

# Web Application Penetration Testing



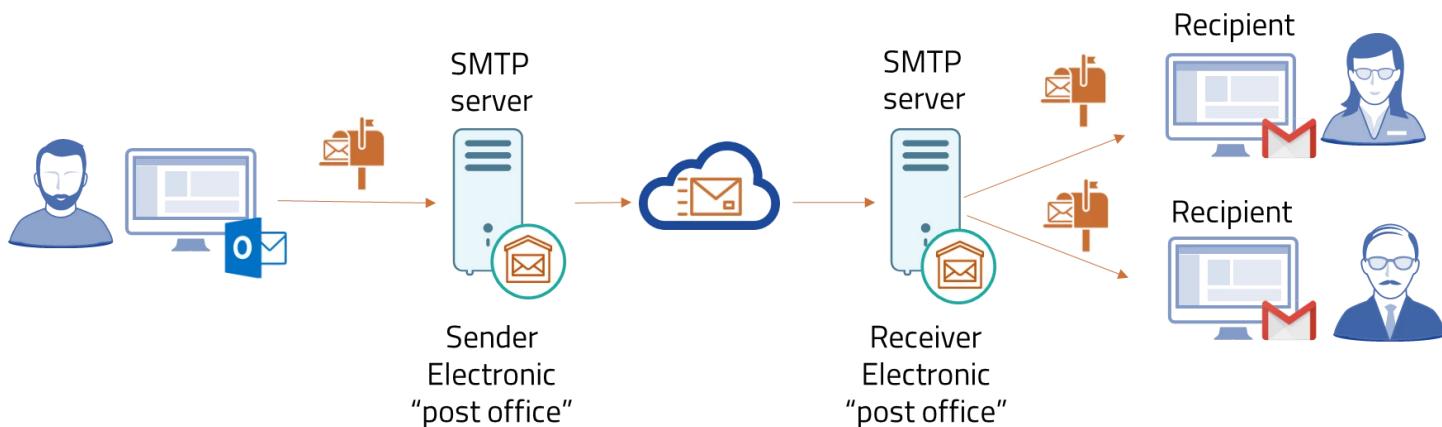
Seventh Note

By TheSecDude

قبل از اینکه برمی سروقت Email Spoofing و توضیحاتش، میخوام یه نگاهی به ایمیل پروتکل ها بندازیم، در جزو های قبلي در حین توضیح SMTP Injection کمی درموردش صحبت کردیم و من برخی از مطالب از گفته هامون توی اون جزو خواهی گرفت . چرا باید درمورد مکانیزم ایمیل پروتکل ها صحبت کنیم ؟ چون مشکل از ایمیل پروتکل هاست که به مهاجم اجازه میده Email Spoofing انجام بد . از معروف ترین پروتکلهای ارسال ایمیل و شایدم تنها پروتکل موجود، SMTP هست که روی پورت 25 کار میکنه و البته ناگفته نمونه که که حالت Encrypted SMTP پروتکل هست روی پورت 587 راه اندازی میشود . این پروتکل قبلا درموردش صحبت کردیم و گفتیم که کارش انتقال موجودیتی به نام ایمیل، از نقطه A به نقطه B است . پروتکل SMTP در لایه اپلیکیشن کار میکنه و پروتکل لایه Transport مورد استفاده این پروتکل، TCP است . پس خیلی ساده چیزی که گفتیم کاری هست که SMTP انجام میده .

یه اصطلاح دیگه که به نظرم بررسی کردنش خوبه، Mail Server هست . این اصطلاح اشاره به سروری داره که کارش پردازش و سرو کردن ایمیل هاست و در صورت وجود Mail Server، کلاینت ها با دامنه های متفاوت میتوانند به هم ایمیل ارسال کنند، مثلما میتوانیم از Yahoo.com به Gmail.com ارسال و دریافت ایمیل داشته باشیم .

تصویر زیر، نحوه عملکرد ایمیل پروتکل ها برای ارسال یک ایمیل و دریافت ایمیل رو نشون میده . میبینید که ایمیل های ما دارند از اپلیکیشن و قاعده از دامنه های متقابله ارسال و دریافت میشوند . ایمیلش رو داره از طریق سرویس Outlook ارسال میکنه به Receiver هایی که سرویشون Gmail هست . به همین خاطر هست که از یک SMTP Server به یک Receiver دیگه داره ارسال میشه .



سوالی که پیش میاد درمورد احراز هویت ارسال کننده یک ایمیل هست . چطوری SMTP Server دریافت کننده میتوانه تشخیص بده که یک ایمیل، واقعا از طرف یک ارسال کننده واقعی ارسال شده ؟ پاسخ این سوال SPF هست . SPF یا Sender Privacy Framework چیزی است که دریافت کننده ازش استفاده میکنه تا تشخیص بده ایا فرستنده ایمیل همونیه که میگه یا خیر ؟ به صورت پیش فرض، توی پروتکل های ایمیل، فرایند احراز هویت تعیینه شده و فقط از طریق همینه مولفه SPF سعی میکن کاری رو انجام بد . این مولفه SPF چیه ؟ برمی سروقت مقاله Cloud Flare درمورد مولفه SPF به ادرس زیر :

<https://www.cloudflare.com/learning/dns/dns-records/dns-spf-record/>

بینید هر ایمیل که ارسال میشه، یک ادرس دامنه داره . مثلا ایمیل های gmail.com، yahoo.com و ... خواهد داشت . SPF رکورد، یک مولفه DNS TXT هست که زمانی که شما DNS Record های یک دامنه رو میگیرید، توی رکورد های TXT ذکر میشه و مشخص میکنه که یک ایمیل از طریق این دامنه، مجاز هست که از چه سرور های ارسال بشه . خیلی مفهوم جالیبه، خودم هم زیاد نمیدونستم ولی به خورده برمی توش بیشتر بفهمیم . اگه یادتون باشه درمورد DNS خیلی صحبت کردیم و نحوه عملکردش رو توی جزو ه دوم سوم کامل بررسی کردیم، گفتیم که هر دامنه DNS Record مختلفی داره که انواعی دارن و کاربردشون هم صحبت کردیم درموردش . DNS TXT Record ها اجازه میدن که مدیریت یک دامنه توی DNS اون دامنه، متن های دلخواه خودش رو بنویسه . این رکورد در ابتدا برای درج اطلاعات مهم درباب دامنه ایجاد شد ولی امروزه کاربرد های دیگه ای هم داره . یکی از کاربرد ها مشخص کردن SPF هست .

علت بوجود امدن SPF Record به خاطر این بود که پروتکل استانداردی که برای ایمیل ها استفاده میشد (SMTP) به صورت ذاتی مولفه "from" ایمیل ها رو احراز هویت نمیکرد . عدم وجود SPF رکورد هست که موجب میشه یک مهاجم بتونه جعل هویت کنه ارسال کننده یک ایمیل رو و دریافت کننده رو قانع کنه که یک کاری رو انجام بده .

مثالی که CloudFlare درمورد SPF رکورد میزنه اینه که، SPF رکورد مثل یک Guest List هست که توسط مهمند اداره (دریافت کننده ایمیل) یک محل مدیریت میشه . هر کسی که توی لیست باشه اجازه ورود داره و هر کسی که نباشه امکان نداره که بتونه وارد بشه؛ همچنین اگه SPF رکورد وجود نداشته باشه، یعنی طبق مثالمنون یک دامنه Guest List نداشته باشه، دریافت کننده اون ایمیل رو دریافت نمیکنه و یا اگه دریافت میکنه اون رو به عنوان Spam نشانه گذاری خواهد کرد . پس دقت کنید، اگه شما یک سرویس ایمیل دارید و کاربراتون ایمیل هاتون رو دریافت نمیکن و یا اگه دریافت میکن توی Spam قرار میگیره، یکی از مواردی که باید بررسی بشه اینه که ایا دامنه ایمیل شما، SPF Record داره یا خیر؟ سوالی که پیش میاد اینه که چطوری میتوانیم dig یک دامنه رو بررسی کنیم؟ برای این کار ابزار های زیادی وجود داره، میتوانید از nslookup که تحت ترمینال هستند استفاده کنید و یا از طریق ابزار های آنلاین به بررسی وجود یا عدم وجود SPF Record بپردازید . من یک به یک این موارد رو بررسی میکنم .

چطوری از طریق dig بررسی کنیم که ایا SPF Record وجود داره یا خیر؟ میدونیم که dig یک ابزار تحت ترمینال هست و روی ویندوز وجود نداره . گفتم که SPF Record توی رکورد های TXT یک دامنه ذکر میشه، پس باید از طریق dig درخواست کنیم که TXT رکورد های یک دامنه چی هستند و توی محتوای TXT رکورد ها به دنبال SPF Record بگردیم :

```
kali@kali:~$ dig TXT [REDACTED].net

; <>> DiG 9.19.17-2~kali1-Kali <>> TXT [REDACTED].net
; global options: +cmd
; Got answer:
; ->>HEADER<- opcode: QUERY, status: NOERROR, id: 47991
; flags: qr rd ra; QUERY: 1, ANSWER: 2, AUTHORITY: 0, ADDITIONAL: 1

; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; MBZ: 0x0005, udp: 512
; QUESTION SECTION:
[REDACTED].net.          IN      TXT

; ANSWER SECTION:
[REDACTED].net.      5      IN      TXT      "google-site-verification=wQFxkmYLNedi2tqoiKh_e8-vWMzaafcaMI5Gi6t8_h4"
[REDACTED].net.      5      IN      TXT      "v=spf1 a mx ip4:79.175.[REDACTED] include:_spf.elasticemail.com ~all"

; Query time: 135 msec
; SERVER: 192.168.89.2#53(192.168.89.2) (UDP)
; WHEN: Fri Feb 23 03:22:51 EST 2024
; MSG SIZE rcvd: 201
```

من یک دامنه رو که توی تصویر بالا با رنگ قرمز سانسور شده تست کردم . میبینید که از dig خواستم که از TXT Record های این دامنه رو بهم بده . رفته و بررسی کرده و دیده دوتا TXT رکورد وجود داره که اولین مورد که google-site-verification داره مربوط به SEO هست و ربطی به مانداره ولی دومین مورد که میبینید v=spf1 شروع کننده هست همون SPF Record ماست . میبینید یک IP Address برash تنظیم شده و درست هم هست . دقت کنید که گاهی اوقات از طریق dig TXT site.com درست که رکورد های TXT درست برگردونه نشه و میتوانید خودتون از طریق dig ANY site.com تمام رکورد ها رو بگیرید و TXT ها رو بررسی کنید .

چطوری از طریق nslookup به بررسی TXT Record ها بپردازیم؟ nslookup یک ابزار ویندوزی هست و البته فک کنم تحت لینوکس هم داریمش . کار باهاش قلق داره و مثل dig سرراست نیست ولی خب میشه یاد گرفت . برای اینکه TXT Reocrd ها رو بگیرید باید از سوئیچ q- با مقدار txt استفاده کنید . به شکل زیر ما اینکار رو کردیم و میبینید که TXT رکورد ها برگشته و SPF Record توشه :

```
C:\Users\alime>nslookup -q=txt [REDACTED].net
Server: dns.google
Address: 8.8.8.8

Non-authoritative answer:
[REDACTED].net  text =
[REDACTED].net  text =
"v=spf1 a mx ip4:79.175.[REDACTED] include:_spf.elasticemail.com ~all"
[REDACTED].net  text =
"google-site-verification=wQFxkmYLNedi2tqoiKh_e8-vWMzaafcaMI5Gi6t8_h4"
```

ابزار mxtoolbox.com و بررسی صحت SPF Record : ادرس زیر یک ابزار آنلاین رو در اختیار ما قرار میده که یک دامنه رو از ما میگیره و SPF Record این دامنه رو بررسی میکنه، درصورت وجود SPF Record، صحت پیکربندی ان را نیز بررسی میکنه و به ما میگه که ایا وجود داره یا نداره؟ و اگه وجود داره ایا درست پیکربندی شده یا خیر؟ چون اگه درست پیکربندی نشده باشه هم امکان اکسپلوبیت کردنش وجود داره .

<https://mxtoolbox.com/spf.aspx>

Kafieh ke adres damine mord nizhaton ro towi qesmet Domain Name warden kndid o sipes roo dekmeh SPF Record Lookup klyik knid :

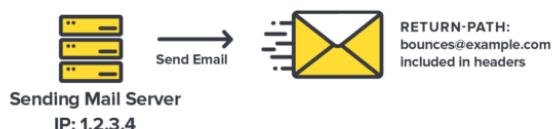
Prefix	Type	Value	PrefixDesc	Description
v	spf1			The SPF record version
+	mx		Pass	Match if IP is one of the MX hosts for given domain name.
+	ip4	86.104.44.103	Pass	Match if IP is in the given range.
+	a	mail.divar.ir	Pass	Match if IP has a DNS 'A' record in given domain.
+	redirect	_spf.aresiranian.com	Pass	The SPF record for Value replaces the current record.
-	all		Fail	Always matches. It goes at the end of your record.

Towt tsoobir bala mibinid ke chahar molfeh bari SPF Record berrsi shde, وجود v, mx, ip4, a, redirect, all و در ston Description tsoobit damine nesbat be ain mوارد ro noshon mide. Ston Description tosbiyati drmord ain molfeh ha dade ke mthla molfeh v orzun SPF roo mesbuc mikkhe .

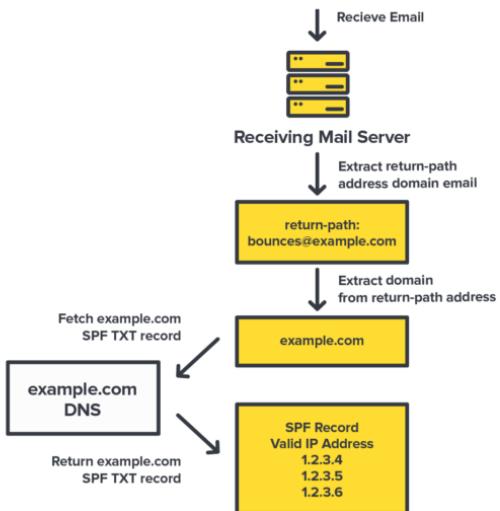
Be nzer mn ke ain abzar waqaa bari berrsi krdn SPF Record khob uml mikenha o hm molfeh ha roo mige o hm drmordshon tosbiy miide .

Tnha yki az molfeh hais DNS hest ke az sh bari taliid minbg arsal kndde yki aimil astqadeh mikenha, molfeh hais diighe mthl ke mxfv DMARC o Domian-Based Message Authentication Reporting and Conformance hest hm wjod dard. Shavid ain du mord ro hm berrsi krdim wli hnuz SPF Record DomainKeys Identified Mail hest . Sxhtari mord berrsi qrar nadayim . Qbil az berrsi sxhtari SPF Record brym bbinim ke chtoroi yki Mail Server bari arzaz hovit .

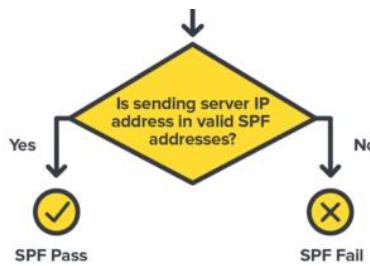
- Srror owl yki aimil ro arsal mikenha . Frras mkgiribm ke Return Path IP Address srror owl 1.2.3.4 o astqadeh shde dr aimil : [bounces@example.com](mailto:bounces@example.com)



- Srror dom ke aimil ro driyافت mikenha, damine SPF Record ro mkgirhe o Return-Path ain damine ro berrsi mikenha .



- در صورتی که سرور دوم که دریافت کننده هست، Return-Path دامنه SPF Record را پیدا کرد، نگاه میکنی که ببینه ایا IP Address ارسال کننده ایمیل که سرور اول هست، توی لیست SPF Record دامنه SPF Record هست یا خیر . اگه SPF Pass سرور ارسال کننده توی SPF Record دامنه SPF Record بود، به معنی درست بودن هویت ارسال کننده هست و خواهد شد، در صورتی که نبود، SPF Fail خواهد شد . در این حالت یا ایمیل Reject میشه و یا به عنوان Spam در نظر گرفته خواهد شد .



البته لازم هست که ذکر کنم، SPF Fail انواعی داره، Hard Fail, Natural Fail, Soft Fail که فعلاً توی بحثمون نیست و فقط جهت اطلاع گفتم .

بسیار هم خوب، این طریقه احراز هویت شدن یک ارسال کننده ایمیل از طریق SPF Record بود . باید یادمون باشه که چی میشه و چطوری اینطوری میشه . چون مشکل امنیتی که میخوایم بررسی کنیم دقیقاً علش SPF Record هست .

مورد بعدی که میخوایم درمورد SPF Record بررسی کنیم، ساختار SPF Record هست . باید بدونیم که SPF Record باید چه ساختاری داشته باشه، چه ساختاری اشتیاس است و چه ساختاری درست است . چرا باید تقاضت ساختار درست و اشتباه رو بدونیم؟ چون علاوه بر اینکه عدم وجود SPF Record میتوانه موجب اسیب پذیری بشه، ساختار اشتباه هم در گاهی اوقات میتوانه منجر به اکسپلوبیت شدن بشه . ساختار کلی یک SPF Record به شکل زیر است :

```
v=spf1 ip4:192.0.2.0 ip4:192.0.2.1 include:examplesender.email -all
```

SPF Record باید یک ساختار مشخص و استاندارد داشته باشه تا سرور دریافت کننده بدونه که چطوری باید SPF Record را تفسیر کنه . مثال ساختاری بالا نوی خودش، نوع Record رو تعیین کرده، IP ها تایید شده رو گفته، دامنه third-party همچنین به سرور میگه که با ایمیل ها ناسازگاری که دریافت میکنه چیکار کنه؟ میبینید که توی یک SPF Record ما 4 مؤلفه رو باید داشته باشیم . حالا بریم سروقت توضیح دادن هریک از مؤلفه ها :

- v=spf1: این عبارت به سرور دریافت کننده میگه که "دارم یک SPF Record رو تعریف میکنم ". پس هر SPF Record با عبارت v=spf1 شروع خواهد شد . اون | هم به نظرم ورژن spf رو بیان میکنه .

- 1.1.1.1: یک مثالی از تگ ipv4:192.0.2.0 ipv4:192.0.2.1: مثالی IP Address های مجاز و Authorized رو توی این قسمت همون Guest List SPF Record هست که توی SPF Record بالا گفته شده که 192.0.2.0 و 192.0.2.1 مجاز هستند که ایمیل ارسال کنند . یعنی زمانی که یک ایمیل رو دریافت کرد، ادرس سرور ارسال کننده باید یکی از این IP Address ها باشه و گرنه SPF Fail .

- یک مثالی از تگ include:examplesender.email: کدام سازمانهای third-party هم از طریق دامنه SPF Record هست . تگ include برای سرور دریافت کننده مشخص میکنه که کدام سازمانهای Return-Path مجوز ارسال ایمیل رو داره . این تگ

اشاره میکنه که محتوای SPF Record دامنه مشخص شده (examplesender.email) باید بررسی شود و IP Address SPF Record این دامنه هم مجاز هستند به ارسال ایمیل . شاید توی برخی موارد مقدار این تگ یه چیزی مثل elasticmail.com مثل باشه که میتونه نشون دهنده پست ابر بودنش تلقی بشه .

- all - مشخص میکنه که هیچ سروری با هیچ IP Address که توی SPF Record مشخص نشده باشه اجازه ارسال ایمیل رو نداره و نمیتونه ایمیل ارسال کنه . اما به جز all- دو مقدار دیگه هم میتونه داشته باشه که عبارت اند از :
  - all- که میگه، از سرور های دیگه هم ایمیل بپذیر و لی اونها رو به عنوان ایمیل ها نامن و Spam نشون بد .
  - +all: میگه، از سرور های دیگه هم ایمیل بپذیر و مشکلی نداره .

یکی دیگه از مولفه هایی که میتونه توی SPF Record وجود داشته باشه redirect= هست . توی مثال ما نبود ولی خب ممکن هست که خیلی از جاهها بهش بر بخورید . این مولفه میگه که SPF Record دامنه مورد نظر رو باید از redirect= مقدار SPF Record بخونید . مثلا به شکل :

```
v=spf1 ... redirect=_spf.example2.com -all
```

البته لازم به ذکر هست که، SPF Record میتونه پیچیده تر بشه و مولفه های دیگه ای هم بهش اضافه گردد ولی چند نکته رو درمورد پیکربندی SPF رکورد به یاد داشته باشیم :

- بیشتر از یک مورد SPF Record برای یک دامنه نباید تعیین شود .
- SPF Record باید با مولفه all خاتمه بپیدا کند یا شامل redirect= باشد .
- SPF Record نمیتوانه شامل حروف بزرگ شود .

حال ساختار SPF Record رو هم فهمیدیم و میتوانیم پیکربندی درست رو از اشتباه تشخیص بدیم . درصورتی که all+ در انتهای SPF Record ذکر شده باشه به معنای پیکربندی اشتباه هست و قابلیت اکسپلوبیت شدن رو داره .

برای اطلاعات کاملتر درمورد SPF Record میتوانید به RFC مربوط به مراجعه کنید که به ادرس زیر است :

<https://datatracker.ietf.org/doc/html/rfc7208>

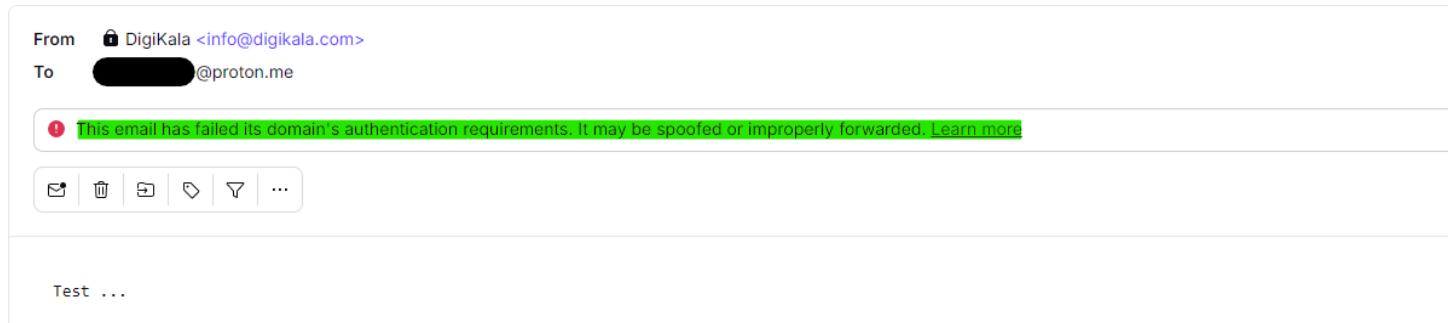
درمورد DKIM, DMARC شاید در اینده صحبت کردیم . بسته به شرایط خواهد داشت . حال که درمورد SPF Record صحبت کردیم میتوانیم بریم و درمورد Email Spoofing که یک نوع حمله است صحبت کنیم . بریم سروقتش ...

