ÖN BİLGİ

Bu belge

- https://www.bgasecurity.com/makale/sizma-testlerinde-parola-kirma-saldirilari/

resmi adresindeki veya linkin kırık olması ihtimaline karşın alternatif olarak

- https://www.includekarabuk.com/kitaplik/indirmeDeposu/Siber_Guvenlik_Teknik_Makaleler/Teori/BaskalarinaAitMakaleler/S%C4%B1zma%20Testlerinde%20Brute%20Force.pdf

adresindeki makaleye çalışılarak elde edilen notlarımı kapsamaktadır. Bu çıkarılan notlar belgemde alıntılar ve/veya kişisel ilavelerim mevcuttur.

1)

Katmanlı güvenlik mimarisi kullanılıyorsa tek bir parola ile yetinilmez ve parolayı ele geçiren saldırganın aşması gereken başka engeller karşısına gelir.

(Page 3)

2)

Parola kırma işlemi yetki yükseltmek için veya girilen sistemde kalıcı olmak için kullanılabilir.

(Page 3)

3)

Encryption vs Encoding

Verinin sadece gizli anahtarı bilen kişilerce okunabilmesini sağlayan format değişikliğine encryption (şifreleme) denir. Verinin farklı sistem ve ortamlarda dolaşabilmesi için bir formattan başka bir formata dönüştürülmesi işlemine ise encoding (kodlama) denir. Şifrelemenin kodlamadan temel farkı şifrelemenin sadece ilgili anahtarı bilen kişilerce çözülebilmesi, yani sadece ilgili anahtarı bilen kişilerce şifreli verinin orijinaline çevrilebilmesidir. Kodlamada ise herkes kodlanan veriyi decode edebilir. Şifrelemelere örnek olarak en sık kullanılan RSA ve AES algoritmaları verilebilir. Kodlamaya örnek olarak ise Base64 algoritması verilebilir.

NOT: Hash'ler tek yönlü oldukları için onlara kodlama denemez. Çünkü kodlamada veri bir formattan bir başka formata dönüştürülebildiği gibi aynı zamanda eski formata geri getirebilir (decoding). Halbuki Hash'te veri bir formattan bir başka formata dönüştürülebilirken eski formata geri döndürülemez! Bu mümkün değildir.

(Page 3-4)

4)

Base64 algoritması ile kodlanmış veriyi decode edelim:

> echo QmlsZ2kgR3V2ZW5saWdpIEFLQURFTUlTSQo= | base64 -d

Output:

Bilgi Guvenligi AKADEMISI

Base64 algoritması ile kodlanmamış veriyi encode edelim:

> echo "Bilgi Guvenligi AKADEMISI" | base64

Output:

QmlsZ2kgR3V2ZW5saWdpIEFLQURFTUlTSQo=

(Page 4)

4)

Parolaları ele geçirmek için kullanılan dört farklı method vardır. Bunlar:

- Pasif Çevrimiçi Saldırılar (Sniffing)
- Aktif Çevrimiçi Saldırılar (Brute Force & Dictionary Attack)
- Çevrimdışı Saldırılar (Cracking Hashes)
- Teknik Olmayan Saldırılar (Social Engineering || Keylogger || Kurbanı Fiziksel Takip)

(Page 4)

5)

Pasif çevrimiçi saldırılarda (sniffing'te) kullanılabilecek araçlara örnek olarak tcpdump, dsniff, ettercap, Cain & Abel, karma, sslstrip verilebilir.

Aktif çevrimiçi saldırılarda (Brute Force'ta) kullanılabilecek araçlara örnek olarak Hydra, Medusa, Fgdump, ncrack ve Cain & Abel verilebilir.

Çevrimdışı saldırılarda kullanılabilecek araçlara örnek olarak John the Ripper, Hashcat, RainbowCrack ve Opcrack verilebilir.

Teknik olmayan saldırılarda kullanılabilecek araçlara örnek olarak ...

(Page [GENEL])

6)

Tcpdump ve Dsniff

Bir kurbanın trafiğini ARP Poisoning ya da bir başka Man in the Middle atağı ile üzerimizden geçirdiğimizi varsayalım. Bu durumda kendi Ethernet kartımızı dinlediğimizde kurbanın trafiğini dinlemiş oluruz.

