**T.C.**

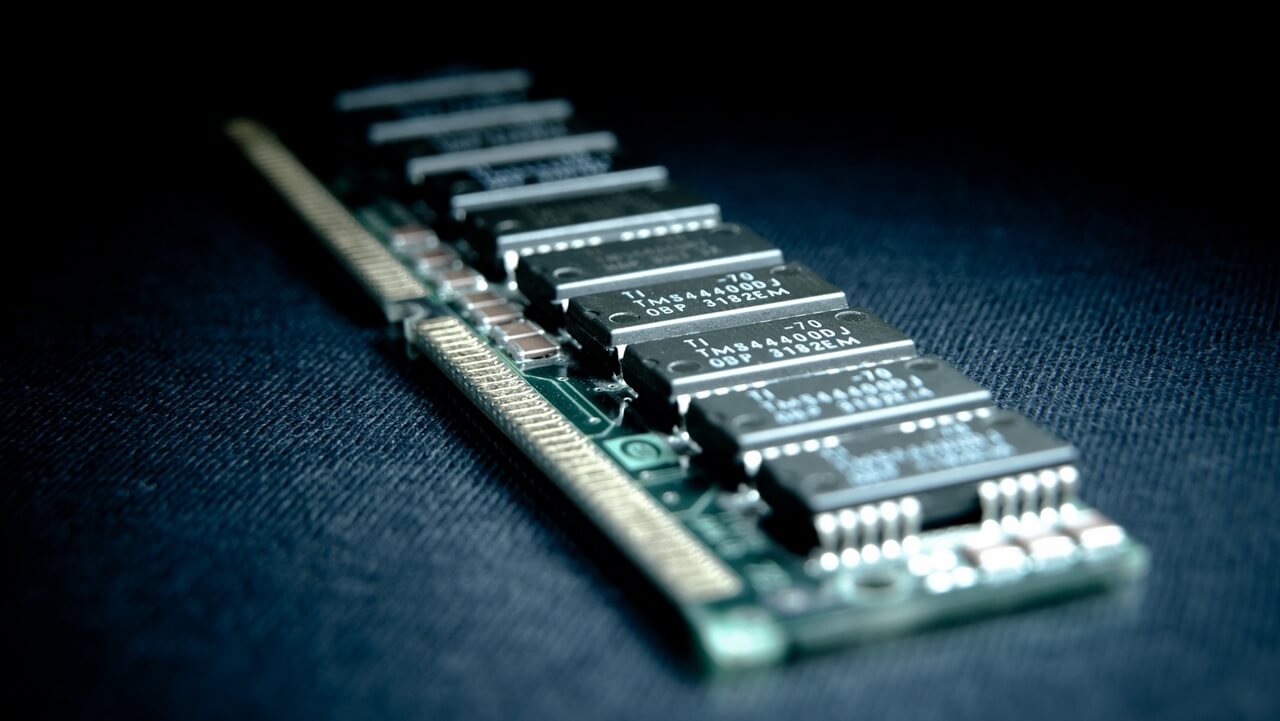
**YOZGAT BOZOK ÜNİVERSİTESİ MÜHENDİSLİK MİMARLIK FAKÜLTESİ BİLGİSAYAR MÜHENDİSLİĞİ**

****

**STACK-BUFFER OVERFLOW**

**Ahmet REÇBER**

**1-RAM NEDİR ?**



**RAM** bilgisayarın kısa süreli hafıza görevini üstlenen, oldukça hızlı bir bileşendir. RAM içerisinde anlık olarak veriler depolanır. Geçici olarak depolanan veriler, işlendikten sonra gerekli bileşene gönderildiği anda silinir. Silinen verilerin yerine ihtiyaç duyulan **yeni veriler**getirilir. Cihaz üzerinde yapılan her bir işlemde bu döngü tekrarlanır.

