینین درواقع هر شرکتی یه سری متخصص امنیت داره که روزانه دارن تست انجام میدن و بررسی می‌کنن که سیستم امن باشه. اما این کافی نیست.

اولاً هرکس یه زاویه دیدی داره و خب هیچ‌کس همه چیز رو بلد نیست. پس درسته شما ۱۰ نفر دارین که تست انجام میدن، اما خب آیا بهتر نیست که صدها نفر وجود داشته باشن که تست انجام بدن؟

مطمئناً باگ‌های بیشتری پیدا میشن و شما امن‌تر میشین.

دوم اینکه کسی که توی شرکت شماست، حقوقش رو داره و خب اون عطش یافتن باگ رو نداره. میگه من که حقوقم رو دارم. حالا هرچی تونستم آسیب‌پذیری پیدا می‌کنم. اما عطش نداره.
اما توی برنامه باگ‌بانتی، افراد عطش دارن که باگ بیشتری پیدا کنن. شما هم صرفاً زمانی بهشون پول میدی که باگ پیدا کنن. درواقع برد-برد هست. هرچی شما رو امن‌تر کنن، پول بیشتری در میارن. پس عطش بالایی دارن.

اما موضوع اینه که آیا میشه به افراد ناشناس اطمینان حاصل کرد؟ شاید باگ پیدا کرد و گزارش نکرد!
خب شما وقتی خودتون تست انجام میدین یا میسپرین به یه شرکت که بیاد تست و penetration انجام بده، اونا هم ممکنه باگ خفنی پیدا کنن و گزارش ندن و توی بازار سیاه و دارک وب بفروشن! اونا هم ممکنه این کار کنن. حتی کارمند خودتون! طرف حقوقش رو گرفته. مثلاً ۲۰ تومن دادین که تست انجام بده. ۲۰ تومنش رو گرفته. پیش خودش میگه ۴ تا از ۵ تایی که پیدا کردم رو گزارش میدم و آخری می‌ذارم برای خودم که از این باگ هم پول در بیارم و توی بازار سیاه بفروشم.

اما توی باگ بانتی، افراد عطش گزارش باگ دارن. چون هرچی بیشتر گزارش بدن، پول بیشتری خواهند داشت و همچنین فرض کنیم که یکی هم پیدا شد که گزارش نداد و خواست توی بازارهای غیرقانونی بفروشه. اوکی! اما افراد خوب دیگه‌ای هم هستن که احتمالاً اون آسیب‌پذیری رو پیدا می‌کنن و زودتر گزارشش میدن. یعنی رقابت هست. تو اگر نگهش داری، ممکنه یکی دیگه پیداش کنه و پولو اون ببره. پس ترجیح میدی که خودت گزارشش بدی!
حتی گزارش باگ برات رزومه میشه. پس‌فردا بخوای بری توی یه شرکت کار کنی، می‌تونی رزومت رو نشون بدی که چقدر آسیب‌پذیری‌های مهم رو پیدا کردی. اونجا استخدام‌کننده متوجه میشه که دانش خوبی داری که فلان‌قدر باگ گزارش کردی و استخدامت می‌کنه.

- عه چه عالی! پس برم آسیب‌پذیری‌های مختلف رو یاد بگیرم تا بتونم پولدار شم.
+ وایسین! به همین سادگی‌ها هم نیست! مثل هر کار دیگه‌ای، این کار هم یه سری پیش‌زمینه می‌خواد. (صحبت کردیم که باید دانش داشته باشین و همچنین در ادامه با مسیر راه هم آشنا میشیم)
و خب یه سری قوانین داره. شما نمی‌تونین همینطوری برین تست انجام بدین. اگر این کار کنین، احتمالاً یه کاری می‌کنین که غیرقانونیه و مجرم شناخته میشین و **میندازنتون زندان!** :)

Denial-of-service attack (DoS) & Distributed Denial-of-service attack (DDoS)

یکی از حملاتی که میشه به یه سرور کرد، حمله Dos/DDos هست. همونطور که از اسمش معلومه یعنی محروم‌سازی از خدمات.
ببینین فرض کنین شما یه کارمند هستین و قراره کار آدمای مختلف رو انجام بدین. هرکی میاد، اول باهاش دست میدین و درود میگین و بعدش کارشو میپرسین. حالا اگر یکی بخواد عملکرد تو رو از بین ببره

چیکار میتونه کنه؟ یه دفعه هزار نفر آدم رو بفرسته دورِت و هرکدوم فقط سلام و درود بگن و دست بدن و بعد برن. اینطوری اینقدر سر تو شلوغ میشه که نمیتونی به مشتریای اصلی جواب بدی.

توی DDoS هم همین اتفاق میوفته. نحوه کار ارتباط با سایتای HTTPS چجور بود؟ اول کامپیوتر شما یه داده ارسال میکنه که آره سلام هستی؟ سرور هم میگه سلام هستم. این بهش میگن هندشیک. عین دست دادن. حالا اگر میلیون‌ها درخواست سلام هست؟ به سرور ارسال شه، سرور هی باید به همه جواب بده که آره سلام هستم. خب این باعث شلوغ شدن سرور میشه. عملاً سرور اونقدر شلوغ میشه و هی داره جواب سلامارو میده که دیگه اصلاً وقت نمیکنه کار دیگه‌ای کنه. عملاً سایت خیلی خیلی کند میشه و از دسترس خارج میشه. اینطوری مشتریان اصلی هم نمیتونن به سرویس دسترسی داشته باشن. حتی شما هم ممکنه نتونی خوب با سرورت در ارتباط باشی!!

