دفترچه راهنمای پیکار

مقدمه

خوش اومدید به پیکار استعدادیابی **Arrival**. پیکار از **۱۶ مهر ساعت ۲۰:۰۰** رسمی شروع می‌شه و تا **۱۹ مهر ساعت ۲۰:۰۰** فرصت شرکت در مسابقه به‌صورت رسمی بازه. پیکار برای تمرین و مرور تا زمان اعلام نتایج قابل دسترسی خواهد بود و نتیجهٔ نهایی تا **۲۳ مهر** اعلام می‌شه. نفراتی که به‌عنوان برنده انتخاب بشن باهاشون تماس گرفته می‌شه و در صورت ثبت‌نام در کلاس **OWASP Zero X** هزینه ثبت‌نام‌شون برگشت داده می‌شه.

این پیکار مناسب کسانی که تازه وارد دنیای امنیت وب شدن *(یا حتی توی امنیت صفرن)* ولی انگیزه یادگیری دارن. ما انتظار نداریم از روز اول همه چیز رو بدونید؛ پیشنهاد ما اینه که حداقل ۳ روز روی مسائل وقت بذارید: سرچ کنید، گیر کنید، تلاش کنید و دوباره تلاش کنید. این روند خود یادگیریه. هدف اصلی پیکار فراتر از جایزه‌ست: «گسترش علم و پرورش طرز فکر هکری». هکرهای حرفه‌ای اغلب از الگوهای ساده استفاده می‌کنن؛ چیزی که مهمه، روش فکر کردن و کنجکاوی شماست. **تیم هانت Voorivex** با همین تکنیک‌های ساده تا امروز آسیب‌پذیری‌های زیادی یافته و بانتی‌های قابل‌توجهی دریافت کرده. ما می‌خوایم این تجربه رو با شما تقسیم کنیم.

چه چیزهایی لازم داری و چه انتظاری داشته باشی

- داشتن پایه‌ای از وب *(HTML، HTTP، فرم‌ها، کوکی‌ها، هدرها)* و کمی آشنایی با مباحث پایه و ابتدایی رمزنگاری *(Encoding، Hashing، Encryption)* بهت خیلی کمک می‌کنه.
- برای آماده‌سازی سریع، پیشنهاد می‌کنیم دوره امنیت اپلیکیشن رو ببینی *(اگر با سرعت ۲x ببینی، حدود ۶ ساعت زمان می‌بره و پایهٔ لازم رو می‌ده)*.
- توی دفترچه برخی اصطلاح‌ها استفاده شده؛ برای درک بهتر اون‌ها تماشای ویدیوها و تمرین عملی مفیده.
- برای حل کردن بعضی مراحل ممکنه نیاز به مطالعه بیشتر باشه؛ این طبیعیه و بخشی از فرایند یادگیریه.
- با طرز فکر *«کاوشگر و کنجکاو»* وارد شو؛ قبل از آزمون و خطای کور، الگوها و نشانه‌ها رو تحلیل کن.

نکته خیلی مهم!

خیلی از مراحل رو می‌شه با استفاده از هوش مصنوعی و ChatGPT حل کرد. ولی بهترین حالتش اینه که اول خودتون روی مسئله فکر کنید و سعی کنید با خلاقیت و سرچ کردن به راه حل برسید که مسیر یادگیری هم براتون طی بشه. بهتره که از هوش مصنوعی برای یادگیری یا تجزیه و تحلیل مسئله استفاده کنید نه گرفتن جواب نهایی و رد شدن از اون مرحله، چون هدف اصلی این پیکار یاد دادن «طرز فکر» و «تحلیل مساله» است و اگه شما خودتون روی مراحل فکر کنین، دانش ارزشمندی بدست میارین که بنظرم ارزشش رو داره.