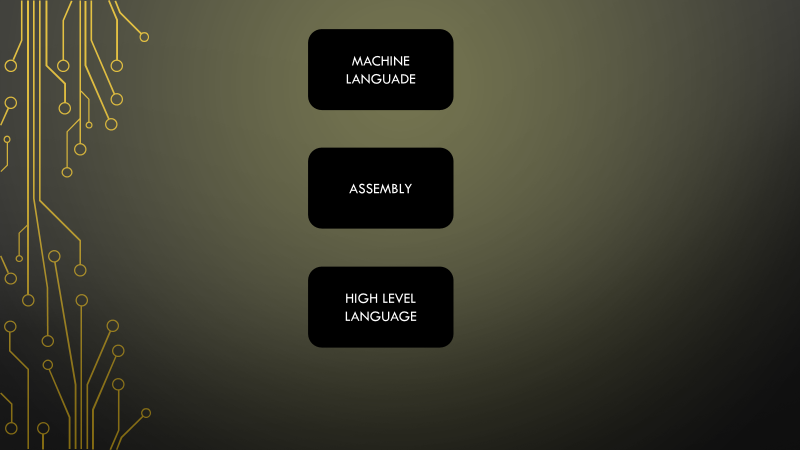
**1.1 RAM’İN İÇ YAPISI**



**2.STACK VE İÇ YAPISI**

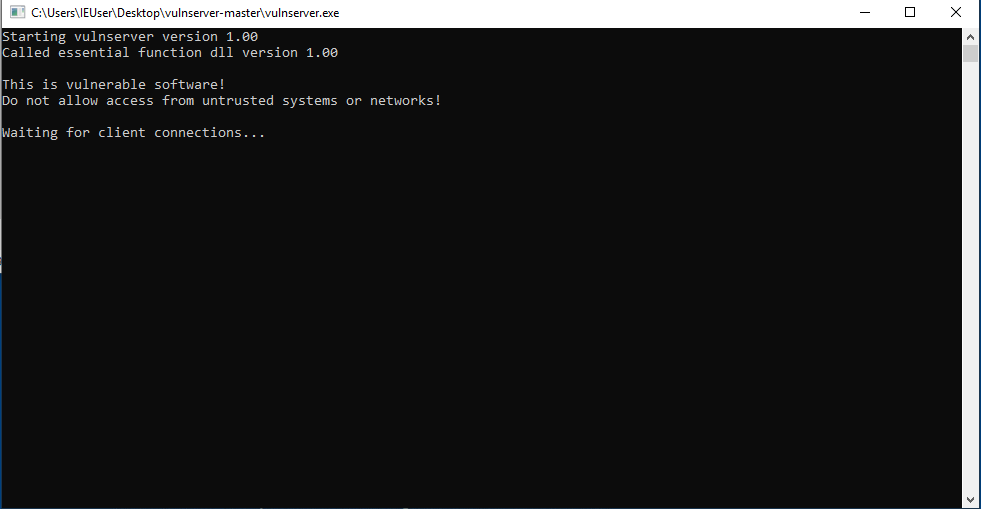


**3.BİNARY**



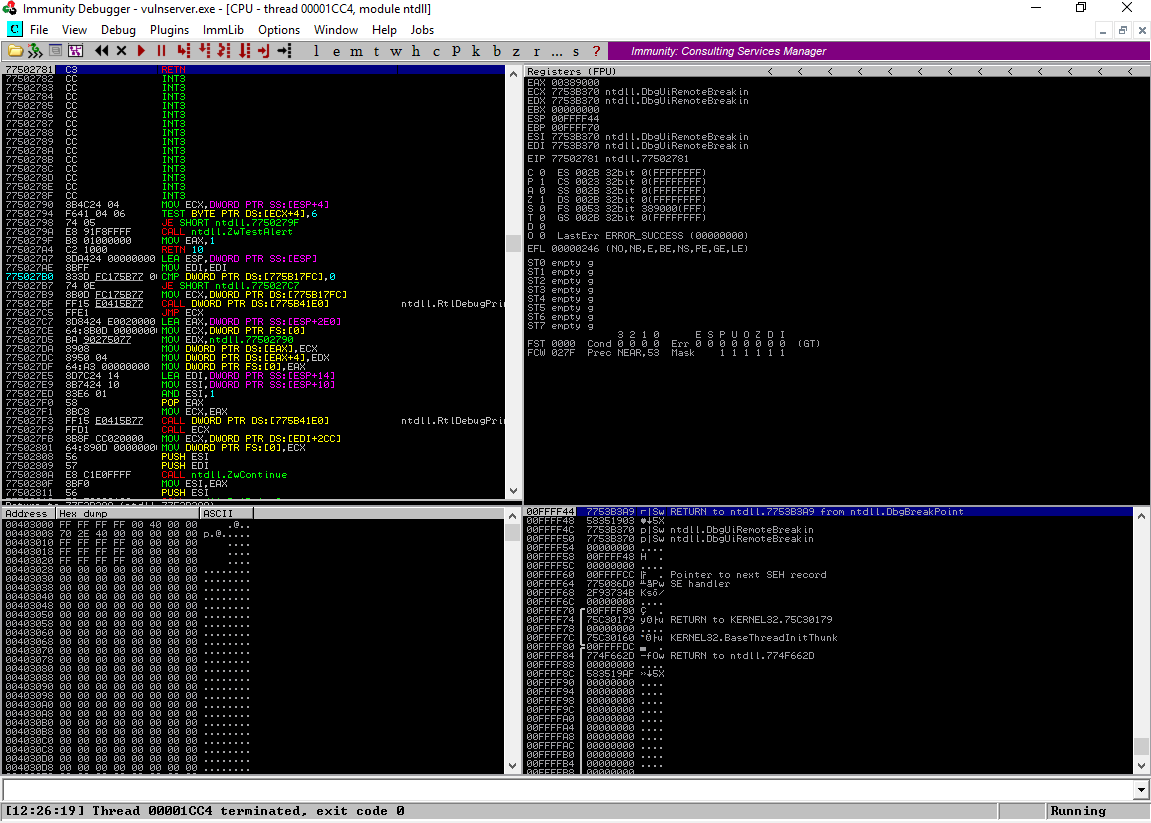
**4-VULNSERVER**

Buffer overflow saldırısı için kullanacağımız sunucu.

****

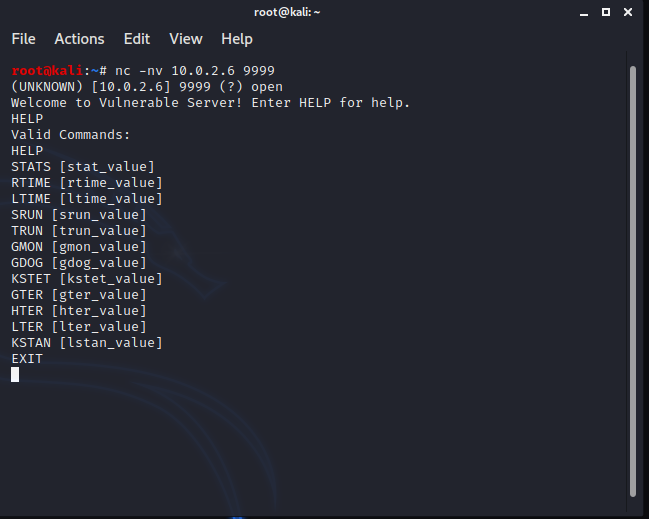
**5-** **İMMUNİTY DEBUGGER**

İmmunity Debugger tersine mühendislik için kullanılan bir araçtır. Programı açtıktan sonra sırasıyla File>Attach dedikten sonra kullanmak istediğimiz sunucuyu seçip play tuşu ile running işlemini başlatıyoruz.

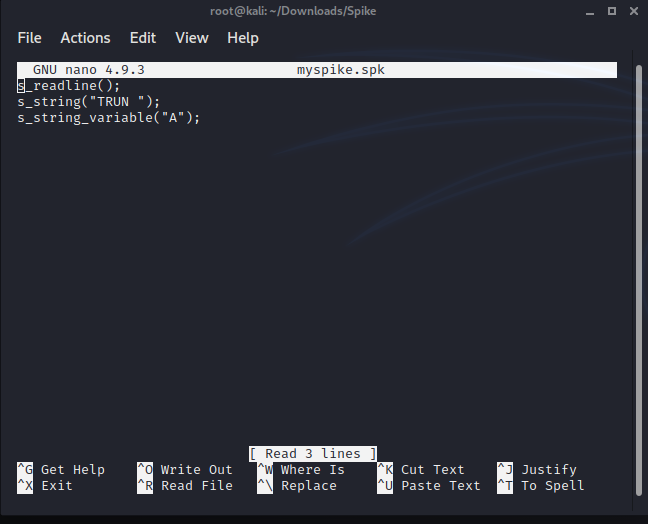
****

**6-SPIKE**

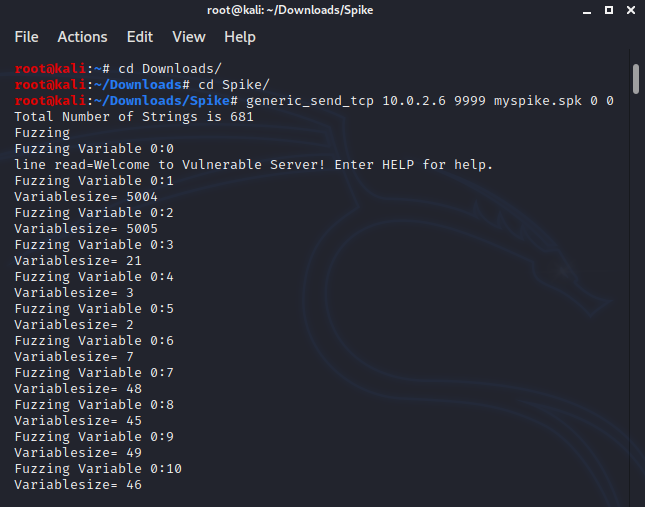
**Adım1 :** Netcat ile sunucuya bağlanıyoruz ve kullanabileceğimiz opsiyonları görüyoruz.

