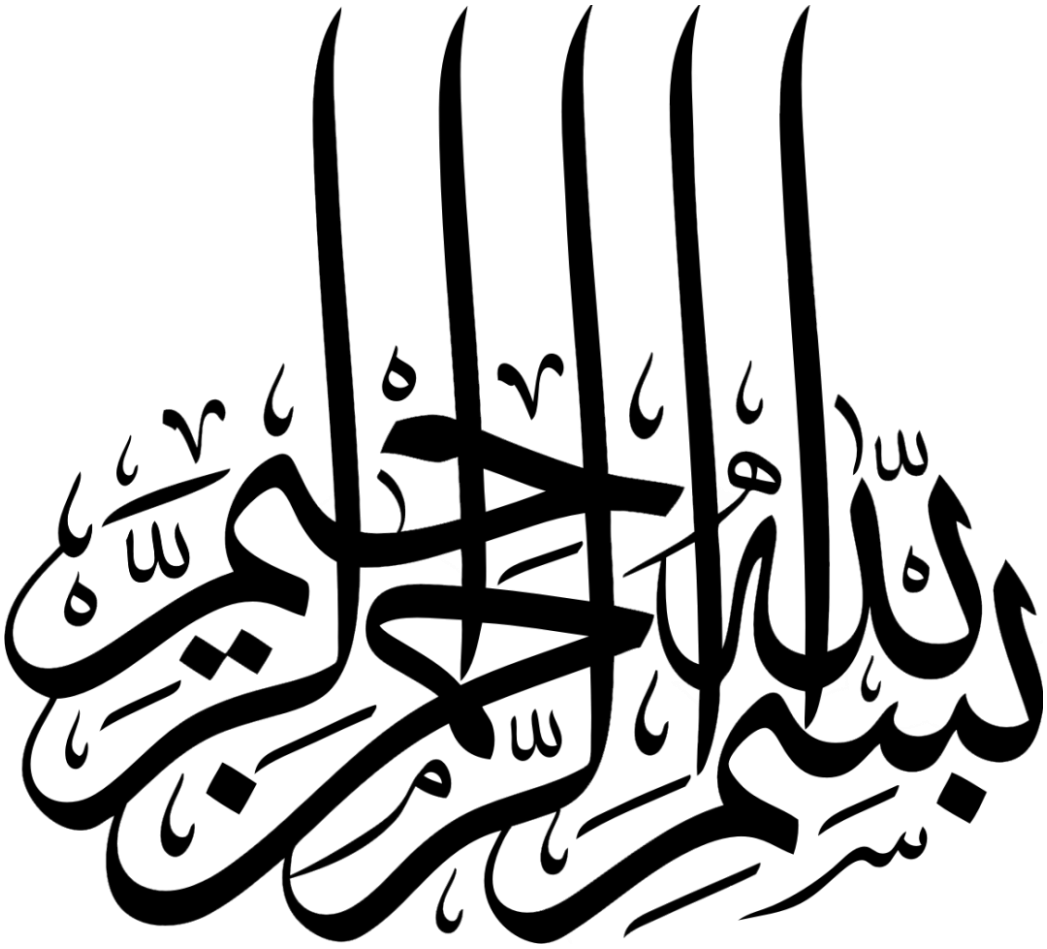

بسم الله الرحمن الرحیم

Laravel Apps Leaking Secrets

تهیه شده توسط تیم تولید محتوای وب سایت چلنجینو

فهرست مطالب

مقدمه

چند وقت پیش، یک تهدیدگر از طریق RDP وارد سیستم شد تا یک ابزار SMTP Cracker را اجرا کند. این ابزار، لیستی از آدرس‌های IP یا URLها را برای یافتن سیستم‌های Laravel که به درستی پیکربندی نشده‌اند، اسکن می‌کند. این مهاجمان به دنبال وب‌سایت‌هایی هستند که Debug Mode در آن‌ها فعال باشد، که به مهاجم اجازه می‌دهد فایل پیکربندی (.env) آن‌ها را مشاهده کند. فایل env. شامل اعتبارنامه‌های AWS، O365، SendGrid، Twilio و موارد دیگر است.