> tcpdump -i eth0 -tn port 21 -X

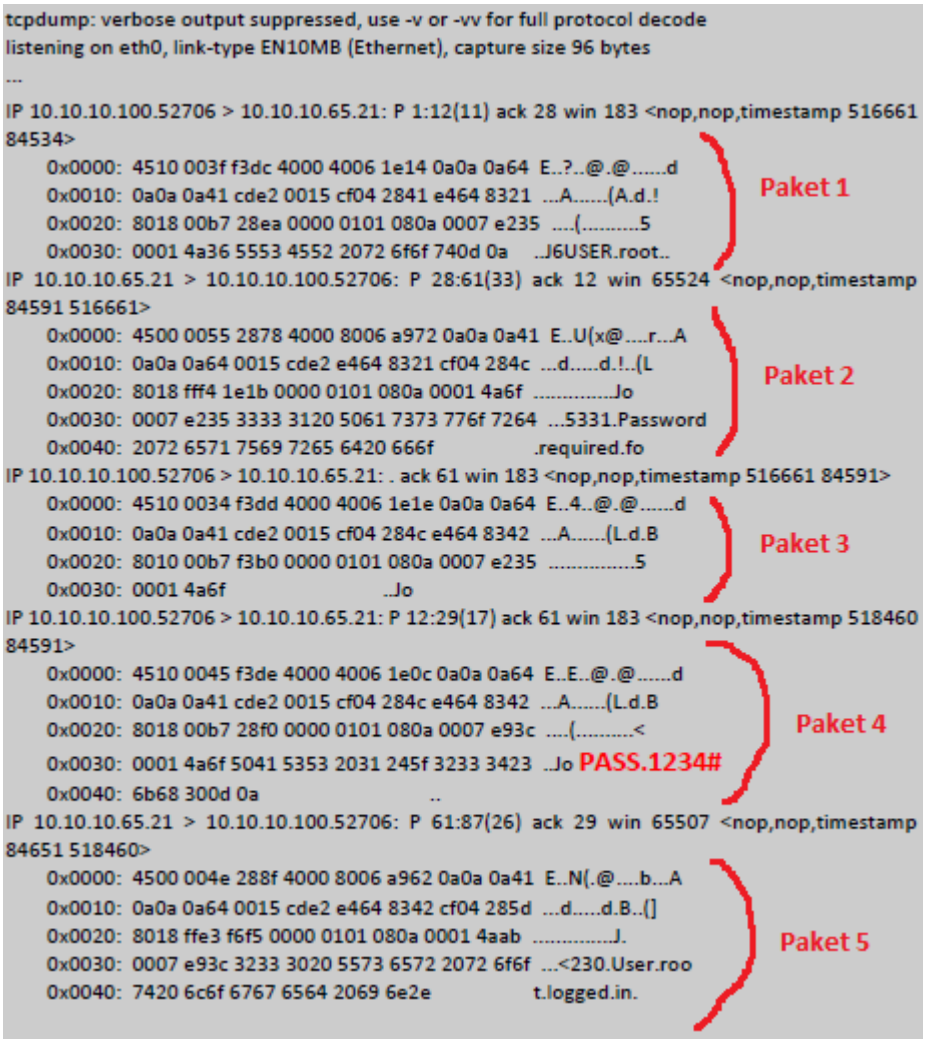
-i : dinlenilecek arayüzü ister.

-t : dinleme esnasında ekrana timestamp'leri yazdırmamayı sağlar.

-n : dinleme esnasında ekrana dökülen bilgilerden IP adreslerini domain'e dönüştürmemeyi sağlar.

-X : dinleme esnasında paketlerin içerdiği verinin de (hex olarak) ekrana yansıtılmasını sağlar.