****

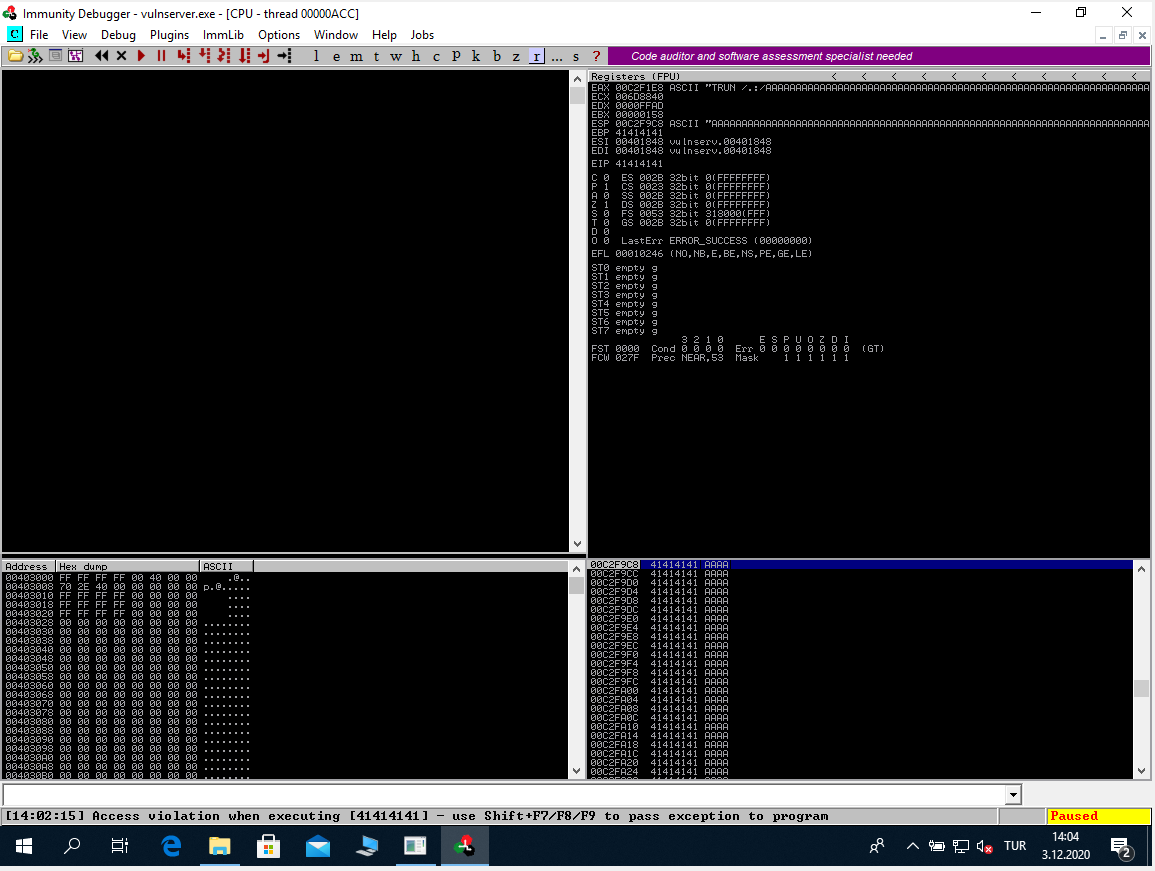
**Adım2 :** Bu açıkları tek tek denedikten sonraTrun ile bir açık yakalıyoruz ve bu açık için küçük bir script yazıyoruz.

****

**Adım3 :** Bu scripti çalıştırarak sunucuya sürekli bir veri gönderiyoruz.

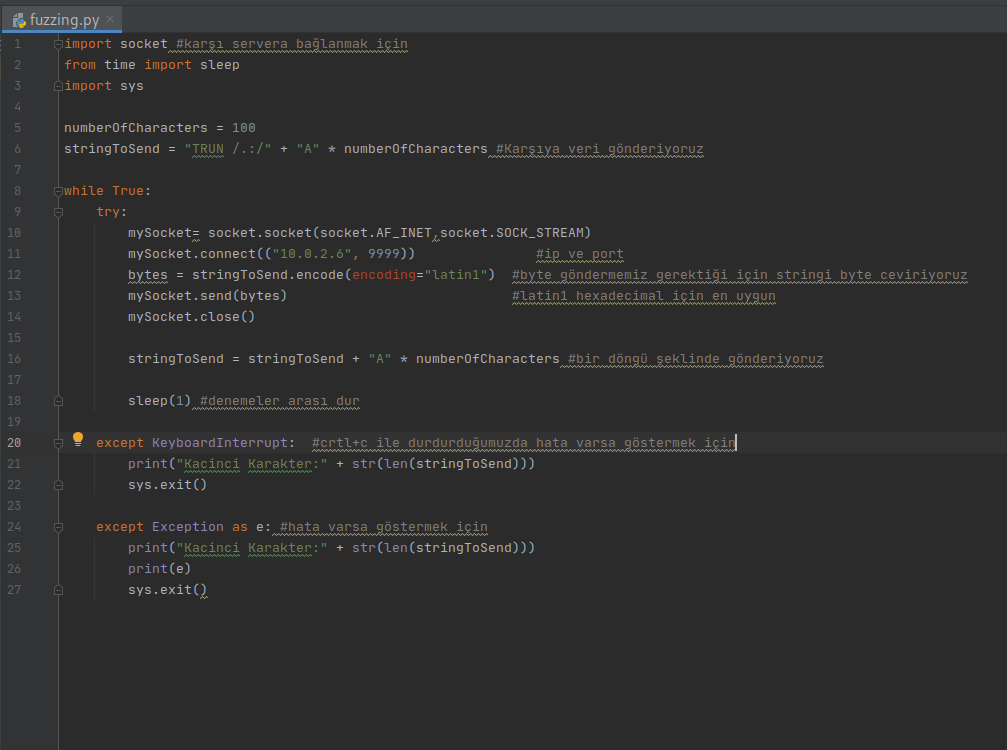
****

**Adım4:** İmmunity Debuggerı açıp baktığımızda sunucunun ve çökmüş olduğunu fark ediyoruz ve bu açığı kullanabiliriz.

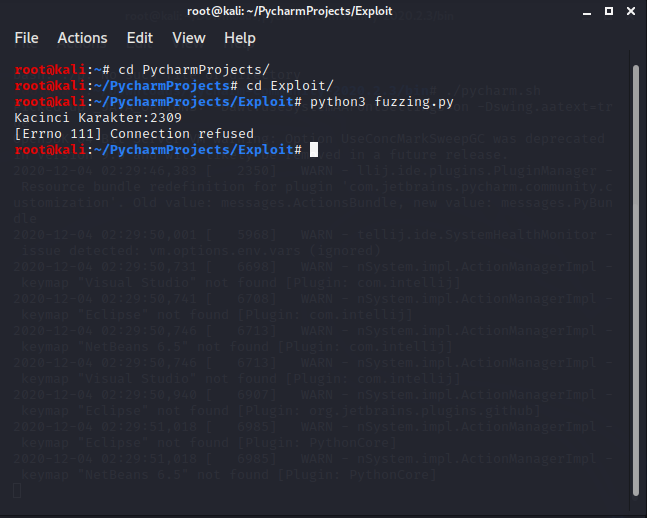
****

**7-Fuzzing**

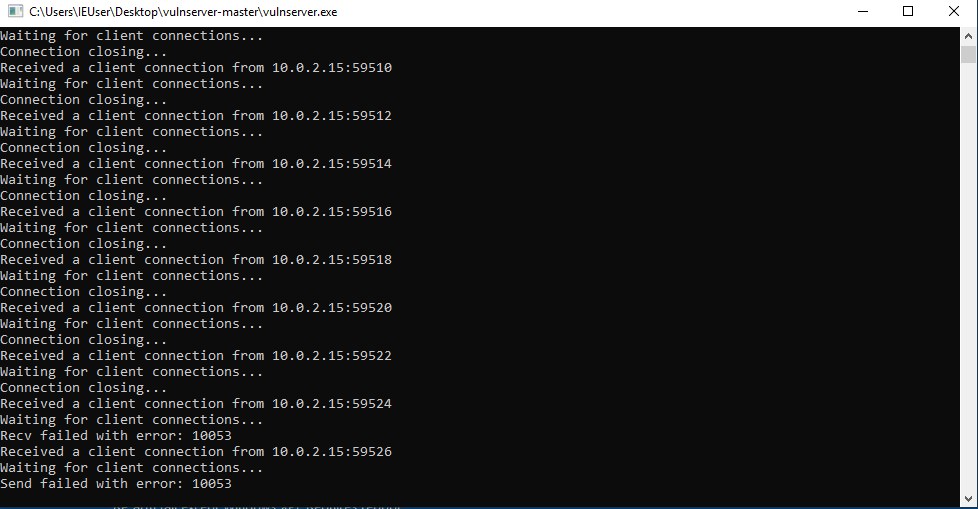
Sunucumun çökebildiğini anladım fakat kaç karakter ile ve kaçıncı karakterden sonra çöktüğünü bilmem gerekiyor bunun için fuzzing işlemini yapıyorum.