Email Spoofing چیه و چطوری انجام میشه ؟ به ایجاد یک پیام ایمیل از طرف یک ارسال کننده جعل شده، Email Spoofing میگن . هموطنوری که گفتم، پروتکل استاندارد ارسال و دریافت ایمیل (SMTP) به صورت ذاتی، مکانیزم احراز هویت نداره و همین موجب میشه که به طریقی بشه ایمیل های جعلی ارسال کرد . امیدوارم SMTP Injection را به یاد داشته باشیم، یک حفره امنیتی که موجب میشد بتونیم، ظاهر ارسال کننده ایمیل رو برای دریافت کننده تغییر بدم و به شکلی به نظر برسه که ایمیل از طرف یک شخص دیگر ارسال شده است و این در حالی بود که میشد در Original Show ایمیل، ارسال کننده واقعی رو پیدا کرد . درمورد Email Spoofing اینطوری نیست و ما از بین سعی میکنیم که ایمیل رو از طرف یک ارسال کننده جعلی ارسال کنیم . این حمله میتوانه موجب Spam و Phishing شود و دریافت کننده رو به شکلی قانع کنه که کاری رو انجام بده .

اگه بخواه به دلایلی که موجب میشه حمله Email Spoofing رخ بده اشاره کنم، میشه عدم وجود SPF یا پیکربندی اشتباه Record رو ذکر کرد . درمورد SPF Record گفتیم که کارش چیه، چه ساختاری داره، پیکربندی اشتباه و درست چی هست و ... .

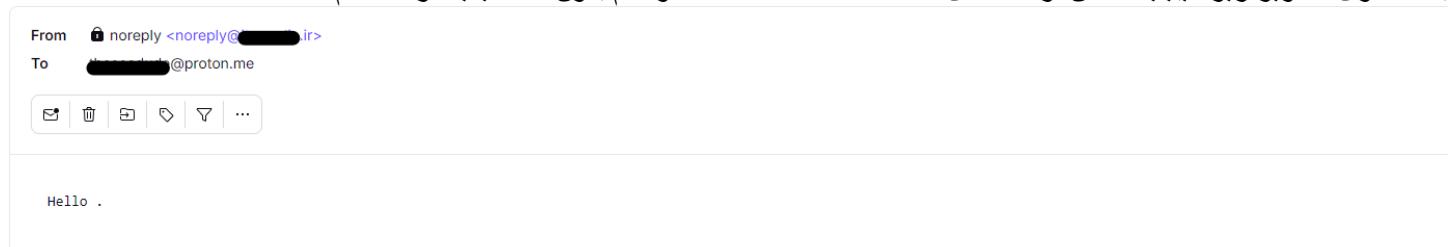
Tاثیر و Impact انجام چنین حمله ای چیه ؟ این حمله میتوانه موجب Spam و Phishing کردن دریافت کننده بشه . مهاجم میتوانه با ارسال یک ایمیل جعلی به دریافت کننده، به حالتی که ارسال کننده ایمیل یک فرد قابل اعتماد است، دریافت کننده رو مجاب به انجام عملی کنه که به صورت عادی نباید انجام بده . مثلا ارسال ایمیل از طرف مدیریت یک سازمان یا ... . پس Email Spoofing یکی از پاهای ثابت Social Engineering, Phishing, ... است .

چه جاهابی هست که میتوانیم ازش استفاده کنیم و Email Spoofing رو انجام بدم ؟ شما قاعدها از هر دامنه ای باید بتونید ایمیل رو ارسال کنید و فقط و فقط به یک SMTP Server برای ارسال احتیاج دارید . اگه به تصویر زیر دقت کنید من تونستم از طریق دامنه digikala.com به ایمیل خودم ایمیل ارسال کنم و همچنین میبینید که یک پیغام خطأ هم به من داده :



Test ...

تقریباً میشه از هر دامنه ای ایمیل ارسال کرد (تا این لحظه که دارم این جزو را مینویسم چنین فکر میکنم) ولی دقت کنید که دریافت کننده ایمیل موظف هست که SPF Record دامنه digikala.com میشه رو بررسی کنه و با سرور ارسال کننده قیاس انجام بده و چون SPF Record دامنه digikala.com با سرور ارسال کننده ایمیل بالا تطابق نداشته به من ایمیل رو نشون داده ولی پیام خطای SPF داره و گفته که این ایمیل خطای Authenticate نمیشه، ممکن هست که Spoof شده باشه. حتی برخی سرویس های ایمیل مثل Gmail ممکن هست اصلاً ایمیل دریافتی رو نشون هم نده، البته این به معنای ارسال ایمیل نیست، ایمیل Spoof شده ارسال میشه ولی سرور دریافت کننده هست که تصمیم میگیره برآساس وضعیت SPF ایمیل دریافتی چه واکنشی رو نشون بده.  
اما چه جاهایی میتوانیم با موفقیت Email Spoofing رو انجام بدیم؟ زمانی که SPF Record وجود نداشته باشه و یا درست پیکربندی نشده باشه. توی تصویر زیر میبینید که من از دامنه ای که SPF Record نداشته تو نstem بدون خطا ایمیل ارسال کنم:



پس نقاط اسیب پذیر کجاست؟ دامنه های تارگتها نقاط اسیب پذیر محسوب میشن و همچنین SMTP Server هاشون.

طریقه کشف اسیب پذیر بود یک تارگت نسبت به SPF Record چطوریه؟ بررسی کردن SPF Record ثبت نشده باشه، تارگت اسیب پذیر خواهد بود و همچنین درصورتی که SPF Record درست پیکربندی نشده باشه هم میتوانه امکان اسیب پذیر بودن رو بالا ببره. فرض کنید که یک تارگت داریم به نام site.com . میخوایم بررسی کنیم که ایا اسیب پذیر هست یا خیر؟ کافیه که از طریق dig, nslookup یا هر چیز دیگه که در دسترسمون هست SPF Record توی TXT Record را بیرون بیاریم. من برای این کار سایت <https://mxtoolbox.com/spf.aspx> رو استفاده میکنم. درصورتی که پاسخی به شکل زیر دریافت کردیم، به معنی پیکربندی نشدن SPF Record است:

https://mxtoolbox.com/SuperTool.aspx?action=spf%3asite.com&run=toolpage

MX TOOLBOX

Pricing Tools Delivery Center Monitoring

SuperTool Beta

SuperTool

MX Lookup Blacklists DMARC Diagnostics Email Health DNS Lookup Analyze Headers

spf:site.com

Find Problems Solve Email Delivery Problems

Test	Result
SPF Record Published	No SPF Record found

More Info

در این صورت ما میتوانیم حمله Email Spoofing رو اجرا کنیم. در برخی اوقات هم SPF Record پیکربندی شده ولی به درستی پیکربندی اعمال نشده است و با وجود SPF Record، باز هم تارگت اسیب پذیر هست. یکی از موارد وجود +all در SPF Record هست که موجب اسیب پذیر شدن خواهد شد. حتی گاهی اوقات وجود ~all هم میتوانه اسیب پذیر باشه، باید از طریق یک SMTP Server سعی کنیم یک ایمیل جعلی به خودمون از طریق تارگت ارسال کنیم تا ببینیم ایا ایمیل ارسال میشه؟ ایا اخطار Spoof بودن ایمیل داده میشه؟ اگه اسیب پذیر باشه، ایمیل دریافت میشه و اخطاری نخواهد داد.

طریقه اکسپلوبیت کردن Email Spoofing چطوریه؟ پس از اینکه کشف کردیم یک تارگت اسیب پذیر هست به ایمیل Spoofing وقت اکسپلوبیت کردن و بدست اورن POC فرا میرسه. ما باید چطوریه اکسپلوبیت کنیم؟ باید سعی کنیم از طرف ایمیل روی دامنه تارگتمن، یک ایمیل جعلی ارسال کنیم، پس نیاز داریم به ارسال ایمیل. خب ارسال ایمیل توسط یک Email Server انجام میشه، پس در ابتدا نیاز به یک SMTP Server یا Email Server خواهیم داشت که بتوانیم ایمیل رو ارسال کنیم. یا باید یک SMTP Server اجاره کنیم و یا سعی کنیم یک SMTP Server را یکان پیدا کنیم تا برآمون ایمیل رو ارسال کنیم. فرض میگیریم که SMTP Server رو پیدا کردیم و برآمون ایمیل رو ارسال کرد، حال نیاز داریم تا ایمیل ارسالی رو دریافت کنیم و وضعیت SPF ایمیل دریافتی رو ببینیم. برای SMTP Server ارسال کننده ایمیل چه راهکارهایی وجود داره؟ میتوانیم یک SMTP Server اجاره کنیم و از طریق یک کد Python بهش متصل بشیم و سعی کنیم ایمیل رو ارسال کنیم و یا از سایت <https://emkei.cz> استفاده کنیم. این سایت یک SMTP Server داره و اطلاعات رو از ما میگیره و ایمیلی رو ارسال میکنه. صفحه سایت به شکل زیر است:



فرض کنید که تارگت ما [info@site.com](mailto:info@site.com) هست که به Email Spoofing اسیب پذیر هست. فرم بالا از ما نام ارسال کننده، ایمیل ارسال کننده، ایمیل دریافت کننده، موضوع ایمیل، فایل های ضمیمه ایمیل و متن ایمیل رو میگیره و ایمیل رو ارسال میکنه. کافیه که فرم رو به شکل زیر پر کنیم:

From Name:	Admin
From E-mail:	<a href="mailto:info@site.com">info@site.com</a>
To:	receiver@attacker.com
Subject:	Email spoofing test
Attachment:	<input type="button" value="Choose File"/> No file chosen <input type="button" value="Attach another file"/> <input type="button" value="Advanced Settings"/>
Content-Type:	<input checked="" type="radio"/> text/plain <input type="radio"/> text/html <input type="checkbox"/> Editor
Text:	
Hello, This is a test for email spoofing from info@site.com  <b>Captcha:</b> <input type="checkbox"/> I am human  <small>Privacy - Terms</small>	
<input type="button" value="Send"/> <input type="button" value="Clear"/>	

حال کافیه که روی دکمه Send بزنیم و ایمیل رو ارسال کنه. پیغام زیر به معنی ارسال موفقیت امیز ایمیل هست ولی نه به معنی اسیب پذیر بودن تارگت:

Free online fake mailer with attachments, encryption,  
HTML editor and advanced settings...

 E-mail sent successfully

حال کافیه که به SMTP Server های دریافت کننده دریافت ایمیل و بیانیم وضعیت ایمیل دریافتی چطوری هست. گاهی SPF Record را بررسی میکن و در صورتی که مشکلی وجود داشته باشد، یا ایمیل را Reject میکن و یا اون رو در قسمت Spam قرار میدن و در صورتی که SPF Record مشکلی نداشته باشد و به عبارتی دیگه SPF Pass بشه، ایمیل دریافت خواهد شد. وقت کنید که Gmail نسبت به این موضوع نسبت به سرویس های ایمیل دیگه سخت گیرانه تر عمل خواهد کرد و حتی به شکلی ممکن هست که ایمیل های دریافتی از <https://emkei.cz> رو به هیچ وجه دریافت نکنه و به کاربر نشون نده. به همین خاطر، به نظر من برای اینکه به درستی بتونید یک POC داشته باشید، از سرویس های ایمیلی مثل Proton Mail استقاده کنید.

بدین شکل ما میتوانیم وجود امکان Email Spoofing را اکسپلوبت کنیم. وقت کنید که Email Spoofing Email رو اگه جامع نگاه کنیم یک حفره امنیتی تحت وب محسوب نمیشه و در حقیقت یک حفره امنیتی Network خواهد بود ولی به علت اینکه توی باگ بانتی ها، اثبات وجود این مشکل امنیتی مورد قبول هست ما هم اینجا در مرورش صحبت کردیم.

موانعی که میتوانن جلوی Email Spoofing را بگیرن چیا هستند؟ گفتم چه چیزی میتوانه موجب Email Spoofing بشه؟ نبود SPF Record یا پیکربندی نادرست SPF Record، پس میتوانیم در ابتدا بگیم که سخت ترین مانعی که میتوانه جلوی Email Spoofing را بگیره، وجود درست و پیکربندی درست SPF Record هست که این قاعده رو ختم به خیر میکنه. اما به صورت کلی موانعی که میتوانه جلوی حمله Email Spoofing به یک سازمان رو بگیره به عبارت زیر هستند:

- Email Filters and Spam Settings : قاعدنا وقتی که وضعیت SPF یک ایمیل Pass بشه این ایمیل باید از نظر سرویس دریافت کننده، مشکوک باشد. سرویس ایمیل یک سازمان باید طوری پیکربندی و نوشته بشه که وضعیت SPF یک ایمیل دریافتی رو بررسی کنه و در صورت Fail بودن، هرنوع Fail، اون ایمیل رو یا از دسترس خارج کنه و یا اون رو به عنوان Spam شناسایی کنه.
- Email Authentication Methods : گفتم که SPF یک روش احراز هویت ایمیل ارسالی هست که توسط دریافت کننده بررسی میشه ولی تنها روش نیست، روش های دیگه ای مثل DKIM و DMARC هم هستند که میتوان یک ایمیل دریافتی رو احراز هویت کنند. وجود هر کدام از این متد ها میتوانه تخریب و احتمال اکسپلوبت شدن درست Email Spoofing رو کاهش بده و یا حتی به کلی رفع کنه.
- User Education and Awareness : هدف اصلی مهاجم از انجام حمله Email Spoofing استفاده از مهندسی اجتماعی برای دسترسی بیشتر به چیزیست که مهاجم نباید دسترسی داشته باشد. وجود یک کاربر آموزش دیده و اگاه نسبت به مسائل امنیتی میتوانه حتی در صورت وجود اسیب پذیری، امکان دسترسی مهاجم به چیزی که نباید دسترسی داشته باشد رو از بین ببره. کاربر باید بدونه که چطوری یک ایمیل Spoof شده رو از یک ایمیل واقعی تشخیص بده، باید به محتوای ایمیل دقت کنه، ایمیلی که در خواستی ناجا رو به کاربر میکنه، ایمیلی که از لحاظ گرامری با چیزی که در گذشته بوده متفاوت هست، ایمیل که لینک ها و فایل های مشکوک رو فرستاده باید توسط کاربر شناسایی بشن.
- Email Security Solutions : راهکار های امنیتی باید برای SMTP Server دریافت کننده لحاظ شده باشد. سرور دریافت کننده باید توانایی بررسی الوده بودن یا نبودن ضمیمه های یک ایمیل دریافتی رو داشته باشد و همچنین بتونه لینک های دریافتی یک ایمیل رو اسکن کنه، چنین مواردی میتوانن موجب بشن که یک مهاجم با انجام Email Spoofing به هدف خودش نرسه.
- ...
- روش های جلوگیری از به هدف رسیدن حمله Email Spoofing به علت اینکه در گروه حملات مهندسی اجتماعی قرار میگیره زیاد هستند و به نظر من ابتدا باید راهکار های امنیتی فی رو به درستی اعمال کرد و سپس کاربر ها رو به اندازه کافی نسبت به مسائل امنیتی اگاهی بخشید.

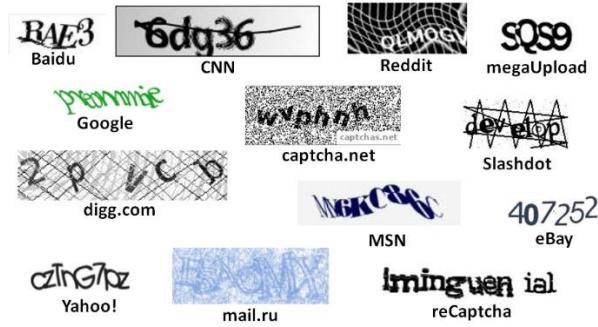
[00:38:23]

توی این قسمت میخوایم درمورد مشکلات امنیتی که ممکن هست کپچا ها داشته باشند صحبت کنیم . اما قبل از اینکه بریم سروقت مشکلات امنیتی، باید دیدی رو نسبت به مفهوم کپچا و انواع ان داشته باشیم . باید بدونیم که چطوری کار میکند و چرا بوجود آمدند ؟

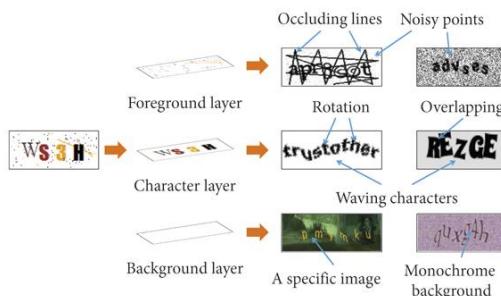
**Completely Automated Public Turing test to tell Computers and Humans Apart** چیست ؟ کلمه **Captcha** هست و معنی این جمله "یک تست تورینگ کاملا خودکار جهت جدا کردن انسان از کامپیوتر" میشود . به طور تخصصی، **CAPTCHA** یک اقدام امنیتی جهت تایید این هست که یک کاربر اینلاین که قصد دسترسی به یک اپلیکیشن رو داره ربات نیست و انسان می باشد پس میشه نتیجه گرفت که **CAPTCHA** Task محسوب میشون که انسان میتونه به راحتی حلش کنه ولی برای کامپیوتر ها و رباتها حل کردن دشوار خواهد بود . مثلاً مجموعه ای از حروف کج و کوله شده که کنار هم قرار گرفته اند، برای انسان قابل تشخیص است ولی کامپیوتر ها در تشخیص این حروف به علت کج و کولگی به مشکل خواهد خورد .

**CAPTCHA** هایکی از اولین اقدامات امنیتی هست که توسط برنامه نویسان جهت بیشتر کردن امنیت کاربران، در قسمت های مختلف یک اپلیکیشن مثل **Form Field** ها و ... قرار داده میشود . یکی از حملاتی که **CAPTCHA** ها برای رفع کردنشون ایجاد شدن، حملات-**Brute Force** هستند . در ادامه با انواع حملات اشنا خواهیم شد .

انواع **CAPTCHA** چیا هستند ؟ **CAPTCHA** ها به طور کلی در سه دسته بندی قرار میگیرند که عبارت اند از :  
 ۱- **Text-Based Captcha** : این نوع کپچا به کاربر متى کج و کوله رو نشون میده که این متن میتونه حروف تصادفي، اعداد تصادفي و عبارات شناخته شده باشه . چنین **CAPTCHA** هایی نسبت به دو نوع دیگه قدیمه تر محسوب میشون چرا که اولین نوع کپچا هستند که ایجاد و استفاده شده اند . دقت کنید که **Text-Based Captcha** ها اینطوری نیستند که فقط یک عبارت شامل حروف و اعداد رو نشون بند، بلکه سعی میکن این حروف و اعداد رو به شکلی کج و کوله کنند تا توسط کامپیوتر ها و باتها قابل تشخیص نباشند . مثلاً ممکن هست که یک کاربر به جای یک عبارت ساده، یک عبارتی رو بینه که اجرای آن **Scale**، **Rotate** و در هم هستند و همچنین شامل اجزای اضافی مثل رنگ، خطوط، نقطه ها و ... هم میشوند . به عبارتی، اون کدی که سعی میکنه یک **Text-Based Captcha** رو بسازه تمام تلاشش رو میکنه تا چیزی رو ایجاد کنه که توسط کامپیوتر و باتها قابل تشخیص نباشه ولی انقدر هم پیچیده نشه که توسط انسان هم تشخیص داده نشود . تصویر زیر نمونه هایی از این نوع کپچا که در سایتها مختلف استفاده میشده رو نشون میده :



توی تصویر زیر هم میبینید که یک **Text-Based Captcha** از چه اجزاء ای تشکیل شده است :

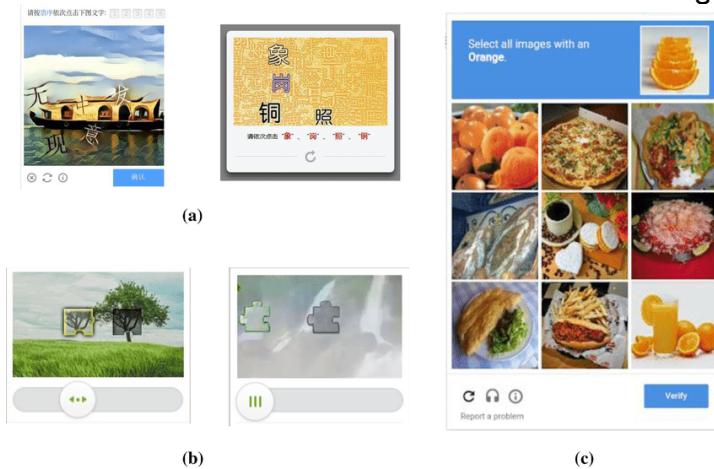


میبینید که یک لایه **Foreground** وجود داره، یک لایه **Character** و یک لایه **Background** و از ترکیب این سه لایه یک **Text-Based Captcha** ایجاد میشود . در ادامه بیشتر درمورد این نوع کپچا صحبت خواهیم کرد و طریقه بایپس کردنش رو خواهیم دید .