Output:



Görüldüğü gibi Paket 4'te bir parola manuel olarak tespit edilebilir. Bu tespit işini bizim yerimize tool'lara da yaptırabiliriz. Mesela dsniff gibi... Yine kurbanın trafiğini ARP Poisoning ya da bir başka Man in the Middle atağı ile üzerimizden geçiriyor olalım. Bu durumda dsniff bir parola yakalar yakalamaz ekrana basar:

> dsniff -n

Output:

```
11/23/10 03:26:59 tcp 10.10.10.100.34040 -> 10.10.10.65.21 (ftp)
USER root
PASS 1234#
```

(Page 6)

7)

Çevrimdışı saldırılarda kullanıcı adları ve parolaların sistemde kayıtlı tutulduğu noktalara erişim sağlanmaya çalışılır. Kullanıcı adları ve parolaları text olarak veya şifrelenmiş (encrypted) olarak veya özetlenmiş (hashing) şekilde tutuluyor olabilir.

(Page 9)

8)

LM Hash

Eski Windows'larda kullanılan bir hash (özetleme) algoritması olan LAN Manager (LANMAN) ya da diğer adıyla LM tamamen savunmasız bir algoritmadır. Bu algoritmaya göre kullanıcının tanımladığı şifre 15 karakterden küçükse 14 karaktere tamamlanır. Eğer tanımlanan şifrede küçük harf kullanılmışsa büyük yapılır. Çıkan 14 karakterli özet 7'şerli iki kısma ayrılır ve DES anahtarı ile şifrelenir. Bu özet Windows'un SAM adlı dosyasında depolanır. Burada değinilmesi gereken nokta parolanın iki kısma ayrılması ve ayrı ayrı şifrelenmesi işlemi aslında parolanın kırılmasını kolaylaştıran bir etkendir. Çünkü 14 karakterli bir parolayı kırmaktansa 7 karakterli iki parolayı kırmak daha kolaydır. Yeni Windows sürümlerinde LM varsayılan olarak kapalıdır.

NTLM Hash

NTLM özetlemesi LM'ye göre daha güvenilirdir. Çünkü parola uzunluğu olarak 256 karaktere kadar destek verir. Hash'leme algoritması olarak MD4 kullanır. Küçük harfleri büyütme gibi bir işlem yapmaz ve parolayı ikiye bölmez. NTLM LM'ye göre daha güvenilir, fakat her ikisi de salt kullanmazlar.

NOT: Salt her parola için rastgele üretilen ve parolaya eklenen bir veridir.

(Page 10)

10)

Windows Sunucu Sistemlerinde Kullanıcı Yetkilendirme

Kullanıcı sisteme girmek istediğinde kullanıcı adı ve parolası bazı yöntemlerle sunucuya gönderilir. Bu işlem eski Windows'larda NTLM Challenge/Response ile yapılırken yeni nesil Windows'larda Kerberos ile yapılmaktadır.

NTLM Challenge/Response

NTLM Challenge/Response yönteminde kullanıcı sisteme girmek istediğinde sunucu kullanıcıya kimlik bilgilerini sorar. Kullanıcının kullanıcı adı sunucuya ağ üzerinden gider, fakat kullanıcının parolası ağ üzerinden gitmez. Bunun yerine kullanıcı parolasını "hash'leyip" sunucuya ağ üzerinden gönderir. Ardından sunucu 16 bitlik bir challenge gönderir. Kullanıcı bu challenge'ı alıp kendi parolasını hash'leyen algoritma ile hash'ler ve Response olarak sunucuya gönderir. Sunucuda böylece kullanıcının kullanıcı adı, parolasının hash'i ve parolasını hash'leyen algoritma ile hash'lenmiş challenge vardır. Sunucu aldığı kullanıcı adı ile kendi içindeki SAM dosyasında ilgili parola hash'ine bakar. Kendi SAM'indeki parolayı hash'leyen algoritma ile kullanıcıya gönderdiği challenge'ı hash'ler ve kullanıcıdan aldığı hash'li challenge'ı bununla kıyaslar. Challenge'lar aynı çıkarsa bu durumda sunucu kullanıcıyı yetkilendirir. Yani kullanıcı oturum açmış olur.