Yazdığımız python3 kodunu çalıştırıyoruz ve artık kaçıncı karakterden sonra çöktüğünü öğreniyoruz.

****

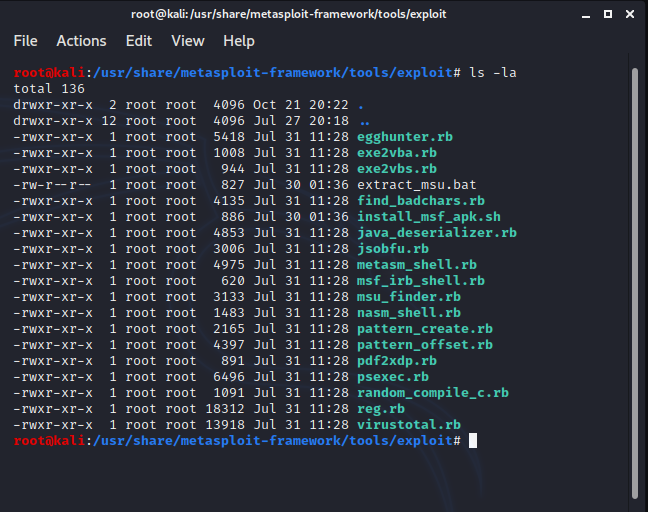
Windows makinemizdeki sunucu istekleri aldıktan bir süre sonra çöküp kapanıyor.

****

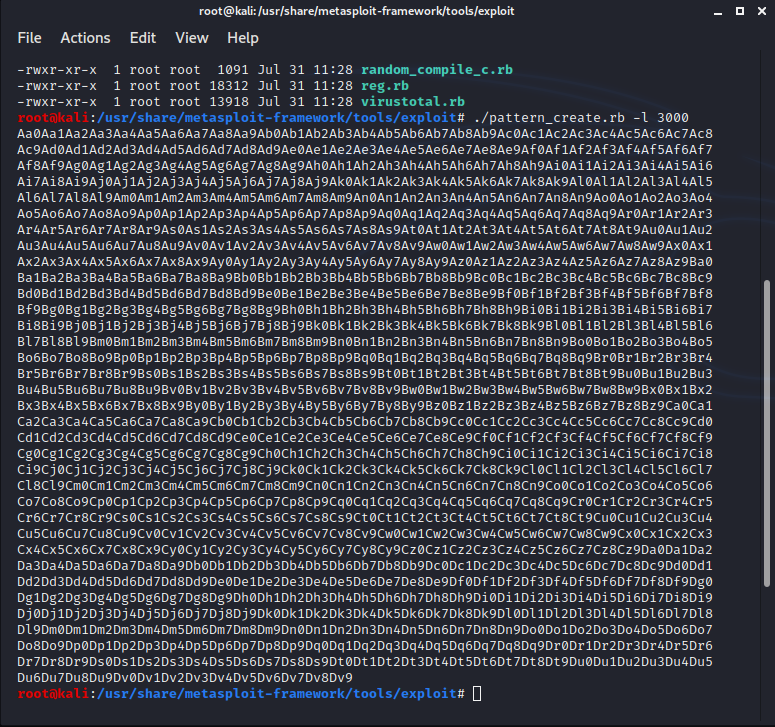
**8-Offset**

Ram içindeki EIP de tam olarak nerede çöktüğünü bulmak için yazdığımız script.

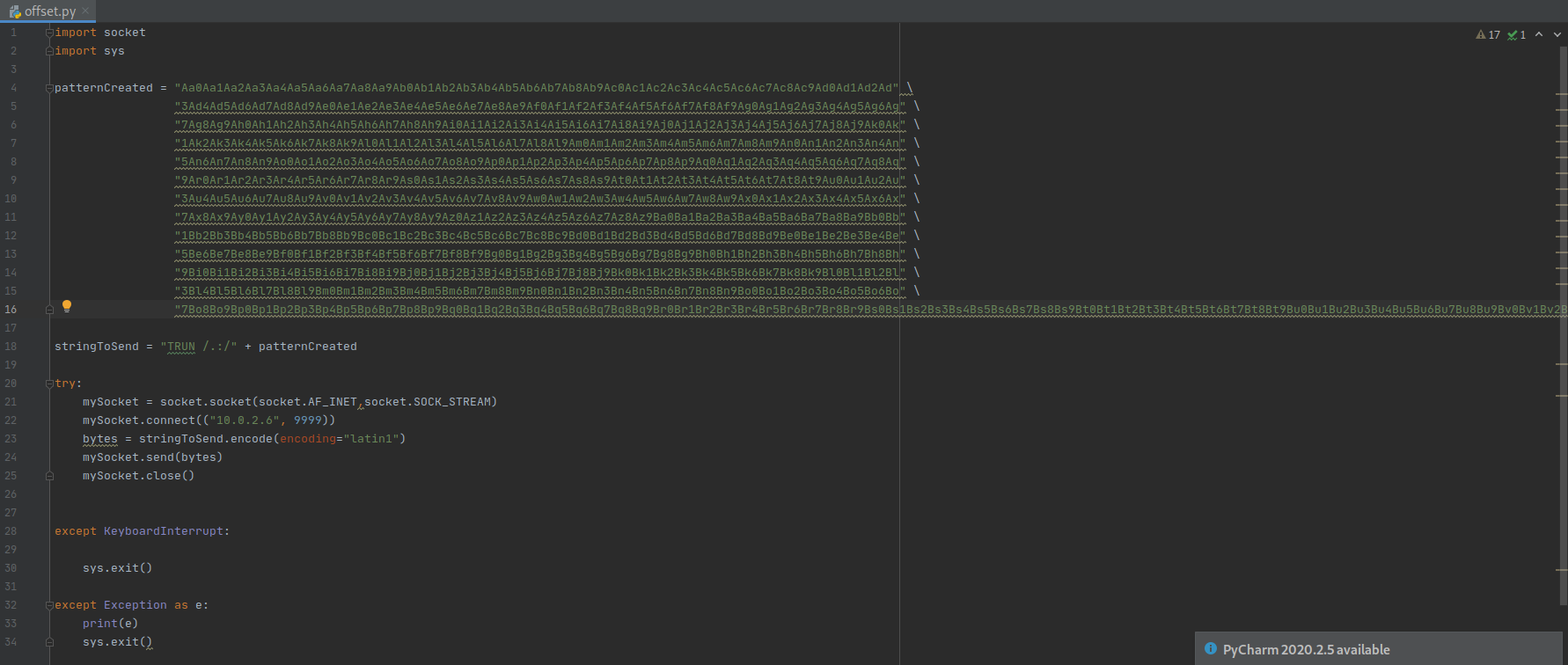
**Adım1** : Kod yazmaya başlamadan önce random bir pattern alıyoruz. Kali linux da metasploit içinde böyle bir araç mevcut bunu yapmamızdaki amaç tam olarak kaçıncı karakterde çöktüğünü bulmak.