مثال دنیای واقعی:

دیدین وقتایی که مثلاً اعلام نتایج کنکوره سایت سنجش خیلی کند میشه؟ چون مثلاً یه میلیون نفر دارن وارد سایت سنجش میشن. عملاً سرور کشش نداره که جواب اینهمه آدم بده و کارهاشون رو راه بندازه. اینطوری سایت به شدت کند میشه.

حالا مثلاً شما میری روی وبسایت دوستت که اجازه هم داده DDoS[۵۳] می‌زنی ولی نمی‌دونی که عملاً DDos تو روی هاستی که گرفته هم اثر داره. (سایت روی هوا که نیست! روی یه سروره! مثلاً داری آسیب به سروری که برای وبسایتش اجاره کرده می‌زنی. شما اجازه دوستت رو داری ولی اجازه صاحب سرور رو نداری که!)

درواقع شرکتا ممکنه برنامه‌ای برای دادن بانتی داشته باشن و ممکنه نداشته باشن. اگر نداشته باشن که هیچ! شما نمی‌تونین تست انجام بدین.

اما ممکنه برنامه داشته باشن. اونا درواقع میان قوانین و شرایط تست رو می‌نویسن و توی سایتشون قرار میدن و شما **باید** دقیقاً مطابق اون شرایط پیش برین. اگر کار اشتباهی انجام بدین و یا حتی همینطور تست ساده انجام بدین ← **شکایت و زندان**

درواقع اونجا میگن که در فلان قسمت سایت، فلان قسمت برنامه و... با فلان شرایط می‌تونین تست انجام بدین. خارج اون نباید عمل کنین! اگر فلان نوع آسیب‌پذیری رو پیدا کردین، مثلاً ۱۰۰ دلار میدیم. فلان نوع ۲۰۰ دلار و...

خب شما گزارش دقیق مطابق چیزی که گفتن می‌نویسین و براشون ارسال می‌کنین. اونا بررسی می‌کنن که آیا اوکیه و یا نه. اگر اوکی بود، مبلغ رو بهتون میدن. (گاهی هم ممکنه باگتون پول نباشه؛ مثلاً اشتراک یه ساله سایت رو بدن)

- خب اگر باگ رو به خودشون گزارش بدیم، بعداً دبه در نمیارن که ما خودمون می‌دونستیم و یا گزارشت رو قبلاً یکی دیگه داده بود و مال تو قبول نیست و... که نخوان پول بدن؟

+ به طور معمول اگر برنامه بانتی داشته باشه و جای اوکی هم باشه، نه. اما بله گاهی اوقات هم ممکنه همچین چیزایی پیش بیاد.

---

۵۳ سَر سرور رو با درخواستای الکی هی شلوغ کنه که بار روی دوش سرور زیاد شه و عملاً اینقدر سرش شلوغ شه، که نتونه به درخواستای درست هم جواب بده. حتی اونقدر داغ کنه که آسیب سخت‌افزاری ببینه!

اینجاست که صحبت پلتفرم‌هایی وسط میاد که میان به عنوان واسط عمل می‌کنن. درواقع یه سری شرکت هستن که میان می‌گن شرکتا شما بیاین قوانینتون رو توی سایت ما قرار بدین و اعلام حضور کنین که افراد بیان تست انجام بدن. هرکی هم آسیب‌پذیری پیدا کرد، به من گزارش بده. من تأیید می‌کنم که معتبره یا نه و بعدش میدم به شرکت. پول رو از شرکت می‌گیرم و میدم به هکر.

درواقع به شکل یه واسط عمل می‌کنن که انصاف رو برقرار کنن و چک کنن ببینن باگ معتبره یا نه. مثلاً «HackerOne» و «Bugcrowd» پلتفرم‌های معروف خارجی هستن.

یادتونه که گفتم در باگ بانتی رقابت زیاده؟ رقابت اونقدر بالاست که اگر شما یه باگ پیدا کنی و گزارش ندی، ممکنه در عرض چند ساعت یکی دیگه پیدا شه و گزارش بده! بله اینقدر رقابت بالا و فشرده هست. (برای همین هم هست که افرادی که تازه شروع کردن، ممکنه نتونن توی این عرصه رقابت کنن. برای همین مثلاً میگن از پلتفرم‌های ایرانی شروع کنین که ساده‌ترن و رقابت کمتره و باگ‌های خیلی ساده‌تری میشه پیدا کرد.)

برای همینم تأکید کردم که برنامه‌نویسی رو به صورت ابزارمحور یاد بگیرین که بتونین کارها رو به صورت خودکار انجام بدین که بتونین زودتر از بقیه آسیب‌پذیری پیدا کنین.

- عه پس برم کالی لینوکس[۵۴] نصب کنم و هزار تا ابزار نصب کنم که بتونم آسیب‌پذیری پیدا کنم.

+ توجه کنین که ابزارهای عادی، شما رو هکر نمی‌کنن. بلکه اون دانش اصلی شماست که شما رو به یه مهندس امنیت تبدیل می‌کنه. بدون داشتن دانش، میلیون تا ابزار هم داشته باشین، نهایتش دوتا آسیب‌پذیری ساده پیدا می‌کنین و تهش در همون سطح می‌مونین. اما کسی موفقه که دانش عمیق کسب کنه و یه بخشیش با ابزارهای عادی و یه بخشیش با ابزارهایی که با دانشش نوشتتشون، استفاده می‌کنه.