نوع **Text-Based CAPTCHA** خوش انواعی خواهد داشت و اگه بخوایم به همه انواع **CAPTCHA** ها و زیر مجموعه هاشون پردازیم باید یک کتاب بنویسیم ولی خب در تصویر زیر برآتون برخی از انواع **Text-Based CAPTCHA** رو اوردم که میتوانید استفاده کنید :

Protection mechanism	Example
Anti-segmentation	Hollow symbols 
	crowding characters together (CCT) 
	Background noise 
	Two-level structure 
Anti-recognition	Different fonts, Rotations, Wave-like symbols 
	Different languages 

2. **Image-Based Captcha**: این نوع کپچا به کپچا های تشخیص تصویر (Image Recognition CAPTCHA) معروف هستند چرا که کاربری که بهشون برخورد میکنه باید یک تصویر را تفسیر و به درستی تشخیص دهد . یک **Image-Based CAPTCHA** میتوانه هرچیزی از تصویر حیوانات تا عناصر گرافیکی و مناظر طبیعی باشه . برای حل این کپچا ها یک مجموعه از تصاویر به کاربر نشون داده میشه و کاربر باید تصاویری که بهش گفته شده رو از بین این تصاویر انتخاب کنه، مثلاً گفته میشه که "تصاویری که شامل گربه توشن هست رو انتخاب کن ". یا در برخی دیگه باید تیکه هایی از یک پازل رو توى جای خودشون قرار بدیم که بهشون **Mouse Drag Based CAPTCHA** میگن که نوعی از **Image-Based CAPTCHA** محسوب میشه و ... حل کردن این نوع کپچا برای انسان نسبت به حل کردن **Text-Based CAPTCHA** ها راحت تر هست . توى تصویر زیر نمونه ای از **Image-Based CAPTCHA** رو میبینید :



البته انواع **Image-Based CAPTCHA** ها به همین تصویر بالا ختم نمیشه و هر کسی او مده طبق خلاقیتی که به خرج داده یه چیزی ایجاد کرده، اما چون همه این کپچا ها از عنصر تصویر توشن استفاده شده بهشون **Image-Based CAPTCHA** میگن . حل کردن این نوع **CAPTCHA** واسه ربات خیلی دشوار و حتی غیر ممکن هست ولی حل کردن **Text-Based CAPTCHA** ها میتوانه راحت انجام بشه . مخصوصاً با کتابخونه های پردازش تصویری که امروز توى زبون های برنامه نویسی پایتون و ... پیدا میشه، کافیه که يه خورده باهاشون ور بری تا کتاب خونه های **OCR** بتونن متن داخل تصویر رو بخونن . 3. **Audio-Based Captcha**: این نوع کپچا رو کمتر دیدیم ولی جاهایی استفاده میشه و یک جایگزین برای کپچاهای وابسته به قدرت بینایی هست و برای کسانی استفاده میشه که اخلاق و مشکل بینایی دارند . در این نوع کپچا به جای استفاده از تصاویر و متن های کج و کوله، کاربر میتوانه به یک صوت گوش بده و توى این صورت مجموعه ای از حروف و اعداد بازگو میشود و باید جهت تایید کپچا این حروف و اعداد رو تشخیص بدنهن . میدونیم که امروزه زبان های برنامه نویسی کتابخونه های پردازش صدا هم دارند و به قدری پیشرفت کردد که به راحتی بتونن هر آنچه در یک صوت گفته میشود را با درصد خطای نه خیلی زیادی تشخیص بدنهن . حال اگر این زبات ها بتونن **Audio-Based CAPTCHA** ها رو تشخیص بندن که کارمون زاره !!! پس میان و توى این کپچاهای به جز صدای اصلی از صدای ای استفاده میکنن تا امکان تشخیص هر چه گفته میشه تو سط ربات کمتر بشه، نویز هایی در پس زمینه صدای اصلی قرار داده میشه .

حال که مفهوم کپچا را درک کردیم، سوال اینه که کپچاها چطوری کار میکنند؟ کپچا ها وظیفه دارند که یک چالش رو برای کاربرانشون ایجاد کنند که کاربر بدون حل کردن این چالش امکان ادامه دادن و دسترسی نداشته باشد. مثلاً توانی صفحه لاگین یک وب اپلیکیشن، شما نام کاربری و رمز عبورتون رو وارد میکنید و ممکن هست که یک عبارت امنیتی هم ببینید، بدون بدست اوردن این عبارت امنیتی، شما امکان لاگین کردن نخواهید داشت چرا که عبارت امنیتی بدست او مده توسط شما به همراه نام کاربری و کلمه عبورتون به سمت وب سرور ارسال و بررسی میشوند و در صورت اشتباه بودن به شما اجازه ادامه دادن نمیشود تا زمانی به درستی عبارت امنیتی رو حل کنید.

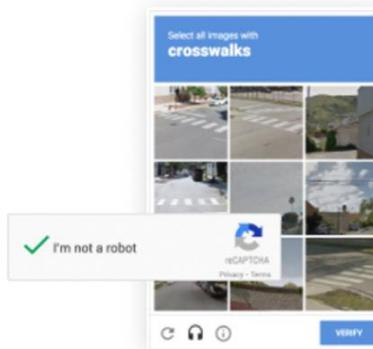
در مرور مفهوم کپچا صحبت کردیم، اما امروزه ما با یک چیز دیگه به نام reCAPTCHA هم سروکار داریم؟ ایشون چی هستند و چطوری کار میکنند؟ یک سرویس رایگان گوگل هست که به عنوان جایگزینی برای CAPTCHA ها قدمی ارائه شد. این سرویس در سال 2009 منتشر شد و مورد استفاده قرار گرفت.

reCAPTCHA نسبت به کپچاهای سنتی و قدیمی پیشرفته تر طراحی شده است و از ماشین لرنینگ و امکانات پیشرفته دیگه ای استفاده میکنه و به این طریق انسان رو از ربات ها تشخیص میدهد. اما چطوری کار میکنه؟ پروسه تابید یک کاربر توسط reCAPTCHA با استفاده از AI صورت میگیره و این AI از طریق رفتار کاربر به این نتیجه میرسه که کاربر مورد نظر ربات هست یا خیر. گفتیم که پروسه تابید کپچاهای قدیمی از طریق مجبور کردن کاربر به وارد کردن یک متن کج و معوج در یک ورودی انجام میشد و خوندن این متون برای ربات ها دشوار بود ولی با پیشرفت کردن تکنولوژی و بوجود امدن ماشین لرنینگ، AI، پردازش تصویر و صدا و ... این ربات ها تونستن از پس حل کردن کپچاهای قدمی بر بیان و تقریباً عموم کپچاهای قدیمی حل میشند. همینجا بود که اولین نسخه ریکپچا به نام AI توسط گوگل منتشر شد.

توی نسخه اول reCAPTCHA تست ها از یک کلمه ای که توسط کامپیوتر ساخته شده و یک متن کج و کوله از یک تصویر کتابهای قدیمی یا مقالات خبری استفاده میکردد. امروزه این ورژن از reCAPTCHA در دسترس نیست.



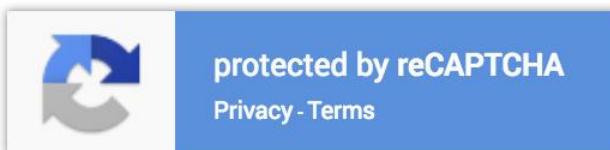
ورژن بعدی v2 reCAPTCHA بود و امید بود که بتونه تست های پیچیده تری رو بوجود بیاره که حل کردنشون برای ربات ها دشوار و برای انسان ها راحتر باشه. این ورژن Image-Based بود و کاربر رو مجبور میکرد که طبق یک متن تصاویری رو از لیستی از تصاویر انتخاب کنه و سپس رو یک دکمه که نوشته بود "I'm not a robot" کلیک کنه.



اما ورژن جدید تر - reCAPTCHA v3 - هدفش این بود که از مختل کردن تجربه کاربر جلوگیری کنه. این نسخه تعامل کاربر رو بر اساس محاسبه یک نمره بر اساس حضور و رفتار و تاریخچه کاربر محدود میکنه یعنی طبق عملکردی که کاربر از خودش نشون میده، طریقه حرکت موس، ورودی هایی که وارد میکنه و به عبارتی دیگه رفتار کاربر و همچنین تاریخچه کاربر این ورژن از reCAPTCHA برای کاربر یک امتیازی رو در نظر میگیره و طبق این امتیاز، وب اپلیکیشن سه پاسخ رو میتونه به کاربر بده:

- Grant Access
- Block the user
- Deploy reCAPTCHA v2 test

گزینه اول یعنی اینکه اجازه بده کاربر ادامه بده و زمانی رخ میده که امتیازی که محاسبه شده نشون دهنده انسان بودن کاربر هست، گزینه دوم زمانی هست که امتیاز محاسبه شده کاملاً نشون میده که کاربر یک ربات است و گزینه سوم در حالتی رخ میده که وب اپلیکیشن نسبت به کاربر مشکوک شده باشد.



حالا که از reCAPTCHA اسم بر دیم باید بگیم که یک نمونه دیگه هم وجود داره به نام hCAPTCHA که به جای reCAPTCHA بعضی از اپلیکیشن ها خواهد دید. همون هم مکانیزم های خودش رو داره و تقریباً میشه گفت رقیب جدی واسه hCAPTCHA محسوب میشه.



اغا ما که نیومدیم اینجا بهتون بگیم چطوری امنیت اپلیکیشنتون رو زیاد کنید که، والا ما مثلًا هکر هستیم یا میخوایم هکر شیم، ما اینجاییم که بگیم چطوری میشه امنیت رو اسکل کرد.

سوال بعدی اینه که CAPTCHA در کجا باید استفاده شود؟ ببینید به طور کلی کچرا رو هر جایی که یک ربات میتونه موجب تخریب بشه باید استفاده بشه. هرجایی که یک ربات میتونه مشکل بوجود بیاره باید از طریق کچرا دسترسی اون ربات رو به اونجا از بین ببریم. حالا باید بگیم که کجاها یک ربات میتونه موجبات تخریب رو فراهم کنه؟

- صفحات ورود: صفحات لاگین از جمله صفحاتیست که معمولاً میتونند CAPTCHA رو توش ببینند. هر برنامه نویسی که پیدا شده

فهمیده که لاگین رو حداقل از طریق CAPTCHA امن نر کنه. چرا این کار رو میکن؟ صفحات لاگین مستعد برخی از حملات

محرب هست، حملاتی که جزو دسته Password Attack هم از جمله حملاتیست که رو صفحات لاگین انجام میشه. همه این حملات کاربران وب اپلیکیشن. همچنین حمله Password Spray هم از طریق اسکریپت ها و ربات ها صورت میگیره پس باید چیکار کنیم؟ باید سعی کنیم جلوی دسترسی ربات ها و اسکریپت ها رو بگیریم، باید کاری کنیم که یک اسکریپت نوشته به زبان پایتون نتونه عملیات لاگین رو انجام بد. کچرا اینجا به کارمون میداد. اما دقت کنید که هر کچایی شاید بتونه کار مهاجم رو دشوارتر کنه ولی نمیتوانه احتمال رخ دادن این حمله رو به صفر برسونه پس

باید توی طراحی کچرا دقت کنیم و پیشنهاد اینه که از hCAPTCHA یا reCAPTCHA به جای کچاهای سنتی استفاده بشه.

البته بگم که کچرا تنها مکانیزم امنیتی صفحات لاگین نیست و بایستی از مکانیزم های دیگه ای مثل WAF Rate Limit و ... هم جهت امن کردن صفحات ورود استفاده کنیم.

- صفحات ثبت نام: صفحات ثبت نام چرا باید کچرا داشته باشند؟ ببینید وقتی یک وب اپلیکیشن ثبت نام میکنه حداقل یک

رکورد برای اون شخص توی پایگاه داده ایجاد میشه. فرض کنید که یک مهاجم داریم که کرم داره، میخواهد پایگاه داده رو کند کنه و

حتی کاری کنه که عملیات هاش مختل بشه، وقتی که بینه فرم ثبت نام هیچ کچایی نداره میتوانه از این فرم استفاده کنه و میلیون ها

میلیون کاربر فیک رو ثبت نام کنه و به از ای هر کاربر رکورد هایی رو تو پایگاه داده شما بوجود بیاره که فیک هستند ولی خب

همین وجودشون موجب کندی پایگاه داده خواهد شد و به این کار به اصطلاح میگن Database Flooding. به همین خاطر پیشنهاد

بر اینه که صفحات ثبت نام در وب اپلیکیشن ها هم کچرا داشته باشند. هر چند وجود کچرا User Experience رو زیاد جالب نمیکنه

ولی برای امنیت بیشتر نیاز هست.

- صفحات 2FA یا OTP: این صفحات هم از اون جاهایی هستن که ممکن هست مورد حملات Bruteforce قرار بگیرند و یک

مهاجم سعی کنه داده ای که در اختیار کاربر هست رو حس بزنه. از این بابت نظر به اینه که توی این صفحات هم Robot بودن یا

انسان بودن کاربر رو بررسی کنیم. نظرم این نیست که سعی کنیم از طریق کچرا سنتی این کار رو بکنیم چون در صورت وجود کچرا

سنتی در صفحات ورود و همچنین در این صفحات تجربه کاربر رو خراب میکنه و نظرم به استفاده از V3 reCAPTCHA یا

معادل های دیگش هست.

- درسته که صفحات ورود، ثبت نام، **From Field** به حساب میان ولی چون مهم تر از بقیه جاها بودند سعی کردم جدآگاهه بهشون پردازم ولی به طور کلی هر جایی که **Form Field** بود و اطلاعاتی از کاربر دریافت میشد و این اطلاعات یا در پایگاه داده ذخیره و یا برای پردازش به وب سرور داده میشد بایستی ربات بودن یا نبودن یک کاربر رو بررسی کنیم . یعنی اینکه از **CAPTCHA** استفاده کنیم و نوع این کپچا بسته به فرمی که وجود داره میتونه متفاوت باشه . چه جاهایی منظورمه ؟ اگه توی وب اپلیکیشنتون یک فرمی دارید که کاربر میتونه نظر خودش رو ثبت کنه، یا فرمی که میتونه به شما تیکت بده و ... میتونن منسعد برخی حملات باشند و به همین خاطر نظر به اینه که حتما کاربر رو صحت سنجی کنید .
- ورود اولیه کاربر: چرا باید اولین ورود کاربر رو به اپلیکیشنمون بررسی کنیم و انسان بودن یک کاربر رو تایید کنیم ؟ علت اینه که حملات **DOS** یا **DDOS** با ارسال درخواست بسیار زیاد به اپلیکیشن انجام میشه و این درخواست ها از طرف انسان ارسال نمیشوند و ربات ها هستند که وظیفه ارسال انها رو به عهده دارند . از این رو پیشنهاد بر این هست که اولین ورود یک کاربر به وب اپلیکیشن رو بررسی کنیم، اون هم نه از طریق خود و ب اپلیکیشن بلکه از طریق یک **Reverse Proxy** مثل **WAF** . یک **WAF** میباشد که مابین کاربر و اپلیکیشن شماست باید درخواست کاربر در اولین ورود رو بررسی کنه و انسان بودن کاربر رو تایید کنه و سپس درخواست این کاربر رو به وب اپلیکیشن بفرسته و گرنه کاربر نباید اجازه دسترسی به وب اپلیکیشن داشته باشه و درصورتی که بتونه دسترسی داشته باشه، اپلیکیشن شما مستعد حملات **DOS** و **DDOS** هست .

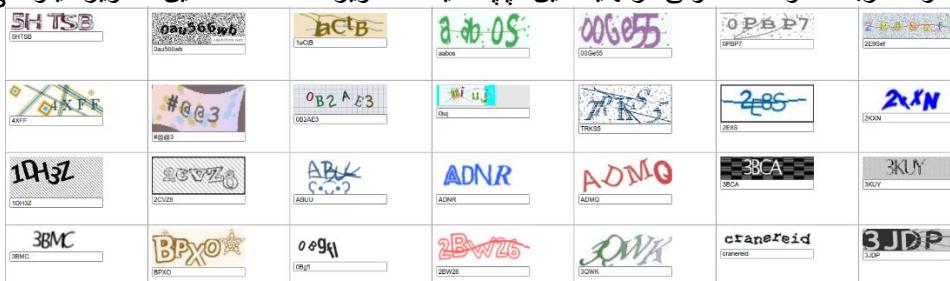


خب درمورد اینکه کپچا چیه و ... گفتیم که واسه امنیت بود ولی بریم سروقت اصل کار یعنی چطوری یک کپچا رو اسکل کنیم و دورش بزنیم ؟ ایا میشه ؟ مشکلات امنیتی کپچاها کجا هستند ؟ اگه توی صفحه **Testing for Weak Lock Out Mechanism** از **OWASP** برید درمورد کپچاها و مشکلات امنیتی اونها صحبت شده و به این طریق ما میتونیم بگیم که مشکلات امنیتی یک کپچا در دسته **Weak Lock Out Mechanism** قرار میگیره . طبق مقاله **OWASP** یک کپچا میتونه جلوی **Brute-Force** رو بگیره ولی مشکلات امنیتی خودش رو هم میتونه داشته باشه . در ادامه ما به روش هایی که میشه از طریقشون یک کپچا رو باپیس کرد صحبت میکیم .

چطوری میشه یک **CAPTCHA** رو باپیس کرد ؟ باپیس کردن کپچا از نظر من میتونه به دو روش انجام بشه که هرکدام از این دو روش خودش شامل روش های دیگه ای خواهد بود . این روش عبارت اند از :

1. **Image processing and OCR methods**
2. **CAPTCHA validation mechanism security flaws**

**OCR** و **Image processing** چیه و چطوری باهش میشه کپچا رو باپیس کرد ؟ میدونیم که کپچاهای سنتی در یک صفحه وب در یک تصویر نشون داده میشوند و توی این تصویر حروف و اعدادی رو کنار هم قرار داده اند و عملیات هایی روی شکل و ظاهر این حروف و اعداد اعمال شده است که توسط ربات خونده نشه ولی درنهایت این کپچاها یک تصویر هستند . حال این تصویر میتونه **png** یا **svg** یا ... باشه .



برای بایس کردن از این روش باید ابتدا تصویر CAPTCHA را پردازش کنیم. در صورت وجود هرگونه خر مگس معرکه توش، مثل خطوط اضافی، پس زمینه نویز دار، رنگ های مختلف و ... سعی کنیم حذف کنیم. برای اینکار باید از کتاب خونه های مختلف پردازش تصویر استفاده کنیم. کتابخونه هایی مثل OpenCV نوی پایتون و C++ برای پردازش تصویر بسیار مناسب هست. در حقیقت اصطلاح پردازش تصویر به معنی اعمال برخی از عملیات ها روی یک تصویر جهت بهتر کردن وضع تصویر یا استخراج اطلاعاتی از اون تصویر است. پس از پردازش تصویر باید سعی کنیم چیزی که توی تصویر نوشته شده رو استخراج کنیم. به این کار OCR گفته میشه که فقط 3 خط کد انجام شده:

```
GNU nano 7.2
from PIL import Image
import pytesseract

print(pytesseract.image_to_string(Image.open('./images/1.png')))
```

```
(.venv)kali㉿kali:~/Projects/pytesseract$ python main.py
2h697
```

توی تصویر زیر هم میبینید که از طریق همین 3 خط کد توانستیم که تمام تن یک تصویر رو با خطوطش به صورت دقیق بیرون بکشیم:

```
(.venv)kali㉿kali:~/Projects/pytesseract$ python main.py
```

Arial

**Lo**rem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.

```
(.venv)kali㉿kali:~/Projects/pytesseract$
```

اما توی تصویر زیر میبینید که به علت وجود خطوطی وسط متن، برنامه نتوانست متن داخل تصویر رو بخونه:

```

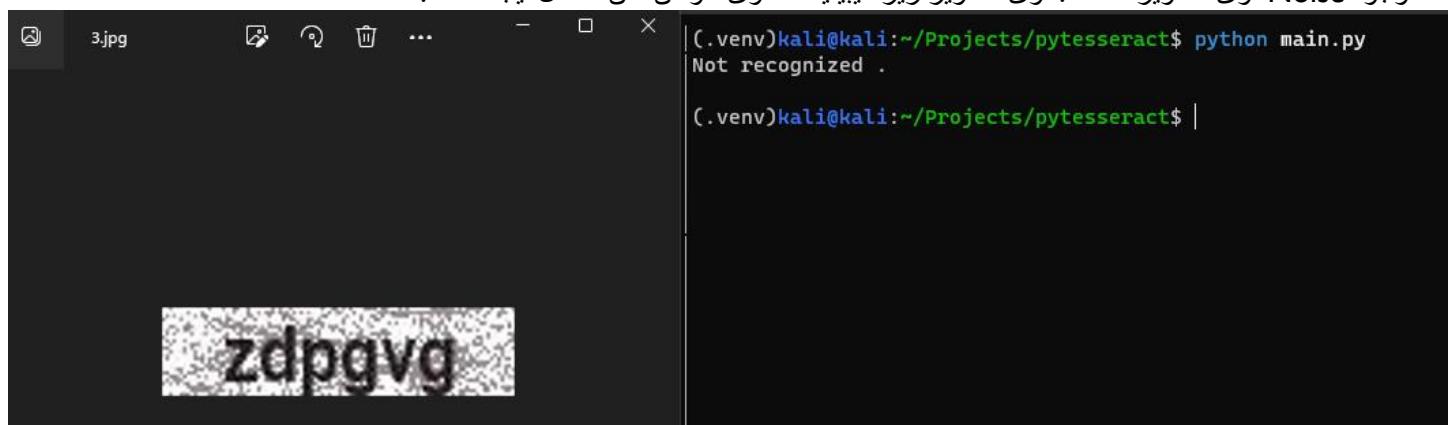
GNU nano 7.2
from PIL import Image
import pytesseract

output = pytesseract.image_to_string(Image.open('./images/2.png'))
if (output == ""):
    print("Not recognized .")
else:
    print(output)

```

(.venv)kali@kali:~/Projects/pytesseract\$ python main.py  
Not recognized .  
(.venv)kali@kali:~/Projects/pytesseract\$

این یکی از انواع خر مگن هاییست که میان توی معرفه و نمیزارن ما متن رو بخونیم. یکی دیگه از چیزهایی که ممکن هست مشکل ایجاد کنه وجود Noise توی تصویر هست. توی تصویر زیر میبینید که توی خوندن متن مشکل ایجاد شده:



خب تا اینجا نکته این بود که متنیم متن های خیلی ساده و بدون Noise و خطوط مخدوش کننده رو با استفاده از کتاب خونه Pytesseract بخونیم. چیزی که مهم هست اینه که شما شاید از هر 100 تا یک مورد کپچا ببینید که ساده و بدون خط و خش هست ولی بقیه حتماً توسط یک مواردی مخدوش شدن و به صورت عادی نمیشه توسط Pytesseract متن داخلش رو استخراج کنیم. برای این کار دو راه جلومن هست :

۱. از نرم افزار ها و ابزار هایی که وجود دارن استفاده کنیم و از طریق اونها سعی کنیم کپچا ها رو بخونیم.