****

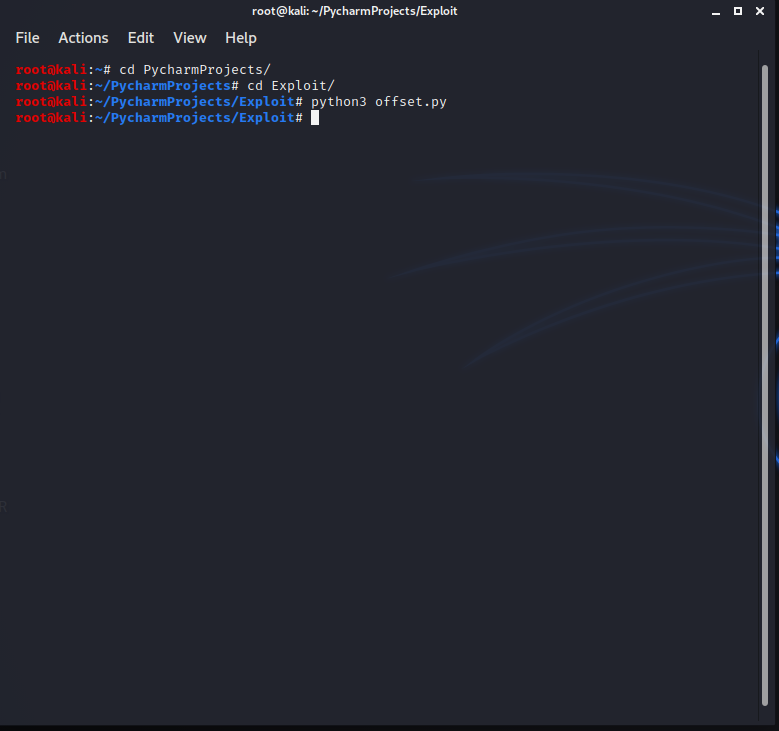
**Adım2:** 3000 karakterden oluşan bir wordlist elde ediyoruz ve bunu kopyalıyoruz.

****

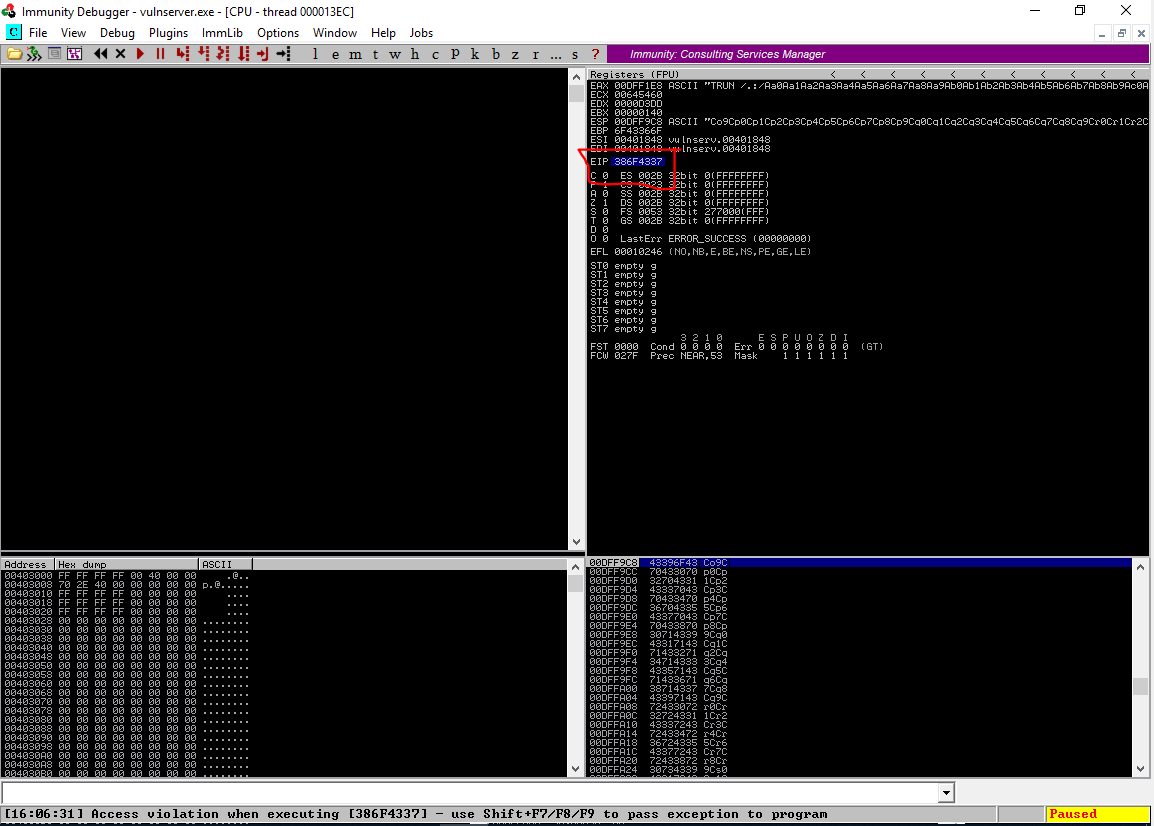
Kod parçasının son hali aşağıdaki gibidir.

****

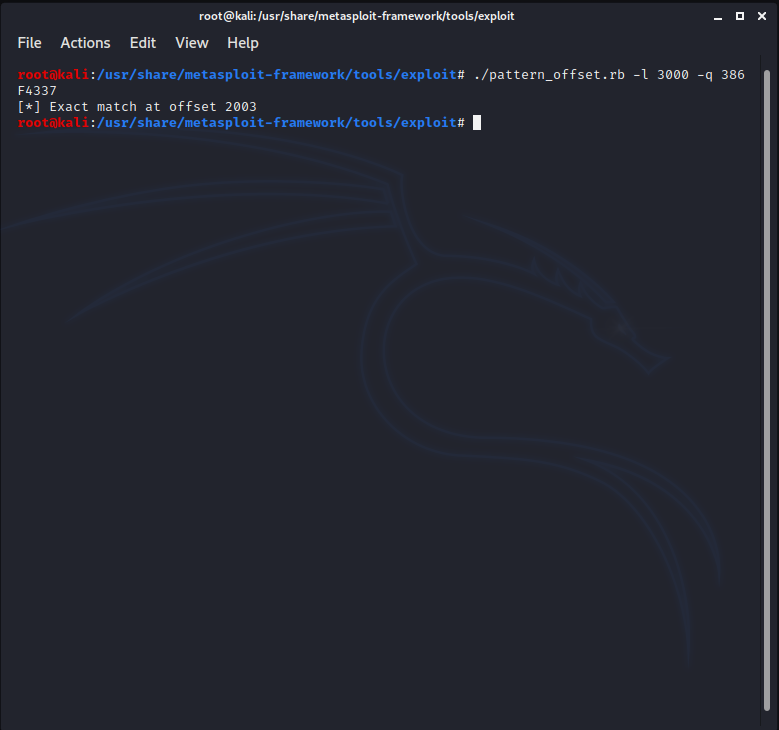
**Adım3:** Offset komutumuzu çalıştırıyoruz tabii ki daha önceden serverimiz açıp İmmunity Debuggerı bağlamamız gerekmektedir.