Kerberos

Kerberos yeni nesil Windows'larda kullanıcı yetkilendirme için kullanıldığı gibi Apple ve açık kaynak kodlu sistemlerde de kullanılır. Kerberos protokolünün tasarımcıları ilk başta istemci-sunucu modelini hedef almışlardır ve bu doğrultuda hem kullanıcının hem de sunucunun birbirlerinin kimliklerini doğrulamasını sağlayan karşılıklı kimlik doğrulama özelliğini sunmuşlardır. Kerberos sadece iki uç tarafından bilinen simetrik bir anahtar kullanarak bağlantı kurmaya izin veren bir protokoldür. Güvenlidir. İzinsiz dinlemelere karşı dayanıklıdır. Kerberos servisi varsayılan port olarak UDP 88.portu kullanır.

(Page 10-11)

11)

Linux sistemlerde kullanıcı hesap bilgileri /etc/passwd dosyasında saklanırken kullanıcı adı/parola bilgileri hash'lenmiş bir şekilde /etc/shadow dosyasında saklanır. /etc/shadow dosyasındaki bir satır örneğine bakacak olursak

root:1Er.GPi0V$7M9NtL.yrh7U1YER2T7/c/.:10063:0:99999:7:::

: sembolü satırı kısımlara ayırmıştır. O kısımlar ve anlamları ise şu şekildedir:

1. İlk kısım kullanıcı adını içerir.
2. İkinci kısım parolanın özet (hash) halini içerir.
3. Üçüncü kısım parolanın en son değiştirildiği tarihi ile 1 Ocak 1970 tarihi arasındaki gün sayısını içerir.
4. Dördüncü kısım parola değişikliği için gereken minimum gün sayısını içerir.
5. Beşinci kısım kullanıcının parola değişikliği yapması için maksimum gün sayısını içerir.
6. Altıncı kısım kullanıcının parola değişikliği yapması için kalan gün sayısını içerir.
7. Yedinci kısım parolanın (hesabın) geçersiz olduğu günden beri geçen gün sayısını içerir (Henüz parola geçersiz olmadığı için yukarıdaki satırda o bölüm boştur).
8. Sekizinci kısım parolanın (hesabın) geçersiz olduğu gün ile 1 Ocak 1970 tarihi arasındaki gün sayısını içerir (Bu bölüm de yukarıdaki satırda boştur).

root:1Er.GPi0V$7M9NtL.yrh7U1YER2T7/c/.:10063:0:99999:7:::

Parolanın hash'lendiği kısımda yer alan hash'in ilk iki dolar işareti arasında bulunan sayı bize parola için hangi özet alma algoritmasının kullanıldığını belirtir. Yukarıdaki örnekte 1 ile MD5 algoritması kastedilmektedir. MD5 gibi başka algoritmalar da kullanılabilirdi. Mesela;