همچنین کسی کالی رو روی سیستم اصلیش نمی‌ریزه. چون کالی برای تست هست، تنظیماتش هم باید مطابق کارهای تست انجام بشه. پس کالی رو روی ماشین مجازی می‌ریزن. *(مراجعه شود به بخش ماشین مجازی چیست)*

تذکر! حافظه پلیس‌ها درباره کارهای غیرقانونی و نفوذهای غیرقانونی پاک نمیشه. شما نمی‌تونین بگین حالا یه بار یه کار بد انجام میدم و بعدش دیگه حواسم هست که همش کار خوب کنم. این جواب نمیده! چیزا توی اینترنت برای همیشه موندگارن!

شما برای اون کار بدتون (حتی اگر ده سال هم گذشته باشه ازش)، جریمه میشین.

بذارین داستانی رو براتون تعریف کنم:

یه زمانی یه بدافزار (نرم‌افزار بد) به نام WannaCry، توی اینترنت پخش شده بود که نصف کامپیوترهای دنیا رو آلوده می‌کرد. شما یهو کامپیوترت رو روشن می‌کردی می‌دیدی ای دل غافل! همه چیزاش قفل شده و بهت می‌گه اگر می‌خوای باز شه، پول بده تا بازش کنم برات! حالا باید به هکر بد باج میدادی که برات بازش کنه.

یه فرد خوب پیدا شد که دنیا رو از دست این بدافزار نجات داد. اما بعداً FBI گرفتش.

- عه شاید بگین FBI مگه بیکار بود گرفتش؟ طرف دنیا رو نجات داده! چرا بگیرنش؟!

---

۵۴ یه سیستم‌عامل گنو/لینوکسی که پر از ابزارهای آماده برای هکینگ هست.

+ نه FBI بیکار نبود! بهش گفت دستت درد نکنه برا نجات دنیا، ولی قبلاً خود برنامه بد نوشته بودی که کامپیوترها رو آلوده می‌کرد. اوکی کار خوب کردی، اما باید جواب کارهای بد گذشتت رو بدی.[۵۵] و بعداً به زندان و جریمه محکوم شد[۵۶]:

"defendant will enter a plea of guilty carry the following maximum terms of imprisonment and fines:

- Count One: up to 6 years in prison, up to $250,000 in fines, up to 1 year of supervised release, and a $100 special assessment.
- Count Two: up to 5 years in prison, up to $250,000 in fines, up to 1 year of supervised release, and a $100 special assessment."

تذکر!

بدون دانش کاری انجام ندین. شما برای مسخره بازی و شوخی میری یه ویروس کامپیوتری[۵۷] (یه برنامه بد که می‌ره کرم ریزی می‌کنه روی سیستم)، رو روی کامپیوتر دوستت تست می‌کنی ببینی چجوریه. اما نمی‌دونی که مثلاً لپ‌تاپ دوستت به وایفای وصله و از طریق همون Wi-Fi به بقیه لپ‌تاپا هم انتقال پیدا می‌کنه :)

تذکر! حواستون باشه که بدون خوندن شرایط و همچنین بدون داشتن دانش کافی، حتی تست‌های خیلی ساده که از نظرتون هیچ هکی هم نیست انجام ندین! چون ممکنه یه چیزی به هیچ وجه از نظر شما هک و یا نفوذ غیرمجاز نباشه ولی از لحاظ قانونی باشه! پس اول دانش کسب کنین و بعد اقدام کنین!
بسیاری از جاها مکانیزم‌های لاگ دقیق دارن که دقیق مشخص میشه شما چیکار کردین و با اون می‌تونن اتهام بزنن. (حتی اگر هزارتا Tor و VPN[۵۸] برای ناشناس‌موندن استفاده کرده باشین!)

تذکر! حواستون باشه که بدافزار نوشتن و پخش‌کردنش (یا ناخودآگاه پخش‌شدنش توی شبکه)، تبعات سنگینی داره!
- پلیسا از کجا می‌فهمن که اون بدافزار رو من نوشتم؟
+ امضای برنامه‌ها و یا حتی نحوه نوشتار کد هرکس، نحوه نامگذاری چیزا، نحوه جداسازی فایل‌ها و... یونیکه. چون تفکر هر کسی با فرد دیگه متفاوته، این می‌تونه یه fingerprint شه که بتونن این برنامه رو به برنامه‌های دیگه‌ای که می‌سازین لینک کنن.[۵۹])

---

55 https://www.theguardian.com/technology/2017/aug/03/researcher-who-stopped-wannacry-ransomware-detained-in-us

56 https://storage.courtlistener.com/recap/gov.uscourts.wied.77855/gov.uscourts.wied.77855.124.0.pdf

۵۷ اصطلاح بهترش، «بدافزار» هست.

۵۸ توجه! VPN برای ناشناسی نیست! بلکه **استفاده درست** از Tor برای ناشناسی هست.

59 https://www.schneier.com/blog/archives/2016/01/de-anonymizing_.html

تذکر! روابط عمومی شرکت، ایمیل عادی شرکت و... جای گزارش آسیب‌پذیری امنیتی نیست! چون اولاً اونا نمیدونن اصلاً شما دارین چی میگین و بعدشم ممکنه بعداً به اینکه یه آسیب‌پذیری رو پخش کردین متهم شین!