Sonuç olarak gördüğümüz gibi server çökmüş ve bufferdan taşan EIP de 386F4337 değeri bulunmaktadır.

****

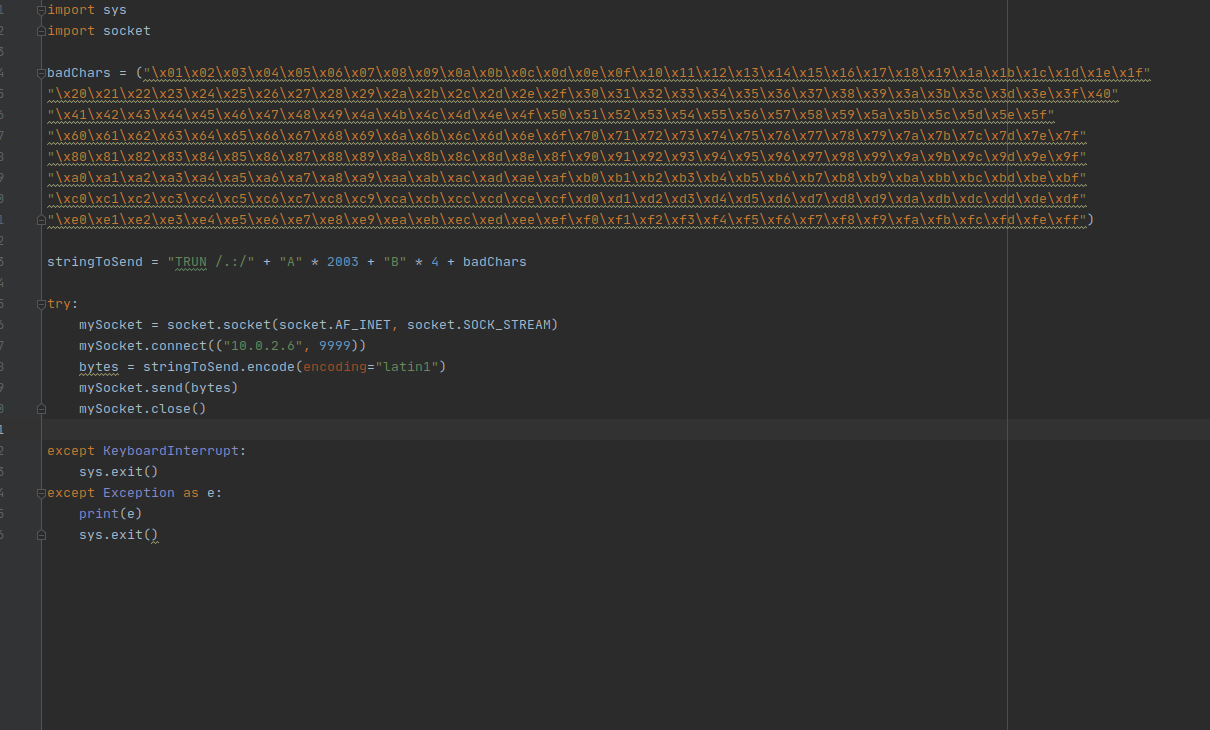
Bu değeri gene metasploit içindeki pattern ile tam olarak kaçıncı pattern olduğunu öğreniyoruz ve sonuç 2003 olarak veriyor.

****

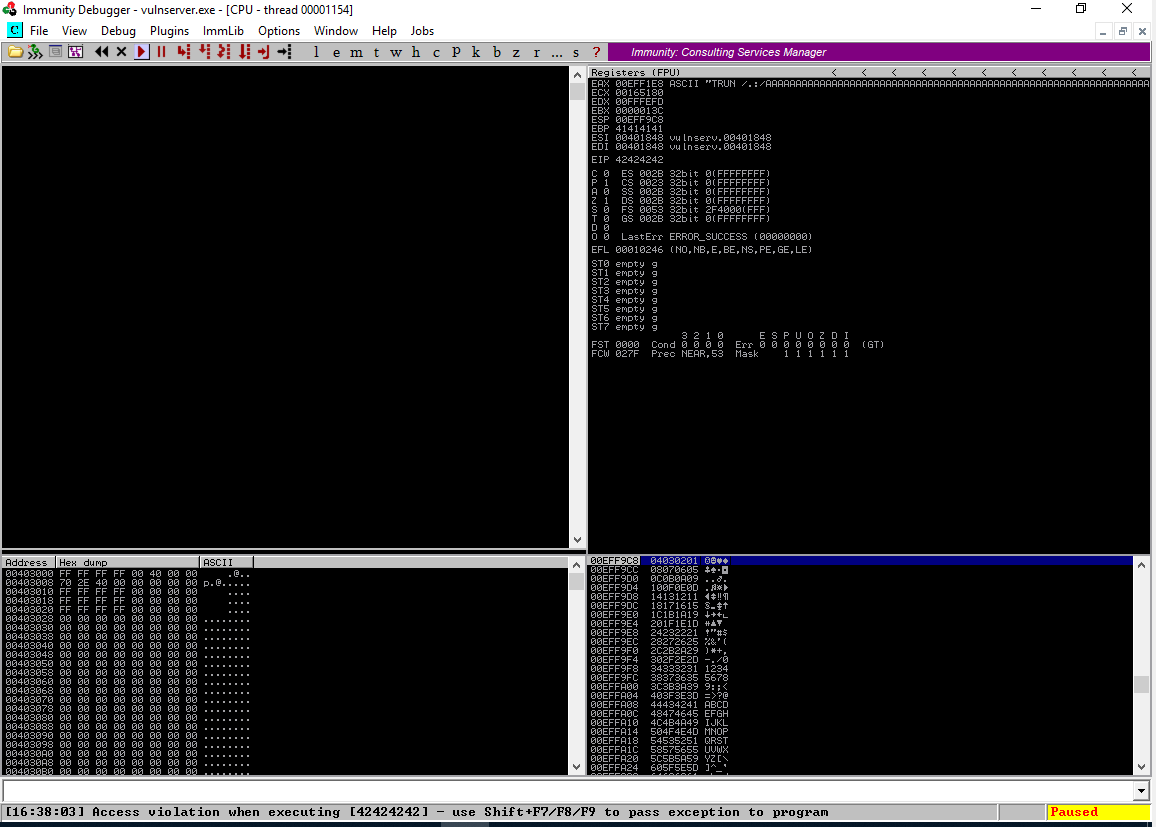
**9-Kötü Karakterler**

Şimdi tam olarak kaçıncı karakterden sonra EIP ye geçtiğini öğrendik ve bu karakterden sonra istediğimiz shellcodeları çalıştırabiliriz. Fakat kötü karakter dediğimiz bazı karakterler burada çalışmayabiliyor bunları tespit edebilmemiz için bir script yazmamız gerekmektedir.

Bunun için internetten hazır bir kötü karakter diziniz alıp yazdığım script aşağıdadır.