مرحله ۱

وقتی از یک وب‌سایت استفاده می‌کنی، اولین چیزی که مرورگر بهش تکیه می‌کنه همون URL یا نشانی اینترنتیه. URL در واقع مثل آدرس خونه‌ست. همونطور که آدرس یک خونه می‌تونه شامل شهر، خیابون، کوچه و پلاک باشه، آدرس یک صفحهٔ وب هم شامل بخش‌های مختلفی می‌شه: پروتکل *(http یا https)*، دامنه *(مثل example.com)* و مسیر *(مثل /level-01/)*. مرورگر این آدرس رو می‌خونه و از سرور می‌خواد محتوای مربوط بهش رو برگردونه.

حالا تصور کن وارد یک کوچه شدی که همه پلاک‌ها پشت سر هم هستن. اگه جلوی پلاک ۱۰ وایسادی، خیلی راحت می‌تونی حدس بزنی پلاک ۱۱ هم وجود داره. توی وب هم همینطوره. بعضی وقت‌ها مسیرها بر اساس یک الگوی مشخص ساخته شدن و تغییر دادن اون‌ها می‌تونه تو رو به بخش جدیدی ببره. البته همیشه اینطور نیست؛ ممکنه بعضی مسیرها اصلا وجود نداشته باشن یا سرور عمدا اجازه دسترسی بهشون رو نده و در عوض *خطای ۴۰۴* نشونت بده. نکته اصلی اینجاست که URL فقط یک نوشته بی‌معنی بالای مرورگر نیست. پشت هر تغییر کوچک در اون یک درخواست واقعی به سرور انجام می‌شه. پس وقتی دنبال سرنخ هستی، دقت به الگوی آدرس‌ها می‌تونه کمکت کنه.

مرحله ۲

وقتی یک صفحه وب باز می‌کنی، مرورگر فایلی به اسم HTML دریافت می‌کنه که ساختار و محتوای صفحه رو مشخص می‌کنه. مرورگر این کدها رو تفسیر می‌کنه و چیزی که تو می‌بینی روی صفحه به نمایش درمیاد. اما همه چیزی که در فایل HTML وجود داره لزوما به چشم کاربر نمیاد. بخشی از این کدها می‌تونه کامنت باشه. کامنت‌ها با *<!-- ... -->* مشخص می‌شن و معمولا برای توضیح دادن یا یادداشت گذاشتن در کد استفاده می‌شن. کاربر عادی اونا رو نمی‌بینه چون روی ظاهر صفحه تاثیری ندارن. اما اگه سورس کد صفحه رو باز کنی، می‌تونی این نوشته‌ها رو پیدا کنی. خیلی وقت‌ها توسعه‌دهنده‌ها یا حتی طراحان چالش از همین فضا برای گذاشتن پیام‌های مخفی استفاده می‌کنن.

دقیقا شبیه کتابیه که لابه‌لای سطرهاش با مداد یادداشت گذاشته شده باشه. متن اصلی کتاب رو همه می‌بینن، اما کسی که دقیق‌تر نگاه کنه می‌تونه نوشته‌های پنهانی رو هم بخونه. برای دیدن این قسمت‌ها کافیه روی صفحه راست‌کلیک کنی و گزینه *"View Page Source"* یا *"Inspect"* رو انتخاب کنی.

مرحله ۳

وب‌سرورها فایل‌ها و پوشه‌های زیادی رو ذخیره می‌کنن. وقتی یک مسیر رو در مرورگر باز می‌کنی، سرور معمولاً یک فایل خاص مثل *index.html* رو نشون می‌ده تا صفحه برایت آماده باشه. اما گاهی اوقات اگر این فایل وجود نداشته باشه، سرور به‌جای آن کل محتویات پوشه رو به صورت یک لیست ساده نمایش می‌ده. این حالت به اسم Directory Listing شناخته می‌شه. تصور کن وارد یک کتابخونه شدی. به جای اینکه فقط کتابی رو که درخواست کردی ببینی، قفسه باز می‌شه و همه کتاب‌های موجود رو جلوی چشمت می‌ذاره. همین اتفاق هم در وب ممکنه بیفته. اگر پوشه‌ای روی سرور برای نمایش عمومی باز گذاشته شده باشه، همه فایل‌هاش رو می‌تونی ببینی. این موضوع در امنیت اهمیت زیادی داره. چون گاهی فایل‌هایی که نباید دیده بشن *(مثل backup.zip یا next.txt)* هم داخل همین لیست‌ها ظاهر می‌شن. کسی که دقیق باشه می‌تونه از این فرصت استفاده کنه و به اطلاعات بیشتری دست پیدا کنه. پس یادت باشه همیشه فقط به فایلی که نشون داده می‌شه اکتفا نکنی. مسیرهای بالاتر رو هم نگاه کن، شاید چیزهایی پیدا کنی که هیچ لینکی به اون‌ها وجود نداره.