۲. خودمون با استفاده از OpenCV و Tesseract-OCR یک اسکریپتی بنویسیم و سعی کنیم کپچا ها رو بخونیم.

اگه بخوایم گزینه دوم رو ببریم جلو کارمون سخته، باید اول بشینیم و پردازش تصویر رو یاد بگیریم، سعی کنیم وقتی یک تصویری رو از یک کپچا بدست میاریم، چیزهایی که مزاحمت ایجاد میکنن رو تشخیص و سپس حذف کنیم و در نهایت سعی کنیم یک تصویر ایجاد کنیم که فقط متن توشه و بعد از طریق Tesseract-OCR متن توش رو بخونیم. واقعاً اگه بخوایم چنین کنیم، برای هر پروژه ای باید به صورت جداگانه سر باپس کردن کپچا وقت بزاریم. این مورد رو من پیشنهاد نمیکنم بدون بررسی کردم ابزار ها و نرم افزار هایی که وجود دارن ببریم سراغش. اول تست کنیم و ببینیم که نرم افزار ها و ابزار هایی که برای خوندن کپچا نوشته شدن کار میکنند یا خیر؟ اگه دیدیم که اونها نتوانستن کار کنن و کپچا رو باپس، اوکی حله میریم و خودمون هم یه تلاشی میکنیم ولی اگه نتوانستن چه کاریه که خودمون رو اذیت کنیم؟

Rumla چه ابزاریه و چیکار میکنه؟ این ابزار در واقع یک افزونه مرورگر هست که میتوانید روی کروم نصب کنید. متاسفانه نسخه فایرفاکس نداره و فقط واسه کروم عرضه شده است. میتوانید از لینک زیر این افزونه به مرورگر خودتون اضافه کنید:

<https://chromewebstore.google.com/detail/rumola-bypass-captcha/bjjgbdlbgjeoankjjibmheneoekbghcg>



skipinput.com 2.8 ★ (664 ratings)

Extension

Accessibility

20,000 users

درمورد قدرت این ابزار بگم که برای حل کردن کپچا های خیلی ساده کاربرد داره و نمیتونه کپچا های پیچیده رو باپس کنه. پس از نصب کردن این افزونه میتوانید ازش استفاده کنید. وارد هر صفحه ای که بشید ابتدا سعی میکنه المتن کپچا رو پیدا کنه و سپس اون رو حل کنه:



اون عکس ربات کوچیکه یعنی اینکه فهمیده کپچا کجاست و کافیه که فرم رو پر کنید با برآتون کپچا رو پر کنه :



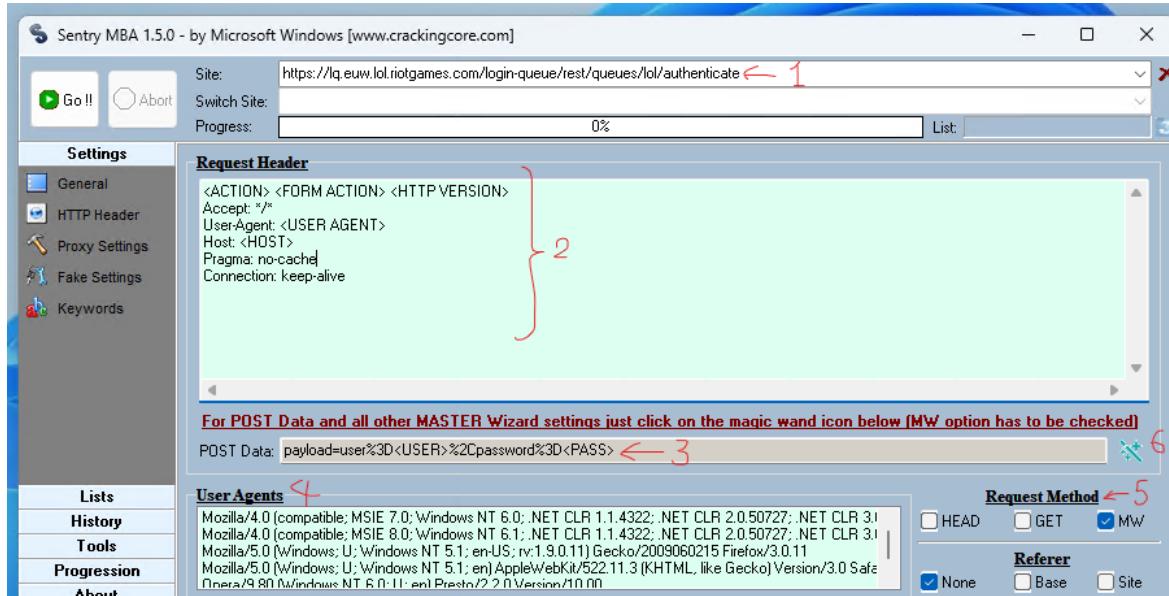
و خواهید دید که به شکل زیر اون رو وارد میکنه :



اما یک مشکلی وجود داره :)) این نکبت پولیه :(((((( اگه پولی نبود و اسه استفاده روزمره به نظرم گزینه مناسبی محسوب میشد ولی متاسفانه باید Credit بخri . اوکی بریم سروقت ابزار بعدی ...

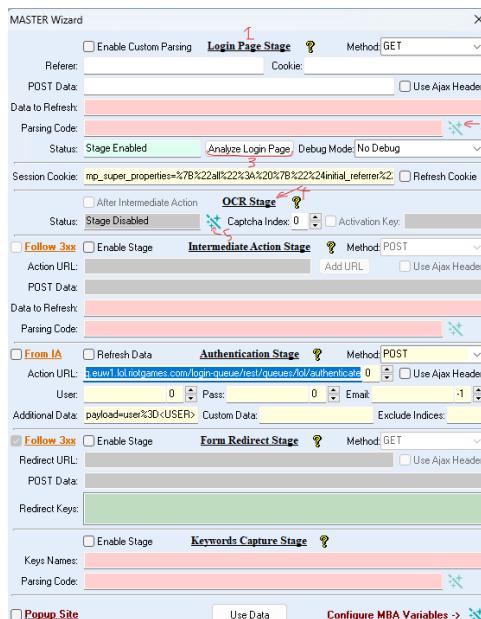
**Sentry MBA** چه ابزاریه ؟ این ابزار میگن یکی از قویترین ابزار هایی هست که واسه حملات Brute-Force استفاده میشه . البته من نمیگم ، کسانی که استفاده کردن میگن . امکاناتی که این ابزار در اختیار مهاجم قرار میده واقعاً زیاد ، مثلًا برآتون یک کپچا رو پردازش تصویر میکنه با تنظیماتی که خودتون بهش میگید ، اون کپچا رو حل میکنه ، ازتون لیست نام کاربری و رمز عبور و اسه حمله Brute-Force میگیره ، میتوانید لیست Proxy بھش بدید و حملتون رو طوری که میخواید انجام بدید . البته بگم که این ابزار قیمتیه ولی هنوز گاهی اوقات میتوانه کارایی خوبی داشته باشه . توی فکرشم از طریق OpenCV پایتون و Pytesseact سعی کنم یک ابزار مخصوص حل کپچا بنویسم ولی

نمیدونم کی چنین کاری رو انجام بدم . اگه دوست داشتید خودتون سعی کنید با ترکیب این دو یه چیزی شبیه به Sentry MBA بسازید . تصویر زیر یک نمایی از این ابزار هست :



نکته ای باید بگم اینه که پیشنهاد نمیکنم که این ابزار رو روی سیستم اصلیتون اجرا کنید . به علت قدیمی بودن این نرم افزار ، خیلی از جاها اون رو به بد افزار الوده کردن و برای دانلود قرار دادن و به همین خاطر پیشنهادم اینه که یا روی ماشین مجازی اجرا کنید و یا روی SandBox ویندوز تستش کنید . پنجره که توی تصویر بالا میبینید صفحه اصلی نرم افزار است و پیکر بندی های یک درخواست HTTP را میتوانید اعمال کنید . کار این نرم افزار اینه که درخواست های HTTP پشت سر هم برای بررسی اطلاعات کاربری رو ارسال کنه و شما میتوانید هدر ها و بدنه درخواست خودتون رو توی این صفحه بنویسید .

- قسمت شماره 1 : ادرس تارگتتون رو مشخص میکنید .
- قسمت شماره 2 : هدر های درخواستتون رو مینویسید .
- قسمت شماره 3 : بدنه و پارامتر های درخواستتون رو قرار میدید .
- قسمت شماره 4 : User-Agent درخواستتون رو تعیین میکنید .
- قسمت شماره 5 : نوع درخواستتون رو مشخص میکنید .
- قسمت شماره 6 : در صورتی که نوع درخواست MW باشه این گزینه فعال خواهد بود و میتوانید تنظیماتی بیشتر اعمال کنید که در تصویر زیر میبینید .



توی پنجره بالا هم تنظیماتی داریم که نمیخوام همه رو توضیح بدم ولی به صورت گذرا از برخیشون میگذریم :

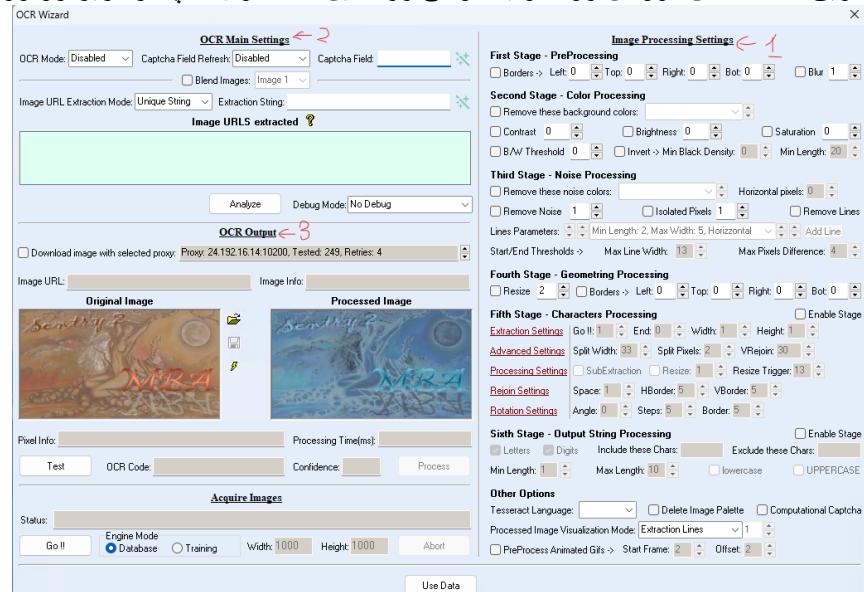
قسمت شماره ۱ : تنظیمات مربوط به صفحه لایکین خواهد بود . میتوانید مولفه ها را مشخص کنید . اون گرینه Parsing رو فعال کنید تا صفحه رو تحلیل و تجزیه کنه .

قسمت شماره ۲ : در صورتی که روی دکمه قسمت شماره ۳ کلید کنید، این دکمه فعال خواهد شد و برخی از مولفه های Login Stage رو پر میکنه .

قسمت شماره ۳ : این دکمه رو که بزنید صفحه Login رو تحلیل میکنه و المنت های داخل رو بررسی میکنه و برخی از اطلاعات مورد نیاز رو بدست میاره .

قسمت شماره ۴ : مربوط به خوندن متن تصویر کپچا هست .

قسمت شماره ۵ : وقتی رو این دکمه کلیک کنید وارد صفحه پردازش تصویر کپچا میشه و میتوانید تصویر کپچا رو مشخص کرده و سپس سعی کنید از طریق OCR متن درونش رو بخونید . وقتی روی این دکمه بزنید با پنجره زیر روبرو میشوید :



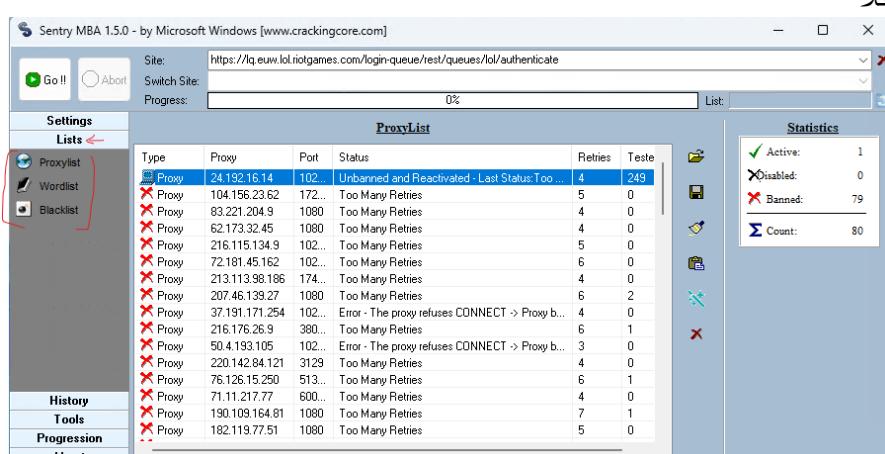
پنجره بالا از قسمت های زیر تشکیل شده است :

قسمت ۱ : تنظیمات مربوط به پردازش تصویر می باشد .

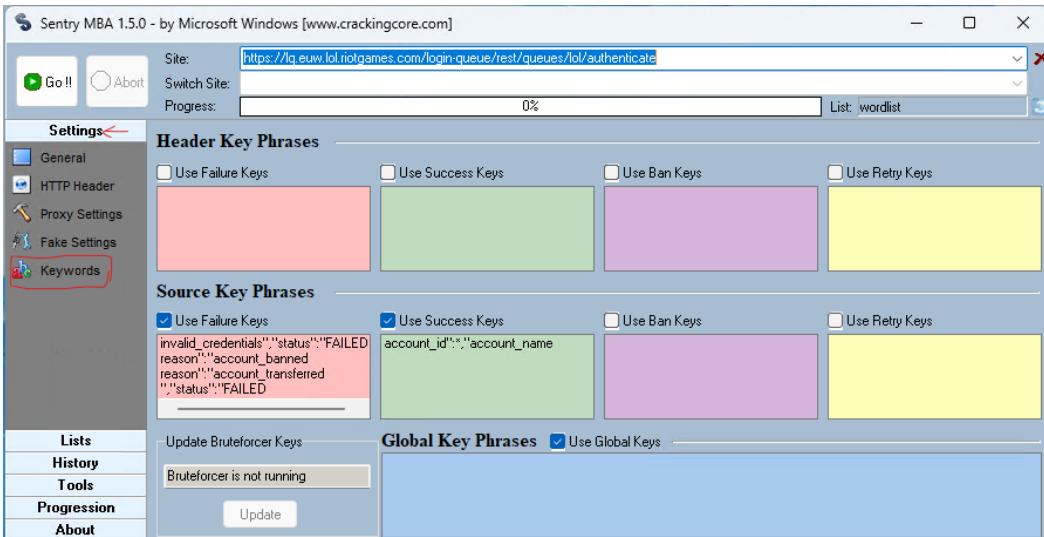
قسمت ۲ : تنظیمات مربوط به OCR می باشد .

قسمت ۳ : خروجی OCR و پردازش تصویر رو در این قسمت داریم .

دقت کنید که وقتی یک تارگت رو مشخص میکنید سعی میکنه تصاویری که احتمال داره کپچا باشه رو توی لیست Image URLs extracted قرار بده . شما میتوانید با دوتا کلیک اون رو انتخاب کنید و بعد با تنظیمات Image Processing Settings اون رو تغییراتی بدهید که بتونه بخونه . بدین شکل شما میتوانید یک کپچا رو بخونید . پس از خوندن کپچا باید تنظیمات مربوط به حمله Brute-Force خودتون رو توی Screenshot MBA انجام بدهید مثلا



از منوی بغل گزینه لیست میتوانید لیست پراکسی ها، نام های کاربری و کلمات عبور و Blacklist رو تنظیم کنید . یکی دیگه از تنظیمات مهم که باید اعمال کنید مربوط به Keyword هاست . باید از منوی سمت چپ گزینه Settings رو انتخاب کنید و سپس رو Keywords بزنید و صفحه زیر رو خواهد دید :



نوى اين قسمت ميتوانيد به نرم افزار بگيد که وجود چه کلماتي در پاسخ به معنى اشتباه بودن نام کاربر است و وجود چه کلمه اي نشونه اشتباه بودن رمز عبور است و همچينين برash تعين کنيد که در صورت وجود فلان کلمه به معنى اشتباه بودن Captcha است. قاعدها که اين را 100 درصد صحيح نميتوانه حل کنه و ممکن هست برخى از اون اشتباه از اب در بياad پس ميتوانيد بهش بفهمونيد که كچاي اشتباه چطور يه و در نهايتي پس از اعمال تمام چيز هاي خواستيد و تنظيمات حملتون روی دكمه !! Go بالا سمت چپ بزنيد تا حمله رو شروع کنه . حقيقتها من نميخوايم در مرور اين نرم افزار امورش کاملی ارائه کنم و مهم ترين دليلم هم اينه که خودم هم بلد نیست ولی خوب تا جايی که تو نستم توضيح دادم . حالا خودتونيد و Sentry MBA و تنظيمات زيادش . بريدي انگولکش کنيد و ببینيد که چي بر چие .

جايگزین هايي جيذا برای Sentry MBA نوشته شده اند که گفتن اسمشون زياد هم بد نیست ، Openbullet , Silverbullet , Cyberbullet . سه تا از نرم افزار هايی هستند که اومند و سعي کردن کاري که Sentry MBA ميکنه رو انجام بدن و در برخى از کارها موفق بودن ولی در برخى ديگه ضعيف تر عمل کردن . دوست داشتيد يه نگاهي هم به اينا بندازيد . کار اصلاحیون انجام حملات Brute-Force هست که با تنظيمات خوب و به صورت درست و حسابي انجام مиде . ازشون برای کاراي خراب کاري و زدن جيپ مردم هم استقاده ميکنن و معمولاً پايه ثابت حملات Cracking و Carding به حساب ميان که غير قانوني و غير اخلاقي هست . اگاهي ازشون خوبه برای اينکه دچارش نشيم .

اگه يه وقتی ديديم که يه کچا هست که نميشه با Sentry MBA و Rumla و ... باليپيش کرد باید بريم سروقت ابزار هايي که باید خودمون طراحیشون کنیم . همون چيزايی که گفتم ، ابتدا پردازش تصویر کنیم و نويز ها خطوطی که متن توی کچا رو مخدوش ميکنن رو برداريم و بعد هم از طریق Tesseract سعی کنیم محتواي کچا رو بخونیم . تصویر زير يك نمونه از ابزار خوندن کچا از طریق پایتون هست :

```

1  from PIL import Image
2  from scipy.ndimage import gaussian_filter
3  import numpy
4  import pytesseract
5  from PIL import ImageFilter
6
7  def solve_captcha(filename):
8      # threshold on the first stage
9      th1 = 140
10     th2 = 140 # threshold after blurring
11     sig = 1.5 # the blurring sigma
12     from scipy import ndimage
13
14     original = Image.open(filename)
15     original.save("original.png") # reading the image from the request
16     black_and_white = original.convert("L") # converting to black and white
17     black_and_white.save("black_and_white.png")
18     first_threshold = black_and_white.point(lambda p: p > th1 and 255)
19     first_threshold.save("first_threshold.png")
20     blur = numpy.array(first_threshold) # create an image array
21     blurred = gaussian_filter(blur, sigma=sig)
22     blurred = Image.fromarray(blurred)
23     blurred.save("blurred.png")
24     final = blurred.point(lambda p: p > th2 and 255)
25     final = final.filter(ImageFilter.EDGE_ENHANCE_MORE)
26     final = final.filter(ImageFilter.SHARPEN)
27     final.save("final.png")
28     number = pytesseract.image_to_string(Image.open('final.png'), lang='eng',
29                                         config='--psm 10 --oem 3 -c tessedit_char_whitelist=0123456789').strip()
30
31     print("RESULT OF CAPTCHA:")
32     print(number)
33     print("*****")
34
35     return number

```

قسمتی که با شماره 1 مشخص شده ، كتاب خونه های مختلف رو به پروژه اضافه کرده و قسمت 2 مربوط به پردازش تصویر اين اسکريپت هست و در نهايتي قسمت 3 هم مربوط به خوندن متن تصویر پردازش شده هست که با Pytesseract انجام ميشه . شما اگه ميخوايد يك اسکريپت بنويسيد باید حداقل اين سه مرحله رو داشته باشيد و ميتوانيد ما بين اين سه مرحله ، ماشين لرنينگ رو هم پياده سازي کنيد تا هر کچايی که حل ميکنه ، ازش يك اطلاعاتي هم بدست بياره و ياد بگيره .

پروژه زیر هم یک پروژه تمیز جهت خوندن کپچا هست که از طریق ماشین لرنینگ و چنین چیزهایی کار میکنه، میتوانید کدهاش رو بررسی کنید و از ماهیت کارش سر در بیارید :

<https://github.com/JasonLiTW/simple-railway-captcha-solver.git>

پروژه زیر هم با پایتون نوشته شده و میتوانید ازش استفاده کنید .

<https://github.com/ptigas/simple-captcha-solver>

تا دلتون بخواهد توی گیت هاب از این پروژه ها نوشته شده و میتوانید یه جستجوی درست و درمون انجام بدید و به کلی از پروژه های موجود دسترسی بگیرید .