$2y$ - Blowfish

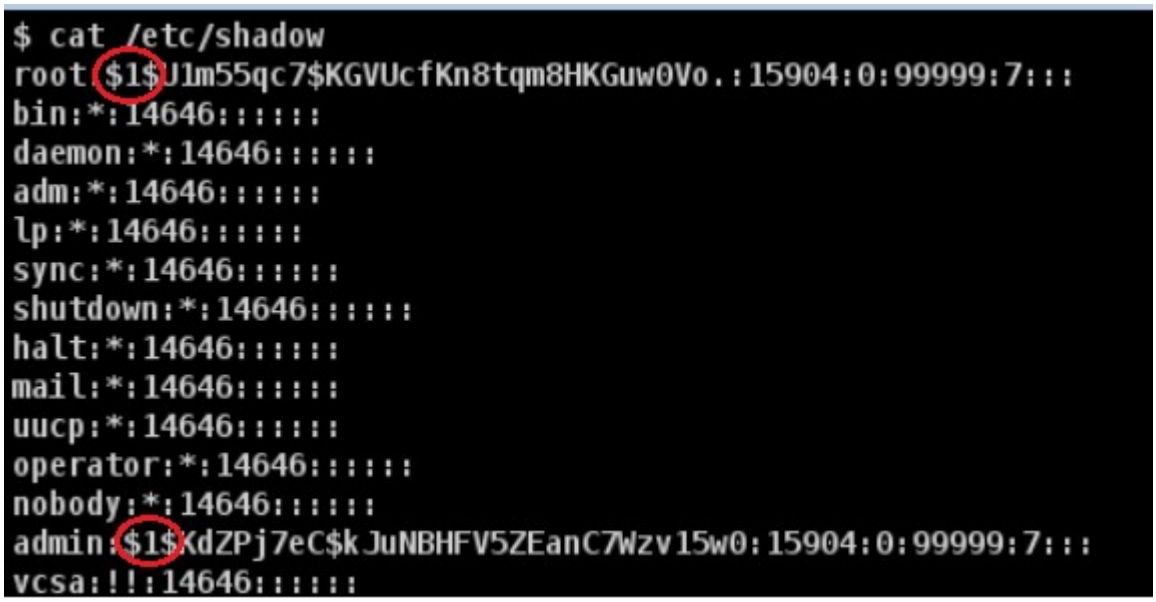
3 - MD4

5 - SHA 256

6 - SHA 512

12)

Varsayalım ki linux bir sisteme sızdık ve cat /etc/shadow komutu ile hedef sistemin hesaplarının parola bilgilerine aşağıdaki gibi ulaşıldı:



Resimde dikkat ettiyseniz root ve admin kullanıcısı 1 içermektedir. 1 sayısı root ve admin parolalarının MD5 algoritması ile özetlendiğini ifade eder. Bu özetler kopyalanıp John the ripper ile ya da John the Ripper'ın görsel arayüzü olan Johnny ile kolaylıkla kırılabilir.

(Page 15)

13)

JTR'da kullanılan parola kırma modları şunlardır:

- Single Crack *(Kendindeki sözlükle kırma işlemi)*
- Wordlist *(Kullanıcının sözlüğüyle kırma işlemi)*
- Kurallı Sözlük Kullanımı
- Incremental

(Page 17)

14)

JTR'da en önemli özelliklerden biri parola kırma işlemine başlamadan önce çeşitli kuralların belirlenebilmesidir. Mesela hedef parolanın 8 karakter uzunluğunda olduğu biliniyorsa JTR'ye parolanın 8 karakterden oluştu kuralı konabilir ve böylelikle kırma işlemi için zaman kazanılmış olur. JTR'a verilebilecek diğer kurallar şunlardır:

- Maximum parola uzunluğu
- Minimum parola uzunluğu
- Özel karakterlerin kullanılıp kullanılmayacağı bilgisi
- Hangi özel karakterlerin kullanılacağı bilgisi
- Alfanumeric değerlerin kullanılıp kullanılmayacağı bilgisi

Kurallar john.conf dosyasına yazılmaktadır. Fakat kural dosyasına kuralları yazmak hiç de kolay olmadığı için Matt Weir tarafından yazılan John the Ripper Config File Generator (http://reusablesec.googlepages.com/) adlı araç kullanılabilir.

(Page 20)

15)

JTR kullanırken “No hash loaded” hatası alınırsa bunun iki nedenden dolayı kaynaklanır:

a. JTR hash tipini tanımlayamamıştır.

b. Daha önce kırmak için üzerinde uğraştığı ve kırdığı bir hash'tir.

İkinci seçenekden dolayı kaynaklanıyorsa john.pot dosyası silinmelidir.

(Page 20-21)

16)

John the Ripper'ın eski versiyonları SHA512 algoritması ile hash'lenmiş parolaları kırmayı desteklemediği için hash'i tanımlayamaz ve aşağıdaki hatayı verir:

No password hashes loaded.

(Page 23)

17)

Kendi hash'imizi openssl ile kendimiz oluşturabiliriz. Aşağıda salt değeri olarak salata'ya sahip deneme parolasının hash'i oluşturulmaktadır:

> openssl passwd -1 -salt salata deneme

Output:

1salata$D2pdYRA7H6IjskEj4Qtue1

(page 27)

18)

Wordlist vs. Rainbowtable

JTR'de çok büyük mb'lı sözlük kullanarak hash kırma işlemi biraz uzun sürebilir. Çünkü sözlükteki her bir kelimenin (satırın) hash'i oluşturuluyor ve o şekilde kırılacak parolanın hash'iyle kıyaslanıyor. Halbuki rainbowtable'da hazır hash'ler yer aldığından bir hash'e dönüştürme işlemi söz konusu değildir. Sadece kıyaslama söz konusudur. Bu da çok büyük oranda performans sağlar.