مرحله ۴

کوکی‌ها مثل یادداشت‌های کوچیکی هستن که مرورگر برای هر سایت نگه می‌داره. این یادداشت‌ها می‌تونن خیلی ساده باشن، مثلا زبونی که انتخاب کردی یا این‌که لاگین هستی یا نه. اما بعضی وقت‌ها همین کوکی‌های ظاهرا بی‌اهمیت، مسیر حرکتت رو عوض می‌کنن. فرض کن وارد یک فضایی شدی و روی دستبندت نوشته باشه *"Guest"*. با همین دستبند اجازه داری وارد سالن عمومی بشی. حالا اگه کسی دستبندت رو عوض کنه و روی اون نوشته باشه *"VIP"*، مسیرهای جدیدی برات باز می‌شن. کوکی‌ها هم دقیقا همین‌طور عمل می‌کنن. تغییر دادن مقدار یک کوکی می‌تونه باعث بشه سرور رفتار متفاوتی نشون بده. اینکه بتونی کوکی‌ها رو ببینی و بفهمی چه نقشی دارن خیلی مهمه. برای همین لازم می‌شه از ابزارهای مرورگر استفاده کنی و مقادیر رو بررسی کنی. شاید چیزی که جلوی راهتو گرفته فقط یک مقدار ساده باشه.

مرحله ۵

هر وقت مرورگر به سرور درخواست می‌ده، جواب فقط شامل متن صفحه یا تصویر نیست. سرور همراه اون داده‌ها، یک سری اطلاعات اضافه به اسم هدر هم می‌فرسته. این هدرها برای مرورگر و برنامه‌ها مهم هستن چون توضیح می‌دن داده‌ای که اومده چه نوعیه، چقدر طول داره، یا باید چطور ذخیره بشه. اما ماجرا اینجا تموم نمی‌شه. هدرها می‌تونن شامل پیام‌هایی باشن که کاربر معمولی هیچ وقت نمی‌بینه. این مثل اینه که علاوه بر نامه‌ای که توی پاکت هست، روی خود پاکت هم چند خط نوشته باشه. نامه رو همه می‌خونن، ولی فقط کسی که به پاکت توجه کنه متوجه اون نوشته‌ها می‌شه.
گاهی برنامه‌نویس‌ها از همین فضا برای گذاشتن سرنخ استفاده می‌کنن. چون هدرها جایی هستن که کمتر کسی نگاهش می‌کنه. پس وقتی دنبال نشونه می‌گردی، حتما هدرها رو هم بررسی کن.

مرحله ۶

وقتی با وب‌سایت‌ها کار می‌کنیم، معمولا به متدهای عادی مثل *GET* یا *POST* عادت داریم. مرورگرها هم بیشتر وقت‌ها همین دو نوع درخواست رو می‌فرستن. اما پروتکل *HTTP* متدهای بیشتری داره: *PUT*، *OPTIONS*، *DELETE* و چندتای دیگه. هرکدومشون کاربرد خاص خودشون رو دارن و ممکنه سرور در برابرشون واکنش متفاوتی نشون بده.
اینو می‌شه مثل زدن در خونه با شکل‌های مختلف تصور کرد. گاهی زنگ رو فشار می‌دی، گاهی در رو می‌کوبی، و گاهی با کلید امتحان می‌کنی. هر کدوم می‌تونه نتیجه متفاوتی داشته باشه. برای همین محدود شدن به فقط یک روش باعث می‌شه بعضی مسیرها رو هیچ وقت نبینی.
برای درک درست این مرحله، به این فکر کن که اگه همین درخواست رو با متد دیگه بفرستی چه می‌شه. همیشه یک راه استاندارد وجود داره، اما راه‌های دیگه هم هست که ارزش امتحان کردن دارن.