من خودم هم به ساده ترین روش ممکن سعی کردم یک پروژه بیارم بالا (توی مثال های قبلی دیدیمش) . کد زیر فقط یک تصویر رو از طریق کتابخونه Pillow باز میکنه و به Pytesseract میده و همین چند خط کد ساده، میتوانه برخی از کپچا های خیلی خیلی ساده رو حل کنه :

```
main.py
1  from PIL import Image
2  import pytesseract
3
4  output = pytesseract.image_to_string(Image.open("./images/25.png"))
5  if (output == ""):
6      print("Not recognized .")
7  else:
8      print(output)
9
```

کافیه که تصویر کپچای مورد نظرتون رو به جای ./images/25.png/. قرار بدید و اسکریپت رو اجرا کنید . اگه تونست بخونه برآتون روی صفحه نشونش میده و اگه نتوانست عبارت Not recognized رو خواهد دید . کافیه که قیل از داده تصویر به pytesseract پردازش هایی رو روی تصویر انجام بدید و تصویر کپچا رو واسه pytesseract شفاف تر کنید تا راحت تر بتونه عبارت داخلش رو بخونه .

خب، اینا از قسمت CAPTCHA validation بودن و قسمت بعدی میریم سروقت mechanism security flaws و با هم مشکلات و نقص های امنیتی کپچا رو بررسی میکنیم . گاهی اوقات ممکن هست که یک کپچا وجود داشته باشه، طوری هم باشه که نشه از طریق ابزار و خوندنش باپیش کرد ولی میشه از طریق مشکلاتی که توی مکانیزمش وجود داره ازش گذر کنیم . باید یک کپچا رو از چند جهت بررسی کنیم .

- 1. Captcha value in the response
- 2. Invalid Captcha value works
- 3. NULL Captcha parameter value works
- 4. Removing Captcha parameter makes it works
- 5. Using valid Captcha value multiple times
- 6. ...

این مشکلات به شکلی Business Logic محسوب میشن و به همین خاطر حس میکنم تعدادشون بیشتر از این حرفاست و میشه روش های مختلف دیگه ای هم برآش نام برد . تا اینجا من همینا رو پیدا کردم . حالا بریم کمی درمورد هرکدو مشون صحبت کنیم .

Captcha value in the response یعنی چی ؟ از معنیش مشخصه که یعنی مقدار درست کپچا توی پاسخی که تو ش درخواست کپچا میکنیم وجود داره . گاهی ممکن هست که یک برنامه نویس به اشتباہ زمانی که کپچا رو میسازه پاسخ اون کپچا رو هم به سمت ما بفرسته، باید درخواست کپچا رو بررسی کنیم و شاید شامل پاسخ باشه . شاید کم پیش بیاد ولی به این معنا نیست که پیش نمیاد . یه نوع کپچا هست که ممکن هست زیاد بینیش به نام SVG-Captcha که یک کتابخونه جاواسکریپتی داره که با لینک زیر میتوانید دانلود کنید :

<https://www.npmjs.com/package/svg-captcha>

این کپچا خیلی جالبه، وقتی درخواست کپچا رو بررسی کنید خواهد دید که یک id و یک مقدار svg داره و اون مقدار svg رو توی صفحه برای شمارندر میکنه که یک کپچا هست . من وقتی دیدم که با svg این کپچا ساخته شده به کم شاخکام تكون خورد . svg یک تگ html هست که برای نشون دادن تصاویر استفاده میشه . کپچا شکلی به شکل زیر داره :



میتوانید خطوط رو اضافه کنید و حتی رنگ بندی هم بهش بدید . ممکن هست که فکر کنید اینو دیگه نمیشه خوند ولی با بررسی که من توی تک svg که بر میگیردونه انجام دادم به این نتیجه رسیدم که جواب کپچا توی خود svg هست، البته نه صورت مستقیم بلکه به صورت کاملاً مخفی و غیر مستقیم . یه اسکریپت نوشتم که اینو حل میکنه .

بریم یه نگاهی به چیزی که به کاربر میده بندازیم . کپچا رو در قالب یک JSON به شما میده به شکل زیر :

```
{
  text: 'a',
  data: '<svg xmlns="http://www.w3.org/2000/svg" width="150" height="50" viewBox="0,0,150,50"><path fill="#444" d="M70.19 28.88L70.29 28.98L70.15 28.84Q67.74 28.98 66.71 30.16L66.72 30.17L66.61 30.06Q65.64 31.38 65.55 33.93L65.71 34.00Q65.56 38.94 70.16 38.75L70.26 38.85Q70.23 38.81Q72.28 38.96 73.63 37.40L73.69 37.46L73.67 37.45Q75.04 35.98 74.81 33.85L74.90 33.93L74.85 33.88Q74.80 31.28 73.93 30.25L73.84 30.17L73.85 30.18Q72.45 28.93 70.24 28.93ZM75.25 38.69L75.22 38.67L75.18 38.63Q74.05 41.16 69.37 41.16L69.30 41.08L67.94 41.18L67.95 41.18Q65.63 41.23 64.53 40.08L64.51 40.06L64.56 40.12Q63.48 38.96 63.14 36.55L63.12 36.55Q62.94 35.03 62.86 33.24L63.05 33.43L63.04 33.42Q62.94 32.37 62.79 30.43L62.70 30.36Q62.64 26.09 67.81 26.09L67.74 26.01L69.46 26.17L69.44 26.15Q72.37 26.19 74.47 27.48L74.33 27.35L74.48 27.42Q75.24 27.95 75.73 28.64L75.74 28.64L75.70 28.61Q75.95 27.79 76.49 26.15L76.51 26.18L76.41 26.07Q77.68 25.92 25.25L79.95 25.28L80.81 25.33Q77.52 29.66 77.52 34.91L77.34 34.34Q77.45 38.58 78.82 41.66L78.81 41.65L78.75 41.59Q77.68 41.35 75.77 41.12L75.81 41.16L75.74 41.09Q75.63 40.88 75.32 38.77ZM78.68 43.96L78.61 43.96L80.87 44.58L80.78 44.42Q82.04 44.84 83.03 45.30L82.97 45.24L83.02 45.29Q79.41 48.99 79.41 34.90L79.38 34.87L79.32 34.81Q79.24 38.20 81.37 26.40L81.54 26.56L81.49 26.50Q80.82 26.72 79.49 27.10L79.56 27.17L79.58 27.19Q79.87 26.38 80.74 24.85L80.73 24.83L80.62 24.73Q80.28 25.07 76.20 25.91L76.95 25.76L75.58 27.68L75.48 27.58Q73.76 25.98 69.35 25.68L69.37 25.70L69.33 25.66Q68.26 25.73 66.01 25.88L66.06 25.94L66.09 25.96Q62.33 26.23 62.44 30.27L62.34 30.17L62.52 30.35Q62.56 33.59 63.13 38.04L63.18 38.08L63.24 38.51L63.20 39.25 64.00 40.36L64.15 40.50L64.03 40.38Q64.17 40.52 64.36 40.64L64.32 40.60L64.48 40.67Q64.64 41.15 65.02 41.72L65.08 41.78L65.09 41.79Q66.44 44.23 69.83 43.44L69.91 43.52L69.87 43.49Q71.57 43.66 71.72 43.66L71.54 43.48L73.11 43.49L73.14 43.51Q76.49 43.47 77.71 42.11L77.61 42.01L77.69 42.09Q77.95 42.61 78.55 43.91L71.95 31.10L72.12 31.26L72.03 31.17Q73.35 31.17 74.19 31.62L74.29 31.72L74.17 31.61Q74.43 32.36 74.43 33.88L74.44 33.85L74.63 35.37 72.73 33.35 37.15L73.33 37.32L73.34 37.14Q72.18 38.67 70.27 38.48L70.24 38.45L70.33 38.54Q68.87 38.52 67.84 38.10L67.78 38.35L67.70 37.96Q67.56 37.18 67.56 35.96L67.49 35.89L67.53 35.92Q67.65 32.40 70.28 31.45L70.21 31.37L70.26 31.43Q71.27 31.22 2.07 31.22Z"/><path d="M1 4 C63 21,59 45,133 45" stroke="#444" fill="none"/></svg>'
```

اون پاسخ text data تصویر Captcha هست . گاهی میان و یک id هم بهش اضافه و توی پایگاه داده ذخیره میکن . چیزی باید به سمت کاربر ارسال کنه قطعاً بايد شامل data باشه و رو نداشته باشه چون جواب Captcha هست ولی به اشتباه جاهایی دیده شده که در پاسخ text رو هم فرستادن اما مهم نیست که text که پاسخ کپچا هست بیاید یا نه . شما میتوانید از طریق svg باید و Captcha رو بدمید بیارید . اگه SVG رو بفهمید بدمید که تمام رسم شکل SVG توی یک Attribute svg باید و Captcha رو بدمست بیارید . این نام d انجام میشه . شما اگه مقدار این Attribute را استخراج کنید و تمام فاصله ها ، اعداد و کاراکتر ". " رو ازش حذف کنید یک عبارت شامل ... MLQ... خواهد داشت . ترتیب قرار گیری این حروف برای هر کاراکتر منحصر به فرد است و از این طریق میتوانید مقدار رسم شده رو بدمست بیارید . این یک تمرین و اسه خودتون ، برید svg-captcha رو نصب کنید و تمام کاراکتر ها رو تولید کنید ، بعد بیاید و یک اسکریپت بنویسید و تمام MLQ... ها بدمست بیارید و سپس سعی کنید کپچا رو ایجاد کنید و حل کنید . البته یه راهنمایی دیگه هم بکنم ، جایگاه حروف کپچا ها با SVG ها متفاوت هست یعنی ممکن هست اولین SVG نشون دهنده اخرين یا دومين یا ... کپچا باشه . باید این مورد رو هم حل کنید .

Invalid Captcha value works چیه دیگه ؟ ممکن هست که اصن برنامه نویس Backend مقدار Captcha رو بررسی نکنه ، این مشکل ممکن هست که وجود داشته باشه و باید تست بشه . شرایط اینطوریه که وارد کردن مقدار اجباری هست ولی برنامه نویس نیومده و مقدار وارد شده Captcha رو با مقداری که داره مقایسه کنه و گفته همین که هست یعنی کاربر وارد کرده . در این صورت ما میتوانیم بیایم و یک مقدار تصادفی رو توی هر درخواستی که به سمت تارگت میفرستیم قرار بدیم ، با اگاهی از اینکه این مقدار بررسی نمیشود . اما نکته ای که مهمه سعی کنید مقداری که به جای مقدار کپچا قرار میدید تعداد کاراکتر های یکسانی با مقدار کپچا داشته باشه

NULL Captcha parameter value works یعنی چی ؟ یعنی اینکه در صورتی که پارامتر Captcha رو حتی خالی هم بررسیت جواب میگیرید . این موارد ممکن هست که کم پیش بیاید ولی احتمالش صفر نیست و برنامه نویس ممکن هست دچار چنین خطاهایی بشه . شما به عنوان یک Pentester باید این موارد رو بررسی کنید . باید توی درخواستی که به سمت وب اپلیکیشن می فرستید بیاید و مقدار خالی Request رو تست کنید . ممکن هست که خالی نبودن مقدار کپچا توی Front-End بررسی بشه به همین خاطر میتوانید توی مقدارش رو خالی کنید .

از Remove Captcha Parameter makes it works منظور مون چیه ؟ در برخی وب اپلیکیشن ها ممکن هست که دیده باشید ، گاهی در ابتدا کپچا رو به شما نشون نمیدن و بعد از چند بار اشتباه زدن یک کپچا رو برآتون قرار میدن که باید حل بشه تا بتونید ادامه بید . گاهی برنامه نویس او مده و بر اساس وجود یا عدم وجود پارامتر Captcha توی درخواست تصمیم گرفته که ایا مقدار کپچا رو بررسی کنه یا خیر ؟ در این صورت شما اگه پارامتر کپچا رو در صورت وجود حذف کنید ، برنامه نویس مقدار اون رو در نظر نمیگیره و شما میتوانید کپچا رو با پیس کنید .

Using valid Captcha value multiple times چیه ؟ دیده شده که در برخی موارد میشه یک مقدار Valid رو چندین بار استفاده کرد . چنین موقعیتی در چه زمانی بوجود میاد ؟ زمانی که برنامه نویس یک کپچا رو تولید و توی پایگاه داده اون رو ثبت کرده باشه و وقتی شما مقدار کپچا رو توی درخواستتون می فرستید ، برنامه نویس از طریق یک id کپچا شما رو تشخیص میده و میره توی پایگاه داده اون رو بیرون میاره و مقدار شما رو با مقدار ثبت شده توی پایگاه داده قیاس میکنه ، در صورتی که یکی بود به شما اجازه ادامه داده و در غیر این صورت خطای کپچا رو برای شما نشون خواهد داد . برنامه نویس بعد از اینکه کپچای وارد کاربر رو توی پایگاه داده بپیدا کرد باید کپچای توی پایگاه داده رو حذف یا غیر قابل استفاده کنه و گرنه مهاجم میتوانه از id و مقدار این کپچا هر وقت که دوست داشت بهره برداری کنه . این مشکل رو من خودم توی یکی از سامانه های یکی از شرکت های بزرگ پیدا کرد و گزارش دادم ، درسته بانتی نمیدادن ولی واسه خود شیرینی گزارش کرد .

یه نکته به نظرم حائز اهمیت هست، زمانی که شما توی درخواست کپچا یک پارامتری به شکل `id` میبینید که واسه هر کپچایی منحصر به فرد هم هست میتوانید به این فکر کنید که کپچا توی پایگاه داده ای یا فایلی یا ... ذخیره میشه که بعد از ارسال پاسخ از طریق همین `id` قیاس صورت میگیره . در این نوع بررسی ها که با `id` انجام میشه، **Using valid captcha multiple time** شاید بشه انجام داد . همچین همین پارامتر `id` میتوانه منجر به **SQLi** هم بشه چون بعدا میره و توی پایگاه داده جستجو میشه . پس باید دقت کرد .

نکته دیگه ای که درمورد کپچا باید در نظر بگیریم، این کپچاهای عجیب و غریبی هست که توسط برنامه نویس ایجاد میشن . مثلا یک کپچا هست که یک عکسه و در حقیقت به صورت تصادفی از میان تعداد بسیار زیادی عکس انتخاب شده و به سمت کاربر او مده . این تصاویر تعدادشون محدوده و میشه از طریق یک اسکریپت به تمامشون دسترسی پیدا کرد . میتوانیم که هر فایلی برای خودش یک مقدار هش خاص داره که به جز زمانی که فایل تغییر کرده باشه همیشه یکسان هست . میتوانیم تمام تصاویر کپچاها را بدست بیاریم و از طریق هششون اونها رو شناسایی کنیم و سپس بشینیم و مقدار کپچای توشون رو حل کنیم . اینطوری میتوانیم به یک پایگاه داده بزرگ از تمام کپچاهای تارگت با مقدار حل شدشون دست پیدا کنیم و سپس وقتی میخواستیم حمله رو پیاده سازی کنیم، هر وقت کپچا رو گرفتیم بیایم و هشش رو بدست بیاریم و از طریق این هش توی پایگاه داده خودمون بفهمیم کدام تصویر هست و سپس مقدارش رو بهش بدیم . بعضی موارد هم کپچا یک عبارت ریاضی هست که گفته "7 منهای 12 چی میشه؟" چنین چیزهایی خیلی راحتتر بایپس میشن و باید فقط خلاقیت به خرج بدیم . اینجا اعلام میکنم که **Captcha** هم توم شد و به نظرم موضوع جذابی بود و یه ایده هایی واسه برخی ابزار ها توی ذهن او مد که دارم روشون کار میکنم، یه چیزی مثل **Sentry MBA** . برمی سر وقت موضوع بعدی ...

## Weak Password & Password Attack

یک رمز به یاد سپردنی هست که از رشته‌ای از کاراکتر‌های حروفی، عددی و ... تشکیل می‌شود و جهت تایید هویت یک کاربر بکار می‌رود. {ویکی‌پدیا} طبق دسته‌بندی‌های WSTG قسمت **Testing for Weak Password** جزو دسته **Weak Password Policy** هست و **Password Attack** جزو دسته **Testing for Weak Lock Out Mechanism** محسوب می‌شود. در اینده درمورد **WSTG** صحبت می‌کنم. تقریباً توی جزوای آخر این دوره بهش می‌پردازم.

رایج ترین و راحت ترین روش مدیریت مکانیزم احراز هویت در حال حاضر استفاده از پسورد‌های ثابت می‌باشد، البته ناگفته نمونه که **OTP** ها هم در حال گسترش هستند. با این وجود هنوز در اسناد فاش شده‌ای وجود داره مشخص هست که کاربران توجهی به استفاده از کلمات عبور پیچیده ندارند و همیشه اسون ترین عبارات رو به عنوان کلمه عبور خودشون در نظر می‌گیرند. مثلاً در این اسناد مشخصاً گفته شده که سه تا از رایج ترین کلمات عبور استفاده شده، **qwerty**, **password**, **123456** هست.

## Password Strength

قدرت یک کلمه عبور به معنی اندازه گیری تاثیر پذیری یک کلمه عبور نسبت به حدس زده شدن و حملات **Brute-Force** هست. به عبارت دیگه به این معناست که، ارزیابی می‌کنه که چقدر از مایش نیاز هست که یک مهاجمی که به کلمه عبور مورد نظر دسترسی مستقیم نداره بتونه اون کلمه عبور رو حدس بزنه.

- **Strength** یک کلمه عبور ارتباط مستقیم با طول، پیچیدگی و غیر قابل پیش‌بینی بودن اون کلمه عبور داره.
- طول یک کلمه عبور : حداقل طول پیشنهادی برای یک کلمه عبور 8 کاراکتر هست. این درحالیه که اینستاگرام به این بزرگی 6 کاراکتر هم قبول می‌کنه.

- پیچیدگی یک کلمه عبور : یک کلمه عبور باید شامل حروف، اعداد و سیمبول‌های مختلفی مثل ... (#\$%^&\*!@#)! باشه.

غیر قابل پیش‌بینی بودن کلمه عبور : بسیاری از موقع، افراد کلمه عبور خود رو از عباراتی که در ذهن دارند می‌گیرند، خیلی از ماها برخی از کلمات عبور مون شماره کاملی مون هست و برخی دیگه ممکن هست ترکیبی از اسمش و تاریخ تولدشون باشه. این به این معناست که یک کلمه عبور که چنین باشه قابل پیش‌بینی هست و یک مهاجم احتمال بیشتری داره که بتونه این کلمه عبور رو حدس بزنه و قاعده‌تا یک کاربر نباید کلمات عبور خودش رو به این شکل ایجاد کنه.

...

وقتی موارد بالا رو در یک کلمه عبور رعایت کنیم، **Password Strength** اون کلمه عبور زیاد می‌شود و احتمال اینکه قابل حدس زدن باشه کاهش می‌پابد.

## Password Guessing

حدس زدن یک کلمه عبور یعنی چی؟ گفتیم که گاهی برخی از افراد کلمات عبور خودشون رو از عبارات اطرافشون می‌سازند. مثلًا کلمه عبور یک نفر می‌توانه نام کوچکش به علاوه چهارتا شماره اخر شماره موبایلش باشه و یا یکی دیگه میاد و کلمه عبورش رو مجموعه نام کوچکش و تاریخ تولد شمسی یا میلادی خودش قرار میده. انتخاب پسورد و کلمات عبور خودتون بدین شکل احتمال **Password Guessing** را افزایش میده. در صورتی که یک مهاجم شماره از عنوان تارگت خودش در نظر بگیره، قاعده‌تا در مرحله **Reconnaissance** اطلاعاتی از شما بدست خواهد اورد و سعی می‌کنه از طریق این اطلاعات و ابزار هایی مثل **CUPP** یک لیست کلمات عبور از شما تهیه کنه. قاعده‌تا این کلمات عبور رو روی حساب‌های کاربری مختلف شما بررسی می‌کنه و در صورتی که کلمه عبور شما از اطلاعات خودتون ایجاد شده باشه، احتمال پیدا کردن کلمه عبور شما زیاد خواهد شد.

## CUPP

**CUPP** یا **Common User Passwords Profiler** یک ابزار است که توسط زبان برنامه نویسی پایتون توسعه داده شده است. این ابزار از شما چند سوال اساسی درمورد هدفون می‌پرسه مثلًا ... **Name**, **Company Name**, **Partner's Name**, **Pet Name**, **Victim** و سپس با ترکیب این کلمات سعی می‌کنه یک لیست از کلمات عبور احتمالی را بسازه. مهاجم می‌توانه از این لیست کلمات عبور و همچنین نامهای کاربری که **CUPP** می‌سازه برای انجام **Password Attack** استفاده کنه. برای دریافت این ابزار می‌توانید به لینک گیت هایش به ادرس زیر بروید:

<https://github.com/Mebus/cupp>

این ابزار توی کالی لینوکس به صورت پیش فرض نصب هست و کافیه که دستور **cupp** رو توی ترمینال وارد کنید. اگه هم نصب نبود می‌توانید از طریق دستور زیر اون نصب کنید:

```
# sudo apt install cupp
```

سپس با دستور زیر **cupp** رو اجرا کنید:

```
# cupp
```

```

cupp.py!
    \          # Common
     \         # User
      \        # Passwords
       \       # Profiler
        \|---* [ Muris Kurgas | j0rgan@remote-exploit.org ]
           [ Mebus | https://github.com/Mebus/]

usage: cupp [-h] [-i] [-w FILENAME | -l | -a | -v] [-q]

Common User Passwords Profiler

options:
-h, --help      show this help message and exit
-i, --interactive  Interactive questions for user password profiling
-w FILENAME   Use this option to improve existing dictionary, or WyD.pl output to make some pwnsauce
-l             Download huge wordlists from repository
-a             Parse default usernames and passwords directly from Alecto DB. Project Alecto uses purified databases of Phenoevit and CIRT which were merged and enhanced
-v, --version   Show the version of this program.
-q, --quiet     Quiet mode (don't print banner)

```

میبینید که گروهی از Option ها رو داره . با سوئیچ -i یا --interactive میتوانید از برنامه استفاده کنید .

```

kali㉿kali:~$ cupp -i
cupp.py!
    \          # Common
     \         # User
      \        # Passwords
       \       # Profiler
        \|---* [ Muris Kurgas | j0rgan@remote-exploit.org ]
           [ Mebus | https://github.com/Mebus/]

[+] Insert the information about the victim to make a dictionary
[+] If you don't know all the info, just hit enter when asked! ;)

> First Name: Ahmad
> Surname: Razeghi
> Nickname:
> Birthdate (DDMMYYYY):

> Partners) name:
> Partners) nickname:
> Partners) birthdate (DDMMYYYY):

> Child's name:
> Child's nickname:
> Child's birthdate (DDMMYYYY):

> Pet's name:
> Company name:

> Do you want to add some key words about the victim? Y/[N]: Y
> Please enter the words, separated by comma. [i.e. hacker,juice,black], spaces will be removed: programmer,hacker,blackhat
> Do you want to add special chars at the end of words? Y/[N]:
> Do you want to add some random numbers at the end of words? Y/[N]:
> Leet mode? (i.e. leet = 1337) Y/[N]: Y

[+] Now making a dictionary...
[+] Sorting list and removing duplicates...
[+] Saving dictionary to ahmad.txt, counting 404 words.
[+] Now load your pistolero with ahmad.txt and shoot! Good luck!