NOT: Eğer rainbowtable'daki bir hash'le parolanın hash'i eşleşirse rainbowtable'daki hash'in karşılığı olarak verilmiş kelime parolanın ta kendisidir.

(Page 27)

19)

Hashcat

Resmi sitesinden *hashcat-2.00.7z* dosyasını indir. Çalıştırmak için;

> cd hashcat-2.00

> ./hashcat-cli64.bin

Hashcat'in Kullanımı:

./hascat-cli64.bin [options] hashfile [mask|wordfiles|directories]

Kullanım Örneği:

> ./hashcat-cli64.bin -m 0 -a 0 hash.txt rockyou.txt

NOT: rockyou.txt sözlüğü /home/hefese/hashcat-2.00 dizininde mevcuttur.

rockyou.txt adlı sözlük yardımıyla hash.txt'teki hash kırılmaya çalışılır. -m parametresi hash type'ı belirtir. 0 rakamı MD5 algoritmasını kasteder. Bu ve diğer algoritmaların numaralarına

> ./hashcat-cli64.bin --help

diyerek ulaşabilirsin. -a parametresi ise attack mode'unu ifade etmeye yarar. 0 rakamı ile Straight (basit) modda kırma işlemi yap emri verilmiş olur (Basit modda sözlükteki tüm kelimeler hash ile karşılaştırılır).

Bazı options'lar ve açıklamaları:

--hash-mode=NUM veya -m parametresi *(Kırılacak hash'in tipini alır)*

-m 0 MD5 algoritması ile hash'i kırmaya çalış denmiş olur

-m 1700 SHA512 algoritması ile hash'i kırmaya çalış denmiş olur.

…

Tüm algoritmaların numaralarına

> ./hashcat-cli64.bin –help

ile ulaşabilirsin. Hashcat varsayılan olarak MD5 algoritması ile hash'leri kırmaya çalışır.

--attack-mode=NUM veya -a parametresi *(Hash'i kırmak için kullanılacak method)*

-a 0 Straight

Sözlükteki tüm kelimeler parola hash'iyle karşılaştırılır.

-a 1 Combination

Sözlükteki kelimelerin birbirleriyle kombinasyonları alınarak karşılaştırma yapılır. Mesela test ve hasan kelimelerini birleştirir ve hash'ini alıp parolanın hash'iyle karşılaştırır.

-a 2 Toggle-Case

Her bir satırın büyük küçük harf dönüşümleriyle kombinasyonları parola hash'iyle karşılaştırılır (Örn; test, Test, tEst, teSt,...)

-a 3 Brute-Force

Belirtilen koşullara (karakter setine, parola uzunluğuna vs...) uyan her türlü kombinasyon denenir.

-a 4 Permutation

Kelimelerden harfleri alır ve permütasyonlarını dener. Örneğin sözlüğün bir satırındaki kelime abc olsun. Permütasyonları abc, acb, bca, bac, cab, cba. (--perm-min=SAYI ve --perm-max=SAYI ile permütasyona sokulacak kelimelerin harf sayısı belirtilebilir).

-a 5 Table-Lookup

Kelimeyi harflerine ayırır ve --table-file ile belirtilen dönüşümleri harflere uygular. Mesela test kelimesi t3St olabilir. Bu yeni kombinasyonları parola hash'iyle karşılaştırarak parolayı kırmaya çalışır. Bunun bir örneğine birazdan değinilecektir.