مرحله ۷

گاهی در صفحات وب چیزهایی هستن که چشم کاربر نمی‌بیند، اما مرورگر آن‌ها را جدی می‌گیرد. یکی از این موارد *«فیلدهای پنهان»* در فرم‌هاست. این فیلدها مقداری را در خود دارند که روی صفحه دیده نمی‌شود، اما وقتی داده‌ها فرستاده شوند، همراه بقیه به سرور می‌روند.
در این مرحله با فرمی روبه‌رو می‌شوی که در ظاهر هیچ کار خاصی نمی‌کند. هیچ دکمه‌ای برای کلیک نیست و هیچ مقصد مشخصی برای ارسال داده‌ها تعیین نشده. با این حال یک مقدار داخل آن مخفی شده که منتظر است به سرور برسد. اینجاست که باید به این فکر کنی: «وقتی چیزی به‌طور معمولی قابل ارسال نیست، چه راه‌های دیگری برای فرستادنش وجود دارد؟» دانستن اینکه فرم‌ها چطور در وب کار می‌کنند، و اینکه مرورگر یا ابزارهای دیگر چگونه می‌توانند داده‌ها را ارسال کنند، کلید پیشروی است.

مرحله ۸

اگه به رشته‌هایی پر از حروف بزرگ و کوچک، عدد و علامت مساوی برخوردی که هیچ معنی مشخصی ندارن، خیلی وقت‌ها پای Base64 وسطه. این روش از قدیم برای تبدیل داده‌های خام به متن استفاده شده تا راحت‌تر منتقل بشن. برای مثال وقتی یک تصویر یا یک فایل باینری باید داخل متن ایمیل جا بشه، معمولا از *Base64* استفاده می‌کنن.

وقتی رشته‌ای رو می‌بینی که به نظر عجیب میاد، می‌تونی حدس بزنی شاید با یکی از همین روش‌های ساده انکود شده باشه. با برگردوندنش *(decode)* ممکنه متن معمولی یا سرنخ مورد نیاز ظاهر بشه. نکته مهم اینه که *Base64* رمز نیست؛ فقط شکلی از نمایش داده‌هاست. یعنی هرکسی می‌تونه راحت اون رو به متن اصلی برگردونه. شناختن الگوی *Base64* باعث می‌شه سریع‌تر به اصل پیام برسی. پس همیشه یک نگاه دقیق به متن داشته باش تا بفهمی آیا پشت اون ترکیب عجیب یک متن ساده پنهان شده یا نه.

مرحله ۹

یکی از ساده‌ترین روش‌های احراز هویت در وب چیزی به اسم Basic Authentication هست. معمولا وقتی سایتی از این روش استفاده می‌کنه، مرورگر یک پنجره کوچک باز می‌کنه و ازت نام کاربری و رمز عبور می‌خواد. اطلاعاتی که وارد می‌کنی به یک فرمت مشخص تبدیل می‌شن و به صورت یک هدر همراه درخواست به سرور ارسال می‌شن.

اما باید بدونی که این روش همیشه به شکل پاپ‌آپ دیده نمی‌شه. در بعضی مواقع، سرور هیچ پنجره‌ای بهت نشون نمی‌ده و فقط انتظار داره هدر مخصوص *Authorization* همراه درخواست باشه. اگر این هدر فرستاده بشه، رفتار صفحه تغییر می‌کنه؛ اگر هم نباشه، شاید با خطا یا دسترسی محدود روبه‌رو بشی.

اینجا مهم اینه که درک کنی *Basic Auth* چیزی فراتر از همون پنجره ورود ساده‌ست. اصل ماجرا در پشت پرده و در هدرهای درخواست اتفاق می‌افته. شناخت ساختار این هدرها و اینکه چطور مرورگر یا ابزارهای دیگه می‌تونن اون رو اضافه کنن، قدم بعدی برای یاد گرفتن این مفهومه.