Laravel چیست؟

Laravel یک چارچوب رایگان و متن‌باز PHP برای توسعه برنامه‌های وب است که توسط تیلور اتول ایجاد شده است. (منبع: ویکی‌پدیا)

Laravel درایورهایی برای SMTP، Mailgun، Mandrill، Amazon SES، تابع mail در PHP و sendmail فراهم می‌کند و به شما امکان می‌دهد به سرعت ارسال ایمیل را از طریق یک سرویس محلی یا ابری دلخواه شروع کنید. (منبع: مستندات رسمی Laravel)

گزینه دیباگ به طور پیش‌فرض در سیستم‌های Laravel غیرفعال است، اما به نظر می‌رسد بسیاری از کاربران آن را فعال کرده و عواقب آن را درک نمی‌کنند. در اینجا نمونه‌ای از فعال بودن دیباگ (تنظیم روی true) در فایل پیکربندی env. نشان داده شده است.

```
<tr>
  <td>
    APP_DEBUG
  </td>
  <td>
    <pre class=sf-dump id=sf-dump-2067749625 data-indent-pad="  ">
      "<span class=sf-dump-str title="4 characters">true</span>
```

پاسخ وب (فایل پیکربندی واقعی نیست)

می‌توانید با بررسی وجود فایل env. در ریشه وب (site.com/.env) یا با ارسال داده‌های تصادفی به وب‌سرور و بررسی پاسخ، متوجه شوید که آیا دیباگ فعال است یا خیر. در پاسخ، گزینه دیباگ و همچنین تمام اطلاعات فایل env. که شامل secret ها می‌باشد را مشاهده خواهید کرد.

در اینجا نمونه‌ای از پاسخ وب با دیباگ فعال آورده شده است:

```
    AWS_ACCESS_KEY_ID
  </td>
  <td>
    <pre class=sf-dump id=sf-dump-1797855799 data-indent-pad="  ">
      "<span class=sf-dump-str title="20 characters">[redacted]</span>
      "
    </pre>
    <script>
      Sfdump("sf-dump-1797855799")
    </script>
  </td>
</tr>
<tr>
  <td>
    AWS_SECRET_ACCESS_KEY
  </td>
  <td>
    <pre class=sf-dump id=sf-dump-706539060 data-indent-pad="  ">
      "<span class=sf-dump-str title="40 characters">[redacted]</span>
      "
    </pre>
    <script>
      Sfdump("sf-dump-706539060")
    </script>
  </td>
</tr>
<tr>
  <td>
    AWS_DEFAULT_REGION
  </td>
  <td>
    <pre class=sf-dump id=sf-dump-840748221 data-indent-pad="  ">
      "<span class=sf-dump-str title="9 characters">us-east-1</span>
```

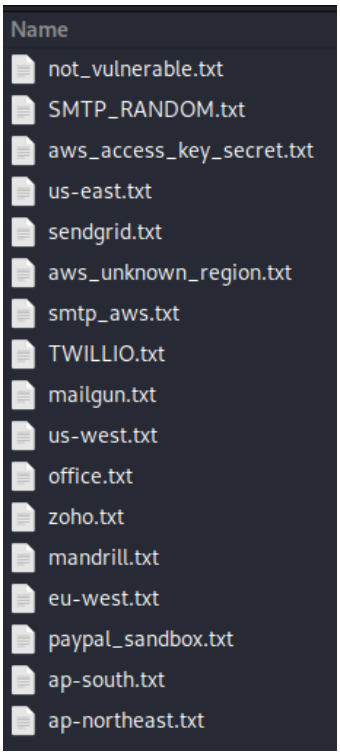
در اینجا نمونه‌ای از یک فایل env. آورده شده است:

اسکریپت SMTP Cracker که اتفاقاً یک Cracker نیست و فقط اعتبارنامه‌های SMTP را دریافت نمی‌کند، از روش‌های فوق برای خزیدن در لیستی از IPها/URLها استفاده می‌کند و به دنبال رشته‌های خاصی در پاسخ می‌گردد، مانند PayPal، AWS_KEY، SES_KEY، Twilio، sendgrid، office365، zoho، mailgun و دیگر موارد.