(Page 28-30)

20)

Table-Lookup Modunda Hash Kırma Örneği

İlk olarak kendimize ait bir table oluşturalım (Dilenilirse hashcat'in mevcut table dosyaları da kullanılabilir):

deneme.table

```
t=t
t=T
t=7
e=e
e=E
e=3
s=s
s=S
s=5
```

Bu table dosyamızı hascat-2.00/tables/ dizinine taşıyalım. Ardından hash dosyamıza MD5 algoritmasından geçmiş beş tane parola koyalım.

hash2.txt

```
751ec45015a704a39dc403001c963e97
e86e107b113b0f830b9b817b4a9addb8
f61954c634231f8bbf0264d9797df9fa
a0c58991fd9a340393d6a021bc490540
cc03e747a6afbbcbf8be7668acfebee5
05a671c66aefea124cc08b76ea6d30bb
```

Yukarıdaki her bir satır bir parola anlamına gelmektedir. Şimdi bu hash'leri oluşturduğumuz table dosyasının yourock.txt sözlüğüne katacağı ekstra kombinasyonlarıyla beraber kırmaya çalışalım:

Terminal:

> ./hashcat-cli64.bin -m 0 -a 5 --table-file=tables/deneme.table hash2.txt rockyou.txt

Output:

cc03e747a6afbbcbf8be7668acfebee5:test123

05a671c66aefea124cc08b76ea6d30bb:testtest

e86e107b113b0f830b9b817b4a9addb8:t3st

a0c58991fd9a340393d6a021bc490540:73s7

f61954c634231f8bbf0264d9797df9fa:TEsT

Görüldüğü üzere rockyou.txt sözlüğünde olmayan t3st, 73s7 gibi kelimeler table sayesinde oluşturuldu ve hash'lerle kıyaslandı. Sonuç olarak hash'ler bu kelimelerle kırılmıştır.

(page 30-32)

21)

Rainbow tabloları rtgen, ophcrack, SMB Hash Generator gibi araçlarla oluşturulabilir. Veyahut internetten hazır rainbow tabloları indirilebilir.

(Page 34)

22)

RainbowCrack Tool'u

a. rtgen

Kali'de yüklü RainbowCrack'in bir alt bileşeni olan rtgen ile belirlenen kriterlere uygun rainbow tablosu oluşturulur. rtgen komutu ismini **r**ainbow **t**able **gen**erating'den almaktadır.

Usage:

> rtgen hashType [loweralpha | loweralpha-numeric | numeric | mixalpha-numeric| alpha-numeric] minLength maxLength tableIndex chainLen chainNum partIndex

NOT: Zincir ile kastedilen şey bir parolanın hash'i alındıktan sonra başından ve sonundan birkaç karakter alıp, gerisini atıp kalanların bir daha hash'ini almaya denir. Böylelikle bir hash zinciri oluşur.

Example:

> rtgen md5 loweralpha 1 5 0 10000 9682 0

Yukarıdaki kodda belirtilen parametrelere göre alfabedeki tüm küçük harflerle minimum 1 karakterli, maksimum 5 karakterli tüm kombinasyonların md5 algoritması ile hash'lerinin yer alacağı bir rainbow table oluşturulur.

NOT: Kullanılacak hash algoritması olarak md5 yerine lm de kullanılabilir (lm = LM)

b. rtsort

rtgen ile tablo oluşturma sonlandığında çıktıda tablonun ismi görünecektir. Mesela md5_loweralpha#1-5_0_10000x9682_0.rt şeklinde. Oluşturulan bu tablo rcrack ile kullanılmadan önce rtsort ile sıralanmaya tabi tutulmalıdır.

Usage:

> rtsort rtFiles [rtFiles ...]   // rtgen'in oluşturduğu rainbow tablo dosyaları

Example:

> rtsort md5_loweralpha#1-5_0_10000x9682_0.rt

c. rcrack

rcrack komutu kırılması gereken parolanın oluşturulan rainbow tablosu ile kırılmasını sağlar.

Usage:

rcrack rainbowTable.rt -h hashValue

Example:

> rcrack md5_loweralpha#1-5_0_10000x9682_0.rt -h fc3f318fba8b3c1502bece62a27712df

Output:

…

…

result

---

fc3f318fba8b3c1502bece62a27712df **hasan** hex:686173616e

NOT: Eğer birden fazla hash bir defada kırılmak isteniyorsa bu durumda hepsi alt alta olacak şekilde bir dosyaya yerleştirilir (e.g. hash.txt) Ardından -l parametresi ile dosya rcrack komutuna dahil edilir.