تذکر! اگر آسیب‌پذیری رو پیدا کردین، زودی به صورت عمومی منتشرش نکنین! درواقع شما نباید تا قبل حل شدن مشکل اون رو منتشر کنین. چون افراد بد بهش پی می‌برن و استفاده بد می‌کنن. می‌دونم وقتی آسیب‌پذیری پیدا می‌کنین، هیجان دارین و می‌خواین اون رو با بقیه به اشتراک بگذارین، اما قبلش باید از سیاست اون شرکت مطمئن شین و ازش اجازه بگیرین که آیا اجازه دارین تجربه‌ای که کسب کردین رو با بقیه به اشتراک بگذارین؟ تا چه حد و چه جزئیاتی رو می‌تونین به صورت عمومی منتشر کنین؟

تذکر! از شرکتای نامعتبر فاصله بگیرین!

گاهی شرکتای هستن که توی سایتشون نوشتن که بله ما بانتی میدیم ولی در عمل نه تنها نمیدن بلکه بد هم برخورد می‌کنن. چند نشانه که ممکنه نشون بده اون سیستم حرفه‌ای نیست یا نامعتبره:

+ باگ پیدا کنین و تا ۱۰۰ میلیون تومن جایزه بگیرین.

هیچ توضیحی نداده که آقا من فلان باگ پیدا کردم چقدر پول میدی؟ تا ۱۰۰ میلیون بازه خیلی بزرگیه. یه عدد مشخص کن. جاهای حرفه‌ای مثلاً می‌گن اگر باگت از فلان دسته بود، ۱۰۰ دلار میدیم. اگر فلان دسته بود ۵۰۰ تا. اگر خیلی حیاتی بود، مثلاً ۵۰۰۰ هزارتا. یعنی دسته‌بندی دقیق کرده.[۶۰] نگفته تا ۱۰۰ میلیون. تا ۱۰۰ میلیون ممکنه ۱۰۰ هزار تومن هم شاملش شه.

تذکر! ما چیزی به نام استخدام هکر نداریم! مثلاً میگن: ((بیا استخدام هکر داریم با حقوق و مزایای عالی. فلان نماینده تو دولت هم رفیق ماست و همیشه هواتو داره که مشکلی برات پیش نیاد. بیا با ما کار کن و...))

به هیچ وجه سمتشون نرید! این افراد به دنبال سوءاستفاده از شما هستن. درواقع می‌خوان از شما بهره‌کشی کنن و مثلاً به بهانه اینکه می‌خوایم ببینیم چقدر بلدی و چقدر حرفه‌ای هستی و رزومه می‌خوایم ازت، بیا فلان جا هک کن و فلان سایت رو دیفیس کن.[۶۱] اما در نهایت می‌خوان که از شما بهره‌کشی کنن که بعداً هر مشکلی پیش اومد بندازن گردن شما.

تذکر! همکاری با سازمان‌های امنیتی (چه در داخل کشور، چه در خارج کشور)، کار خطرناکیه.

درواقع شما ممکنه درگیر یه سری بازی بشین که تا همیشه شما رو درگیر کنه.

اگر با سازمان امنیتی در خارج کشور همکاری کنین، به عنوان جاسوس می‌گیرنتون.[۶۲]

اگر با سازمان امنیتی در داخل کشور همکاری کنین، بعداً ممکنه در رفتن به کشورای دیگه به مشکل بر بخورین. (چه از طرف داخل جلوتون بگیرن و چه از خارج توی لیست تحت تعقیب باشین.[۶۳])

---

60 Example: https://hackerone.com/torproject

۶۱ به طور خلاصه به عوض‌کردن صفحه یه وبسایت به چیزی که می‌خواین، website defacement گفته میشه.

۶۲ فرقی نداره ایرانی باشین یا کانادایی یا هرچیز دیگه. زمانی که با سازمان امنیتی در یه کشور دیگه همکاری کنین، ممکنه به جرم جاسوسی بگیرنتون! یعنی فرض کنین شما استرالیایی هم باشین ولی با کانادا همکاری کنین، استرالیا دستگیرتون می‌کنه.

63 https://www.fbi.gov/wanted/cyber/behzad-mohammadzadeh

حواستون به شرکتای سایه باشه.
بعضی شرکتا هستن که در اصل وابسته به سازمان امنیتی هستن ولی در سایه عمل می‌کنن که شما فکر کنی یه سازمان خصوصی هستن. پس حواستون باشه توی تله‌شون نیوفتین!