```

میبینید که تعدادی سوال از من پرسیده و من جواب دادم و برام یک لیستی از 404 کلمه در یک فایل به نام ahmad.txt ساخته .

```

kali㉿kali:~$ head -n 20 ahmad.txt
4hm4d2008
4hm4d2009
4hm4d2010
4hm4d2011
4hm4d2012
4hm4d2013
4hm4d2014
4hm4d2015
4hm4d2016
4hm4d2017
4hm4d2018
4hm4d2019
4hm4d2020
4hm4d_
4hm4d_2008
4hm4d_2009
4hm4d_2010
4hm4d_2011
4hm4d_2012
4hm4d_2013

```

این ابزار یک فایل داره به نام cupp.cfg که توی دایرکتوری /etc/ قرار داره و پیکربندی هایی رو توی خودش نگهداری میکنه که توی ساخته لیست های کلمات عبور دخیل هستند . مثلا قسمت زیر توی این فایل عدد سالهایی رو به ته برخی از کلمات عبور اضافه میکنه و میتوانید تغییرش بدهید :

```
[years]
years = 2008,2009,2010,2011,2012,2013,2014,2015,2016,2017,2018,2019,2020
```

یا قسمت زیر مربوط به LEET کردن کلمات عبور هست به این معنا که کلمات بساز که و به جای A عدد 4 و به جای Z مثلا عدد 2 رو قرار بده . به این کار میکن LEET یا 1337 . این کار رو معمولا هکر ها انجام میدن و اسه توجه :

```
[leet]
a=4
i=1
e=3
t=7
o=0
s=5
g=9
z=2
```

میتوانید این لیست رو به دلخواه خودتون عوض کنید . مثلا توی پیکربندی زیر هم مجموعه از کاراکتر ها رو تعریف کرده که اگه شما بخوايد میتوانه توی کلمات عبور ازش استفاده کنه :

```
[specialchars]
chars!=,@,#,$,%,&,*
```

میتوانید از طریق پیکربندی زیر، حداقل طول و حداکثر طول کلمات عبوری که میسازه رو هم مشخص کنید :

```
# [ Word length shaping ]
# This setting will exclude words from compiled wordlist that are shorter
# than [wcfrom] and longer than [wcto].
```

```
wcfom=5
wcto=12
```

دقت کنید که وقتی میخوايد یک لیست کلمه عبور رو بسازید به این پیکربندی ها توجه کنید و گرنه ممکن هست که کلمات عبوری نامفهوم و بی ربط به شما تحويل بده .

## CeWL Tools

یه ابزار دیگه ای که میتوانیم استفاده کنیم ازش تا یک لیست از کلمات عبور مرتب با تارگتمنوں رو ایجاد کنیم CeWL هست . این ابزار با روبی نوشته شده و یک URL رو از شما میگیره، یک Depth برآش مشخص میکنید و برآتون صفحه رو بررسی میکنه، کلمات استفاده شده رو بدست میاره و اونهایی که بیشتر تکرار شدن رو لیست میکنه . اون Depth هم بهش میگه که لینک های توی صفحه رو تا عمق X برو جلو و اطلاعات رو بدست بیار . میتوانید این ابزار رو از لینک زیر دانلود و استفاده کنید :

<https://github.com/digininja/CeWL>

همچنین به صورت پیشفرض هم در کالی لینوکس نصب است و میتوانید از طریق دستور زیر اجرایش کنید :

```
# cewl
CeWL 6.1 (Max Length) Robin Wood (robin@digi.ninja) (https://digi.ninja/)

Missing URL argument (try --help)
```

همچنین از این ابزار برای FUZZING کردن مسیر ها و پیدا کردن Subdomain ها هم میشه استفاده کرد چون ممکن هست برخی از کلمات پیدا شده توی منتهی به یک Subdomain یا مسیر بشه .

## Brute-Force Attack

یک نوع Password Attack هست که برخلاف Password Guessing که یک لیست از کلمات عبور احتمالی واسه Password Attack ایجاد میشد، یک لیست بزرگ از کلمات عبور رو بدون اینکه اطلاعی از شخص داشته باشیم ایجاد میکنیم، مثلا لیست کلمات عبور رایج 2020، و حمله رو انجام میدیم . توی این حمله به صورت کورکرانه کلمات عبور رو بررسی میکنیم . تفاوتش با Password Guessing اینه که :

- لیست پسوردموں بسیار بزرگتر هست . مثلا تمام عبارات 8 کاراکتری ساخته شده از A-Z, a-z, 0-9

- به علت بزرگ بودن لیست کلمات عبورموں، زمان انجام حمله بسیار زیاد تر خواهد بود

- به علت اینکه کلمات عبور توی لیستموں، همه کلمات عبور رو شامل میشه در نهایت کلمه عبور تارگت رو بدست خواهیم اورد،

برخلاف Password Guessing که احتمال موغنتیش کمه

یه مثالی اگه بخوایم از لیست کلمات عبور حمله Brute-Force بزنیم میتوانیم توی تصویر زیر ببینیم . تصویر زیر لیست کلمات عبور عددی 8 کاراکتری رو نشون میده :

```
00000000
00000001
00000002
00000003
00000004
00000005
00000006
00000007
00000008
00000009
00000010
00000011
```

از 00000000 تا 99999999 رو شامل خواهد شد . قابل درک هست که چه لیست بزرگی خواهد بود . علاوه بر لیست کلمات عبور میتوانیم توی حملمون یک لیست از نام های کاربری رو هم داشته باشیم . یعنی توی **Brute-Force** میتوانیم حمله رو روی یک تارگت و همچنین میتوانیم روی چندین تارگت انجام بدم . مثلا لیست زیر هم لیست نام های کاربری است :

```
username1
username2
username3
admin1
admin2
admin3
...
...
```

وقتی ما هم لیست کلمات عبور و هم لیست نام های کاربری رو داریم ، حمله به این شکل رخ میده که ، اولین نام کاربری انتخاب میشه و سپس تمام کلمات عبور روی اون نام کاربری بررسی خواهد شد و سپس میره سروقت نام کاربری بعد با تست کردن تمام کلمات عبور و ... این عمل تا اخرين نام کاربری و بررسی اخرين کلمه عبور برای اون انجام میشه . به این حالت میگن **Cluster Bomb** که در ادامه در **Burp Suite** هم میبینیمش . راه ساختن این لیست از طریق ابزار هایی که استفاده میشه اسمش **Crunch** هست .

## Crunch

به مجموعه ای از ترکیب های متفاوت از کاراکتر ها **wordlist** میگن که برای **Brute-Force** کردن یا شکست یک هش ما نیاز خواهیم داشت به یک **wordlist** خوب . برای ساختن یک **wordlist** خوب میتوانیم از ابزار **Crunch** استفاده کنیم . این ابزار به صورت پیش فرض توی کالی لینوکس نصب هست و میتوانید با استفاده از دستور زیر اون رو اجرا کنید :

```
# crunch
```

```
kali㉿kali:~$ crunch -h
crunch version 3.6

Crunch can create a wordlist based on criteria you specify. The output from crunch can be sent to the screen, file, or to another program.

Usage: crunch <min> <max> [options]
where min and max are numbers

Please refer to the man page for instructions and examples on how to use crunch.
```

این ابزار یک **wordlist** رو با توجه **Option** هایی که بهش میدیم از طریق جایگشت و ترکیب کردن می سازه و همچنین میتوانیم بهش یک الگو و پترن خاص رو بدهیم که بر اساس اون الگو برای ما لیست کلمات عبور رو ایجاد کنه . طریقه استفاده از این ابزار به شکل زیر هست :

```
Usage: crunch <min> <max> [options]
where min and max are numbers
```

میبینید که ابتدا دستور **crunch** رو وارد کنید، **<min>** و **<max>** نشون دهنده تعداد کاراکتر ها خواهد بود . مثلا دستور زیر به معنی اینه که، از تمام کاراکتر ها استفاده کن و کلمات عبور ۱ کاراکتری تا ۳ کاراکتری رو بساز :

```
# crunch 1 3
```

اون قسمت **[options]** میتوانه سوابع ها و مقادیر متفاوتی رو داشته باشه . برای اگاهی ازشون میتوانیم با دستور زیر صفحه **Manual** مربوط به ابزار **Crunch** رو بخونیم :

```
# man crunch
```

```
CRUNCH(1)                                         General Commands Manual

NAME
    crunch - generate wordlists from a character set

SYNOPSIS
    crunch <min-len> <max-len> [<charset string>] [options]

DESCRIPTION
    Crunch can create a wordlist based on criteria you specify. The output from crunch can be sent to the screen, file, or to another program. The required parameters are min-len and max-len. The minimum length string you want crunch to start at. This option is required even for parameters that won't use the value. The maximum length string you want crunch to end at. This option is required even for parameters that won't use the value.

MAKE_NIXOAIM_TAMAM_Option_HA_AIN_EBZAR_RO_BERSE_KINIM_WLI_OUNEHAYI_KHE_SHALID_NIYAZ_BASHE_RO_TOHPOOSH_MIDEYM. AKHE_JLOU_DSTOUR_KARAKTER_HAYI_RU_WARD_KNID, KLAMAT_UBOR_SHMA_BYA_MIN_WARD_SHDE, TAMAM_AZ_OVN_KARAKTER_HA_SAXTEN_XOWAHED_SHD. MELA_DSTOUR_ZIR_MIDAD_TAMAM_KLAMAT_UBORI_RU_MISAZEH_KHE_AZ_KARAKTER_HA_0123456789_SAXTEN_SHDE_ANDO_1TA4_KARAKTER_DARND:
```

```
# crunch 1 4 0123456789
```

```
kali@kali:~$ crunch 1 4 0123456789
Crunch will now generate the following amount of data: 54320 bytes
0 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 11110
```

میبینید که پس از اجرای دستور یک اطلاعاتی رو به ما نشون میده، مثلا میگه حجم پسوردهایی که ساخته میشه 54320 بایت خواهد بود و سپس اون به MB, GB, TB, PB هم بیان میکنه. در خط اخر هم بهمون میگه که چند خط پسوردهای خواهد بود، چون هر پسوردهایی توی یک خط نوشته میشه پس تعداد کلمات عبور ما در این دستور 11110 خواهد بود. اما crunch کلمات عبور رو روی صفحه ترمینال مینویسه و به صورت پیش فرض توی فایل ذخیره نمیکنه، مگر اینکه بهش بگیم. چطوری بهش بگیم؟ از طریق سوئیچ -o :

```
# crunch 1 4 0123456789 -o password_list.txt
```

دستور بالا خروجی رو توی یک فایل به نام password\_list.txt ذخیره خواهد کرد. یا هم میتوانیم از < استفاده کنیم و خروجی رو به یک فایل ارجاع دیم :

```
# crunch 1 4 0123456789 > password_list.txt
```

سوئیچ -t توی crunch چه میکند؟ گفتیم که crunch میتوانه بر ساس Pattern هم و اسه ما تولید کلمات عبور کنه. میتوانیم یه Pattern بهش بدم و بگیم بر اساس این Pattern کلمات عبور رو ایجاد کن. مثلا به بگیم که، لیست ما باید حاوی پسوردهایی باشد که با کلمه secret شروع میشه و سپس چهار کاراکتر حروف بزرگ هم تهش هست، این چهار کاراکتر میتوانه هر چیزی باشد ولی مهم اینه که کلمه عبور با secret شروع بشه. برای این کار باید ابتدا چهار تا کاراکتر خاص توی دستورات رو بفهمیم :

1. ":" : برای تمام کاراکتر های بزرگ

2. "@" : برای تمام کاراکتر های کوچک

3. "%" : برای تمام کاراکتر های عددی

4. "^^" : برای تمام کاراکتر های خاص (\_-#\$%^&\*!@#)

حال اگه بخوایم بگیم که پسوردها باید با secret شروع بشن و بعد از این کلمه 2 کاراکتر دیگه از حروف بزرگ هم قرار بده باید به شکل زیر عمل کنیم :

```
# crunch 8 8 -t secret,,
```

دقت کنید که توی تعریف کردن Pattern باید تعداد کاراکتر هایی که مشخص میکنید با تعداد کاراکتر های Pattern یکسان باشد.

سوئیچ -p چیه و چیکار میکنه؟ جایگشت: فرض کنید میخوايد کلمات عبوری رو بسازید که از ترکیب حروف کلمه password بدست میداد، برای اینکار از -p استفاده میکنیم. مثلا دستور زیر :

```
# crunch 1 1 -p password
```

دقت کنید که هنگام استفاده از سوئیچ `p`-، `crunch` اعداد `<min>`, `<max>` را در نظر نمیگیره و `ignore` میکنه . اگه مقادیری که در جلوی سوئیچ `p`- قرار میدید رو با کاراکتر فاصله جدا کنید، اینبار `crunch` و اونا رو به عنوان کلمه در نظر میگیره و جایگشت رو روی کلمات اعمال میکنند :

```
# crunch 1 1 -p black hacker
Crunch will now generate approximately the following amount of data: 18 bytes
0 MB
0 GB
0 TB
0 PB
Crunch will now generate the following number of lines: 2
blackhat
hatblack
```

میبینید که فقط دو مقدار رو در جواب برگرداند .

## Password Spray

یکی دیگه از Password Spray های مرسوم Bruteforce هست . در این نوع حمله بر خلاف Bruteforce که مجموعه ای از نام های کاربری داشتیم در کنار مجموعه ای از کلمات عبور و همه کلمات عبور برای هر کدام از نام های کاربری تست میشد، ما مجموعه ای از نام های کاربری داریم و یک کلمه عبور، این یک کلمه عبور روی تمام این نامهای کاربری تست میشه . در حقیقت کلمات عبور رو روی یک مجموعه ای از نام های کاربری اسپری میکنیم .

دوستان دقت کنید که ما استاندارد هایی رو داریم که شروطی رو برای امن سازی یک اپلیکیشن توضیح میدن . مثلا شروط لازم برای امن کردن کلمات عبور یک اپلیکیشن رو توی یک قسمتی مثلا Password Policy توضیح میده . وقتی که داریم یک وب اپلیکیشن رو بررسی میکنیم طبق یکی از این استاندارد ها گزارشمنوں رو بنویسیم . مثلا توی ASVS – Application Security Verification Standard که متعلق به OWASP هست درمورد Password Spray زیر رو گفته :

#	Description	L1	L2	L3	CWE	NIST §
2.1.1	Verify that user set passwords are at least 12 characters in length. <a href="#">(C6)</a>	✓	✓	✓	521	5.1.1.2
2.1.2	Verify that passwords 64 characters or longer are permitted. <a href="#">(C6)</a>	✓	✓	✓	521	5.1.1.2
2.1.3	Verify that passwords can contain spaces and truncation is not performed. Consecutive multiple spaces MAY optionally be coalesced. <a href="#">(C6)</a>	✓	✓	✓	521	5.1.1.2
2.1.4	Verify that Unicode characters are permitted in passwords. A single Unicode code point is considered a character, so 12 emoji or 64 kanji characters should be valid and permitted.	✓	✓	✓	521	5.1.1.2
2.1.5	Verify users can change their password.	✓	✓	✓	620	5.1.1.2
2.1.6	Verify that password change functionality requires the user's current and new password.	✓	✓	✓	620	5.1.1.2
2.1.7	Verify that passwords submitted during account registration, login, and password change are checked against a set of breached passwords either locally (such as the top 1,000 or 10,000 most common passwords which match the system's password policy) or using an external API. If using an API a zero knowledge proof or other mechanism should be used to ensure that the plain text password is not sent or used in verifying the breach status of the password. If the password is breached, the application must require the user to set a new non-breached password. <a href="#">(C6)</a>	✓	✓	✓	521	5.1.1.2
2.1.8	Verify that a password strength meter is provided to help users set a stronger password.	✓	✓	✓	521	5.1.1.2
2.1.9	Verify that there are no password composition rules limiting the type of characters permitted. There should be no requirement for upper or lower case or numbers or special characters. <a href="#">(C6)</a>	✓	✓	✓	521	5.1.1.2
2.1.10	Verify that there are no periodic credential rotation or password history requirements.	✓	✓	✓	263	5.1.1.2
2.1.11	Verify that "paste" functionality, browser password helpers, and external password managers are permitted.	✓	✓	✓	521	5.1.1.2
2.1.12	Verify that the user can choose to either temporarily view the entire masked password, or temporarily view the last typed character of the password on platforms that do not have this as native functionality.	✓	✓	✓	521	5.1.1.2

میتوانید فایل PDF موارد ذکر شده توی ASVS رو از لینک زیر ببینید :

[https://owasp.org/www-pdf-archive/OWASP\\_Application\\_Security\\_Verification\\_Standard\\_4.0-en.pdf](https://owasp.org/www-pdf-archive/OWASP_Application_Security_Verification_Standard_4.0-en.pdf)

## Dictionary Attack

در مورد **Dictionary Attack**، **Password Guessing** و **Brute-Force** محسوب میشند، **Dictionary Attack** هم به همین منوال هست و یکی از انواع **Password Attack** هاست. دیکشنری چیه؟ فارغ از اینکه چه کاری میکنه، دیکشنری یک کتاب هست که تو ش پر است از کلمات معنا دار و معنای هر کلمه رو جلوش نوشتن. توی **Password Attack** ها ما لیستی از پسورد ها رو داشتیم. مثلا توی **Password Guessing** ما لیستی داشتیم از کلمات مربوط به تارگت، توی **Brute-Force** لیستی داشتیم مثلا شامل تمام کلمات عبور 8 کاراکتری، توی **Password Spray** ما یک کلمه عبور داشتیم. در **Dictionary Attack** هم ما یک لیستی از کلمات عبور داریم. اما تفاوت اینجاست که لیست کلمات عبور ماتوی این حمله، برخلاف **Password Guessing** تارگتمنون نیست، برخلاف **Brute-Force** شامل تمام کلمات عبور X کاراکتری نمیشه بلکه شامل لیستی از کلمات عبوری هست که زیاد ازشون استفاده میشه. شما میتوانید این لیست ها رو از لینک زیر دریافت کنید:

<https://github.com/danielmiessler/SecLists/>

مثلا لیستی از 10000 کلمه عبور پر استفاده در سال 2022 یا لیستی از نامهای انگلیسی که در کلمات عبور استفاده میشند و ... توی این مخزن گیتها بیهوده است.

## Rainbow Table Attack

این حمله یکی از انواع **Password Attack** هست که جهت **Crack** کردن کلمات عبور استفاده میشه. در این حمله از جداول خاصی به نام **Rainbow Table** جهت شکستش هش یک کلمه عبور در پایگاه داده تارگت استفاده میشه. پس نتیجه میگیریم که نسبت به حملات دیگه **Password Attack** متفاوت است. ما در این حمله قصدمنون بدست اوردن کلمه عبور داخل پایگاه داده نیست، بلکه ما کلمات عبور رو بدست اوردهیم ولی این کلمات عبور Hash شدن و میخوایم این هش رو بشکنیم. میدونیم که در پایگاههای داده تقریباً تمامی اپلیکیشن های امروزی، کلمات عبور رو به شکل **Plain Text** ذخیره نمیکنند، بلکه میان و اونها رو به شکل Hash در میارند که قابل بازگشت نیست و در پایگاههای داده خود قرار میدهند. اینکار جهت دست نیافتن مهاجم به **Credentials** کاربران در صورت دسترسی به پایگاه داده انجام میشه. در مورد **Hash** قبل صحبت کردیم. اما Rainbow Table چیه که در این حمله استفاده میشه؟ یک جدول شامل رکورد هایی از یک کلمه عبور **Plain Text** و هش مربوط به اون کلمه عبور هست. مثلا رکورد ها به شکل زیر تو شقرار گرفته:

```
password: 286755fad04869ca523320acce0dc6a4
123456: f447b20a7fcfb53a5d5be013ea0b15af
```

چیزی که ما به عنوان یک نفوذگر در اختیار داریم یک هش هست که نمیدونیم مقدار معادلش به صورت **Plain Text** چی هست، مثلا:

```
f447b20a7fcfb53a5d5be013ea0b15af
```

حال میایم و یه اسکریپت مینویسیم تا توی **Rainbow Table** که در اختیار داریم به دنبال این هش توی رکورد ها بگردد، در صورتی که هش رو پیدا کرد از میخوایم که عبارت **Plain Text** این هش رو به ما بده. در مثال بالا کلمه عبور 123456 خواهد بود چرا که هش تارگت ما با هش این عبارت یکسان است. دقت کنید که باید قل از حمله حتما نوع هش تارگت رو از طریق ابزار هایی مثل hash-identifier تشخیص بدم و هش های توی **Rainbow Table** ما هم باید با نوع هش مورد نظر ما یکسان باشه.