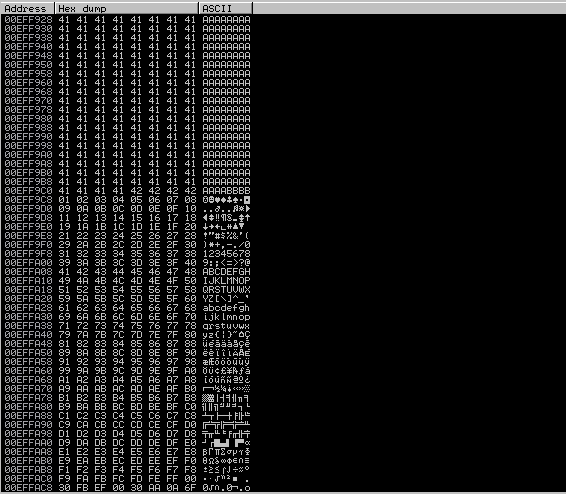


Bu Scripti çalıştırdıktan sonra ESP A yani 41 ve EIP değeri B yani 42 olmustur.



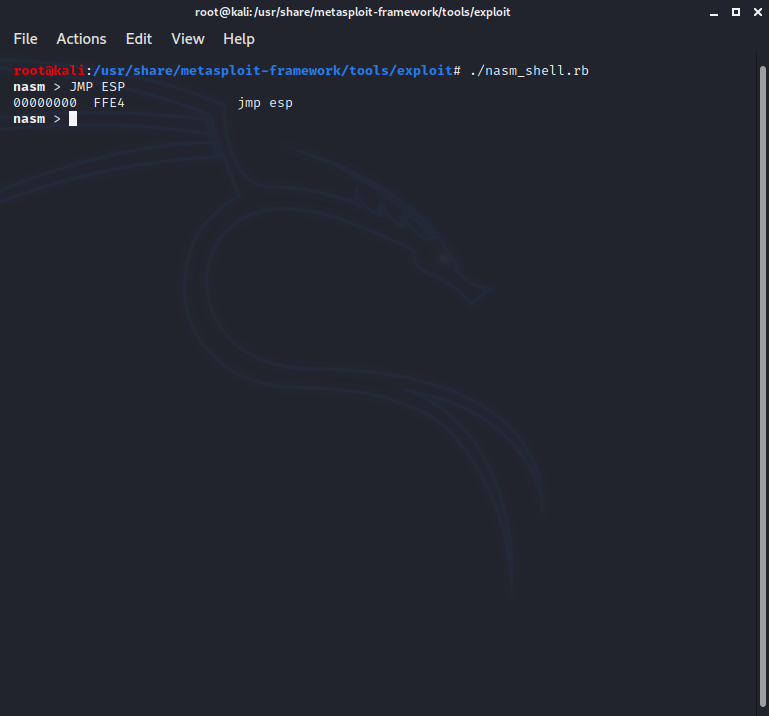
Şimdi kötü karakterleri öğrenmek için ESP’nin üzerine sağ tıklayıp Follow in Dump’a basıyoruz. Gönderdğimiz 2003 A ve 4 B den sonra 01 den FF e kadar tüm kötü karakterler bulunmaktadır eğer sırayı bozan bir karakter varsa onu not alıp shellcode yazarken kullanacağız.

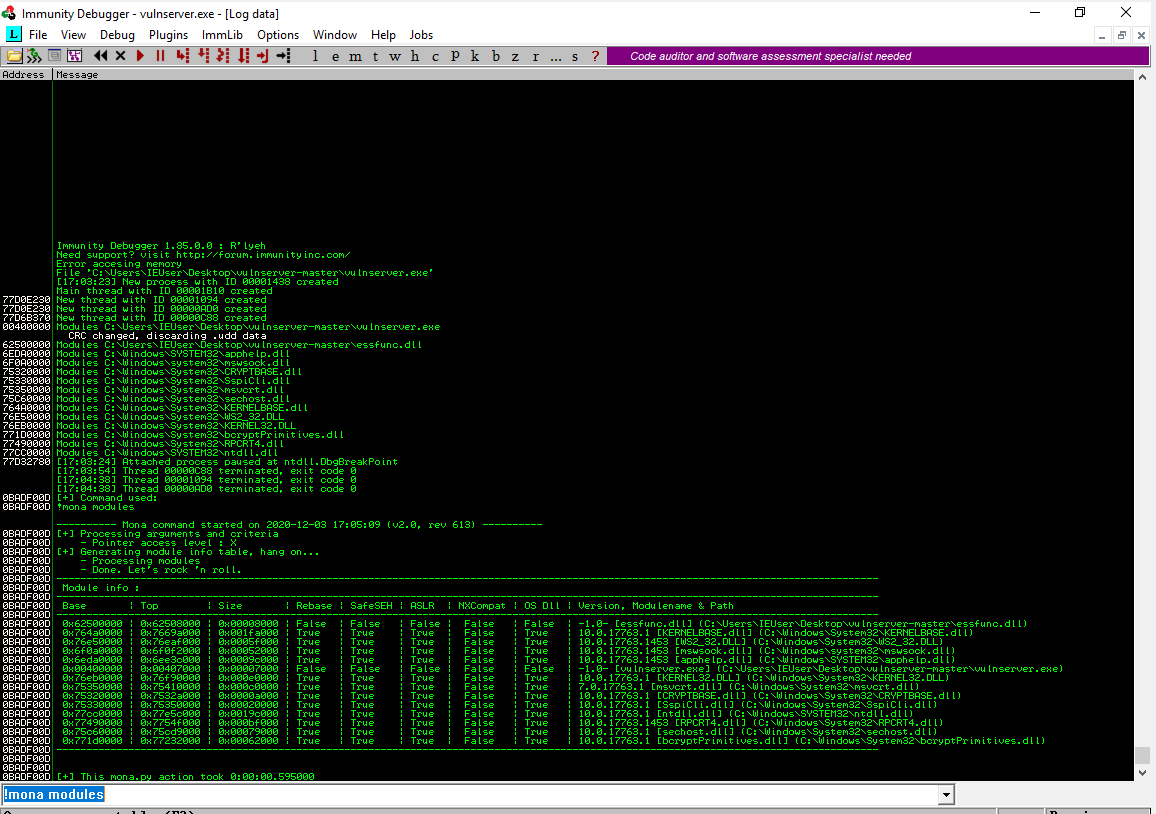
Bu listeye göre tek bir tane var oda x00’dır.



**10-Shellcode öncesi**

EIP ye B yerine kendi Shellcodemuzu göndereceğiz. Lakin bundan önce JMP ESP ile gene metasploit ile bir araç kullanarak assembly kodlarını hex koduna çevirerek bir karakter elde etmeliyiz. Bu karakteri Mona.py olan bir script üzerinde kullanacağız.