تذکر! مراقب گروه‌های هک و امنیت (مثل گروه‌های تلگرمی و اینا) باشین. میدونم بعضیاتون خیلی علاقه دارین وارد این گروه‌ها شین و بیشتر درباره هک و امنیت بدونین و از بقیه چیز میز یاد بگیرین.
ولی اول اینکه ۹۰ درصد گروه‌های هک و امنیت داخل تلگرم، دانش چندانی بهتون نمیدن تقریباً صفر! بخش بزرگیش درباره کرک‌کردن و پخش‌کردن برنامه‌های کرکی که توسط کرکرای روسی کرک‌شدن و اکانت کرکی و پخش بدافزار و آلوده‌کردن این کارهای غیرقانونیه! و همچنین معمولاً افرادی که دانش زیادی دارن اونجا فعالیت نمی‌کنن. صرفاً چهارتا ترجمه اشتباه و غیردقیق و ایناست. چهارتا چیز اشتباه و غیردقیق می‌گن. من بارها حتی کانال‌های معروف با صدها هزار نفر رو دیدم ولی خیلی وقتا تابلوعه که یه چیز رو طوطی‌وار دارن می‌گن که خیلیاش هم اشتباهه!
ولی یک چیزم بدونین اون افراد، معمولا افرادی هستن که از شما یه چیزایی بیشتر میدونن و بینشون هم افراد مجرم سایبری (مثل باج‌افزار نویسا) هستن. پس هر فایلی رو از اونجاها دانلود نکنین یا حتی PDF! بله!PDF می‌تونه آلوده باشه که بعداً دربارش صحبت می‌کنیم! پس جوانب احتیاط رو رعایت کنین. نمی‌خوام بترسونمتون ولی حداقل چیزا رو در مورد بدافزار و انواع فایل‌ها و لینک آلوده و اینا بدونین.

تذکر! به دلیل تحریم، گاهی شما نمی‌تونین با شرکتای خارجی کار کنین. (بعضی وقتا! قوانین رو مطالعه کنین که متوجه شین میشه یا نه و اگر نمیشه و می‌خواین کار کنین، نباید بفهمن شما ایرانی هستین... :))

تذکر! یکی از چیزایی که اگر در جای دیگه و مخصوصا رشته کامپیوتر میرید درس می‌خونین، با تکیه بر اینکه نیاز به پول و حمایت در یه کشور دیگه دارین، با چهره مهربون و دوستانه بهتون نزدیک میشن ولی کم‌کم سعی می‌کنن کاری کنن که با سرویس‌های جاسوسیشون همکاری کنین...[۶۴] (مطمئنا این شیوه تهش هیچی نیست. دنبالش نباید رفت. چون از هر طرف زیر پاتون خالی میشه...)

مسیر راه امنیت وب اپلیکیشن:

برای مسیر راه، ابتدا ویدیوهای زیر از «*یاشار شاهین‌زاده*» رو ببینین و همراهش توضیحات منو بخونین:
+ «چگونه یه باگ‌هانتر یا هکر بشیم؟ نقشه راه شروع امنیت»[۶۵] که از «وبسایت خودش»[۶۶] توضیح میده.
+ کنفرانس امنیتی شبگرد (چند تا ویدیوعه. این ویدیوش درباره باگ بانتیه. بقیش هم ببینین خوبه)[۶۷]

---

۶۴ داستان دانشجویی که در شانگهای درس می‌خوند و تو تله سرویس‌های امنیتی چین افتاد. مستند ۲۸ دقیقه‌ای از FBI:
https://www.fbi.gov/video-repository/newss-game-of-pawns/view
65 https://www.youtube.com/watch?v=_UxO2qKvCEQ
66 https://securityflow.io/roadmap/
۶۷ شبگرد از اولین گروه‌های هک و امنیت ایران بود.

حالا روی چیزای یاشار، من یه سری چیزای دیگه اضافه کنم:

شبکه:

همونطور که صحبت کردیم، امنیت یعنی شما اول می‌دونین اون چیز چطور کار می‌کنه و بعدش حالا سراغ اینکه چه‌جوری امنش کنین یا از مشکلات بهره‌برداری کنین میرین. وب روی شبکه و اینترنت بنا شده. پس شما اول باید بتونین درک خوبی از شبکه به دست بیارین.

حالا چه قسمتایی از شبکه رو نیازه بدونین؟ «یاشار شاهین‌زاده»، توی یه ویدیو بهتون گفته که چه قسمتایی از کتاب «Network+» رو نیازه بخونین.[۶۸]

اگر کتاب شبکه رایگان می‌خواین که کاپی‌رایت حفظ شه، من کتاب

"TCP/IP Guide: A Comprehensive, Illustrated Internet Protocols Refrence" by "Charles M. Kozierok"

رو پیشنهاد می‌کنم که سازندش توی وبسایتش گذاشته.[۶۹] (قبلاً یاشار اینو معرفی می‌کرد) البته قدیمی‌تره ولی توضیحاتش خیلی عمیقه. (البته یه موضوع رو ممکنه چندین بار تکرار کنه. یه حالت مروره. خسته نشین :))

حواستون باشه که شبکه رو با دیدی که یاشار گفته (دید یه فردی که می‌خواد امنیت لایه وب اپلیکیشن کار کنه)، کار کنین.

شبکه گستردس. قرار نیست شما رو همش تسلط داشته باشین! برای همینم قرار نیست هر کتاب شبکه‌ای به دردتون بخوره. ممکنه بعضیاشون با دیدگاه‌های دیگه‌ای نوشته شده باشه.

برنامه‌نویسی (ابزارنویسی):

بعدش باید یه زبون برنامه‌نویسی که لایبرری‌های متعدد (لایبرری‌ها یه سری کد هستن که دیگران نوشتن که کار شما رو راحت کنن. مثلاً لایبرری کار و یافتن آدرس‌های مختلف درون یه وبسایت) برای کار با چیزای متفاوت داشته باشه رو یاد بگیرین.

اینجا یاشار، «Python» و «Go» رو پیشنهاد داده.