یکی از راههایی که برای جلوگیری از بدست اوردن کلمه عبور از هش اتخاذ کردن، **Salting** است. **Salting** به معنی اینه که وقتی کاربر ثبت نام میکنه و میخوان هش این کلمه عبورش رو توی پایگاه داده ذخیره کنن، کلمه عبور + یک عبارت خاص (**Salt**) رو هش میکنند و این عبارت خاص رو هم نگهداری میکنند تا زمانی که خواستن کاربر رو تشخیص بدن. این عبارت خاص رو یک به صورت **Hard Coded** توی سورس اپلیکیشن قرار میدن و یا به صورت **Dynamic** برای هر کاربری جداگونه میسازند. اضافه شدن این عبارت به کلمه عبور و سپس هش شدن هر دوی اینها یکجا موجب میشه که بدست اوردن کلمه عبور از هش بسیار بسیار سخت تر و حتی غیر ممکن بشه. یکی از مواردی که باید در **Salting** رعایت کنند ذخیره سازی امن **Salt** هست، در صورتی که مهاجم علاوه بر هش به **Salt** هم دسترسی پیدا کنه، عملاً **Salting** بی فایده میشه. چند مدت پیش که پایگاه داده یکی از شرکت ها لیک شده بود مشخص شد که **Salt** کلمه عبور هر کاربر در یک ستون به نام **salt** ذخیره شده. یعنی مهاجم وقتی به پایگاه داده دسترسی پیدا کرد، عملاً **Salt** رو هم گرفت و این موضوع واقعاً کار احتمانه ای بود که توی یک شرکت بزرگ انجام شده بود (:)) منظورم اینه که چنین اتفاقی هم می افته.

نکته قابل توجه اینه که توی **Rainbow Table** رکورد ها باید شامل **Hash** و کلمه عبور باشه، نه اینکه فقط شامل کلمه عبور باشه و موقع کرک اون رو به هش تبدیل و قیاس کنیم، اینکار موجب بالا رفته مصرف منابع سیستممنون میشه و عملاً حمله رو به شدت کند میکنه. اعمال الگوریتم هش روی یک عبارت به شدت منابع رو مصرف میکنه.

## RainbowCrack Tools

یه ابزاری توی کالی لینوکس وجود داره به نام **RainbowCrack** که واسه انجام **Rainbow Attack** ها استفاده میشه . درصورتی که روی کالیتون نصب نیست میتونید از طریق دستور زیر اقدام به نصبش کنید :

```
# sudo apt install rainbowcrack -y
```

البته نسخه تحت ویندوز هم داره که میتونید از لینک زیر دانلود و نصب کنید :

<http://project-rainbowcrack.com/>

پس از نصب یه مجموعه دستورات بهتون میده به شکل زیر :

```
# rcrack
# rt2rtc
# rtgen           //Generate Ranbow Table
# rtmerge
# rtsort
```

من از این ابزار استفاده نکردم و به همین خاطر بیشتر از این توضیح نمیدم و فقط جهت اطلاع از وجودش گفتم اینجا بنویسم درموردش .

## Burp Suite Intruder

برای **Brute-Force** کردن صفحات وب خیلی از افراد به اشتباه میگن که باید از ابزاری مثل **Hydra** استفاده کرد . این ابزار درسته که از پروتکل **HTTP** پشتیبانی میکنه ولی برای این کار ساخته نشده، قبلا که یادمه سیستم عامل **Back Track** او مده بود دوستان از این ابزار جهت Crack کردن جیمیل و ... استفاده میکردن و زیاد هم پیشنهاد میشد ولی به علت **False Positive** های زیادش اشتباه بود . حتی با وجود اینکه **Hydra** گفته از پروتکل های ... **SSH, SMTP, FTP, Telnet**, ... پشتیبانی میکنه بازم پیشنهاد به استفاده ازش نمیشه و همیشه ابزار **ncrack** رو پیشنهاد میکن . اینو لازم بود بگم تا مسیر اشتباه رو نریم . اما یکی از بهترین ابزار هایی که برای **Brute-Force** وب پیشنهاد میشه **Burp Suite Intruder** هست . این ابزار بسیاری از ابزار های لازم رو در اختیار ما قرار میده و کارمون رو به شدت راحت میکنه . **Burp Intruder** یک ابزار قادرمند جهت اجرای حملات کاستوم خودکار بر روی وب اپلیکیشن هاست . این ابزار به شدت انعطاف پذیر و قابل پیکربندی هست و میتوانه تمام کارهایی که لازمه واسه تست کردن یک وب اپلیکیشن رو در اختیارمون قرار بده . اما چطوری کار میکنه ؟ این ابزار یک **HTTP Request** رو از ما میگیره که بهش میگن **Base Request**، درخواست رو به اشكال مختلف تغییر میده و به سمت وب اپلیکیشن میفرسته، پاسخ وب اپلیکیشن رو دریافت و این پاسخ ها رو بررسی میکنه . برای هر حمله ای باید یک یا بیشتر از یک پیلود رو تنظیم کنیم و توی **Base Request** محل قرار گیری پیلودمون رو مشخص کنیم . برای ساختن پیلود تعداد زیادی روش در اختیار ما قرار میده مثلا میتوانیم ... **Simple List, String, Number, Date, ...** رو ایجاد کنیم . درمورداشون به صورت عملی صحبت میکنیم . طریقه استفاده به صورت خلاصه میشه :

۱. یک **HTTP Request** رو به **Intruder** میدیم .

۲. نوع **Brute-Force** رو مشخص میکنیم . (صحبت میکنیم درمورداشون)

۳. محل قرار گیری پیلود هامون رو مشخص میکنیم .

۴. پیلود هامون رو میسازیم .

۵. اگه پیکربندی برای حمله نیاز بود انجام میدیم .

۶. حمله رو اجرا میکنیم .

۷. پاسخ ها رو بررسی و انالیز میکنیم .

۸. اطلاعاتی میخوایم رو از انالیز پاسخ ها بدست میاریم .

کجاها از **Burp Intruder** استفاده میکنیم ؟

- **Enumerating (IDOR, ...)**

- **Harvesting**

- **Fuzzing**

مود های و نوع های **Brute-Force** چیا هستند ؟ توی **Intruder** ما مود های مختلفی برای حمله **Brute-Force** که میخوایم انجام بدیم داریم و طبق نیازمون باید ازشون استفاده کنیم . این مود ها عبارت اند از :

۱. **Sniper**

۲. **Battering Ram**

۳. **Pitchfork**

مد **Intruder Sniper** در چیه؟ توی این مد، هر پیلودی توی یک **Position** قرار میگیره. فرض بگیرید که یک **Payload** داریم و یک **Position** توی درخواستمون، مثلاً:

```
# Payloads
admin
user1
user2
user3
user4
```

درخواستمون هم به شکل زیر هست:

```
# Base Request
POST /login.php HTTP/1.1
Host: site.com
...
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: csrfToken=p9hkoLwPNxY3ljrglMym2FpNbPsvqHoN; PHPSESSID=erfuf11382hn38m5qsc22et1hh
Connection: close

username=$username$&password=password
```

اون قسمتی که **Highlight** شده جایی هست که قراره پیلود ها رو به جاش قرار بدیم. میبینید که با کاراکتر \$ در اطرافش هم مشخص شده و توی **Intruder** هم چنین خواهد بود. در این حالت وقتی حمله انجام میشه به جای مقدار پارامتر **username** مقادیر پیلود هامون قرار میگیره، یعنی بدنه درخواستمون به ترتیب به شکل زیر میشه:

```
username=admin&password=password
username=user1&password=password
username=user2&password=password
username=user3&password=password
username=user4&password=password
```

چیزی که بالا میبینیم **Password Spray Attack** هست. حال اگه توی **Base Request** دو تا جای پیلود رو مشخص کنیم:

```
# Base Request
POST /login.php HTTP/1.1
Host: site.com
...
Accept-Language: en-US,en;q=0.9
Cookie: csrfToken=p9hkoLwPNxY3ljrglMym2FpNbPsvqHoN; PHPSESSID=erfuf11382hn38m5qsc22et1hh
Connection: close

username=$username$&password=$password$
```

پیلودمون هم همون چیزی هست که بالا گفتیم. در این حالت حمله ما شامل 10 درخواست با بدنه زیر میشه:

```
username=admin&password=password
username=user1&password=password
username=user2&password=password
username=user3&password=password
username=user4&password=password
username=username&password=admin
username=username&password=user1
username=username&password=user2
username=username&password=user3
username=username&password=user4
```

توی حمله از نوع **Sniper** ما فقط یک لیست **Payload** خواهیم داشت ولی میتونیم به هر تعدادی که نیاز باشه **Position** توی **Base Request** داشته باشیم. تمام پیلود ها رو ابتدا در 1 **Position** و سپس در 2 **Position** و تا در N **Position** درخواست خواهیم داشت. تعداد درخواست های ما در این نوع حمله **Payloads \* Position** خواهد بود.

بریم توی Burp Suite ... ببینیم توی منوی اصلی Burp Suite هست و برای ورود کافیه که روش کلیک کنید :

اما چطوری یک Base Request بهش بدیم؟ کافیه که یک درخواستی رو از Proxy بگذروند :

```

1 POST /login.php HTTP/1.1
2 Host: 192.168.89.128
3 Content-Length: 35
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36
7 Origin: http://192.168.89.128
8 Content-Type: application/x-www-form-urlencoded
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Referer: http://192.168.89.128/login.php
11 Accept-Encoding: gzip, deflate, br
12 Accept-Language: en-US,en;q=0.9
13 Cookie: csrfToken=p9hkoLwPNxY3ljqrg1MyM2FpNbPsvqHoN; PHPSESSID=g2e3g3uuulolig7apu44c3o6tlo
14 Connection: close
15
16 username=username&password=password

```

حال کافیه که Ctrl+Alt+Shift را بزنیم تا این درخواست به Intruder برده یا هم میتواند کلیک راست کنید و رو Send to Intruder بزنید :

حال اگه وارد تب Intruder بشید به چیزی به شکل زیر مواجه خواهد شد :

Choose an attack type

Attack type: Sniper

Payload positions

Configure the positions where payloads will be inserted, they can be added into the target as well as the base request.

Target: http://192.168.89.128

POST /login.php HTTP/1.1
 Host: 192.168.89.128
 Content-Length: 35
 Cache-Control: max-age=0
 Upgrade-Insecure-Requests: 1
 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/122.0.0.0 Safari/537.36
 Origin: http://192.168.89.128
 Content-Type: application/x-www-form-urlencoded
 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,\*/\*;q=0.8,application/signed-exchange;v=b3;q=0.7
 Referer: http://192.168.89.128/login.php
 Accept-Encoding: gzip, deflate, br
 Accept-Language: en-US,en;q=0.9
 Cookie: csrfToken=p9hkoLwPNxY3ljqrg1MyM2FpNbPsvqHoN; PHPSESSID=g2e3g3uuulolig7apu44c3o6tlo
 Connection: close

بریم و چیزایی که مشخص کردیم رو توضیح بدیم :

1. وسط صفحه شما **Base Request** رو میبینید.
2. قسمت شماره 1 : **Attack type** را میتوانیم از این قسمت تعیین کنید و میبینید که به صورت پیش فرض روی **Sniper** هست.
3. قسمت شماره 2 : با این دکمه میتوانید قسمتی از **Base Request** را انتخاب کنید و سپس تبدیل کنیم به **Payload Position** یک **Mutator** توی تصویر زیر میبینید که چنین کردم :

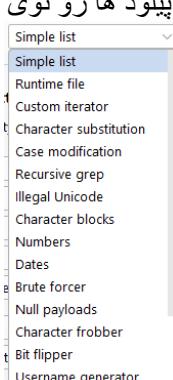
میبینید که **username** محصور به دو تا کاراکتر ظاهر شده و این به معنی **Position** شدنش هست، یعنی پیلود هایی که قراره انتخاب کنیم رو جای ایشون قرار میده. میبیند که **password** تبدیل به **Position** نشده و ثابت خواهد بود. اگه میخواهد اون رو هم تبدیل کنید، کافیه که **Select** کنید و سپس رو دکمه **Add** کلیک کنید.

4. قسمت شماره 3 : این دکمه همه **Position** های مشخص شده توی **Base Request** را حذف میکنه.
5. قسمت شماره 4 : این دکمه مارو وارد صفحه تنظیمات **Payload** میکنه، پس از اینکه **Position** ها و **Attack Type** را تعیین کردیم برای پیکربندی **Payload** هامون باید وارد این تب بشیم.

## تب **Payload** چیا توش داره ؟

1. قسمت شماره 0 : توی این قسمت مجموعه پیلود ها رو خواهیم داشت. تعداد مجموعه پیلود ها بر اساس نوع حمله ای که توی تب **Attack type** تعیین کردیم تفاوت خواهد بود. مثلا حمله نوع **Sniper** فقط و فقط یک مجموعه پیلود خواهد داشت.

2. قسمت شماره 1 : اینجا میتوانیم مجموعه ای از پیلود هامون که میخوایم تغییرات روش اعمال کنیم و پیکربندی هاش رو عوض کنیم رو انتخاب میکنیم. میبینید که مجموعه شماره 1 انتخاب شده و من هر پیکربندی رو اعمال کنم روی مجموعه پیلود شماره 1 اعمال خواهد شد.
3. قسمت شماره 2 : هر مجموعه پیلود میتوانه نوع خواص خودش رو داشته باشه، یک مجموعه پیلود میتوانه **Simple list** و ... باشه. انواع مجموعه پیلود ها رو توی تصویر زیر میبینید :



4. قسمت شماره 3 : بر اساس نوع مجموعه پیلود که انتخاب میکنیم این قسمت هم متفاوت خواهد بود . چیزی که توی تصویر بالا میبینید در صورتی نشون داده میشه که Simple list Payload type شما

5. قسمت شماره 4 : زمانی که نوع مجموعه پیلود شما Simple list باشه شما چنین چیزی رو خواهید دید . یک لیست ساده به عنوان پیلود میتوانه اعضايی داشته باشه، شما میتوانید عضو بهش اضافه کنید، ازش کم کنید و ...

6. قسمت شماره 5 : پس از پیکربندی مجموعه پیلودهاتون میتوانید روی این دکمه کلیک کنید تا حمله شروع شود .

در صورتی که نوع مجموعه پیلودتون رو Numbers قرار بدید قسمت مربوط به Payload Settings به شکل زیر خواهد بود :

**Payload sets**

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set:	1	Payload count:	1
Payload type:	Numbers	Request count:	0

**Payload settings [Numbers]**

This payload type generates numeric payloads within a given range and in a specified format.

Number range

Type:  Sequential  Random

From: 0

To:

Step: 1

How many:

Number format

Base:  Decimal  Hex

Min integer digits: 0

Max integer digits: 0

Min fraction digits: 0

Max fraction digits: 0

Examples

0  
0

میبینید که میتوانید یک Range از اعداد رو به عنوان پیلود داشته باشید . میتوانید مشخص کنید از 0 تا 10000 را به عنوان پیلود انتخاب کنه و قدم های 2 تابی برداره، یا هم میتوانید مشخص کنید که عدد مورد نظر Random باشه .

در صورتی که نوع مجموعه پیلودتون رو Dates انتخاب کنید، تنظیمات پیلودتون به شکل زیر خواهد بود :

**Payload sets**

You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.

Payload set:	1	Payload count:	1
Payload type:	Dates	Request count:	0

**Payload settings [Dates]**

This payload type generates date payloads within a given range and in a specified format.

From: 3 March 2024

To: 3 March 2024

Step: 1 Days

Format:  3/3/24  dd.MM.yyyy

Example: 3/3/24

میتوانید یک رینج تاریخ انتخاب کنید و بهش بگید که از فلان تاریخ تا بهمان تاریخ رو یک روز یا یک ماه یا یک سال به جای پیلود قرار بده .

میتوانید خودتون انواع پیلود ها رو بررسی کنید و ببینید چه چیز های جالبی نوشته، این انواع پیلود ها میتوانه خیلی کمک کننده و راحت کننده انجام حمله باشه .

خب من یک Simple list ساده به عنوان مجموعه پیلود انتخاب کردم، به شکل زیر و میخواهم حمله رو اجرا کنم :

**Payload settings [Simple list]**

This payload type lets you configure a simple list of strings that are used as payloads.

Paste	admin
Load ...	user1
Remove	user2
Clear	user3
Deduplicate	user4
Add	Enter a new item
Add from list ...	

## توى تصوير زير اجرای حمله رو مىبىنيد :

Attack Save Columns  
Results Positions Payloads Resource pool Settings

3. Intruder attack of http://192.168.89.128 - Temporary attack - Not saved to project file

Filter: Showing all items

Request ^	Payload	Status code	Error	Timeout	Length	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	1269	
1	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	795	
2	user1	200	<input type="checkbox"/>	<input type="checkbox"/>	1269	
3	user2	200	<input type="checkbox"/>	<input type="checkbox"/>	1269	
4	user3	200	<input type="checkbox"/>	<input type="checkbox"/>	1269	
5	user4	200	<input type="checkbox"/>	<input type="checkbox"/>	1269	

Request Response  
Pretty Raw Hex

```

1 POST /login.php HTTP/1.1
2 Host: 192.168.89.128
3 Content-Length: 32
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.0.0 Safari/537.36
7 Origin: http://192.168.89.128
8 Content-Type: application/x-www-form-urlencoded
9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
10 Referer: http://192.168.89.128/login.php
11 Accept-Encoding: gzip, deflate, br
12 Accept-Language: en-US,en;q=0.9
13 Cookie: csrfToken=p9Xk0lwPwX31jrg1MymCFpNbPsvqHoN; PHPSESSID=g2e3g3uuolig7apu44c3o6t1o
14 Connection: keep-alive
15
16 username=admin&password=password

```

مىبىنيد كە پىلۇد ھا رو نشون داده، درخواست ارسالى بە سمت تارگت رو نشون داده و مىبىنيد كە بە جاي مقدار پارامتر **username** در بىلدۈرۈشى دەرىخواست، پىلۇد من رو قرار داده، مىتوتىد از دكمە **Response** پاسخى كە تارگت بە اين درخواست داده رو بىبىنيد، توى حدولى كە پىلۇد ھامون تووشە ستۇن ھاى مختلفى رو مىبىنيد، مثلا ستۇن مربوط بە **Status code**، **Error**، **Timeout**، **Columns** پاسخ درخواستى كە فرستاده، **TimeOut**، **Error**، طول پاسخى كە تارگت فرستاده و حتى خودتون هم مىتوتىد ستۇن ھاىي رو از منىي بالا دكمە **Columns** بە اين موارد اضافە كىنىد . پس از اينكە حملە اجرا شد شما بايد پاسخ ھاىي كە درىافت شدە رو بىرسى كىنىد و بىبىنيد كە تارگت نسبىت بە درخواست ھاى مختلف چە واكنىشى رو نشون داده، بايد تقاوۇت پاسخ ھا رو پىدا كىنىد، مثلا گاھى تقاوۇت توى **Length** ھىست و اندازە پاسخى كە ما مىخوايم خىلى كەمتر و يا خىلى بىشتر از طول دىيگر پاسخ ھاست، گاھى اوقات مثل شكل بالا تقاوۇت توى **Status code** ھىست . اينجاست كە بايد خودتون بر اساس استدلال خودتون تىصىم بىگرىد .

يە تبى اون بالا وجود داره بە نام **Settings** كە نوش كارھاىي رو مىشە انجام داد . توى تصوير زير تظيماتى تحت عنوان **Grep – Match** رو مىبىنيد :

Attack Save Columns  
Results Positions Payloads Resource pool Settings

3. Intruder attack of http://192.168.89.128 - Temporary attack - Not saved to project file

### Grep - Match

These settings can be used to flag result items containing specified expressions.

Flag result items with responses matching these expressions:

Paste Load ... Remove Clear Add Enter a new item

not found  
unknown  
uid=  
c:\  
varchar  
ODBC  
SQL

Match type:  Simple string  Regex

Case sensitive match  Exclude HTTP headers

يکى از جاھايى ھىست كە گاھى نياز مىشە ما كارھاىي رو باھاش انجام بىيم . فرض كىنىد بە دنبال يك عبارت خاص توى پاسخ ھا ھىستىد، مثلا مىخوايد مشخص كىنىد كە ايا توى يك پاسخ عبارت . "Wrong username or password ." وجود داره يا خىر ؟ كافىيە كە تمام موارد توى **Add** جدول بالا رو باز زدن روی دكمە **Clear** پاڭ كىنىد، سپس عبارت مورد نظرتون رو توى باكس **Enter a new item** وارد كىنىد و روی **Add** بىزنىد . اون **Checkbox** كە جلوش نوشتە ... **Flag result items with ...** بە شكل زىر :

Flag result items with responses matching these expressions:

Paste Load ... Remove Clear Add Wrong username or password .

حالا اگه برد توی جدول نتایج حمله که توی تب **Results** هست خواهد دید که یک ستون به جدول اضافه شده معادل چیزی که شما به جدول بالا اضافه کردید.

Request	Payload	Status code	Time of day	Response size	Error	Timeout	Length	Cookies	Wrong username or password .	Comment
0		200	00:29:29 3 Mar 2024	556	<input type="checkbox"/>	<input type="checkbox"/>	1269		1	
1	admin	302	00:29:29 3 Mar 2024	344	<input type="checkbox"/>	<input type="checkbox"/>	795		1	
2	user1	200	00:29:29 3 Mar 2024	600	<input type="checkbox"/>	<input type="checkbox"/>	1269		1	
3	user2	200	00:29:29 3 Mar 2024	263	<input type="checkbox"/>	<input type="checkbox"/>	1269		1	
4	user3	200	00:29:29 3 Mar 2024	424	<input type="checkbox"/>	<input type="checkbox"/>	1269		1	
5	user4	200	00:29:29 3 Mar 2024	482	<input type="checkbox"/>	<input type="checkbox"/>	1269		1	

میبینید که توی این ستون یا مقدار | قرار داره و یا مقدار خالی، در واقع **Grep - Match** عبارتی که شما بهش میدید رو توی پاسخ ها جستجو میکنه و اگه اون عبارت توی پاسخ وجود داشت توی ستون اون عبارت عدد | رو قرار میده و اگه وجود نداشت خالی میداره جاش . توی تصویر بالا میگه که توی پاسخ با پیلود admin چنین عبارتی وجود نداره . دقت کنید که توی تنظیمات **Grep - Match** عبارت user1, user2, user3, user4 هم بتوانید Regex شما میتوانید همچنین **Case sensitive match** رو زمانی فعال کنید که میخوايد جستجوتون به حروف بزرگ و کوچک فعال باشه . اگه میخوايد جستجوتون توی **Header** ها پاسخ هم انجام بشه کافیه که تیک **Exclude HTTP headers** رو بردارید .