مرحله ۱۰

گاهی رشته‌ای می‌بینی که فقط از *ارقام ۰ تا ۹* و *حروف a تا f* تشکیل شده و شبیه یک کد به‌نظر می‌رسه. یکی از روش‌های ساده نمایش داده‌ها، نمایش بایت‌ها به صورت هگزادسیمال؛ هر جفت کاراکتر هگز نمایانگر یک بایت است. این یعنی ممکنه متن قابل‌خواندن یا داده باینری‌ای پشت اون رشته هگز باشه؛ چیزی که با تبدیل مناسب دوباره به شکل اصلی برمی‌گرده.

مهمه این را بفهمی که هگز صرفا یک شیوه نمایش است، نه یک روش رمزگذاری امن یا هش. وقتی با چنین رشته‌ای روبه‌رو شدی، قبل از اینکه نتیجه‌گیری کنی «این یک هش است»، از خودت بپرس آیا این می‌تونه نمایش ساده یک فایل یا متن باشد که فقط به شکل هگز نوشته شده؟

مرحله ۱۱

گاهی با رشته‌ای برخورد می‌کنی که ظاهرا شبیه هگز هست؛ اما این بار ممکنه قابل برگرداندن نباشه و خروجی یک تابع هش باشه. توی هش‌کردن، تابعی ورودی *(مثلاً یک کلمه یا فایل)* رو می‌گیره و یک مقدار با طول ثابت تولید می‌کنه؛ خصوصیت مهم اینه که *تابع یک‌طرفه* است. یعنی از روی خروجی معمولا نمی‌تونی ورودی را بازسازی کنی. MD5 یکی از این توابعه که خروجی‌ای ۳۲ کاراکتری می‌سازه.
با این حال در عمل، بسیاری از مقادیر هش‌شده مربوط به ورودی‌های ساده یا متداول هستند؛ به همین خاطر ابزارهایی ساخته شده که هش‌های رایج رو با مقدار اصلی‌شون نگه می‌دارن که با استفاده از اونا ممکنه بتونیم مقدار اصلی رو پیدا کنیم. بنابراین وقتی با یک رشته ۳۲ کاراکتری مواجه شدی، مهمه تفاوت بین *«رشته هگز که قابل دیکد است»* و *«هش که معمولا برگشت‌پذیر نیست»* رو متوجه بشی.

مرحله ۱۲

وقتی یک برنامه در سرور خطا می‌ده، بعضی وقت‌ها به جای فقط یک پیام ساده، گزارشی دقیق‌تر به اسم Stack Trace نمایش داده می‌شه. این گزارش نشون می‌ده خطا کجا رخ داده و چه کدی اجرا شده. برای توسعه‌دهنده‌ها این اطلاعات ارزشمنده، ولی برای کسی که دنبال سرنخه هم می‌تونه اطلاعات زیادی داشته باشه. مثلا ممکنه مسیر فایل‌های داخلی سرور لو بره، یا حتی داده‌هایی که نباید دیده بشن همراه خطا چاپ بشن. درست مثل اینه که پرده پشت صحنهٔ یک تئاتر کنار بره و تو همه‌چیز رو ببینی.
بعضی وقت‌ها می‌تونی با ایجاد خطاهای کوچک و کنترل‌شده *(مثلا وارد کردن اطلاعات غیرمنتظره در آدرس یا فرم‌ها، یا تبدیل رشته به آرایه)* می‌تونی باعث ایجاد خطا توی سرور بشی. البته همیشه هم جواب نمی‌گیری، چون خیلی از سرورها جلوی این اتفاق رو می‌گیرن.