درواقع شما باید ابزارنویسی کنین. یعنی اینکه بتونین کارهای خسته‌کننده رو خودکار انجام بدین.

مثلاً فرض کنین شما می‌خواین صفحه ورود صاحب سایت (صفحه ادمین) رو پیدا کنین. یه راهش اینه دستی یه سری آدرس رو تست کنین:

```
admin.example.com/
example.com/admin
aria.example.com/
```[۷۰]

اما این راه خسته‌کنندس. شما وقتی روی یه وبسایت تست انجام میدین، ده‌ها تست مختلف رو باید انجام بدین. نباید وقتتون تلف شه. وگرنه نمی‌رسین که مشکل پیدا کنین و بقیه زودتر پیدا می‌کنن و زودتر

68 https://www.youtube.com/watch?v=m1jlkO6gQ_0

69 http://tcpipguide.com/free/t_toc.htm

۷۰ درواقع مثلاً من دونستم اسم صاحب سایت، «آریا» هست. پس سعی می‌کنم ساب‌دامین‌هایی که اسم آریا توشه رو چک کنم شاید شانسی به صفحه ورود بر خوردم.

گزارش میدن. برای همین باید خودکارسازی (automation) انجام بدین. یعنی با ساخت ابزار و برنامه، این کارها رو به صورت خودکار انجام بدین. برنامه بره براتون بگرده.

یا حتی فرض کنین یه دیتابیسی رو پیدا کردین:

example.com/database/1.png

شما می‌تونین با عوض‌کردن این عدد ۱، به عکسای دیگه‌ای که توی همون پوشه توی سرور هستن دسترسی داشته باشین.[۷۱] مثلاً:

example.com/database/2.png
example.com/database/3.png
example.com/database/4.png
...

فرض کنین یه پوشه شامل عکسای کارت ملی رو پیدا کردین. حالا اینطوری میاین کل عکسای پوشه رو دانلود می‌کنین! تبریک! شما کل کارت‌های ملی رو دارین.

درواقع همه این‌ها رو باید با خودکارسازی انجام بدین.

درواقع سعی کنین نحوه نوشتن یه برنامه برای حل موضوعات رو یاد بگیرین. یعنی سعی کنین که بتونین درک کنین که چطور باید برای حل مسأله‌ها مختلف، برنامه نوشت.

تمرین مهم‌ترین چیز در برنامه‌نویسیه. هرجا گفت بیا آموزش برنامه‌نویسی ۰ تا ۱۰۰ پایتون در ۱۰۰ روز، فرار کنین! آموزش بدون تمرین = هیچ

برای تمرین به قسمت «*معرفی وبسایت* Quera» مراجعه کنید.

لینوکس:

مورد بعد، یادگیری لینوکسه. چون اکثر سرورهایی که اون وبسایت باهاش بالا اومده، لینوکسی هستن. شما توی سرور که دسترسی به زبون Go ندارین که! باید از ابزارهای تحت ترمینال استفاده کنین.

بعدشم ابزارهای لینوکسی به شدت قدرت‌مند هستن. یعنی شما به جای اینکه ۱۰۰ خط کد پایتون بزنین، می‌تونین با یه خط کد داخل ترمینال لینوکس، همون کار رو انجام بدین![۷۲] بله همینقدر قدرت‌مند! درواقع ابزارهای لینوکس، یه سری ابزارهای کوچیک هستن که یه کار انجام میدن ولی همون کار رو به گونه‌های مختلفی که بخواین و به صورت خیلی خوبی انجام میدن. این براتون قابل‌درک نخواهد بود، تا وقتی که یه آموزش **خوب** (نه هر آموزشی!) از لینوکس ببینین.

برای اینکه بدونین چقدر لینوکس قدرتمنده، یه ویدیو از جادی هست، اونو ببینین.[۷۳]

همچنین برای یادگرفتن ابتدایی و درک لینوکس، *می‌تونین به آموزش‌های من مراجعه کنین! (به‌زودی در گیتهابم قرار می‌گیره)*

دنبال‌کردن آدمای خفن این حوزه:

---

۷۱ به این کار می‌گن «Crawling».
۷۲ البته هیچکدوم (پایتون و لینوکس) جای همو نمی‌گیرن!
۷۳ «چرا گنو/لینوکس رو دوست دارم: ترکیب ابزارها و کشیدن نمودار لاگ‌ها در کامندلاین»:
https://www.youtube.com/watch?v=hCkII2FEOLU

هیچکی از بدو تولد دانش نداشته. ما دانشمون رو از بقیه یاد می‌گیریم. یکی از مهم‌ترین چیزا اینه که شما نگاه بقیه کنین ببینین اونا چه کارهایی انجام میدن و خوب‌خوباشو برای خودتون بردارین و استفاده کنین. ببینین دنیا داره به چه سمتی میره؟ چیا داره روی کار میاد؟ چیا داره رایج میشه؟ یاد بگیرین یاد بگیرین و یاد بگیرین! به قول جادی: ((بیشترین چیزی که شما باید یاد بگیرین، یادگرفتنه.))