These settings can be used to flag result items containing specified expressions.

Flag result items with responses matching these expressions:

Wrong username or password .

Add Wrong username or password .

Match type:  Simple string  Regex

Case sensitive match  Exclude HTTP headers

تنظیمات دیگری که میتوانید توی تب **Settings** اعمال کنید و بسیار کاربردی است **Grep - Extract** هست .

These settings can be used to extract useful information from responses into the attack results table.

Extract the following items from responses:

Add

Extract

Remove

Duplicate

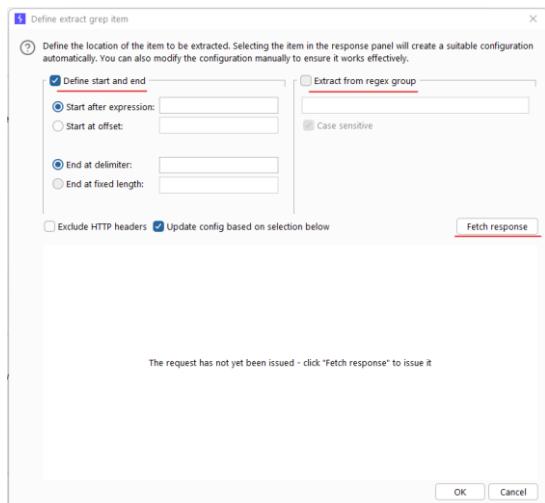
Up

Down

Clear

Maximum capture length: 100

فرض کنید که یک حفره امنیتی پیدا کردید که اطلاعات حساسی رو توی پاسخ به شما بر میگیردونه، یه چیزی شبیه به **IDOR** یا **User Enumeration** به هر طریقی، حال میخواهد که با انجام یک حمله تمام اطلاعات رو بیرون بکشد، اما ما میدونیم که توی پاسخ صرفاً اطلاعات مورد نظر ما وجود نداره و علاوه بر اونها میتوانه تگ های مختلف دیگه ای وجود داشته باشه . از طریق تنظیمات **Grep - Extract** میتوانید به **Intruder** بگید که اطلاعات خاصی رو بر اساس **Regex** برام بیرون بکش (: هر چیزی رو میخوايد باید به لیست تصویر بالا اضافه کنید . برای اینکار باید روی دکمه **Add** کلیک کنید . پس از اینکار یک پنجره به شکل زیر باز میشه :



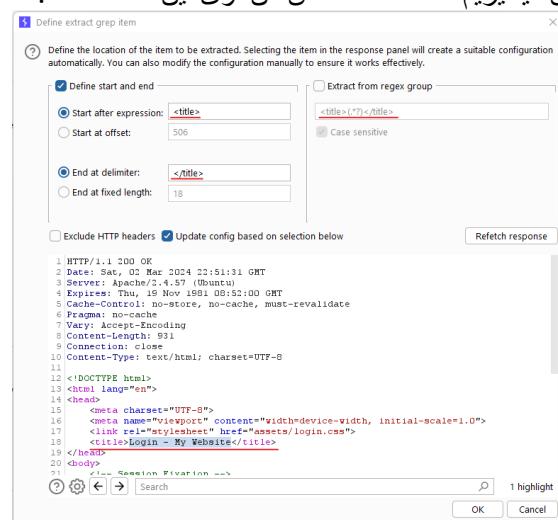
اینجاست که باید ابتدا و انتهای جایی که میخواید اطلاعات را از توی پاسخ بیرون بکشید مشخص کنید. میتوانید یا به صورت دستی اینکار را انجام بدهید و یا از طریق دکه Fetch response یک پاسخ را دریافت کنید و از توی اون پاسخ قسمت مورد نظرتون رو Select کنید. وقتی روی دکمه Fetch response بزنید این صفحه به شکل زیر در میاد:

```

HTTP/1.1 200 OK
Date: Sat, 02 Mar 2024 22:51:31 GMT
Server: Apache/2.4.57 (Ubuntu)
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Vary: Accept-Encoding
Content-Length: 931
Content-Type: text/html; charset=UTF-8
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<link rel="stylesheet" href="assets/login.css">
<title>Login - My Website</title>
</head>
<body>
    ...
</body>

```

یک پاسخ برآتون پایین این دکمه نشون داده میشه. حالا میتوانید با انتخاب هر قسمتی که خواستید، اطلاعات اون قسمت رو استخراج کنید. مثلا من تگ title رو میخوام استخراج کنم، فرض میگیریم اطلاعات حساس من توی این تگ هست:



میبینید که توی قسمت Define start and end که اطلاعات مورد نظر من ما بین دو تگ <title> و </title> هست. اگه میخوايد با Regex کار کنه میتوانید به جای تیک Define start and end، تیک گزینه Extract from regex group رو بزنید و همونطور که میبینید برامون Regex استخراج اطلاعات مابین دو تگ <title>, </title> رو نوشته. پس از انتخاب، کافیه رو اوکی کلیک کنید تا به لیست تنظیمات Grep – Extract اضافه شود:

Grep - Extract

These settings can be used to extract useful information from responses into the attack results table.

Extract the following items from responses:

- Add
- Edit
- Remove
- Duplicate
- Up
- Down
- Clear

From [<title>] to [</title>]

Maximum capture length: 100

پس از این کار توی جدول حمله که ستون اضافه خواهد شد که اطلاعاتی که ازش خواستیم استخراج کنے رو برای هر درخواست جلوش نوشته، توی تصویر زیر میبینید :

Attack Save Columns

4. intruder attack of http://192.168.89.128 - Temporary attack - Not saved to project file

Request	Payload	Status code	Error	Timeout	Length	<title>	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	1269	Login - My Website	
1	admin	302	<input type="checkbox"/>	<input type="checkbox"/>	795	Login - My Website	
3	user2	200	<input type="checkbox"/>	<input type="checkbox"/>	1269	Login - My Website	
4	user3	200	<input type="checkbox"/>	<input type="checkbox"/>	1269	Login - My Website	
5	user4	200	<input type="checkbox"/>	<input type="checkbox"/>	1269	Login - My Website	
2	user1	200	<input type="checkbox"/>	<input type="checkbox"/>	1269	Login - My Website	

توی مثل من همه پاسخ ها تگ <title> داشتند و مقدار داخل همیشون Login – My Website بوده است. خب حالا اطلاعات رو استخراج کردیم، چطوری میتونیم ذخیره کنیم؟ برای این کار کافیه که روی منوی Save در بالای تصویر بالا، کلیک کنید، سپس رو بزنید :

Attack Save Columns

4. Intruder attack of http://192.168.89.128 - Temporary

Request	Payload	Status code	Error	Timeout	Length	<title>	Comment
0		200	<input type="checkbox"/>	<input type="checkbox"/>	1269	Login - My Website	

پس از اینکار پنجره ای به شکل زیر برآتون باز میشه :

Save results table

All rows       Save header row

Selected rows      Delimiter:  Tab  Custom:

Wrap data safely for Microsoft Excel

Include columns:

<input checked="" type="checkbox"/> Request	<input checked="" type="checkbox"/> Payload	<input type="button"/> Select all
<input checked="" type="checkbox"/> Status code	<input type="checkbox"/> Time of day	<input type="button"/> Select none
<input type="checkbox"/> Response received	<input type="checkbox"/> Response completed	
<input checked="" type="checkbox"/> Error	<input checked="" type="checkbox"/> Timeout	
<input checked="" type="checkbox"/> Length	<input type="checkbox"/> Cookies	
<input checked="" type="checkbox"/> <title>	<input checked="" type="checkbox"/> Comment	

Save

Cancel

کافیه که ستون مورد نظرتون رو انتخاب کنید و ستون های دیگه Unselect کنید که در تصویر بالا باید ستون <title> را انتخاب کنیم و بعد رو دکمه Save بزنید و اطلاعات رو در یک فایل txt ذخیره کنید . به همین سادگی .

اما تمام این چیز هایی که گفتیم رو میتوانیم قبل از اجرای حمله هم تنظیم کنیم . کافیه که توی صفحه پیکربندی پیلود ها و Position ها تب Settings رو باز کنیم و کاراش رو انجام بدیم .

یه قسمتی که در مodus حرف نزدیم تب مربوط به Resource pool هست .

این تب چه چیز هایی رو توی خودش داره ؟ Resource pool به معنی جمع کردن مجموعه ای از منابع مورد نیاز جهت انجام یک کار هست . توی این تب هم ما همینکار رو انجام میدیم، مبایم و منابعی رو برای انجام حمله در اختیار Burp Intruder قرار میدیم، مثلًا مبایم و تعداد thread ها رو برای تعیین میکنیم، زمان ارسال درخواست رو میتوانیم تعیین کنیم و همچنین میتوانیم به Burp Intruder بفهمونیم که کی حمله رو Throttling کنه یعنی خفه کنه پس کار Resource pool رو فهمیدیم . حالا چطوری این کار رو انجام بدیم ؟ کافیه که یه تعیین کنیم :

ابتدا باید روی Create new resource pool بزنیم .  
قسمت شماره ۱ : نام Resource Pool مورد نظرمون رو وارد میکنیم .

- 

-

- قسمت شماره 2 : حداقل تعداد Thread ها رو باید مشخص کنیم . دقت کنید که نباید زیاد باشه چرا که حمله رو قابل تشخیص میکنه
- قسمت شماره 3 و 4 : زمان تأخیر ما بین هر درخواست توی حملمون رو میتوانیم به میلی ثانیه مشخص کنیم . حتی میتوانیم اون رو به صورت تصادفی هم تعیین کنیم . بگیم که مثلاً توی یک درخواست 1000 میلی ثانیه و توی یه درخواست دیگه 1100 میلی ثانیه صبر کنه . همچنین میتوانیم یه زمانی رو هم مشخص کنیم که تأخیر رو به یک اندازه ای افزایش بده .
- قسمت شماره 5 : اینجا میتوانیم به Intruder بگیم که چه زمانی یک حمله رو خفه کنه، یعنی چی؟ یعنی اینکه برآش مشخص میکنیم که زمانی که Status code شماره 429 یا 503 او مد به معنی شناسایی شدن و بلاک شدن ماتوسط WAF هست و حمله رو متوقف کن .

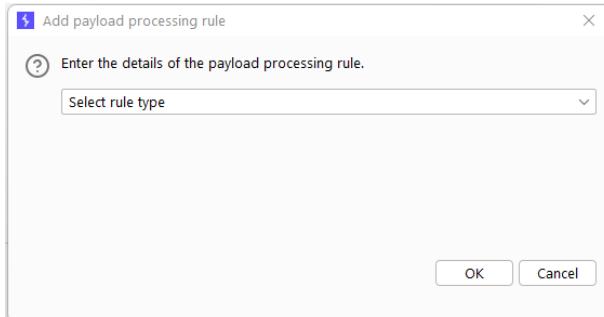
قسمت بعدی که توضیح ندادیم مربوط به Feature های تب Payload هست، دو تاش رو گفتیم ولی بعضیاش رو توضیح ندادیم . **Payload Processing**

#### **Payload processing**

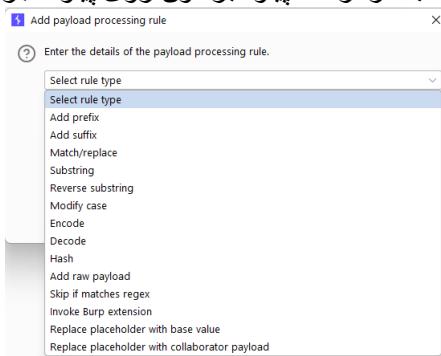
You can define rules to perform various processing tasks on each payload before it is used.



وقتی روی دکمه Add کلیک کنید پنجره زیر رو خواهد دید :



ما میتوانیم قبل از ارسال یک پیلود روی اون یک کارهایی رو انجام بدیم . مثلاً بایم و ... URL Encode, Base64، ... انجام بدیم روش . کافیه که از لیست زیر کاری که میخوایم رو بهش بگیم تا قبل از ارسال پیلود برآمون روی پیلود اجراش کنه :



تنظیمات Payload Encoding توبه Payload چیه و چیکار میکنه؟ زمانی پیش میاد که ما نیاز داشته باشیم پیلودمون رو URL Encode کنیم، برای بایپس کردن برخی مکانیزم های امنیتی مثل WAF کاربرد داره، شاید بخوایم مشخص کنیم که کدوم کاراکتر ها رو URL Encode کن و کدوم ها رو نکن، برای اینکار میتوانیم از قسمت Payload Encoding استفاده کنیم و برآش مشخص کنیم که چه کاراکتر هایی رو در صورتی که میخوایی URL Encode کنی، کدوم ها رو نکن :

#### **Payload encoding**

This setting can be used to URL-encode selected characters within the final payload, for safe transmission within HTTP requests.

URL-encode these characters: #

از این جهت مهم هست که گاهی لازم هست عبارات توی یک پیلود URL Encode بشن و گاهی نیاز هست که بدون Encode شدن و اصل پیلود ارسال بشه .

مد حمله والا ربط داره بزارید توضیح بدم . فرض کنید که یک Base Request داریم که چندین پارامتر داره که به سمت تارگت میفرسته ، این پارامتر ها مقادیری دارند . اگه ما نوع حملمن رو Battering Ram بزاریم و این چند پارامتر رو توی درخواست به عنوان Position انتخاب کنیم، در هر درخواست، هر مقداری از پیلود رو که میگیره، این مقدار رو به عنوان مقدار تمام پارامتر ها اتخاذ میکنه و بعد درخواست رو میفرسته . خب حالا چه ربطی به دژکوب داره؟ یعنی طورایی می شه فهمید که وقتی همه پارامتر ها رو پر میکنه یعنی طورایی میشه ارتباط ما بینشون دید، من که میبینم حالا شما نمیبینی مشکل از توعه(:)). حالا اینی که گفتم یعنی چی؟ فرض کنید که Base Request ما به شکل زیر هست :

```
# Base Request
POST /login.php HTTP/1.1
Host: site.com
...
Accept-Language: en-US,en;q=0.9
Cookie: csrfToken=p9hkoLwPNxY3ljrglMym2FpNbPsvqHoN; PHPSESSID=erfuf11382hn38m5qsc22et1hh
Connection: close

username=$username$&password=$password$
```

ما دو پارامتر username و password رو به عنوان محل قرار گیری پیلود انتخاب کردیم . لیست پیلود های ما هم به شکل زیر هست :

```
# Payloads
admin
user1
user2
user3
user4
```

وقتی حمله انجام میشه درخواست های ما به شکل زیر پیلود هامون رو توی درخواست ها قرار میدن :

```
# Requests
username=admin&password=admin
username=user1&password=user1
username=user2&password=user2
username=user3&password=user3
username=user4&password=user4
```

یعنی اینکه اولین پیلود که انتخاب میشه توی هر دو Position قرار میگیرند و ارسال میشوند .

حالا این چه کاربردی داره؟ بیشتر به نظرم برای Fuzzing استفاده میشه یا زمانی که میخوايد روی همه پارامتر ها یک سری پیلود رو به صورت مساوی تست کنیم، مثلا SQLi یا Directory Traversal و ... رو .

توی هر دو مد Sniper و Battering Ram ما یک گروه پیلود خواهیم داشت ولی دو مد بعدی حاوی بیشتر از یک گروه پیلود نمیتونن باشند .

- درمورد Password Attack ها و انواعش صحبت کردیم . حال وقت اینه که درمورد روش جلو گیری از این حملات صحبت کنیم . چه کارهایی رو نمیتونیم انجام بدیم تا یک مهاجم نتونه Password Attack انجام بده ؟
- Use CAPTCHAs : درمورد کپچا در ایندی همین جزو صحبت کردیم . اگه از یک کپچای خوب که قابل باپیس نباشه و مکانیزم هاش رو درست پیاده سازی کرده باشیم استفاده کنیم، نمیتونه جلوی Password Attack از هر نوعش رو بگیره .
- Rate Limit Mechanism : اگه بیایم و این مکانیزم رو توی WAF پیاده سازی کنیم، مهاجم نمیتونه تعداد زیاد درخواست رو به سمت وب اپلیکیشن بفرسته و به همین خاطر در انجام حملات Password Attack مهاجم، خلل ایجاد خواهد شد و البته بگم که صد درصد از بین نمیبره احتمال حمله رو .
- IP Blocking (Time-Base) : این مکانیزم هم باید توی WAF اجرا بشه و درصورتی که تعداد زیاد درخواست اوmd و مشکوک بود WAF باید بیاد و IP Address کلاینت رو برای یک مدت زمان مشخص بلاک کنه . این مورد هم نمیتونه صد درصد جلوی حملات Password Attack رو بگیره ولی نمیتونه احتمال انجامش رو کاهش بده .

• این مورد از موثر ترین روش‌هاست، اگه به کاربر اجازه ندیم که پسورد ضعیف استفاده کنه و مجبورش کنیم که یک پسورد طویل و پیچیده استفاده کنه، احتمال اینکه یک مهاجم از طریق **Password Attack** بتوانه به این کلمه عبور دسترسی پیدا کنه خیلی کم میشه پس مشکل رو حل میکنه.

نکته ای نیاز نیام بگم اینه که، توی لیست کلمات عبورتون برای حملات **Password Attack** در صورتی که اپلکیشن تارگتتون ممکن هست که کلمات عبور پیش فرض داشته باشه، حتما توی **Wordlist** که تهیه میکنید قرارش بدید چون احتمال اینکه این کلمه عبور تعییر نکرده باشه وجود داره.

مد Pitchfork چیه؟ از لحاظ لغوی Pitchfork به چیزایی مثل چنگال میگن. حالا اینجا چیکار میکنه؟ توی این مد ما میتوونیم هر چند تا مجموعه پیلود که بخوايم داشته باشیم ولی باید تعدادشون یکسان باشه، اگه نباشه اضافی ها رو تست نمیکنه. فرض بگیرید که یک Base Request به شکل زیر داریم:

```
# Base Request
POST /login.php HTTP/1.1
Host: site.com
...
Accept-Language: en-US,en;q=0.9
Cookie: csrfToken=p9hkoLwPNxY3ljrglMym2FpNbPsvqHoN; PHPSESSID=erfufl1382hn38m5qsc22et1hh
Connection: close

username=$username$&password=$password$&captcha=$captcha$
```

بعد سه تا هم گروه پیلود داریم به شکل زیر:

```
# Payloads1
admin
user1
user2
user3
user4
# Payloads2
pass1
pass2
pass3
pass4
pass5
pass6
# Payloads3
captcha1
captcha2
captcha3
captcha4
captcha5
captcha6
captcha7
captcha8
```

وقتی مد Pitchfork رو انتخاب کنیم و پیلود های بالا رو هم تعیین کنیم، درخواست هامون به شکل زیر ارسال میشن:

```
# Requests
username=admin&password=pass1&captcha=captcha1
username=user1&password=pass2&captcha=captcha2
username=user2&password=pass3&captcha=captcha3
username=user3&password=pass4&captcha=captcha4
username=user4&password=pass5&captcha=captcha5
```

يعني اینکه توی گروه دوم پیلود هامون pass6 حذف و توی گروه سوم پیلود هامون captcha6, captcha7, captcha8 حذف میشن و تست خودش میده. در این مد وقتی که یک گروه پیلود داریم همون حالت Sniper خواهد بود.

مد Cluster Bomb چیه؟ Cluster Bomb به معنی بمب خوشه ای هست، مدعی هست که تقریبا همیشه ازش استفاده میکنیم، این چهار نوع مد که توضیح دادم، Cluster Bomb و Sniper استفاده خواهیم کرد و کمتر پیش میداریم که از Cluster Bomb استفاده کنیم. در این مد هم ما هر تعداد که بخوایم میتوانیم مجموعه پیلود داشته باشیم و بسته به تعداد هایی داره که تعریف میکنیم. فرض بگیرید که یک Base Request به شکل زیر داریم:

```
# Base Request
POST /login.php HTTP/1.1
Host: site.com
...
Accept-Language: en-US,en;q=0.9
Cookie: csrfToken=p9hkoLwPNxY3ljrglMym2FpNbPsvqHoN; PHPSESSID=erfufl1382hn38m5qsc22et1hh
Connection: close

username=$username$&password=$password$&captcha=$captcha$
```

سه گروه پیلود هم تعریف میکنیم به شکل زیر :

```
# Payloads1
admin
user1
user2
user3
user4
# Payloads2
pass1
pass2
pass3
pass4
pass5
pass6
# Payloads3
captcha1
captcha2
captcha3
captcha4
captcha5
captcha6
captcha7
captcha8
```

در این حمله، تعداد  $Payload1 * Payload2 * Payload3$  تا درخواست ارسال میشه . یعنی طبق پیلود های ما تعداد درخواست های ما میشه  $240 = 8 * 6 * 5$  درخواست . چطوری درخواست ها رو ارسال میکنه؟ اولین پیلود از اولین گروه رو انتخاب میکنه، دومین پیلود از دومین گروه رو انتخاب میکنه و سپس تمام پیلود های گروه سوم رو برای اونها تست میکنه، بعد میاد و اولین پیلود از اولین گروه رو انتخاب میکنه، دومین پیلود از دومین گروه رو انتخاب میکنه و سپس تمام پیلود های گروه سوم رو روی اونها تست میکنه و همینطور میره تا آخرین پیلود از گروه اول، اخرین پیلود از گروه دوم که تمام پیلود های گروه سوم رو روش تست میکنه . درخواست های نمایی به شکل زیر خواهد داشت :

```
# Requests
username=admin&password=pass1&captcha=captcha1
username=admin&password=pass1&captcha=captcha2
...
username=admin&password=pass1&captcha=captcha8
username=admin&password=pass2&captcha=captcha1
username=admin&password=pass2&captcha=captcha2
...
username=admin&password=pass2&captcha=captcha8
...
username=user4&password=pass1&captcha=captcha1
username=user4&password=pass1&captcha=captcha2
...
username=user4&password=pass6&captcha=captcha8
```

این مد رو میتوانیم توی حملات **Brute-Force** و **Password Spray** استفاده کنیم .