مرحله ۱۳

خیلی وقت‌ها توسعه‌دهنده‌ها برای تست یا حتی اشتباه، فایل دیتابیس رو روی سرور می‌ذارن و کسی فراموش می‌کنه دسترسی به اون رو محدود کنه. یکی از دیتابیس‌های سبک و پرکاربرد SQLite هست که همه‌چیزش داخل یک فایل ذخیره می‌شه. اگر این فایل قابل دانلود باشه، می‌تونی با باز کردنش تمام جداول و داده‌ها رو ببینی.
این دقیقا مثل اینه که دفترچه یادداشت شخصی کسی روی میز جا بمونه و هرکس بتونه ورق بزنه. داخل این فایل‌ها ممکنه اطلاعات کاربرها، مسیرها یا اطلاعات حساس دیگه ذخیره شده باشه.
برای بررسی چنین فایل‌هایی ابزارهای ساده‌ای وجود داره مثل *DB Browser for SQLite* که با اونا می‌تونی جدول‌ها رو باز کنی، رکوردها رو بخونی و دنبال چیزهایی بگردی که به چالش مربوط می‌شن.

مرحله ۱۴

توی مرورگر، جاوااسکریپت مسئول بخش بزرگی از معماری و تعاملات یک صفحه وبه. گاهی اوقات یک خطای ساده مثل نبودن یک متغیر یا کلید در localStorage می‌تونه باعث بشه بخشی از کد اجرا نشه.
اینجا باید به مفهوم دیباگ دقت کنی. خطاها معمولا در *Console* مرورگر نشون داده می‌شن. مثل اینه که یک ماشین روشن نشه و چراغ خطای موتور جلوی چشمت چشمک بزنه. نشونه وجود داره، فقط باید نگاه کنی. اگر بفهمی چه چیزی کم یا اشتباهه و اون رو درست کنی، جریان برنامه دوباره کامل اجرا می‌شه.

مرحله ۱۵

کدهای جاوااسکریپت گاهی متغیرهایی رو ایجاد می‌کنن و بلافاصله هم از بین می‌برن. این یعنی داده مهمی ممکنه فقط برای یک لحظه وجود داشته باشه. دیدن چنین چیزی با نگاه کردن به سورس فایل کافی نیست، چون همه‌چیز به سرعت در حال تغییر و حذف شدنه.
اینجاست که ابزارهای دیباگ به کار میان. مرورگرها امکان گذاشتن *Breakpoint* دارن، یعنی می‌تونی اجرای کد رو در یک خط خاص متوقف کنی و ببینی چه متغیرهایی چه مقداری دارن. درست مثل اینه که از یک فیلم سریع اسکرین‌شات بگیری تا لحظه‌ای رو ببینی که در حالت عادی دیده نمی‌شه.
با یاد گرفتن این مهارت می‌فهمی که بعضی اطلاعات فقط در زمان اجرا وجود دارن. دیدن اون‌ها نیازمند دخالت در روند اجراست، نه صرفا نگاه به فایل‌های استاتیک.

مرحله ۱۶

راجع به این مرحله توضیحی نمی‌دم که همیشه یادتون بمونه: برای هک باید چشای تیزبینی داشته باشین و به جزئیات دقت کنین.

مرحله ۱۷

زبان‌های برنامه‌نویسی معمولا پر از کلمات و دستورهای قابل خوندن هستن. اما زبان‌هایی هم هستند که در آن‌ها همه دستورات فقط با کاراکترهای *«فضای خالی»*، *«تب»* و *«خط جدید»* نوشته می‌شن. برای چشم انسان تقریبا مثل یک فایل خالیه، ولی در واقع پشت هر فضای خالی یک دستور پنهان شده. این زبان بیشتر برای سرگرمی و چالش ساخته شده، اما بهت نشون می‌ده که همیشه نباید دنبال چیزهای آشکار باشی. حتی نبودن کاراکترهای قابل دیدن هم می‌تونه معنا داشته باشه.