در اینجا بخشی از اسکریپت آورده شده است که در آن به دنبال یک رشته در mailhost می‌گردد و secretها را در فایل مناسب خروجی می‌دهد.

```
else:
        # mod aws
        if '.amazonaws.com' in mailhost:
                getcountry = reg('email-smtp.(.*?).amazonaws.com', mailhost)[0]
                build = 'URL: '+str(url)+'\nMETHOD: '+str(method)+'\nMAILHOST: '
                remover = str(build).replace('\r', '')
                save = open('Results/'+getcountry[:-2]+'.txt', 'a')
                save.write(remover+'\n\n')
                save.close()
                remover = str(build).replace('\r', '')
                save2 = open('Results/smtp_aws.txt', 'a')
                save2.write(remover+'\n\n')
                save2.close()
        elif 'sendgrid' in mailhost:
                build = 'URL: '+str(url)+'\nMETHOD: '+str(method)+'\nMAILHOST: '
                remover = str(build).replace('\r', '')
                save = open('Results/sendgrid.txt', 'a')
                save.write(remover+'\n\n')
                save.close()
        elif 'office365' in mailhost:
                build = 'URL: '+str(url)+'\nMETHOD: '+str(method)+'\nMAILHOST: '
                remover = str(build).replace('\r', '')
                save = open('Results/office.txt', 'a')
                save.write(remover+'\n\n')
                save.close()
        elif '1and1' in mailhost or '1und1' in mailhost:
                build = 'URL: '+str(url)+'\nMETHOD: '+str(method)+'\nMAILHOST: '
                remover = str(build).replace('\r', '')
                save = open('Results/1and1.txt', 'a')
                save.write(remover+'\n\n')
                save.close()
        elif 'zoho' in mailhost:
                build = 'URL: '+str(url)+'\nMETHOD: '+str(method)+'\nMAILHOST: '
                remover = str(build).replace('\r', '')
                save = open('Results/zoho.txt', 'a')
                save.write(remover+'\n\n')
                save.close()
        elif 'mandrillapp' in mailhost:
                build = 'URL: '+str(url)+'\nMETHOD: '+str(method)+'\nMAILHOST: '
                remover = str(build).replace('\r', '')
```

خروجی اسکریپت در پوشه‌ای به نام Results ذخیره می‌شود. نتایج به گروه‌های مختلفی تقسیم می‌شوند، همانطور که در تصویر زیر مشاهده می‌کنید.



در داخل آن فایل‌ها، secretها و روش استفاده شده برای استخراج اطلاعات قرار دارد. این یک مثال جزئی از فایل sendgrid است:

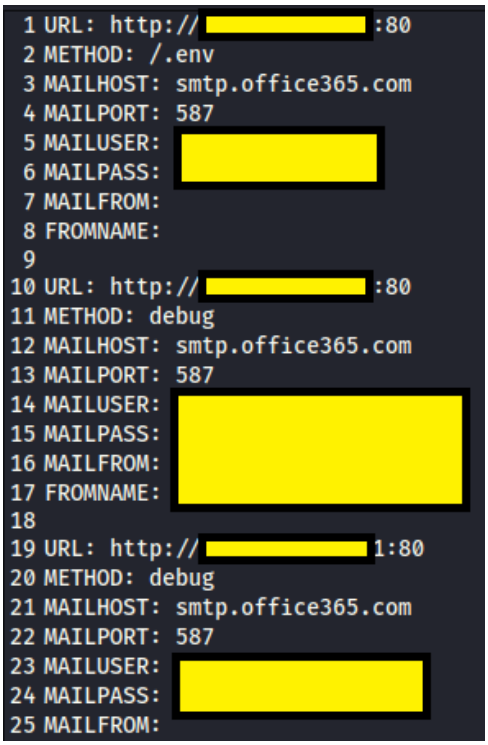
```
URL: http://[redacted]:80
METHOD: debug
MAILHOST: smtp.sendgrid.net
MAILPORT: 587
MAILUSER: [redacted]
MAILPASS: [redacted]
MAILFROM: [redacted]
FROMNAME: [redacted]

URL: http://[redacted]:80
METHOD: debug
MAILHOST: smtp.sendgrid.net
MAILPORT: 465
MAILUSER: apikey
MAILPASS: [redacted]
MAILFROM:
FROMNAME:

URL: http://[redacted]:80
METHOD: /.env
MAILHOST: smtp.sendgrid.net
MAILPORT: 587
MAILUSER: apikey
MAILPASS: [redacted]
MAILFROM:
FROMNAME:

URL: http://[redacted]:80
METHOD: debug
MAILHOST: smtp.sendgrid.net
MAILPORT: 587
MAILUSER: apikey
MAILPASS: [redacted]
MAILFROM:
```