حتی اگر خفن‌ترین آدم یه حوزه‌ای هستین، بازم نیازتون به یادگرفتن از بین نمیره. گروهی از شرکتا که یه زمانی غول حوزه خودشون بودن، به خاطر اینکه توجهی به گذر زمان نکردن که چه چیزایی داره رایج میشه، ورشکسته شدن. فکر می‌کردن چون فروششون از همه بیشتره، پس همون روند درسته و باید ادامش بدن. درحالی که این جلوی چشمشون رو گرفته بود و ندیدن که دنیا داره عوض میشه و نیاز بازار داره تغییر می‌کنه. برای اینکه نتونستن خودشونو تغییر بدن، از بازار کم‌کم حذف شدن...

برای همین Twitter، Medium، Github و وبسایت‌های آدمای معروف این حوزه رو حتماً دنبال کنین. ببینین اونا چطور تونستن باگ پیدا کنن. چه ابزارهایی به کار می‌برن. چطور فکر می‌کنن و ازشون یاد بگیرین.

توی همایشا هم معمولاً افرادهای مختلفی میان و درباره کارهای جدیدی که کردن صحبت می‌کنن. این همایشا رو هم دنبال کنین:

BlackHat – DEFCON – Nahamcon – RSA Conference

رمزنگاری:

یکی از پایه‌ای‌ترین مسائل موجود در دنیای امنیت، مبحث «رمزنگاریه». شما درکی از امنیت نخواهید داشت اگر رمزنگاری رو متوجه نشین. پس حتماً درباره رمزنگاری اطلاعات داشته باشین.

برای خوندن مفاهیم مهم رمزنگاری، می‌تونین به مطلب من درباره رمزنگاری مراجعه کنین. در حد ۵ ساعت زمان می‌بره ولی نکات خیلی مهمی داره. (به زودی در گیتهاب منتشر میشه :))

لینک‌های بیشتر:

+ یاشار شاهین‌زاده (با آیدی «Voorivex» می‌تونین پیداش کنین):

Youtube[74] – Twitch[75] – Website[76] (Farsi) – Website[77] (English) – Twitter[78]

به شدت پیشنهاد میشه یاشار رو دنبال کنین.

+ محسن طهماسبی (با آیدی «moh53n» می تونین پیداش کنین)

+ The security mindset[79]

74 https://www.youtube.com/@Voorivex/
75 https://www.twitch.tv/voorivex
76 https://memoryleaks.ir/
77 https://securityflow.io/
78 https://twitter.com/voorivex
79 https://www.schneier.com/blog/archives/2008/03/the_security_mi_1.html

+ What is a hacker?[80]

+ OWASP Vulnerabilities knowledge base (*Highly recommended*)[81]

+ Web Security Academy[82] (PortSwigger – *Highly recommended*)

+ CWE (Common Weakness Enumeration) and the CWE Top 25 Explained[83] (*Highly recommended*)

+ Awesome Bug Bounty Roadmap[84] (بخون)

+ وبسایتی که لینک به جاهای خوبی داده. (مثلاً CTF های خوبی معرفی کرده)[85]

+ Security blogs (e.g. Portswigger – Intigriti – Bugcrowd)

+ InfoSec Write-ups[86]

توی ردیت و stackexchange هم می‌تونین سوالاتتون رو بپرسین[87]

چند نکته پایانیِ بسیار مهم:

مسیر امنیت، مسیری طولانی هست و پیوستگی توش به شدت اهمیت داره. یعنی اگر سه ماه خوندین، سه ماه ول کردین، اون نتیجه‌ای که باید و شاید رو نمی‌گیرین و زحماتتون هدر میره. برای همین پیوستگی مهمه. این آیندتونه! براش تلاش کنین. هر روز توی برنامتون بگنجونین. نذارین به خاطر درس و دانشگاه، مثلاً یه دفعه دو ماه بین مطالعتون فاصله بیوفته. درس و دانشگاه جای خود، مطالعه خارج دانشگاه هم جای خود. با برنامه‌ریزی درست، به هردوش میرسین! بله تلاش می‌خواد. یه مقداری از کارهای بیهوده مثل اینستاگردی و توییترگردی که تهش به سلامت روانیتونم آسیب می‌زنه کم کنین و به آیندتون بیشتر بها بدین. اینستا و توییتر به شما حس ناکافی‌بودن، عقب‌بودن و خستگی میدن. ازشون فاصله بگیرین. به جاش به تفریحات دیگه‌ای مثل ورزش، وقت‌گذروندن با دوستان، آشپزی، موسیقی، طراحی و نقاشی، کاردستی و... بپردازین.

رهرو آن نیست که گهی خسته گهی مانده رود / رهرو آن است که آهسته و پیوسته رود

آهسته ولی پیوسته برید. هیچ راه میونبری نیست که یه ماهه بخواین توی مسیر باگ‌بانتی (یا هر مسیر دیگه‌ای) موفق شین. عجله‌کردن صرفاً باعث این میشه که خشت اول رو کج بنهید که تا ثریا خونه کج میره.

- باش. پس نمیشه تو مدت کم رسید، من از هر چیز جانبی و تفریح می‌زنم و روزی ۱۴ ساعت زمان می‌ذارم که زودتر برسم.