مرحله ۱۸

گاهی فایل دیتابیس روی سرور پیدا می‌شه، اما اطلاعات داخلش پراکنده شده. بعضی رکوردها مفیدن و بعضی‌ها هم فقط نویز هستن و باید نادیده گرفته بشن. مثل پیدا کردن تکه‌های یک پازل در میان انبوهی از کاغذهای بی‌ربطه. باید تکه‌های واقعی رو پیدا کنی، اون‌ها رو بر اساس مرتب کنار همدیگه بچینی و متن نهایی رو به دست بیاری. با کمی SQL ساده می‌شه رکوردهای معتبر رو جدا کرد و کنار هم گذاشت.

مرحله ۱۹

بعضی وقت‌ها به رشته‌ای برمی‌خوری که شبیه متن‌های رمزگذاری‌شده‌ست، اما کامل نیست. انگار چند تا حرف یا کاراکترش پاک شده یا جایگزین شده. این رشته‌ها در نگاه اول بی‌معنی به‌نظر می‌رسن، ولی ساختار کلی‌شون نشون می‌ده که از یه روش مشخص ساخته شدن. اینجور وقت‌ها باید به شکل کلی داده نگاه کنی: طولش، کاراکترهایی که داخلشه، و اینکه چه فرمتی می‌تونه داشته باشه. بعدش باید به این فکر کنی که «چطوری می‌تونم مطمئن بشم حدسی که می‌زنم درسته؟» برای این کار معمولا یه شاخص بیرونی وجود داره به اسم *Checksum* که نشون می‌ده کدوم جواب واقعا درسته.
پس ماجرا فقط حدس‌زدن تصادفی نیست، بلکه پیدا کردن قطعه‌های گم‌شده با توجه به الگو و بعد مقایسه با نشونه‌ی اصلیه. مثل وقتی که یه پازل ناقص داری و عکس روی جعبه بهت کمک می‌کنه بفهمی جای خالی باید چی باشه.

مرحله ۲۰

گاهی چیزی درست جلوی چشم قرار داره، اما چون عجولانه نگاه می‌کنیم، به‌سادگی نادیده گرفته می‌شه. چشم ما عادت داره روی بخش‌های اصلی تمرکز کنه و ریزه‌کاری‌ها رو پس بزنه.
به یک اثر هنری قدیمی فکر کن که بارها دیده شده؛ بیشتر آدم‌ها فقط سوژه‌ی اصلی رو به خاطر می‌آرن، اما کسی که با حوصله نگاه کنه، نشونه‌های کوچیک‌تری هم می‌بینه. الگویی که تا قبل از دقت کردن به چشم نمی‌اومد.

مرحله ۲۱ تا ۳۰

از مرحله ۲۱ به بعد هیچ راهنمایی وجود نداره و باید با سرچ و مطالعه یا دانش قبلی پیکارها رو حل کنین. با حل کردن مرحله ۳۰ پیکار استعدادیابی کامل میشه و فلگ نهایی پیکار رو دریافت می‌کنید. نفر اولی که مرحله ۳۰ رو حل کنه و فلگ رو داخل سایت آکادمی ثبت کنه، یک اچیومنت مخصوص *First Blood* برای این پیکار دریافت می‌کند.
شما با انجام دستورالعملی که توی همون مرحله می‌گیرین می‌تونین توی قرعه‌کشی شرکت کنین. یادتون باشه پیغام‌های شما رو *Bot* استخراج می‌کنه، پس دقیقا طبق اون چیزی که گفته شده عمل کنین. مثلا یه سری افراد سری پیش *Hashtag* درست رو نزدن و در قرعه‌کشی حتی وارد نشدن. پس موقع ارسال نتیجه نهایی به دستورالعمل بیشتر دقت کنید.
دقت کنید که کل اطلاعاتی که گفته شده رو فقط داخل یک پیام برای ربات ارسال کنید تا موقع استخراج به مشکل نخورید و داخل قرعه‌کشی شرکت داده بشید.

مرحله Bonus

در آخر یک مرحله اضافه هم به صورت *Bonus* در نظر گرفته شده که با حل کردنش یک فلگ جدا دریافت میکنید که با اون شانس شما در قرعه‌کشی ۳ برابر می‌شه و داخل سایت آکادمی یک اچیومنت برای حل مرحله بونس دریافت می‌کنید.