Uyarı: Çoklu hash kırma yöntemi normalde var olan bi'şey, fakat pratikte çalışmadı. Dosya açılamadı hatası verdi.

hash.txt :

fc3f318fba8b3c1502bece62a27712df // hasan

87e9f0f7ed20bdf96ba715980e2aaa2b // nasah

b4b643cb1547b52057fdd15336581ca8 // fatih

Terminal:

> rcrack md5_loweralpha#1-5_0_10000x9682_0.rt -l hash.txt

23)

Ophcrack

Rainbow tablolarını kullanarak parola kırmaya yarayan bir programdır. Sadece LM ve NTLM hash'lerini kırabilir. Ophcrack kali'de yüklü olarak gelmektedir.

24)

Parola Kırmada CPU Yerine GPU Kullanmak

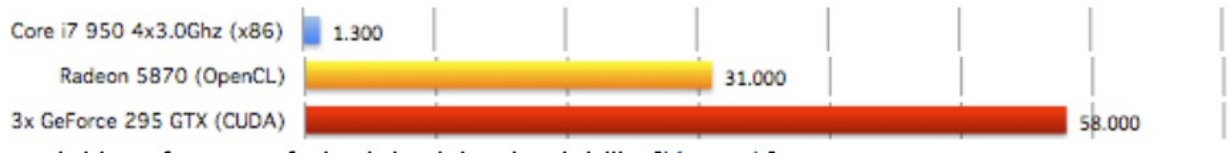
Parola kırmak epey bellek ve vakit isteyen bir uğraştır. Parola kırmak için gerekli bellek sorunu harici HDD'ler ile çözülebilirken hız sorunu ise GPU kullanımı ile çözülebilmektedir. GPU fiziksel olarak ve çalışma mantığı olarak CPU'dan farklıdır. İşlemciler ayrı ayrı işlem yapabilen çekirdeklere sahiptirler. Bu sayede mesela 2 çekirdekli bir işlemci ile aynı anda iki farklı işlem yapılabilir. CPU'larda çekirdek sayısı olarak 2, 4 ve 8 yaygınken GPU'larda ise bu rakamlar yüzler seviyesinde seyretmektedir. CPU'nun GPU ile fiziksel farkı budur. Çalışma mantığı olarak farkına gelecek olursak CPU'da “merkezi işlem” konsepti varken GPU'da “birlikte işlem” konsepti vardır. NVIDIA, AMD gibi firmaların paralel işlem yapmak için çıkarttıkları CUDO, OpenCL, DirectCompute gibi modüller sayesinde yüksek işlem gücü gerektiren işler için GPU kullanılabilmektedir.

NOT: C, C++, Fortran gibi dillerle CUDA mimarisi kullanılabilmektedir.

(Page 41)

25)

Parola kırmada oclHashcat, Pyrit gibi araçlar GPU'nun gücünü kullanarak Brute Force gibi tüm kombinasyonların denendiği zaman alıcı process'leri makul sürelere indirgeyebilmektedirler. Aşağıda Pyrit aracının i7 işlemcisi üzerinde, OpenCL modülünün ATI Radeon üzerinde ve Cudo modülünün NVIDIA üzerinde WPA/WPA2 key'lerini kırmak için saniyede denediği key sayılarının grafiksel gösterimini görmektesiniz.



Görüldüğü üzere i7 işlemcisi üzerinde saniyede yapılan deneme sayısı nerdee.... GPU'ların ki nerdee...

(Page 40-41)

26)

Teknik Olmayan Saldırılar

Bu saldırı tipinde

- Fiziksel olarak kullanıcıya ulaşmak ve takip etmek
- keylogger tarzı yazılımlar kullanmak
- Sosyal Mühendislik

gibi saldırı teknikleri kullanılır.

(Page 42)

27)

Psexec Microsoft'un Telnet'e alternatif olarak sunduğu uzak sistemlerde kod çalıştırmayı sağlayan bir tool'dur.

(Page 48)