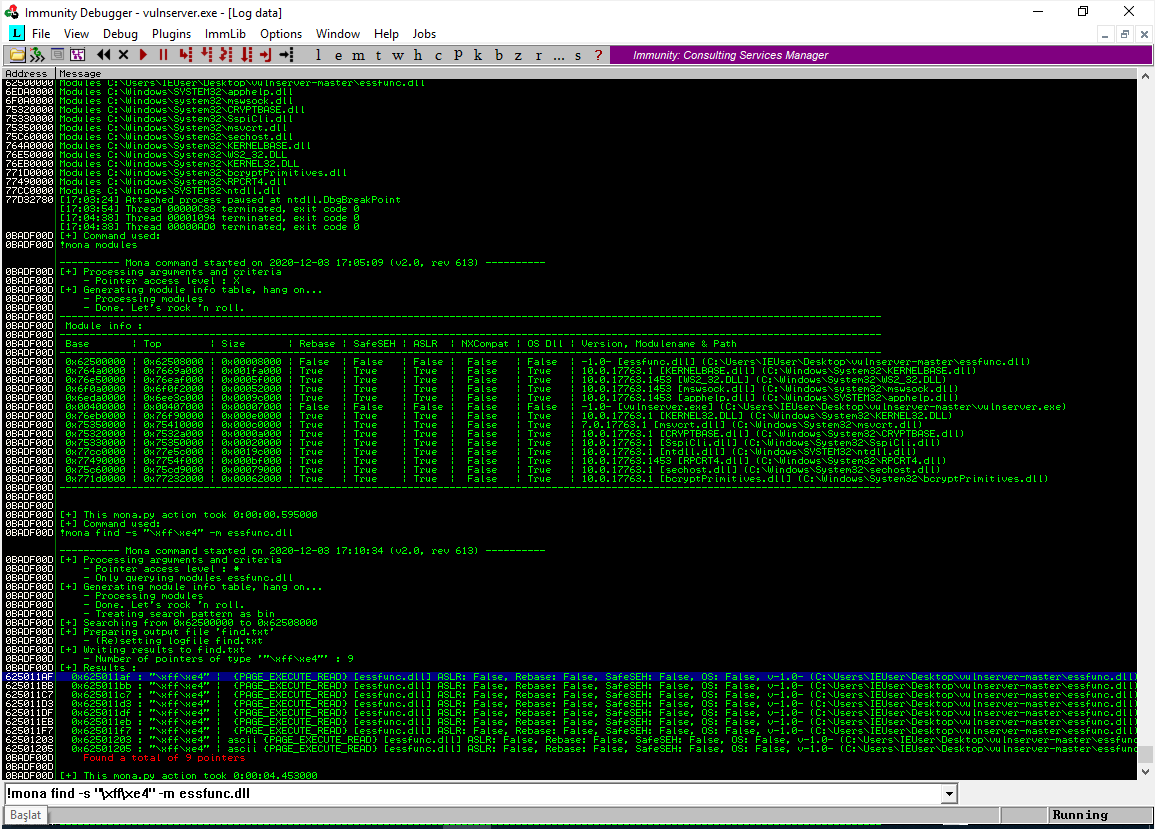




Mona.py dosyasını Immunity Debugger dosyalarının içine attıktan sonra programı açıyoruz ve alt kısımda bulunan alana **!mona modules** yazarak tüm windowsda ki çalışan servisleri görüyoruz. Bunlardan herhangi birinde eğer false varsa memory koruması yok demektir.

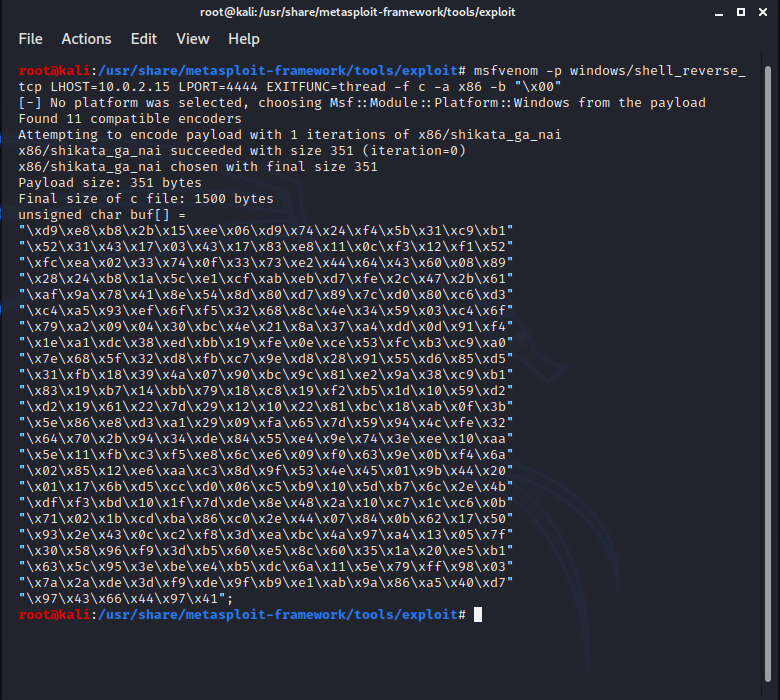
Elde ettiğimiz hex kodunu burada vulnserver için kullanıyoruz.

**!mona find -s “\xff\xe4” -m essfunc.dll** komutunu yazarak aradığımız 625011af adresine ulaşıyoruz artık son adım için hazırız.

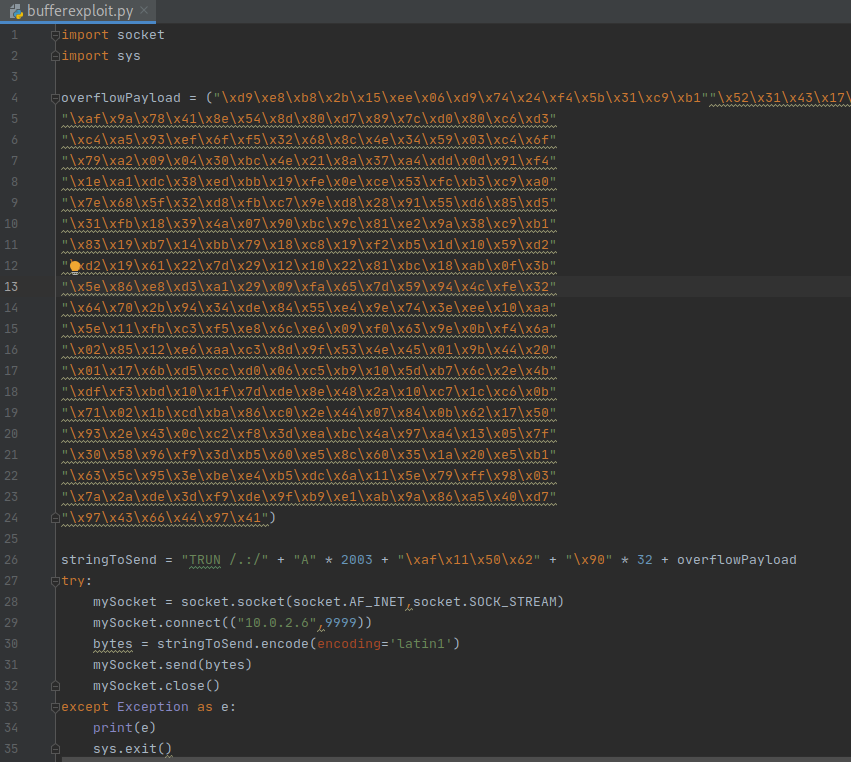


**11.Son Saldırı**

Msfvenom ile bir dinleme yapıyoruz.



Msfvenom ile bir shellcode oluşturuyoruz ve bunu code parçamıza ekliyoruz.



Son olarak shellcode da ekledğimize göre dinlemeye başlayabiliriz.

Bu adımdan sonra Windows makinesine tam anlamıyla sızmış bulunmaktayız whoami gibi istediğimiz kodları çalıştırabiliriz.