---

80 https://www.schneier.com/blog/archives/2006/09/what_is_a_hacke.html
81 https://owasp.org/www-community/vulnerabilities/
82 https://portswigger.net/web-security
83 https://www.hackerone.com/vulnerability-management/cwe-common-weakness-enumeration-and-cwe-top-25-explained
84 https://attacker-codeninja.github.io/2022-06-06-awesome-bug-bounty-roadmap/
85 https://github.com/bittentech/Bug-Bounty-Beginner-Roadmap
86 https://infosecwriteups.com
87 i) https://security.stackexchange.com/
ii) https://www.reddit.com/r/websecurity/
iii) https://www.reddit.com/r/netsec/

+ نمیشه یه نسخه رو برای همه پیچید ولی پیشنهاد می‌کنم تعادل رو رعایت کنین. چه تفریح زیاد و چه درس زیاد، به شما ضرر می‌رسونه. درستون سر جاش، ورزش و سلامتیتون سر جاش، تفریحات دیگه هم سر جاش. تلاش کنین و پیوسته باشه. نه اینکه یه روز ده ساعت بخونین یه روز یه ساعت. اینطوری فایده‌ای نداره.

سعی کنین هدفاتونو به بخشای کوچیک‌تر تقسیم کنین. اگر بخواین باگ‌بانتی کار شین، ممکنه چندین ماه زمان ببره که تازه شروع بشه کارتون. توی این زمان خسته میشین و میگین ای بابا! نرسیدم. کی می‌تونم باگ بزنم پس؟!
ولی اگر مثلاً بگین توی یه ماه آینده می‌خوام شبکه تموم کنم اون قسمتی که می‌خوام. فلان چیز تموم کنم و بعد از یه منبع (مثل کتاب) جلو برین و پیشرفتتون رو ببینین که ایول این فصل رو خوندم؛ ایول فلان چیز رو خوندم، احساس می‌کنین که زحماتتون نتیجه داده و با اعتماد به نفس و انگیزه مسیر رو ادامه میدین.

اوایل چون تازه وارد شدین، به خاطر نداشتن تجربه و دانش، شاید سخت باشه براتون. ممکنه یه خرده ناامید شین. برای اینکه انگیزه داشته باشین، می‌تونین سراغ برنامه‌هایی برین که رقابت توش کمتره. مثلاً پلتفرم‌های ایرانی. یا برنامه‌هایی که زمان خاص داره. یعنی مثلاً میگن از فلان تاریخ تا فلان تاریخ بیشتر وقت نداره. احتمال اینکه یه سری آدما نباشن اون زمان، هست و خب رقابت یکم کمتر ممکنه باشه.

- بابا من خیلی عقبم. خسته میشم. حوصله ندارم شروع کنم. همه از من جلوترن!
+ خیر اینطور نیست! علم کامپیوتر روز به روز داره تغییر می‌کنه. اکثر دانش افراد، دانشی هست که در ۳ – ۴ سال اخیر یاد گرفتن. شما هم اگر واقعاً می‌خوای نتیجه بگیری، نیازه به زندگیت نظم بدی. قرار نیست بگیم همش می‌خوام بازی کنم یا تو اینستا بچرخم، از اون طرفم می‌خوام علمم زیاد شه. نمیشه! تفریح جای خودش، علم جای خودش، ورزش و خورد و خوراک و... هم جای خودش. همش با نظم به دست میاد. بله اون سختی باعث میشه یه مدت نشه مثلاً مثل گذشته ده ساعت توی اینترنت چرخید، شاید این از نظرتون خسته‌کننده باشه ولی باید بگین کدوم به درد آیندم می‌خوره؟ من باید یه زندگی خودم رو بچرخونم و تصمیمتون رو بگیرین...
ولی هدفتون بزرگ‌تره و در نهایت می‌تونین به شغلی که می‌خواین برسین. مطمئناً توی اون مسیر سختی هست. خستگی هست. موفق‌نشدن هست، ولی باید پیوستگی داشته باشین و به راهتون اطمینان داشته باشین. از تجربیات دیگران بهره بگیرین و خسته نشین. ادامه بدین که در نهایت لذتی که می‌خواستین قبلاً ببرین رو بعد داشتن شغل مورد علاقه می‌برین.[۸۸]

- آیا ورود به رشته کامپیوتر، کمک می‌کنه؟
+ بله! برای حوزه امنیت، شما باید اول دانش کامپیوتر داشته باشین و بعد دنبال امنیتش باشین. درس‌هایی مثل کامپایلر، برنامه‌نویسی، معماری کامپیوتر و... پایه خیلی از چیزای امنیت هستن.

---

۸۸ میگن کاری کن که نخوای کار کنی. (اینقدر اون کار لذت بخشه که مثل تفریحه)

اما لزوماً اینطور نیست که فقط و فقط باید رشته کامپیوتر باشی تا بتونی بری توی حوزه امنیت. خودتونم می‌تونین بخونین. وبسایت‌های ایرانی مثل «مکتب‌خونه»[۸۹]، «فرادرس»[۹۰] و وبسایت‌های خارجی مثل «coursera»[۹۱] و «Udemy»[۹۲] کورس‌های دانشگاه‌های معتبر رو دارن که می‌تونین ببینین. یا حتی خود دانشگاه‌هایی مثل استنفورد و اینا توی سایتشون هم کورس‌های رایگانی دارن.

Division by Zero

به طور کلی خود Division by zero، یه باگ امنیتی نیست ولی ممکنه باعث چیزایی بشه که عملاً مشکل امنیتی ایجاد کنن.[۹۳]

89 https://maktabkhooneh.org/
90 https://faradars.org/
91 https://coursera.org/
92 https://www.udemy.com/
93 https://security.stackexchange.com/questions/204669/is-divide-by-zero-a-security-vulnerability