این یک مثال جزئی از فایل متنی office365 است:



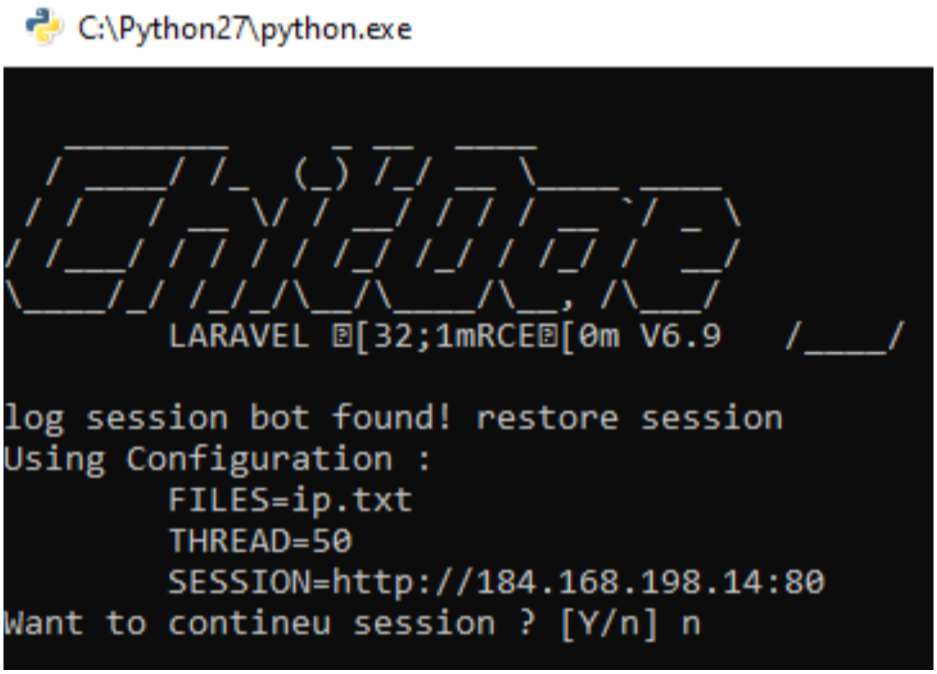
ما در حال تماس با بیش از ۱۰۰ نفر/سازمان هستیم که سیستم‌هایشان از طریق دیباگ Laravel موارد مهمی را نشت می‌دهند، به این امید که این مشکل را برطرف کرده و رمزهای عبور خود را تغییر دهند.

دسترسی اولیه

ورود RDP به عنوان Local Administrator از آدرس 64.86.198[.]22 بود و هیچ تلاشی مبنی بر Brute Force از این IP مشاهده نشد.

اجرا

مهاجم پایتون ۲.۷ را نصب و از آن برای اجرای اسکریپت SMTP استفاده کرد.



از Pip برای نصب ۴ ماژول استفاده شد.

```
pip  install requests
pip  install colorama
pip  install bs4
pip  install tldextract
```

commandLine	pip install bs4
currentDirectory	C:\\Python27\\Scripts\\
hashes	SHA1=3B8F65BDBDC2AE4D6785F5B294A4F495C3 026
image	C:\\Python27\\Scripts\\pip.exe
integrityLevel	High
logonGuid	{008cde44-071e-601f-0273-ce0f00000000}
logonId	0xfce7302
parentCommandLine	\"C:\\Windows\\System32\\cmd.exe\"

تشخیص‌ها (Detections)

درخواست‌های وب به مسیر root/.env یا داده‌های ارسالی (post data) که شامل 0x[]:androxgh0st (یک رشته Hardcode شده در smtp.py) باشد.

IOCs

File

filename	smtp.py
md5	39e1ec2c704bcdb57034da4ac288446e
sha1	352e9c78f08574adbaa0aaf49c19d5853bb4be36
sha256	478290f801dafc9086810f857c123d74690920a4c44fb573cd01463c1b6fb432

Network

type	value	comment
ip-src	64.86.198.22	RDP Unauthorized Access

```
Valid Accounts - T1078
Python - T1059.006
Scripting - T1064
User Execution - T1204
Graphical User Interface - T1061
Command and Scripting Interpreter - T1059
```

منبع:

https://thedfirreport.com/2021/02/28/laravel-debug-leaking-secrets/